PLANNING, CONTROL AND MANAGEMENT OF MULTICELLULAR MANUFACTURING SYSTEMS BY PRODUCTION AUTHORIZATION CARDS (PAC) SYSTEM

by

KRYSTYNA BIELUNSKA - PERLIKOWSKI

A Thesis Submitted to the
Faculty of Engineering
in Partial Fulfillment of the Requirements
for the Degree of

DOCTOR OF PHILOSOPHY

Major Subject: Industrial Engineering

APPROVED:

[redacted]

Dr. Eldon A. Gunn (Supervisor)

[redacted]

Dr. Benoit Montreuil (External)

[redacted]

Dr. Carl-Louis Sandblom

[redacted]

Dr. Tao Yang [redacted]

Dr. Harvey H. Millar

TECHNICAL UNIVERSITY OF NOVA SCOTIA

Halifax, Nova Scotia, Canada 1997

0-612-31517-7

TECHNICAL UNIVERSITY OF NOVA SCOTIA LIBRARY

"AUTHORITY TO DISTRIBUTE MANUSCRIPT THESIS"

TITLE

PLANNING, CONTROL AND MANAGEMENT OF

MULTICELLULAR MANUFACTURING SYSTEMS BY

PRODUCTION AUTHORIZATION CARDS (PAC) SYSTEM

The above library may make available or authorize another library to make available individual photo/microfilm copies of this thesis without restrictions.

Full Name of Author **KRYSTYNA BIELUNSKA - PERLIKOWSKI**

Signature of Author [redacted]

Date August 6, 1997

to my children Emilia and Daniel

TABLE OF CONTENTS

Page

Page

Page

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS AND SYMBOLS USED

ABBREVIATIONS

BMAP	batch Markovian arrival process
BOM	bill of material
BSS	base stock system
CONWIP	constant work-in-process
CORD	queue to calculate delay of delivering product type *f* or *fa* to customer, calculated from the time of arrival of an order tag
CPM	critical path method
CPU	computer processing unit
CUST	queue to calculate delay of delivering product type *f* or *fa* to customer, calculated from the time of arrival of a requisition tag
DCI	delay costing option I
DCII	delay costing option II
FE	function evaluation
FIFO	first-in first-out
IC	integral control
IID	independent and identically distributed (random variables)
JIT	just-in-time
Kanban	Japanese card system
LC	local control
MARRVT	mean of Poisson arrival process (=mean interarrival time of customer orders)
MPS	master production schedule
MRP	material requirements planning

MRPII	manufacturing resource planning
MSERVT	mean processing time
opi	optimization algorithm used for IC and CONWIP
opk	optimization algorithm used for Kanban
opt	optimization algorithm used for PAC, BSS, MRP, PTO-$\tau \geq 0$ and LC
OPT	optimized production technology
ORD	queue for order tags in stores
PA	production authorization (card)
pac	optimization algorithm used for PTO; it is a single simulation run as the PAC parameters are set to fixed values
PAC	production authorization cards or queue for PA cards
PACOPT	PAC optimization algorithm
PACRAN	random optimization algorithm
$PPRD_j$	queue for parts in storage per product type j
PROC	queue for process tags in stores
PROD	queue for parts in stores
PTO	produce-to-order
$PWIP_{ij}$	queue for parts in a work-in-process storage at cell i per product type j
REQ	queue for requisition tags in stores
rop	random optimization algorithm
ROP	reorder point system
SIMLIB	set of FORTRAN support routines
SAMPST	sample statistics (discrete-time data, as delays in queues)
TIMEST	time statistics (continuous-time data)
WIP	work-in-process storage

SYMBOLS

a	assembly type of product
b	processing batch (no. of items in BOM)
c	number of identical servers (machines)
cc_j	customer delay costs per product j
cp_j	inventory costs in product stores per product j
cw_j	inventory costs in cells per product j
d	downstream
f	final type of product
fa	final/assembly type of product
m	number of cells
M	big number, equivalent to unlimited number of process tags
n	number of products ($n=f+fa+a$)
k	number of process tags (PAC parameter)
p_j	probability that arriving customer demand is for product j
r	PA cards batch size (PAC parameter) or number of different raw materials
S	processing time
t_{ij}	time product j requires for transportation from storage location to cell i.
u	upstream
z	initial/terminal inventory level at the storage location (PAC parameter)
λ	arrival rate
μ	mean
$1/\mu_j$	mean processing time of product j
τ	time lag between receiving an order tag and a requisition tag at the storage location (PAC parameter)

ACKNOWLEDGMENTS

First I wish to thank Dr. Eldon Gunn for being a supervisor who stimulated me through his comments and constructive suggestions as well as his wide interest in the field of study. Dr. Gunn attracted me to the PAC concept and encouraged me in every stage of the research and preparation of the thesis. I am also very grateful to him for all the help and support in preparation of various publications and making it possible for me to participate in professional conferences. I would like also to express my gratitude to Dr. Gunn for allowing me to conduct my computations on the RISC/6000 computer and enable me to use the Internet facility.

I wish to thank Dr. John Buzacott for his comments and suggestions especially in the first stage of the research. I am very grateful to Dr. Wanda Rosa-Hatko for her moral support during my first year at TUNS and also for financial support, which made it possible for me to attend the CORS conference in Montreal in 1994. I also wish to thank Dr. Qi-Ming He for helping me to analyze the BMAP/G/1 queues. I thank my guiding committee, Dr. E. A. Gunn, Dr. C.-L. Sandblom, Dr. T. Yang and Dr. H. H. Millar for their support during the course of research and for giving of their time in evaluating the thesis. I also thank Prof. Peter Wilson, Dr. Biman Das, Dr. J. Pemberton Cyrus, Dr. Ronald Pelot, Prof. Ken Sadler, Mrs. Gail Veinot, Mrs. Cindi Hardiman and Mrs. Linda Seamone for all their support, encouragement and help during my studies.

My research was funded by the Natural Sciences and Engineering Research Council of Canada in the form of Postgraduate Scholarship PGS B, by the Dorothy Rosetti Engineering Research Scholarship from the Technical University of Nova Scotia, through Dr. Gunn's NSERC research grant and through part-time employment by TUNS and Atlantic Industrial Research Institute.

ABSTRACT

The object of this research is to study the issue of proper coordination in a multicellular manufacturing system. In 1992, Buzacott and Shanthikumar introduced the Production Authorization Cards (PAC) system, a new coordination scheme for material and information flow control in multicellular manufacturing systems. They show that most traditional material and information flow control mechanisms can be seen as special instances of this scheme. They highlighted the need to solve the problem of optimal parameter values for a given flow control problem, but left the solution of this problem open. This task requires two steps, the development of a performance evaluation model and the development of an optimization approach. In this thesis we describe both a simulation model and optimization method for the PAC control and show their application in a number of settings.

The thesis first describes and reviews the literature for common manufacturing control policies and then focuses on the description of the PAC concept. Since the PAC system forms a basis for the research, a complete presentation is essential for understanding of the research issues.

The development of a simulation model of a multi-cell multi-product manufacturing system coordinated by Production Authorization Cards was an essential part of this research. This model allows for the performance evaluation of the PAC controlled manufacturing system. It permits the analysis of systems with assembly aspects, batching and complex routing. Execution speed and the ability to be called as a subroutine in an overall optimization are key features.

The PAC optimization algorithm uses the simulation model for the evaluation of the cost function to be optimized. The optimization algorithm is based on hill climbing methods similar to Hooke and Jeeves. It is a search consisting of a sequence of

exploration steps about a base set of PAC parameters, which, if successful, are followed by pattern moves. The PAC optimization model can, at best, be guaranteed to produce local optima. A second algorithm, based on random search principles, has been developed to verify how well the first method performs.

The use of the simulation model and the PAC optimization algorithm are illustrated with 5 different examples of manufacturing configurations, ranging from a very simple one to a quite complex one. The optimization algorithm is quite feasible for systems with a small number of final and intermediate products (less than 20). Numerical results for nine different coordination policies (PAC-general, PTO, PTO-variant with $\tau \geq 0$, MRP, Kanban, LC, BSS, IC, CONWIP) for all five examples are presented for a variety of cases aimed at exploring the dependence of results on the particular form of arrival time and processing time distributions. The results demonstrate that, for each specific instance of the manufacturing process, there is an optimal coordination scheme, which strongly depends on distributional assumptions. This implies that findings, so often reported in the research literature, based on models with exponential processing times and Poisson arrival of demand, do not directly apply to systems operating on other conditions.

Much of the analysis of PAC systems to date has been based on queueing theory and considers relatively simple models. The approach described here facilitates research on systems of greater complexity, where the assumptions necessary for queueing results are not really appropriate. We feel that our PAC model is a starting point in representing realistic factory problems to study their behavior, and design operating policies.

Effects of deterministic setup times and different cost factors are analyzed in case of few manufacturing layouts. The results related to the BMAP/G/1 queue are compared with the PAC optimization results. Properties of the departure/shipment to customer process of the PAC system are examined for some manufacturing models. Impact of processing times variability on the system performance and optimal parameter values for different control policies, are studied.

Chapter 1

INTRODUCTION

1.1 Objectives

The objective of this research is to study the proper coordination in a multicellular manufacturing system. Buzacott and Shanthikumar (1992a) introduced the Production Authorization Cards (PAC system), a new coordination scheme for material and information flow control in multicellular manufacturing systems. They showed that almost all traditional material and information flow control mechanisms such as MRP (material requirements planning), Kanban (Japanese card system), OPT (optimized production technology), BSS (base stock system), IC (integral control), and CONWIP (constant work-in-process) can be seen as special instances of this scheme.

To obtain the best possible coordination system for a certain set of products. one needs to obtain optimal parameter values, given the data that characterize these products. This requires the development of a performance evaluation model for a given choice of parameters and the development of an optimization model that can find the best choice of parameters. These two steps have been discussed in Buzacott and Shanthikumar (1992b) for single-cell systems and for some special cases of multiple-cell series systems. However, no general model has been developed so far. One contribution of this thesis is to develop such a general performance model and to develop efficient solution procedures to obtain the optimal parameters value for the PAC scheme.

Proper coordination of production at different cells of a multicell manufacturing system is essential in improving customer service, inventory turnover, and productivity. There is as yet no definitive approach to achieve this coordination. In North America, the

MRP system is often used (Wight 1984). In Japan, the Kanban procedure is the most popular (Berkley 1992). Many companies try to find ways to choose between these two, or to find a hybrid system that combines MRP and Kanban, or others. For a given type of a manufacturing system, there should be one production policy which makes the best choice in comparison with others.

This thesis contributes to a better understanding of the main features of the PAC concept itself and demonstrates its feasibility for complex systems. In this work, we show that for each specific instance of the given manufacturing process there is an optimal coordination scheme, which highly depends on distributional assumptions. This implies that results based on exponential processing times and Poisson arrival of demand cannot be applied to systems operating under different conditions. By examination of the properties of the departure/shipment to customer process of the PAC system, we demonstrate quite interesting results with regard to the impact of batching orders for products which are required for processing other products. By carrying out a number of experiments, we study the impact of processing times variability on the system performance and for different coordination schemes. The results indicate that in some cases designing on exponential and deterministic assumptions gives good results, if the higher variability is not encountered.

1.2 Thesis Outline

The thesis first reviews the literature for some common production control policies, and then presents the integrated PAC scheme as a more general approach for control of production process. The reason for this approach is to provide the necessary background for the reader before discussing the overall objective.

The thesis consists of ten chapters and twelve appendices. In what remains of *Chapter 1*, a brief description of the research problem considered in the thesis and the thesis outline are given.

The next three chapters provide required background material. *Chapter 2* contains a brief description of manufacturing systems and production planning issues, including an overview of the main themes found in the manufacturing literature. In *Chapter 3*, we review each of the traditional production coordination policies and give a short characteristic of its primary features. In *Chapter 4*, we describe the main properties of the PAC system. In order to give the reader understanding of the research issues, we need to discuss the PAC system in some detail in this chapter.

The main body of the thesis is contained in chapters 5 to 10. Chapters 5 and 6 give the description of the models and algorithms developed for the PAC systems. In *Chapter 5*, we describe a FORTRAN simulation model for performance evaluation of a manufacturing system coordinated by PAC control. We present the structure of the model using an example manufacturing configuration. We also describe some general modeling issues, with emphasis on the assumptions and simplifications, and indicate where different approaches could be implemented.

In *Chapter 6*, the main features of the optimization algorithm for the PAC controlled manufacturing system are presented. This permits an optimization of PAC parameters for any given coordination policy. The program uses the PAC simulation model as a cost function to be optimized, and is based on a *direct search* technique similar to the Hooke and Jeeves method. In chapter 6, we also present the structure of another optimization algorithm based on random search.

Chapters 7, 8 and 9 present the results of analysis on various issues of the PAC system. In *Chapter 7*, the use of the simulation model and the optimization algorithm for

PAC controlled production system is illustrated in the cases of five different manufacturing configurations (models), ranging from a relatively simple one to a quite complex one. Numerical results for nine different coordination policies for all five models for cases of exponential processing times, uniform processing times and "bulk" arrivals of demand are presented and evaluated. Effects of deterministic setup times and different cost factors are analyzed in the case of a few manufacturing layouts. Some results of optimization by using the "random" optimization algorithm are presented and evaluated. The costs for each optimal policy for "bulk" arrivals of demand are significantly higher than in the case of a constant demand and we try to explain this phenomenon using the BMAP/G/1 results (BMAP = batch Markovian arrival process). In the optimization process, we chose to evaluate the cost function by a single simulation run, which each time uses the same random number stream. The immediate concern here is the robustness of the optimal solution obtained in this fashion. Some numerical results indicating that variation does not significantly affect the final results are presented and discussed.

Buzacott and Shanthikumar identified some properties for the departure or shipment to the customer process of the PAC system for cells in series. *Chapter 8* deals with verification of these properties and checking their validity for more complex manufacturing configurations.

Chapter 9 contains study results of impact of processing times variability on the system performance and optimal parameter values for different control policies. We used shifted exponential and shifted Weibull distributions to model processing times at the machines and to change their coefficient of variations.

Conclusions and suggestion for further research can be found in *Chapter 10*.

Appendices A1 through *A4* are related to the PAC simulation program. They contain a short description of the PAC program with the computer code, definition of variables. example of a simulation output and some validation results for the model.

Appendices B1 through *B3* are related to the PAC optimization algorithm. They contain selective listings of the computer code and examples of the optimization reports.

Appendices C1 through *C5* contain computational results of the PAC optimization algorithm for five different manufacturing configurations formulated in the thesis.

Chapter 2

MANUFACTURING AND PRODUCTION PLANNING ISSUES

2.1 Introduction

In this chapter, we describe manufacturing process and overview the different types of manufacturing system configurations with an emphasis on the cellular manufacturing layout. We also review the literature on manufacturing and production planning issues and classify it according to specific themes.

2.2 Manufacturing Process

Manufacturing systems are complex, dynamic, and stochastic. They consist of a number of subsystems interacting and intercommunicating in an attempt to make the overall system function profitably. Manufacturing facilities create wealth by transforming material into functionally desirable, aesthetically pleasing, environmentally safe, economically affordable, highly reliable, top-quality products. The manufacturing system also provides gainful employment to drive the economy.

A manufacturing system is composed of many resources, such as machines, transportation elements, storage buffers and computers. People are also part of the system. As the resources interact with each other, they create problems of coordination of the production processes. The design and control difficulties stem from the quantity of data to be handled within the system, the uncertainty of an environment where many disturbances may occur, and the structure of the system and the complex relationships between the interacting subsystems that are part of it. To find the best strategy, researchers build mathematical models for the manufacturing processes using analytical and/or simulation approaches. If an analytical solution to the mathematical model is available and computationally efficient, it is often desirable to study the system in this way. Unfortunately most manufacturing facilities are highly complex, so that finding and validating the analytical model without using simulation is virtually impossible. Shanthikumar and Sargent (1983) stimulated the use of hybrid simulation/analytic models by describing four different cases of such models and showing their application.

The large scale development of different modeling tools for production processes is due to the variety of manufacturing systems. Manufacturing systems may be classified by a number of characteristics: volume and diversity of jobs, discrete versus process industry, type of raw materials. The most common configurations include (Askin and Standridge, 1993):

- product or flow lines (assembly lines, transfer lines),
- process or job shops, where production departments are formed by machines with similar capabilities and performing similar functions;
- group technology or cellular manufacturing, which convert what would otherwise be process layout systems to pseudo product layout environments;
- fixed position, which is used for production of very large objects as ships, airplanes.

Each configuration has its strengths and weaknesses (Askin and Standridge, 1993) and requires a different modeling approach. Models form a rational basis for designing new systems and learning about existing systems. Models can be used for system optimization, performance prediction, control, or for gathering insight into the system (Karmarkar, 1994).

In this thesis, we focus mainly on cellular manufacturing. Nevertheless, the results of our study are also applicable to the flow line layouts.

2.3 Cellular Manufacturing

Cellular manufacturing is in place when a company has established one or more manufacturing cells (Montreuil and Lefrancois, 1996). A cell is a collection of machines or processes, located closely together, and dedicated to a set of similar parts or products. Parts are classified into families on the basis of size, geometry, and machining requirements. The relations between cells does not need to be tight. The number of cells, their capability and configuration can be adjusted to the changing production requirements.

There are many different cells applied in industry. Cell classifications are made with regard to the operations performed in the cell, the degree of labor intensity, and the internal flow pattern (Wemmerlov and Hyer, 1987). Cells can perform fabrication, machining, or assembly operations. Sometimes all these types of operations can be combined in one cell.

The grouping of machines in specialized cells is aimed at improving efficiency by the simplification of processes and production of similar parts. Additionally, it has advantages of reducing transportation distances and times between workstations. It also can allow operators to serve multiple machines and work as a team for the production of a complete item rather than performing just a single operation.

2.4 Literature Review

Recent books on the modeling and analysis of manufacturing systems give many algorithms and mathematical tools to deal with manufacturing and production planning issues. Askin and Standridge (1993), show how heuristic and exact algorithms can help manufacturing designers and analysts in their job. The authors consider traditional models found in operations management and also cover recent developments in modeling manufacturing cells and work stations. Gershwin (1994) is largely devoted to studying transfer lines and examines issues that arise in the design and operation of manufacturing systems. Additionally, queueing network models, dynamic programming and linear programming models are applied to flexible and other manufacturing systems. The theme of Buzacott and Shanthikumar (1992b) is that manufacturing systems are fundamentally stochastic systems. Randomness is due not only internally to equipment failures, human errors, and variability in processing times, but also externally to changes and fluctuations in both supply and demand. The authors organize the book according to the various manufacturing system configurations, from the simple, classical ones to more advanced and sophisticated new systems.

The application of different approaches to the modeling and control of manufacturing systems has a voluminous literature. It is impossible to give a complete list of all published works. However, we try to classify the studies according to specific themes in order to overview the main research issues and establish their reference to this thesis.

One possible taxonomy of approaches to manufacturing system modeling, with respect to the models used, is the following:

- tandem queues with blocking: Suri and Diehl (1986), Melamed (1986), Gershwin (1987), Tsiotras, Badr, and Beltrami (1991), Shanthikumar, Yamazaki, and Sakasegawa (1991), Lee and Zipkin (1992), Yamazaki, Sakasegawa, and Shanthikumar (1992), Cheng (1992), Cheng and Zhu (1993), Cheng and Yao (1993), Cheng (1995). Tandem queueing networks are used as models for many types of production lines such as automatic transfer lines and mixed-item assembly lines. The design issues often addressed include optimal allocation of workload and buffer spaces, and optimal ordering of stations. The models usually require exponential processing time distributions and external Poisson arrival process of demand. The main difficulties in analyzing those models, involving two or more queues, is in the requirement of infinite queues. This requirement allows an exact decomposition of the network. However, in reality, storage space is always finite and blocking and starvation may occur.

- Markov or semi-Markov models: Foster and Garcia-Diaz (1983), Altiok and Stidham (1983), Stecke and Solberg (1985), Yao and Buzacott (1987), Davis and Kennedy (1987), Glasserman and Yao (1992a), (1992b), (1992c), (1994a), (1994b), So and Tang (1995). Markov models are mostly used to study transfer lines. The basic idea of a Markov process is that all information about the future behavior of its state exists in the present. Markov systems can be formulated with discrete or continuous time and discrete, continuous or mixed state spaces. Markov models are easy to explain, since each element of a Markov transition matrix can be thought of as the relative frequency with which a transition takes place from one named state to other. The main disadvantage for modeling of production processes by Markov models is in the distributional assumptions required and the difficulty of application to large systems.
- open queueing network models: Shanthikumar and Buzacott (1981), Altiok and Perros (1986). Open queueing network models are used to analyse dynamic job shops. Those models are mostly limited to job shops with exponential processing times and first-in first-out (FIFO) service discipline. Obtaining the analytical solution for those models is very difficult. Many studies are related to approximation procedures for different cases.
- simulation models: Albin (1982), Albin (1984), Altiok and Perros (1986). Simulation models are mostly applied to determine to accuracy of the developed approximation procedures.

- linear and dynamic programming models: Cho, Kim and Kim (1994), Bai and Gershwin (1994). Linear and dynamic programming models are used to develop a real-time control to generate production schedules in a dynamic environment. Since a large number of machines, workers, part types, and operations are involved, hierarchical structures are proposed to reduce the problem size and complexity. However, the models studied are often limited to simple processes, such as two-machine two-part-type systems.

Alternatively, we can consider a taxonomy used on physical layouts and flows:

- assembly lines: Smith and Daskalaki (1988) examine the physical layout and location problems of automated assembly lines. McClain and Moodie (1991) comment on buffer space allocation in automated assembly lines. Simon and Hopp (1995) derive steady-state average throughput and inventories for a system consisting of two input machines and an assembly machine with finite buffers.
- transfer lines: Buzacott (1967) reports the results of a theoretical study of the effects of buffer stocks on automatic transfer lines. Buzacott and Hanifin (1978) evaluate and compare past models of transfer lines. Ohmi (1981) considers the production efficiency of automatic transfer lines where in-process buffers are provided for stochastic station breakdowns. Gershwin and Berman (1981) analyse transfer lines consisting of two unreliable machines with random processing times and finite storage buffers. Gershwin and Schick (1983) determine steady-state probabilities of three-stage transfer lines in order to establish relationships between system parameters and performance measures such as production rate, forced-down times and expected WIP. Ho, Eyler, and Chien (1983) develop a complete and practical method for efficient computation of the sensitivity information of a transfer line throughput. Conway, Maxwell, McClain and Thomas (1988) study the behavior of the line with buffers and

explore the distribution and quantity of WIP inventory that accumulates. Van Ryzin, Lou, and Gershwin (1993) investigate optimal production controls for two-machine models with extensions to more complicated systems. Bai and Gershwin (1994) consider operations, failures or repairs and starvation or blocking to develop a real-time control model to generate production schedules.

- operation and control in cellular manufacturing: Sinha and Hollier (1984) try to identify and establish some of the most significant and important control problems in cellular manufacture. Kekre (1987) investigates the impact of increasing the number of items made in a cell on its performance. Wemmerlov and Hyer (1987), (1989), report the findings of a survey of 53 American users of group technology and give an overview of a large number of research topics related to cellular manufacturing. Karmarkar and Kekre (1990) formulate the problem of selecting equipment mix and capacity for a manufacturing cell producing a group of items; lead time performance is considered together with the capital costs of equipment. Kamrani and Parsaei (1992) develop a methodology for forming machine cells using part's design and manufacturing dissimilarities. Ravindranath and Ebeling (1992) report on cell controller research. Liu and Lin (1993) investigate cells efficiency on the sequencing of robot moves. Lin and Chiu (1993) generalize the performance behavior of some manufacturing cells from the results of extensive simulation runs.

Yet another taxonomy deals with:

- scheduling issues: Karmarkar, Kekre, and Kekre (1985) represent the relationship between lot-sizing and shop performance using a queueing model which is then embedded in an optimization routine that searches for optimal lot sizes. Glassey and Resende (1988) develop release policy for job shops where the main source of randomness is due to machine failure and repair. Zipkin (1991) computes optimal lot sizes in the economic lot scheduling problem for several items on a single machine. Rosa-Hatko and Gunn (1991) review possible approaches to building queueing control models with switchover via stochastic dynamic programming formulation. Gascon, Leachman, and Lefrancois (1994) compare six different heuristics for a single-machine scheduling problem with stochastic demands. Ou and Wein (1995) study a dynamic scheduling of production/inventory system with by-products and random yield.

- inventory policies and control: Sahin (1979) derives time-dependent and stationary distributions of inventory position and on-hand inventory under (s,S) policy for continuous review inventory systems with general inter-arrival and demand distributions and a constant lead time. Zipkin (1986) proposes an approach to modeling a production facility that makes many products in large, discrete batches, when demands and the production process are stochastic. Karmarkar (1987) examines the relationships between lot sizes and lead times and their implications for lot sizing and WIP inventories for batch manufacturing shops with queues. Bielecki and Kumar (1988) show that there are ranges of parameter values describing an unreliable manufacturing system for which zero-inventory policies are exactly optimal even when there is uncertainty in manufacturing capacity. Altiok and Ranjan (1995) develop the approximation procedure to obtain the steady-state performance measures of the system controlled by a continuous-review inventory policy. Glasserman and Tayur (1995) develop estimators of derivatives with respect to base stock levels in multiechelon systems operating under base stock policies. Zipkin (1995) explores the performance of a multi-item production/inventory system operating under FIFO policy and the longest-queue (LQ) policy.

2.5 Closing Remarks

There is considerable research literature on manufacturing systems and related topics. Science often works most efficiently by studying the simplest possible system that exhibits a phenomenon. The initial systems that have been studied appear to be simplistic and academic, while the practical production processes are manifold and difficult to approach.

In this thesis, we focus on cellular manufacturing, but the results are also applicable to flow line systems. Previous studies do not directly apply to our research. They are mostly based on modeling approaches like Markov processes and queueing networks, and therefore are only suitable for relatively simple systems with restrictions on distributional assumptions. However, the results found in the literature do raise the question of whether the findings from the simple models are sensible for the practical and more complicated manufacturing layouts. The modeling techniques, used in studies of the described various taxonomies, tend to assume exponential processing times and Poisson arrivals. One issue we would like to address in this thesis is whether those are appropriate ways to design a system.

From the literature review presented here, we learned that manufacturing systems studied are mostly quite simple and often limited to unrealistic assumptions. The results from the theoretical models are not verified in more complex settings. We feel that our research on the PAC system opens possibilities to reduce the gap between theoretical models and true systems. The PAC model is suitable not only to test the results obtained by queueing theory, but it can be used to study issues of physical design, operation, control, inventory and scheduling for cases of more complex manufacturing settings.

Chapter 3

PRODUCTION POLICIES - DESCRIPTION AND SURVEY

3.1 Introduction

The production policies discussed in this thesis are the coordination schemes which determine when and how material and information flow through the system. Manufacturing systems operate on different coordination policies. The formal schemes which are the most often discussed include Materials Requirements Planning (MRP), Kanban, Base Stock Systems (BSS), Optimized Production Technology (OPT), Local Control (LC), Integral Control (IC) and Constant Work-in-Process (CONWIP). These will be key themes in this thesis along with the more general PAC framework.

Coordination policies are often described by push or pull control characteristics. In this chapter, we outline the main features of the push and pull control systems, and then, we describe various production coordination schemes together with a literature review of relevant recent studies.

3.2 Push versus Pull Systems

The terms "push" and "pull" refer to the means for releasing jobs into the production facility. The useful distinction is the way in which these methods utilize the capacity at the individual work centers, that is, whether these centers are allowed to produce without an imposed end item schedule (Vollmann, Berry, and Whybark, 1992). The traditional European-American industries operate on push systems, where jobs are triggered by paper-work authorizing production batches, routing and due dates. In the produce-to-stock environment, jobs are released to the first stage for manufacturing, and, in turn, this stage pushes the work in process (WIP) to the following stage and so forth until the production reaches the final stage (Sarker and Fitzsimmons, 1989). In a push system, the work is released to the stage according to the time of the expected customer demand, that is, in anticipation of a future need. As this is uncertain, the push system tends to send orders for production earlier or keeps extra parts in inventory between stages to overcome an incorrect forecast. This can result in satisfied customers, but may increase carrying costs significantly.

The Japanese pull system, often referred as the just-in-time (JIT) technique, is designed to minimize WIP and its fluctuations (Groenevelt, 1993). Increasingly, JIT is to be found in non-repetitive environments, where products are moved through the manufacturing process in short lead times, without tracking or paper-work and with capacity utilization being a result and not an objective. It is characterized by the practice of downstream stages pulling stock from previous operations as needed. In an ideal situation, the pull system holds just one item in WIP inventory (Philipom et al., 1987). When an item is taken from a stage by a succeeding stage, this authorizes this production stage to manufacture an additional unit to replace the one just taken. As a result, each stage produces "just-in-time" to meet the demand needed by succeeding stages, which is finally controlled by the customer product demand. The ideal pull system with one unit of inventory at each stage is seldom achievable in a real manufacturing system, because of variations in processing times, imbalance of workloads between stages, demand fluctuations and machine breakdowns. Consequently, a substantial amount of WIP may still be maintained at each stage of the system in order to reduce the effects of these variations. Generally, implementation of pull systems requires short setup times, small batch sizes, few part types, and processing times with very little variability.

3.3 MRP: Material Requirements Planning

Traditional western approaches to multistage production scheduling have been characterized by a production schedule for each stage. The first simple scheme, the use of economic order quantities or production quantities at each cell, was not successful (Vollmann, Berry, and Whybark, 1992). The direct extension to these approaches was the development of the base stock system. Next, a focus on job-lot manufacturing environments led to the introduction and refinement of material requirements planning (MRP). Authors such as Orlicky (1975) and Wight (1984) have described the virtues of MRP over methods, such as reorder point system (ROP). The MRP approach has several characteristics:

- all relevant information (including material requirements, work-in-process levels, machine status, and inventory levels) is stored in a central computer, implying a centralized control and a coordination among the work centers;
- implementation has stimulated the development and use of well organized information systems;
- three stages of capacity planning (resource requirements planning, rough-cut capacity planning and capacity requirements planning) are included and examined;
- the effect of a master schedule on the detailed plan is examined rather than developing a plan that lives within the bounds of capacity.

To a certain extent, under a MRP type system, work is "pushed" through the system (MRP can be also seen as a "forecast pulling system"). In a push system, throughput is equal to the input rate (assuming the system is stable) and work releases are directly linked to due dates. MRP controls throughput by establishing a Master Production Schedule (MPS) and measures work-in-process (WIP) to detect problems in meeting a schedule. MRP determines the period-by-period (time-phased) plans for all component parts and raw materials required to produce all the products in the MPS. This material plan can thereafter be utilized in the detailed capacity planning systems to compute labour or machine center capacity. For a detailed description of MRP and its components, see Vollmann, Berry, and Whybark (1992).

Although MRP systems are widely used, they have not always lived up to expectations. MRP systems take actual demands into account and trigger production of "dependent" items in the required quantities, and required times as estimated by the lead time offsetting procedure. A fundamental problem with MRP logic is that the dependence of production lead times on shop conditions, batch sizing and release timing is not recognized (Karmarkar, 1993), (Maxwell, Muckstadt, Thomas, and VanDerEecken, 1983). Instead, lead times are taken to be given exogenously. They are assumed constant at any point in time, whereas, in reality, they vary perhaps dramatically based on the current load in the plant. This inability of MRP to relate lead times to capacity loading and order release policies can lead to poor on-time, work-in-process and lead time performance. Another commonly recognized problem with MRP is so called "nervousness", in which only minor changes in the master production schedule can create instability in the MRP plans (Vollmann, Berry, and Whybark, 1992). Nervousness created by such changes is most damaging in MRP systems with many levels in the product structure.

Some reasons for MRP failure can be summarized as follows:

- the inability to maintain data information at a level of high reliability;
- a lack of an inherent improvement mechanism;
- a lack of real-time coordination among the consecutive stages means that frequent rescheduling is necessary to keep the total system under control;
- nervousness.

Recently, the MRP system has usually been analyzed in direct comparison with other coordination policies. Some researchers have tried to formulate a hybrid approach that combines MRP and some other approach in a hierarchical way. Studies of MRP versus Kanban include Karmarkar (1986), Krajewski, King, Ritzman, and Wong (1987), Sarker and Fitzsimmons (1989), Rees, Huang, and Taylor (1989), Deleersnyder, Hodgson, King, O'Grady and Savva (1992), Spearman and Zazanis (1992). Krajewski et al. identify critical factors for improving performance in the manufacturing environment and relate them to the MRP and Kanban policies. Sarker and Fitzsimmons perform a simulation and comparative study of the performance of push and pull systems, where Rees et al. report on a comparative analysis of an MRP lot-for-lot system and a Kanban system for a multistage operation. Deleersnyder et al. focus mainly on operational control problems. They develop a discrete time Markov model for a single-card Kanban system, which has extensions to more complex, hybrid push/pull, control schemes. Spearman and Zazanis examine the behavior of push and pull systems and identify a control strategy that has push and pull characteristics and appears to outperform both. Some studies of MRP and BSS are by Lambrecht, Muckstadt, and Luyten (1984) and Buzacott, Price, and Shanthikumar (1991). Lambrecht et al. describe theoretical models for determining optimal decisions in the uncertain environment and develop a heuristic to examine safety stocks. Buzacott et al. study the two-cell flow line under MRP and BSS and develop an approximation scheme for the service level measures. Some comparison studies of MRP and CONWIP are by Roderick, Toland, and Rodriguez (1994) and of MRP, JIT and OPT are by Plenert and Best (1986). Studies in the field of MRP performance evaluation, lot sizing, implementation, and general planning framework are done by Maxwell, Muckstadt, Thomas, and VanDerEecken (1983), Narasimhan and Melnyk (1984), Karmarkar, Kekre,

Kekre, and Freeman (1985), Takahashi, Muramatsu, and Ishii (1987), Hong and Hayya (1993), Lagodimos (1993),Cho, Kim, and Kim (1994).

3.4 Kanban: Japanese Card System

The Kanban production authorization system has been widely implemented as a control scheme for just-in-time manufacturing philosophy. "Kanban" which is the Japanese word for card is an information transmission device used in the Kanban control system to coordinate production and material flows. This system is normally applied in systems with reliable processes, low setup times, static demand and excess capacity. The use of kanbans avoids the need for complex information and hierarchical control systems on the shop floor. The first English language article on Kanban production control appeared about 20 years ago, when Sugimori, Kasunoki, Cho, and Uchikawa (1977) described the Toyota production system. Since then, there have been many articles. There are a variety of models and approaches. These include single-card systems, two-card systems, models with flows of material as well as with flows of kanbans. Schonberger (1983) tried to examine the weaknesses and strengths of a single-card and dual-card Kanban system and gave general comments on the successfulness of Kanban system implementation to Western manufacturing.

The common element that distinguishes Kanban systems from conventional methods of control is the existence of finite in-process inventory buffers and, therefore, station blocking. The finished product is "pulled" from the system downstream at the actual (or less) demand rate. New orders are placed only when parts are withdrawn from the stock areas, and only when the appropriate authorization is presented. This has the effect of tightly controlling production, and sharply limiting the total amount of work-in-process and finished goods inventory. Kanbans can be used to implement both pull production and pull transportation control systems where the physical removal of inventory authorizes production and material handling, respectively.

A good modeling of Kanban systems would enhance the understanding of their behavior. One difficulty, when interpreting the research results, is due to the wide variety of Kanban systems. There is a lack of a uniform definition of Kanban production control and it is difficult for authors to compare their Kanban system to others appearing in the literature or to standard models in queueing theory. Reviewing the recent literature about Kanban systems, reveals the impracticality of giving a complete list of all published works. Berkley (1992) attempted such a review, where he grouped the studies according to the Kanban models described and the approaches used. The most recent studies try to develop an optimal policy for the number of kanbans. Gursoy, Altiok, and Danhong (1994) use a semi-Markov decision model for two-stage, pull-type production system. Chang and Yih (1994a), (1994b) describe the generic Kanban system for dynamic environments and apply a simulated annealing method. Mitwasi and Askin (1994) propose a heuristic algorithm for selecting the number of kanbans. Berkley (1994) tests the minimum performance level for Kanban production lines. Co and Jacobson (1994), Askin, Mitwasi, and Goldberg (1993) and Fukukawa and Hong (1993) examine the Kanban system in the context of the JIT production system. Studies on approximating the performance are done by Berkley (1993), Tayur (1993) and Buzacott, Price, and Shanthikumar (1994). Recent studies by Muckstadt and Tayur (1995) extend the structural results of Tayur (1993) and, using a serial manufacturing model, demonstrate that CONWIP and the traditional kanban control are just two extremes in a finite family of implementable pull controls.

Other studies are a mixed integer programming model for determining the number of kanbans in a multiproduct, multistage production system (Bard and Golany 1991); a two-card Kanban-controlled line, viewed as a generalization of the tandem queue (Berkley

1991); an analysis of Kanban discipline and comparison with classical production controls including the simulation results (Mitra and Mitrani 1991, 1990). Simulation studies are presented by Huang, Rees, and Taylor (1983), Davis and Stubitz (1987), Rees, Philipoom, Taylor, and Huang (1987), Sarker and Harris (1988), Gravel and Price (1988), Gupta and Gupta (1989), Sarker (1989), Berkley and Kiran (1991).

Among the most often referenced studies of Kanban systems are results described by Mitra and Mitrani (1991), (1990), Wang and Wang (1990). All three formulate the Kanban-controlled lines as Markov chains and define general blocking principles. Deleersnyder, Hodgson, Muller, and O'Grady (1989) with their model for 3-stage system, describe the effects of the number of kanbans, the machine reliability, the demand variability and safety stock requirements on the performance of a Kanban controlled pull system. So and Pinault (1988) report on a method to measure the performance via the average percentage of demand backlogged of a pull system. Karmarkar and Kekre (1987) study the effects of batch sizing on expected inventory and costs. Kimura and Terada (1981) focus on the general description of aims and methodology of the pull system including some basic equations for a Kanban system and a mathematical model with deterministic settings. Other studies include Bitran and Chang (1987), Philipoom, Rees, Taylor, and Huang (1987), Seidmann (1988), Li and Co (1991). More of the Markovian modeling approach is found in studies by Kim (1985), Graham (1992) and Siha (1994).

3.5 BSS: Base Stock System

The base stock system (BSS) was introduced in the late 1950s and early 1960s as a direct extension of trying to coordinate production using economic order quantities. One of the BSS system pioneering works is the study by Clark and Scarf (1960). The main concept behind the BSS is the inventory control for multi-echelon stock (originally proposed for the control of inventories in distribution systems). BSS systems were developed in order to overcome the problems that arise when each stocking point in a multi-echelon inventory system bases its reorder decision just on the demand it receives from the next lower echelon. Information on final demand is made available at all stocking points so each level can make decisions on inventory replenishment by comparing actual downstream inventory with a target total downstream inventory.

The most common type of BSS is one where an order point and an order-up-to-level are used for each stocking point, that is, an (s,S) system (Silver and Peterson, 1985). The order-up-to-level S, also called the base stock level, is determined through the relation:

$$S = s + Q$$

that is, echelon inventory position = (echelon stock) + (on order). The "on order" term refers to an order placed by echelon on the next higher echelon. The ordering decisions at any stocking point in the BSS are made as a result of end-item demand, not orders from the next level downstream. This results in much less variability in inventory levels and lower safety stocks.

Lambrecht, Muckstadt, and Luyten (1984) not only compare the BSS with MRP, but also develop two modeling approaches (dynamic programming and Markovian decision problem) of the BSS as well as improve the Clark and Scarf's approximation procedure. Buzacott, Price and Shanthikumar (1991) focus on the fact that the BSS reduces the likelihood that each level (echelon) will make a decision that results in fluctuations in final demand being amplified as information about final demand is transmitted to higher levels, and develop a framework for information flow that gives an alternative implementation of base stock control. Lee and Zipkin (1992) explore a natural generalization of the classic tandem-queue model. Its simplest version, which results in a stationary demand-pull or base stock policy, is used to propose and test a tractable approximation scheme.

3.6 OPT: Optimized Production Technology

Optimized Production Technology (OPT) as a philosophy and a trademark of a software system was developed in the beginning of the 1980s when a great deal of attention was focused on the so called "theory of constraints" (Goldratt and Cox, 1986). (OPT originally stood for optimized production timetable, but now stands for optimized production technology.) In the context of a situation where splitting is irrelevant, the key idea of Optimized Production Technology is to control the level of work in progress upstream of any bottleneck facility. In OPT, production is not scheduled with either a push or pull technique, but on a "bottleneck" basis. The bottleneck areas in a facility are analyzed. Production is planned so that the bottleneck work centers will be utilized to the maximum and all other departments which are not bottlenecks will be planned to keep the bottleneck departments working at full production at all time. In OPT, production waves are prevented by tighter scheduling and through the use of safety capacity. Nonbottleneck work centers all have some amount of excess capacity that is used to handle overloads of production. The emphasis is not on "keeping the worker busy", but rather on keeping production flowing smoothly.

There are only a few articles that deal with the OPT system. Lundrigan (1986) and Plenert and Best (1986) present OPT as a new way of thinking, describe its advantages and disadvantages and relate it to both MRP and JIT policies. Vollmann, Berry, and Whybark (1992) give a quite comprehensive description of OPT and place it in the general frame of the MPS system with comparison to MRP and JIT policies. Buzacott, and Shanthikumar (1992a), (1992b) include OPT in the category of traditional production control methods and define OPT via a certain parameter setting for the PAC system.

3.7 LC: Local Control

Local Control is performed when buffers have physical space restrictions, for instance, a transportation chain with limited space for product carriers. In a locally controlled system, an order for an item is generated if the level of the inventory in the buffer after a production cell falls below the control limits. On the other hand, a production cell stops producing if the buffer is full. Local Control only uses information about availability of production capacity at a cell and its buffer, and no information from other cells in the system.

Wijngaard (1979) and De Koster (1988) show how in flow lines it is possible to use regeneration methods in models with continuous production rates instead of service times. In almost all these studies, the buffers are local.

3.8 IC: Integral Control

Integral Control was a further development of control policies in the general search for improvement. Like OPT, it decomposes the manufacturing system and applies some MRP and BSS features to coordination of the entire system.

By integral control, we mean release rules that take into account the total inventory over a range of cells through which work flows. Instead of basing the decision purely on the inventory position in the product store associated with the cell, as in local control, the release decision is based on the total inventory in a number of cells and their respective product stores. Since more flexible automated production systems are becoming available, especially with respect to buffer usage, instead of being hardware controlled, the buffers are software controlled. Therefore, it is possible to implement non-local control limits. For example, the sum of the contents of a number of buffers may not exceed some critical value N. Non-local buffers caused by physical restrictions occur in flexible manufacturing systems, where the machines have a common storage, realized by a loop conveyor. The possibility of the use of software-induced control limits leads to the question of how to set these control limits, in order to get optimum performance.

De Koster (1988) presented the behavior of production lines with integral control limits and suggested that such lines can be approximated by flow lines with local buffers. De Koster and Wijngaard (1989) studied some simple models from which they conclude that there is no significant difference in performance of control systems with local or integral control.

3.9 CONWIP: Constant Work-in-Process

The CONWIP order release strategy seeks to operate the system at a constant level of work in process. This involves selecting an appropriate estimated WIP level. CONWIP can be considered a hybrid of both push and pull production control systems. Under CONWIP, jobs are pulled into the factory by the completion of any other job and then they are pushed from one cell to another. CONWIP can be also called a generalized form of kanbans. In a Kanban system, each card is used to signal production of a specific part. CONWIP production cards are assigned to the whole production line and are not part number specific. Kanban pulls at each cell, CONWIP pulls only at the front of the line. The main advantage of the CONWIP system over the strictly pull system is holding of a minimum WIP at each production cell. The goal in developing CONWIP was to create a system that possesses the benefits of a pull system and can be used in a wide variety of manufacturing environments. One of the most recent studies on CONWIP system is by Muckstadt and Tayur (1995). They demonstrate that in a case of a serial system, CONWIP and kanban are just two extremes in a finite family of implementable pull controls. By using simulation they show a set of conditions and performance criteria under which CONWIP is preferred over the kanban mechanism, and vice versa. Roderick, Toland, and Rodriguez (1994) performed a simulation in an actual plant environment (The Westinghouse Electric, College Station, Texas Plant) using MRP and CONWIP procedures and showed the superiority of CONWIP above MRP for the tested facility. Other interesting studies on CONWIP system are done by Spearman, Woodruff, and Hopp (1990), Hopp and Spearman (1991), Spearman and Zazanis (1992) and Duenyas, Hopp, and Spearman (1993). The study of Spearman, Woodruff, and Hopp models a CONWIP production line with deterministic processing times and exponential failures and repair times as a closed queueing network and derives an approximation technique to estimate performance. This approximation is robust and useful as the basis for a procedure for selecting an economic production quota and a card count for a CONWIP line. Hopp and Spearman develop an approximate regenerative model (ARM) for estimating throughput and average cycle time as a function of WIP level for several machine models under a CONWIP control strategy. Spearman and Zazanis compare MRP and Kanban, offer theoretical motivations for the apparent superior performance of pull system and describe the benefits of using CONWIP as the solution to the present MRP users. Duenyas et al. give practical advantages of CONWIP over push and pull systems and illustrate their conclusions by presenting the simulation results.

3.10 Closing Remarks

There is an extensive literature on performance analysis of production systems operating under various control policies. Different techniques are used to study the proposed models. The main tools are simulation and heuristics based on queueing theory. A considerable effort has been spent in modeling production systems within the framework of push systems. Recently, models of multi-stage pull type manufacturing systems have appeared in various publications, mainly due to the success of the Kanban concept. There are many research studies, wherein the Kanban policy is compared with some other production policy such as MRP, CONWIP, and BSS. These focus on understanding the system behavior, the impact of buffer storage, and the computation of performance measures such as average inventory levels, throughput, station utilizations and percentage downtime. Many researches try to specify circumstances, where one production coordination policy does better than the other. Other studies try to recognize advantages of various approaches and develop hybrid concepts to improve the production and control of the manufacturing process, but none is perfect. The problem of controlling the flow of materials in a modern, cellular manufacturing system is one of the fundamental problems of manufacturing engineering. Our research on the PAC system will contribute to a better understanding of those issues.

Chapter 4

THE PAC SYSTEM: A MORE GENERAL APPROACH FOR CONTROL OF PRODUCTION PROCESS

4.1 Introduction

The literature reviewed in the previous chapter focused on specific control policies used in manufacturing. No one policy is inherently better than the others. However for a given manufacturing process, there should exist an optimal scheme for coordinating its production and inventory. This thesis will focus on a new conceptual model, developed by Buzacott and Shanthikumar (1992a, 1992b) with the goal of determining the optimal production policy for a given manufacturing situation. Both authors introduced a general approach to the coordination and control of material and information flow in a multiple-cell manufacturing system. They called this the "Production Authorization Card (PAC) System".

There is earlier work that moves in this direction. Veatch and Wein (1991) presented an optimal coordination scheme for the two-cell flow line system. Although the model is quite simple, the results give much insight. Veatch and Wein studied optimal controls using dynamic programming and compared the different control mechanisms that have been proposed for manufacturing facilities such as Kanban, BSS and CONWIP.

A third source of material for development of a general coordination scheme is research done by Cheng. Although we do not follow her approach directly, her work introduces a general class of control rules which unifies and generalizes a number of control rules based on the so called "blocking mechanisms". A feature that is characteristic in all types of blocking mechanisms is that they focus on the control of the maximum number of completed (but blocked) jobs at each station, and they differ in the timing at which the service facility is turned off when the downstream station is full. Some of those blocking issues are related to ideas in the design of computer operating systems. Cheng devoted her Ph.D. thesis (1991) to the study of the tandem queues with general blocking. Some related issues can be also found in her recent publications (1995), (1993), (1992) and also in Cheng and Yao (1993) and Cheng and Zhu (1993).

In the remainder of this chapter, we describe the main features of a PAC system with emphasis on control procedures and parameter settings for various coordination schemes. For the most part, our description comes from Buzacott and Shanthikumar. However, in section 4.2.6.2 we comment on some constraints on the parameter choices that we have discovered in our research. Parameter choices that violate these constraints will cause the system to become blocked in the sense that, if the information flow and material flow rules are adhered to, then neither material nor information will continue to flow. We also reformulate the parameter choice for IC/CONWIP policies in the case of a non-serial assembly system (refer to 4.2.6.3). Buzacott and Shanthikumar discuss parameter choices only for cells in series. In the case of IC/CONWIP, those parameter settings require some adjustments to be applied to more complex manufacturing configuration. Finally, we draw some conclusions that are related to our research.

4.2 PAC System

4.2.1 Components and Description of the Control Process

As conceptualized by Buzacott and Shanthikumar, the manufacturing system consists of production cells and product stores connected by material flow. Each cell produces some set of parts, either subassemblies (intermediate products) or finished products. We assume that each part completed by the cell has a unique storage location, a store located directly after the cell. From each storage location, the part can be supplied to a number of subsequent cells as a component in various production processes. In some cases, it might be appropriate to distinguish several stores containing the same part (due to the geographical location or perhaps ownership), which will require the cell to choose between storage location. However, for this thesis, we proceed with the assumption of a unique storage location for each product type.

Production management is viewed as decentralized, but operating under a control policy. Each production cell has its own management responsible for a range of production tasks within the cell. No production activities can be performed without a specific production authorization. The production cell may consist of a number of work centers or machines. The cell management is responsible for coordination of their internal operations.

Each product store can keep a number of distinct parts. If the same part is produced by more than one cell, then all cells would deliver to the same product store. Product stores can be logically considered to be "between cells", at both the output of cells where the part is produced and at the input of cells where the parts are delivered for processing. There can also be an inventory of parts within the cell, but such inventory is considered to be "owned" by the cell and enables the cell management to carry production operations in an effective manner.

At the boundaries of the manufacturing system, there are production cells that receive parts or raw materials from the external suppliers and product stores that deliver the finished parts to the external customers.

In order to manage the production process, the cells must receive information on what and when to produce and where to deliver, and the stores on where and when to supply certain materials. The flows of material and information between cells and stores are coordinated by issuing various tags either by cells or stores. Instructions and authorizations are embodied in tags; each tag is being associated with a single item of a specific part.

A tag can perform two functions:

- to instruct somebody to do something;
- to provide information about the existence of a part or about a present of future demand.

In the system there are different types of tags: requisition tags, order tags, process tags, material tags, PA tags, surplus tags, cancellation notes and order cancellations.

Next, we describe in detail the different types of tags used in a PAC system and outline the procedures for control and coordination of material and part flow within the system. This description closely follows that of Buzacott and Shanthikumar (1992a).

4.2.2 Tags

Cell Management Generated Tags:

Requisition Tags

A requisition tag is sent by a cell to a storage location of a part type. By sending this tag, the cell requests immediate shipment of the part item to the cell. If the store is empty, the requisition tag waits in a queue at the product store. The number of requisition tags at the store represents the backlog of unmet demand for parts. Parts requested by a cell are supplied according to some priority rules defined for the requisition tags, when there are not enough parts in the cell to meet all the requisitions. The requisition tag would also accompany the delivered product part to the cell and function as a delivery advice note.

Order Tags

An order tag is also sent from a cell to a product store. Its purpose is to provide the product store with the information on the planned demand in the future. When the product store receives an order tag, it can expect that at some time in the future it will also receive a requisition tag. Each order tag is followed by a requisition tag with some delay, characteristic of the part requested. If there is no advanced notice about future demand, an order tag and its associated requisition tag are sent together from the cell to the product store.

The delay in the releasing of the requisition tags for final products is associated with forecasting. In this case, we assume that customers will not be sending order tags, only requisition tags. The order tags will be generated by production management. Conceptually, the requisition delay at any stage is equated to the production lead time, that is, to the time taken from the time production is authorized to this stage, to the time it is completed and the part is available to be delivered to a customer or the next stage.

Cancellation Notes

A cancellation note might be issued by a cell to annul an order tag. This message notifies a product store that an order tag which was previously received will not be followed by a matching requisition tag and allows the store to destroy the order tag.

Material Tags

Material tags are generated to provide information about the existence of physical parts. When the requisition tag returns to the cell as a delivery advice note, it is converted to a material tag. Each material tag accompanies a part during the production process at the cell. We can identify assembly, disassembly and scrap operation in the cell, that is, different material tags are combined into one material tag, a single material tag becomes a set of material tags or an associated material tag gets destroyed. A material tag accompanies each produced item to the product store, so that the number of material tags is identical to the inventory of the part in the product store. A material tag is destroyed when the part item on the receipt of a requisition tag is supplied from the product store to cell.

Process Tags

When the production process in the cell is completed, the cell will issue a process tag, which will authorize delivery of the completed part to its unique storage location. The process tag will accompany the part item to the product store and will wait there with other process tags.

Product Store Generated Tags:

Production Authorization Cards (PA Cards)

PA cards are sent from a product store to a cell. Their purpose is to advise the cell about the existence of a demand and authorize the cell to begin production. Before a PA card is sent from the store to the cell, the store must already have received an order tag and have a process tag available. The match of order tag with a process tag generates a PA card.

Surplus Tags

A surplus tag is generated when a part must be produced because it is a byproduct of the part whose manufacture was authorized by the PA card. The surplus tag indicates that the part was not produced as a result of a previously received order tag. An arrival of the next order tag will cause a cancellation of the existing surplus tag and no generation of a PA card. Surplus tags can be also created by receiving order cancellation notes after the production of the part has been already completed.

Order Cancellations

An order cancellation will destroy the order tag to which the cancellation note relates. In case the order tag has already generated a PA card, then the cancellation note will be passed to the cell, which in turn will generate cancellation notes to be sent to all product stores supplying parts, unless the parts have already been received by the cell. If parts have been already received by the cell, the cell management has to decide to continue or to stop production. In case the production is allowed, then a surplus tag will be generated and will accompany the part to the product store.

4.2.3 Control Procedures

Figure 4.1 shows schematically the movements of the tags between cells and stores (in case of cells in series). Requisition tags are sent from cells to stores when items are needed. On receipt of the requisition tag, the store either supplies the cell immediately with the item or the requisition tag is transformed into the backlog of the unmet demand for this item if the store does not have the requested item. Before the cell issues the requisition tag, it sends an order tag for the item. At each store, there are a number of process tags. An arriving order tag is matched with a process tag and eventually becomes a PA card. The PA cards are accumulated to the required PA batch size and sent as a packet to the production cell. On receiving the PA card, the cell is authorized to start the production.

Because the cell may require component parts and raw materials, it will, in turn, generate order tags and requisition tags for the required items and send them to the appropriate stores. Requisition tags do not need to be sent at the same time as the order tags. There can be some delay between issuing the order tag and the corresponding requisition tag. The requisition tag usually accompanies the item from the store to the cell as a delivery advice note. Once the cell receives all required parts it can start the production. After completion of the production process, the PA card accompanies the part to the store and is converted into the process tag.

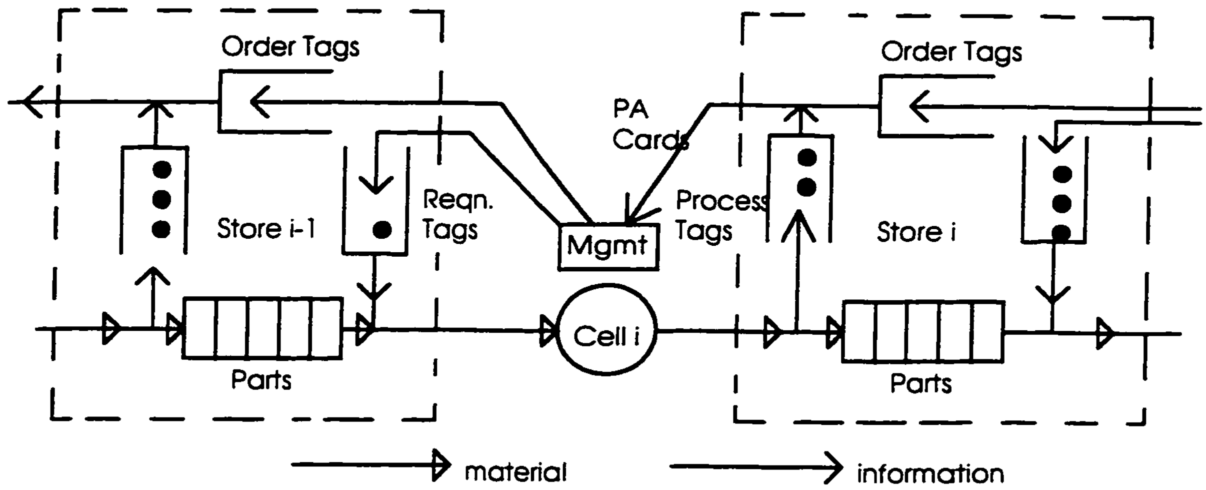


Figure 4.1 Information and Material Flow for Cell *i*
(after Buzacott and Shanthikumar 1992a)

There is one potentially important aspect of control that Buzacott and Shanthikumar essentially ignore. This involves the assignment of priorities when decisions are required to resolve competing use of resources. There are two circumstances in which this is important. The first case deals with supplying products from stores to cells in response to requisitions, and the second one is with regard to the starting the production process when there are PA cards present for different parts. In the remainder of these thesis, we follow Buzacott and Shanthikumar's lead, which is to assume a FIFO policy. However, these issues are potentially important.

4.2.4 System Parameters

In the previous section, we described the general framework of the PAC system. We now turn to the set of control parameters which, under appropriate assumptions, are sufficient to determine the nature of the production system and its performance. We consider a PAC system consisting of m multiple cells ($i = 1,\ldots,m$) and n different product types ($j = 1,\ldots,n$).

The parameters are as follows:

- z_{ij} - initial (terminal) inventory level of the output store i for cell i for product j (also associated with the safety stock or planned minimum on-hand inventory in conventional MRP calculations). When there are no external demands for a long time, the z_{ij} will be the static inventory levels of the system.
- k_{ij} - process tags limits. The number of process tags for product j (at the same time the PA cards limit) assigned to store i.
- r_{ij} - PA batch size. The size of packet for transmission of PA cards for product j from store i to cell i.
- τ_{ij} - delay in releasing requisition tags. It is the delay at cell i for product j between transmitting the order tag for product and releasing the associated requisition tag (also identified with the lead time used in the MRP setting).

4.2.5 Parameter Choice for Different Production Policies

In this section, we review how, through the appropriate choice of parameters, the PAC system can be specialized into a wide variety of traditional coordination schemes. As we have already seen, many traditional production policies are either ones that include forecasting or not, that is, push versus pull. The PAC system approaches forecast of a future delivery by issuing order tags. The time between the issue of the order tag and the requisition tag can be seen as the forecast interval. The character of the system, push or pull, can be modeled by requisition delay. A pure pull system corresponds to a requisition delay of 0. As the requisition delay increases, the nature of the system changes towards a push system, as the cells are allowed to produce (utilize capacity) by a specific end item schedule.

Before describing the PAC parameter setting for each policy, first we discuss the issue of the requisition delay given by parameter τ, as it is fundamental for policies with $\tau \geq 0$.

4.2.5.1 Requisition Delay

A delay in issuing a requisition tag for policies with τ_{ij} ($i=1,\ldots,m$, $j=1,\ldots,n$) can be explained by representing manufacture by a network and using the critical path method (CPM) approach. The requisition tag delay is related to the lead time, that is, to the flow time of a product through the upstream stages. A manufacturing process is represented by a critical path network, and the CPM is used to find the duration of the lead time, assuming that the duration of each activity is known. Activities in the critical path network are equal to an estimate of the flow time of a product j through a cell i (l_{ij}). Events are the delivery of products from a cell i to a store i. Extra events and dummy activities (with zero time) are defined to represent the delivery of all required raw materials and intermediate products to a cell. CPM computes early event times, late event times, total float and critical activities called a critical path. The late event times in the critical path network determine the time of issuing requisition tags and correspond to the beginning of a manufacturing process at each cell.

For illustration purposes, Figure 4.2 shows a critical path network for a two-cell flow line (j=1). In parenthesis, above each node x, are indicated an early event time ET(x) and a late event time LT(x). Table 4.1 gives a description of activities for this network. In the case of this simple example, τ_1=4 and τ_2=9. In a case of m cells in series system ($i = 1,\dots,m$), this implies that the requisition tag authorizing shipment to the customer must be delayed by an amount $\tau_m = \sum_{i=1}^{m} l_i$. It also results in the following relationship between the delay times: $\tau_1 \le \tau_2 \le \dots \le \tau_m$. In chapter 7, when describing optimization experiments, we will observe that this is not always the case for the results obtained.

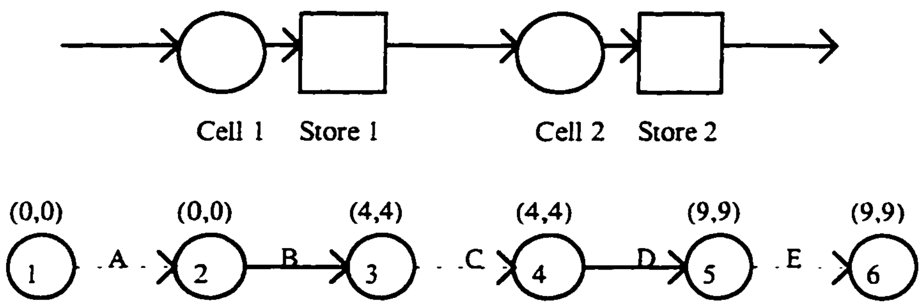


Figure 4.2 Critical path network for a two-cell flow line.

Table 4.1 Description of activities of the network representation for the two-cell flow line

Activity(x,y)	Predecessor	Duration
(1,2): A - delivery of raw material to cell 1	-	0
(2,3): B - processing time at cell 1 and delivery of product to store 1	A	l_1=4
(3,4): C - delivery of product to cell 2	B	0
(4,5): D - processing time at cell 2 and delivery of product to store 2	C	l_2=5
(5,6): E - delivery of final product to customer	D	0

4.2.5.2 PAC Parameter Settings

In the case of cells in series, the PAC parameters for different coordination policies are set as follows:

Produce-to-Order (PTO)

There are many variants of Produce-to-Order (PTO) systems. One common characteristic of all of them is the fact that production does not begin earlier than receipt of the order. The simplest PTO is the one where there is no attempt at controlling the level of work in process at any cell. The choice of parameters for such a case is as follows:

1. $z_i = 0$, $i = 1,\ldots,m$. Initially, there is no inventory in the manufacturing system.
2. $k_i = \infty$, $i = 1,\ldots,m$. PA cards are generated at stores as soon as the order tag arrives.
3. $r_i = 1$, $i = 1,\ldots,m$. PA cards are moved in batches of size 1.
4. $\tau_i = 0$, $i = 1,\ldots,m$. There is no delay between order tags and requisition tags.

A variant with $\tau_i \geq 0$, $i = 1,\ldots,m$, corresponds to the CPM approach: that is, the delay in issuing a requisition tag at cell would be set at the latest start time of the event in the critical path network that corresponds to the beginning of manufacture at the cell.

Base Stock System (BSS)

The choice of parameters is as follows:

1. $z_i > 0$, $i = 1,\ldots,m$. There is an initial inventory in each production store.
2. $k_i = \infty$, $i = 1,\ldots,m$. PA cards are generated at stores as soon as the order tag arrives.
3. $r_i \geq 1$, $i = 1,\ldots,m$. PA cards are moved in batches of any size.
4. $\tau_i = 0$, $i = 1,\ldots,m$. There is no delay between order tags and requisition tags.

The size of safety stock in each store is determined by weighing the penalties of a product store being empty (that is, not being able to supply another cell) against the inventory costs.

Material Requirements Planning (MRP)

The choice of parameters is as follows:

1. $z_i \geq 0$, $i = 1,\ldots,m$. The initial inventory can be identified with the safety stock or the planned minimum on-hand inventory in conventional MRP calculations.
2. $k_i = \infty$, $i = 1,\ldots,m$. PA cards are generated at stores as soon as the order tag arrives.
3. $r_i \geq 1$, $i = 1,\ldots,m$. PA cards are moved in batches of any size.
4. $\tau_i \geq 0$, $i = 1,\ldots,m$. The delay can be identified with the lead time used in the MRP calculations.

In MRP calculations, the work is released to a stage by taking into account the flow time of a typical job through the stage, its "lead time". Lead time is calculated by constructing and analyzing the critical path network of all activities required to produce the end products. The requisition delay at a cell is set equal to the latest start time of the corresponding activity in the critical path network. MRP systems are driven by a master schedule, which in our PAC setting can be considered to be a forecast of future receipts of requisition tags. It is also more appropriate to see customer demands coming directly as requisition tags and management implementing the MRP control in the form of generating matching order tags. If τ_m is the overall time to complete the product, then each order tag has to be generated at a time τ_m before the forecast receipt of the requisition tag or customer demand. τ_m is called the forecast time horizon. Note that MRP essentially uses a base stock mechanism with the modification that the demand forecast is included by way of the lead times.

Note: According to the above description of MRP, the relationship between the delay times $\tau_1 \leq \tau_2 \leq \ldots \leq \tau_m$ should also be satisfied (refer to 4.2.5.1). However, Buzacott and Shanthikumar do not include this in their parameter specification.

Kanban

In a conventional Kanban system parameter choices are as follows:

1. $z_i > 0$, $i = 1,\dots,m$. There is an initial inventory in each production store.
2. $k_i = z_i$, $i = 1,\dots,m$. If only one cell can supply one product store.
3. $r_i \geq 1$, $i = 1,\dots,m$. PA cards are moved in batches of any size.
4. $\tau_i = 0$, $i = 1,\dots,m$. There is no delay between order tags and requisition tags.

Note that the limit on k_i influences the system in two ways:

- the work-in-process at any cell is limited to k_i ,
- the information about the demands for the final products does not pass back to earlier stages of manufacture immediately. If a cell becomes a bottleneck (due to machine failure or labor problem), the upstream cells have no knowledge about any downstream information, and when the problem is resolved, these cells can become overflowed with work and in turn become bottlenecks.

Local Control (LC)

By local control we understand the policy in which a cell produces a product whenever the following conditions are met:

- input products/raw materials are available;
- machine and labor capacity are available;
- the product store to which the cell delivers completed products is not full (that is, it has still free storage space).

The parameter choice for the system operating on the basis of the local control is as follows:

1. $z_i > c_i$, $i = 1,\dots,m$.
2. $k_i = c_i$, $i = 1,\dots,m$.

3. $r_i = 1, \quad i = 1,\ldots,m.$
4. $\tau_i = 0, \quad i = 1,\ldots,m.$

Note: c_i indicates a number of parallel machines in a cell.

Integral Control (IC)

Integral Control is based on the total inventory in a number of cells and their respective product stores. The IC policy for the series of m cells as a whole can be applied with the following choice of the parameters:

1. $z_i \geq 0, \quad i = 1,\ldots,m.$
2. $k_i = z_i + k_{i-1}, \quad i = 1,\ldots,m-1, \quad k_m = z_m.$
3. $r_i = 1, \quad i = 1,\ldots,m.$
4. $\tau_i = 0, \quad i = 1,\ldots,m.$

Note: This is only one example of an integral control policy. For instance we might divide a line into stages and apply IC to each stage. Thus we can use IC among the subsets of cells and stores in the system. However, for this thesis, we restrict our interest in IC to the case above.

We discuss the parameter setting for the case of an assembly system in section 4.2.6.3.

Optimized Production Technology (OPT)

OPT controls the level of work-in-process upstream of any bottleneck facility. This can be realized by:

- setting $k_i \geq z_i + k_{i-1}$ for all cells i upstream of the bottleneck facility j;
- setting $k_j < z_j + k_{j-1}$ (and desirably $k_j \leq z_j$);
- having $\sum_{i=1}^{j-1} z_i + k_j$ equal to the work-in-process limit over cells 1,2,...,j.

Constant Work-in-Process (CONWIP)

If we apply Integral Control to the whole system, then we have a constant work-in-process equal to $\sum_{i=1}^{m} z_i$. CONWIP is aimed at maximizing the customer service subject to this policy. Placing initial inventory at the latter stages of the system provides better service to customers. When the production process adds a negligible value to the product at each stage, the optimal policy is to keep all of the initial inventories at the final stage. Thus, the optimized system known as CONWIP, comes from setting:

$$z_i^* = 0, \quad i = 1, \ldots, m-1; \qquad z_m^* = \sum_{i=1}^{m} z_i$$

4.2.5.3 Illustrative Example

We discussed the parameter choice for different production policies. Now, using a simple example, we illustrate behavior of the system controlled by different coordination schemes. Figure 4.3 presents a layout of a two-cell flow line, which produces a single product (j=1). This system is characterized by a total of 8 PAC parameters, two sets of z_i, k_i, r_i, τ_i for each cell i and store i (i=1,2).

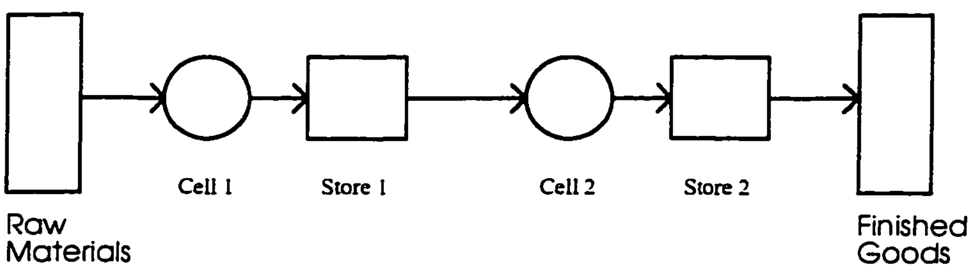


Figure 4.3 Two-cell flow line.

We investigate the production operation of this system for a customer demand for a total of 4 single items of the final product, arriving at time 50, 60, 70 and 80 respectively.

We assume, that the processing times are deterministic and equal to 30 and 20 time units at cell 1 and cell 2 respectively. Table 4.2 gives one parameter setting for each coordination scheme. These are chosen arbitrarily, within the range of parameter choices for each policy.

We performed hand simulations for the system operating under each policy and calculated the shipment times of all 4 product items to a customer. The results are given in Table 4.3.

Table 4.2 Parameter setting for different policies for a two-cell flow line

	z_1	k_1	r_1	τ_1	z_2	k_2	r_2	τ_2
PTO	0	∞	1	0	0	∞	1	0
BSS	1	∞	1	0	1	∞	2	0
MRP	0	∞	1	0	2	∞	1	50
Kanban	1	1	1	0	2	2	1	0
LC	2	1	1	0	2	1	1	0
IC	1	2	1	0	1	1	1	0
CONWIP	0	2	1	0	2	2	1	0

Table 4.3 Shipment time to customer

Arrival of customer demand	Shipment Time						
	PTO	BSS	MRP	Kanban	LC	IC	CONWIP
50	100	50	50	50	50	50	50
60	130	80	60	60	60	70	60
70	160	110	80	70	70	100	100
80	190	130	110	100	90	130	130

For clarity, we describe in some detail the hand simulation performed for Kanban. Figure 4.4 shows the time-phased schematic of the hand simulation for Kanban.

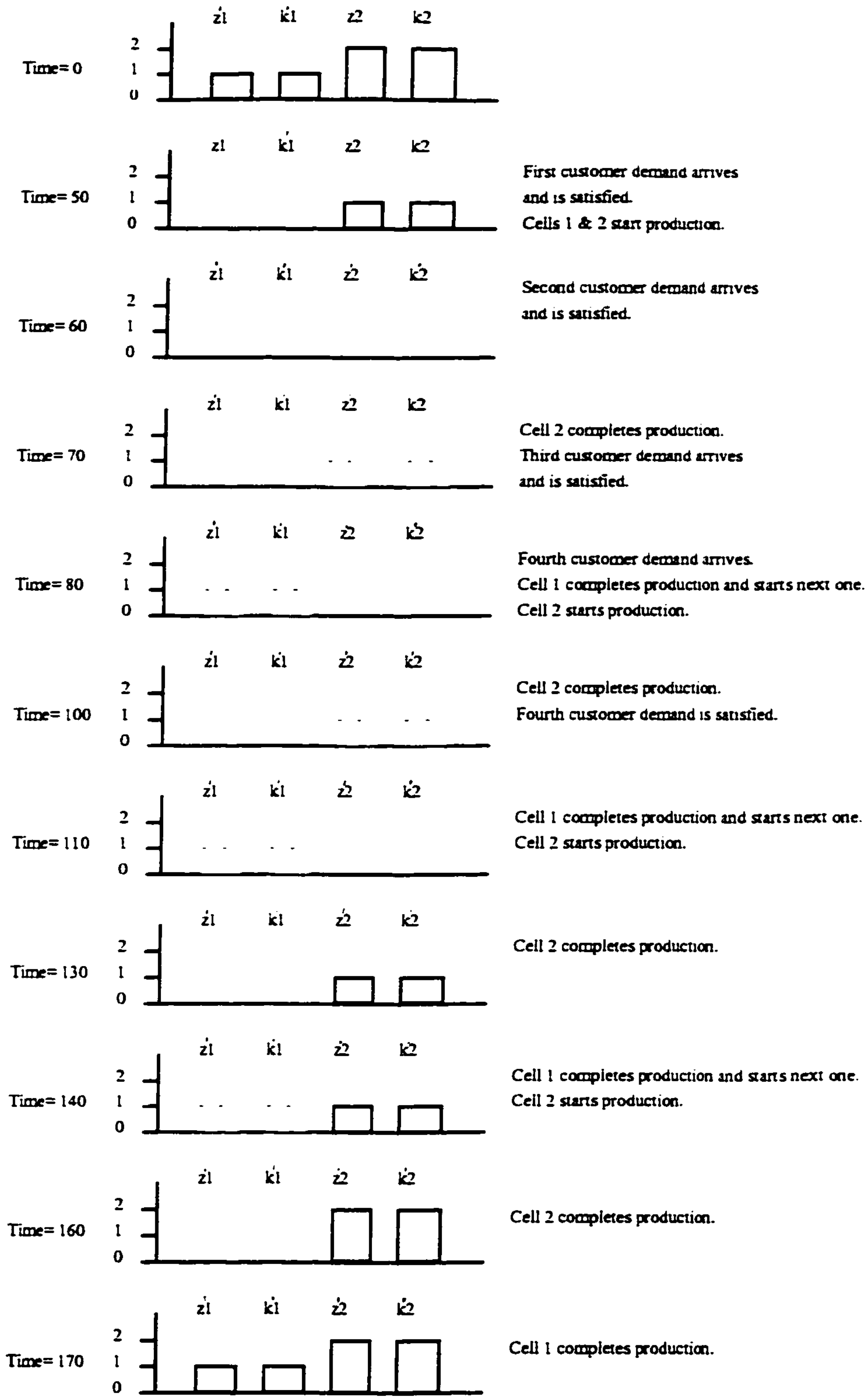


Figure 4.4 Time-phased schematic for Kanban.

We indicate by z'_i the actual value of a number of parts in inventory in store i and by k'_i the actual value of a number of process tags in store i. The first customer demand (arriving at time 50) is immediately satisfied, as z'_2=2. At time 50: z'_2=1 (it decreases by 1), PA card is sent to cell 2, k'_2 decreases by 1 (k'_2 =1), cell 2 sends order tag and requisition tag to store 1, store 1 sends a part to cell 2, z'_1 decreases by 1 (z'_1 =0), cell 2 starts production which will end at time 70, the PA card is sent to cell 1, k'_1 decreases by 1 (k'_1 =0); cell 1 starts production which will end at time 80. The second customer demand (arriving at time 60) is immediately satisfied, as z'_2=1. At time 60: z'_2=0 (it decreases by 1), the PA card is sent to cell 2, k'_2 decreases by 1 (k'_2 =0), cell 2 sends order tag and requisition tag to store 1. At time 70 cell 2 completes production (z'_2=1, k'_2=1) and the third customer demand (arriving at time 70) is immediately satisfied. Then z'_2=0 (it decreases by 1), the PA card is sent to cell 2, k'_2 decreases by 1 (k'_2 =0), cell 2 sends order tag and requisition tag to store 1. At time 80: cell 1 completes production (z'_1=1, k'_1=1), store 1 sends the PA card to cell 1 (k'_1=0); cell 1 starts production which will end at time 110; store 1 supplies cell 2 with a part (z'_1=0); cell 2 starts production which will end at time 100. The fourth customer demand (arriving at time 80) will be satisfied by the product ready at cell 2 at time 100.

The parameter settings were chosen in such a way to demonstrate that each policy can have its own unique sequence of shipment times. If customer satisfaction was the only measure of performance for this system, the parameter setting of the LC policy resulted in the best solution, and the PTO policy in the worst one.

4.2.6 Effects of the PAC Parameters on the Control Procedure

In the previous section, we described the PAC parameter settings for different policies. Now, we discuss overall effects of the parameter choice on the control procedure.

4.2.6.1 Information and Material Flow

Limits on process tags and/or batching of PA cards impede the passing of information about customer demand to the upstream cells. We saw in the example of the hand simulation for Kanban (refer to 4.2.5.3), when cell 1, with a number of process tags limited to 1, did not know about customer demand arriving at time 60 till it completed production at time 70, and so forth. For cases with no limits on process tags and batch sizes equal to 1 ($k_i = \infty$, $r_i = 1$), as soon as there is a customer demand, all cells simultaneously receive a PA card and issue requisition tags. As a result, customer demand is transferred directly to the first stage of the production process and then the work is pushed through the consecutive stages.

A bottleneck cell i in the system with the BSS control will receive a large number of PA cards and will have large amount of work in process. As in the case of OPT, increasing the upstream z_i will only increase congestion at the bottleneck cell. If also the inventory carrying costs are higher at each consecutive stage (due to added by processing value), it will be desirable to pull products from the bottleneck cell. If there is a desired overall limit of the work-in-process, the control policies as IC or CONWIP are applied.

4.2.6.2 Feasibility Condition for Parameters *k* and *r*

An obvious condition that has to be satisfied is a relation between process tags and the batch size of PA cards, $k_i \geq r_i \geq 0$ ($i = 1,\ldots,m$). That is, the number of process tags must be at least equal to the PA cards batch size to allow forming of a PA cards batch by matching order tags with process tags.

During our study, another important condition, not discussed by Buzacott and Shanthikumar, related to parameters r and k has become more evident. First, let us explain the problem for a case of two consecutive production units, the upstream u and the downstream d, with r_u , k_u and r_d , k_d parameters respectively, schematically shown in Figure 4.5.

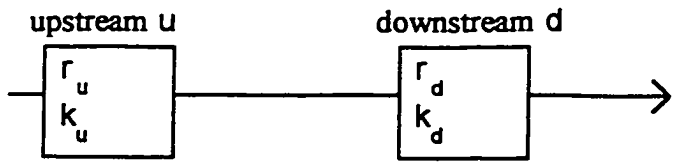


Figure 4.5 Upstream/downstream product schematic, case I.

The following relationship has to be satisfied to avoid production blocking:

$$k_d - r_d + \left\lfloor \frac{r_d}{r_u} \right\rfloor r_u \geq r_d \qquad [4.1]$$

To see this, consider the following. Store d sends a batch r_d of PA cards to the cell. This leaves (k_d-r_d) process tags in the store d. The r_d batch of PA cards results in production of $p=\lfloor r_d/r_u \rfloor r_u$ products (as batch r_d releases $\lfloor r_d/r_u \rfloor$ batches from u), which will eventually be sent together with p process tags to store d. In order to be able to send the next batch r_d of PA cards, the process tags, which are in the store d (k_d-r_d) plus the process tags released from the cell (p) must be at least equal to the batch size r_d.

To illustrate this, consider r_u =2, k_u =2, r_d =3 and k_d =3. The condition [4.1] is not satisfied as $3-3+\lfloor 3/2 \rfloor 2=2<3$. In this case, 3 PA cards arrive from store d to cell d. Upon arrival of these cards, cell d sends 3 order tags and 3 requisition tags for product u to store u; 2 order tags are matched with 2 process tags at store u, and 2 items of product u are sent to cell d after production process is completed at cell u. Cell d produces 2 items of product d and sends them with 2 process cards to store d. The next PA cards batch at store d, cannot be accommodated, as it requires 3 process tags and only 2 are available. The process is blocked. By increasing k_d to 4, the condition [4.1] is satisfied. One process tag remains at store d after sending 3 PA cards (r_d =3) to cell d. After processing 2 items of product d, cell d sends them to store d with 2 process tags. Those 2 process tags and the one remaining at store d, can be matched with arriving order tags to form the next PA cards batch at store d.

In the case of product u being a component for products $d_1, d_2, \ldots, d_q$ (refer to Figure 4.6), the number of PA cards sent from stores $d_1, d_2, \ldots, d_q$ cannot be less than the batch size of product u, as given by [4.2].

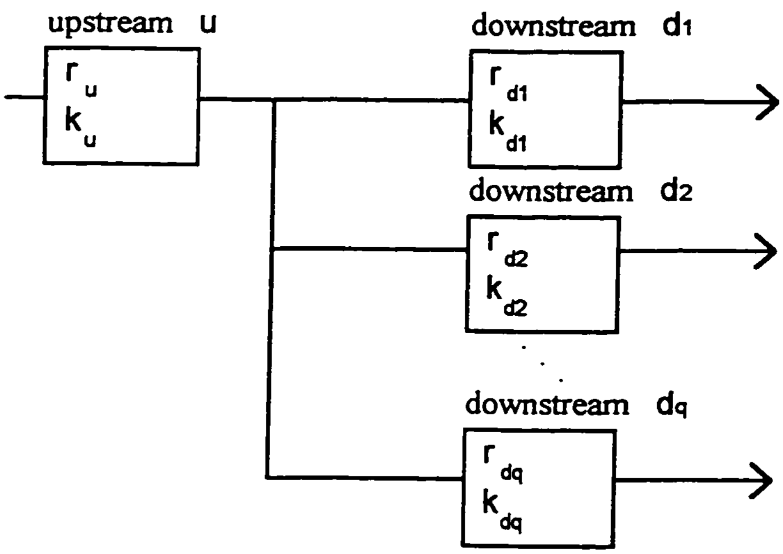


Figure 4.6 Upstream/downstream product schematic, case II.

$$\left\lfloor \frac{k_{d_1}}{r_{d_1}} \right\rfloor r_{d_1} + \left\lfloor \frac{k_{d_2}}{r_{d_2}} \right\rfloor r_{d_2} + \cdots + \left\lfloor \frac{k_{d_q}}{r_{d_q}} \right\rfloor r_{d_q} \geq r_u \qquad [4.2]$$

Note that condition [4.1] does not need to be satisfied between each single downstream cell and upstream cell. If the condition [4.2] is satisfied, the parameter setting is feasible. Again, let us illustrate with a numerical example, where $k_{d_1}=r_{d_1}=k_{d_2}=r_{d_2}=1$, and $k_u=r_u=2$. The single relationships between each of the downstream cells d_1 and d_2, and upstream cell u are not satisfied (each condition results in $1-1+\lfloor 1/2 \rfloor 2=0+0\text{x}2=0<1$). However, the condition [4.2] is satisfied ($\lfloor 1/1 \rfloor 1+\lfloor 1/1 \rfloor 1=2\geq 2$). Single one-item PA cards batches from d_1 and d_2, make the process on-going by generating an arrival of 2 order tags to store u and then accumulating the required two-item PA cards batch which can be sent to the upstream cell.

4.2.6.3 Parameter Choice for IC/CONWIP in Case of Complex Systems

For IC and CONWIP, when we have cells in series, the number of process tags for a stage is calculated as the sum of an initial inventory at this stage and the amount of process tags from the next (downstream) stage: $k_i=z_i+k_{i+1}$, $k_m=z_m$ (i=1,.....,m). Integral Control applied to the whole system is characterized by the total number of items in the system (N) equal to $\sum_{i=1}^{m} z_i$, and for CONWIP we should set $z_i^*=0$, $i=1,\ldots,m-1$, $z_m^*=\sum_{i=1}^{m} z_i$ in order to maximize customer satisfaction. The question arises as to how we can implement those policies to general systems.

The extension of Integral Control to a non serial assembly system can be implemented using the following choice of parameters:

1. $z_{ij}\geq 0$ $\quad$ i=1,.....,m, $\quad$ j=1,...,n.
2. $k_{ij}=z_{ij}+\sum_{l\in J(i,j)}\sum_{p\in P(i,j)} k_{lp}b_{lp}$ $\quad$ $i=1,\ldots,m-1$, $\quad$ j=1,...,n.

   $k_{mj}=z_{mj}$ $\quad$ j=1,...,n.
3. $r_{ij}=1$ $\quad$ i=1,.....,m, $\quad$ j=1,...,n.
4. $\tau_{ij}=0$, $\quad$ i=1,.....,m, $\quad$ j=1,...,n.

where $I(i,j)$ is a set of cells which use product j produced at cell i, $P(i,j)$ is a set of products produced at cell i, and b_{lp} is a number of items of product p belonging to the set $P(i,j)$ required at cell l belonging to the set $I(i,j)$.

CONWIP requires the following setting for each j (j=1,...,n): $z^{\bullet}_{ij} = 0$, $i = 1,\ldots,m-1$, $z^{\bullet}_{mj} = \sum_{i=1}^{m} z_{ij}$, and has for other parameters the same setting as for IC.

As there can be more then one component to make a certain product, and more than one item of each component can be required for processing, we have to show that each final product leaving the finished good storage will result in the supply of an equivalent amount of raw materials to the system. The total number of items in the system will be measured in terms of final product equivalents.

Each product j is controlled by parameters z_{ij}, k_{ij}, r_{ij} and τ_{ij}. Products processed in the same cells do not influence a calculation of parameters z_{ij} and k_{ij}. Thus each product can be seen as having its separate production line. A similar discussion to that given by Buzacott and Shanthikumar (1992b) can be applied here to demonstrate that this choice of parameters indeed results in the IC (CONWIP) policy. Having k_{mj}=z_{mj} assures that each order tag can be matched with a process tag and can generate a PA card, if and only if a final product is in store m. That means, a requisition tag, which comes together with an order tag, causes a final product to leave the system. At the same time the final product leaves the system, the order tag from a customer is transferred as a sequence of the PA card and order tag (or order tags, if the processing batch is more than 1) to each upstream cell and finally to the raw material store. The required number of raw materials will be supplied to the first cell of the production line of product j by the Raw Material Storage. If a final product demand arrives with no products in store, requisition is not processed and order tag cannot be matched with process tag. Thus WIP does not change. If product is in store and it leaves the system, all parts to make this product enter. Thus we have constant WIP in product equivalents.

We will now give an example to demonstrate that this choice of parameters indeed results in the IC policy. Figure 4.7 shows a manufacturing layout with 6 cells and 9 different products.

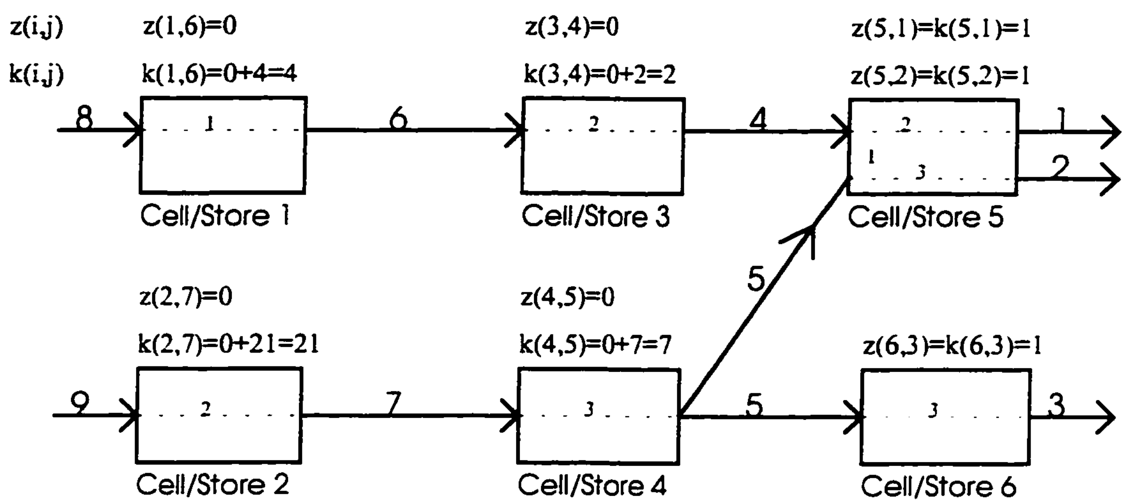


Figure 4.7 Example of IC/CONWIP parameter settings for a more complex layout.

Final product 1 is made in cell 5 from two items of product 4 and one item of product 5. Final product 2 is also processed in cell 5 and requires three items of product 5. Final product 3 is made in cell 6 from three items of product 5. Product 4 is processed in cell 3 from two items of product 6, and product 6 requires one item of raw material 8 for processing in cell 1. Product 5 is processed in cell 4 from three items of product 7, which is processed in cell 2 from two items of raw material 9. We set $z_{5,1}=k_{5,1}=1$, $z_{5,2}=k_{5,2}=1$, and $z_{6,3}=k_{6,3}=1$, as $k_{mj}=z_{mj}$, and $k_{mj}>0$. We assume that all initial inventories in stores 1 to 4 are equal to 0 ($z_{ij}=0$, i=1,...,4). Note that this special case of IC results in CONWIP. Then we calculate the values of k_{ij} in stores 1 to 4, according to the equation for the choice of parameters. The results of these calculations are indicated in Figure 4.7. When an order tag and a requisition tag for product 1 arrive to store 5, final product 1 leaves the system and PA card is sent to cell 5 by matching it with a process tag. At the same time, cell 5 issues 2 order tags for product 4 and one order tag for product 5, as one item of product 1 requires two items of product 4 and one item of product 5. Order tags from store to store will be immediately propagated into the raw material store, and it will result in the supply of four items of raw material 8 to cell 1 and six items of raw material 9 to cell 2. The case of the final product 2 or 3 leaving the system results each in immediate supply of 18 items of raw material 9 to cell 2. We can see that the total number of items in the system will always remain the same, that is equal to final products in stores 5 and 6 or their equivalent numbers in terms of BOM through the different stages of the process.

4.2.7 Performance Evaluation of PAC Systems

Buzacott and Shanthikumar (1992b) developed one approximation procedure to evaluate the average inventory, delivery performance and the production capacity for a special case of a PAC system with no time delay between order tags and requisition tags. Each cell of the system consists of a single machine. The processing times on the machine i are IID exponential random variables with mean $1/\mu_i$, $i = 1,\ldots,m$. There is an infinite supply of raw materials and the customer demand occurs according to a Poisson process with rate λ.

The basis of this approximation method is the recognition that, in the multiple-cell system, there are two types of process dynamics:

- Circulation through cell i. The main idea is based on representing the circulation of the PA cards, requisition tags and process tags by a cyclic queue with k_i customers and three service centers: store i where a process tag waits until it generates a PA card, store i-1 where a requisition tag waits to be filled, and cell i where the part undergoes the actual manufacturing process.

- Inventory position fluctuations at store i. As parts enter the store, the inventory level increases and, as parts leave the store, the inventory level decreases.

First the circulation through the cell was modeled as a Markov process. Then the changes in the inventory position were determined. Finally, both processes were interrelated with each other.

The performance measures of the approximation method are:

- Service Level (measured by the probability that an arriving demand is met immediately and by the average delay in meeting a demand);
- Inventory Levels (the average inventory of parts in stores and the average work-in-process inventory in cells);
- Throughput.

We will use Service Level and Inventory Levels as measures of performance in the PAC simulation model. In carrying out optimization studies, we will need to combine these measures of performance into one overall criterion using appropriate costs (refer to 5.6).

4.2.8 Impact of Choice of Parameters on the System Performance

Buzacott and Shanthikumar (1992b) analyze the system dynamics of the PAC system for the case of cells in series and describe the effect of the parameters (z, k, r, τ) choice on the performance of such a system. They show five properties of the departure process (and shipment to customers process).

Property 1. Departure times are decreasing in z_i , $i = 1,\ldots,m$.

Property 2. Departure times are decreasing in k_i , $i = 1,\ldots,m$.

Property 3. Departure times are increasing in r_i , $i = 1,\ldots,m$.

Property 4. Departure times are increasing and convex in τ_i when $c_i = 1$, $i = 1,\ldots,m$.

Property 5. Departure times are increasing and convex in $S^{(i)}$(= processing times) when $c_i = 1$, $i = 1,\ldots,m$.

Note: Departure times define times when parts leave the storage location to be delivered to the requesting cell, or times when parts leave the cell after processing has been completed.

Let L_k, k=1,2,.., be the shipment delay, that is, the time a customer waits for filling a requisition for a product. Then it follows that L_k will also have properties 1, 2, 3, and 5, but property 4 will become the following:

Property 4'. Shipment delays are increasing and convex in τ_i, $i = 1,\ldots,m-1$ but decreasing and convex in τ_m when $c_i = 1$, $i = 1,\ldots,m$.

This result also implies that the optimal delay times τ_i, will have $\tau_i = 0$, $i = 1,\ldots,m-1$ and τ_m as large as possible, if the system is aimed on the maximum customer satisfaction.

In chapter 8, we will examine the properties in the case of two manufacturing configurations, a simple cells in series layout and a more complex production process.

4.3 Concluding Remarks

The literature on manufacturing issues contains a considerable variety of approaches that are used. For those models and cases of manufacturing that have been tested, each have many assumptions and are often difficult to relate, one to another. What seems to be optimal for a certain configuration, will not necessarily perform so well in other conditions.

The PAC system, as a control policy for coordination of manufacturing processes in a multicellular environment, gives us a possible unification of existing production/ inventory control policies. It also opens possibilities for new directions of manufacturing research. One of the outcomes of this thesis is to contribute to a better understanding of each of the existing manufacturing policies, their differences and conditions for optimality.

To date, the simple models that have been tested, by other researchers, give insight into the basic working principles of the PAC system and some indications of parameter optimization. There are, however, still many research areas, which require more investigation and attention.

Some research issues are the following:

- The PAC parameter settings and their impact on the system performance for various production policies have been developed and tested mainly for the case of cells in series yielding only one product. More studies are required to check whether these relations will still hold in the case of a more general multicellular production layout with possibilities of processing and/or assembling of many different products.
- The models, which have been tested, assume exponential processing times on the machines and Poisson interarrival times for the customer demand as this allows the application of queuing and Markov process theory to develop the approximation calculation of the system performance in case of a very simple model. In practice, both assumptions are rather far from reality. It would be desirable to study the distribution dependence of system performance under various PAC parameter settings.
- Another pertinent question is how different demand processes influence the PAC system. The models assume that the arriving demand is only for one item of the needed

product at a time. This does not often happen in real-life situations. The most common practice is to request many items of the same product per order. Investigation of the effect of bulk arrivals is one of the things we look at in this thesis.

- MRP policy mainly proceeds on discrete time decisions. The PAC system operates on continuous time events. It would be desirable to check if the present parameter settings for the MRP coordination are indeed operating as a "real-life" MRP scheme.
- In the case of CONWIP the total level of inventory is fixed. What should be the value of the inventory level, and how does it impact on system performance?
- Requisitions tags wait in a queue at the product store until there is a unit of product available and higher priority requisition tags in the queue have been met. Similarly, in the case of PA cards which are waiting in a queue at the cell and authorize production, they are removed from the queue according to some rules. It is possible for more than one cell to supply a given store, which introduces the need to specify which cell the PA card would be sent to once it is generated. Again, there can be a number of different policies for this, such as cyclic or random assignment, or relative costs. In cases where multiple cells can manufacture a component, priority choices also arise. Some of these control policies may violate the idea of decentralized information and control. It would be desirable to study the role of these decisions in a PAC system.
- It has been assumed that the cell management is fully responsible for internal coordination of operations in the cell and does its work effectively. A small workstation with one machine can function as a cell, or a large workshop with many machines as well. Different activities in other cells influence tasks and responsibilities of each cell. Cells can be automatically controlled locations without operators and units where processing highly depends on human labor. All those aspects influence the

decision that the cell management makes. It would be useful to study the role of cell management and its effect on the total PAC system.

- Are the present PAC parameter settings for various coordination policies completely adequate and sufficient? Can we, by using only those parameters, clearly define all traditional production control schemes? More detailed studies of different coordination policies operating in different manufacturing settings should be done in the future.

Chapter 5

SIMULATION WITH SIMLIB OF MULTIPLE-CELL SYSTEMS COORDINATED BY PA CARDS (PAC SYSTEMS)

5.1 Introduction

In this chapter, we describe the primary features of a simulation program for a multiple-cell multiple-product PAC controlled system. The purpose of the program is to provide a performance evaluation tool for a manufacturing system coordinated by a PAC system. We have chosen to implement this simulation by modifying the SIMLIB routines of Law and Kelton (1991).

Before presenting the main characteristics of the simulation model using an example configuration, we briefly describe SIMLIB and outline assumptions and conditions of the manufacturing model. After program routines, we describe measures of performance and costing scenarios included in the simulation. We also summarize some general modeling issues, with emphasis on assumptions and simplifications, and indicate where different approaches could be implemented. We discuss the statistical analysis of output data and the validation results for our model. Finally, we close with some conclusions.

5.2 Simulation Language, SIMLIB

SIMLIB is a set of FORTRAN support routines, which take care of some standard simulation tasks, such as, maintaining queues as lists, processing the event list, accumulating statistics, generating random numbers and observations from a few distributions, and reporting out results. This language implements the concept of linked storage allocation between records stored in one REAL storage array MASTER. Columns of the MASTER array contain the different attributes of the records, which can be grouped in different lists. For the detailed description of SIMLIB, refer to Law and Kelton (1991, pages 133-150).

In the Law and Kelton version of SIMLIB, the MASTER array has fixed dimensions. For a complex manufacturing system, these array dimensions will usually be not large enough. Therefore, we modified the SIMLIB version reported in Law and Kelton to enable the simulation of a large system by replacing predefined limits with regard to number of records, lists, and statistical accumulators by parameters which can be set to desired values in a parameters declaration file. Appendix A1 contains the FORTRAN code of SIMLIBG (our general version of SIMLIB) that was used for the PAC simulation.

5.3 Problem Statement

We consider a manufacturing system, which consists of:

- m - cells;
- r - different raw materials;
- n - different products;
- c_i - identical machines (servers) per each cell i.

Our current model represents the manufacturing systems as consisting of cells, each capable of processing certain products and delivering them to their product store. We represent the cell as consisting of c_i identical parallel machines. Although this is admittedly a simplified view of a cell, the focus of this thesis is on the material and information flow between cells rather than on inter cell issues.

Raw materials are parts and components supplied to the cells from the Raw Materials Storage, which is exogenous to the manufacturing system. We assume that parts required from the Raw Materials Storage are always in stock.

Each distinct product has one unique cell, where it is produced and one unique storage location, where it is kept after being produced.

Among all *n* products, it is useful to make a distinction of:

- *f* - final products;
- *a* - assembly products;
- *fa* - final/assembly products;

with $n = f + a + fa$.

Final products are goods which are delivered only to the customers. They leave the manufacturing system. *Assembly* products are intermediate products, which are delivered to other cells for further processing as a component either of a final product or another assembly product. *Final/assembly* are products which are final products for a customer and can also be used by a cell for processing into another product.

Each product is uniquely characterized by:

- production cell;
- store;
- a probability distribution of processing time at the production cell;
- number of components from which it is made;
- names of those components;
- number of each component required.

In defining the simulation model, it is useful to develop the idea of a production unit. We think of a unit as made up of a cell plus all the stores for products made in that cell. Each product then is associated uniquely with a production unit. We assume that the system has m units and, since it is possible that more than one product can be made in a cell, then we must have $m \leq n$.

For each product j, we specify the following 4 PAC parameters with regard to its production unit:

- z_j - initial/terminal inventory level at the storage location;
- k_j - number of process tags, which will eventually become PA cards;
- r_j - PA cards batch size;
- τ_j - time lag between receiving an order tag and a requisition tag at the storage location.

The SIMLIB simulation of the manufacturing system is an event simulation based on the movement of different tags (order tags, requisition tags, PA cards, process tags) as well as on the movement of different product items. We will assume that the cell management coordinates the internal operations effectively and every production authorization results in a part with an acceptable quality. Because of our assumptions of 100% production yield and no disassembly operations, cancellation notes, order cancellations and surplus tags, are not required.

The state of the system is modeled by the number of tags and parts in different queues, as well as busy/idle indicators from the various cells. The queues are managed as different lists in the MASTER array of SIMLIB. Each tag or part form a record, which is filled in a designated order characteristic at the queue.

In the PAC simulation model, each production unit in the manufacturing system has six unique queues associated with it:

- 4 for different tags:
  - ORD - for order tags in stores;
  - REQ - for requisition tags in stores;

- PROC - for process tags in stores;
- PAC - for PA cards in cells;

- 2 for product items:
  - PROD - parts in stores;
  - WIP - component parts awaiting processing within cells (work-in-process storage).

In order to make it easy to calculate delays, and hence inventory costs per product, it is convenient that each product be also separately registered in the following queues:

- $PPRD_j$ - parts in storage per product type j;
- $PWIP_{ij}$ - parts in a work-in-process storage at cell i per product type j.

Additionally, the PWIP queues are necessary to account for any time delays in moving assembly products and raw materials from their store to the cell. Only when all required components to make a product exist in the PWIP queues of that product's cell, can production of the product begin. The PPRD queues provide no extra information but are convenient for statistical purposes.

Figure 5.1 shows the schematic of different queues per each unit.

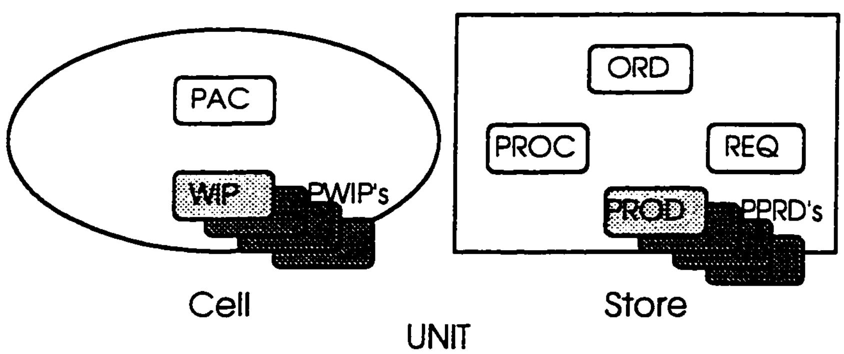


Figure 5.1 Storage schematic.

Each product type ($f+fa$) has an additional list (queue), the CUST queue, which is convenient to gather statistics about the delay of a product to a customer. When a product item is required by a customer (a requisition tag arrives from a customer to the storage location), then an entity enters this queue and stays there until the required item is delivered to a customer.

In order to differentiate the cost from the time an order appears until the requisition arrives, additional lists are required for product types ($f+fa$). Those lists are the CORD queues. Generally, the CORD queue registers similar data as the CUST queue, but from the time of the arrival of an order tag. The function of CUST and CORD lists will become more clear, when we describe the performance evaluation aspects of the PAC simulation model (refer to 5.6).

For the most part, we model customer arrivals as a compound Poisson process (see section 7.9 where we examine bulk arrivals). Order tags for *final* and *final/assembly* products arrive at the system according to a Poisson process with mean MARRVT. Each arrival can request one of $f+fa$ product types. There is a probability p_j that an order tag is for product j ($j=1,...,f+fa$).

To help explain our simuiation model, consider an example manufacturing configuration. This system consists of 4 production units (cell and stores), and each cell consists of 1 machine. The manufacturing system can process 4 final or final/assembly products plus 7 intermediate or raw material products. Arriving order tags are for products 1, 2, 3, and 4 with respective probabilities p_1, p_2, p_3, and p_4. Each final or intermediate product has a deterministic setup time s_j, the time required to adjust the machine when starting production of a new batch. Each product that can be delivered for processing to the cell, has a transportation time t_{ij}, the deterministic time product j requires for transportation from storage location to cell i.

Figure 5.2 shows the configuration layout of our example-model and indicates the final, assembly, and raw materials products with their required number of items for processing and routing. Note that the product numbering is backwards from Buzacott and Shanthikumar's cell numbering (refer to Figure 4.1), where m is last. For the convenience of the computer program, we chose product numbering starting with final, final/assembly, assembly, and raw materials. That means that lower numbers of products indicate that products are "closer" to the end of the production process, and the largest product numbers are for raw materials.

Table 5.1 gives the products, their production cells and their assembly components. The processing times $1/\mu_j$, the mean interarrival time MARRVT of customer orders, the setup times s_j, the transportation times t_{ij}, and the probabilities p_j of a particular order being for product j are data that we treat as model input. Table 5.2 indicates the product numbering. In this example, we have m=4 units, n=7 products, r=4 raw materials, and c_i=1 machines at each cell.

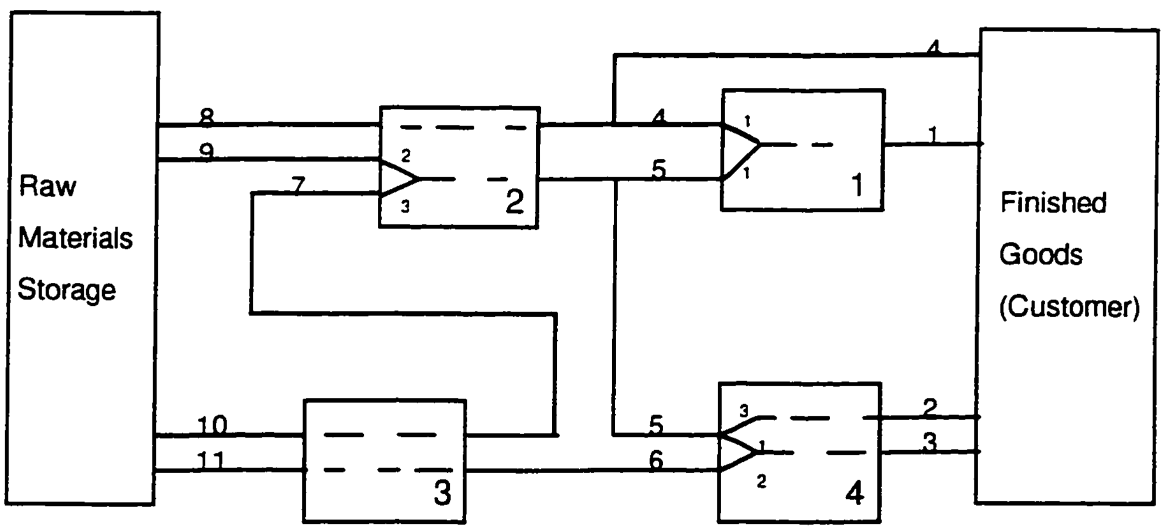


Figure 5.2 Example manufacturing system with 4 units and 4 final products.

Table 5.1 Processing data

Product type	Production cell	Mean process. time	# of components	Names of components	# of items per each component	Probability	Setup time
1	1	$1/\mu_1$	2	4,5	1,1	p_1	s_1
2	4	$1/\mu_2$	1	5	3	p_2	s_2
3	4	$1/\mu_3$	2	5,6	1,2	p_3	s_3
4	2	$1/\mu_4$	1	8	1	p_4	s_4
5	2	$1/\mu_5$	2	7,9	3,2	-	s_5
6	3	$1/\mu_6$	1	11	1	-	s_6
7	3	$1/\mu_7$	1	10	1	-	s_7

Table 5.2 Product types

Product type kind	Number of different product types	Product type number
final	3	1,2,3
final/assembly	1	4
assembly	3	5,6,7
raw material	4	8,9,10,11

5.4 PAC SIMLIB Program

The PAC simulation model can represent quite complex manufacturing systems through a data set similar to that in Tables 5.1 and 5.2. What is interesting is that the simulation is reasonably easy since the program needs only to recognize 8 types of events, one of which is the end of simulation event. These events are indicated in Table 5.3.

Table 5.3 Events in the PAC simulation model

Event Description	Event Type
Arrival of an order tag for product *j* to store *i* from customer	1
Arrival of a requisition tag for product *j* to store *i*	2
Arrival of PA card for product *j* to cell *i*	3
Arrival of a product *j* to cell *i*	4
Departure of a product *j* from cell *i*	5
Arrival of a product *j* with a process tag to store *i*	6
End of the simulation	7
Arrival of an order tag for product *j* to store *i* from cell	8

The PAC model does, however, need quite a large number of lists. As indicated in 5.3, there are 6 queues for each production unit, a total of $6*m$ lists. Additionally we have the $2*(f+fa)$ lists to keep track of customer waiting time. Each product j is recorded in the PPRD list and products and raw materials delivered to cells are recorded in PWIP lists. Obviously, since not all products are likely assembly inputs in every cell, the number of such lists will be much less than its potential $m*(fa+a+r)$. If we let nwq be the number of PWIP lists, in our example nwq=9. Summarizing the above, we have the following lists:

- production unit queues: $6*m$;
- customer waiting time queues: $2*(f+fa)$;
- parts in storage queues, PPRD lists: $1*n$;
- parts in work-in-process, PWIP lists: nwq.

The lists and the attributes of their records form the SIMLIB array MASTER. The list structure for our example-model is given in Table 5.4. Most of the terms used there correspond to the descriptions above. By "address" we mean the number of the cell, where the product has to be delivered (used only for event type 2 and 4). The "Yes/No process tag" is set to 1 or 0 and indicates a match of an order tag with a process tag.

Table 5.4 List structure for the example-model

List	Attribute 1	Attribute 2	Attribute 3	Attribute 4
1 ORD -unit 1	TA of order tag	Product type	Yes/No process tag	
2 REQ-unit 1	TA of requisition tag	Product type	Address	
3 PROD-unit 1	TA of product	Product type		
4 PROC-unit 1	TA of process tag	Product type		
5 WIP-unit 1	TA of product	Product type		
6 PAC-unit 1	TA of PA card	Product type		
7 -12	similar to list 1-6, but for unit 2			
13-18	similar to list 1-6, but for unit 3			
19-24	similar to list 1-6, but for unit 4			
25 CUST-product 1	TA of requisition tag from customer			
26 CUST-product 2	TA of requisition tag from customer			
27 CUST-product 3	TA of requisition tag from customer			
28 CUST-product 4	TA of requisition tag from customer			
29 CORD-product 1	TA of order tag from customer			
30 CORD-product 2	TA of order tag from customer			
31 CORD-product 3	TA of order tag from customer			
32 CORD-product 4	TA of order tag from customer			
33 PPRD-product 1	TA of product	Product type		
34 PPRD-product 2	TA of product	Product type		
35 PPRD-product 3	TA of product	Product type		
36 PPRD-product 4	TA of product	Product type		
37 PPRD-product 5	TA of product	Product type		
38 PPRD-product 6	TA of product	Product type		
39 PPRD-product 7	TA of product	Product type		
40 PWIP-unit 1, product 4	TA of product	Product type		
41 PWIP-unit 1, product 5	TA of product	Product type		
42 PWIP-unit 2, product 7	TA of product	Product type		
43 PWIP-unit 2, product 8	TA of product	Product type		
44 PWIP-unit 2, product 9	TA of product	Product type		
45 PWIP-unit 3, product 10	TA of product	Product type		
46 PWIP-unit 3, product 11	TA of product	Product type		
47 PWIP-unit 4, product 5	TA of product	Product type		
48 PWIP-unit 4, product 6	TA of product	Product type		
49 Event List	Event time	Event type	Product type	Address

TA = time of arrival

Once we know *f*, *fa*, *a*, *r*, *n*, *m*, calculations of relative positions of the lists are straightforward. The event list is always the last list in our set of lists, and is list number

$$m*6 + 2*(f+fa) + n + nwq + 1 \qquad [5.1]$$

For the example-model the event list is list 49.

Within the PAC simulation, we collect a number of sample statistics (discrete-time data), as average delays in queues and time statistics (continuous-time data), as the time-average number in queue and the proportion of time the machine is busy. Using Law and Kelton's acronym, we refer to these as SAMPST and TIMEST variables, respectively. The estimator of an average delay in queue $\hat{d}(w)$ is a discrete-time statistics, since it is defined based on the collection of random variables denoting product delays $\{D_1, D_2, \ldots, D_w\}$ that have a discrete time index, l=1,...,w. From a single simulation run, we estimate the expected average delay in queue of w products as follows:

$$\hat{d}(w) = \frac{\sum_{l=1}^{w} D_l}{w} \qquad [5.2]$$

The time-average number in queue (not counting the one being served), $\hat{q}(w)$, is an example of continuous-time statistics, since it is based on the collection of random variables denoting the number of products in queue at time t, $\{Q(t)\}$, each of which is indexed on the continuous time parameter $t \in [0,\infty)$. If we denote by $T(w)$ the time required to make our observations, then the estimator of the expected time-average number in queue can be expressed as follows:

$$\hat{q}(w) = \frac{\int_0^{T(w)} Q(t)dt}{T(w)} \qquad [5.3]$$

Table 5.5 and 5.6 summarize statistical variables for a PAC simulation model. The SAMPST variables are collected from the entities traversing the relevant queues. The delays of the product in queues are used in different ways. We want to know the average delay in the WIP queue for each machine group (regardless of the product type). For our example, SAMPST variable 1 through 4 will be used for these average delays (as m=4). We also want to know the average delay in all the WIP queues visited by each product type (regardless of machine group). For this, our example model will require SAMPST variables 5 through 15 (as $n+r$=11). The delays in CUST and CORD queues indicate the average waiting time of a customer for a product since sending, respectively, a requisition tag and an order tag. In our example-model, there are 4 product types (f+fa). It results that SAMPST variables 16 through 19 will be used for delays in CUST queues, and variables 20 through 23 will be used for delays in CORD queues.

Table 5.5 SAMPST variables for the PAC model

Variables			Meaning
1	-	m	Delay in queue at machine in cell i (WIP -i queue)
m+1	-	$m+n+r$	Delay in WIP-queue for product type j (j=1,...,$n+r$)
$m+n+r$+1	-	$m+n+r+f+fa$	Delay in CUST-queue of delivering product type f or fa to customer, calculated from the time of arrival of a requisition tag
$m+n+r+f+fa$+1	-	$m+n+r$+2*(f+fa)	Delay in CORD-queue of delivering product type f or fa to customer, calculated from the time of arrival of an order tag

Table 5.6 TIMEST variables for the PAC model

Variables			Meaning
1	-	m	Number of machines busy in cell i

The machine utilization is defined as the time-average number of machines that are busy in the cell, divided by the total number of machines in the cell. To find the average number of busy machines in a cell, we keep an array NBUSY(I), the number of machines currently busy in cell I, and TIMEST is called whenever this changes for any cell. Our example uses TIMEST variables 1 through 4 to keep track of "busy/idle indicators" changes of machine groups per each cell.

For continuous-time statistics, we also use the SIMLIB subroutine FILEST. SIMLIB treats the number of records in a list as a continuous-time function, whose values may rise or fall only at the time of events.

The random number generator RANFN(ISTRM) included in SIMLIB has built-in default seeds for producing 100 separate "streams" of random numbers (refer to Appendix A1 and comments in the code of the generator for its use). A stream is a sub-segment of the random numbers produced by the generator, with one stream beginning where the previous stream ends. Streams can be seen as independent generators and they allow the user to assign a certain stream to a certain source of randomness in the simulation. To use a stream of random numbers means to generate a sequence of identical random variables for each new simulation run. Using different streams for different purposes facilitates reproducibility and comparability of simulation results. For our model, there are three types of random variables needed, to which we assign the following streams:

- stream 1: interarrival times
- stream 2: product types
- stream 3: processing times

Fixed streams 1 and 2 assure the same characteristics of the arrival process for each new simulation run regardless of the PAC parameters. Stream 3 is used to generate the processing times for all processing jobs, regardless of type. For simulation runs with different PAC parameter values, the sequence of processing times for products and machines will also be different. In some cases we might want to dedicate a separate stream to generate the processing times per each product type or each machine group in order to control the exact characteristics of each processing job or each machine group.

5.5 Description of Program Routines

As in any event driven simulation, the program consists of a main routine, an initialization routine, some general utility routines and most importantly the routines to process each event. Table 5.7 indicates the basic structure. For a listing of all programs and declaration files, as well as for a definition of all variables used for the simulation program, refer to Appendix A2. An event graph for the PAC model is given in Figure 5.3.

Table 5.7 Subprograms list for PAC simulation

Subprogram	Purpose
SIMLIBG	Set of subroutines of SIMLIB adjusted for PAC model.
ORDARR(NEW)	Processes arrival of an order tag, where NEW=1 if this is a new order from a customer, type 1 events, and NEW=2 if the order tag comes from a cell (called by PACARR), type 8 events; NEW is an INTEGER.
REQARR	Processes type 2 events (see table 5.3)
PACARR	Processes type 3 events (see table 5.3)
WIPARR	Processes type 4 events (see table 5.3)
DEPART	Processes type 5 events (see table 5.3)
PPARR	Processes type 6 events (see table 5.3)
REPORT	Generates report, called when the simulation ends (type 7 event).

The description of the main tasks, done by subroutines for processing events 1 through 8, is as follows:

(1) Order Tag Arrival Routine (ORDARR(NEW))

Subroutine ORDARR(NEW), flowcharted in Figure 5.4, serves to process type 1 and 8 events, which are the arrivals of order tags from customers (1) and cells (8) events. This routine can trigger three events, the arrival of order tags from customers, processed by the routine ORDARR with NEW=1, the arrival of requisition tags, processed by the routine REQARR, and the arrival of PA cards, processed by the routine PACARR.

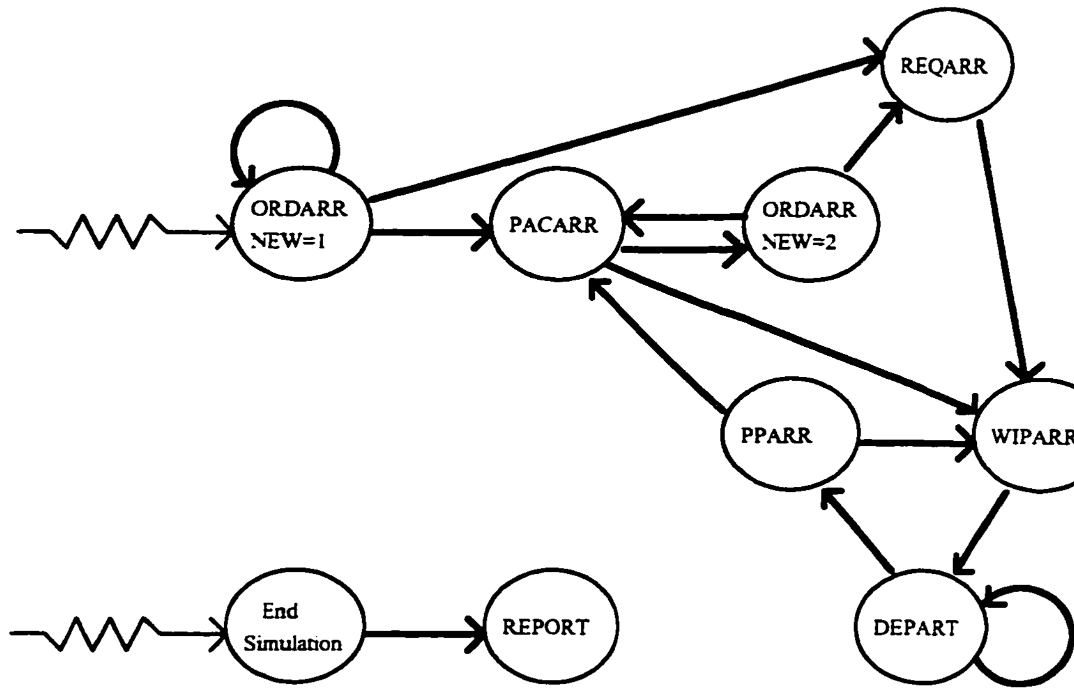


Figure 5.3 Event graph, PAC simulation model.

The subroutine begins by checking NEW to determine whether it is being used as an event routine type 1 to process an arrival of an order tag from a customer for a "final" product type (NEW=1), or whether it is being used as an event routine type 2 to process an arrival of an order tag from a cell for an "assembly" product type (NEW=2). If this is a new "arrival", the next order tag arrival is scheduled, and the arriving customer demand is registered by placing a record at the end of a CORD-queue for the required product. Regardless of whether the order tag is from a customer or from a cell, the routine continues by determining the store number to which the order tag arrives. Each arriving

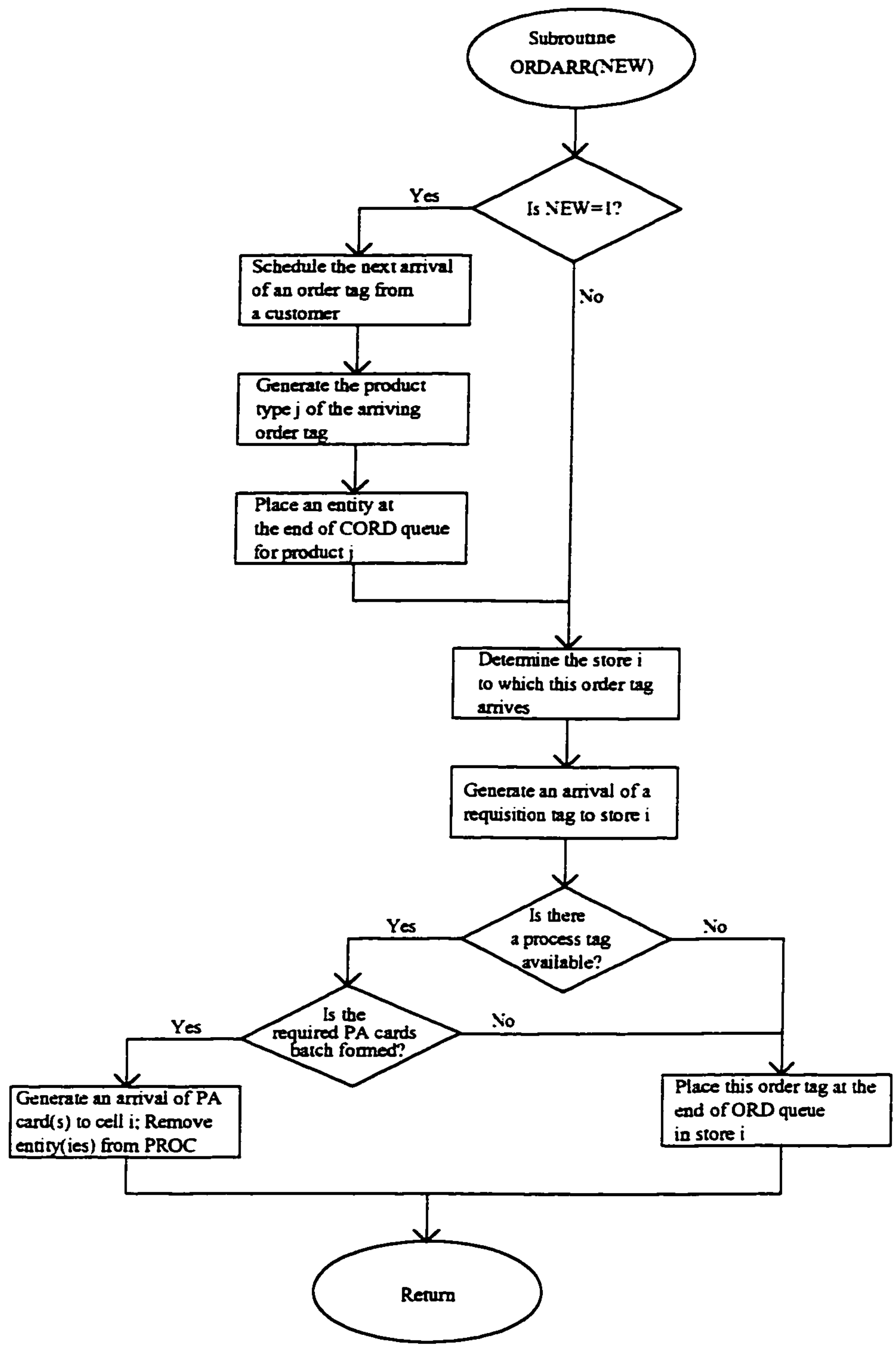


Figure 5.4 Flowchart for ORDARR(NEW) routine.

order tag generates an arrival of a requisition tag event for the same product (event type 2). If the arriving order tag can be matched with a process tag and also forms a required r_j batch size of PA cards, then the PA cards arrival to the cell event is scheduled (event type 3).

(2) Requisition Tag Arrival Routine (REQARR)

Subroutine REQARR, flowcharted in Figure 5.5, serves to process a type 2 event, the arrival of requisition tags event. This routine triggers the arrival of products to cells event processed by the routine WIPARR.

The routine starts with an identification of a product type for which a requisition tag arrives, a store number to which a requisition tag arrives and a cell number or customer ("address") which issued the requisition tag. If the arriving requisition tag is for a "final" product, the arriving customer demand is registered by placing a record at the end of a CUST-queue for the required product. If the required product is available at the storage location, then the product is removed from PROD and PPRD queues and its arrival to the requiring cell is scheduled. In case of a "final" product type, the product is delivered to a customer (it leaves the system) and the first record from the CUST-queue and from the CORD-queue for this product is removed. In case of an "assembly" product type, an arrival of this product to the requesting cell is scheduled (event type 4) including the transportation time. If the required product is not available at the store, then the arriving requisition tag is placed at the end of the REQ-queue.

(3) PA Card Arrival Routine (PACARR)

Subroutine PACARR, flowcharted in Figure 5.6, serves to process a type 3 event, the arrival of PA cards event. This routine triggers the arrival of order tags to stores event processed by the routine ORDARR with NEW=2, that is type 8 events.

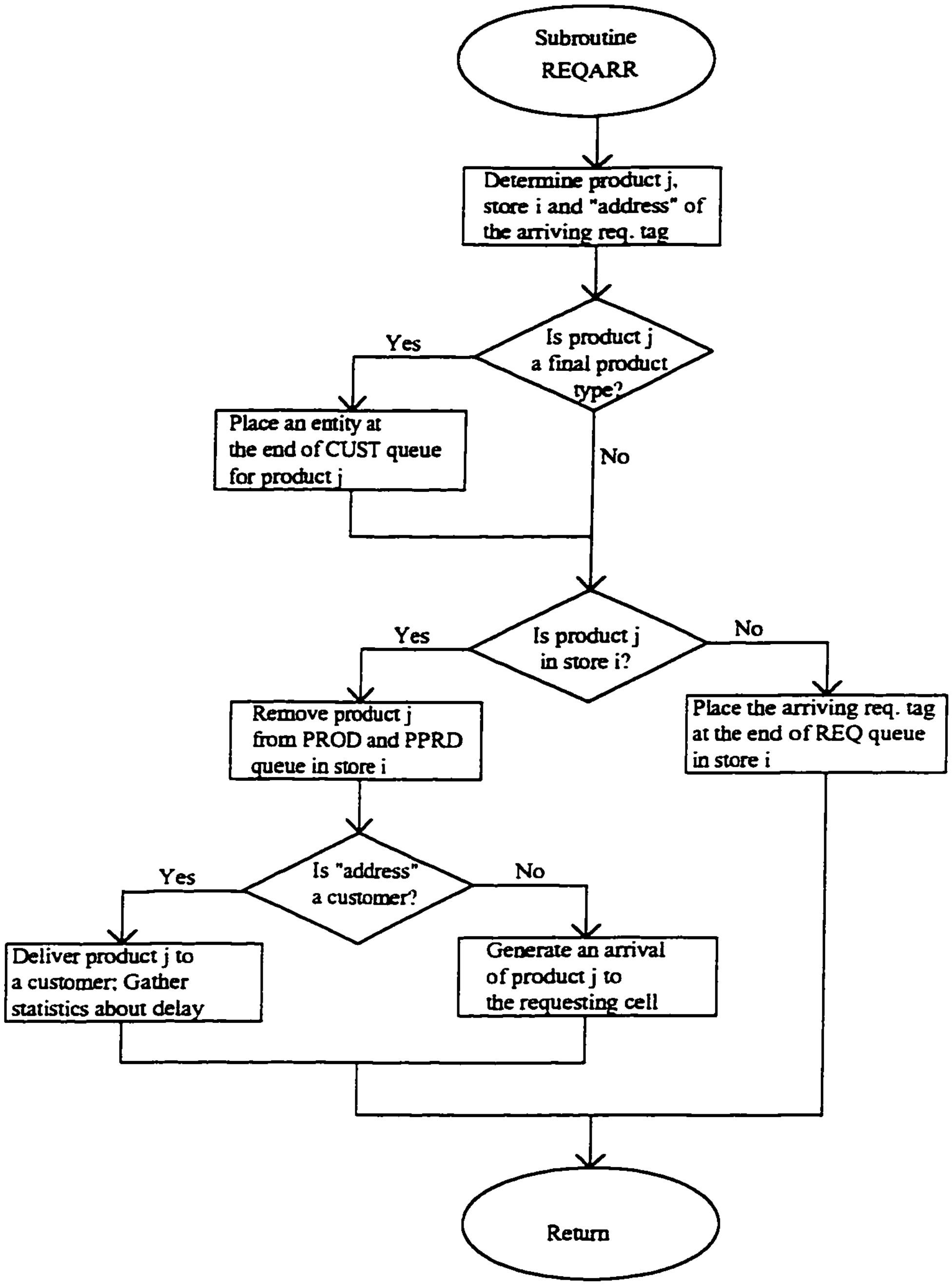


Figure 5.5 Flowchart for REQARR routine.

The routine starts with an identification of a product type for which a PA card arrives, and a cell number to which the PA card arrives. The arriving PA card is placed at the end of the PAC-queue in the cell and it generates the arrival of order tags for all components needed for processing of a product type. One PAC can generate one or more arrivals of order tags to one or more different stores following the product specification given by the components and number of items of those components. If a component is an "assembly" product, the subroutine ORDARR(2) is called. If a component is a "raw material", the arrival of a raw material to the cell event is scheduled (event type 4) including the transportation time.

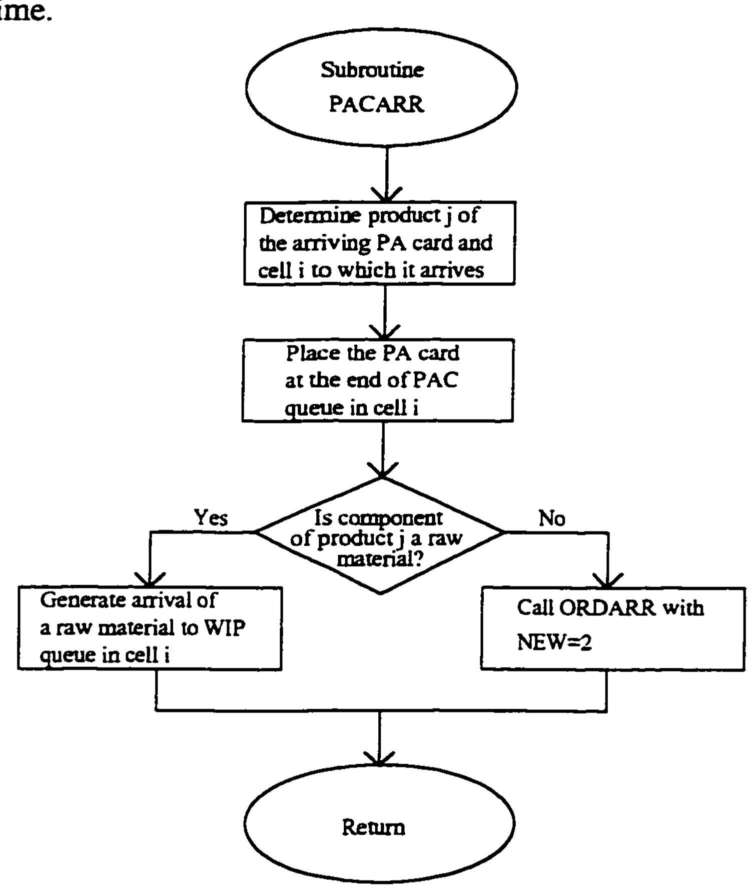


Figure 5.6 Flowchart for PACARR routine.

(4) Product to Cell Arrival Routine (WIPARR)

Subroutine WIPARR, flowcharted in Figure 5.7, serves to process a type 4 event. the arrival of raw material or assembly products to a work-in-process storage event. This routine triggers the departure of products and process tags from cells to stores event processed by the routine DEPART.

The routine starts with an identification of an arriving product type and a cell number to which this product arrives. The arriving product is placed at the end of the WIP and PWIP queues, and the total number of units of this product in this cell is increased by 1. If all machines at this cell are busy, the routine exits. If there is a machine free, the processing can begin for the first product in the PAC-queue for which all required components have been already delivered to the WIP-queue of this cell. For all cards in the PAC-queue, each PA card is removed; a check for components is made, and the PA card is again placed at the end of the PAC-queue. If there is no product in the PAC-queue for which all components are in the WIP-queue, the routine exits. If there is such a product, then the machine continues processing, the end of the processing is scheduled and a departure of the product after processing is scheduled (event type 5). All components required for processing are removed from WIP and PWIP queues. The total number of units of each component in this cell is decreased by the amount used for processing. If the item being processed is different than the last processed item at the machine, then a setup is carried out and the setup time is added to the processing time.

(5) Product from Cell Departure Routine (DEPART)

Subroutine DEPART, flowcharted in Figure 5.8, serves to process a type 5 event, the departure of products after processing from cells event. This routine can trigger two events, the arrival of products with process tags to stores event, processed by the routine

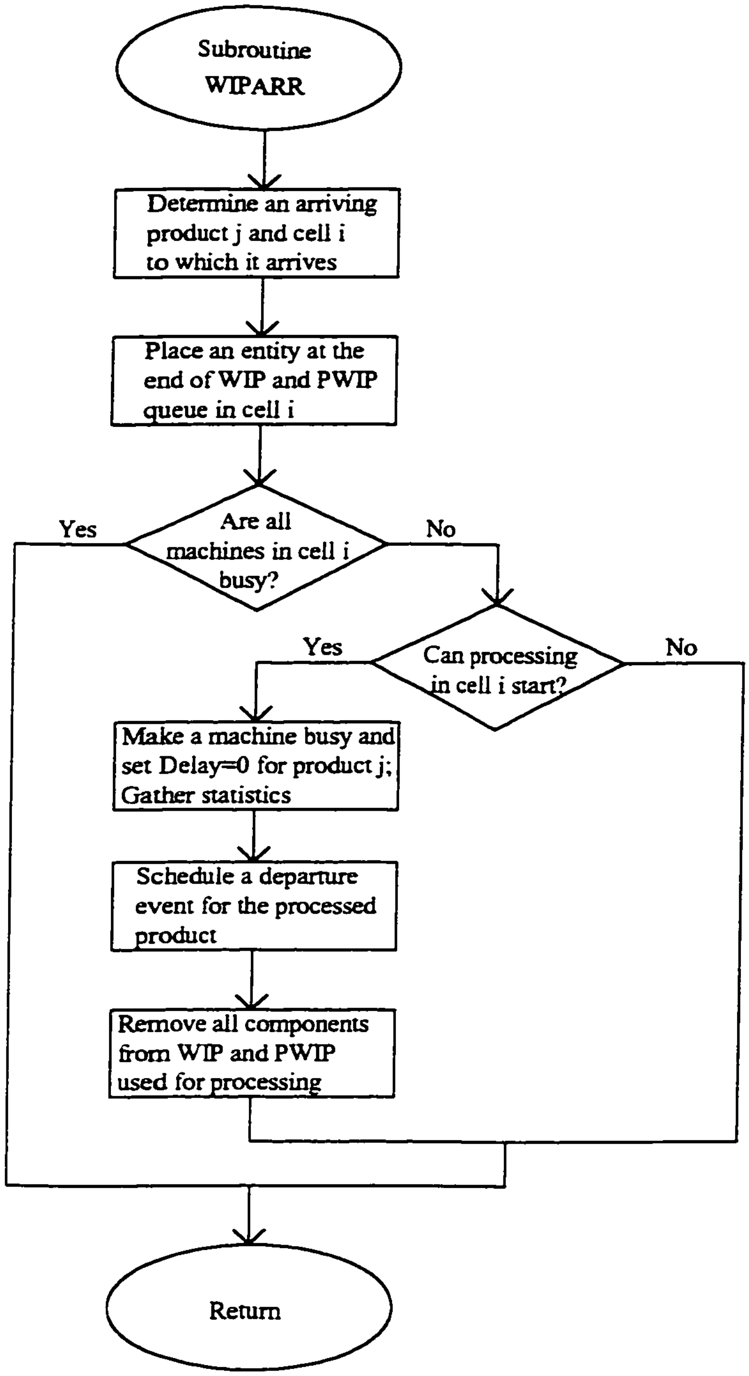


Figure 5.7 Flowchart for WIPARR routine.

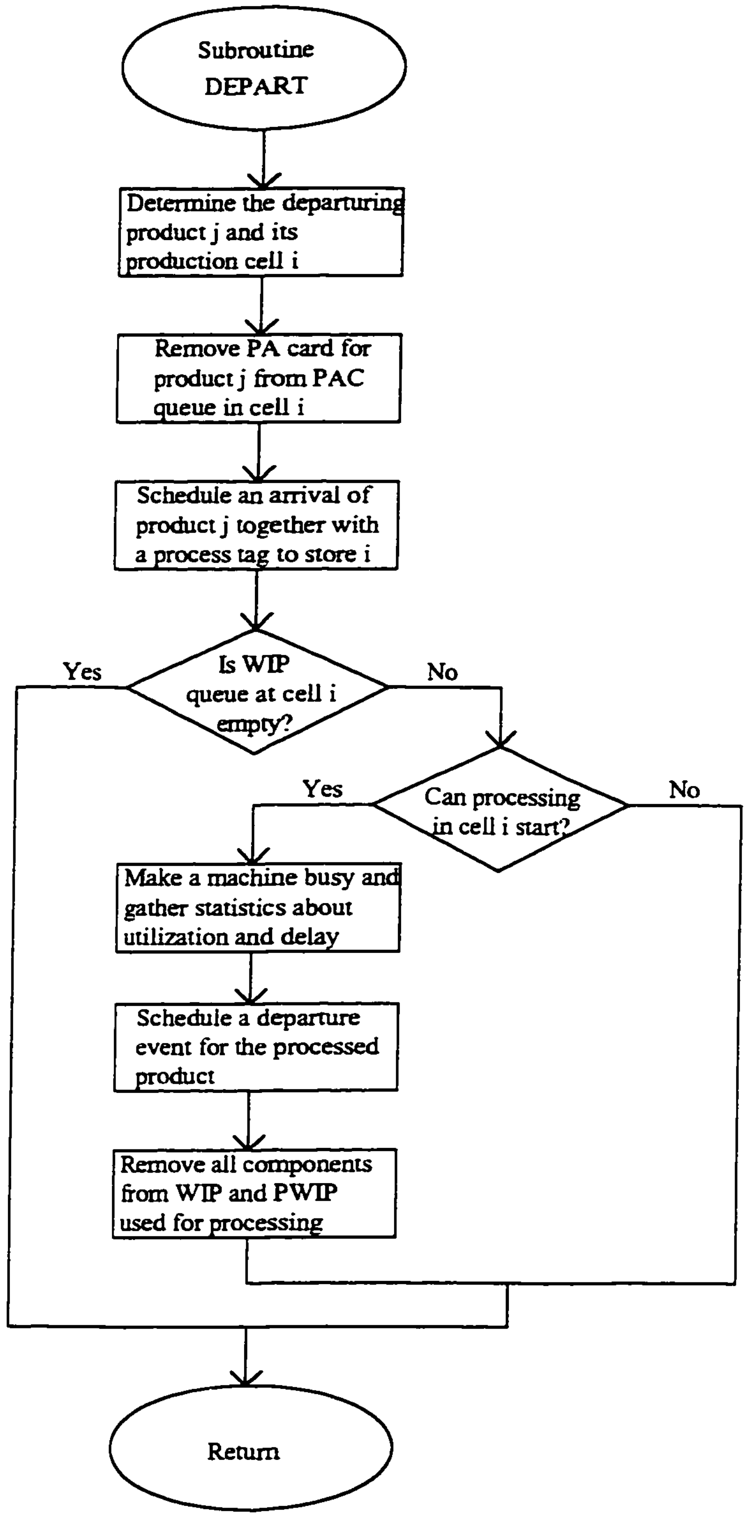


Figure 5.8 Flowchart for DEPART routine.

PPARR, and the departure of products from cells to stores event, processed by the routine DEPART.

The PA card for the departing product is removed from the PAC-queue and an arrival of this product, with a process tag, to the store is scheduled (event type 6). Product departing from a cell makes a machine available for processing. If the WIP-queue in front of the machine is empty, the machine becomes idle and the routine ends. If the WIP-queue is not empty, the search for the next product in PAC-queue begins. For all cards in the PAC-queue, each PA card is removed; a check for components is made, and the PA card is again placed at the end of the PAC-queue.

If there is no product in the PAC-queue for which all components are in the WIP-queue, the machine becomes idle and the routine ends. If there is such a product, then the machine continues processing, the end of the processing is scheduled and a departure of the product after processing is scheduled (event type 5). All components required for processing are removed from WIP and PWIP queues. The total number of units of each component in this cell is decreased by the amount used for processing. If the item being processed is different than the last processed item at the machine, then a setup is carried out and the setup time is added to the processing time.

(6) Product/Process Tag Arrival Routine (PPARR)

Subroutine PPARR, flowcharted in Figure 5.9, serves to process a type 6 event, the arrival of products together with process tags to stores after processing at cells event. This routine can trigger two events, the arrival of PA cards event, processed by the routine PACARR, and the arrivals of products to cells event, processed by the routine WIPARR.

A product's leaving the cell, after processing, removes the PA card from the PAC-queue. This PA card is transformed into a process tag and arrives at the store together with the product. The routine PPARR deals with two "arrivals" which occur

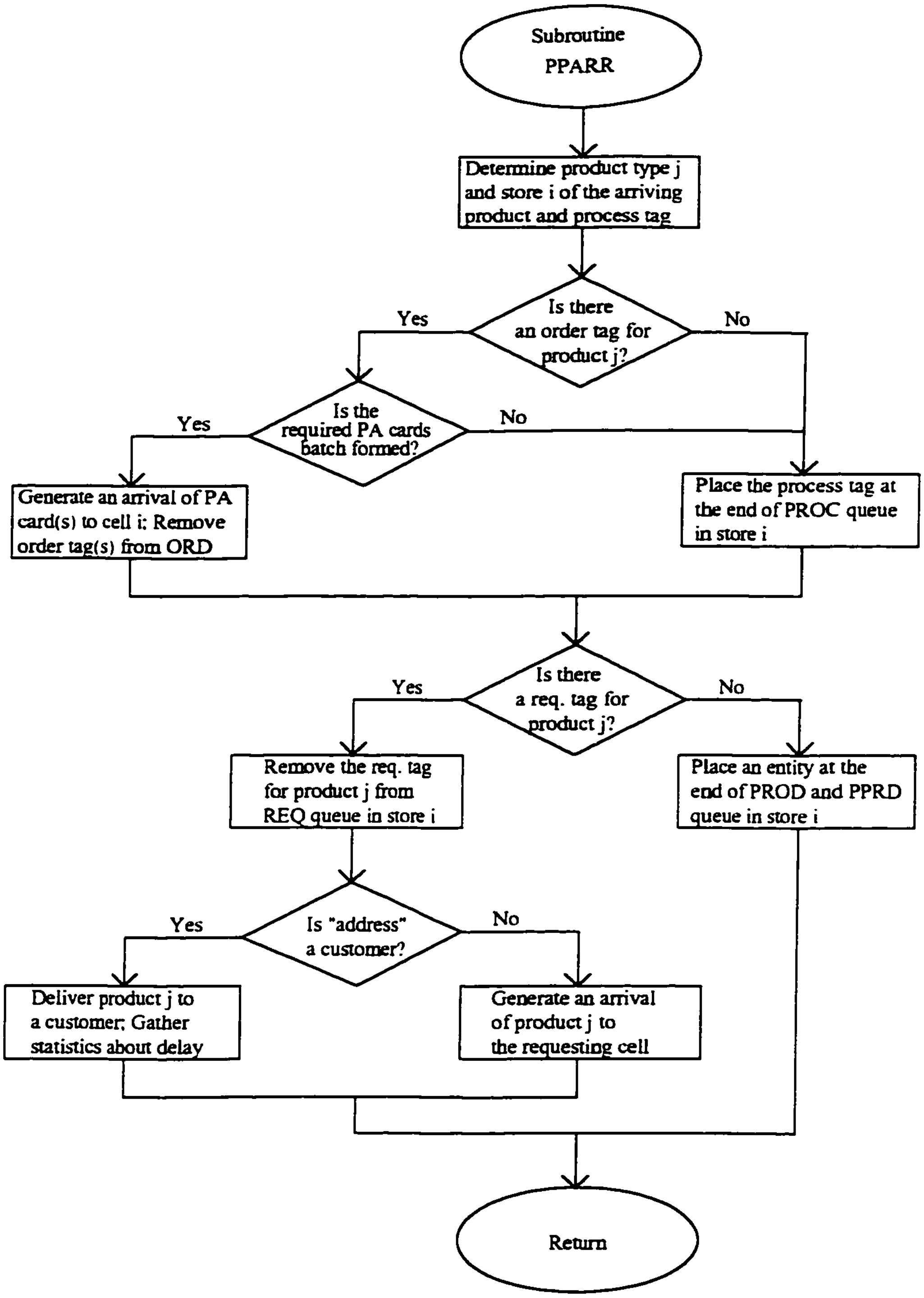


Figure 5.9 Flowchart for PPARR routine.

simultaneously: of a product and of a process tag, which became available after a completion of a processing in the cell. The routine starts with an identification of a product type and store number of the arriving product and process tag.

Process tag arrival subevent:

If the ORD-queue is not empty, the search for an order tag for the product type of an arriving process tag begins. If the arriving process tag can be matched with an order tag and also forms a required r_j batch size of PA cards, then the events of PA cards' arrival to the cell are scheduled (event type 3). Transmitting the packet of r_j PA cards, as soon as r_j cards are accumulated, causes removal of associated order tags from the ORD-queue. The arriving process tag, which cannot be matched with an order tag is placed at the end of the PROC-queue.

Product arrival subevent:

If the REQ-queue is not empty, the search for a requisition tag for the arriving product type begins. If a requisition tag is found, it is removed from the REQ-queue, and an arrival of this product type to the requiring cell is scheduled. In the case of a "final" product type, the product is delivered to a customer and leaves the system. In the case of an "assembly" product type, an arrival of this product to the cell is scheduled (event type 4). If there is no requisition tag in the REQ-queue for the arriving product type, then the arriving product type is placed at the end of the PROD and PPRD queues.

(7) End of the Simulation

At the end of the simulation (event type 7), the routine REPORT is called and the output report is generated. For the detailed description of the generated report, refer to Appendix A3. For illustration purposes, Appendix A3 contains also the complete simulation report of our example-model manufacturing configuration for an example with MRP policy parameters.

5.6 Performance Evaluation of the PAC System

The simulation report contains many measures of performance:

- inventory levels in product stores and in cells (work-in-process) per product type;
- service levels measured by probability of meeting customer demand immediately and the average delay in meeting a customer demand per product type.

In order to carry out optimization studies, we need to combine all measures of performance into one overall criterion using appropriate costs. This costing will be necessary both to simplify design and analysis of the simulation experiments and because of the very large number of simulation trial runs required for our intended response surface optimization. Costing is itself, however, problematic in that it introduces yet other fundamental characteristics of the manufacturing system in addition to arrival rates, product structure and processing times.

Generally, the following cost factors have to be specified (in dollars, per unit of product, per unit of time):

- customer delay costs (cc_j): applied to the average delays in meeting the customer demand per product type;
- inventory costs in product stores (cp_j): applied to the time averages in PPRD-queues;
- inventory costs in cells (cw_j): applied to the time averages in PWIP-queues.

We consider two different aspects of the estimation of the customer delay costs in order to analyse two different view points:

Delay Costing Option I (DCI)

For any coordination policy, the delay cost in meeting customer demand is calculated from the time of arrival of an order tag. The CORD-queue measures its unmet customer demand. Customers issue a requisition tag when they require products and they want to get their required products as soon as possible. This approach is purely from the point of view of a customer and will be best satisfied by τ_j=0 for "final" product types.

Delay Costing Option II (DCII)

For the coordination polices, which allow for $\tau_j \geq 0$ for "final" product types (Produce-to-Order: variant with $\tau_j \geq 0$, MRP and PAC in general), we apply two sorts of costs:

- Relatively "small" cost for a delay in meeting the customer demand in the time lag between issuing of an order tag by, for example, the production forecasting department and receiving of a requisition tag issued as a demand for a product by a customer. This is an artificial cost designed to keep lead times from becoming unrealistically long.
- Relatively "large" cost in the time lag elapsed between the time from the receiving of the requisition tag and delivering the product to the customer. This cost is meant to reflect the customer's disutility due to not receiving the item when the requisition is placed.

Splitting the costs in such a manner allows for implementing of a forecasting element to the expected customer demand and will result in $\tau_j \geq 0$ for "final" product types during the optimization process.

In DCII, the CORD-queues register statistics about the entities in the time lag between time of arrival of an order tag and a requisition tag.

5.7 CPU Time

The simulation program was compiled with AIX XL FORTRAN, using "-O3" optimization level, and executed on an IBM RISC/6000 model computer.

Table 5.8 shows CPU time for the simulation of different manufacturing configurations (refer to chapter 7). All runs simulate 300 days, each of 24 hours, including a 40 days warming-up. We use the DCI cost scenario, and the PTO policy parameters setting for these runs. In the case of the first four models, a single simulation run is relatively fast. However, for model 5 (with $n+r=15$) the required number of CPU seconds significantly increases. As we can see the time rises as a function of lists, the number of product delivered, and the complexity of the product structure. Run 5 indicates that the simulation as a function evaluation process could become computationally quite expensive for complex models.

Table 5.8 Statistics on simulation times

Model no.	Model Size (in number of products: $n+r$)	Mean interarrival time for all "final" products (hours)	Total number of "final" products delivered to customers	CPU (seconds)
1	2+1=3	1	6293	0.6
2	3+1=4	1	6292	0.9
3	3+2=5	1	6293	1.2
4	7+4=11	3	2080	1.8
5	11+4=15	2	3128	9.5

5.8 Description of Modeling Issues

In this section we describe some modeling issues, with emphasis on assumptions and simplifications, and with indications where different approaches could be implemented.

- Customer demand:

  Successive order tags from customers arrive according to some given stochastic process. The example simulation assumes a Poisson arrival process, but any other distribution can be used. A Poisson process is the most commonly used approach for the arrival process of customers to a service facility.

  However, at present, we assume each arriving customer requests only one item of "final" product. In chapter 7, we report on the effects of bulk arrival. The simulation program can be easily extended with the possibility of having any number of items required for each customer arrival. This is a first step in what we believe to be an important issue, namely, the dependence of model results on the distribution of the demand process.

- Priority rules:

  As we indicated earlier, product items required by different cells/customers are supplied according to a FIFO priority. Other priority procedures could be easily implemented into the program and may need to be investigated.

  Priority issues arise both in scheduling production at the cell and in meeting demand for requisitions at the stores. Buzacott and Shanthikumar dismissed the issues of priority rules assuming that some sort of simple priority schemes will suffice in both cases. These issues are quite complicated. There is a fairly substantial body of literature on single server queues, which shows that the simple priority rules are not optimal (Nunen and Puterman, 1983; Seidmann and Schweitzer, 1984, Rosa-Hatko and Gunn, 1997). Because of the complexity of the problem, we do not intend to investigate the issues of priority rules in this thesis.

- Processing times:

  Modeling processing times as exponentially distributed is often an analytical convenience. However, other distributions will often be more appropriate. In the PAC model just described, the time to perform an operation at a particular machine is an exponential random variable whose mean depends on the product type and the cell where the machine is located. However, any distribution function can be used for generating the processing times. In chapter 7, we report on some study results with different processing time distributions.

- Production activities:

  There are three types of production activities that occur at cells:

  - Process.

    By a process activity, we understand making some changes to one item of an input product, which results in one item of a different output product.

  - Assembly.

    An assembly activity is the making of a new item of output product by combining a certain number of items of each of several components (subassemblies).

  - Disassembly.

    By disassembly, we understand processing activities at cells, which transform one item of input product into a number of different end products.

  To date, the PAC simulation model allows for any combination of process and assembly activity at cells. It would be possible to add disassembly processing to the program as well. For those situations, we would need to add surplus tags, which will be generated at cells during the disassembly type of processes. Movement of products with surplus tags will require introducing additional queues for keeping the surplus tags at production stores.

- Cancellation notes and orders:

  At present, cancellation notes and orders are not included in the program. The generation and movement of both types of notes could be implemented into the simulation program. Together with surplus tags, it will allow for modeling of imperfect yield and rework.

5.9 Statistical Analysis of Output Data

Since the PAC simulation model uses random variables as input, the simulation output data are themselves random variables and care must be taken in drawing conclusions about the model's results. Each run of a PAC simulation model is a single realization of a model's true characteristics for a particular set of input parameters. These realizations could, in a particular simulation run, differ greatly from the corresponding true values for the model, as they are just particular realization of random variables that may have large variances.

There are many ways to estimate simulation output. The methods depend on the objectives of the simulation study. The options available in designing and analyzing simulation experiments depend on the type of simulation at hand, that is, terminating or nonterminating. A terminating simulation is one for which the length of simulation can be determined by a "natural" event *E*; a nonterminating simulation is one for which there is no natural event *E* to specify the length of a run. Additionally, the type of parameters for nonterminating simulations determines the method used for output analysis of the simulation. Since the initial conditions for a simulation generally affect the desired measures of performance, care must be taken in choosing appropriate initial conditions. The technique used to deal with this problem is called "warming up the model" or "initial-data deletion".

A real-life manufacturing system does not work continuously; that is, the system may be in operation for a number of hours per day, per week or per year; then it shuts down and then it operates again. However, the manufacturing operation can be seen as a continuous process, with the ending conditions for one day being the initial conditions for the next day. If we want to know the characteristics of such a system operating in a given time period, an output analysis has to be approached by methods for a terminating type of simulation. For a situation where there is a single measure of performance of interest, as can be the case with the PAC model (one performance measure: the total cost), the statistical analysis is quite easy. We perform n replications of the simulation by using different random numbers for each run, which result in $Y_1, Y_2, \ldots, Y_n$ IID random variables. The so called "fixed-sample-size procedure" is further used to construct an unbiased point estimator for the mean value of the performance measure.

The stochastic processes for most "real" manufacturing systems do not have steady-state distributions, since the characteristics of the system change over time. However, a simulation model (which is an abstraction of reality) may have steady-state distributions, if we assume that the characteristics of the model are not changing over time. This simulation model can be approached by methods for a nonterminating type of simulation. If the random variable Y has the steady-state distribution, then we would like to estimate the steady-state mean $\nu = E(Y)$. However, one of the most important problems encountered in a simulation study is that of constructing a confidence interval for the steady-state mean of a stochastic process $Y_1, Y_2, \ldots$. In the case of the PAC simulation model the overall criterion of the total cost will have to undergo statistical analysis.

Techniques to construct a confidential interval for the steady-state mean have received considerable attention in the literature, and many methodologies have been proposed, which can be grouped into two general strategies:

1. Fixed-sample-size procedures. A single simulation run of an arbitrary fixed length is made, and then one of a number of available procedures is used to construct a confidence interval from the available data.
2. Sequential procedures. The length of a single simulation run is sequentially increased until an acceptable confidence interval can be constructed. There are several techniques for deciding when to stop the simulation run.

Law and Kelton (1982), (1984) surveyed procedures of both strategies available at those times. They recommend that, if affordable, a sequential procedure be used that determines the length of the simulation during the course of the run. They concluded that sequential procedures by Fishman and by Law and Carson provide good performance relative to the criterion probability of coverage. Fishman's method is more suitable for regenerative simulation, where the Law and Carson procedure would appear to be preferable in nonregenerative applications.

It is not only important to know how to obtain the "true" characteristics of the simulation model, but also how to compare competing system alternative operating policies. Law and Kelton (1991) describe two techniques: "Paired-*t* Confidence Interval Method" and "Two-Sample-*t* Method" (known also as Welch approach), which allow for comparing two systems on the basis of some performance measure or expected response. The choice of the method depends strongly on the situation.

For the purpose of the remainder of this thesis, we will not use the stochastic analysis described above. This is because our emphasis is on the optimization approach. We will evaluate the total cost of a manufacturing system in the overall optimization algorithm by performing only a single run of the PAC simulation model. Reader is further referred to section 7.10 for a discussion of this issue. The proper stochastic analysis requires a significant amount of time and therefore it is computationally too expensive and even infeasible for more complex layouts. The philosophy underlying our approach is discussed further in 6.3. However, if the simulation model described here is being used for a true simulation study, then the issues of proper statistical analysis of the results cannot be ignored.

5.10 Validation of the Simulation Model

Validation of the simulation model is concerned with determining if the computer program is working as intended, that is, if it gives an accurate representation of the system under study. The initial validation efforts included the following:

- The model was coded and debugged in steps.
- Several simulation runs were performed under a variety of parameter settings and each output was checked for reasonableness.
- Trace and animation of the simulation model were done by generation of charts of the machine/cell states (refer to Appendix A4) and by checking large segments of computer codes behavior by printing additional information.

In addition, we compared the results of the PAC model with the results of the simulation presented in the book of Buzacott and Shanthikumar. The results of this comparison are given in Table 5.9. Our results are for a single simulation run of a total length of 292,000 hours including the warm-up time of 29,200 hours. Later, through personal communication, we learned that the results in Buzacott and Shanthikumar had a warm-up period of 1000 (customer) orders, and then the simulation run itself was 300,000 orders divided into 6 groups of 50,000 each. Buzacott and Shanthikumar used batch means to get the means of the average delay to the customer (numbers in parenthesis in Table 5.9 and cited in their book). We were also informed that our results were all within the 95% confidence intervals obtained by Buzacott and Shanthikumar. The unstable situations corresponds to cases when the system is not capable of processing the required throughput.

We delay a discussion of the validation exercise with respect to bulk demand to Chapter 7. However, as reported there, the fact that results are consistent with the theoretical calculations based on queueing theory, assists in validating our PAC simulation model.

Table 5.9 Average delay to customer (in hours) in case of simulation for two-stage Kanban system with $\lambda = 1$ and $k_1 = k_2 = 2$

ρ_1 / ρ_2	0.1	0.3	0.5	0.7	0.9
0.1	**0.001** (0.001)	**0.006** (0.006)	**0.067** (0.067)	**0.568** (0.549)	**5.571** (5.418)
0.3	**0.039** (0.039)	**0.046** (0.046)	**0.129** (0.127)	**0.715** (0.689)	**6.263** (6.064)
0.5	**0.245** (0.251)	**0.262** (0.266)	**0.401** (0.402)	**1.276** (1.23)	**11.177** (10.5)
0.7	**1.132** (1.14)	**1.197** (1.19)	**1.576** (1.55)	**3.844** (3.70)	**(***)** (***)
0.9	**6.908** (6.94)	**7.439** (7.45)	**11.934** (11.4)	**(***)** (***)	**(***)** (***)

(***) unstable

5.11 Closing Remarks

In this chapter, we have outlined the structure of a simulation model for a multiple-cell multiple-product manufacturing system coordinated by PA cards. Although this model is not at a final stage of development, in the sense that it can be extended, it does permit a rapid performance evaluation of a PAC controlled system for a given choice of parameters. The model implements the main features of the PAC concept. It has been developed mainly to function as a cost evaluation in the overall optimization process. However, this model is a contribution to the research on various manufacturing issues and is essential to study the PAC control for simple and complex manufacturing systems.

We hope to eventually apply this model to a "real life" manufacturing structure. We anticipate that this will require us to implement certain system-specific changes to the PAC simulation model to deal with details which may be of particular importance.

We believe that a simulation model, such as that developed here is essential to understand how real systems can behave. Much of the analysis of PAC systems to date has been based on queueing theory and considers relatively simplified situations. If we wish to do research on systems of greater complexity, particularly those where the distributional assumptions necessary for queueing results are not really appropriate, then simulation will be an essential tool.

Chapter 6

PAC OPTIMIZATION ALGORITHM

6.1 Introduction

In this chapter we describe the primary features of PACOPT, an optimization algorithm for the PAC system. The simulation problem described in Chapter 5 was aimed at evaluating the performance of a PAC system with given parameters (z_j, k_j, r_j, τ_j, j=1,...,n). The optimization problem is to choose those parameter values that give the "best" system performance. Our optimization algorithm is based on multidimensional search methods using the PAC simulation as a function evaluator.

Optimizing a simulation poses several conceptual and practical difficulties. For the most part, we take the view that the function we are optimizing is deterministic, because we use identical random number seeds for each function evaluation. In section 6.3, we discuss some of the issues involved with trying to optimize with respect to the stochastic process instead of dealing only with the sample path for the given seeds.

By describing principles of the optimization methods, we will notice that there are not many differences between the Tabu Search method and the pattern search of Hooke and Jeeves. A pattern point in Hooke and Jeeves does not need to improve the solution and can be regarded as an equivalent of the next solution being possibly worse than the preceding solution in the Tabu Search.

We go on to describe another optimization algorithm, PACRAN. This algorithm is based on a random search principle. As we will demonstrate later (refer to Chapter 7), this algorithm can produce a better solution, but requires significantly more computation time than PACOPT. PACRAN can be used as an additional check on the optimal solution candidate.

Before describing the PAC optimization algorithms, first we outline our considerations for the choice of the optimization method with emphasis on difficulties, constraints and the overview of the existing optimization procedures.

6.2 Description of the Optimization Problem

6.2.1 General

The PAC system is a manufacturing process. Optimization of such a process requires the following components:

- a mathematical model of the process and the process control variables (parameters);
- an economical model of the process, that is a function representing the costs associated with the production process;
- an optimization method to select the values of parameters, which result in the minimum costs.

The PAC simulation model described in Chapter 5 is a mathematical model of the PAC system. The PAC simulation can generate many measures of performance. For the remainder of this thesis, we assume that we are given prices, which then result in one overall criterion of total cost. The PAC simulation evaluates the system performance for a given set of parameters. In this sense, the simulation mechanism is a function that turns input parameters into the output performance measure. In the case of the PAC model, the total cost is a function of $4*n$ decision variables, z_j, k_j, r_j and τ_j (j=1,...,n).

This function has no explicit form and is certainly very complicated. The potential interpretation of this function as a random variable raises additional complications that we discuss at length in 6.3.

The PAC optimization problem is a nonlinear programming problem. There are many methods for nonlinear problems. Our choice of a method has been guided by a number of aspects that we discuss in 6.2.2 - 6.4.

6.2.2 Difficulties

An objective function given by a simulation model, and not by an equation, introduces many problems. In this section, we discuss the main difficulties.

6.2.2.1 Convexity

Nonlinear optimization algorithms can generally find only a local optimum. Therefore, it is important to know when the local minimum is certain to be a global minimum in the specified feasible region. If the feasible region is a convex set and the objective function is a convex function, then these are sufficient conditions to guarantee that a local optimum is global.

Verifying the convexity of an arbitrary function is usually a difficult task. For the PAC cost function, generated by a simulation, it does not seem to be possible. We expect that the cost function will be neither convex nor concave and may have many extrema.

In order to verify how satisfactory the obtained local optimum is, we could perform a number of optimization runs with different starting points. In addition, applying of some sort of a random search as a check could be worthwhile consideration.

6.2.2.2 Derivatives

Most methods for nonlinear optimization require derivatives. The PAC cost function is not differentiable analytically and may not be differentiable at all. The choice for the optimization method will be limited to methods, which do not require derivative measurements.

6.2.2.3 Accuracy of Time Delays

Another problem is the degree of accuracy required in evaluation of the performance of the PAC system. There is a question of how exactly the values of delay times (parameters τ_j) should be specified. We make an arbitrary decision to approximate delay times to integer values in minutes. In the case of the PAC model, the approximation of delay times to minutes will be sufficient enough to estimate the system behavior.

6.3 Stochastics

The function generated by a PAC simulation is a stochastic function. If the issue is the precise statistical estimation of outputs from the system, then the simulation results properly should be analyzed by techniques described in Chapter 5 (refer to 5.9). However, from an optimization view point, it is worth noting different views for the goal of the optimization process. If we let p indicate the set of parameters, s the stream numbers chosen for generating various random variables, and T the time we carry out the simulation, then there are several possible options:

1. $\underset{p}{\text{Min}}\, E(f(p/s,T))$ - find the optimal expected value of the system
2. $E(\underset{p}{\min}(f(p/s,T)))$ - find the expected value of the least cost solutions for each given sample path
3. $\underset{p}{\text{Argmin}}\, E(f(p/s,T))$ - find the optimal value of the parameters that minimize the cost of the system

4. $E(\underset{p}{\operatorname{argmin}}(f(p/s,T)))$ - find the expected value of the optimal parameters that minimize the cost of the system for each given sample path

It is evident that the stochastic analysis will need a significant amount of computation time in order to determine a mean value estimation of a single value of the total cost for a set of given PAC parameters. If f(p) is a random variable, then we are required to construct a confidence interval to estimate E(f(p)). In order to compare two function values E(f(p_1)) and E(f(p_2)), we are required to do a hypothesis test to see whether the observed difference between these two expectations is significantly different from zero. On the other hand, the optimization process typically demands many function evaluations (=PAC simulations) and these can be computationally expensive, for a more complex manufacturing layout. In most cases, it would not be feasible to perform optimization with a complete output data analysis per each function evaluation.

Generally, we only evaluate the performance of the system, characterized by a set of PAC parameters, by a single simulation run. Within the optimization process, we chose to start each simulation with the same values of the stream numbers for the random variables. This means that all simulations are realizations of the same system conditions, and are in fact deterministic. If f($p/s,T$) is deterministic, then finding p that minimizes a particular simulation sample path can be done straight forwardly by comparing f($p_1/s,T$) and f($p_2/s,T$). In our optimization process, we find p that minimizes a given realization of arrival and processing times.

There are many ways of managing to obtain these realizations. One is to generate all processing times as attributes of the product (and its components) immediately when the initial demand for that product arrives in the system. In this way the processing times are completely independent of the choice of the PAC parameters, which is desirable in most circumstances. Our approach was less sophisticated. We used a single stream number to generate all processing times. The function generated by a simulation is deterministic, but the process times are not independent from the PAC parameters.

Based upon this, it is clear that a single simulation run does not correspond directly to the expected function value. However, if we choose the length of simulation reasonably long, and the process is homogenous, the realization tested will be a fair approximation of the function value (refer to 7.2.3). Choosing the simulation length will depend on the size and complexity of the model and available computer time. Generally, some compromise between the CPU time required and the accuracy of the generated results will be necessary. In addition, any inaccuracy in the estimation of the proper length of a simulation can be accepted, as our aim is, in the first place, to test and develop the optimization procedure itself and further to make some indicative comparison studies for different coordination policies.

6.4 Constraints

The basic elements of any optimization model are decision variables, an objective function and constraints. There are a number of constraints related to the PAC parameters. If we consider the variables z_j, k_j, r_j and τ_j, and the definitions of various policies, we observe that the choices of those variables have restrictions. First, there is a non negativity constraint for all PAC parameters. Furthermore, the various systems correspond to:

- discrete choices in z_j, $z_j \in [0,\infty)$;
- discrete choices in k_j, $k_j \in [1,\infty)$;
- discrete choices in r_j, $r_j \in [1,\infty)$;
- discrete choices in τ_j, $\tau_j \in [0,\infty)$; with an assumption that the τ_j values are in minutes and the integer approximation is accurate enough.

In addition, policies impose further restrictions on parameters. Therefore, the number of possible PAC parameters combinations, which should be taken into consideration during the optimization procedure, will be significantly reduced.

This implies the following constraints:

- all parameter values are required to be integers;
- each parameter has a feasible range. The minimum of a parameter can always be defined, where its maximum is either equal to a large number (in the case of an inequality) or equals its minimum (in the case of an equality).

Another constraint is set by a relationship between process tags k_j and the batch size of PA cards r_j for each product j. The constraint requires that the batch size of PA cards does not exceed the number of process tags ($r_j \leq k_j$; j=1,....,n) to allow forming of a PA cards batch by matching order tags with process tags. For non-serial assembly systems, there are also feasibility constraints for parameters r and k, as discussed in detail in 4.2.6.2.

In case of some control policies, there are constraints in the form of direct relationships between parameters. Kanban policy sets an equality between the amount of process tags and the initial inventory for a given product (k_j=z_j; z_j>0; j=1,....,n). In case of IC and CONWIP, a number of process tags for a stage is calculated, as a sum of an initial inventory at this stage and the amount of process tags from the next downstream stage multiplied by a number of items required for processing.

6.5 Optimization Method

In this section, we first give an overview of multivariable optimization methods and describe general techniques of dealing with constraints. Then we discuss the main features of various *logical search methods* with a detailed description of the Hooke and Jeeves algorithm, on which our PACOPT is based. Finally, we describe the specialization of the

Hooke and Jeeves method to our PAC problem and discuss the particulars of the family of the PAC optimization algorithms.

6.5.1 Multivariable Optimization Methods

The PAC optimization problem belongs to the category of constrained multivariable nonlinear optimization problems. There are two ways to deal with such problems. One way is to use a constrained multivariable search method. The other is to convert the constrained problem to an unconstrained one and then, to apply an unconstrained multivariable search method. As some constrained methods convert the problem into an unconstrained one anyway (Lagrange multipliers) and other are not applicable to our PAC problem (quadratic-, separable-, geometric programming, etc.), the methods for unconstrained problems are of more interest here.

The existing methods for multivariable unconstrained optimization procedure can be classified into two main groups (Pike 1986):

- methods, which use a local, geometric property to find a direction having improved value of the economic model. Typically, derivative measurements are required for those methods, e.g., the direction of steepest descent (quasi-Newton methods) and quadratic fit to the objective function (conjugate gradient methods).
- *logical* (*hill climbing*) *methods* as they use an algorithm based on a logical concept to find an improved direction of the profit function. Typically, derivative measurements are not required for those methods, e.g. pattern search and polyhedron search.

The main difference between "local geometry" and "hill climbing" is in using derivative measurements. However, both types of methods are based on a similar search principle to find a direction of improving the value of the function.

We also could consider using heuristics as Simulated Annealing or Tabu Search method. Simulated Annealing was derived from an analogy to a physical annealing process for condensed matter and was first introduced by Kirkpatrick, Gelatt and Vecchi, 1983. Simulated Annealing is essentially a randomized local improvement method. Tabu Search (Glover and Laguna, 1992) is in many ways similar to hill climbing methods. The procedure moves from one solution to the best neighbouring solution found. The main difference is that the next solution can be worse than the preceding solution. The reason for allowing a solution to be worse than the previous one is to give the procedure the opportunity to move away from a local minimum and have a chance to find a lower minimum. The so called Tabu techniques prevent cycling. Both techniques can be viewed as generalization of the iterative improvement approach to combinatorial optimization problems. Both methods can be applied to the problems without much knowledge of the structural properties of the problem. Both are relatively easy to program, however, the amount of computation time needed to obtain a solution tends to be relatively long in comparison with more direct pattern-search approaches. Both methods require a significant number of function evaluation, and in the case of a PAC system, where each function evaluation is computationally expensive, they will result in very long computer runs.

6.5.2 Methods of Dealing with Constraints

There are several methods of dealing with constraints. For example, penalty, barrier and augmented Lagrangian functions convert the constrained optimization problem into an unconstrained one (Bertsekas, 1995). A penalty function method adds a penalty term to the objective function to incur a positive penalty for infeasible points. Barrier function methods are similar to penalty function methods except that they start at an interior point of the feasible region and set a barrier against leaving the feasible region. Augmented Lagrangian function method combines a penalty function with the Lagrangian function to modify the objective function with constraint equations.

All the methods require an evaluation of modified objective function. In the case of the PAC problem, it would be rather desirable to check feasibility of parameters before executing an expensive function evaluation. The hill climbing methods can deal better with the PAC constraints, then the local geometry methods. These methods are designed to unconstrained problems, but let us plan search steps in such a way that moves into unfeasible solutions are not allowed.

6.5.3 Hill Climbing Methods

Hill climbing methods belong to the category based on logical concepts to find a sequence of improved values of the economical model leading to an optimum. These techniques do not require derivative measurements, and the method compares the computed values of the economical model; they begin with local exploration, and then attempt to accelerate in the direction of success. In our case, the PAC simulation model will function as the economical model. Cost values generated by simulation runs will be compared with each other.

There are a number of methods of this type. We could consider the pattern search algorithm of Hooke and Jeeves (Bertsekas 1995), Rosenbrock's method (Bazaraa and Shetty 1979), and the simplex algorithms first described by Spendley, Hext and Himsworth (1962), and later developed by Nelder and Mead (1965).

The Hooke and Jeeves method is one of the most widely used optimization techniques. This method is an old (Hooke and Jeeves, 1961) but efficient, robust procedure, which is also relatively easy to implement. The method starts by selecting an initial base point and step lengths for each direction. The procedure alternates a sequence of exploratory steps about a base point, which if successful are followed by pattern moves. The exploratory search begins by adding to the first variable the step length and the evaluation of the optimization function. If this improves (decrease) the function value, it is used as a new reference value. If it does not, the step length is subtracted from the first variable. If neither step results in a reduction, the procedure leaves the current variable and considers changes in the next component. The final result of a successful exploratory search finds a new base point. Each pattern move uses the information from the previous exploration search to calculate the next pattern point. Each pattern move is made by moving each variable from the latest base point with an amount equal to the difference between the new and old base point values. This difference will generally include a previous move. If the pattern move fails to improve the optimization function, it is canceled, and a new exploration search begins around the base point. The process is repeated until no better point can be located. Then the step length is reduced by some arbitrary fraction, and the exploratory search is continued. The process is terminated when the step lengths have been reduced to a predetermined small value.

The method of Rosenbrock can be regarded as a further development of the Hooke and Jeeves technique. The continuous version of this method is aimed at guaranteeing orthogonality at search directions. The originally proposed method does not use line searches, but takes discrete steps along the search directions. It uses preassigned multipliers (expansion factor $\alpha>1$, contraction factor $-1<\beta<0$) to control the lengths of the steps. The discrete steps are taken along the n search directions, which are mutually orthogonal. Initial directions are formed by coordinate directions. At each iteration, the procedure searches along n orthogonal directions. When a point with an improved function value is reached (at the end of an iteration), a new set of orthogonal directions is formed by the Gram-Schmidt procedure, or orthogonalization procedure. The main difficulty in applying the method of Rosenbrock to the PAC problem will be in complex calculations of mutually orthogonal search directions. The continuous version cannot deal with discrete changes of PAC parameters.

The simplex method uses different coefficient (reflection coefficient: $\alpha>1$, expansion coefficient: $\gamma>1$ and contraction coefficient: $0<\beta<1$) for the three basic operations used in the method. The idea of the method is to compare the value of the function at the $(n+1)$ mutually equidistant extreme points in n-dimensional space, which define a regular simplex. The function is evaluated at each extreme point of the simplex. The procedure moves toward the optimum by reflecting the extreme point with the worst function value through the centroid (average) of the remaining n extreme points, to identify a new simplex adjacent to the previous one. Nelder and Mead proposed several modifications to the method which allow the simplices to become non-regular. Simplex reflections are expanded in the same directions if the reflected value is particularly good. A poor value results in a contraction. If the function value at the contracted point is poorer, the size of the simplex is reduced. The main difficulty in applying this method to the PAC problem will be in generating the initial $(n+1)$ points, especially when the PAC optimization will deal with many parameters. In addition, it is not clear how to deal with reflection for discrete values of PAC parameters.

Taking all considerations into account, we conclude that for the PAC system the most suitable approach will be to base the optimization algorithm on the method of Hooke and Jeeves.

6.5.4 The Method of Hooke and Jeeves

Since we want to base our optimization algorithm on the method of Hooke and Jeeves, we need to describe this method in some detail. The direct search of Hooke and Jeeves for function minimization can be described as follows:

Initialization

- Let $\mathbf{d}_1,\ldots,\mathbf{d}_n$ be the coordinate directions of in $\mathbf{R}^n$.
- Let $\varepsilon>0$ be a scalar to be used for terminating the algorithm
- Choose an initial step size $\Delta\geq\varepsilon$, step reduction interval $\Theta>0$ and an acceleration factor $\alpha>0$
- Choose an initial base point $\mathbf{x}_1(\mathbf{x}_1=x_1,\ldots,x_n)$, let $\mathbf{y}_1=\mathbf{x}_1$
- Let $k=j=1$

Main Loop

1. If $f(\mathbf{y}_j+\Delta\mathbf{d}_j)<f(\mathbf{y}_j)$, the exploratory move is a success; let $\mathbf{y}_{j+1}=\mathbf{y}_j+\Delta\mathbf{d}_j$ and go to step 2. If $f(\mathbf{y}_j+\Delta\mathbf{d}_j)\geq f(\mathbf{y}_j)$, the exploratory move is a failure. In this case, if $f(\mathbf{y}_j-\Delta\mathbf{d}_j)<f(\mathbf{y}_j)$, let $\mathbf{y}_{j+1}=\mathbf{y}_j-\Delta\mathbf{d}_j$ and go to step 2; if $f(\mathbf{y}_j-\Delta\mathbf{d}_j)\geq f(\mathbf{y}_j)$, let $\mathbf{y}_{j+1}=\mathbf{y}_j$, and go to step 2.
2. If $j<n$, replace j by $j+1$, and go to step 1 (continue exploratory search). Otherwise, go to step 3 if $f(\mathbf{y}_{j+1})<f(\mathbf{x}_k)$, and go to step 4 if $f(\mathbf{y}_{j+1})\geq f(\mathbf{x}_k)$.
3. Make a pattern move. Let $\mathbf{x}_{k+1}=\mathbf{y}_{j+1}$, and let $\mathbf{y}_1=\mathbf{x}_{k+1}+\alpha(\mathbf{x}_{k+1}-\mathbf{x}_k)$. Replace k by $k+1$, let $j=1$, and go to step 1.
4. If $\Delta\leq\varepsilon$, stop; $\mathbf{x}_k$ is the solution. Otherwise replace Δ by Δ/Θ. Let $\mathbf{y}_1=\mathbf{x}_k$, $\mathbf{x}_{k+1}=\mathbf{x}_k$, replace k by $k+1$, let $j=1$, and go to step 1.

Steps 1 and 2 consider an exploratory search. Step 3 describes a pattern move along the direction $\mathbf{x}_{k+1}-\mathbf{x}_k$. It is interesting to observe that the decision whether to accept or reject the pattern move is not made until after an exploratory search is completed. In this sense, we can recognize some similarities of the method of Hooke and Jeeves with the

Tabu Search algorithm. A pattern point does not need to improve the solution and can be regarded as an equivalent of the next solution being possibly worse than the preceding solution in the Tabu Search. Step 4 is used to reduce the step size.

The method is illustrated for two parameters in Figure 6.1. The points generated are represented by dots. The numbers indicate the order in which points are generated.

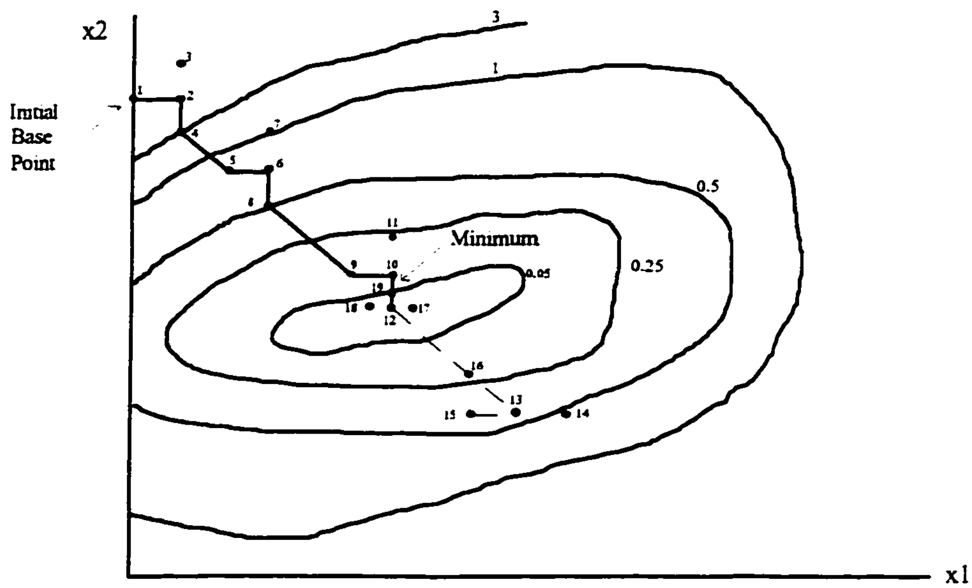


Figure 6.1 Illustration of the direct search of Hooke and Jeeves.

6.5.5 Specialization of Hooke and Jeeves to the PAC Problem

We need to adjust the Hooke and Jeeves method to make it applicable for the PAC problem. The main changes are caused by some constraints. In this section, we describe the general ideas and give the specifics of implementation in the computer program.

6.5.5.1 General Ideas

We require parameters to take only on discrete values. This can be easily implemented in the model by defining parameters as integers and by assuring further in the program. that the calculations of any new values of parameters will result in integers. This can be done by either an approximation of any real number to an integer, or by choosing the step lengths, step interval reductions and an acceleration factor in such a way that all calculations will result in integers.

Restrictions on the feasible region of parameters can be accommodated by implemention of the feasibility check and rejection of the move.

There are a number of restrictions between parameters r and k (refer to 4.2.6.2). It is desirable to eliminate unworkable situations before carrying out each time demanding simulation. The unfeasible situation should result in a rejection of the move.

The specific relationships between parameters z_j and k_j for Kanban and IC/CONWIP policies will need separate procedures, where the total number of parameters to be optimized will be reduced, and k_j parameters will be related to the z_j parameters according to the specific requirements.

6.5.5.2 Specifics of Implementation in the Computer Program

Discrete values of parameters

We define all variables and their step lengths as integers. We set the acceleration factor α equal to 1. This guarantees that all parameters will remain integers after calculations of base points and pattern points. Here, the step reduction interval equals the step length, and the predetermined small value to stop the algorithm equals 1. Consequently, the first reduction of the step length results always in the step length of one unit, and the second one, in a value less than 1, which will stop the algorithm.

Feasible region of parameters

We specify the minimum and the maximum value for each parameter. Each exploration move into the infeasible region will be considered as "bad". The bad move will add a large constant value to the best objective function value found, and will not require a function evaluation (=PAC simulation). This procedure will guarantee the rejection of the exploration move. In case of a "bad" pattern move, the values of parameters will be adequately reduced to maximum or minimum values. That means, if the parameter value will be larger than its maximum, it will be set to its maximum; and if the parameter value will be lower than its minimum, it will be set to its minimum.

For parameters set to a certain constant value, their initial, maximum and minimum values will equal this constant. For a parameter having an unlimited maximum value, its maximum will be set to a large number. If the optimal solution will contain the maximum value of a parameter, we will have to increase the maximum value and run the optimization again. It is also possible that the maximum value has been used, but it does not explicitly show on the best optimal solution found. For this reason it would be convenient to keep track of the maximum values of parameters ever being used in the optimization process. Keeping track of lowest values ever used by parameters can also be useful to directly estimate the range of parameters used in the solution of the problem.

Feasibility check for parameters k and r

There are three situations, which require a feasibility check for parameters k and r (refer to 4.2.6.2).

For all products j the following relationship between parameters k_j and r_j always has to be satisfied:

$$r_j \leq k_j \qquad j=1,....,n \qquad [6.1]$$

To assure this constraint we adequately change maximum and minimum values of the r_j and k_j parameters before any new exploration search begins. That means, that we temporarily change the feasible region of those parameters, if they are considered for the search. In the case of a pattern move, we lower the value of r_j parameter to the k_j parameter value if the condition [6.1] is not satisfied.

For a case of a flow line, the parameters r_u , k_u of the upstream u production unit and the parameters r_d , k_d of the downstream d production unit have to satisfy the following relationship to avoid production blocking:

$$k_d - r_d + \left\lfloor \frac{r_d}{r_u} \right\rfloor r_u \geq r_d \qquad [6.2]$$

In a case of product u being a component for products $d_1, d_2, \ldots, d_q$, the number of PA cards sent from stores $d_1, d_2, \ldots, d_q$ cannot be less than the batch size of product u, as given by [6.3]. Condition [6.3] is stronger than condition [6.2].

$$\left\lfloor \frac{k_{d_1}}{r_{d_1}} \right\rfloor r_{d_1} + \left\lfloor \frac{k_{d_2}}{r_{d_2}} \right\rfloor r_{d_2} + \cdots + \left\lfloor \frac{k_{d_q}}{r_{d_q}} \right\rfloor r_{d_q} \geq r_u \qquad [6.3]$$

Conditions [6.2] and [6.3] can be easily implemented in the computer program, when executing the exploration steps for r and k parameters. If the adequate condition is not satisfied, the function evaluation is not performed by running the simulation. Instead, a large constant value is added to the so far best function result (the lowest cost), which guarantees a rejection of the exploration move.

It is possible that [6.3] may be satisfied, but there are still situations, when the production process is blocked. We have not yet been able to eliminate those situations.

We implement an extra check for feasibility of the simulation after each simulation is completed. A very large penalty function replaces the total cost of the simulation run. if at least one final product is not delivered to customers. One good aspect of the simulation runs with blocking is that they take significantly less time than "proper" ones.

The PAC simulation defines the relationship between each product and its components; that means for each product the names of its upstream products and a number of each component required. To check the conditions [6.2] and [6.3], the PAC model has to "know" (for each product j) the names of all downstream products (product numbers) for which the product j is a component, in order to calculate the values of parameters r_d and k_d properly. This does not require additional input data. The existing data regarding products and their subassemblies can be used to make calculations in the "opposite" direction, that is, information about products and their assemblies.

Kanban constraint

The Kanban policy requires equality between the number of process tags and the initial inventory for a given product: $k_j=z_j$ ($z_j>0$, $j=1,....,n$). This constraint can be implemented to the model by using only one set of those parameters (k_j or z_j) for optimization.

IC /CONWIP constraint

For IC and CONWIP, the number of process tags for a stage is calculated as a sum of an initial inventory at this stage and the number of process tags from the next (downstream) stage multiplied by a number of items required for processing (refer to 4.2.6.3).

First, we need to "translate" this constraint according to the PAC simulation approach, that is, to relate the parameter values to products j (j=1,.....,n), final (f), assembly (a) and final/assembly (fa) and not stages i (i=1,.....,m).

IC	CONWIP	
$z_j>0$	$z_j>0$	$j=1,...,f+fa$
$z_j\geq 0$	$z_j=0$	$j=f+fa+1,....,n$
$k_j=z_j$	$k_j=z_j$	$j=1,...,f$
$k_j=z_j+\sum_{p=1}^{q}(k_p b_{pj})$	$k_j=z_j+\sum_{p=1}^{q}(k_p b_{pj})$	$j=f+1,....,n$

where:

- k_p is a number of process tags for product p, and product p requires product j for processing or assembly operations.
- b_{pj} is a number of items of product j required for production of product p.
- q is the number of all different products which require product j for processing or assembly operations.

The optimization algorithm for IC/CONWIP, as in case of Kanban, will be required to optimize only one set of parameters k_j or z_j. In this case, z_j parameters will be optimized and k_j parameter values will be related to the optimized z_j. The PAC model will need to "know" for each product j the names of all downstream products for which the product j is a component, and the number of items of product j required, in order to calculate the values of parameter k_j properly. The existing data regarding products and their subassemblies can be used to make calculations in the "opposite" direction, that is, information about products, their assemblies and the bill of material (BOM).

6.5.6 Family of PAC Optimization Algorithms

For optimization procedure, we decided to consider the following 9 coordination schemes:

- PAC   Production Authorization Cards
- BSS   Base Stock System
- MRP   Material Requirement Planning
- PTO   Produce-to-Order
- PTO,$\tau \geq 0$ Produce-to-Order, variant with $\tau \geq 0$
- LC   Local Control
- Kanban   Kanban
- IC   Integral Control
- CONWIP   Constant Work-in-Process

We did not consider OPT, as it deals with a bottleneck facility, and cannot be characterized by a consistent parameter setting for a system.

Instead of 9 separate programs, one for each coordination policy, we have actually developed 4 different variants. We tried to accommodate as many coordination schemes as possible within one variant of the PAC algorithm to reduce the number of required variations of the algorithm to minimum.

The specific PAC algorithms required are as follows:

1. *opt* - used for PAC, BSS, MRP, PTO variant with $\tau \geq 0$ and LC.
2. *opk* - used for Kanban.
3. *opi* - used for IC and CONWIP.
4. *pac* - used for PTO; it is a single simulation run as the PAC parameters are set to fixed values.

Note: We could use just one program to combine all variants. However, we found it more practical to maintain separate programs.

Table 6.1 schematically overviews the main parameter settings for control policies approached with *opt* algorithm. Table 6.2 shows parameter settings for the control policies approached with *opk*, *opi* and *pac*.

Table 6.1 Parameter settings for control policies optimized via *opt*

opt:				
PAC	**BSS**	**PTO, $\tau \geq 0$**	**MRP**	**LC**
$z \geq 0$	$z > 0$	$z = 0$	$z \geq 0$	$z > c$
$k \geq 1$	$k = \infty$	$k = \infty$	$k = \infty$	$k = c$
$r \geq 1$	$r \geq 1$	$r = 1$	$r \geq 1$	$r = 1$
$\tau \geq 0$	$\tau = 0$	$\tau \geq 0$	$\tau \geq 0$	$\tau = 0$

Table 6.2 Parameter settings for control policies optimized by *opk*, *opi* and *pac*

opk:
Kanban
$z > 0$
$k = z$
$r \geq 1$
$\tau = 0$

opi:	
IC	**CONWIP**
$z_i > 0$, $(j=1,\ldots,f+fa)$	$z_i > 0$, $(j=1,\ldots,f+fa)$
$z_i \geq 0$, $(j=f+fa+1,\ldots,n)$	$z_i = 0$, $(j=f+fa+1,\ldots,n)$
$k_i = z_i$, $(j=1,\ldots,f)$	$k_i = z_i$, $(j=1,\ldots,f)$
$k_j = z_j + \sum_{p=1}^{q}(k_p b_{pj})$, $(j=f+1,\ldots,n)$	$k_j = z_j + \sum_{p=1}^{q}(k_p b_{pj})$, $(j=f+1,\ldots,n)$
$r_i = 1$, $(j=1,\ldots,n)$	$r_i = 1$, $(j=1,\ldots,n)$
$\tau_i = 0$, $(j=1,\ldots,n)$	$\tau_i = 0$, $(j=1,\ldots,n)$

pac:
PTO
$z = 0$
$k = \infty$
$r = 1$
$\tau = 0$

The *opt* optimization algorithm considers all 4**n* PAC parameters, i.e. z_j, k_j, r_j, τ_j, $j=1,\ldots,n$. For parameter settings fixed to a certain value (=1, =0, =∞, =c), the initial, minimum and maximum values of those parameters will be set to this value.

The general PAC policy does not set any limitation on the parameters. Generally, all 4*n parameters are optimized within given maximum and minimum values. BSS searches for optimal values of parameters z_j and r_j, that is, 2*n parameters are optimized. MRP sets all k_j parameters to a large number M, which is equivalent to an unlimited quantity of process tags, and optimizes the remaining 3*n parameters. PTO, variant with $\tau \geq 0$, optimizes only parameters τ_j, where the LC does search for optimal vales of only z_j parameters (both policies are n parameters optimizations).

We want to underline that in *opt* all 4*n parameters are used for optimization, but when for a parameter its initial, maximum and minimum values are set to the same number, the *exploration* move around this parameter will not result in the function evaluation (PAC simulation). Setting parameters to fixed values reduces the number of parameters for effective optimization and at the same time the number of required PAC simulation as cost function evaluations within the optimization algorithm.

The *opk* algorithm is used only for optimization of PAC parameters for a case of the Kanban policy. Here the optimization considers only parameters z_j and r_j, so it is 2*n parameters optimization. Values of k_j are related to the values of z_j and τ_j values are fixed before hand to zero.

The *opi* optimization algorithm optimizes only z_j parameters. It is at most n parameters optimization. Values of r_j and τ_j are fixed in the program respectively to 1 and 0. Values of k_j are related to the optimized z_j parameters. For CONWIP policy the effective number of optimized z_j values is equal to number of *final* and *final/assembly* products (j=1,...,f+fa). IC policy does optimize all z_j parameters (j=1,...,n).

6.6 PACOPT Optimization Algorithm

We are now ready to describe the main structure for *opt* variant of the algorithm.

PACOPT Optimization Algorithm:

Input

- $\mathbf{x}_1$ initial base point with integer values of total N=4*n PAC parameters;

  $\mathbf{x}_1 = (z_1, k_1, r_1, \tau_1,..., z_n, k_n, r_n, \tau_n) = (x_1,...,x_N)$
- $x_{j,max}$ maximum values of PAC parameters, (j=1,...N), as integers
- $x_{j,min}$ minimum values of PAC parameters, (j=1,...N), as integers
- s_j step length of PAC parameters, (j=1,...N), as integers
- FE_{max} maximum number of function evaluations (= PAC simulations)

Initialization

- Let $\mathbf{d}_1,...,\mathbf{d}_N$ be the coordinate directions of N variables, $x_1,...,x_N$
- Let ε=1 be a scalar to be used for terminating the algorithm
- Let $\mathbf{y}_1=\mathbf{x}_1$, $\mathbf{y}_1=(y_1,...,y_N)$, ($j$=1,...$N$)
- Let $x'_{j,max}=x_{j,max}$ and $x'_{j,min}=x_{j,min}$, (j=1,...N)
- Let $NSUB_p$ indicate no. of components (subassembly) of product p without raw materials, (p=1,...,n)
- Calculate $NASS_p$ which indicates no. of products (assembly), which use product p as component, (p=1,...,n)
- Let u be product upstream of product p, its subassembly
- Let d be product downstream of product p, its assembly
- Let indicate by C1: $k_d - r_d + \left\lfloor \frac{r_d}{r_u} \right\rfloor r_u \geq r_d$ and by C2: $\left\lfloor \frac{k_{d_1}}{r_{d_1}} \right\rfloor r_{d_1} + \left\lfloor \frac{k_{d_2}}{r_{d_2}} \right\rfloor r_{d_2} + \cdots + \left\lfloor \frac{k_{d_q}}{r_{d_q}} \right\rfloor r_{d_q} \geq r_u$
- Let *pol* be the name of policy for which optimization is performed (names of policies are PAC, MRP, PTO-$\tau \geq 0$, BSS, LC)

- Let M indicate a large constant number
- Set counter FE to 0
- Let $h=j=1$
- Calculate cost function $f(\mathbf{y}_j)$

Main Loop

While no. of iterations (FE) < FE_{max}

1. Set $y'_j=y_j+s_j$. If $y'_j>x_{j.max}$ then $f(\mathbf{y'}_j)=f(\mathbf{y}_j)$+M. Go to step 2.

   If $y'_j\leq x_{j.max}$ and $pol\neq$PAC, calculate $f(\mathbf{y'}_j)=f(\mathbf{y}_j+s_j\mathbf{d}_j)$. Go to step 2.

   If $y'_j\leq x_{j.max}$ and pol=PAC and y'_j is none of r_p, calculate $f(\mathbf{y'}_j)=f(\mathbf{y}_j+s_j\mathbf{d}_j)$. Go to step 2.

   If $y'_j\leq x_{j.max}$ and pol=PAC and y'_j is one of r_p, perform checks for conditions C1 and C2: {for each u of product p (no. of u =$NSUB_p$), if $NASS_u$=1, set $r_d=y'_j$ and check C1; otherwise set one of $r_{d_q}=y'_j$ and check C2; set $r_u=y'_j$ and if $NASS_p$=1 check C1; otherwise check C2}. If y'_j satisfies all performed checks on C1 and C2 conditions, calculate $f(\mathbf{y'}_j)=f(\mathbf{y}_j+s_j\mathbf{d}_j)$; otherwise $f(\mathbf{y'}_j)=f(\mathbf{y}_j)$+M. Go to step 2.

2. If $f(\mathbf{y'}_j)<f(\mathbf{y}_j)$, the exploratory move is a success, go to step 5. Otherwise the exploratory move is a failure; go to step 3.

3. Set $y'_j=y_j-s_j$. If $y'_j\leq x_{j.min}$ then $f(\mathbf{y'}_j)=f(\mathbf{y}_j)$+M. Go to step 4.

   If $y'_j>x_{j.min}$ and $pol\neq$PAC, calculate $f(\mathbf{y'}_j)=f(\mathbf{y}_j-s_j\mathbf{d}_j)$. Go to step 4.

   If $y'_j>x_{j.min}$ and pol=PAC and y'_j is none of r_p and y'_j is none of k_p, calculate $f(\mathbf{y'}_j)=f(\mathbf{y}_j-s_j\mathbf{d}_j)$. Go to step 4.

   If $y'_j>x_{j.min}$ and pol=PAC and y'_j is one of r_p, perform checks for conditions C1 and C2: {for each u of product p (no. of u =$NSUB_p$), if $NASS_u$ =1, set $r_d=y'_j$ and check C1; otherwise set one of $r_{d_q}=y'_j$ and check C2; set $r_u=y'_j$ and if $NASS_p$ =1 check C1; otherwise check C2}. If $y'_j>x_{j.min}$ and pol=PAC and y'_j is one of k_p, perform checks for conditions C1 and C2: {for each u of product p (no. of u =$NSUB_p$), if $NASS_u$ =1, set

$k_d = y'_j$ and check C1; otherwise set one of $k_{d_q} = y'_j$ and check C2; set $k_u = y'_j$ and if $NASS_p = 1$ check C1; otherwise check C2}. If y'_j satisfies all performed checks on C1 and C2 conditions, calculate $f(\mathbf{y}'_j) = f(\mathbf{y}_j - s_j \mathbf{d}_j)$; otherwise $f(\mathbf{y}'_j) = f(\mathbf{y}_j) + M$. Go to step 4.

4. If $f(\mathbf{y}'_j) < f(\mathbf{y}_j)$, the exploratory move is a success, go to step 5. Otherwise the exploratory move is a failure; go to step 6.
5. For each y'_j equal to one of r_p, if $y'_j + s_j > y'_{j-1}$, set $x_{j.max} = y'_{j-1}$, and for each y'_j equal to one of k_p, if $y'_j - s_j < y'_{j+1}$, set $x_{j.min} = y'_{j+1}$. Let $\mathbf{y}_{j+1} = \mathbf{y}'_j$.
6. If $j < N$, replace j by $j+1$, and go to step 1 (continue exploratory search). Otherwise, set $x_{j.max} = x'_{j.max}$ and $x_{j.min} = x'_{j.min}$, and if $f(\mathbf{y}_{j+1}) < f(\mathbf{x}_h)$ go to step 7, and if $f(\mathbf{y}_{j+1}) \geq f(\mathbf{x}_h)$ go to step 8.
7. Make a pattern move. Let $\mathbf{x}_{h+1} = \mathbf{y}_{j+1}$, let $j=1$, and let $\mathbf{y}_j = \mathbf{x}_{h+1} + (\mathbf{x}_{h+1} - \mathbf{x}_h)$. If $y_j > x_{j.max}$ then set $y_j = x_{j.max}$, and if $y_j < x_{j.min}$ then set $y_j = x_{j.min}$. For each y_j equal to one of r_p, if $y_j > y_{j-1}$, set $y_j = y_{j-1}$. Replace h by $h+1$ and go to step 1.
8. If $s_j \leq \varepsilon$, stop; $\mathbf{x}_h$ is the solution. Otherwise replace s_j by s_j / s_j. Let $j=1$, and let $\mathbf{y}_j = \mathbf{x}_h$, $\mathbf{x}_{h+1} = \mathbf{x}_h$; replace h by $h+1$ and go to step 1.

End While

Output

- Summary of all input data.
- Sequence of iterations with a best solution.
- Maximum and minimum values of PAC parameters used during the optimization process.
- Complete simulation output for the best solution found.

The structure of the *opk* variant of the algorithm is similar to the *opt*, except that it considers PAC parameters in sequence z_j, r_j, for product j ($j=1,...n$) and sets a base point of total $N=2*n$ parameters.

Opi algorithm considers only z_j parameters for product j (j=1,...n) and sets a base point of total n parameters. It does not require any feasibility checks for parameters r_j and k_j, as all r_j values are required to be 1.

6.7 Optimization Routines and Report

The main optimization program (*opt*, *opk* or *opi*) operates according to the basic algorithm procedure described in section 6.6 with specific characteristics for each optimization variant. The program routines used for PAC simulation (refer to 5.5) are in most cases just slightly adjusted to be used as routines in the optimization model.

For listings of main programs, selected routines, declaration files, as well as for a definition of all additional variables used for the PACOPT optimization program, refer to Appendix B1.

The optimization program was compiled with AIX XL FORTRAN by using "-O3" optimization level, and all optimization runs were executed on the IBM RISC/6000 model computer.

For illustration purposes, Appendix B2 contains also some characteristic parts of the optimization reports for optimization of our example manufacturing configuration (refer to Figure 5.2) for PAC, Kanban and IC optimization.

6.8 PACRAN Optimization Algorithm

In this section we describe a random optimization algorithm for the PAC system.

The PAC optimization model based on the pattern search of Hooke and Jeeves produces local optima. We need some way of verifying how well the hill climbing procedure performs. Therefore, we developed another algorithm, which will be referred to as the PACRAN algorithm. It performs the optimization by direct search utilizing a random number generator in combination with a systematic reduction of the size of the search area and the number of function evaluations in the search region. This algorithm is relatively easy to implement and it can deal with constraints similarly than PACOPT by not allowing function evaluation for unfeasible solutions.

Before describing the specifics of a random algorithm to the PAC models, first, we give a description of a general context of the technique.

6.8.1 Random Optimization Method

Random optimization methods, often referred to as Monte Carlo techniques, are concerned with using randomly generated points to obtain the optimal solution. The main principle of any random optimization procedure is to generate a sequence of independent, uniformly distributed points in the feasible region and to return the best point as an approximation to the optimal solution. These methods are especially suitable for optimization of nonlinear problems, where the properties of the optimized function are unknown and the passage of time plays no substantive role.

For the PAC problem, we apply a random optimization procedure based on the method proposed by Luus and Jaakola (1973). This method for minimization of a function $\{f(\mathbf{x}); \mathbf{x} \in \mathbf{X}$, where $\mathbf{X}$ is a feasible region$\}$ is as follows:

1. Choose an initial point and a range for each variable x_i (i=1,...,n); denote these by $x_i^{o\,(0)}$ and $r_i^{(0)}$. Choose maximum number of iterations J and maximum number of generated sets K of n random numbers per iteration. Set the iteration index j to 1.
2. Read K sets of n random numbers from a uniform distribution U(-0.5,0.5); denote these by y_{ki}.
3. Calculate each variable x_i as: $x_i^{(j)} = x_i^{o\,(j-1)} + y_{ki}\, r_i^{(j-1)}$.
4. Evaluate $f(\mathbf{x})$ for each of K sets.

5. Find the set which minimizes $f(\mathbf{x})$; denote this set by $x_i^{o\,(j)}$; replace j by $j+1$.
6. If the number of iterations has reached the maximum allowed, stop the procedure.
7. Reduce the range by an amount ε: $r_i^{(j)} = (1-\varepsilon)\, r_i^{(j-1)}$; $\varepsilon>0$
8. Go to step 2 and continue.

The method is illustrated for two parameters and 3 iterations in Figure 6.2. Dotted frames represent the consecutive (reduced) feasible regions per iteration, where sets of solutions can be generated. Points P1 to P3 indicate the best result per each iteration.

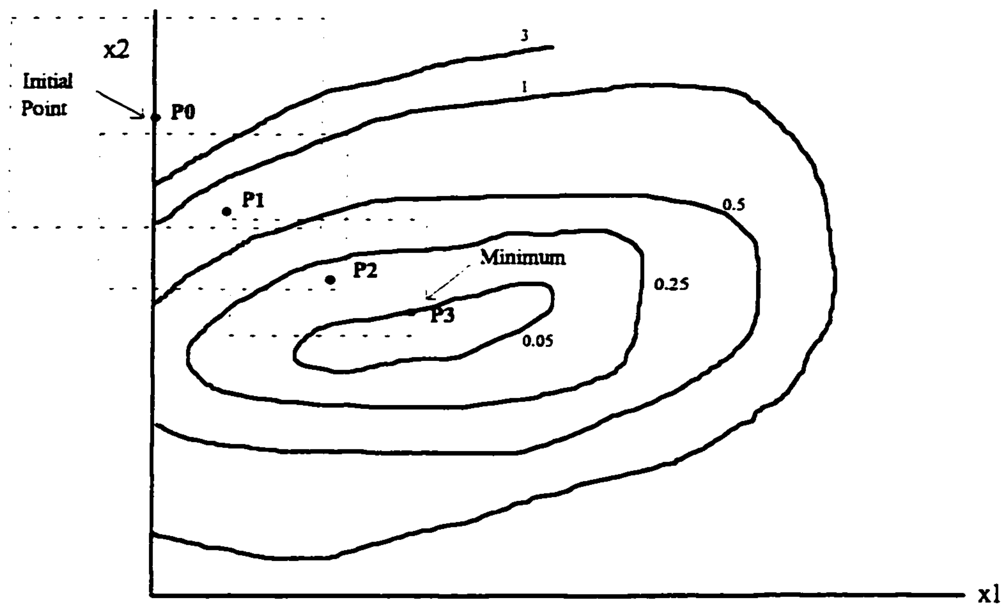


Figure 6.2 Illustration of the random search proposed by Luus and Jaakola.

6.8.2 Structure of the Random Algorithm for the PAC Model

The PACRAN algorithm is based on the random search principle proposed by Luus and Jaakola. The dimensions of an initial search region, referred to as box dimensions depend on the initial range values of the PAC parameters. The initial solution lies in the middle of the initial box. The parameter values are uniformly generated in the box space and approximated to integers. If the generated value for a parameter lies in infeasible region, we continue to generate next value for the same parameter until it is in accordance with parameter limits (minimum and maximum values) and additional feasibility requirements for k and r parameters, if applicable. The number of function evaluations within the box is predetermined. After completion of the search in the initial box, the box dimensions and the number of function evaluations are reduced by predetermined reduction values. The whole procedure is repeated for the second box and the succeeding boxes until the number of required search boxes has been reached, or until no more reduction of box dimensions or function evaluation is possible.

We developed so far only one variant of the PAC random optimization algorithm, referred to as *rop*, which can be used for optimization of all 4*n PAC parameters for policies BSS, MRP, PTO variant with $\tau \geq 0$ and LC. Creating algorithm variants for optimization of parameters for PAC, Kanban and IC/CONWIP requires some adjustments of the *rop* algorithm.

The main structure of the random algorithm is as follows:

PACRAN Optimization Algorithm:

Input

- $\mathbf{x}_0$ initial optimum point with integer values of total N=4*n PAC parameters;
  $\mathbf{x}_0 = (z_1, k_1, r_1, \tau_1, \ldots, z_n, k_n, r_n, \tau_n) = (x_1, \ldots x_N)$
- $z_{p,min}$ minimum value for z_p PAC parameters, (p=1,...,n)
- g_j range for PAC parameters, (j=1,...N)
- $g_{j,min}$ minimum value of the range for PAC parameters, (j=1,...N)
- s_j reduction value for the range of PAC parameters, (j=1,...N)
- FE_{max} maximum no. of function evaluations (=PAC simulations) per box

- NB_{max} maximum no. of boxes
- Δ reduction value for no. of cost evaluations per each new box

Initialization

- Let U(a,b) denote a random number from a uniform distribution with a range [a,b]
- Calculate cost function: $f(\mathbf{x}_0)$
- Calculate the dimensions of the first box for each parameter j: $x_{j,max}=x_j+g_j*0.5$; $x_{j,min}=x_j-g_j*0.5$
- Set counters: NB to 1, FE to 0.
- Let $h=0$, $p=1$

Main Loops

<u>While</u> no. of boxes (NB) < NB_{max}

<u>While</u> no. of function evaluations (FE) < FE_{max}

1. Generate value for k'_p parameter: $k'_p=\lfloor U(x_{4p-2,min},x_{4p-2,max})\rfloor$. If $k'_p<1$ repeat step 1. If $p<n$, replace p by $p+1$, and go to step 1; otherwise let $p=1$; go to step 2.
2. Generate value for r'_p parameter: $r'_p=\lfloor U(x_{4p-1,min},x_{4p-1,max})\rfloor$. If $r'_p<1$ and $r'_p>k'_p$ repeat step 1. If $p<n$, replace p by $p+1$, and go to step 2, otherwise let $p=1$; go to step 3.
3. Generate value for z'_p parameter: $z'_p=\lfloor U(x_{4p-3,min},x_{4p-3,max})\rfloor$. If $z'_p<z_{p,min}$ repeat step 3. If $p<n$, replace p by $p+1$, and go to step 3; otherwise let $p=1$; go to step 4.
4. Generate value for τ'_p parameter: $\tau'_p=\lfloor U(x_{4p,min},x_{4p,max})\rfloor$. If $\tau'_p<0$ repeat step 4. If $p<n$, replace p by $p+1$, and go to step 4; otherwise let $p=1$; go to step 5.
5. Calculate cost function: $f(\mathbf{x}_{h+1})$; $\mathbf{x}_{h+1}=(z'_1, k'_1, r'_1, \tau'_1,\ldots, z'_n, k'_n, r'_n, \tau'_n) = (x'_1,\ldots,x'_N)$. If $f(\mathbf{x}_{h+1})<f(\mathbf{x}_h)$ then $\mathbf{x}_h=\mathbf{x}_{h+1}$. Replace h by $h+1$. Increment FE by 1.

<u>End While</u>

6. Set $FE_{max} = FE_{max} - \Delta$. If $FE_{max}<1$, stop. Increment NB by 1. If $NB>NB_{max}$, stop. For each parameter j set $g_j=g_j-s_j$. If any $g_j< g_{j,min}$ then set $g_j=g_{j,min}$.

Calculate the dimensions of the next box for each parameter j: $x_{j,max}=x_j+g_j*0.5$; $x_{j,min}=x_j-g_j*0.5$.

End While

Output

- Summary of all input data.
- Sequence of random iterations with a best solution.
- Maximum and minimum values of PAC parameters used during the optimization process.
- Complete simulation output for the best solution found.

6.8.3 Optimization Program

The optimization program *rop* operates as previously described PAC optimization programs (*opt*, *opk* or *opi*) together with a set of routines of a PAC simulation model.

For listing of the main program and a definition of specific variables of the PACRAN algorithm refer to Appendix B3.

The optimization program was compiled with AIX XL FORTRAN by using "-O3" optimization level, and all optimization runs were executed on the IBM RISC/6000 model computer.

For illustration purposes, Appendix B3 contains also some characteristic parts of the optimization report generated by the PACRAN algorithm for the case of our example manufacturing configuration (refer to Figure 5.2) for a case of the MRP scheme.

Chapter 7

OPTIMIZATION RESULTS

7.1 Introduction

In this chapter we illustrate the use of the PAC optimization algorithm (PACOPT) in the case of 5 different manufacturing configurations (models), ranging from a relatively simple one (8 PAC parameters) to a quite complex one (44 PAC parameters). We also demonstrate the use of the random optimization algorithm (PACRAN) in the case of 2 models.

Before presenting the specific characteristics of each model, we summarize some general considerations and assumptions for our optimization study of all 5 models. After a description and a brief evaluation of results of each model, we compare the PAC optimization findings to the theoretical formula related to the BMAP/G/1 queues. We also justify the single simulation run approach used as a cost function evaluation in an overall optimization by presenting some numerical results. Finally, we draw some conclusion on the effectiveness of the optimization algorithm and the significance of the results obtained.

7.2 Design of Experiments

The development of the PAC optimization algorithm was done simultaneously with performing test runs for different manufacturing configurations. We wanted to apply the PAC algorithm to various production environments in order to estimate its performance and applicability. We have designed a series of experiments, starting with a relatively simple production layout, systematically increasing the degree of complexity of each successive model.

The objective of the experiments was not only to test and develop the optimization algorithm itself, but also to perform comparison studies for different coordination schemes. We developed 3 variants of the algorithm (refer to 6.5.6), which provide the means for the optimization of parameters for 9 different coordination policies (PAC-general, PTO, PTO-variant with $\tau \geq 0$, MRP, BSS, LC, Kanban, IC, CONWIP). This enabled us to examine the effectiveness of those policies in cases of different production layouts and various production scenarios.

7.2.1 Choice of Models

During our study, the following 5 manufacturing configurations were chosen as test cases for performing a series of experiments:

1. two-cell flow line;
2. three-cell flow line;
3. two-cell system with an assembly cell;
4. four-cell system with multi functional cells;
5. six-cell system with specialized (processing or assembly) cells.

Figure 7.1 shows the layout configurations of all 5 models.

First, we tested the PAC optimization algorithm on the very simple model 1. Buzacott and Shanthikumar (1992b) studied this model to develop an approximation procedure to evaluate the system performance, and we also used this configuration for verification of the PAC simulation program. Model 1 can be analyzed using queueing theory, and therefore is attractive for testing the optimization algorithm. Next, we extended model 1 with an additional processing cell and obtained model 2, which is controlled by 12 PAC parameters, instead of 8, as in the case of model 1. The next model, model 3, is the simplest configuration which allows for both processing and assembly operations, as well

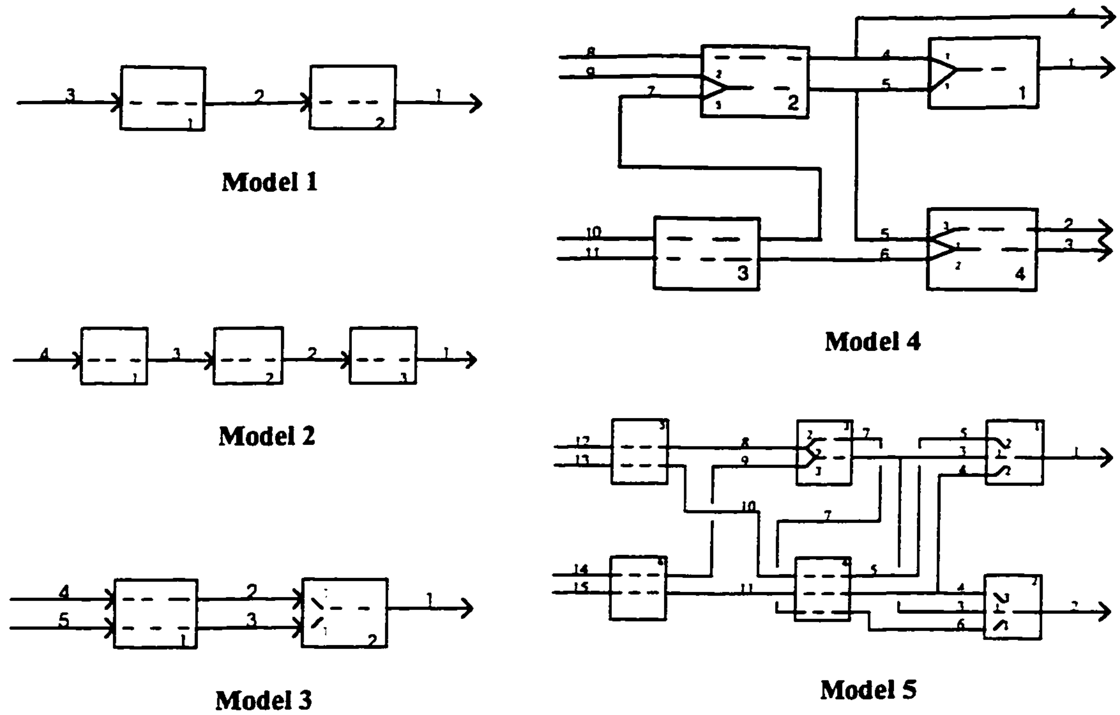


Figure 7.1 Overview of layouts of all 5 models.

as an examination of the impact of setup times. A similar model considering assembly operations was also briefly studied by Buzacott and Shanthikumar (1992b). In general, this production layout is still simple enough to allow some theoretical analysis. Model 4 is the example configuration used to develop and explain the simulation procedure (refer to chapter 5). This model contains all aspects of a complex manufacturing layout that are implemented in the PAC simulation model. It results in an optimization problem of 28 PAC parameters. The last model, model 5, was based on the simulation model found in studies by Philipoom et al. (1987), which are related to an examination of the factors influencing the number of kanbans required in the implementation of the JIT technique. Model 5 is the most complex manufacturing configuration used in our study. It requires an optimization of 44 PAC parameters.

7.2.2 Series of Experiments

Each specific model case is optimized for 9 different coordination policies: Production Authorization Cards (PAC), Produce-to-Order (PTO), Produce-to Order, variant with $\tau \geq 0$ (PTO-$\tau \geq 0$), Material Requirements Planning (MRP), Base Stock System (BSS), Local Control (LC), Kanban, Integral Control (IC), Constant Work-in Process (CONWIP).

For the results presented in this chapter, the processing times at machines are modeled as either exponentially or uniformly distributed with a mean processing time MSERVT. For all models, the processing time is generated as a uniform random variable between *a* (=MSERVT-1 minute) and *b* (=MSERVT+1 minute). This gives a variance always equal to 0.33, which is virtually deterministic.

The arrival of customer demand is modeled as a Poisson process with mean MARRVT. In order to consider standard Poisson demand versus bulk demand, the arriving batch size of customer demand is generated in two different ways. One approach assumes that every customer demands a single unit of a product. This can be thought of as the batch size of an arriving demand is constant and equal to 1. A second approach generates bulk arrivals. There are many ways to model the bulk arrival. We chose to generate the batch size modeled as a geometric distribution with mean equal to 1 since this is easy to simulate and leaves the mean arrival rate unchanged. As a result many of the arrival of customer demand are actually pseudo-arrivals, since the probability that a geometric random variable x takes on the value x=0 is 0.5 if $E(x)$=1. In the remainder of this thesis refer to these two ways of modeling demand as the Poisson standard and bulk arrival process, respectively.

For all 5 models, we performed a series of experiments for exponential processing times and standard arrival process, exponential processing time and bulk arrival process, and uniform processing times and standard arrival process. Due to the limitation of time, we did optimizations for the case of uniform processing times and bulk arrival process only for models 1 and 4.

In the case of model 3, 4 and 5, we studied the effects of setup times on different policies. In the case of model 2, we performed additional optimization runs for different values of the customer delay cost.

7.2.3 Length of the Simulation Run

For each optimization run of every model, we evaluate the performance of the given situation, characterized by a set of PAC parameters, by a single simulation run of the same length. Having the same length of all simulation runs will facilitate the comparative evaluation of different systems. We choose to run each simulation for a total of 300 days of 24 working hours each: a warm-up time of 40 days plus an evaluation period of 260 days. This is an arbitrarily chosen length. However, we did some statistics on the length of the simulation, and on the values of the generated results, together with registration of the required computer time. We did runs with different number of days (20, 40, 60,..., 260, 280, 300, 600, 10000, 100000) and compared the results. The first 40 days of each simulation produced much higher values for different measures of performance (product delay time to customer, inventory in stores and WIP), so we implement in each simulation a warm-up period of 40 days. Results from simulations over 200 days and up were similar to the longest simulation of 100000 days. We choose 260 days as an evaluation period. This period is equivalent to a yearly production process (52 weeks/year x 5 days/week) and seems a good approach for studying a manufacturing facility. It is also reasonably long run to justify that the results of a given simple path are fair approximation of the function value (refer to 6.3). A single simulation run of this length is fast enough to make optimizations of all 5 models executable in a reasonable duration, and allows performing quite a large number of optimizations for a variety of situations, for each tested model.

7.2.4 Cost Factors and Cost Scenarios

During the PAC simulation run, various measures of performance are combined into one total cost using appropriate cost factors (refer to 5.6). The total cost is a sum of customer delay costs, inventory holding costs in stores and inventory holding costs in cells. To calculate these costs, we need to specify three basic cost factors: delay costs, inventory holding costs in stores and cells. The choice of cost factors is problematic, especially if we want to make the comparative studies of different models. Cost factors are set arbitrarily and we are aware that the different cost factors will result in different values of the total cost. We set the costs factors according to the following procedure:

- inventory holding cost factors (in stores) for raw materials are 1 or 2 ($/day/item);
- inventory holding cost factors per product increase with each next stage of the production process; for models 1 and 3, they are 2 and 4 ($/day/item) and for model 2, they are 1, 2 and 4 ($/day/item) respectively; in the case of models 4 and 5, generally, the inventory holding costs after processing or assembly activities are the sum of inventory holding costs of all components +1 ($/day/item);
- delay cost factors are set to 10, 8, 6, 5 ($/day/item) for final product 1, 2, 3, 4 respectively (when applicable). In the case of models 1, 2 and 3, the delay cost factor of 10 $/day/item (those models have only one final product) may seem large in comparison with inventory holding cost factors. Therefore, in the case of model 2, we performed optimization runs for a value of the delay cost factor equal to 1.

For each coordination policy, the optimization is performed for different delay cost scenarios (refer to 5.6): delay costing option I - DCI and delay costing option II - DCII (when applicable). In the case of DCII, all "small" costs are calculated with 5% of the customer delay cost factors, and the "large" ones with 100%.

7.2.5 Number of Machines

In the case of all the manufacturing layouts studied, the number of machines (identical servers) at cells equals to 1 ($c_i = 1$, $i=1,...,m$). The parameter setting for the LC policy requires: $z>c$, and $k=c$. This results in minimum values of all parameters z equal to 2, and k parameters equal to 1 for the LC policy of all models.

7.2.6 Generated Results

The results of our experiments are summarized in Appendices C1 to C5. For each policy, they indicate the total cost (in $), number of performed function evaluations (PAC simulations) and the optimal values of parameters (z, k, r, τ) for product j ($j=1,...,n$). As Buzacott and Shanthikumar discuss (1992b), the MRP, BSS, IC, PTO, CONWIP, Kanban and LC strategies imply restrictions on the parameters choices. What we show are the (approximately) optimized values of the parameters within each of these restricted classes together with the general optimization PAC results.

"M" is sometimes substituted for the number of process tags. It is a large number and is equivalent to setting the process tags to an unlimited quantity. For each tested model M-value is different. We did assume the M-value for each model and then checked the minimum length of the PROC queue by running a single PTO simulation. If the PROC queue length was equal to 0 that indicated, that there might be not sufficient number of process tags. The M-value had to be increased and the PTO simulation was performed again as a check. The M-value for models 1, 2, 3, 4 and 5 equals 100, 100, 100, 220 and 250 respectively.

The initial values of all parameters were set to the minimum values with step lengths usually given by:

- for z parameters = 1 or 2;
- for k parameters = 1, 2 or 5;
- for r parameters = 1 or 2;
- for τ parameters = 5, 10 or 20 (minutes).

For most of the cases for PAC, MRP and PTO-$\tau \geq 0$, two or more optimization runs were executed for different step lengths of some parameters (mostly step lengths of τ=10 or 20, sometimes in combination with different step lengths of z or k).

Results indicated with "*" were obtained by setting the initial values of parameters to the optimal values obtained during the optimization procedure of another policy scheme. Often it was done to improve the MRP optimization results by using the PTO-$\tau \geq 0$ results. It also happens, for some cases of the general PAC policy, when the best optimization results obtained so far by some other policy were used as initial values for the PAC policy optimization.

In the case of a standard arrival process, for interarrival times, product types and processing times the stream numbers 1, 2 and 3 respectively, have been used (refer to 5.4). In order to get approximately the same number of customer arrivals, when running the optimizations with bulk arrival process, we had to find the appropriate stream number for a geometric distribution for each model. Again, a single simulation run for the PTO policy was used. The search runs were executed with stream numbers varying from 1 to 100 (as SIMLIB contains 100 different values for stream numbers).

7.3 Model 1

7.3.1 Description of Model 1

Figure 7.2 presents the layout of the first manufacturing configuration considered in our study. This model is a very simple flow line, where a single product undergoes processing operations on two consecutive machines. Translation of this production layout into a PAC model results in Table 7.1.

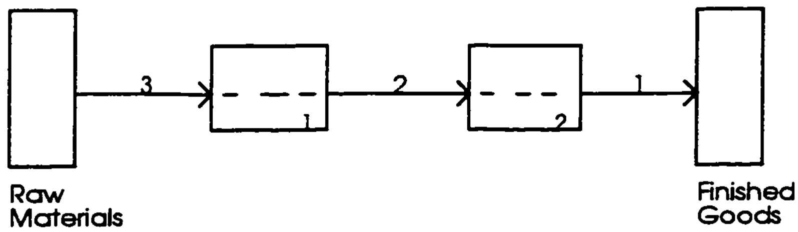


Figure 7.2 Layout of model 1.

Table 7.1 Input data for model 1

Product Number	1	2	3
Product Kind	*f*	*a*	*r*
Production Cell	2	1	-
Mean Processing time (min.)	$1/\mu_1$	$1/\mu_2$	-
Number of Components	1	1	-
Names of Components	2	3	-
Probability of an arriving demand being for a product *j*	1	-	-
Transportation Time from Store to Cell (min.)	-	0.01	0.01
Customer Delay Cost (S/day/item)	10	-	-
Inventory Holding Cost at Store (S/day/item)	4	2	-
Inventory Holding Cost at Cell (S/day/item)	-	4	2

As n=2, this model has a maximum of 8 PAC parameters for optimization. The input data, given in Table 7.1, were chosen somewhat arbitrarily and form a specific instance for the model 1. Model 1 is characterized by: $m = 2, f = 1, fa = 0, n = 2, r = 1$; and requires 23 lists and 7 SAMPST variables (refer to 5.4).

The mean customer demand arrival rate is $\lambda = 1$ customer/hour; that means that the mean interarrival time MARRVT = 60 minutes. We consider the cases for a mean processing time equal to 6 or 42 minutes per product j, which give the utilization factor ρ_j equal to 0.1 and 0.7 respectively ($\rho_j = \lambda * 1/\mu_j$). The expected number of arrivals of customer demand is 6240, since the evaluation time per each simulation run is 6240 hours (260 days * 24 hours/day). We use a single arrival stream (stream number 1) for all simulation runs, which results in a generation of approximately 6292 arrivals. In the case of bulk arrival of demand, the stream number for geometric distribution was set to 5.

7.3.2 Evaluation of Results for Model 1

Appendix C1 contains 4 tables with optimization results for different scenarios for model 1. We consider the following 4 combinations of mean processing times (in min.) for products [product 1, product 2]: [6,6], [42,6], [6,42] and [42,42]. The longest optimization (PAC, 430 iterations, Table c1-4) took 323.3 CPU seconds. The various PTO simulations required between 0.6 and 1.0 CPU seconds.

As must be the case, the general PAC runs give the best results, since they have the fewest restrictions. It is interesting to compare how the various types of control schemes vary in their performance for the different scenarios. Values of the total costs for different policies are much lower for cases with uniform processing times in comparison with exponential ones, and for standard Poisson arrivals of a customer demand. The bulk demand increases costs significantly. Costs for PAC, MRP and PTO-$\tau \geq 0$, in the case of a delay costing option II are lower than in the case of a delay costing option I approach. DCII allows for the implementing of forecasting, and calculates relatively small cost for a delay in meeting customer demand in the time lag between issuing of an order tag and receiving of a requisition tag. The optimal values of τ parameters for those polices for "final" products are also significantly different for both approaches, as previously expected and explained (refer to 5.6).

To facilitate a more detailed evaluation, we summarize the results in the form of bar charts for two main scenarios of processing times, exponential and uniform (refer to Figures 7.3 and 7.4), using the delay costing option II (DCII) approach. Each figure shows results for two cases of the system utilization; low, with the mean processing times on both machines equal to 6 min., and high, when the mean processing times are 42 min., and with a combination of standard and bulk demand. The costs are normalized with the total cost generated by PAC set to 1, and the costs of all other policies proportional to the cost of the PAC policy. For all policies presented in Figure 7.3, we give, in Table 7.2, a percentage breakdown of different costs categories, as well as the average level of inventory in stores, in WIP, customer delay probabilities, and the average final product delay to customer.

Note the similarity of all charts for the case of low system utilization and the similarity of all charts for the case of high system utilization. For the case of low system utilization, the sequence of the first 6 policies and the "worst" one are exactly the same for all 4 charts. The high utilization system does not show large differences between the costs of the policies in the sequence, except the "worst" one, which is always the PTO. Because PTO has fixed values for all parameters, it has no ability to adjust itself to the demands of any system.

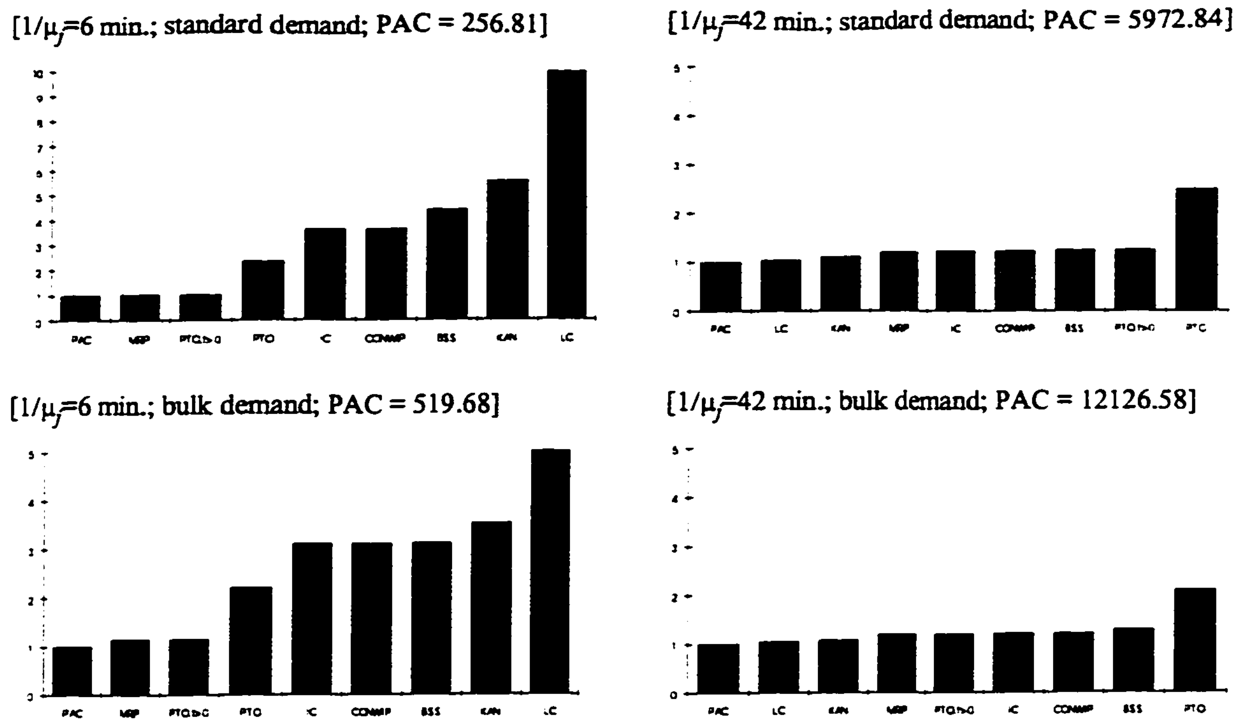


Figure 7.3 Model 1: DCII, exponential processing times.

In all cases of systems with significant over-capacity, PAC, MRP and PTO-τ>0 perform almost equally well, and notably better than the other policies. This can be explained by the implementation of forecasting by those schemes, which reduces the total customer delay cost, and by still relatively small WIP holding costs, due to the short processing times of the products in the system. Among PAC, MRP and PTO-τ>0, the PAC policy has the lowest WIP holding costs. This is due to limits on process tags, and Kanban-like effects on the WIP. IC, CONWIP, BSS and Kanban require some or all values of initial inventory (parameters z) to be more than 0, which increases the inventory holding costs in stores for those policies. LC has the worst performance, as all values for z parameters have to be more than 1. The main component in the total cost of LC is the inventory holding cost in stores, which is almost 100%, and 93% for the case of exponential processing times with standard demand and bulk demand respectively (refer to column I and II in Table 7.2).For the LC policy, the WIP holding costs are always equal to 0, when only 1 product is processed per each cell. PTO does not require initial inventory (for PTO all inventory holding costs in stores are always equal to 0), but also does not implement forecasting, and therefore does better than policies IC, CONWIP, BSS, Kanban and LC, but worse than MRP.

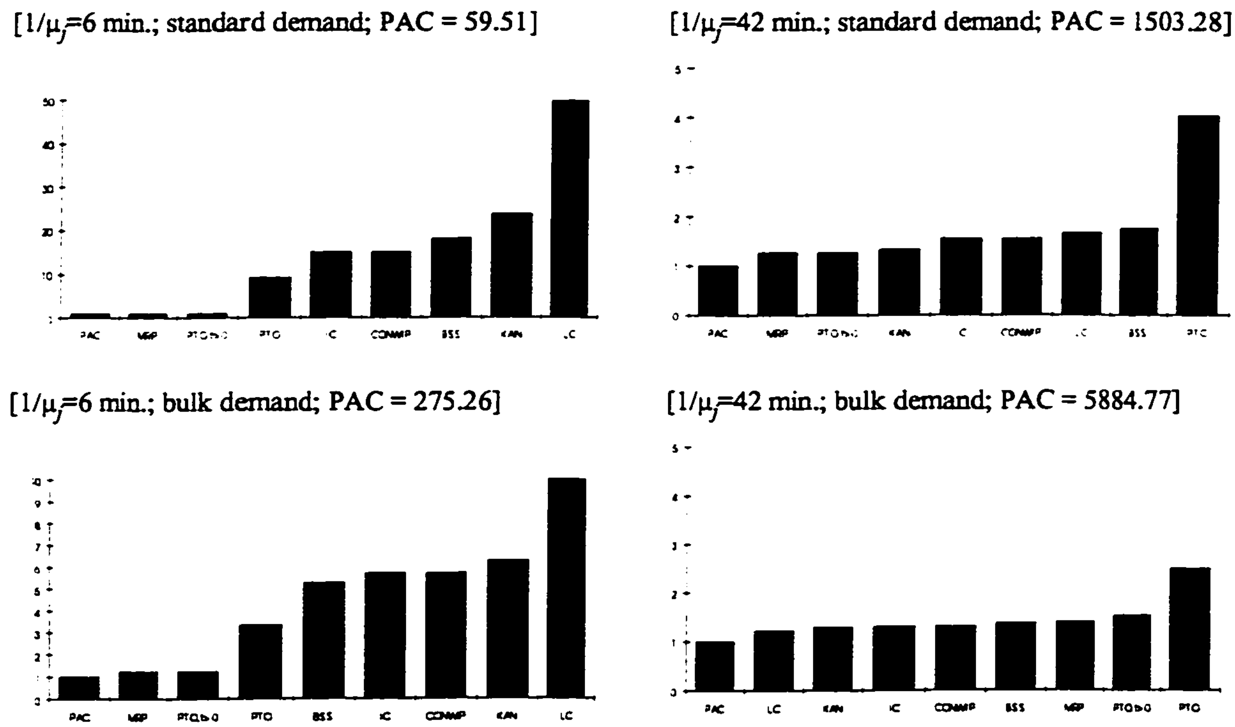


Figure 7.4 Model 1: DCII, uniform processing times.

In the case of systems with large mean processing times on machines, all policies perform equally well, except PTO, for which the longer processing times result in both larger WIP holding costs and customer delay costs (refer to column III and IV in Table 7.2). Generally, LC, Kanban and IC are slightly better than MRP.

Table 7.2 Model 1: additional data for results presented in Figure 7.3 (DCII, exponential processing times)

Best Solution	Legend	low utilization		high utilization	
		standard demand (I)	bulk demand (II)	standard demand (III)	bulk demand (IV)
1	Policy Cost a1 - a2 - a3 b1 - b2 / c - d z_1 - k_1 - r_1 - τ_1 z_2 - k_2 - r_2 - τ_2	PAC 256.81 34 - 4 - 62 (=48+14) 0.1 - 0.01 / 0.7 - 2.8 0 3 1 16 0 1 1 4	PAC 519.68 31 - 12 - 57 (=46+11) 0.2 - 0.1 / 0.7 - 5.4 0 11 1 27 0 1 1 1	PAC 5972.84 41 - 23 - 36 (=35+1) 2.9 - 1.4 / 0.7 - 47.5 4 15 1 28 2 1 1 32	PAC 12126.58 33 - 17 - 50 (=42+8) 4.9 - 1.9 / 0.7 - 117.2 0 7 1 439 4 1 1 60
2	Policy Cost a1 - a2 - a3 b1 - b2 / c - d z_1 - k_1 - r_1 - τ_1 z_2 - k_2 - r_2 - τ_2	MRP 262.43 33 - 7 - 60 (=47+13) 0.1 - 0.02 / 0.7 - 2.8 0 M 1 16 0 M 1 4	MRP 591.93 27 - 22 - 51 (=41+10) 0.2 - 0.2 / 0.7 - 5.5 0 M 1 27 0 M 1 0	LC 6226.76 67 - 0 - 33 6.0 - 0 / 0.7 - 47.5 4 1 1 0 6 1 1 0	LC 12788.70 54 - 0 - 46 9.8 - 0 / 0.6 - 134.3 7 1 1 0 9 1 1 0
3	Policy Cost a1 - a2 - a3 b1 - b2 / c - d z_1 - k_1 - r_1 - τ_1 z_2 - k_2 - r_2 - τ_2	PTO, τ≥0 262.43 33 - 7 - 60 (=47+13) 0.1 - 0.02 / 0.7 - 2.8 0 M 1 16 0 M 1 4	PTO, τ≥0 591.93 27 - 22 - 51 (=41+10) 0.2 - 0.2 / 0.7 - 5.5 0 M 1 27 0 M 1 0	Kanban 6618.27 39 - 23 - 38 3.2 - 1.9 / 0.7 - 57.4 1 1 1 0 3 3 1 0	Kanban 13093.55 39 - 16 - 45 5.0 - 2.0 / 0.7 - 135.8 10 10 1 0 1 1 1 0
4	Policy Cost a1 - a2 - a3 b1 - b2 / c - d z_1 - k_1 - r_1 - τ_1 z_2 - k_2 - r_2 - τ_2	PTO 600.63 0 - 3 - 97 0 - 0.03 / 0 - 13.3 0 M 1 0 0 M 1 0	PTO 1148.33 0 - 11 - 89 0 - 0.2 / 0 - 23.3 0 M 1 0 0 M 1 0	MRP 7104.55 35 - 36 - 29 (=28+1) 2.9 - 3.3 / 0.7 - 46.5 5 M 2 40 2 M 1 50	MRP 14418.85 25 - 34 - 41 (=33+8) 3.8 - 7.0 / 0.7 - 109.2 0 M 1 526 2 M 1 40
5	Policy Cost a1 - a2 - a3 b1 - b2 / c - d z_1 - k_1 - r_1 - τ_1 z_2 - k_2 - r_2 - τ_2	IC 933.36 89 - 0 - 11 0.8 - 0 / 0.8 - 2.4 1 1 1 0 0 1 1 0	IC 1617.75 51 - 0 - 49 0.8 - 0 / 0.4 - 18.0 1 1 1 0 0 1 1 0	IC 7245.95 34 - 21 - 45 3.5 - 2.1 / 0.6 - 75.0 3 3 1 0 4 7 1 0	PTO, τ≥0 14400.48 2 - 34 - 44 (=36+8) 3.1 - 6.8 / 0.7 - 118.7 0 M 1 538 0 M 1 2
6	Policy Cost a1 - a2 - a3 b1 - b2 / c - d z_1 - k_1 - r_1 - τ_1 z_2 - k_2 - r_2 - τ_2	CONWIP 933.36 89 - 0 - 11 0.8 - 0 / 0.8 - 2.4 1 1 1 0 0 1 1 0	CONWIP 1617.75 51 - 0 - 49 0.8 - 0 / 0.4 - 18.0 1 1 1 0 0 1 1 0	CONWIP 7291.88 30 - 27 - 43 2.1 - 6.3 / 0.6 - 72.5 6 6 1 0 0 6 1 0	IC 14682.68 30 - 22 - 48 4.2 - 4.4 / 0.6 - 159.5 10 10 1 0 0 1 1 0
7	Policy Cost a1 - a2 - a3 b1 - b2 / c - d z_1 - k_1 - r_1 - τ_1 z_2 - k_2 - r_2 - τ_2	BSS 1125.41 78 - 5 - 17 0.4 - 0.3 / 0.3 - 10.0 1 M 2 0 1 M 1 0	BSS 1620.75 55 - 8 - 37 1.3 - 0.2 / 0.2 - 13.5 1 M 2 0 1 M 1 0	BSS 7427.41 36 - 33 - 31 3.3 - 3.2 / 0.7 - 53.0 4 M 1 0 3 M 1 0	CONWIP 14682.68 30 - 22 - 48 4.2 - 4.4 / 0.6 - 159.5 10 10 1 0 0 1 1 0
8	Policy Cost a1 - a2 - a3 b1 - b2 / c - d z_1 - k_1 - r_1 - τ_1 z_2 - k_2 - r_2 - τ_2	Kanban 1427.71 98 - 0 - 2 1.8 - 0 / 0.9 - 0.8 1 1 1 0 1 1 1 0	Kanban 1833.40 75 - 0 - 25 1.8 - 0 / 0.4 - 10.5 1 1 1 0 1 1 1 0	PTO, τ≥0 7436.17 25 - 35 - 40 (=30+10) 1.8 - 3.3 / 0.7 - 52.2 0 M 1 334 0 M 1 30	BSS 15565.21 33 - 34 - 33 5.7 - 7.4 / 0.7 - 117.8 9 M 2 0 5 M 4 0
9	Policy Cost a1 - a2 - a3 b1 - b2 / c - d z_1 - k_1 - r_1 - τ_1 z_2 - k_2 - r_2 - τ_2	LC 2948.86 99.9 - 0 - 0.1 3.8 - 0 / 0.99 - 0.08 2 1 1 0 2 1 1 0	LC 3087.60 93 - 0 - 7 3.7 - 0 / 0.7 - 5.0 2 1 1 0 2 1 1 0	PTO 14739.34 0 - 18 - 82 0 - 3.4 / 0 - 277.4 0 M 1 0 0 M 1 0	PTO 25443.59 0 - 19 - 81 0 - 6.8 / 0 - 470.6 0 M 1 0 0 M 1 0

Legend

For each policy are given the total cost (in $) and a set of the following data:

a1 percentage of "inventory in stores" cost in total cost

a2 percentage of "inventory in cells" cost in total cost

a3 percentage of "customer delay" cost in total cost, where for PAC, MRP and PTO-τ≥0, this cost is a sum of "small" and "large" cost, which are given in brackets

b1 time average inventory level in all stores (in all PROD-queues)

b2 time average inventory level in all cells (in all WIP-queues)

c probability that an arriving demand from a customer for a final product is met immediately

d overall average final product delay to customer (in min.)

This can be explained by longer waiting times of products in WIP for the MRP and therefore an increase of inventory holding costs in cells. Smaller delay costs (due to forecasting) cannot compensate for the increased WIP holding costs. It is interesting to observe how much better the LC does in the case of longer processing times. The values of z parameters are larger than the required minimum of 2. Here they can be justified as a safety stock to satisfy the customer demand (refer to column III and IV in Table 7.2). In addition, in the case of a high utilization factor (ρ_j =0.7), LC can be very efficient, as it assures processing in cells if parts and machine capacity are available and the stores are not full. As expected, LC should perform well, if those conditions are satisfied. We see that the high utilization system, with exponential processing times, is equally difficult for all policies. Only where the demand is standard, and processing times are uniformly distributed (in a very small range), does MRP a little better than others, due to lowering delay costs by implementing forecasting and not carrying a safety stock. We expect that this can be different for other costs factors, higher WIP holding costs and lower PROD holding costs.

In Table 7.2, we also indicate the optimal parameters for each policy. It is interesting to observe that the PAC solution, in both cases of low utilization and high utilization, has aspects of both the MRP and Kanban control. In the low utilization system with standard demand, the PAC requisition delays are identical to the MRP settings. The improved performance is due to limiting the numbers of process tags, strongly in the case of product 2 (k_2=1). In the case of high utilization, the ability to include requisition delays and extra process tags give PAC advantages over Kanban. Yet the limits on process tags enables PAC to do better than MRP. For case III, the PAC has about the same delay costs and holding cost for stores inventory as MRP, but has sharply reduced WIP costs in the cells.

Most of the cost advantage of PAC over Kanban is due to reduced customer delay costs although inventory costs are also slightly better.

The results discussed were for the low and the high utilization systems, in which the system is balanced, as both machines have the same mean processing times, 6 min. or 42 min. Figures 7.5 and 7.6 present bar charts for the unbalanced scenarios for exponential and uniform processing times respectively. Each figure shows results for two different cases of mean processing times for product 1 and 2, [6 min. and 42. min] and [42 min. and 6 min.] respectively, and with combination of standard and bulk demand. Case [6, 42] means that the processing times at cell 1 are higher than in cell 2 (product 2 is made in cell 1), and case [42, 6] indicates that the processing times at cell 2 are higher than in cell 1 (product 1 is made in cell 2).

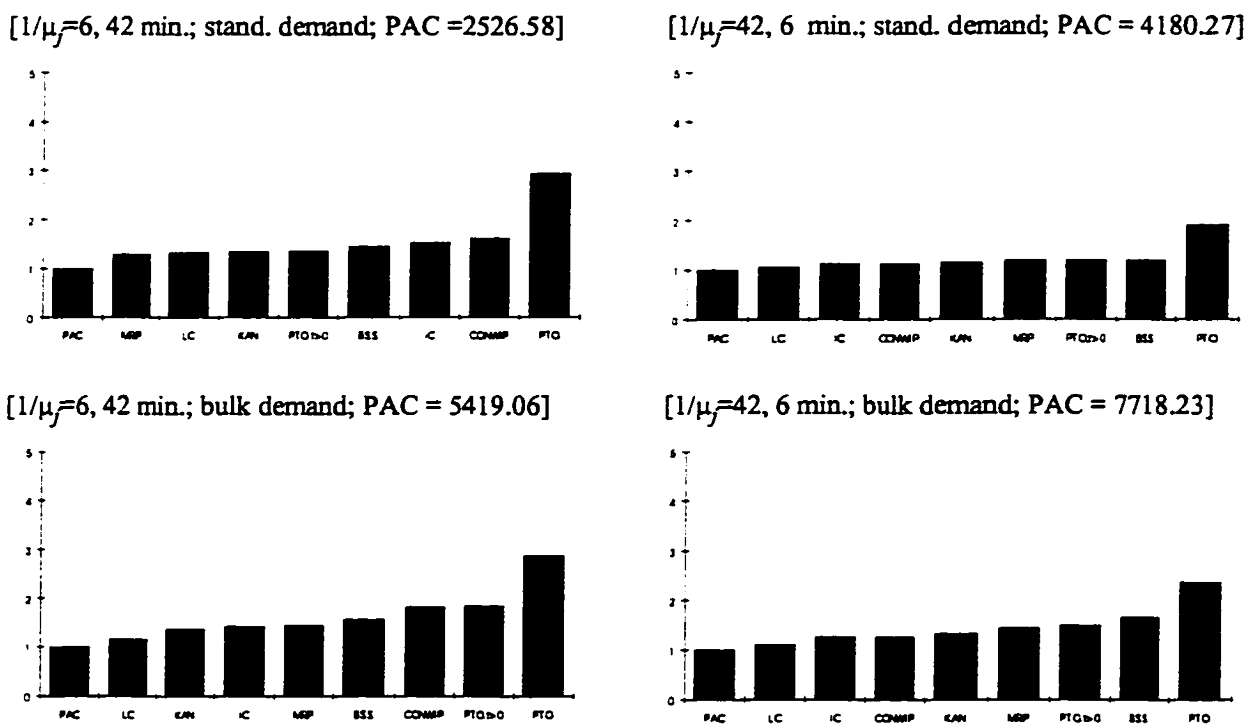


Figure 7.5 Model 1: DCII, exponential processing times, unbalanced.

Generally, we observe some similarities between charts for exponential and uniform scenarios. However, the sequencing of the policies is quite different in each case, except that with mean processing times [42, 6] and bulk demand, where it is identical. All policies do equally well, except PTO which has the worst performance. LC is preferred for all cases with bulk demand. MRP does well in cases of standard demand except the exponential case, when the processing times at the first cell are lower than on the second one. The differences between costs are slightly higher for bulk arrival of demand.

When comparing the results for unbalanced and balanced scenarios, we observe that all charts of unbalanced cases are somewhat similar to charts for high utilization systems, as the differences in costs between policies are relatively small. However, the sequencing of policies differs significantly.

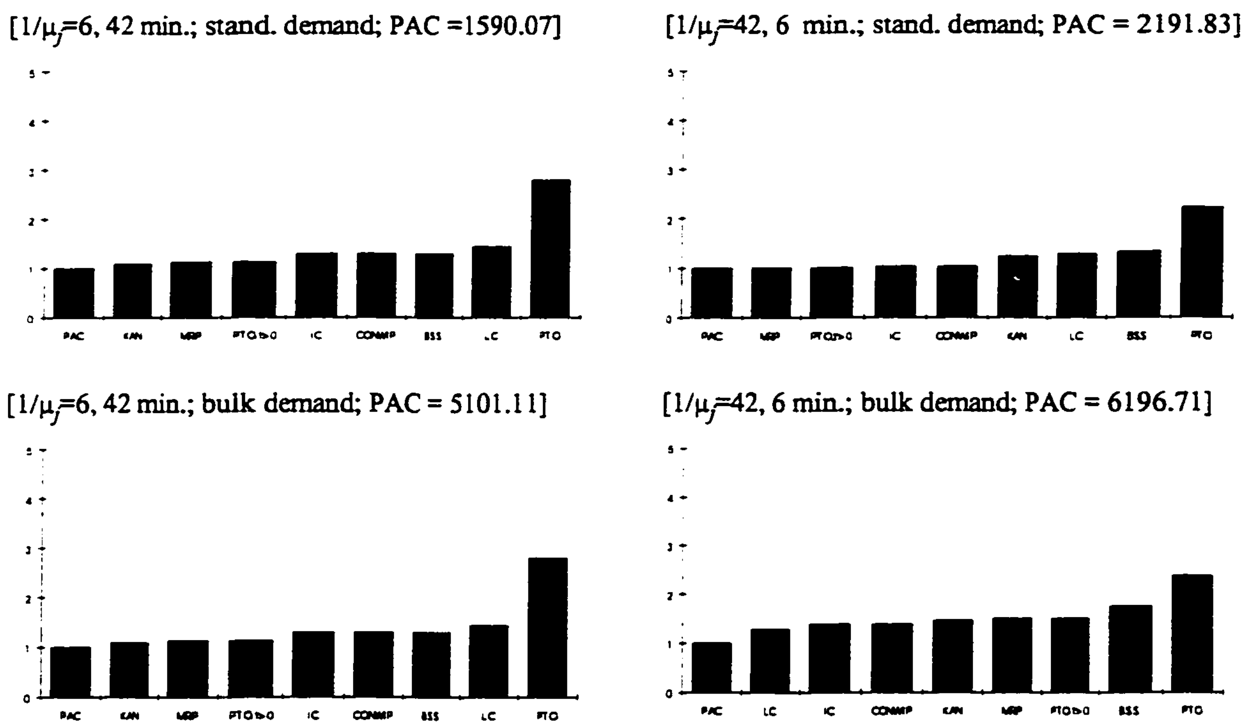


Figure 7.6 Model 1: DCII, uniform processing times, unbalanced.

For cells in series layouts and for policies with $\tau \geq 0$, we established the following relationship between the requisition delay times for n products (j=1,...,n): $\tau_1 \geq \tau_2 \geq \ldots \geq \tau_n$ (refer to 4.2.5.1). Policies that allow for $\tau \geq 0$ and relate the requisition tag delay to the lead times are MRP and PTO-$\tau \geq 0$. Model 1 is a cell in series layout, therefore it is interesting to comment on the obtained values for parameter τ in case of both policies. In the case of a delay costing option I, the values of τ for a "final" product (product 1) are always equal to 0, and for an "assembly" product (product 2) they are more than or equal to 0. Cases with $\tau_2 > 0$ violate the monotonicity relationship. This can be explained by the choice of cost factors, which are lower for product unit 1 than for product unit 2. Thus it is cheaper to keep product 2 in store 1 than in WIP at cell 2. In the case of longer waiting times in WIP, cell 2 issues requisition tags with a delay in order to receive the product some time later. It implies that the requisition delay is not related to the lead time of product 2, but aimed on the reduction of the WIP holding costs. In most of the cases with DCII, the monotonicity relationship between parameters τ is not violated. However, the values calculated of the τ parameters seem often much higher than the expected lead times. This could be explained by the fact that the results obtained are not necessarily the global optimum.

Generally, the model 1 results can be easily explained and understood. We can also expect, that the optimization results, in most cases, obtained the global minimum and therefore are very reliable. This occurs because the number of parameters to be optimized is relatively small and the optimal parameters values are equal to or are relatively close to their minimum values, which also initiated the optimization process. Future research might involve similar experiments with radically changed cost factors to compare the findings and to study the influence of cost factors on the performance of the system and the choice of the policy.

7.4 Model 2

7.4.1 Description of Model 2

Figure 7.7 shows the second manufacturing configuration we considered. This is a flow line, where a single product undergoes processing on three consecutive machines. Translation of this production layout into a PAC model results in Table 7.3.

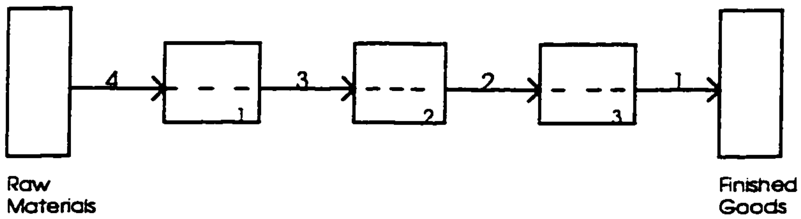


Figure 7.7 Layout of model 2.

Table 7.3 Input data for model 2

Product Number	1	2	3	4
Product Kind	*f*	*a*	*a*	*r*
Production Cell	3	2	1	-
Mean Processing time (min.)	$1/\mu_1$	$1/\mu_2$	$1/\mu_3$	-
Number of Components	1	1	1	-
Names of Components	2	3	4	-
Probability of an arriving demand being for a product	1	-	-	-
Transportation Time from Store to Cell (min.)	-	0.01	0.01	0.01
Customer Delay Cost (S/day/item) **Model 2A**	10	-		-
Customer Delay Cost (S/day/item) **Model 2B**	1	-		-
Inventory Holding Cost at Store (S/day/item)	4	2	1	-
Inventory Holding Cost at Cell (S/day/item)	-	4	2	1

As n=3, this model has 12 PAC parameters for optimization. The input data, given in Table 7.3, were chosen somewhat arbitrarily and form a specific instance for the model 2.

Model 2 is characterized by: m = 3, f = 1, fa = 0, n = 3, r = 1; and requires 36 lists and 9 SAMPST variables (refer to 5.4).

The mean customer demand arrival rate is λ = 1 customer/hour. We consider the cases for mean processing times equal to 6 or 42 minutes per product j. This gives utilization factors ρ_j equal to 0.1 and 0.7 respectively.

The characteristic of the arrival process is identical to that described in the case of model 1 (refer to 7.3.1).

7.4.2 Evaluation of Results for Model 2

Appendix C2 contains 16 tables with optimization results for different scenarios for model 2. The first 8 tables are for variant A of model 2 (model 2A), and the remaining 8 are for variant B (model 2B). The difference between these two main variants are in the values for the customer delay cost (refer to Table 7.3). For each model variant, we considered 8 combinations of mean processing times (in min.) for products [product 1, product 2, product 3]: [6,6,6], [42,6,6], [6,6,42], [42,6,42], [6,42,6], [42,42,6], [6,42,42] and [42,42,42].

For variant A, the longest optimization (PAC, 668 iterations, Table c2-4) took 670.1 CPU seconds and various PTO simulations required between 0.9 and 1.6 CPU seconds.

For variant B, the longest optimization (MRP, 365 iterations, Table c2-14) took 389.4 CPU seconds and various PTO simulations also required between 0.9 and 1.6 CPU seconds.

Again, as must be the case, the general PAC runs give the best results. The preference within traditional policies is different for almost each situation. Again it is very interesting to compare how the various types of control schemes vary in their performance for the different scenarios.

We summarized results for low and high system utilization, and for exponential processing times in a combination with standard or bulk demand on charts in Figures 7.8 and 7.9 for model 2A and 2B, respectively. As in the case of model 1, all charts contain the results for the DCII cost approach, and they present a sequence of policies related with their costs to the first and the best PAC scheme.

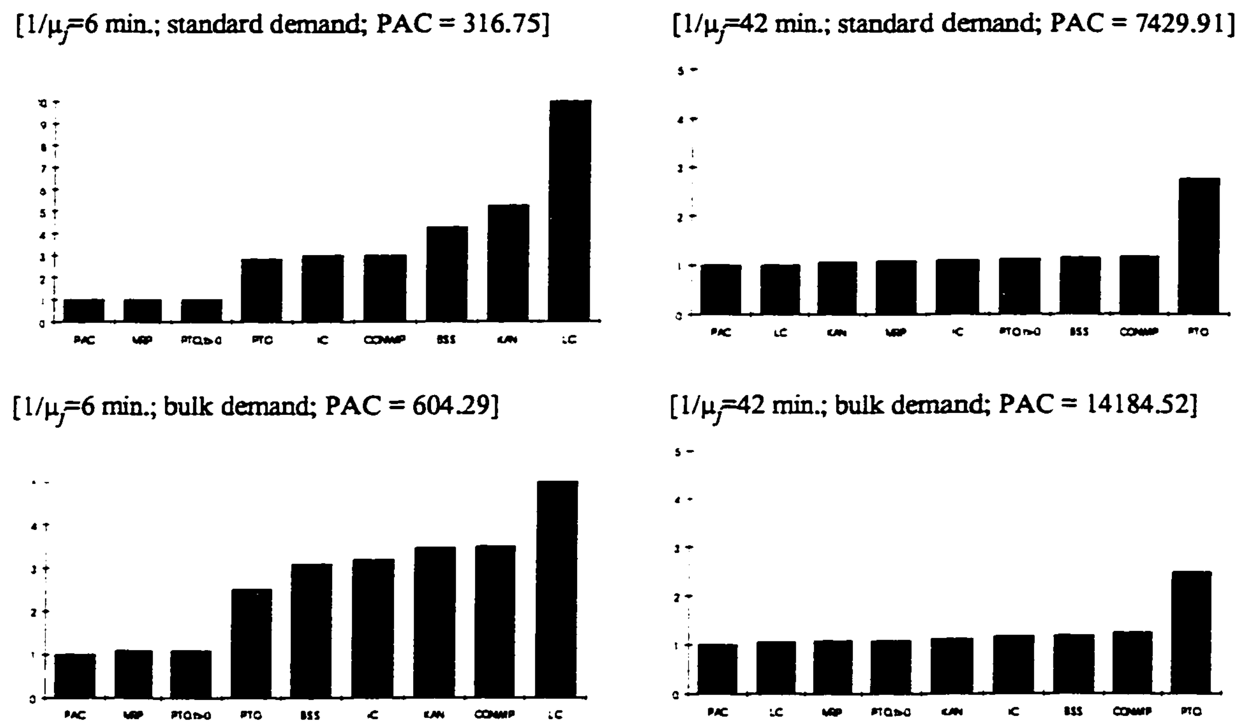
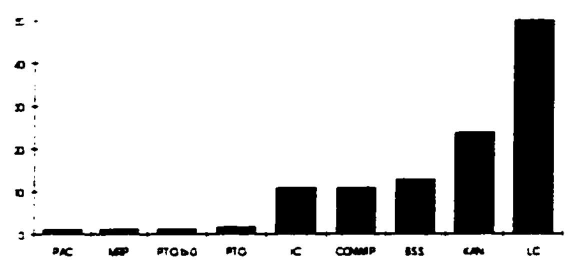


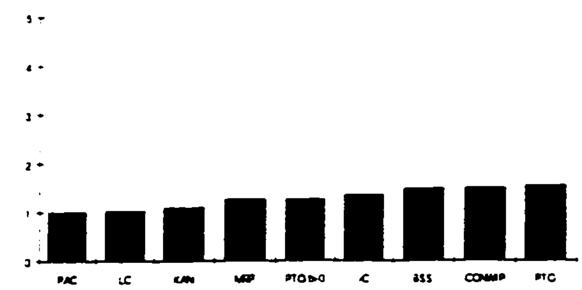
Figure 7.8 Model 2A: DCII, exponential processing times.

Decreasing the delay cost factor from 10 to 1 ($/day/item) generally results in a reduction of costs by more than 4 times in the case of a low system utilization, and by almost 3 times in the case of a high system utilization. The sequence of policies is similar for the presented cases of variant A and B; however the differences between the "best" policy and the "worst" one are much larger in the case of 2B in comparison with 2A and for the case of short processing times. This can be explained by relatively higher inventory holding costs in stores for IC, CONWIP, BSS, Kanban and LC (values of parameters z are required to be more than 0 and for LC more than 1).

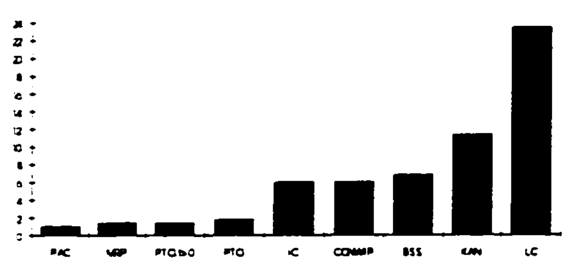
[$1/\mu_j$=6 min.; standard demand; PAC = 68.65]



[$1/\mu_j$=42 min.; standard demand; PAC = 2983.08]



[$1/\mu_j$=6 min.; bulk demand; PAC = 143.95]



[$1/\mu_j$=42 min.; bulk demand; PAC = 4860.35]

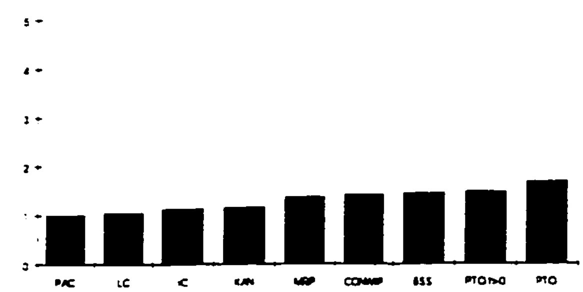


Figure 7.9 Model 2B: DCII, exponential processing times.

For the high utilization systems, the costs of different policies are close to each other. Only the PTO policy differs significantly in the case of model 2A, as the larger value of a customer delay cost factor seriously increases the total cost.

We compare the results for model 1 and model 2A for the exponential processing times, refer to Figure 7.3 and 7.8 respectively. In the case of model 2A, the costs of all policies increase by factor 1.2; however the policy sequencing and relations to the best PAC scheme are practically identical. In the case of model 2A, BSS performs slightly better than IC for low utilization and bulk demand This was just the opposite in the case of model 1. Table 7.4 shows the results of the comparison for IC and BSS for both models. BSS in model 2A benefits more from its safety stock in every cell (z_j=1). This results in increasing the average delay time (when adding a third cell to the flow line) by 1.4 min. instead of 2 min. as for IC.

Table 7.4 Model 1 and 2A: Comparison of IC and BSS results for low system utilization and bulk demand

Legend for Additional data (refer to Table 7.2)	Model 1		Model 2A	
	Additional data	Optimal parameter values	Additional data	Optimal parameter values
Policy Cost a1 - a2 - a3 b1 - b2 / c - d	IC 1617.75 51 - 0 - 49 0.8 - 0 / 0.4 - 18	z k r τ 1 1 1 0 0 1 1 0	IC 1920.24 55 - 0.2 - 45 1.6 - 0.02 / 0.4 - 20	z k r τ 1 1 1 0 0 1 1 0 1 2 1 0
Policy Cost a1 - a2 - a3 b1 - b2 / c - d	BSS 1620.75 55 - 8 - 37 1.3 - 0.2 / 0.2 - 13.5	z k r τ 1 M 2 0 1 M 1 0	BSS 1865.22 59 - 7 - 34 2.2 - 0.3 / 0.2 - 14.9	z k r τ 1 M 2 0 1 M 1 0 1 M 1 0

As in the case of model 1, it is also interesting to observe the results for unbalanced scenarios. In our study, the unbalanced cases are with mean processing times (in min.) for product 1, 2 and 3: [42,6,6], [6,6,42], [42,6,42], [6,42,6], [42,42,6] and [6,42,42]. Figures 7.10, 7.11 and 7.12 present bar charts for model 2A and for cases [42,6,6] and [6,6,42] (I), [42,42,6] and [6,42,42] (II), [42,6,42] and [6,42,6] (III) respectively, for exponential processing times, DCII cost approach, and standard and bulk demand. When comparing the results of unbalanced cases with the results for balanced scenarios from Figure 7.8, we can observe that all "unbalanced" charts are similar to "balanced" ones for high utilization systems. This means that all policies perform equally well, except PTO which has the worst performance. Except scenario [42,6,42], where policies are ranked almost identical, as in the case of the high utilization system, the sequencing of the policies is different almost for each "unbalanced" case.

[$1/\mu_j$= 42, 6, 6 min.; stand. demand; PAC =4077.81] [$1/\mu_j$= 6, 6, 42 min.; stand. demand; PAC = 2509.58]

[$1/\mu_j$=42, 6, 6 min.; bulk demand; PAC = 7802.97] [$1/\mu_j$= 6, 6, 42 min.; bulk demand; PAC = 3710.00]

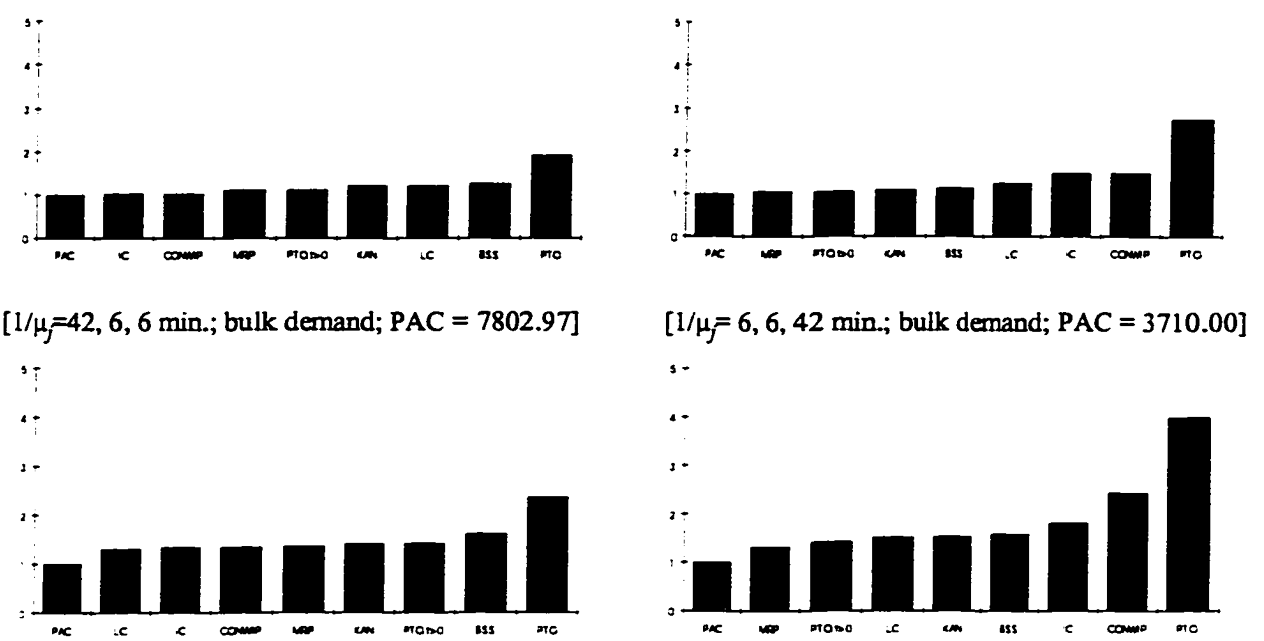


Figure 7.10 Model 2A: DCII, exponential processing times, unbalanced (I).

With regard to the values obtained for the τ parameters for the MRP and PTO-τ≥0 policies, the general comments are similar to those made in the case of model 1. In the case of the DCI approach, the requisition delays are not related to the lead times, but are applied to reduce the WIP holding costs. This implies that for a serial system, the monotonicity relationship should be included in the description of the required parameter settings for MRP. In the case of DCII, the values of τ parameters seem to confirm the monotonicity relationship. However, they are often much higher than the expected lead times. Again, we could explain it by obtaining results, which are not yet the global optimum.

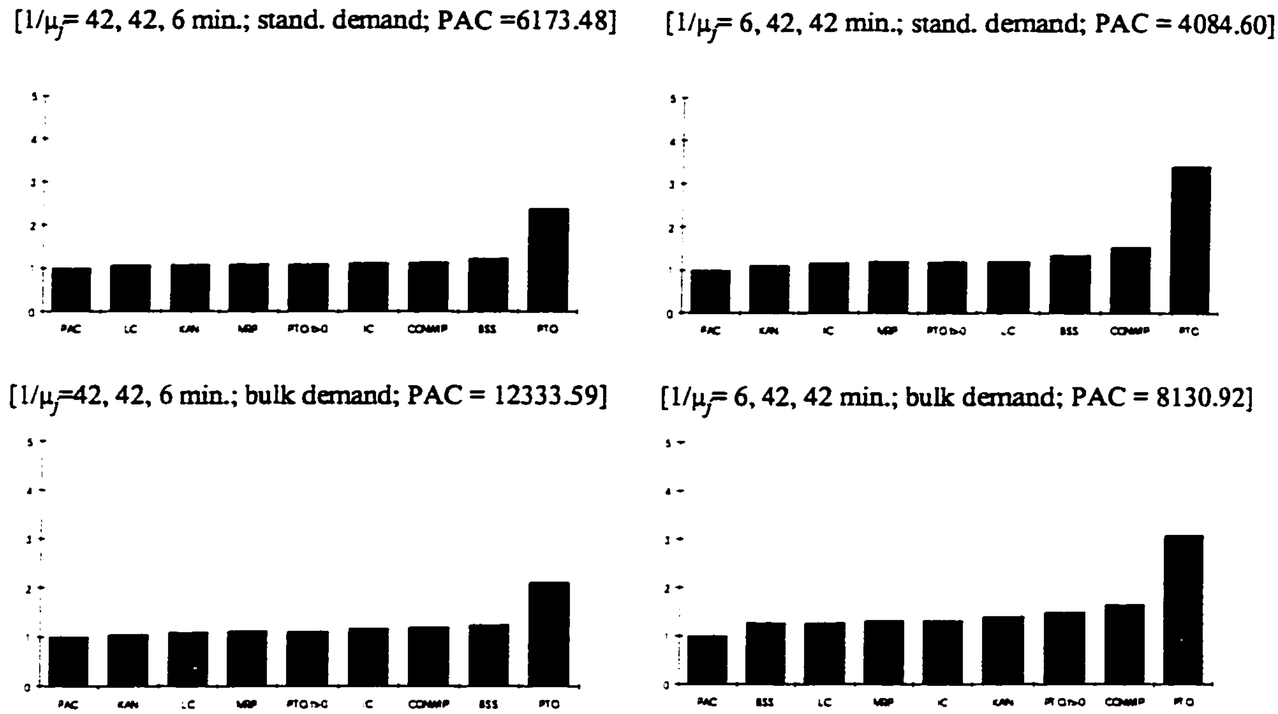


Figure 7.11 Model 2A: DCII, exponential processing times, unbalanced (II).

Recapitulating our findings about model 2, we observe that the results are similar to those obtained from model 1. These two models operate on similar principles and with similar input data, and, as was expected, they generate similar results. The optimization of a maximum of 12 parameters seems still quite reliable, as the number of parameters to be optimized is relatively small, and the parameter values are often close to their minimum values, which initiated the optimization process.

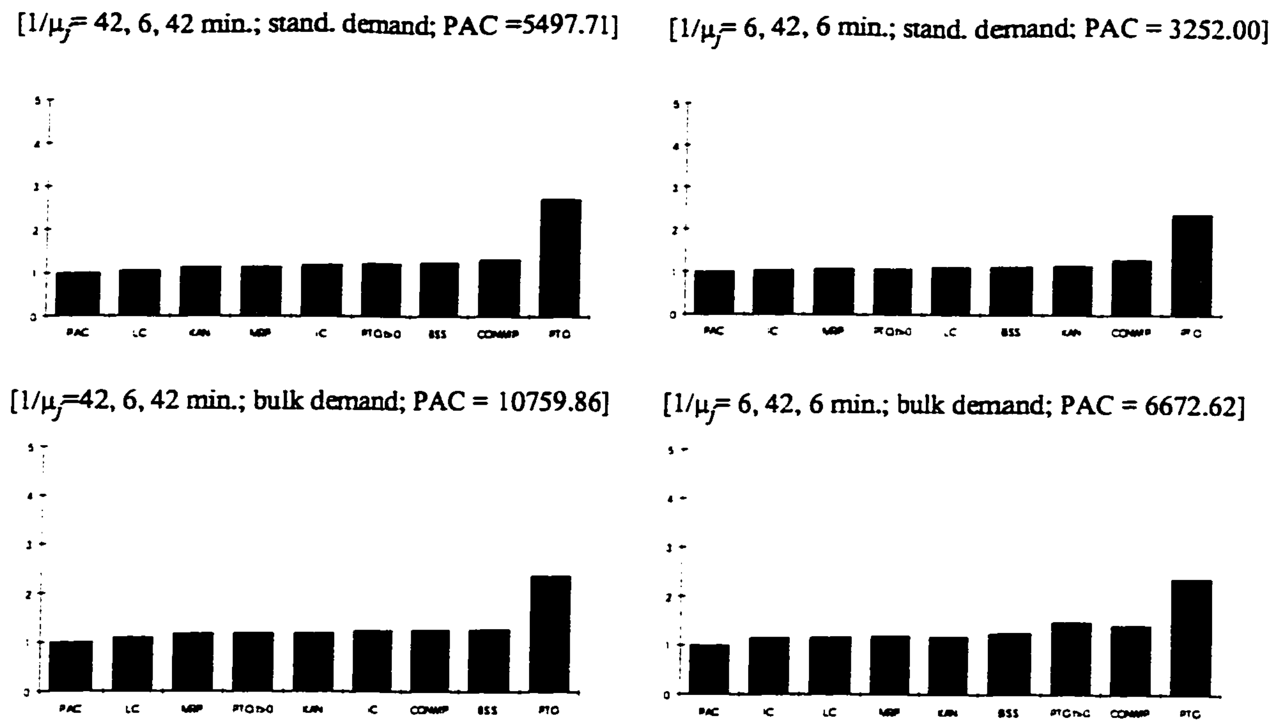


Figure 7.12 Model 2A: DCII, exponential processing times, unbalanced (III).

7.5 Model 3

7.5.1 Description of Model 3

Figure 7.13 presents the layout of the third manufacturing configuration considered in our study. This model is the simplest possible configuration, where both process and assembly activities can be included, and the impact of setup times can be investigated.

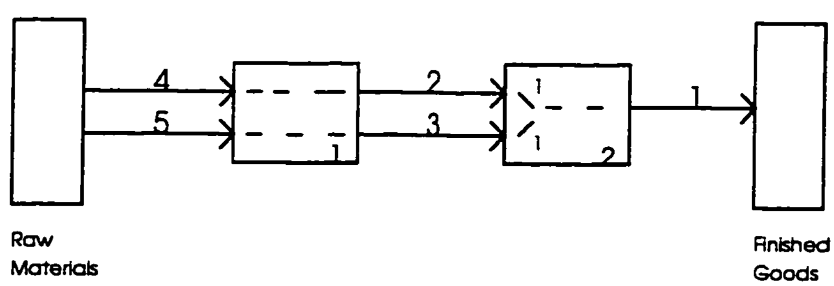


Figure 7.13 Layout of model 3.

Translation of this production layout into a PAC model results in Table 7.5. As n=3, this model has also 12 PAC parameters for optimization. The input data, given in Table 7.5, were chosen somewhat arbitrarily and form a specific instance for the model 3.

Table 7.5 Input data for model 3

Product Number	1	2	3	4	5
Product Kind	*f*	*a*	*a*	*r*	*r*
Production Cell	2	1	1	-	-
Mean Processing time (min.)	$1/\mu_1$	$1/\mu_2$	$1/\mu_3$	-	-
Setup Time (min.) **Model 3A**	0	0	0	-	-
Setup Time (min.) **Model 3B**	0	2	5	-	-
Number of Components	2	1	1	-	-
Names of Components	2,3	4	5	-	-
Probability of an arriving demand being for a product	1	-	-	-	-
Transportation Time from Store to Cell (min.)	-	0.01	0.01	0.01	0.01
Customer Delay Cost (S/day/item)	10	-		-	-
Inventory Holding Cost at Store (S/day/item)	4	2	2	-	-
Inventory Holding Cost at Cell (S/day/item)	-	4	4	2	2

Model 3 is characterized by: m = 2, f = 1, fa = 0, n = 3, r = 2; and requires 28 lists and 9 SAMPST variables (refer to 5.4).

The mean customer demand arrival rate is λ = 1 customer/hour. We consider the cases for mean processing times equal to 6 or 42 minutes per product j. This gives utilization factors ρ_j equal to 0.1 and 0.7 respectively.

The characteristic of the arrival process is identical to that described in the case of model 1 (refer to 7.3.1).

7.5.2 Evaluation of Results for Model 3

Appendix C3 contains 12 tables with optimization results for different scenarios for model 3. The first 6 tables are for variant A of model 3 (model 3A), and the remaining 6 are for variant B (model 3B). The difference between these two main variants are in the values for the setup times, that is, variant A does not apply setup times, when in variant B setup times are added to every processing in cell 1, when a new production batch is started. For each variant of the model, we consider the following 6 combinations of mean processing times for products [product 1, product 2, product 3]: [6,6,6], [42,6,6], [6,6,42], [42,6,42], [6,42,6] and [42,42,6].

For variant A, the longest optimization (PAC, 710 iterations, Table c3-4) took 887.5 CPU seconds and various PTO simulations required between 1.0 and 3.0 CPU seconds.

For variant B, the longest optimization (PAC, 1184 iterations, Table c3-10) took 2125.1 CPU seconds and various PTO simulations required between 1.1 and 6.0 CPU seconds.

As must be the case, the general PAC runs give the best results. We give an overview of some selected results in charts in Figures 7.14 and 7.15 for variant 3A and 3B respectively. Each figure shows the cost relation between policies (as in case of previous two models) for two main scenarios of utilization of the system: low, when all mean processing times are equal to 6 min., and relatively high, when product 1 and 2 require on average 42 min. and product 3 is processed on average within 6 min. Both processing scenarios are given for the case of exponential processing times and DCII cost calculation approach, and with a combination of standard or bulk arrival of customer demand.

Comparing the results between the two variants of model 3, without setup and with setup times, shows general increase of costs for each policy. The sequence of policies is identical for both variants and for the case of short processing times. Including the setup times in the case of a high utilization system does not change the policy sequence for bulk demand, but changes the order of policies in the case of standard demand.

It is interesting to compare IC and BSS. We present these policies with the additional data in Table 7.6. To reduce influences of setup costs in model 3B, BSS sets batch for PA cards (parameter r_2) to 2, for product 2 (with larger setup cost and longer processing time than product 3). This results in longer average delay time, but reduces WIP inventories.

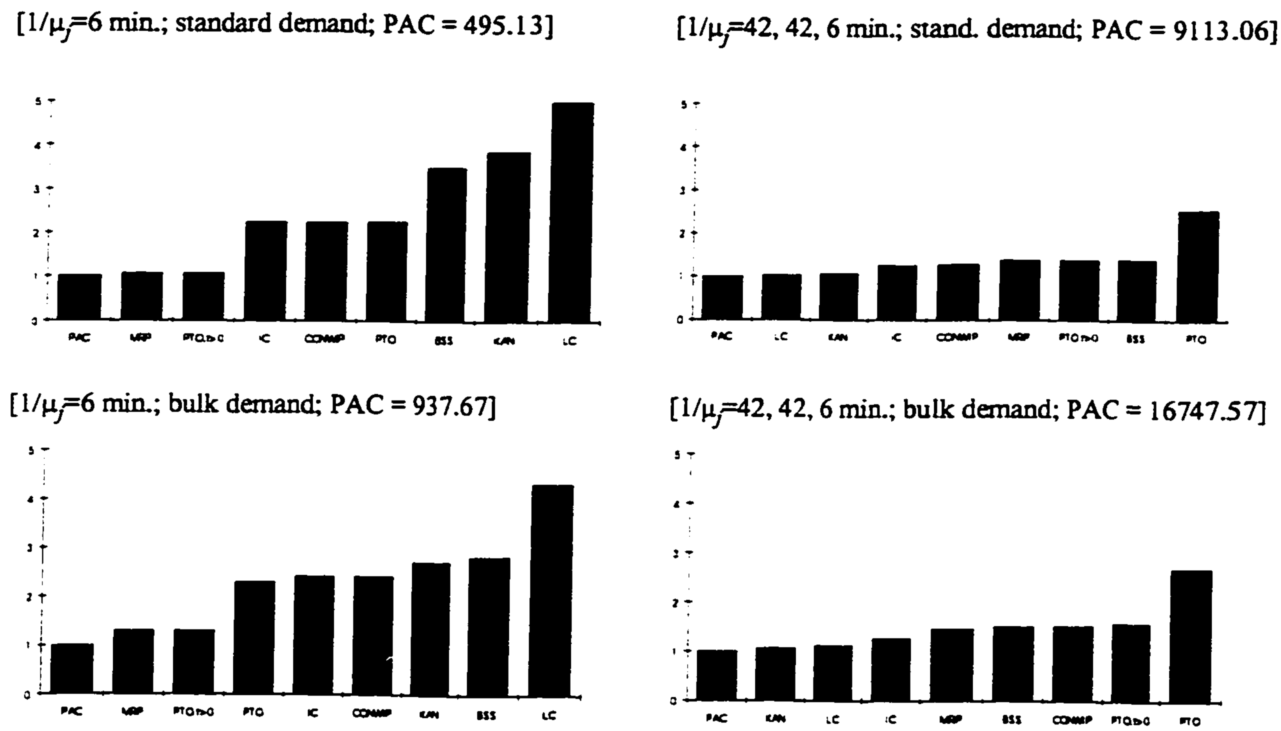
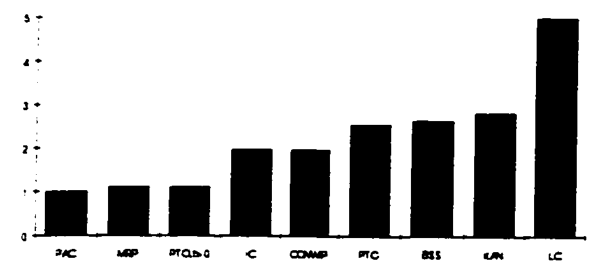


Figure 7.14 Model 3A: DCII, exponential processing times.

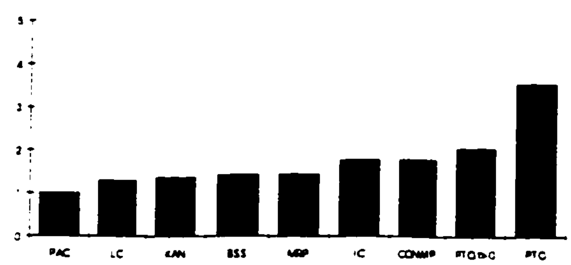
Table 7.6 Model 3: Comparison of IC and BSS results in the case of high system utilization and standard demand

Legend for Additional data (refer to Table 7.2)	Model 3A		Model 3B	
	Additional data	Optimal parameter values	Additional data	Optimal parameter values
Policy Cost a1 - a2 - a3 b1 - b2 / c - d	IC 11542.54 25 - 46 - 29 3.6 - 7.3 / 0.7 - 77.4	z k r τ 6 6 1 0 1 7 1 0 2 9 1 0	IC 18337.93 35 - 50 - 15 7.4 - 14.1 / 0.8 - 62.8	z k r τ 10 10 1 0 4 14 1 0 5 15 1 0
Policy Cost a1 - a2 - a3 b1 - b2 / c - d	BSS 12777.45 23 - 53 -24 3 - 9.6 / 0.7 - 71.7	z k r τ 7 M 2 0 1 M 1 0 1 M 1 0	BSS 14670.64 23 - 53 - 24 4 - 12 / 0.7 - 81.3	z k r τ 6 M 1 0 3 M 2 0 3 M 1 0

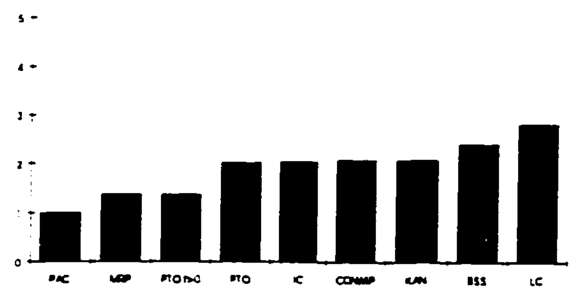
[$1/\mu_j$=6 min.; standard demand; PAC = 668.38]



[$1/\mu_j$=42, 42, 6 min.; stand. demand; PAC = 10362.86]



[$1/\mu_j$=6 min.; bulk demand; PAC = 1448.28]



[$1/\mu_j$=42, 42, 6 min.; bulk demand; PAC = 20419.95]

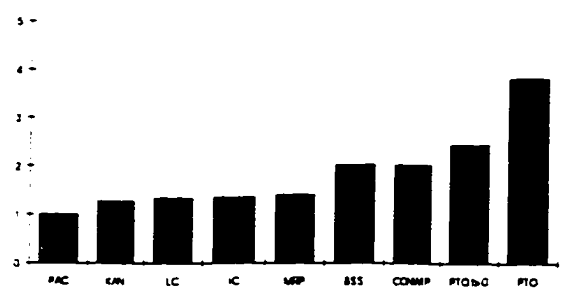


Figure 7.15 Model 3B: DCII, exponential processing times.

Variant B includes setup times at machines. Among the results obtained for different scenarios of the PAC policy, we can see many cases, where PA cards batch size *r* is more than 1, especially for situations of higher utilization of the system. In variant A, for all optimal solutions generated for the PAC policy, the batch sizes were equal to 1.

We also examine results for two other processing time scenarios of the system where the mean processing times for product 1 (final product) are always equal to 6 min. In the first case the mean processing times of product 2 and 3 are 6 and 42 min., respectively ([6,6,42]). In the second case the mean processing times of these assembly products are interchanged and are 42 and 6 min., respectively ([6,42,6]). Figure 7.16 presents the bar charts for these two cases. All policies perform equally well, except the PTO which has the worst performance. LC, Kanban and IC do relatively better than the other policies, while BSS and PTO-$\tau \geq 0$ generally have the worst ranking.

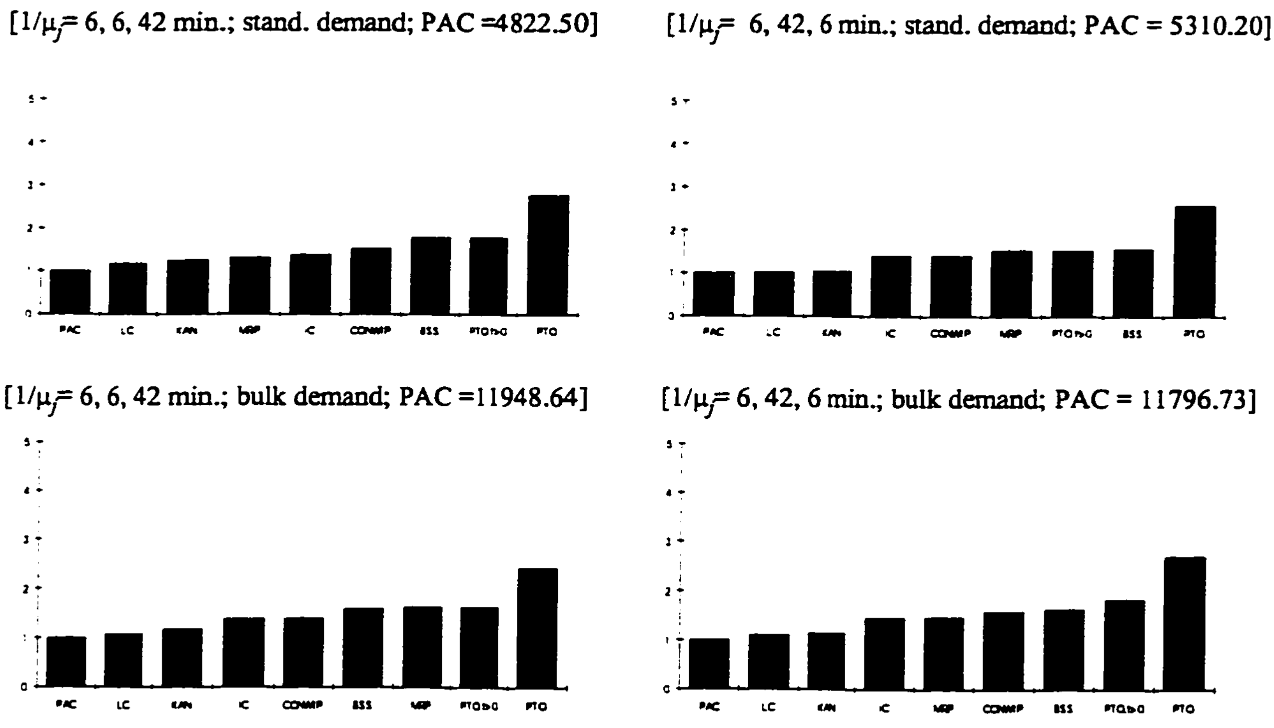


Figure 7.16 Model 3A: DCII, exponential processing times, other cases.

It is worth noting that these results also give some insight into the impact of priority in sending order tags for assembly products with different processing times on the choice of the parameters for each policy and the total cost. At first, both cases seem identical and suggest generating similar costs for each policy. However, the results show different outcomes. Both assembly products (products 2 and 3) are always required by the cell 2 at the same time. As well, both raw materials (products 4 and 5) are supplied to the WIP of cell 1 at the same time. If the first PA card in the PAC 1 queue is for product j, this product will be processed as the first at cell 1. The other raw material will have to wait in the WIP 1 queue. IC, CONWIP and PTO have always lower costs when product 2 has shorter processing time than product 3, while Kanban operates on lower costs when the processing times of product 2 are longer than those of product 3. In the case of the remaining policies, the performance of the policy is not consistent in the case of standard demand and bulk demand. For example, when product 2 has shorter processing time than product 3, the PAC policy results in lower costs in the case of standard demand, and slightly increases costs in the case of bulk demand. Future research might involve more detailed study of this phenomenon.

The best way to evaluate the results for model 3 is to compare the results of variant 3A from Figure 7.14 to the "DCII and exponential processing times" results of model 1 (refer to Figure 7.3) and model 2 (refer to Figure 7.8). All three models produce similar results, as all three models have much in common. Model 3 is similar to model 2, in the sense that it processes product 2 and 3 in common cell and not on two separate cells. Model 3 can be seen as aggregated model 1, where production of two separate products 2 and 3, can be seen as production of one product, which requires longer processing time.

7.6 Model 4

7.6.1 Description of Model 4

Figure 7.17 presents the layout of the fourth manufacturing configuration considered. This model is the example configuration used to help explain the main features of the PAC simulation program (refer to 5.3).

Translation of this production layout into a PAC model results in Table 7.7. Here the number of products, n, equals 7, which results in 28 PAC parameters for optimization. The input data, given in Table 7.7, were chosen somewhat arbitrarily and form a specific instance for the model 4.

Model 4 is characterized by: $m = 4, f = 3, fa = 1, n = 7, r = 4$; and requires 84 lists and 23 SAMPST variables (refer to 5.4).

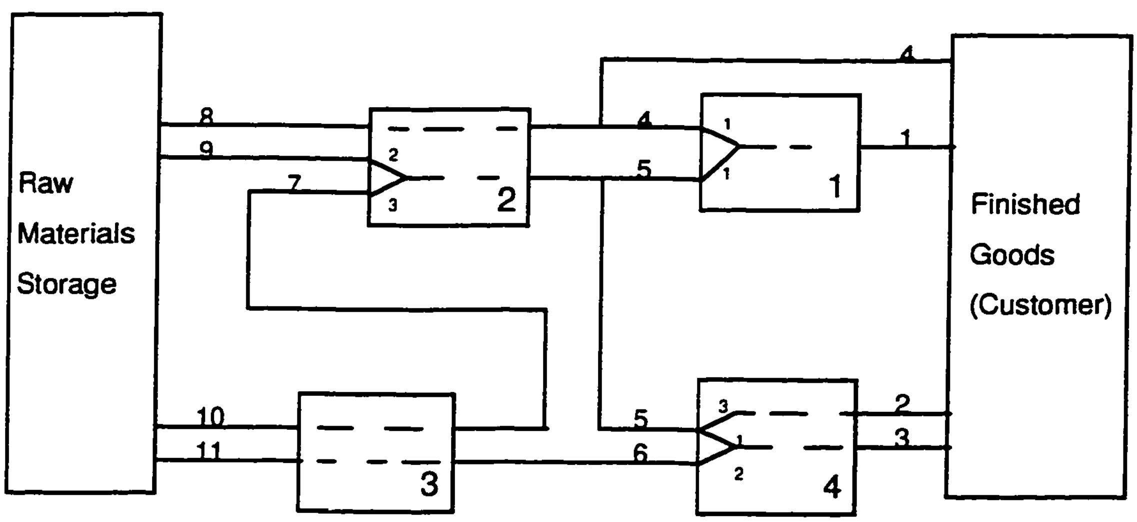


Figure 7.17 Layout of model 4.

Table 7.7 Input data for model 4

Product Number	1	2	3	4	5	6	7	8	9	10	11
Product Kind	*f*	*f*	*f*	*fa*	*a*	*a*	*a*	*r*	*r*	*r*	*r*
Production Cell	1	4	4	2	2	3	3	-	-	-	-
Mean Processing time (min.) **Model 4A**	42	18	36	18	24	18	30	-	-	-	-
Mean Processing time (min.) **Model 4B**	63	27	54	27	36	27	45	-	-	-	-
Setup Time (min.)	0	7	1	1	9	1	8	-	-	-	-
Number of Components	2	1	2	1	2	1	1	-	-	-	-
Names of Components	4,5	5	5,6	8	7,9	11	10	-	-	-	-
No. of Items per Each Component	1,1	3	1,2	1	3,2	1	1	-	-	-	-
Probability of an arriving demand being for a product *j*	0.5	0.1	0.3	0.1	-	-	-	-	-	-	-
From Store To Cell: Transp. Time (min.):	-	-	-	1 12	1,4 12,18	4 6	2 18	2 24	2 24	3 18	3 18
Customer Delay Cost ($/day/item)	10	8	6	5	-	-	-	-	-	-	-
Inventory Holding Cost at Store ($/day/item)	16	40	20	3	12	3	3	-	-	-	-
Inventory Holding Cost at Cell ($/day/item)	-	-	-	3	12	3	3	2	1	2	2

Note: Mean processing times of model 4B are increased by 50%.

The mean customer demand arrival rate for all 4 final products is λ = 1 customer/3 hours. The expected number of arrivals of customer demand for all products is 2080 (=6240/3), since the evaluation time per each simulation run is 6240 hours (260 days * 24 hours/day). The expected number of arrivals of customer demand for products 1, 2, 3 and 4 are 1040, 208, 624 and 208 respectively, based on the defined probabilities (refer to Table 7.7). We use a single arrival stream (stream number 1) for all simulation runs, which results in a generation of approximately 2080 arrivals. In the case of bulk arrival of demand, the stream number for a geometric distribution was set to 46.

7.6.2 Evaluation of Results for Model 4

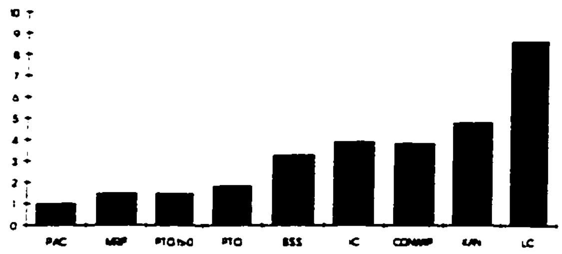
Appendix C4 contains 4 tables with optimization results for different scenarios for model 4. The first 2 tables are for model 4A, the remaining 2 are for model 4B. Model 4B is a variant of model 4A where mean processing times are increased by 50%. The results are for situations without and with setup times for each case of processing time considered, exponential and uniform, for standard and bulk demand, and for DCI and DCII customer cost calculation approach.

For variant A, the longest optimization (MRP, 1228 iterations, Table c4-2) took 8841.6 CPU seconds and various PTO simulations required between 1.8 and 7.8 CPU seconds.

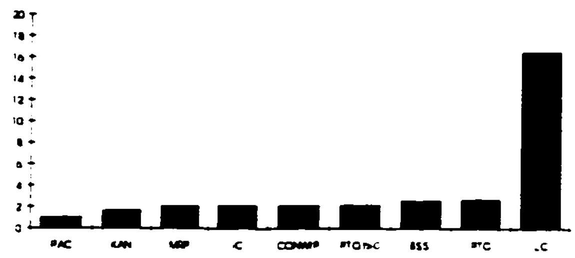
For variant B, the longest optimization (MRP 3090 iterations, Table c4-4) took 28412.5 CPU seconds and various PTO simulations required between 4.3 and 9.9 CPU seconds.

Model 4 is much more complex than previously described models. To facilitate the evaluation of the results, we present bar charts for DCII and the two main scenarios of processing times, exponential and uniform (refer to Figure 7.18 and 7.19). After careful examination of the results, we decided to consider variant A without setup times, as a low system utilization situation, and variant B with setup times, as a relatively high utilization system. There were no differences in sequencing or relations of the policies to the best PAC scheme, when comparing the case "without setup" and "with setup" within the same variant of the model. Table 7.8 contains some additional data for all policies from Figure 7.18, which help evaluate the obtained sequencing of the policies.

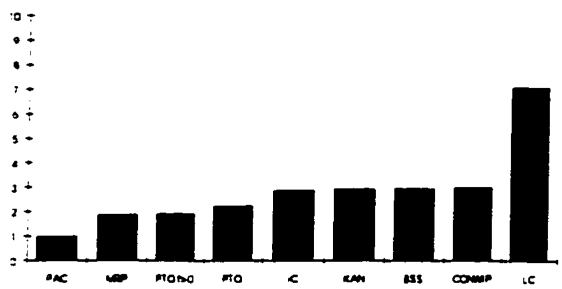
[standard demand; 4A no setup; PAC = 5012.81]



[standard demand; 4B with setup; PAC = 16582.80]



[bulk demand; 4A no setup; PAC = 8364.27]



[bulk demand; 4B with setup; PAC = 26055.62]

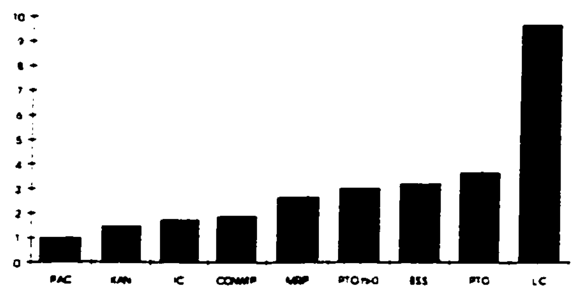
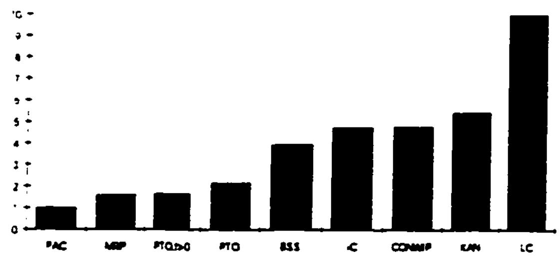


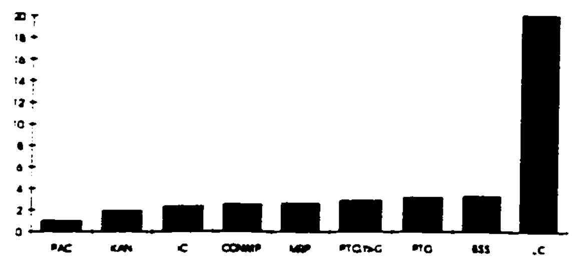
Figure 7.18 Model 4: DCII, exponential processing times.

All charts, for each case and for both processing time scenarios, are similar. The total costs for the case of uniform processing times are reduced by approximately 20% compared to exponential processing times. MRP performs well for systems with a low system utilization. It appears to be due to the implementation of forecasting, thus reducing the customer delay costs, and also due to relatively short processing times, which result in modest WIP holding costs (refer to column I and II in Table 7.8). Longer processing times significantly increase the WIP holding costs for MRP in the case of high system utilization (refer to column III and IV in Table 7.8). Kanban does much better in cases of high system utilization, as it better controls the WIP inventory. The additional output data of the best PAC parameter settings for each policy and for each specific system scenario help in understanding how the different policies adjust to the system condition and why one policy can do better than other.

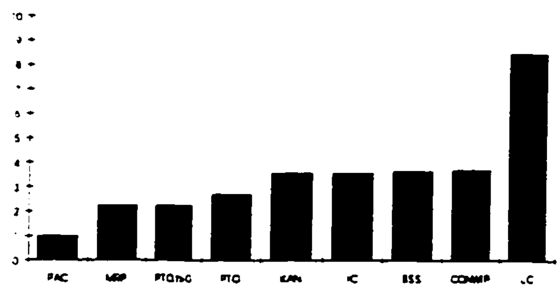
[standard demand; 4A no setup; PAC = 4003.59]



[standard demand; 4B with setup; PAC = 12718.88]



[bulk demand; 4A no setup; PAC = 6627.09]



[bulk demand; 4B with setup; PAC = 24456.81]

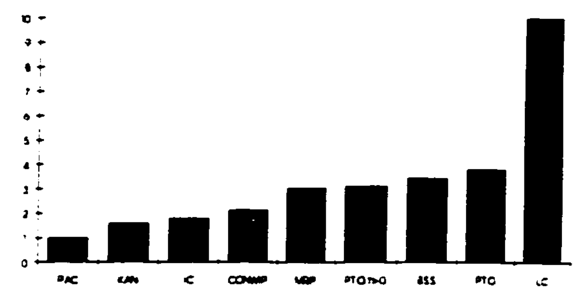


Figure 7.19 Model 4: DCII, uniform processing times.

In Table 7.8, we also indicate the optimal parameters for PAC and the second best policy in order to observe the differences in optimal parameter values. In the case of low system utilization, the second best is MRP. The delay costs for PAC and MRP are almost identical. The inventory costs are much lower for PAC due to the limiting of process tags. In the case of high system utilization, the second best is Kanban. The delay costs for PAC and Kanban are almost identical, but the holding inventory costs are much higher for Kanban. This form of Kanban demands to put initially a certain amount of products in inventory, while PAC does not imply any relations on the number of process tags and the inventory limits.

Table 7.8 Model 4: additional data for results presented in Figure 7.18 (DCII, exponential processing times)

Best Solution	Legend	low utilization: variant 4A, no setup		high utilization: variant 4B with setup	
		standard demand (I)	bulk demand (II)	standard demand (III)	bulk demand (IV)
1	Policy Cost a1 - a2 - a3 c - d	PAC 5012.81 14 - 56 - 30 (=28+2) 0.3 - 127 z k r s 0 3 1 184 0 11 1 408 0 3 1 207 0 3 1 20 0 3 1 20 0 1 1 0 0 3 1 0	PAC 8364.27 23 - 30 - 47 (=46+1) 0.3 - 346 z k r s 0 10 1 134 0 19 1 344 0 1 1 156 0 1 1 0 0 1 1 21 0 15 1 0 4 3 1 0	PAC 16582.80 31 - 35 - 33 (=32+1) 0.3 - 465 z k r s 0 15 1 193 0 1 1 53 0 5 1 188 0 1 1 0 0 3 1 0 0 11 1 0 0 3 1 0	PAC 26055.62 12 - 21 - 67 (=66+1) 0.3 - 1485 z k r s 0 3 1 649 0 11 1 1176 0 3 1 313 0 1 1 0 0 1 1 0 0 [illegible] 1 0 8 3 1 1
2	Policy Cost a1 - a2 - a3 c - d	MRP 7360.96 20 - 59 - 21 (=20+1) 0.3 - 130 z k r s 0 M 1 181 0 M 1 377 0 M 1 236 0 M 1 0 0 M 1 0 0 M 1 141 0 M 1 0	MRP 15661.73 6 - 69 - 25 (=24+1) 0.3 - 342 z k r s 0 M 1 232 0 M 1 409 0 M 1 311 0 M 1 1 0 M 1 12 0 M 1 122 1 M 1 52	Kanban 27495.04 56 - 24 - 20 0.6 - 440 z k r s 3 3 1 0 1 1 1 0 1 1 1 0 1 1 1 0 4 4 1 0 1 1 1 0 3 3 1 0	Kanban 37986.14 33 - 20 - 47 0.3 - 1530 z k r s 3 3 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 2 2 1 0 9 9 2 0
3	Policy Cost a1 - a2 - a3 c - d	PTO, τ≥0 7360.96 20 - 59 - 21 (=20+1) 0.3 - 130	PTO, τ≥0 16111.03 4 - 72 - 24 (=23+1) 0.3 - 347	MRP 33583.64 8 - 71 - 21 (=20+1) 0.3 - 635	IC 44744.64 25 - 28 - 47 0.3 - 1966
4	Policy Cost a1 - a2 - a3 c - d	PTO 9162.75 0 - 60 - 40 0 - 317	PTO 18627.24 0 - 61 - 39 0 - 627	IC 34897.77 36 - 41 - 23 0.5 - 689	CONWIP 448930.98 27 - 29 - 44 0.3 - 2039
5	Policy Cost a1 - a2 - a3 c - d	BSS 16297.58 48 - 40 - 13 0.3 - 174	IC 24151.10 64 - 15 - 21 0.3 - 422	CONWIP 34897.77 36 - 41 - 23 0.5 - 689	MRP 68562.38 10 - 65 - 25 (=24+1) 0.3 - 1520
6	Policy Cost a1 - a2 - a3 c - d	IC 19367.25 68 - 19 - 13 0.5 - 194	Kanban 24423.73 72 - 9 - 19 0.3 - 373	PTO, τ≥0 35368.88 4 - 72 - 24 (=23+1) 0.3 - 745	PTO, τ≥0 78259.98 4 - 69 - 27 (=26+1) 0.3 - 1856
7	Policy Cost a1 - a2 - a3 c - d	CONWIP 19516.47 63 - 19 - 18 0.5 - 273	BSS 24656.81 28 - 46 - 26 0.1 - 570	BSS 42837.00 14 - 64 - 22 0.3 - 829	BSS 82984.86 14 - 60 - 26 0.2 - 1915
8	Policy Cost a1 - a2 - a3 c - d	Kanban 23907.48 66 - 19 - 15 0.7 - 299	CONWIP 24873.01 63 - 17 - 20 0.2 - 458	PTO 44496.42 0 - 66 - 34 0 - 1367	PTO 94874.47 0 - 62 - 38 0 - 3150
9	Policy Cost a1 - a2 - a3 c - d	LC 43310.17 75 - 6 - 19 0.7 - 587	LC 58988.26 53 - 4 - 43 0.4 - 1832	LC 273963.94 44 - 1 - 55 0.7 - 15637	LC 251825.12 40 - 1 - 59 0.5 - 14944

Legend

For each policy is given the total cost (in $) and a set of the following data:

a1 percentage of "inventory in stores" cost in total cost

a2 percentage of "inventory in cells" cost in total cost

a3 percentage of "customer delay" cost in total cost, where for PAC, MRP and PTO-τ≥0, this cost is a sum of "small" and "large" cost, which are given in brackets

c probability that an arriving demand from a customer for a final product is met immediately

d overall average final product delay to customer (in min.)

Observe that the results for model 4 with a relative low system utilization are similar to the "DCII and exponential processing times" results for the previous three models. However, the high system utilization scenario of model 4 shows more differences in the costs between the policies, with Kanban significantly better, than MRP or BSS. Still, the largest change is in the performance of the LC policy, which did so well in the previous three models. Now, it is clearly the worst scheme. This can be explained by a very high inventory level in stores, and 30 to 50 % of final products having a very large delay time to customer. LC has the lowest WIP holding costs among all policies, but it does not compensate the other costs. Clearly, for more complex manufacturing layouts the LC policy does not have enough flexibility to deal with many factors of the production environment. It can only change an initial inventory level that results in high inventory holding costs in stores for the whole system. Even PTO, with fixed values of parameters, does better than LC. PTO transfers information about a customer demand directly to the first cells which immediately start production. The WIP holding costs for PTO are high, but both inventory in stores and delay costs are much lower than in the case of LC (refer to column III and IV in Table 7.8).

7.7 Model 5

7.7.1 Description of Model 5

The last manufacturing configuration considered in our study is a model with a quite complex structure. This model has been based on the simulation case from Philipoom et al. (1987).

The model is a production of two end items, X and Y, with a product structure shown on Figure 7.20. Two units of component E and three units of component F are required to produce one unit of assembly A, which is subsequently combined with two units of B and two units of C to produce one unit of product X.

Item Y is manufactured in a similar manner. Figure 7.21 illustrates a multistage production operation encompassing six workcentres in which items X and Y and their component parts are manufactured. Cells 1, 2 and 3 represent assembly-type operations whereas cells 4, 5, and 6 do not; they perform only processing activities. Figure 7.22 translates the given production process into the production scheme, which can be used by the PAC simulation model. This results in a total of 11 product types (2 "final" and 9 "assemblies") and 4 "raw materials".

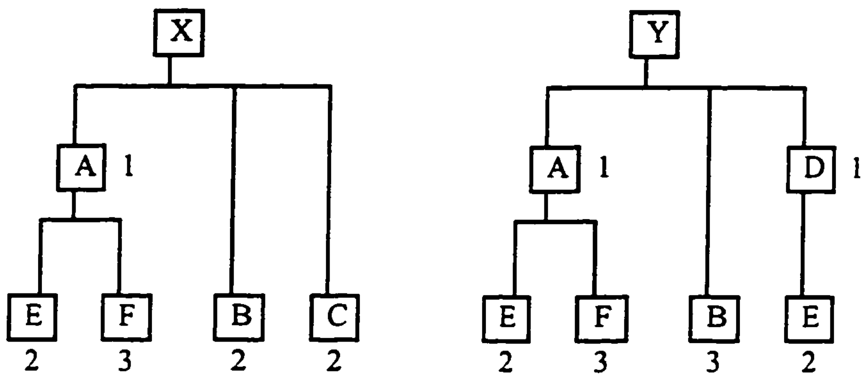


Figure 7.20 The product structure for X and Y.

Table 7.9 summarizes all input data for the model layout 5. The total number of 11 product types, as n=11, results in 44 PAC parameters for optimization. The input data, given in Table 7.9, were chosen somewhat arbitrarily and form a specific instance for the model 5.

Model 5 is characterized by: $m = 6, f = 2, fa = 0, n = 11, r = 4$; and requires 142 lists and 25 SAMPST variables (refer to 5.4).

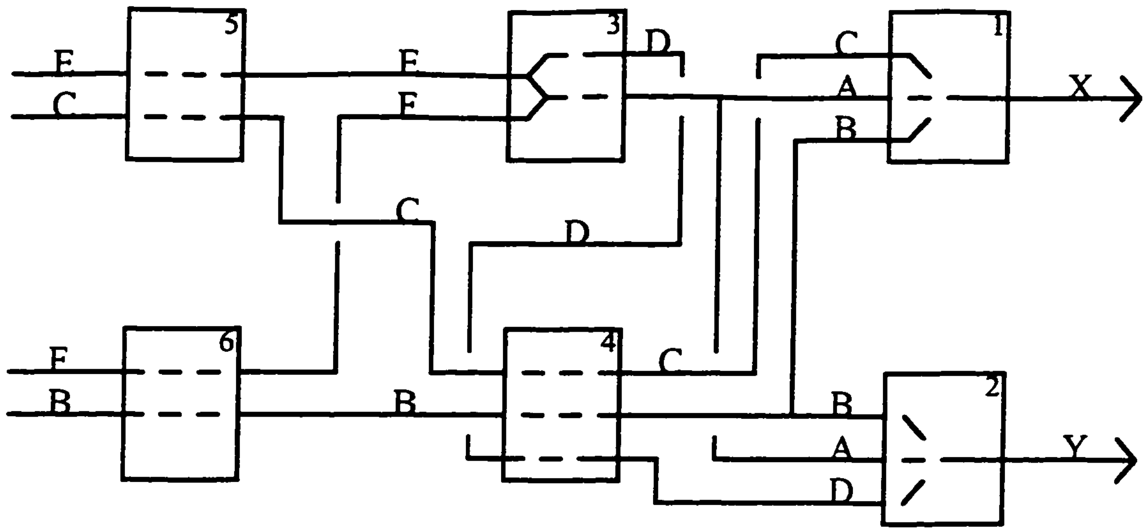


Figure 7.21 Layout and flows for production of X and Y.

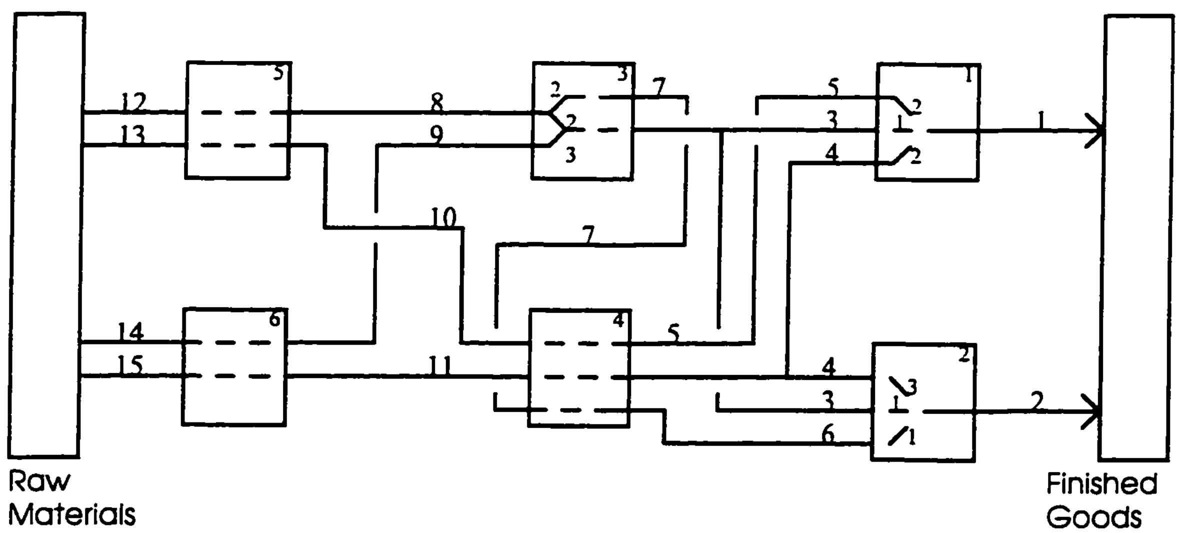


Figure 7.22 Layout of model 5.

Table 7.9 Input data for model 5

Product Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Product Kind	*f*	*f*	*a*	*a*	*a*	*a*	*a*	*a*	*a*	*a*	*a*	*r*	*r*	*r*	*r*
Production Cell	1	2	3	4	4	4	3	5	6	5	6	-	-	-	-
Mean Processing time (min.) **Model 5A**	18	42	36	18	24	24	18	12	18	18	12	-	-	-	-
Mean Processing time (min.) **Model 5B**	21.6	50.4	43.2	21.6	28.8	28.8	21.6	14.4	21.6	21.6	14.4	-	-	-	-
Setup Time (min.)	0	0	6	4	1	8	2	5	3	1	1	-	-	-	-
Number of Components	3	3	2	1	1	1	1	1	1	1	1	-	-	-	-
Names of Components	3,4, 5	3,4, 6	8,9	11	10	7	8	12	14	13	15	-	-	-	-
No. of Items per Each Component	1,2, 2	1,3, 1	2,3	1	1	1	1	1	1	1	1	-	-	-	-
Probability: an arriving demand is for a product *j*	0.67	0.33	-	-	-	-	-	-	-	-	-	-	-	-	-
From Store To Cell: Transp. Time (min.):	-	-	1,2 12,18	1,2 18,12	1 18	2 12	4 24	3 6	3 12	4 12	4 6	5 10	5 12	6 12	6 18
Customer Delay Cost ($/day/item)	10	8	-	-	-	-	-	-	-	-	-	-	-	-	-
Inventory Holding Cost at Store ($/day/item)	25	30	12	3	3	6	5	2	3	2	2	-	-	-	-
Inventory Holding Cost at Cell ($/day/item)	-	-	12	3	3	6	5	2	3	2	2	1	1	2	1

Note: Mean processing times of model 5B are increased by 20%

The mean customer demand arrival rate for both final products is $\lambda = 1$ customer/2 hours. The expected number of arrivals of customer demand for all products is 3120 (=6240/2), since the evaluation time per each simulation run is 6240 hours (260 days * 24 hours/day). The expected number of arrivals of customer demand for product 1 and 2 are 2090 and 1030 respectively, based on the defined probabilities (refer to Table 7.9). We use a single arrival stream (random number stream 1) for all simulation runs, which results in a generation of approximately 3130 arrivals. In the case of bulk arrival of demand, the stream number 18 was used for the geometric distribution.

7.7.2 Evaluation of Results for Model 5

Appendix C5 contains 4 tables with optimization results for different scenarios for model 5. The first 2 tables are for model 5A, the remaining 2 are for model 5B. Model 5B is a variant of model 5A where mean processing times are increased by 20%. The results are for situations without and with setup times for each case of processing time, customer arrival and cost scenario.

For variant A, the longest optimization (PAC, 2776 iterations, Table c5-2) took 11339.6 CPU seconds and various PTO simulations required between 9.5 and 17.2 CPU seconds.

For variant B, the longest optimization (MRP, 1730 iterations, Table c5-4) took 23298.2 CPU seconds and various PTO simulations required between 10.9 and 28.6 CPU seconds.

As much be the case, the general PAC runs give the best results. However it was almost impossible to obtain the best solution by initializing the parameters to their minimum values. The PAC parameters set to the minimum values resulted in an infeasible solution, so it was also not such a good idea to start optimization with an infeasible scenario. The optimization for the PAC policy was obtained by initializing the parameters with values obtained as best for MRP, Kanban or IC.

As in the case of model 4, influences of setups on the choice of parameters are difficult to explain. There is not a clear trend pointing to a relationship of setup times on the size of a batch of PA cards (parameter *r*). In the case of a flow line in model 3, high setup times clearly increased the batch sizes of PA cards.

As in the case of model 5, we decide to consider variant 5A without setup times, as a system with low utilization, and variant 5B with setup times, as a relatively high utilization system. To facilitate the evaluation of results for model 5, in Figure 7.23 we present bar charts for the exponential processing times scenario, DCII cost calculation approach, and for low and high system utilization in combination with standard and bulk demand. When comparing the results for model 5 from Figure 7.23 with the "DCII and exponential processing times" results for model 4 from Figure 7.18, we observe that the sequence of policies is similar for all 4 cases. However the differences of costs between policies are generally much smaller, except the case with high system utilization and bulk demand, where PAC and Kanban are significantly better than other policies.

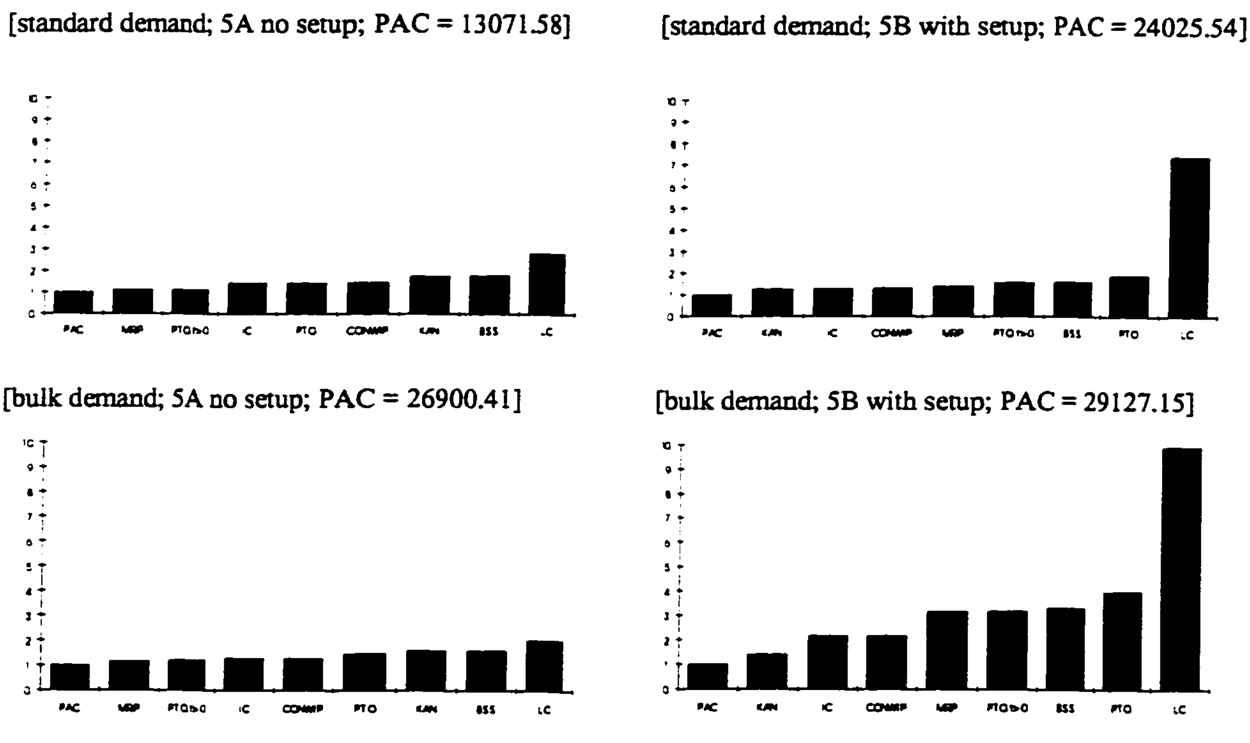


Figure 7.23 Model 5: DCII, exponential processing times.

Model 5 is much more complex, than model 4. However; as both models operate on similar principles and input data, the results for both layouts are very alike. It is interesting to note that the two top classical policies in industry, Kanban and MRP, end up head-to-head as second best for model 4 and 5 depending upon the system utilization scenario. Each is outclassing the other in its preferred context. They jointly beat all other schemes. This leaves the pair as best classical policies for more complex systems, which is also most common in practice. The optimization procedure of PAC parameters for model 4 and 5 also has more creditability, as the results are comparable with the first three relatively simple models.

7.8 Effectiveness of Random Optimization Algorithm

We tested the PACRAN optimization algorithm in the case of models 1 and 4. This time we did not perform many test runs, as it became quickly evident that the algorithm does not perform as fast as the PACOPT optimization algorithm. In the case of all 5 models tested and especially for simple models, the characteristic of the PAC parameters optimization solution is that many parameter values are equal or are relatively close to their minimum value. The Hooke and Jeeves method, with a general approach of starting with the minimum values and slowly climbing the hill, is much more successful than randomly searching even the smallest space. For the relatively simple manufacturing configuration, the PACRAN algorithm needs a large number of function evaluations to come close to the optimum solution obtained with the first method. The optimization runs performed for model 1, 2 and 3 never could improve on the optimum solution previously obtained by the PACOPT algorithm.

Table 7.10 shows, as an example, the optimization results for the MRP scheme for model 1, for the case of exponential processing times, the DCI cost calculation approach, and mean processing times for product 1 and 2, 42 and 6 min., respectively. In Table 7.10, we show the results obtained by both optimization algorithms, the PACOPT and the PACRAN algorithms. For each optimization by PACRAN, we indicate a total number of search boxes, the number of function evaluations in the first search box, the reduction of number of function evaluations in each next box. We also give for each group of parameters (z, k, r, τ) the initial range, the range reduction (when starting search in the new box) and the minimum range (refer to 6.8.2).

Table 7.10 Random optimization results for model 1
(MRP, exponential processing times, DCI, $1/\mu_1$ =42 min., $1/\mu_2$ =6 min.)

Optimization Method	Initial values of parameters for product 1 and 2	No. of boxes	• No. of function evaluations (FE) per first box • reduction of FE	Per each parameter: • range • range reduction • minimum range	Solution Cost - No. of FE Parameter setting
PACOPT (refer to Table c1-3)	(initial cost: 8122.59) z k r τ 0 M 1 0 0 M 1 0	-	-	-	3399.93 - 119 z k r τ 1 M 1 0 4 M 1 9
PACRAN	z k r τ 0 M 1 0 0 M 1 0	5	• 100 • 10	z k r τ • 10 0 6 20 • 1 0 1 2 • 2 0 2 10	3599.17 - 401 z k r τ 0 M 1 4 4 M 1 6
PACRAN	z k r τ 0 M 1 0 0 M 1 0	10	• 200 • 10	z k r τ • 10 0 6 40 • 1 0 1 1 • 2 0 2 10	3499.53 - 1551 z k r τ 0 M 1 0 4 M 1 1
PACRAN	z k r τ 0 M 1 0 0 M 1 0	10	• 300 • 20	z k r τ • 10 0 6 40 • 1 0 1 1 • 2 0 2 10	3499.53 - 2101 z k r τ 0 M 1 1 4 M 1 1
PACRAN	z k r τ 0 M 1 0 0 M 1 0	10	• 1000 • 50	z k r τ • 10 0 6 40 • 1 0 1 2 • 4 0 2 10	3399.93 - 7751 z k r τ 1 M 1 0 4 M 1 9

We could not improve the previously obtained result of 3399.93 with the random search method. In the case of model 1, PACRAN requires significantly more function evaluations (PAC simulations) to result in the solution value previously obtained by the

PACOPT algorithm. Based on the Hooke and Jeeves method, the PACOPT algorithm requires only 119 function evaluations in comparison with 7751, which in this case were executed by the PACRAN technique in order to find the optimum.

We also tested the PACRAN algorithm for the case of a more complex manufacturing configuration. Some selection of optimization runs for model 4A, are presented in Table 7.11. We perform the optimization for the variant A of model 4, for the MRP policy with

Table 7.11 Random optimization results for model 4A
(MRP, exponential processing times, no setup, DCII)

Optimization Method	Initial values of parameters for product 1 to 7	No. of boxes	• No. of function evaluations (FE) per first box • reduction of FE	Per each parameter: • range • range reduction • minimum range	Solution Cost - No. of FE Parameter setting
PACOPT (refer to Table c4-3)	(initial cost: 12192.77) z k r τ 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0	-	-	-	7360.96 - 429 z k r τ 0 M 1 181 0 M 1 377 0 M 1 236 0 M 1 0 0 M 1 0 0 M 1 141 0 M 1 0
PACRAN	z k r τ 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0	10	• 1000 • 100	z k r τ • 10 0 6 1000 • 2 0 1 50 • 2 0 2 10	8039.49 - 5501 z k r τ 0 M 1 273 0 M 1 210 0 M 1 229 0 M 1 80 0 M 1 75 0 M 1 160 0 M 1 9
PACRAN	z k r τ 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0	10	• 1000 • 50	z k r τ • 10 0 6 1000 • 2 0 1 100 • 2 0 2 50	7394.24 - 7751 z k r τ 0 M 1 194 0 M 1 426 0 M 1 259 0 M 1 2 0 M 1 7 0 M 1 162 0 M 1 18
PACRAN	z k r τ 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0	10	• 2000 • 100	z k r τ • 10 0 6 1000 • 2 0 1 100 • 2 0 2 50	7394.24 - 15501 z k r τ 0 M 1 194 0 M 1 426 0 M 1 259 0 M 1 2 0 M 1 7 0 M 1 162 0 M 1 18
PACRAN	z k r τ 0 M 1 181 0 M 1 377 0 M 1 236 0 M 1 0 0 M 1 0 0 M 1 141 0 M 1 0	10	• 1000 • 100	z k r τ • 10 0 6 1000 • 2 0 1 100 • 2 0 2 50	7278.44 - 5501 z k r τ 0 M 1 190 0 M 1 381 0 M 1 236 0 M 1 28 0 M 1 28 0 M 1 152 0 M 1 32

exponential processing times, DCII costs calculation and without setups. Here, we could not improve the result obtained by the PACOPT algorithm, when starting with the initial solution equal to the minimum parameter values. However, by starting with the computed optimal parameter setting by the PACOPT algorithm, and carefully choosing the dimensions of the search box, number of function evaluations, and reduction of the range of parameters, we can improve the final solution. In this case, the total improvement of costs was only about 1%. Unfortunately, the required number of function evaluations can become very large. This makes the application of the random algorithm relatively difficult and expensive.

Although the PACRAN algorithm requires a large number of function evaluations, it still can find some applications for trying to improve the already obtained acceptable solution.

7.9 Bulk Customer Demand

For all 5 models, the optimized solution for the bulk customer demand resulted in significant increase of the total cost when compared to non bulk (standard) arrivals. At first the magnitude of this increase seemed to be surprising. We resorted to queueing theory to make some comparisons and to shed some light on this phenomenon.

Before presenting the results of application of queuing theory to one of the models, we briefly summarize some theoretical formulations with regard to the BMAP/G/1 queue (BMAP=batch Markovian arrival process).

7.9.1 BMAP/G/1 Queue

The BMAP/G/1 queue is a single server queue with general independent and identically distributed (IID) processing times and a batch Markovian arrival process (BMAP).

Within the class of models that can be analyzed by M/G/1 type Markov chains is BMAP/G/1 queue. The BMAP/G/1 is a generalization of the M/G/1 model in which the Poisson arrival process is replaced by a batch Markovian arrival process (BMAP).

It is possible to derive the formula for the average length of the BMAP/G/1 queue. This comes from He (1997). We make the following assumptions:

Poisson input (arrival rate of batches) : λ

Successive batch sizes i have

- probability mass function : $\{p_i, i \geq 1\}$
- probability generating function : $p(z) = \sum_{i=1}^{\infty} p_i z^i$; $Ep = \sum i p_i$; $Ep^2 = \sum i^2 p_i$

Arrival rate of customers : $\hat{\lambda} = \lambda \sum_{i=1}^{\infty} i p_i = \lambda Ep$

Processing time : $\nu = F(x)$

Traffic intensity : ρ

From the original transform matrix (refer to Lucantoni, 1993) the expected queue length at departure is:

$$X(z) = \frac{(1-\rho)}{\hat{\lambda}} D(z) A(z)(z - A(z))^{-1}$$

where: $D(z) = \lambda(-1 + p(z))$ and $A(z) = \int_0^{\infty} \exp\{\lambda(-1 + p(z))x\} dF(x)$

Then

$$X'(z) = \frac{(1-\rho)}{\hat{\lambda}}\left[D'(z)A(z)(z-A(z))^{-1} + D(z)A'(z)(z-A(z))^{-1} - \frac{D(z)A(z)(1-A'(z))}{(z-A(z))^2}\right]$$

$$= \frac{(1-\rho)}{\hat{\lambda}}\left[\frac{D(z)}{(z-1)}\frac{(z-1)}{(z-A(z))}A'(z) + \frac{A(z)(z-1)^2}{(z-A(z))^2}\left(\frac{(z-A(z))D'(z)-D(z)(1-A'(z)}{(z-1)^2}\right)\right]$$

$$\lim_{z\to 1} X'(z) = \frac{(1-\rho)}{\hat{\lambda}}\left[\frac{D'(1)A'(z)}{1-A'(1)} + \frac{A(1)}{(1-A'(1))^2}\left(\frac{-A''(1)D'(1)+2(1-A'(1))D''(1)-D''(1-A'(1))+2D'(1)A''(1)}{2}\right)\right]$$

where: $A(1) = 1$ $\qquad$ $D(1) = 0$

$A'(1) = \lambda Ep EF(x) = \rho$ $\qquad$ $D'(1) = \lambda Ep$

$A''(1) = \lambda^2 (Ep)^2 Ev^2 + \lambda Ev(Ep^2 - Ep)$ $\qquad$ $D''(1) = \lambda(Ep^2 - Ep)$

$$X'(1) = \frac{(1-\rho)}{\hat{\lambda}}\left[\frac{\lambda Ep\rho}{1-\rho} + \frac{1}{2(1-\rho)^2}\left((1-\rho)\lambda(Ep^2-Ep) + \lambda Ep[\hat{\lambda}^2 Ev^2 + \lambda Ev(Ep^2-Ep)]\right)\right]$$

$$= \frac{(1-\rho)}{\hat{\lambda}}\left[\frac{\hat{\lambda}\rho}{1-\rho} + \frac{1}{2(1-\rho)^2}\left((1-\rho)\lambda(Ep^2-Ep) + \hat{\lambda}[\hat{\lambda}^2 Ev^2 + \lambda Ev(Ep^2-Ep)]\right)\right]$$

$$X'(1) = \rho + \frac{Ep^2 - Ep}{2(1-\rho)Ep} + \frac{\hat{\lambda}^2 Ev^2}{2(1-\rho)} \qquad \text{or} \qquad X'(1) = \rho + \frac{\hat{\lambda}^2 Ev^2}{2(1-\rho)} + \frac{Ep^2 - Ep}{2(1-\rho)Ep}$$

Variance: $Var(p) = Ep^2 - (Ep)^2$ then:

$$Ep^2 - Ep = Ep^2 - (Ep)^2 + (Ep)^2 - Ep = Var(p) + (Ep)^2 - Ep$$

and: $Ev^2 = Var(v) + (Ev)^2$

The formula including variances and means is as follows:

$$X'(1) = \rho + \frac{(\lambda Ep)^2 (Var(v) + (Ev)^2)}{2(1-\rho)} + \frac{Var(p) + (Ep)^2 - Ep}{2(1-\rho)Ep} \qquad [7.1]$$

Note: the last term is related to the batch size.

According to Lucantoni, the expected queue length at an arbitrary time is:

$$Y(z) = (1-\rho)(z-1)A(z)(z-A(z))^{-1}$$

$$Y'(z) = (1-\rho)\left[A(z)(z-A(z))^{-1} + \frac{z-1}{z-A(z)} A'(z) - \frac{(z-1)A(z)}{(z-A(z))^2}(1-A'(z)) \right]$$

$$= (1-\rho)\left[\frac{z-1}{z-A(z)} A'(z) + \frac{A(z)(z-1)^2}{(z-A(z))^2} \left(\frac{z-A(z)-(z-1)(1-A'(z))}{(z-1)^2} \right) \right]$$

$$Y'(z) = (1-\rho)\left[\frac{\rho}{1-\rho} + \frac{1}{2(1-\rho)^2}(-A''(1) + 2A''(1)) \right]$$

$$= \rho + \frac{1}{2(1-\rho)}\left[\hat{\lambda}^2 Ev^2 + (\lambda Ev)(Ep^2 - Ep) \right]$$

$$= \rho + \frac{\hat{\lambda}^2 Ev^2}{2(1-\rho)} + \frac{(Ep^2 - Ep)}{2(1-\rho)}(\lambda Ev)$$

$$Y'(1) = \rho + \frac{\hat{\lambda}^2 Ev^2}{2(1-\rho)} + \frac{Ep^2 - Ep}{2(1-\rho)Ep} \cdot \rho$$

The formula including variances and means is as follows:

$$Y'(1) = \rho + \frac{(\lambda Ep)^2 (Var(v) + (Ev)^2)}{2(1-\rho)} + \frac{Var(p) + (Ep)^2 - Ep}{2(1-\rho)Ep} \cdot \rho \qquad [7.2]$$

Formulas [7.1] and [7.2] are for the mean queue length including the one in server. The mean queue length for waiting customers (without the one in the server) is: $EX = EX - \rho$

This results in the mean queue length for waiting customers, without the one in the server:

- at departure time

$$X'(1) = \frac{(\lambda Ep)^2 (Var(v) + (Ev)^2)}{2(1-\rho)} + \frac{Var(p) + (Ep)^2 - Ep}{2(1-\rho)Ep} \qquad [7.3]$$

- at an arbitrarily time

$$Y'(1) = \frac{(\lambda Ep)^2 (Var(v) + (Ev)^2)}{2(1-\rho)} + \frac{Var(p) + (Ep)^2 - Ep}{2(1-\rho)Ep} \cdot \rho \qquad [7.4]$$

7.9.2 PAC System Modeled as a Queueing System

We want to model the PAC system as a queueing system in order to make some comparison studies with the theoretical formulations for BMAP/G/1 queues. We choose to consider model 1, the two-cell flow line, as the simplest manufacturing layout used in our study. In the case of the PTO policy, with a a fixed PAC parameter setting (z=0, k=100, r=1, τ=0), the arriving customer demand will immediately trigger a PA card at both cells and each cell will release a requisition tag. Thus, the customer demand will result in an arrival of a raw material (product 3) into the WIP queue of cell 1, if the transportation time will be set to 0. The WIP queue and the machine at cell 1 will form a single queue model with either a standard Poisson arrival process or a batch Markovian arrival process. The same will be true for MRP with r=1, for BSS with r=1, and for CONWIP with the value of parameter z_l large enough to satisfy every customer demand immediately, as in all these cases it will assure an arrival of a raw material to WIP 1 with each arrival of customer demand.

If our simulation results produce queueing behavior for this queue similar to the BMAP/G/1 analysis, this will serve as at least partial conforming of the validity of our simulation.

7.9.3 Theoretical Results

Using [7.4], we can calculate the mean queue length for waiting customers for the case of a standard Poisson demand (Ep=1 and $Var(p)$=0), and for the case of a bulk demand modeled by the geometric distribution with mean Ep=1 and variance $Var(p)$=2 (as for geom(p), mean =$(1-p)/p$ =1 $\rightarrow p$ =0.5 and variance = $(1-p)/p^2$ =2). We determine the queue lengths for the case of exponential processing times with variance $Var(p)=(E\nu\ \ )^2$, and uniform processing times with variance $Var(p)=(b-a)^2/12=1/3$. The results of those calculations are presented in Table 7.12.

The ratio indicates an increase in the mean queue length when customer demand changes from a standard Poisson to a bulk demand. We emphasize, that these results apply to an unlimited queue with a single machine with mean processing time equal to $1/\mu$.

Table 7.12 Mean queue lengths for waiting customers [$Y(1)$] for Poisson arrival with rate λ=1/60

$1/\mu$ [min.]	ρ	$Y(1)$: exponential standard demand [1]	$Y(1)$: exponential bulk demand [2]	Ratio [2]:[1]	$Y(1)$: uniform standard demand [3]	$Y(1)$: uniform bulk demand [4]	Ratio [4]:[3]
6	0.1	0.011	0.122	11.3	0.006	0.117	19.8
12	0.2	0.050	0.300	6.1	0.025	0.275	11.2
18	0.3	0.129	0.558	4.4	0.064	0.493	7.8
24	0.4	0.267	0.934	3.5	0.133	0.800	6.1
30	0.5	0.500	1.500	3.0	0.250	1.250	5.1
36	0.6	0.900	2.400	2.7	0.450	1.950	4.4
42	0.7	1.633	3.966	2.4	0.817	3.150	3.9

7.9.4 Simulation Results

We performed a number of simulation test runs for model 1 operating under the PTO policy with a fixed parameter setting (z=0, k=100, r=1, τ=0) and all transportation times set to 0 for different scenarios. We run the PAC simulations for model 1 for a total number of 600 days, including 40 days of warm-up. We vary the mean processing times for product 2 (in cell 1), but keep the mean processing time for product 1 (in cell 2) equal to 6 minutes. The simulation report generates the time average of each WIP queue. The mean queue length of the WIP queue of cell 1 is equivalent to the mean queue length, calculated for different scenarios in Table 7.12. The simulation results of the mean queue length for WIP 1 queue and the total cost are presented in Tables 7.13 and 7.14, for exponential and uniform processing times, respectively.

Table 7.13 Cost and the average length of the WIP 1 queue for PTO setting for model 1 with exponential processing times

Product 2 1/μ [min.]	standard demand		bulk demand		Ratio	Ratio
	WIP 1 [1]	Cost [2]	WIP 1 [3]	Cost [4]	WIP 1 [3]:[1]	Cost [4]:[2]
6	0.011	1180.8	0.126	2243.6	11.5	1.9
12	0.051	1938.8	0.307	3702.8	6.0	1.9
18	0.136	2970.7	0.560	5721.6	4.1	1.9
24	0.280	4383.6	0.920	8406.0	3.3	1.9
30	0.534	6508.3	1.531	12542.2	2.9	1.9
36	0.960	9690.7	2.518	19277.0	2.6	2.0
42	1.740	14995.7	4.192	30256.7	2.4	2.0

Table 7.14 Cost and the average length of the WIP 1 queue for PTO setting for model 1 with uniform processing times

Product 2 1/μ [min.]	standard demand		bulk demand		Ratio	Ratio
	WIP 1 [1]	Cost [2]	WIP 1 [3]	Cost [4]	WIP 1 [3]:[1]	Cost [4]:[2]
6	0.006	1085.3	0.118	1829.5	19.7	1.7
12	0.025	1722.8	0.279	3308.9	11.2	1.9
18	0.065	2490.0	0.499	5186.0	7.7	2.1
24	0.134	3441.8	0.804	7613.9	6.0	2.2
30	0.248	4668.7	1.258	10922.6	5.1	2.3
36	0.447	6404.5	1.985	15921.3	4.4	2.5
42	0.819	9202.0	3.217	24124.1	3.9	2.6

We also perform similar simulation runs for the BSS policy parameter setting (z=1, k=100, r=1, τ=0). Tables 7.15 and 7.16 give the mean queue length for WIP 1 queue and the total cost for exponential and uniform processing times, respectively.

The simulation results for the mean queue length of WIP 1 are practically identical to the theoretical calculations. The application of queueing theory not only explains the increased values of the costs, but at the same time validates our PAC simulation program. It is interesting to observe the ratio of costs for both scenarios of processing times. For PTO, in the case of exponential processing times, the ratio remains on a constant level for all tested cases of the mean processing time of product 2, and in the case of uniform processing times, the ratio increases with the increased value of the mean processing time value for product 2. For BSS, values of ratio of uniform processing times are identical to the PTO case with uniform processing times, while in the case of exponential processing times, the ratio tends to increase with the increased value of the mean processing time value for product 2.

We also performed a number of additional simulation runs for BSS and CONWIP to show how different values of parameters affect the mean queue length of the WIP 1 and the total costs. Tables 7.17 and 7.18 give the results for BSS and CONWIP respectively. All results are for the case of exponential processing times, for standard and bulk demand, and for the mean processing times of product 2 equal to 6 and 42 min. In all cases, the mean processing times of product 1 are equal to 6 min. In the case of BSS, we can observe that batching of PA cards increases the average length of WIP 1 and generally increases the costs. In the case of CONWIP, different scenarios require different values of parameter z_1 to assure an arrival of a raw matrial to WIP 1 with each arrival of customer demand.

Table 7.15 Cost and the average length of the WIP 1 queue for BSS setting for model 1 with exponential processing times

Product 2 1/μ [min.]	standard demand WIP 1 [1]	standard demand Cost [2]	bulk demand WIP 1 [3]	bulk demand Cost [4]	Ratio WIP 1 [3]:[1]	Ratio Cost [4]:[2]
6	0.011	3108.5	0.126	2428.7	11.5	0.8
12	0.051	3045.5	0.307	4011.9	6.0	1.3
18	0.127	3102.4	0.560	6184.1	4.4	2.0
24	0.260	3389.9	0.920	9082.0	3.5	2.7
30	0.484	4152.4	1.514	13498.8	3.1	3.3
36	0.896	6020.3	2.518	20624.9	2.8	3.4
42	1.606	9758.4	4.192	32178.3	2.6	3.3

Table 7.16 Cost and the average length of the WIP 1 queue for BSS setting for model 1 with uniform processing times

Product 2 1/μ [min.]	standard demand WIP 1 [1]	standard demand Cost [2]	bulk demand WIP 1 [3]	bulk demand Cost [4]	Ratio WIP 1 [3]:[1]	Ratio Cost [4]:[2]
6	0.006	1168.5	0.118	1971.7	19.7	1.7
12	0.025	1856.7	0.279	3571.55	11.2	1.9
18	0.065	2684.3	0.499	5601.15	7.7	2.1
24	0.134	3708.9	0.804	8214.71	6.0	2.2
30	0.248	5027.2	1.258	11761.81	5.1	2.3
36	0.447	6900.1	1.985	17072.88	4.4	2.5
42	0.819	9914.9	3.217	25776.44	3.9	2.6

Table 7.17 Cost and the average length of the WIP 1 queue for BSS setting with exponential processing times and for different values of parameter r

	standard demand				bulk demand			
	$1/\mu_2 = 6$		$1/\mu_2 = 42$		$1/\mu_2 = 6$		$1/\mu_2 = 42$	
	WIP 1	Cost	WIP 1	Cost	WIP 1	Cost	WIP 1	Cost
r_1=1, r_2=1	0.011	3108.5	1.606	9758.4	0.126	2428.7	4.192	32178.3
r_1=2, r_2=1	0.053	2412.2	1.792	11405.9	0.142	5410.0	4.304	35647.2
r_1=3, r_2=1	0.103	4433.0	1.999	14951.3	0.166	8361.2	4.419	39321.9
r_1=1, r_2=2	0.053	2569.7	1.792	11483.0	0.142	5410.0	4.304	35647.3
r_1=1, r_2=3	0.103	2184.4	1.998	13904.9	0.166	8361.2	4.419	39321.9
r_1=2, r_2=3	0.105	2956.1	2.085	16625.3	0.178	4041.5	4.234	31011.8
r_1=3, r_2=2	0.118	4120.7	2.162	17496.3	0.179	5027.3	4.366	32252.8

The value of z_1 ranges from 3, for standard demand and the mean processing time of product 2 equal to 6 min., to about 60, for the bulk demand case with the mean processing time of product 2 equal to 42 min. Setting a limit on the initial inventory in store 2 (for final product 1) decreases the average length of WIP 1 and generally reduces the costs. In both cases studied, the different values of PAC parameters, r (for BSS) and z_1 (for CONWIP), change the arrival pattern of the customer demand to the first WIP queue. At that moment, the WIP 1 queue does not show any more queueing behavior similar to the BMAP/G/1 analysis.

Now, let us examine the costs and some additional and relevant data taken from the simulation report for the optimum found for policies PAC, MRP, Kanban and IC for model 1, uniform processing times and DCI cost calculation approach (refer also to Tables c1-1 and c1-3 in Appendix C1).

Table 7.18 Cost and the average length of the WIP 1 queue for CONWIP setting with exponential processing times and for different values of parameter z_1

	standard demand				bulk demand			
	$1/\mu_2 = 6$		$1/\mu_2 = 42$		$1/\mu_2 = 6$		$1/\mu_2 = 42$	
z_1	WIP 1	Cost	WIP 1	Cost	WIP 1	Cost	WIP 1	Cost
70							4.192	150302.2
60							4.192	127902.7
50							4.189	105525.6
25			1.740	52249.1			4.027	50537.9
20			1.740	41049.2	0.126	44207.29	3.883	40235.2
15			1.735	29881.0	0.126	33009.9	3.613	30753.4
10			1.663	19167.3	0.124	21823.7	2.776	22992.3
5	0.011	10739.1	1.270	10514.8	0.105	10762.1	1.875	19765.4
3	0.011	6262.6	0.814	9092.5	0.070	6596.3	1.044	20743.9
2	0.009	4043.5	0.453	9618.2	0.040	4728.3	0.537	23204.1
1	0.000	1998.3	0.000	15809.9	0.000	3425.7	0.000	38027.0

Table 7.19 presents, for each combination of mean processing times studied, the following:

- total cost;
- average delay per final product demanded by a customer;
- time average number of products in CUST queue, and in all WIP queues, that is the mean queue length of respectively the CUST-queue and the WIP-queues;
- relevant ratios.

Table 7.19 Additional data for some optimized scenarios of model 1

Policy	uniform:	standard	demand	uniform:	bulk	demand	ratios		
$1/\mu_1$ - $1/\mu_2$	Cost	Product Delay	CUST/WIP queue	Cost	Product Delay	CUST/WIP queue	Cost	Product Delay	CUST/WIP queue
case I	[1]	[2]	[3]	[4]	[5]	[6]	[4] : [1]	[5] : [2]	[6] : [3]
PAC / 6 - 6	542.5	12.4	0.209 / 0.001	860.1	19.6	0.329 / 0.006	1.6	1.3	1.6 / 6
MRP / 6 - 6	545.5	12.4	0.209 / 0.006	921.8	19.5	0.328 / 0.126	1.7	1.6	1.6 / 21
Kan / 6 - 6	1416.7	0.3	0.006 / 0.000	1725.4	7.5	0.126 / 0.000	1.2	25	21 / NA
IC / 6 - 6	897.2	1.5	0.026 / 0.000	1571.2	17.0	0.287 / 0.000	1.8	11.3	11 / NA
case II									
PAC / 42 - 6	2252.1	23.2	0.430 / 0.359	7502.0	102.1	1.797 / 0.505	3.3	4.4	4.2 / 1.4
MRP / 42 - 6	2700.4	23.3	0.431 / 0.791	10630.9	102.1	1.800 / 3.570	3.9	4.4	4.2 / 4.5
Kan / 42 - 6	2717.0	44.5	0.791 / 0.000	9074.2	119.3	2.091 / 1.448	3.3	2.7	2.6 / NA
IC / 42 - 6	2253.7	92.9	1.629 / 0.360	8651.5	122.4	2.143 / 1.426	3.8	1.3	1.3 / 4
case III									
PAC / 6 - 42	1535.7	15.1	0.285 / 0.010	5541.8	69.1	1.237 / 0.000	3.6	4.6	4.3 / NA
MRP / 6 - 42	2289.7	23.2	0.430 / 0.790	7424.8	68.8	1.233 / 3.645	3.2	3	2.9 / 4.6
Kan / 6 - 42	1733.0	23.3	0.432 / 0.000	6831.3	102.2	1.800 / 0.000	3.9	4.4	4.2 / NA
IC / 6 - 42	2066.3	23.2	0.430 / 0.000	6878.7	75.3	1.343 / 2.226	3.3	3.2	3.1 / NA
case IV									
PAC / 42 - 42	2013.0	21.2	0.396 / 0.013	7151.3	100.9	1.777 / 0.026	3.6	4.8	4.8 / 2
MRP / 42 - 42	2535.9	20.5	0.385 / 0.803	9016.0	100.9	1.778 / 3.604	3.6	4.9	4.6 / 4.5
Kan / 42 - 42	2013.0	21.2	0.396 / 0.013	7151.3	100.9	1.777 / 0.026	3.6	4.8	4.8 / 2
IC / 42 - 42	2335.7	20.5	0.385 / 0.418	8091.6	100.9	1.778 / 1.826	3.5	4.9	4.6 / 4.4

NA - not applicable

We are aware that the results in Table 7.19 are for a much more complex model that the single server unlimited queue, but they indicate the general trends in relation between costs and other characteristic system data.

The cost ratio for the case of mean processing time for product 2 is equal to 6 and 42 min. and product 1 is equal to 6 min. from Tables 7.14 and 7.16 (uniform processing times) are 1.7 and 2.6, respectively. Case I from Table 7.19 presents results for different polices (PAC, MRP, Kanban and IC) when mean processing times of product 2 and 1 is equal to 6 minutes. The costs ratios for policies from case I can be compared with the value 1.7 from Table 7.14. We see that only for Kanban, the cost ratio (equal to 1.2) is a little bit lower. Case III results can be compared with the value 2.6 of the cost ratio from Table 7.14.. For the four policies, the cost ratios are higher than the PTO or BSS results.

It is very difficult to present any "ready for use" formula to explain the increase of costs for the case of bulk customer demand. However the more detailed analysis of relevant data shows some relations between theory and the empirical results and maybe in the future can lead to some middle ground formulations.

7.10 Justification of a Single Simulation Run

In the optimization process, the cost function evaluation is performed by a single simulation run with fixed stream numbers. The main concern is the robustness of the optimal solution obtained in this manner. Therefore we execute additional optimization runs with different stream numbers to observe the value of the total costs and the parameter settings. We perform these optimization runs for model 1 and 4 as examples of a simple and a more complex layout, respectively. For both models, we chose to do it for the BSS policy, which requires optimization of 4 and 14 parameters for models 1 and model 4 respectively. Table 7.20 contains the optimization results for BSS for model 1. We executed five different optimizations for four different initial solutions. As must be the case, the total cost is different for each of the five different optimizations, however the optimal parameters setting is always the same. The optimization results initiated with different starting solution have different values of the initial cost, but are identical across the same optimization case (the same stream numbers). For the five different values of the total costs the mean value is 5096.2 and the 95% confidence interval is [4759.9, 5432.5]. The BSS solution of 5017.97 is ranked as sixth and eighth for DCI and DCII of the delay cost scenario respectively (refer to Tables c1-1 and c1-2 in Appendix C1). As the cost values of the worse and the better policies in the overall ranking are very close to the BSS policy solution and the 95% confidence interval is quite large, it is impossible to make any conclusions about obtained results. Therefore, we perform optimization runs for all 9 coordination policies, exponential processing times, DCII cost scenario for five different values of stream numbers. The results of these optimizations are given in Table 7.21. In all 5 cases the sequence of the policies is almost identical, except runs 2 and 5 where Kanban is just slightly better than IC/CONWIP. Note how policies with $\tau=0$ have identical optimal parameter settings across different optimization runs, while policies with $\tau\geq0$ have different parameter solutions for each optimization run.

In the case of model 4, we perform very similar optimization test runs with results given in Tables 7.22 and 7.23. First, we generate the optimization results for the BSS policy, using four different initial solutions for five different optimization runs, each with fixed, but different stream numbers. These results are summarized in Table 7.22.

Table 7.20 Optimization results for model 1, for BSS, exponential processing times, $1/\mu_1 = 42$ min., $1/\mu_2 = 6$ min.

Optimization Run (with different streams numbers)		Initial Solution			
		1	2	3	4
		z k r s 1 M 1 0 1 M 1 0	z k r s 2 M 2 0 2 M 2 0	z k r s 3 M 3 0 3 M 3 0	z k r s 5 M 5 0 5 M 5 0
1	Initial cost: Solution:	6255.08 5017.97 -25 z k r s 3 M 1 0 1 M 2 0	7080.13 5017.97 -30 z k r s 3 M 1 0 1 M 2 0	7784.00 5017.97 -90 z k r s 3 M 1 0 1 M 2 0	9133.53 5017.97 -87 z k r s 3 M 1 0 1 M 2 0
2	Initial cost: Solution:	6604.01 4858.79 -46 z k r s 3 M 1 0 1 M 2 0	6438.93 4858.79 -46 z k r s 3 M 1 0 1 M 2 0	7612.24 4858.79 -110 z k r s 3 M 1 0 1 M 2 0	8892.19 4858.79 -135 z k r s 3 M 1 0 1 M 2 0
3	Initial cost: Solution:	5810.99 4813.66 -25 z k r s 3 M 1 0 1 M 2 0	6355.99 4813.66 -28 z k r s 3 M 1 0 1 M 2 0	7342.94 4813.66 -40 z k r s 3 M 1 0 1 M 2 0	8928.91 4813.66 -65 z k r s 3 M 1 0 1 M 2 0
4	Initial cost: Solution:	7641.80 5761.05 -25 z k r s 3 M 1 0 1 M 2 0	8218.61 5761.05 -29 z k r s 3 M 1 0 1 M 2 0	9240.88 5761.05 -57 z k r s 3 M 1 0 1 M 2 0	10959.52 5761.05 -92 z k r s 3 M 1 0 1 M 2 0
5	Initial cost: Solution:	6282.32 5029.57 -25 z k r s 3 M 1 0 1 M 2 0	6557.53 5029.57 -30 z k r s 3 M 1 0 1 M 2 0	7344.63 5029.57 -48 z k r s 3 M 1 0 1 M 2 0	8564.24 5029.57 -73 z k r s 3 M 1 0 1 M 2 0

For each optimization case, we obtained the same optimization results for four different initial parameter settings. The optimal parameter settings were identical for cases 1 and 2 - solution A, and for cases 3, 4 and 5 - solution B. We calculated the total cost by setting the parameters according to the solution A for optimization runs 3, 4 and 5 and according to the solution B for optimization runs 1 and 2 (refer to Table 7.22). For the five different values of the total cost for the solution A, the mean value is 16538.5 and the 95% confidence interval is [15906.0, 17171.0]. For the five different values of the total cost for the solution B, the mean value is 16498.6 and the 95% confidence interval is [15814.3, 17182.9]. The BSS solution of 16297.58 is better than PTO with 9162.75 and worse than IC with 19367.25 (refer to Tables c4-1 and c4-2 in Appendix C4). The differences between these three cost values are relatively significant. This suggests that the sequence of these three policies should be the same for different optimization runs. The results in Table 7.23 confirm that for the five different optimization scenarios. As well, they show that the sequence of all 9 policies is the same for all five optimization cases tested.

The performed sample of optimization test runs for two different models indicate that the obtained results are quite robust, especially if the differences of the cost values between policies are not too close to each other.

Table 7.21 Optimization results for model 1, exponential processing times, DCII
$1/\mu_1 = 42$ min., $1/\mu_2 = 6$ min.

Best Solution	Optimization Run	(with different	stream numbers)		
	1	2	3	4	5
1	**PAC** 4180.27 - 142 z k r τ 2 2 1 69 0 1 1 0	**PAC** 3910.55 - 126 z k r τ 2 2 1 56 0 3 1 2	**PAC** 3780.90 - 218 z k r τ 0 4 1 162 0 3 1 20	**PAC** 4452.72 - 158 z k r τ 0 2 1 179 0 2 1 2	**PAC** 3740.40 - 172 z k r τ 2 2 1 67 0 1 1 8
2	**LC** 4407.07 - 21 z k r τ 3 1 1 0 2 1 1 0	**LC** 4347.42 - 21 z k r τ 3 1 1 0 2 1 1 0	**LC** 4135.61 - 21 z k r τ 3 1 1 0 2 1 1 0	**LC** 4949.56 - 21 z k r τ 3 1 1 0 2 1 1 0	**LC** 4477.46 - 21 z k r τ 3 1 1 0 2 1 1 0
3	**IC** 4705.03 - 14 z k r τ 3 3 1 0 0 3 1 0	**Kanban** 4455.82 - 21 z k r τ 2 2 1 0 1 1 1 0	**IC** 4181.25 - 14 z k r τ 3 3 1 0 0 3 1 0	**IC** 5004.24 - 14 z k r τ 3 3 1 0 0 3 1 0	**Kanban** 4518.00 - 21 z k r τ 2 2 1 0 1 1 1 0
4	**CONWIP** 4705.03 - 10 z k r τ 3 3 1 0 0 3 1 0	**IC** 4499.60 - 14 z k r τ 3 3 1 0 0 3 1 0	**CONWIP** 4181.25 - 10 z k r τ 3 3 1 0 0 3 1 0	**CONWIP** 5004.24 - 10 z k r τ 3 3 1 0 0 3 1 0	**IC** 4561.50 - 14 z k r τ 3 3 1 0 0 3 1 0
5	**Kanban** 4805.23 - 21 z k r τ 2 2 1 0 1 1 1 0	**CONWIP** 4499.60 - 10 z k r τ 3 3 1 0 0 3 1 0	**Kanban** 4242.37 - 21 z k r τ 2 2 1 0 1 1 1 0	**Kanban** 5437.67 - 21 z k r τ 2 2 1 0 1 1 1 0	**CONWIP** 4561.50 - 10 z k r τ 3 3 1 0 0 3 1 0
6	**MRP *** 4991.14 - 25 z k r τ 2 M 1 70 0 M 1 0	**MRP** 4614.96 - 89 z k r τ 0 M 1 201 0 M 2 40	**MRP** 4248.26 - 84 z k r τ 2 M 1 52 0 M 1 10	**MRP *** 5685.92 - 16 z k r τ 0 M 1 184 0 M 1 2	**MRP** 4746.18 - 121 z k r τ 2 M 2 92 0 M 1 0
7	**PTO, τ≥0** 4991.14 - 55 z k r τ 0 M 1 168 0 M 1 3	**PTO, τ≥0** 4820.65 - 35 z k r τ 0 M 1 162 0 M 1 0	**PTO, τ≥0** 4372.44 - 39 z k r τ 0 M 1 154 0 M 1 0	**PTO, τ≥0** 5685.92 - 53 z k r τ 0 M 1 184 0 M 1 2	**PTO, τ≥0** 4946.88 - 49 z k r τ 0 M 1 176 0 M 1 19
8	**BSS** 5017.97 - 25 z k r τ 3 M 1 0 1 M 2 0	**BSS** 4858.79 - 46 z k r τ 3 M 1 0 1 M 2 0	**BSS** 4813.66 - 25 z k r τ 3 M 1 0 1 M 2 0	**BSS** 5761.05 - 25 z k r τ 3 M 1 0 1 M 2 0	**BSS** 5029.57 - 25 z k r τ 3 M 1 0 1 M 2 0
9	**PTO** 8023.74 - 1 z k r τ 0 M 1 0 0 M 1 0	**PTO** 7694.69 - 1 z k r τ 0 M 1 0 0 M 1 0	**PTO** 7171.35 - 1 z k r τ 0 M 1 0 0 M 1 0	**PTO** 9323.47 - 1 z k r τ 0 M 1 0 0 M 1 0	**PTO** 8001.21 - 1 z k r τ 0 M 1 0 0 M 1 0

Table 7.22 Optimization results for model 4, for BSS, exponential processing times, variant A, no setup

Optimization Run (with different stream numbers)		Initial Solution			
		1	2	3	4
		z k r s 1 M 1 0 1 M 1 0 1 M 1 0 1 M 1 0 1 M 1 0 1 M 1 0 1 M 1 0	z k r s 2 M 2 0 2 M 2 0 2 M 2 0 2 M 2 0 2 M 2 0 2 M 2 0 2 M 2 0	z k r s 3 M 3 0 3 M 3 0 3 M 3 0 3 M 3 0 3 M 3 0 3 M 3 0 3 M 3 0	z k r s 5 M 5 0 5 M 5 0 5 M 5 0 5 M 5 0 5 M 5 0 5 M 5 0 5 M 5 0
1	Initial cost: Solution: **Solution A** Cost of solution B 17440.93	22978.35 16297.58-70 z k r s 1 M 2 0 1 M 2 0 1 M 2 0 1 M 2 0 1 M 1 0 1 M 1 0 1 M 1 0	35476.14 16297.58-234 z k r s 1 M 2 0 1 M 2 0 1 M 2 0 1 M 2 0 1 M 1 0 1 M 1 0 1 M 1 0	51295.05 16297.58-173 z k r s 1 M 2 0 1 M 2 0 1 M 2 0 1 M 2 0 1 M 1 0 1 M 1 0 1 M 1 0	77748.44 16297.58-494 z k r s 1 M 2 0 1 M 2 0 1 M 2 0 1 M 2 0 1 M 1 0 1 M 1 0 1 M 1 0
2	Initial cost: Solution: **Solution A** Cost of solution B 16616.98	22602.85 16619.62-71 z k r s 1 M 2 0 1 M 2 0 1 M 2 0 1 M 2 0 1 M 1 0 1 M 1 0 1 M 1 0	35917.57 16619.62-199 z k r s 1 M 2 0 1 M 2 0 1 M 2 0 1 M 2 0 1 M 1 0 1 M 1 0 1 M 1 0	49626.30 16619.62-234 z k r s 1 M 2 0 1 M 2 0 1 M 2 0 1 M 2 0 1 M 1 0 1 M 1 0 1 M 1 0	77435.55 16619.62-326 z k r s 1 M 2 0 1 M 2 0 1 M 2 0 1 M 2 0 1 M 1 0 1 M 1 0 1 M 1 0
3	Initial cost: Solution: **Solution B** Cost of solution A 15876.77	22708.48 15658.85-113 z k r s 1 M 2 0 1 M 3 0 1 M 2 0 1 M 2 0 1 M 2 0 1 M 1 0 1 M 1 0	36015.88 15658.85-247 z k r s 1 M 2 0 1 M 3 0 1 M 2 0 1 M 2 0 1 M 2 0 1 M 1 0 1 M 1 0	49878.39 15658.85-178 z k r s 1 M 2 0 1 M 3 0 1 M 2 0 1 M 2 0 1 M 2 0 1 M 1 0 1 M 1 0	77599.38 15658.85-291 z k r s 1 M 2 0 1 M 3 0 1 M 2 0 1 M 2 0 1 M 2 0 1 M 1 0 1 M 1 0
4	Initial cost: Solution: **Solution B** Cost of solution A 17737.52	22777.74 17023.19-111 z k r s 1 M 2 0 1 M 3 0 1 M 2 0 1 M 2 0 1 M 2 0 1 M 1 0 1 M 1 0	36605.77 17023.19-220 z k r s 1 M 2 0 1 M 3 0 1 M 2 0 1 M 2 0 1 M 2 0 1 M 1 0 1 M 1 0	50283.46 17023.19-356 z k r s 1 M 2 0 1 M 3 0 1 M 2 0 1 M 2 0 1 M 2 0 1 M 1 0 1 M 1 0	76667.21 17023.19-413 z k r s 1 M 2 0 1 M 3 0 1 M 2 0 1 M 2 0 1 M 2 0 1 M 1 0 1 M 1 0
5	Initial cost: Solution: **Solution B** Cost of solution A 16161.23	22315.40 15753.18-113 z k r s 1 M 2 0 1 M 3 0 1 M 2 0 1 M 2 0 1 M 2 0 1 M 1 0 1 M 1 0	36999.90 15753.18-241 z k r s 1 M 2 0 1 M 3 0 1 M 2 0 1 M 2 0 1 M 2 0 1 M 1 0 1 M 1 0	50645.23 15753.18-278 z k r s 1 M 2 0 1 M 3 0 1 M 2 0 1 M 2 0 1 M 2 0 1 M 1 0 1 M 1 0	77981.30 15753.18-455 z k r s 1 M 2 0 1 M 3 0 1 M 2 0 1 M 2 0 1 M 2 0 1 M 1 0 1 M 1 0

Table 7.23 Optimization results for model 4, exponential processing times, DCII, variant A, no setup

Best Solution	Optimization Run (with different stream numbers) 1	2	3	4	5
1	**PAC** 5012.81 - 672 z k r τ 0 3 1 184 0 11 1 408 0 3 1 207 0 3 1 20 0 3 1 20 0 1 1 0 0 3 1 0	**PAC** 5920.48 - 703 z k r τ 0 4 1 147 0 1 1 302 0 3 1 178 0 5 1 20 0 3 1 20 0 15 1 0 2 7 1 59	**PAC** 4886.94 - 708 z k r τ 0 21 1 223 0 1 1 344 0 2 1 206 0 15 1 58 0 13 1 60 0 1 1 0 0 3 1 10	**PAC** 5613.51 - 1141 z k r τ 0 15 1 137 0 1 1 294 0 11 1 111 0 5 1 0 0 9 1 0 0 2 1 0 3 3 1 11	**PAC** 4712.29 - 784 z k r τ 0 21 1 175 0 1 1 405 0 2 1 215 0 1 1 21 0 7 1 22 0 1 1 140 0 3 1 20
2	**MRP** 7360.96 - 429 z k r τ 0 M 1 181 0 M 1 377 0 M 1 236 0 M 1 0 0 M 1 0 0 M 1 141 0 M 1 0	**MRP** 7302.22 - 364 z k r τ 0 M 1 122 0 M 1 245 0 M 1 106 0 M 1 0 0 M 1 0 0 M 1 20 4 M 2 0	**MRP** 6272.93 - 668 z k r τ 0 M 1 130 0 M 1 239 0 M 1 107 0 M 1 11 0 M 1 11 0 M 1 0 4 M 2 11	**MRP** 7449.12 - 500 z k r τ 0 M 1 173 0 M 1 230 0 M 1 144 0 M 1 57 0 M 1 100 0 M 1 0 0 M 1 18	**MRP** 6205.15 - 515 z k r τ 0 M 1 161 0 M 1 282 0 M 1 163 0 M 1 44 0 M 1 59 0 M 1 20 2 M 1 40
3	**PTO, τ≥0** 7360.96 - 202 z k r τ 0 M 1 181 0 M 1 377 0 M 1 236 0 M 1 0 0 M 1 0 0 M 1 141 0 M 1 0	**PTO, τ≥0** 7441.19 - 245 z k r τ 0 M 1 167 0 M 1 357 0 M 1 216 0 M 1 20 0 M 1 20 0 M 1 0 0 M 1 0	**PTO, τ≥0** 6310.32 - 322 z k r τ 0 M 1 231 0 M 1 321 0 M 1 221 0 M 1 59 0 M 1 40 0 M 1 157 0 M 1 20	**PTO, τ≥0** 7704.10 - 315 z k r τ 0 M 1 249 0 M 1 336 0 M 1 274 0 M 1 97 0 M 1 119 0 M 1 240 0 M 1 20	**PTO, τ≥0** 6248.12 - 212 z k r τ 0 M 1 174 0 M 1 384 0 M 1 231 0 M 1 1 0 M 1 2 0 M 1 0 0 M 1 0
4	**PTO** 9162.75	**PTO** 9227.26	**PTO** 8286.29	**PTO** 9905.43	**PTO** 7908.93
5	**BSS** 16297.58 - 70 z k r τ 1 M 2 0 1 M 2 0 1 M 2 0 1 M 2 0 1 M 1 0 1 M 1 0 1 M 1 0	**BSS** 16619.62 - 70 z k r τ 1 M 2 0 1 M 2 0 1 M 2 0 1 M 2 0 1 M 1 0 1 M 1 0 1 M 1 0	**BSS** 15658.85 - 113 z k r τ 1 M 2 0 1 M 3 0 1 M 2 0 1 M 2 0 1 M 2 0 1 M 1 0 1 M 1 0	**BSS** 17023.19 - 111 z k r τ 1 M 2 0 1 M 3 0 1 M 2 0 1 M 2 0 1 M 2 0 1 M 1 0 1 M 1 0	**BSS** 15753.18 - 113 z k r τ 1 M 2 0 1 M 3 0 1 M 2 0 1 M 2 0 1 M 2 0 1 M 1 0 1 M 1 0
6	**IC** 19367.25 - 33 z k r τ 1 1 1 0 1 1 1 0 1 1 1 0 1 2 1 0 0 5 1 0 0 2 1 0 1 16 1 0	**IC** 18968.08 - 15 z k r τ 1 1 1 0 1 1 1 0 1 1 1 0 1 2 1 0 0 5 1 0 0 2 1 0 0 15 1 0	**IC** 19447.03 - 51 z k r τ 1 1 1 0 1 1 1 0 1 1 1 0 1 2 1 0 0 5 1 0 0 2 1 0 1 16 1 0	**IC** 20032.53 - 34 z k r τ 1 1 1 0 1 1 1 0 1 1 1 0 1 2 1 0 0 5 1 0 0 2 1 0 2 17 1 0	**IC** 19093.32 - 33 z k r τ 1 1 1 0 1 1 1 0 1 1 1 0 1 2 1 0 0 5 1 0 0 2 1 0 1 16 1 0
7	**CONWIP** 19516.47 - 9 z k r τ 1 1 1 0 1 1 1 0 1 1 1 0 1 2 1 0 0 5 1 0 0 2 1 0 0 15 1 0	**CONWIP** 18968.08 - 9 z k r τ 1 1 1 0 1 1 1 0 1 1 1 0 1 2 1 0 0 5 1 0 0 2 1 0 0 15 1 0	**CONWIP** 19828.85 - 9 z k r τ 1 1 1 0 1 1 1 0 1 1 1 0 1 2 1 0 0 5 1 0 0 2 1 0 0 15 1 0	**CONWIP** 20837.41 - 21 z k r τ 2 2 1 0 1 1 1 0 1 1 1 0 1 3 1 0 0 6 1 0 0 2 1 0 0 18 1 0	**CONWIP** 19210.58 - 9 z k r τ 1 1 1 0 1 1 1 0 1 1 1 0 1 2 1 0 0 5 1 0 0 2 1 0 0 15 1 0
8	**Kanban** 23907.48 - 83 z k r τ 3 3 1 0 1 1 1 0 1 1 1 0 1 1 1 0 2 2 1 0 1 1 1 0 1 1 1 0	**Kanban** 27111.11 - 102 z k r τ 3 3 1 0 1 1 1 0 3 3 2 0 3 3 3 0 2 2 1 0 1 1 1 0 1 1 1 0	**Kanban** 23907.48 - 83 z k r τ 3 3 1 0 1 1 1 0 1 1 1 0 1 1 1 0 2 2 1 0 1 1 1 0 1 1 1 0	**Kanban** 20967.28 - 90 z k r τ 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 2 2 1 0	**Kanban** 24282.20 - 38 z k r τ 3 3 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0
9	**LC** 43310.17 - 34 z k r τ 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 4 1 1 0 2 1 1 0 2 1 1 0	**LC** 45995.99 - 29 z k r τ 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0	**LC** 46257.27 - 29 z k r τ 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0	**LC** 45979.24 - 29 z k r τ 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0	**LC** 46313.70 - 29 z k r τ 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0

7.11 Concluding Remarks

With our optimization procedure, we are able to determine the sequence of the policies for a given realization of the manufacturing system operating in a given (not too long) time period. However, we are not able to determine the sequence of the policies for all realizations of the system and we do not know if it will ever be possible to solve this problem, especially for more complex manufacturing layouts.

We had tested the working of the algorithm on quite a large number of situations for 5 different manufacturing configurations. For many scenarios the parameter setting is the global optimum, as for policies PTO, LC, CONWIP, where number of parameters for optimization is fixed or very small. For some others, we know only that we obtained one of possibly many local optima. We did not exhaust all possibilities to search for better result within a certain control policy, as we wanted to perform many optimization runs for different scenarios for all 5 models and we were limited by time. When the result obtained seemed to be a good approximation of the optimum, we did accept it as the solution.

We performed all the test runs in order to develop and gradually improve performance of the algorithm, and in order to present its wide range of possible future applications.

In the case of a system with relatively many parameters to be optimized, it becomes more difficult to obtain a satisfactory solution. We need to perform many runs with different step lengths of parameters, which still do not guarantee success. Therefore, it is useful first to optimize a policy scheme which requires a smaller number of parameters, and then use the obtained optimal parameter values as the initial solution for the more complex and demanding case.

The issue of an initial solution and step lengths of parameters used for optimization is the most difficult to approach. The human perception and gaining knowledge through experimenting are much faster than any computerized procedure. It is clear, that after computing of many optimization runs for a certain layout, a researcher can get quite a good feeling about the choice of starting values for parameters, their step lengths and their maximum values. In the case of each model, we performed a number of optimizations with different step lengths of parameters and sometimes with different initial solution to examine which values give the best results. The PTO solution was used as a first basis reference. The simulation results of the best solution obtained by the optimization of the next policy was compared with the results for PTO.

We generated many results, which illustrate working of various policies in the same environment. We could observe how performance of different policies vary with the changed processing situation. We used the queueing theory approach to explain effects of bulk demand on the performance of the system.

Based on the experiments performed for 5 manufacturing layouts, we can conclude that the algorithm gives the optimization framework to find (approximately) optimal parameter settings for various control policies operating under given conditions and is quite feasible for models with a small number of products (less than 20). It performs quite well in comparison with the random search technique, as it requires significantly less function evaluation and generates results equal or relatively close to the global optimum. The algorithm facilitates studies on systems of greater complexity and where the distributional assumptions necessary for queueing are not really appropriate. By the development of the PAC optimization algorithm, we constructed a laboratory environment which enables us to examine the behaviour of many different manufacturing control systems of almost any realistic situation for systems not too large.

Chapter 8

EXAMINATION OF PAC SYSTEM PROPERTIES

8.1 Introduction

In this chapter we examine the properties of the PAC system, described in Chapter 4, (section 4.2.8). The five different properties describe the impacts of certain choices of PAC parameters (z, k, r, τ) and processing times (S) on the system performance, and specifically on the characteristics of the departure process and the shipment to customers process.

Buzacott and Shanthikumar explained the PAC system dynamics considering a PAC system consisting of multiple cells in series. They developed formulas for the time when process tags and material are moved from cells to stores. The five properties can be verified from those formulas. In this chapter, we first confirm that one simulation model shares these properties. We then go on to examine if they hold for the case of more complex manufacturing layouts.

Before presenting results of an examination study for both models, first we outline general characteristics of the experiment.

8.2 Design of Experiment

The five PAC properties can be phrased either in terms of the departure times of products from cells (after completion of processing) or delays in the shipment to customers process (see Buzacott and Shanthikumar, 1992b, page 494). We chose to examine these properties only with respect to the shipment to customers process, since this is easier to obtain from our simulation output. We consider our analysis in terms of shipment delays, that is, the delays in filling a requisition from a customer.

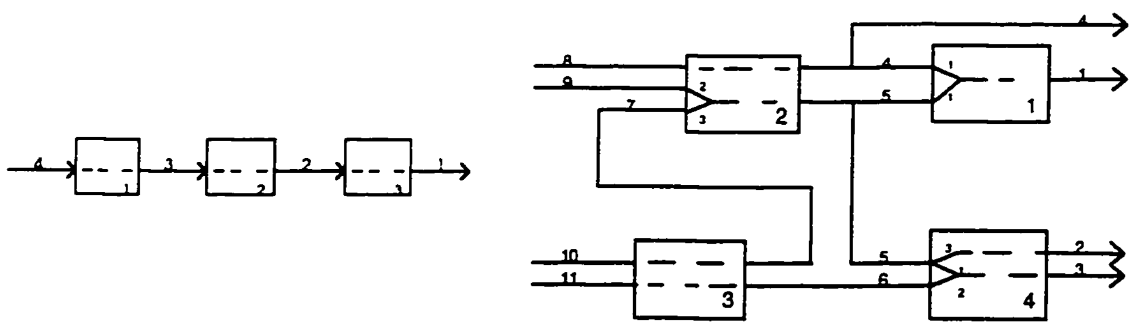
The PAC simulation model produces sufficient data to analyze the shipment to customer process. In the PAC simulation output, the shipment to customer process is characterized by an average, minimum and maximum value of delay in meeting customer demand, the total number of units delivered to the customer during the simulation run, and an overall product delay to customer. The delay to customer, that is the waiting time of a customer for a product, is measured from the time of issuing a customer's order tag (in case of delay costing option I - DCI) or a requisition tag (in case of delay costing option II - DCII) until the time the required product is delivered.

In our experiments, we vary values of the chosen parameters and register the value of the overall product delay to customer. We also examine how the total cost varies. We adjusted the PAC simulation model to facilitate several replications of a single simulation run, and for testing each group of parameters z_j, k_j, r_j, τ_j and S_j (j=1,...,n) with a number of different values during one long run. We test each parameter (z_j, k_j, r_j, τ_j, S_j) with 6 different values l (l=1,...,6), and we execute 10 replications of a single simulation for each of the 6 different values of each parameter. For each $z_{j,l}$, $k_{j,l}$, $r_{j,l}$, $\tau_{j,l}$ and $S_{j,l}$, we calculate the overall average delay to customer and the total cost, as the average of the 10 single simulation runs, and also we calculate the 95 percent confidence interval for the mean values of delay and cost.

Buzacott and Shanthikumar state the properties using expressions "decreasing" or "increasing". For the results presented here, the formulation of the properties in such a manner appears too strong. We replace "decreasing" with "non-increasing" and "increasing" with "non-decreasing" to formulate the properties in a more general way.

We chose models 2 and 4 for validation of the PAC properties. Model 2 is a simple flow in series configuration and model 4 is a more complex manufacturing layout. Figure 8.1 shows layouts of both tested configurations. Model 2 has one "final" product (product 1), two "assembly" products (product 2 and 3) and one "raw material" (product 4). Final and assembly products are characterized by four main PAC parameters (z, k, r, τ) and processing time S, that is, there is a total of 15 parameters to be examined in the case of model 2. Model 4 has 7 final and assembly products (n=7), which result in 35 parameters to be studied.

For both models, we choose the base setting of the parameters to be that of the PTO policy (z=0, k=∞, r=1, τ=0). When we change the value of the tested parameter, all others remain set as for PTO. The large number M, equivalent to setting the process tags to an unlimited quantity, is equal 100 and 220 for model 2 and 4 respectively (refer to 7.2.6). We model all processing times in cells as exponentially distributed. In the case of model 2, we set all mean processing times to 20 minutes, and all other input data are given as for the variant A of model 2 (refer to Table 7.3). Model 4 has the mean processing times as used for model 4A (refer to Table 7.7) and the processing on the machines is with no setups.



Model 2　　Model 4

Figure 8.1 Layout of models 2 and 4.

We choose to get the same number of observations for both manufacturing systems. In the case of model 4, the mean customer arrival rate for all 4 final products is 1 every 3 hours. Thus, it is 3 times lower than in the case of model 2. Therefore, we selected a longer time for a single simulation run for model 4, to consider approximately the same number of shipment delays for both layouts. In the case of model 2, all single simulation runs have a total of 600 days of 24 working hours each, a warm-up time of 40 days plus an evaluation period of 560 days. In the case of model 4, all single simulation runs have a total of 2000 days of 24 working hours each, a warm-up time of 40 days plus an evaluation period of 1960 days.

8.3 Properties of the Shipment to Customer Process

8.3.1 Property 1

Property 1: Shipment delays to customer are non-increasing in $z_j, j=1,...,n$.

First we perform a simulation test run for model 2 which has three cells in series. The results of the overall average delay to customer and the total cost are given in Table 8.1. For each product j (j=1,2,3), we executed 10 replications for each of 6 different values of parameter z_j. For each run we set all z_j to 0, except the z_j tested, which in turn takes values from 1 to 6. We set the remaining PAC parameters to those of the PTO policy. We indicate the product type by "(*f*)" as final, by "(*a*)" as assembly and by "(*fa*)" as final/assembly. By "+/-", we indicate the 1/2 width range values of 95 percent confidence-interval endpoints of the delay and the total cost respectively. That means, we can obtain a 95 percent confidence interval for "Delay μ" (mean value of the overall delay in minutes) and "Cost μ" (mean value of the total cost in $) by subtracting and adding "+/-" to the μ.

Table 8.1 Results of examination of property 1 for model 2

l		1	2	3	4	5	6
$z_{1.l}$	Delay μ	46.9	↓ 23.7	↓ 10.9	↓ 4.7	↓ 2.0	↓ .7
	+/-	.8	.4	.4	.4	.2	.1
(f)	Cost μ	5941.9	⇓ 5030.5	⇑ 5500.9	⇑ 6901.0	⇑ 8791.7	⇑ 10828.0
	+/-	121.5	56.4	43.0	33.7	27.1	17.6
$z_{2.l}$	Delay μ	55.3	↓ 39.5	↓ 33.5	↓ 31.3	↔ 30.4	↔ 30.1
	+/-	.5	.5	.5	.4	.4	.3
(a)	Cost μ	6634.9	⇓ 5879.3	⇑ 6295.1	⇑ 7142.8	⇑ 8183.0	⇑ 9256.8
	+/-	73.5	82.9	68.9	46.7	71.3	42.6
$z_{3.l}$	Delay μ	68.3	↓ 62.0	↔ 61.2	↔ 60.1	↔ 59.9	↔ 60.4
	+/-	.8	.5	.7	.5	.7	.5
(a)	Cost μ	7850.9	⇓ 7656.6	⇑ 8195.8	⇑ 8631.2	⇑ 9143.8	⇑ 9801.9
	+/-	123.0	63.1	114.4	71.1	105.3	70.6

CPU of the long run: 329.9

We indicate, by "↑" and "↓", increasing and decreasing respectively of the overall delay between two consecutive $z_{j.l}$. The value of the overall delay for $z_{j.l}$ is increasing or decreasing if the range of its 95% confidence interval is not overlapping with the range of the 95% confidence interval of the overall delay for $z_{j.l-1}$ and the mean value of delay for $z_{j.l}$ is larger or smaller, respectively, than the mean value of delay for $z_{j.l-1}$. We assume that the consecutive $z_{j.l}$ are essentially equal to each other, if the ranges of their confidence intervals are overlapping. This we indicate by "↔". In a similar way, we state increasing, decreasing and equality of the costs for two consecutive $z_{j.l}$. Here, we use "⇑", "⇓" and "⇔" to indicate increasing, decreasing and equality, respectively.

For all z_j and for the values tested from 1 to 6, the average delay time of shipment of product to customer is always non-increasing in z_j. However, the total cost does not follow this trend. It is characteristic for this model, that for all $z_{j.2}$, the costs are decreasing and then they increase for all remaining $z_{j.l}$. As expected, the property 1 for cells in series holds for our experiment. Note how increased z_j values for products which are relatively closer to a customer (j=1) result in more reduction in the overall average delay to customer than the products several stages removed.

Next we perform similar test runs for model 4. The results of these runs are given in Table 8.2. For each product j (j=1,...,7), we executed 10 replications for each of 6 different values of parameter z_j. For each run we set all z_j to 0, except the tested z_j, which in turn takes values from 1 to 6. We set the remaining PAC parameters to those of the PTO policy.

The results for model 4 show that the property 1 is not contradicted by these experiments.

For product types f (product 1, 2 and 3), the costs are increasing with z_j. For typical assembly products (products 5, 6 and 7), the costs are increasing except the case when $z_{7,3}$ equals 3. It can be explained by the specifics of this production process. Product 5 is made from three items of product 7 and two items of raw material 9 (refer to Figure 8.1). Product 5 is (relatively) in large demand, as it is used to make final products 1, 2 and 3, a total of 90% of customer orders. Product 7, required in the quantity of three items per each item of 5, is even in larger demand than product 5 itself. An initial inventory z_7=3, equal to the required number of items of product 7 to make product 5, has decreasing effect on the total cost, as it significantly decreases waiting time for product 7 and improves the total system performance to such a degree that it compensates the higher storage costs of product 7.

Product 4 is a final/assembly product type; that means, it is delivered to customer and it is also used to produce final product 1. The results for product 4 show neither increase nor decrease of the overall delay to customer; however the costs are increasing with larger value of z_4. Again, we try to explain these results by a careful examination of the specifics of this production process. Only 10% of a total customer demand is for product 4, which is processed from one item of raw material 8 (relatively simple and quick process). Product 4 is used to produce final product 1, which is characterized by 50% of the total demand. However, increasing inventory z_4 does not influence the average delay for product 1 because product 1 requires also one item of product 5 and z_5=0.

Table 8.2 Results of examination of property 1 for model 4

l		1	2	3	4	5	6
$z_{1.1}$	Delay μ	230.7	↓ 207.3	↓ 197.5	↔ 197.1	↔ 195.5	↔ 197.8
	+/-	2.7	3.0	1.7	2.3	1.6	2.5
(f)	Cost μ	73174.2	⇑ 98615.8	⇑ 126835.9	⇑ 157678.7	⇑ 188731.7	⇑ 221194.2
	+/-	904.9	1175.7	683.3	765.0	704.5	1274.9
$z_{2.1}$	Delay μ	259.1	↓ 245.4	↓ 241.3	↔ 241.8	↔ 242.3	↔ 242.1
	+/-	3.0	2.4	1.5	2.0	1.6	2.9
(f)	Cost μ	114467.6	⇑ 186173.1	⇑ 262605.6	⇑ 341290.9	⇑ 419589.0	⇑ 498386.4
	+/-	897.4	596.4	496.9	445.4	533.4	612.9
$z_{3.1}$	Delay μ	238.6	↓ 210.4	↓ 201.6	↓ 198.0	↔ 198.8	↔ 199.6
	+/-	2.0	2.6	2.0	1.0	2.0	2.5
(f)	Cost μ	83380.8	⇑ 114964.9	⇑ 151614.5	⇑ 189768.2	⇑ 229033.2	⇑ 269123.5
	+/-	354.7	1026.1	828.4	540.9	983.4	999.4
$z_{4.1}$	Delay μ	335.2	↔ 337.9	↔ 338.7	↔ 340.7	↔ 337.5	↔ 339.4
	+/-	3.7	3.2	5.0	3.5	5.5	5.1
(fa)	Cost μ	74911.6	⇑ 81283.0	⇑ 86999.9	⇑ 93502.7	⇑ 98591.1	⇑ 105291.5
	+/-	1179.9	944.4	1526.2	1047.9	1862.2	1629.7
$z_{5.1}$	Delay μ	234.8	↓ 194.3	↓ 168.8	↓ 154.6	↓ 145.5	↓ 139.9
	+/-	4.2	4.0	2.8	1.5	1.8	1.7
(a)	Cost μ	64128.6	⇑ 73691.4	⇑ 86910.8	⇑ 104698.8	⇑ 124658.7	⇑ 146070.3
	+/-	1189.5	1255.3	706.6	405.3	492.1	318.0
$z_{6.1}$	Delay μ	315.7	↓ 297.5	↔ 298.3	↔ 295.3	↔ 295.0	↔ 295.7
	+/-	2.5	4.1	3.1	4.0	2.3	2.8
(a)	Cost μ	72890.0	⇑ 75496.3	⇑ 81261.7	⇑ 86881.8	⇑ 92433.3	⇑ 99171.3
	+/-	717.5	1292.8	975.9	1151.2	1298.4	1088.1
$z_{7.1}$	Delay μ	292.7	↓ 269.9	↓ 236.4	↓ 226.7	↓ 214.9	↓ 205.1
	+/-	3.5	2.2	3.9	3.1	3.3	2.1
(a)	Cost μ	64616.8	⇔ 63484.2	⇓ 59641.6	⇑ 62096.0	⇑ 64185.4	⇑ 66893.5
	+/-	1336.3	600.1	1112.2	1038.1	938.1	568.8

CPU of the long run: 6338.3

Figure 8.2 shows a graphic representation of the delay time for different z_j parameters for model 4 using the mean values. Parameters z_5 and z_7 show the largest impact on the decrease of the delay, as products 5 and 7 are important components in this more complex assembly process. This is not consistent with model 2, that is cells in series system, where products relatively closer to a customer had the largest impact on the decrease of the delay.

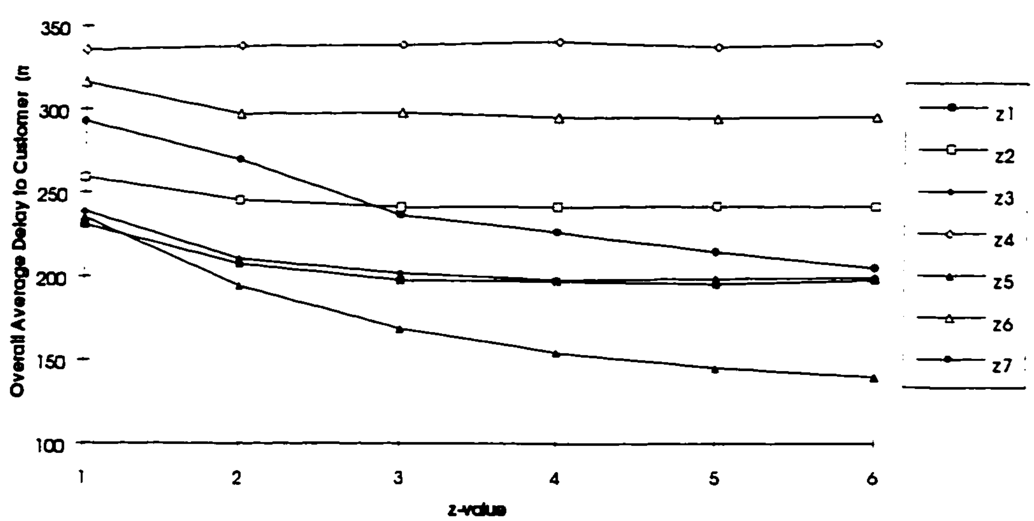


Figure 8.2 Illustration of impact of z parameter in the case of model 4.

We can conclude that property 1 is not violated by the experiments performed for both manufacturing configurations.

8.3.2 Property 2

Property 2: Shipment delays to customer are non-increasing in k_j, j=1,...,n.

Again, we perform several test runs for model 2, which has cells in series. The results of the overall average delay to customer and the total cost are given in Table 8.3. For each product j (j=1,2,3), we executed 10 replications for each of 6 different values of parameter k_j. For each run we set all k_j to 100 (value of M for model 2), except the k_j tested, which in turn takes values from 1 to 6. We set the remaining PAC parameters to those of the PTO policy.

The number of process tags for the last product (product 3) does not influence the system, as raw material 4 is always available. The experiments in Table 8.3 are consistent with Buzacott and Shanthikumar's property 2. For this specific case of model 2 (particular cost factors), the property also holds for the total cost.

Table 8.3 Results of examination of property 2 for model 2

l		1	2	3	4	5	6
$k_{1.1}$	Delay μ	2885.3	↓ 118.8	↓ 96.2	↓ 91.5	↔ 89.8	↔ 90.2
	+/-	1087.7	2.1	.8	.7	1.0	.8
(*f*)	Cost μ	283191.2	⇓ 12280.5	⇓ 10117.5	⇓ 9753.6	⇓ 9527.1	⇔ 9658.7
	+/-	106358.0	273.3	115.3	90.3	151.0	95.8
$k_{2.1}$	Delay μ	127.9	↓ 93.6	↓ 90.9	↔ 90.2	↔ 89.9	↔ 90.0
	+/-	1.9	1.0	.6	.6	.9	.2
(*a*)	Cost μ	13076.4	⇓ 9865.8	⇔ 9676.5	⇔ 9594.9	⇔ 9659.9	⇔ 9629.3
	+/-	204.9	176.6	116.3	128.3	170.1	44.5
$k_{3.1}$	Delay μ	90.3	↔ 90.0	↔ 90.2	↔ 90.0	↔ 89.9	↔ 90.1
	+/-	.7	.8	.8	.8	.9	.7
(*a*)	Cost μ	9636.1	⇔ 9573.9	⇔ 9691.5	⇔ 9662.5	⇔ 9622.2	⇔ 9699.9
	+/-	113.0	90.1	142.1	103.1	170.5	116.1

CPU of the long run: 331.4

Table 8.4 and Figure 8.3 present the results of checking the impact of k_j parameter on the shipment to customer process in the case of model 4. For each product j (j=1,...,7), we executed 10 replications for each of 6 different values of parameter k_j. For each run we set all k_j to 220 (value of M for model 4), except the k_j tested, which varies from 1 to 6. We set the remaining PAC parameters to those of the PTO policy.

Table 8.4 Results of examination of property 2 for model 4

l		1	2	3	4	5	6
$k_{1.1}$	Delay μ	355.8	↓ 321.3	↔ 318.4	↔ 316.3	↔ 316.4	↔ 318.8
	+/-	3.5	3.1	2.3	3.0	2.1	3.1
(f)	Cost μ	67951.7	⇔ 67152.8	⇔ 66854.5	⇔ 66568.7	⇔ 66377.5	⇔ 67450.1
	+/-	1033.4	1186.0	877.9	1300.9	552.9	1248.8
$k_{2.1}$	Delay μ	322.3	↓ 316.5	↔ 315.6	↔ 317.2	↔ 315.3	↔ 317.2
	+/-	3.1	2.1	2.4	2.8	2.7	3.4
(f)	Cost μ	63933.0	⇑ 65958.4	⇔ 66129.8	⇔ 66870.3	⇔ 66367.4	⇔ 66904.0
	+/-	810.0	706.5	575.7	1077.2	666.2	1233.3
$k_{3.1}$	Delay μ	360.1	↓ 319.5	↔ 315.4	↔ 313.4	↔ 315.6	↔ 316.0
	+/-	3.9	4.1	3.3	1.4	3.5	4.1
(f)	Cost μ	65287.1	⇔ 65628.3	⇔ 65804.3	⇔ 65658.4	⇔ 66184.3	⇔ 67074.4
	+/-	462.5	1401.4	1110.0	472.2	1422.1	1637.1
$k_{4.1}$	Delay μ	316.1	↔ 316.5	↔ 315.3	↔ 318.8	↔ 317.5	↔ 317.5
	+/-	2.6	3.0	2.1	3.6	3.7	3.8
(fa)	Cost μ	66617.6	⇔ 66939.3	⇔ 66293.4	⇔ 67182.4	⇔ 66870.0	⇔ 67353.4
	+/-	1032.9	1002.5	925.4	1090.7	1608.0	1362.2
$k_{5.1}$	Delay μ	1304.0	↓ 324.6	↓ 311.5	↔ 314.1	↔ 314.5	↔ 315.5
	+/-	183.1	3.6	3.0	2.5	3.1	2.3
(a)	Cost μ	158620.1	⇓ 55875.4	⇑ 58043.4	⇑ 61461.5	⇑ 63603.7	⇑ 65105.0
	+/-	19198.9	1001.2	635.8	710.9	910.6	802.3
$k_{6.1}$	Delay μ	348.9	↓ 326.6	↔ 321.9	↔ 321.4	↔ 318.0	↔ 320.5
	+/-	4.5	7.0	3.1	4.6	3.4	2.6
(a)	Cost μ	66730.6	⇔ 66126.4	⇔ 66044.7	⇔ 67215.5	⇔ 66356.9	⇔ 67746.1
	+/-	996.5	1770.5	799.2	1327.8	1413.4	793.1
$k_{7.1}$	Delay μ	888.1	↓ 336.1	↓ 309.9	↔ 308.0	↔ 308.7	↔ 309.7
	+/-	48.2	2.2	3.5	2.3	2.8	2.9
(a)	Cost μ	148096.2	⇓ 58189.5	⇓ 54795.1	⇔ 56003.3	⇑ 57580.3	⇑ 59335.1
	+/-	8511.8	445.2	748.3	668.2	801.1	749.3

CPU of the long run: 8672.0

The results indicate that property 2 is not violated for model 4. Note how only small values of the *k* parameter (1, 2 or 3) decrease the overall delay. The total costs do not show much change in the case of final products. For assembly products, first, the costs tend to decrease and then increase. The results of the delay and the costs for product 4 (final/assembly) show neither increase nor decrease. Again, we can look for some explanation of this fact in the production process itself. Product 4 has a low demand as a final product (only 10%) and has a relatively simple and quick production process, as it is made from one item of raw material 8. The number of process tags for product 4 does not influence the production at cell 2, as raw material 8 is always available.

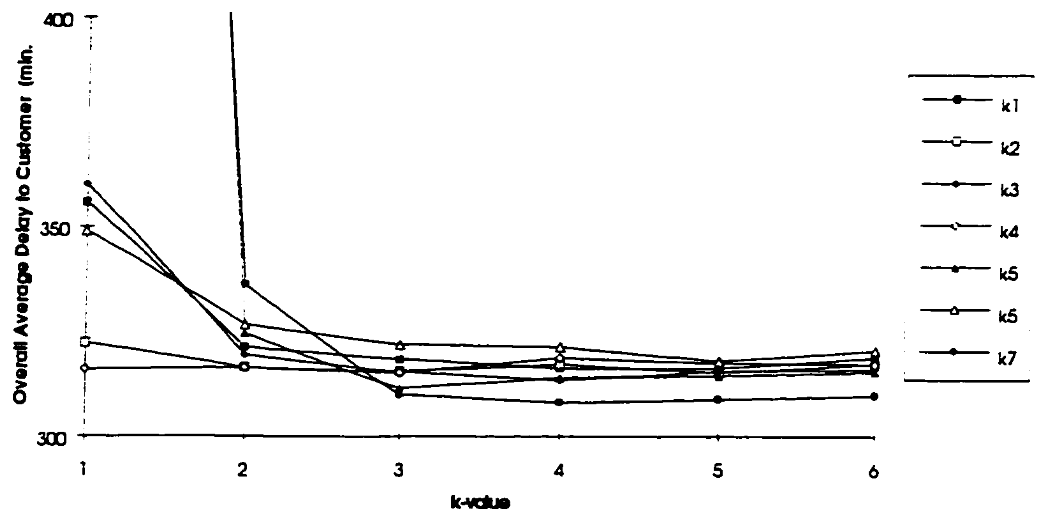


Figure 8.3 Illustration of impact of k parameter in the case of model 4.

After examination of property 2 in the case of our two manufacturing configuration, we can observe that the property appears to hold for both of them.

8.3.3 Property 3

Property 3: Shipment delays to customer are non-decreasing in r_j, j=1,...,n.

We executed several simulation test runs for model 2. Table 8.5 shows obtained results of the overall delay and the total cost. For each product j (j=1,2,3), we executed 10 replications for each of 6 different values of parameter r_j. For each run we set all r_j to 1, except the r_j tested, which in turn takes values from 2 to 7. We set the remaining PAC parameters to those of the PTO policy.

Property 3 again is demonstrated for this case of cells in series. Note how the changes in the value of parameter r have the same impact regardless of the product. For the specific cost factors in the case of model 2, the increase of r_j always results in an increase of the total cost.

Table 8.5 Results of examination of property 3 for model 2

l	2	3	4	5	6	7
$r_{1.1}$ Delay μ	129.2	↑ 170.5	↑ 213.3	↑ 256.1	↑ 300.1	↑ 342.4
+/-	.7	.7	.7	1.1	1.2	1.3
(f) Cost μ	13520.1	⇑ 18105.3	⇑ 22444.9	⇑ 26971.4	⇑ 31284.2	⇑ 35903.5
+/-	185.2	96.1	110.6	148.8	189.2	113.2
$r_{2.1}$ Delay μ	129.0	↑ 170.4	↑ 212.8	↑ 256.4	↑ 298.3	↑ 342.2
+/-	.4	.6	.6	1.5	1.6	1.7
(a) Cost μ	13723.1	⇑ 17990.8	⇑ 22399.0	⇑ 26794.2	⇑ 31373.5	⇑ 35798.8
+/-	71.5	157.9	120.8	226.0	239.8	92.8
$r_{3.1}$ Delay μ	129.3	↑ 171.3	↑ 213.0	↑ 255.3	↑ 299.0	↑ 341.4
+/-	.6	1.1	1.2	.8	1.3	1.3
(a) Cost μ	13772.8	⇑ 17991.8	⇑ 22460.0	⇑ 26854.3	⇑ 31308.5	⇑ 35943.2
+/-	107.6	118.9	179.1	114.1	162.4	236.1

CPU of the long run: 393.0

Table 8.6 show results of an examination of property 3 in the case of model 4. For each product j (j=1,...,7), we executed 10 replications for each of 6 different values of parameter r_j. For each simulation run we set all r_j to 1, except the r_j tested, which varies from 2 to 7. We set the remaining PAC parameters to those of the PTO policy. Because the results showed some interesting patterns in the case of products 6 and 7, we performed additional test runs with values of parameter r_j from 7 to 12. Figure 8.4 gives a graphical representation of the impact of parameter r on the shipment delay for all the values of r_j tested (from 2 to 12).

Table 8.6 Results of examination of property 3 for model 4

l	2	3	4	5	6	7
$r_{1,1}$ Delay μ	437.4	↑ 559.5	↑ 676.4	↑ 790.5	↑ 908.9	↑ 1036.6
+/-	3.3	5.2	4.0	3.8	6.2	9.7
(*f*) Cost μ	81662.1	⇑ 99885.0	⇑ 114958.3	⇑ 131723.8	⇑ 146906.6	⇑ 165125.9
+/-	981.4	1720.5	1113.0	1715.1	1260.2	1555.5
$r_{2,1}$ Delay μ	433.2	↑ 551.1	↑ 666.9	↑ 787.9	↑ 911.5	↑ 1044.3
+/-	4.0	4.7	5.4	6.2	9.6	12.7
(*f*) Cost μ	82556.3	⇑ 97596.0	⇑ 113516.0	⇑ 130628.0	⇑ 148449.0	⇑ 166003.6
+/-	1298.1	762.3	1342.5	1332.5	1422.5	2035.3
$r_{3,1}$ Delay μ	429.3	↑ 546.8	↑ 665.0	↑ 793.4	↑ 912.5	↑ 1040.1
+/-	2.4	3.9	5.0	4.8	2.8	6.1
(*f*) Cost μ	77580.1	⇑ 90134.2	⇑ 102136.8	⇑ 116383.9	⇑ 129391.0	⇑ 144819.7
+/-	806.3	1702.7	1072.6	1230.1	1379.6	1783.2
$r_{4,1}$ Delay μ	387.4	↑ 468.9	↑ 556.2	↑ 648.8	↑ 739.5	↑ 839.5
+/-	2.9	3.3	4.9	3.4	4.4	3.5
(*fa*) Cost μ	78106.6	⇑ 93849.0	⇑ 111040.3	⇑ 130671.0	⇑ 149487.0	⇑ 171331.9
+/-	1067.3	972.8	1196.2	1218.9	1472.9	1124.5
$r_{5,1}$ Delay μ	406.9	↑ 485.9	↑ 551.5	↑ 618.9	↑ 683.1	↑ 752.8
+/-	2.8	2.6	5.9	6.3	3.5	5.6
(*a*) Cost μ	84031.8	⇑ 98642.6	⇑ 109034.3	⇑ 120317.1	⇑ 130789.2	⇑ 142691.1
+/-	1000.9	1368.8	1345.0	1092.3	1089.6	1374.8
$r_{6,1}$ Delay μ	315.9	↑ 399.8	↓ 383.6	↑ 484.8	↓ 467.3	↑ 578.0
+/-	2.6	3.7	2.2	3.3	3.7	4.5
(*a*) Cost μ	66560.8	⇑ 79513.1	⇓ 75368.5	⇑ 94012.7	⇓ 88948.8	⇑ 110609.1
+/-	767.0	1244.6	511.1	1189.3	1400.3	842.2
$r_{7,1}$ Delay μ	369.1	↓ 315.4	↑ 408.1	↑ 437.9	↓ 405.8	↑ 476.8
+/-	1.7	2.3	3.2	1.8	3.6	3.4
(*a*) Cost μ	80372.6	⇓ 66095.9	⇑ 89264.4	⇑ 96677.9	⇓ 85603.9	⇑ 105596.1
+/-	1055.5	765.6	1203.7	1132.2	1337.0	1002.5

CPU of the long run: 7842.4

The property holds for all products, except two assembly products, products 6 and 7. Again, we examined this production process in a search for some explanation. It is interesting to see the relation between the required number of items of products for processing and the value of parameter *r* in case of these two products. Two items of product 6 are required to make product 3, and three items of product 7 are used for processing of product 5. In the case of products 6 and 7, if the parameter r is equal to the bill of material amount (here the BOM amount equals 2 and 3 respectively) or its multiple (4,6,8,10,12 and 3,6,9,12 respectively), it reduces both the overall delay and the cost. Product 5, which is also an assembly type of product does not show this characteristics, as it is a component of three different products with different BOM amounts (1, 3 and 1). Final/assembly product 4 is in large number used to make the final product 1. Product 1 requires only one item of product 4 (BOM amount equals 1) and therefore product 4 increases the delay and costs with larger value of r_4.

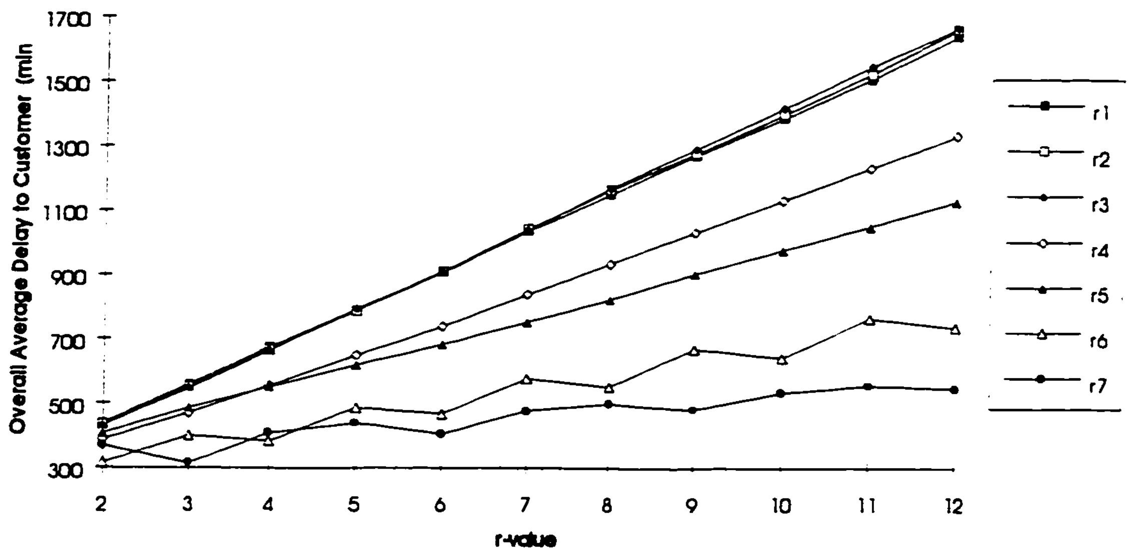


Figure 8.4 Illustration of impact of r parameter in the case of model 4.

For all tested values of parameter r, the costs follow the same trend that the delay. That means that they are increasing when the delay increases and they are decreasing when the delay is decreasing.

Also, it appears that enlarging the PA batch size for products "closer" to customers have more impact on the delay (refer to Figure 8.4). This is not consistent with model 2, where the impact on the delay is the same regardless of the product. Additionally, we can observe that products with a larger number of items in the BOM have smaller impact on the delay. We can try to explain this phenomenon by a careful examination of the two systems. Accumulating PA cards to batch sizes more than 1, impedes the passing of information about customer demand to the upstream cells. For both models, final products are always triggered by a constant customer demand of one unit. In the case of model 2, all assembly products are also triggered by a demand of one unit, which results in the same impact of the PA batch size on delay regardless of the product. In the case of model 4, assembly products are triggered by final or other assembly products with a demand of more than one unit, which lowers the impact of batching of PA cards on the delay.

When discussing the PAC properties, Buzacott and Shanthikumar did not consider issues of BOM, assembly operations and products, which have both characteristics, final and assembly. Model 4 is an example of a manufacturing configuration, wherein all of those aspects are represented. We can observe that property 3 is contradicted for a more complex manufacturing configuration for assembly or final/assembly products if a number of items in the BOM for these products is equal to the PA cards batch or its multiplication, and the specifics of the process favor it. In the case of model 4, final products do not violate property 3.

8.3.4 Property 4

Property 4: Shipment delays to customer are non-decreasing and convex in τ_j, $j=a,...,n$, but non-increasing and convex in τ_j, $j=1,...,f$, when $c_i=1$, $i=1,...,m$.

The results of the examination of property 4 for model 2 are given in Table 8.7 and graphically presented in Figure 8.5. Again, for each product j (j=1,2,3), we did perform 10 replications for each of 6 different values of parameter τ_j. For each run we set all τ_j to 0, except the τ_j tested, which in turn takes values from 10 to 160. We set the remaining PAC parameters to those of the PTO policy.

The DCII cost calculation approach is used when an influence of a parameter τ on the system performance is studied. This cost calculation procedure is used for coordination policies, which allows for $\tau \geq 0$ for "final" product types. We assume here, that the system management generates order tags and customers issue the requisition tags. The delay cost in meeting the customer demand in the time lag between issuing of an order tag and a requisition tag is calculated by applying a relatively "small" cost factor, designed to keep

Table 8.7 Results of examination of property 4 for model 2

l		10	40	70	100	130	160
$\tau_{1.1}$	Delay μ	79.3	↓ 52.2	↓ 30.5	↓ 16.2	↓ 8.2	↓ 4.0
	+/-	.8	.5	.5	.5	.3	.2
(*f*)	Cost μ	8510.9	⇓ 6184.6	⇓ 4438.9	⇓ 3765.1	⇑ 3906.5	⇑ 4610.2
	+/-	154.7	75.5	78.1	67.6	43.9	22.7
$\tau_{2.1}$	Delay μ	90.3	↑ 95.5	↑ 111.7	↑ 135.0	↑ 161.9	↑ 190.7
	+/-	.5	.5	.6	.4	.7	.4
(*a*)	Cost μ	9723.6	⇑ 10303.0	⇑ 12214.3	⇑ 14907.4	⇑ 18215.9	⇑ 21555.2
	+/-	73.4	132.0	115.7	159.4	218.2	79.3
$\tau_{3.1}$	Delay μ	91.8	↑ 107.3	↑ 132.7	↑ 161.0	↑ 190.1	↑ 220.3
	+/-	.8	.7	.7	.6	.7	.5
(*a*)	Cost μ	9902.3	⇑ 11466.4	⇑ 14351.7	⇑ 17437.8	⇑ 20539.0	⇑ 23990.6
	+/-	130.5	99.3	156.9	108.9	225.6	198.1

CPU of the long run: 350.3

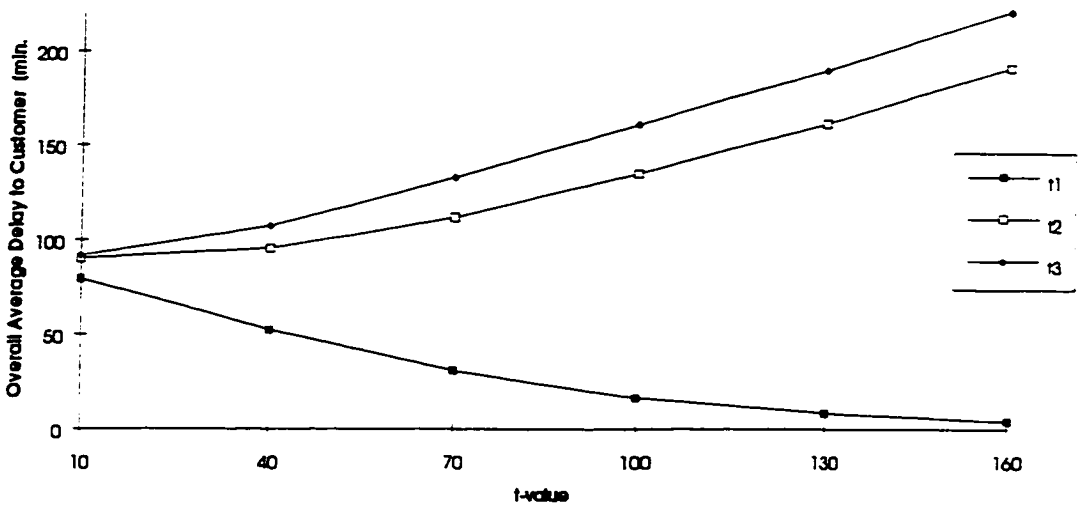


Figure 8.5 Illustration of impact of τ parameter in the case of model 2.

requisition tag and delivering the product to the customer. Splitting the costs in such a lead times from becoming unrealistically long. A relatively "large" cost factor is applied to calculate the delay costs in the time lag elapsed between the time from the receiving of themanner allows for implementing of forecasting to the expected customer demand and results in $\tau \geq 0$ for "final" product types. Property 4 differentiates the impact of the τ parameter to the strictly final and assembly products. Larger τ of a final product postpones an arrival of a requisition tag from customer and therefore can decrease the delay. Larger τ of an assembly product postpones an arrival of the product to the cell and can increase the delay.

As we can see, the system that we have tested does not contradict property 4. For a final product, the costs first decrease and then tend to increase, and for assembly products, the costs follow the same trend as the shipment delays for this particular case of model 2.

Similar results for model 4 are presented in Table 8.8 and Figure 8.6. For each product j (j=1,...,7), we executed 10 replications for each of 6 different values of parameter τ_j. For each simulation run we set all τ_j to 0, except the τ_j tested, which varies from 10 to 310. We set the remaining PAC parameters to those of the PTO policy. Here, it was necessary to enlarge the interval between different l values, to be able to observe the differences in delays.

For cases of final types of products, the delays are decreasing in τ_j and they seem to be convex. For assembly product types the delays are increasing in τ_j, and except product 5 (for τ_j<70) they seem to be convex. Product 5 is in large demand for production of all three final products, and this can explain this relative large increase in the shipment delay for those values of parameter τ. Product 4, which has both characteristics, that is final and assembly, tends to behave as an assembly product. This is not surprising as approximately 83% of the total production of product 4 is used to make product 1 (the assembly function dominates). We can expect that a final/assembly product with more emphasis on a "final" function will behave as final products.

Generally, we can conclude that the monotonicity property 4 holds for both models. However, the convexity can be violated in some special cases of assembly products. For this particular scenario of model 4, the costs appear to follow the same trend as the delays.

Table 8.8 Results of examination of property 4 for model 4

l		10	70	130	190	250	310
$\tau_{1,1}$	Delay μ	310.0	↓ 281.0	↓ 252.0	↓ 229.6	↓ 219.3	↓ 208.1
	+/-	2.9	2.3	2.1	2.8	3.7	2.0
(f)	Cost μ	65223.6	⇓ 62759.6	⇓ 59688.5	⇔ 58509.8	⇔ 61840.3	⇔ 63527.2
	+/-	961.0	833.4	683.2	935.8	1411.3	759.1
$\tau_{2,1}$	Delay μ	317.1	↓ 307.8	↓ 303.2	↔ 298.3	↓ 290.4	↔ 287.9
	+/-	3.2	1.8	2.5	2.9	2.1	3.1
(f)	Cost μ	67361.7	⇔ 65522.0	⇔ 65452.9	⇔ 65276.2	⇔ 64008.8	⇔ 64820.6
	+/-	1202.7	770.5	684.1	1095.7	655.5	1198.4
$\tau_{3,1}$	Delay μ	315.6	↓ 293.6	↓ 276.8	↓ 262.7	↓ 247.6	↓ 234.3
	+/-	2.2	3.2	3.5	3.2	2.6	1.8
(f)	Cost μ	66770.3	⇓ 64813.9	⇔ 63827.0	⇔ 64288.9	⇔ 63576.9	⇔ 63807.9
	+/-	533.1	1185.2	1439.7	1203.1	910.0	749.8
$\tau_{4,1}$	Delay μ	344.8	↔ 339.7	↔ 340.5	↑ 349.5	↑ 361.7	↑ 383.0
	+/-	4.4	3.9	4.9	4.0	3.1	3.5
(fa)	Cost μ	70496.5	⇔ 69847.4	⇔ 70383.1	⇔ 73269.9	⇑ 76640.8	⇑ 83088.9
	+/-	1335.1	1332.0	1704.9	1278.6	875.4	1100.5
$\tau_{5,1}$	Delay μ	318.9	↑ 341.2	↔ 346.5	↑ 359.0	↑ 379.4	↑ 405.3
	+/-	3.5	4.9	3.6	1.7	4.9	3.5
(a)	Cost μ	67260.0	⇔ 69611.2	⇔ 71030.8	⇑ 74015.2	⇑ 80101.0	⇑ 87407.5
	+/-	1087.5	1349.5	1231.4	569.2	1679.1	1154.3
$\tau_{6,1}$	Delay μ	317.4	↔ 316.1	↔ 317.6	↔ 316.8	↑ 324.6	↑ 332.9
	+/-	2.5	3.4	3.9	1.8	3.3	3.8
(a)	Cost μ	66896.1	⇔ 66321.1	⇔ 66452.3	⇔ 66321.1	⇔ 67994.5	⇔ 70520.2
	+/-	770.0	859.8	1642.2	602.0	1485.5	1545.9
$\tau_{7,1}$	Delay μ	315.4	↔ 315.5	↑ 326.4	↑ 348.0	↑ 376.0	↑ 409.8
	+/-	3.3	4.4	3.3	2.5	2.8	3.0
(a)	Cost μ	66036.2	⇔ 65878.4	⇑ 69115.1	⇑ 75281.6	⇑ 83480.0	⇑ 93350.8
	+/-	1332.4	1355.1	1179.0	972.4	818.0	1146.7

CPU of the long run: 6417.5

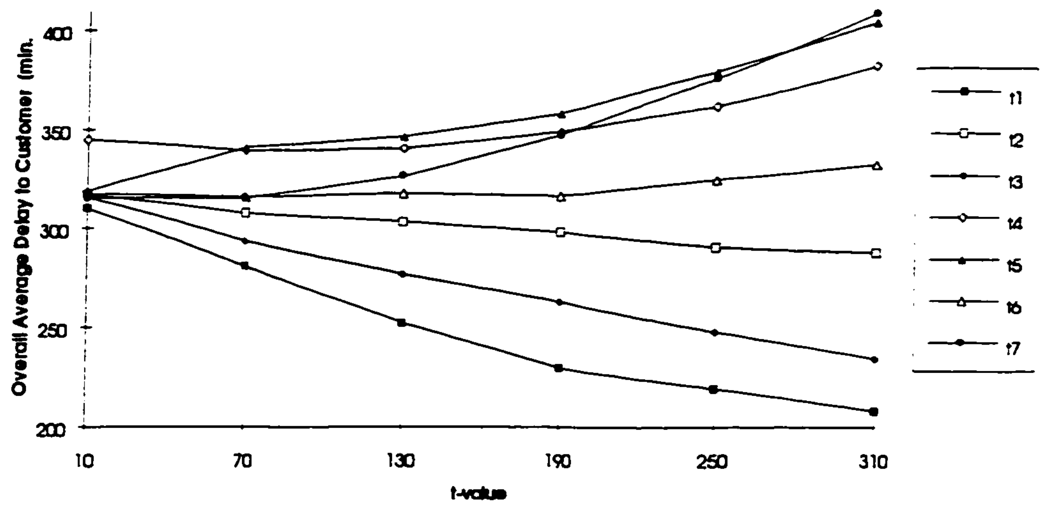


Figure 8.6 Illustration of impact of τ parameter in the case of model 4.

8.3.5 Property 5

Shipment delays to customer are non-decreasing and convex in processing times S_j, j=1,...,n, when c_i=1, i=1,...,m.

Table 8.9 and Figure 8.7 summarize the test results of property 5 for model 2. For each product j (j=1,2,3), we executed 10 replications for each of 6 different values of mean processing time S_j. For each simulation run we set all S_j to 20, except for the S_j tested, which in turn takes values from 15 to 40. All PAC parameters have a setting of the PTO policy.

The property holds for this model of cells in series. For specific cost factors in the case of model 2, the increase of S_j always results in an increase of total costs. Here also we observe that changes in processing times have the same impact, regardless of the machine.

Table 8.9 Results of examination of property 5 for model 2

l	15	20	25	30	35	40
$S_{1.1}$ Delay μ	79.3	↑ 90.4	↑ 103.1	↑ 120.2	↑ 142.1	↑ 180.2
+/-	.7	.5	1.0	1.7	1.7	2.1
(*f*) Cost μ	8276.2	⇑ 9759.6	⇑ 11264.4	⇑ 13453.3	⇑ 16146.0	⇑ 21282.3
+/-	150.0	93.7	146.8	236.2	268.6	281.4
$S_{2.1}$ Delay μ	80.2	↑ 89.8	↑ 102.4	↑ 119.6	↑ 143.7	↑ 178.6
+/-	.5	.8	1.1	1.3	2.9	2.8
(*a*) Cost μ	8609.9	⇑ 9637.5	⇑ 11029.4	⇑ 12912.3	⇑ 15780.7	⇑ 19758.8
+/-	56.7	157.4	177.6	226.9	440.8	359.6
$S_{3.1}$ Delay μ	80.4	↑ 90.1	↑ 103.2	↑ 120.0	↑ 143.1	↑ 180.0
+/-	.7	.8	1.1	.9	2.5	4.7
(*a*) Cost μ	8699.1	⇑ 9603.4	⇑ 11084.6	⇑ 12853.3	⇑ 15287.6	⇑ 19383.0
+/-	112.9	90.9	190.7	122.0	379.6	567.5

CPU of the long run: 348.6

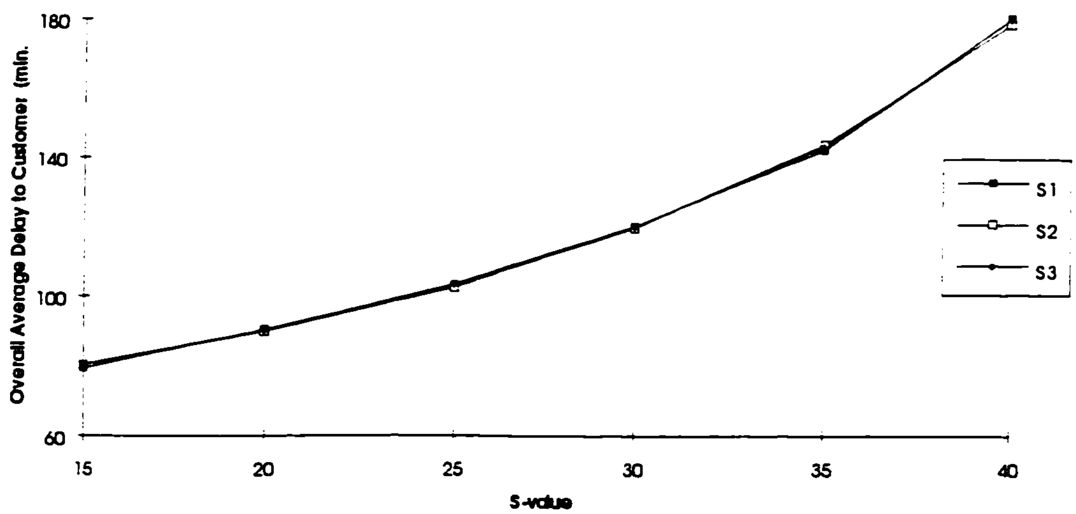


Figure 8.7 Illustration of impact of processing times *S* in the case of model 2.

Table 8.10 and Figure 8.8 present similar test results of the examination of property 5 for model 4. For each *j* (j=1,...,7), we executed 10 replications for each of 6 different values of mean processing time S_j. For each run we set all S_j as for model 4A (refer to Table 7.7), except the S_j tested, which in turn takes values from 20 to 70. All PAC parameters have a setting of the PTO policy. Again, it was necessary to enlarge the interval between different l values, to be able to observe the differences in delays.

Table 8.10 Results of examination of property 5 for model 4

l		20	30	40	50	60	70
$S_{1.1}$	Delay μ	306.5	↔ 310.0	↑ 317.0	↔ 319.8	↑ 330.5	↔ 335.2
	+/-	4.0	3.2	3.5	3.1	4.1	2.9
(f)	Cost μ	66179.5	⇔ 65794.2	⇔ 67070.8	⇔ 66402.2	⇑ 69536.6	⇔ 69437.4
	+/-	1505.6	1189.4	1278.1	944.8	1489.6	1150.5
$S_{2.1}$	Delay μ	317.4	↔ 317.7	↔ 319.1	↔ 320.1	↔ 319.7	↔ 324.0
	+/-	1.7	2.7	3.8	2.9	1.9	3.1
(f)	Cost μ	66826.2	⇔ 66811.6	⇔ 67033.9	⇔ 67060.2	⇔ 66444.0	⇔ 67806.8
	+/-	560.8	989.0	1348.5	1125.4	646.7	1241.8
$S_{3.1}$	Delay μ	311.8	↔ 311.3	↑ 317.2	↔ 322.7	↔ 325.5	↔ 326.9
	+/-	3.6	1.4	4.1	3.1	2.6	2.0
(f)	Cost μ	66467.4	⇔ 65560.9	⇔ 67155.3	⇔ 67848.6	⇔ 67624.2	⇔ 67354.3
	+/-	1454.6	468.8	1640.7	1179.4	888.0	824.4
$S_{4.1}$	Delay μ	318.6	↔ 323.7	↔ 327.9	↑ 334.9	↑ 343.9	↑ 356.2
	+/-	2.8	3.9	3.5	3.3	2.6	3.4
(fa)	Cost μ	67399.0	⇔ 67948.7	⇔ 69147.9	⇔ 70226.1	⇔ 71951.1	⇑ 74850.2
	+/-	857.8	1101.7	1363.8	1169.7	714.7	1246.8
$S_{5.1}$	Delay μ	314.2	↑ 325.0	↑ 339.0	↑ 357.9	↑ 379.0	↑ 405.0
	+/-	3.1	2.8	3.2	1.7	5.4	4.2
(a)	Cost μ	66652.4	⇔ 68109.0	⇑ 70958.0	⇑ 74057.2	⇑ 78532.3	⇑ 83837.0
	+/-	1539.6	902.5	1187.1	598.8	1981.2	1413.0
$S_{6.1}$	Delay μ	320.2	↑ 350.6	↑ 391.9	↑ 426.0	↑ 482.3	↑ 565.1
	+/-	4.2	5.1	4.8	5.5	8.5	19.1
(a)	Cost μ	67508.7	⇑ 74450.0	⇑ 84375.0	⇑ 92015.8	⇑ 105614.2	⇑ 127028.7
	+/-	1307.3	1496.0	1540.8	1725.4	2874.4	5817.0
$S_{7.1}$	Delay μ	221.9	↑ 316.8	↑ 557.0	↑ 4120.6	↑ 86658.5	↑ 113172.3
	+/-	.7	2.8	13.5	1772.0	4179.2	19643.7
(a)	Cost μ	41271.5	⇑ 66426.5	⇑ 130763.2	⇑ 989921.4	⇑7135014.5	⇑6346092.5
	+/-	272.1	1082.0	4175.7	373437.5	201862.0	193588.0

CPU of the long run: 6343.2

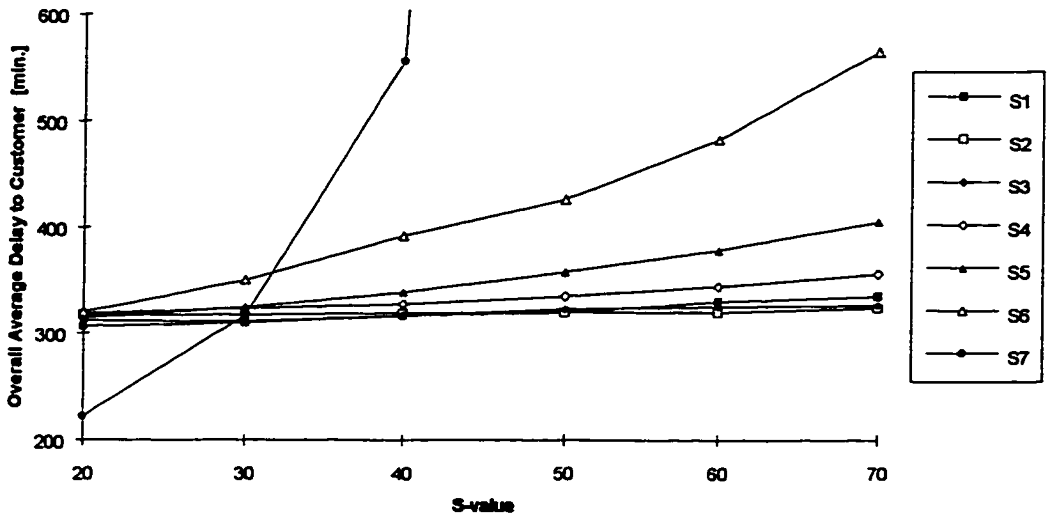


Figure 8.8 Illustration of impact of processing times S in the case of model 4.

The increasing impact on the shipment delays in processing times applies to all products, and tends to increase the costs for this particular scenario of model 4. Additionally, the overall delays seem to be convex for processing times of all types of products. It is interesting to observe how an increase in processing times results in relatively small changes in delays, except for product 7. Product 7 is important in the production process of subsequent products 5, 1, 2, and 3.

We can conclude that the property 5 holds in the case of both manufacturing configurations tested.

8.4 Closing Remarks

We experimentally checked five properties defined by Buzacott and Shanthikumar using two different manufacturing configuration. For cells in series, these were confirmed. They also seemed to apply to more complex manufacturing setting with two exceptions. Property 3 can be violated for assembly and final/assembly product types, if the number of items of these products required in the bill of material (BOM) is equal to their PA card batch size (*r* parameter) and the specifics of the process favor it. For property 4, assembly products show that shipment delays are non-increasing, but they appear to violate the convexity requirement in the case of one product.

The examination of the properties for a more complex layout shows how important are issues of BOM and assembly operations on the impact of the four PAC parameters and the processing times on the shipment delay and the total cost. In the case of parameter *z* the largest impact on delays have products which are in great demand (product 5 and 7) and not the products which are relatively closer to a customer, as it was for the cells in series model (product 1). There is also a relation between the reduction of costs for parameter *z* of product 7, when it is set to 3 (product 5 requires 3 items of product 7). In the case of parameter *k*, only for product 5 and 7 the costs first tend to decrease and then increase, when for all other products of model 4 and all products of model 2, the costs were non-increasing. Examination of the impact of parameter *r* shows very interesting results for products 6 and 7, which are required for processing of other products in a quantity more than 1. By setting *r* parameter of those products to their BOM amount or its multiplication, we observe a repeating pattern of decreasing of the delay and costs. As well, increased values of *r* of products with a BOM amount more than 1 tend to have lower impact on the delay than products required in the quantity of one item. In the case of cells in series the impact of parameter *r* was the same regardless of the product. Property 4 differentiates the impact of the parameter τ of final and assembly products. In the case of product 4, which is an example of a final/assembly product, the impact of τ on the delays depends on the domination of one of those characteristics. Processing times show again that the largest impact on delays have products which are in great demand. Processing times of products in cells in series had the same impact on the delay.

Chapter 9

IMPACT OF PROCESSING TIME VARIABILITY ON THE SYSTEM PERFORMANCE AND DESIGN

9.1 Introduction

An important issue in manufacturing is that of the impact of processing time variability on the system performance and especially the implications for a coordination scheme. We would like to examine how the variability of the processing times changes the system performance. Can we design a system robust enough to deal with the variability of the processing times? How does the performance of various policies depend on the different variability in processing times?

In this chapter, we present some study results on the effect of variability in processing times on the total cost. We restrict simulation and optimization runs to two cases: model 2, which is a simple case with cells in series and model 4, an example of a more complex layout.

Before discussing the results for both models, we give a general description of the experiment.

9.2 Choosing Processing Times Distributions

We perform simulation and optimization runs for 2 different models to compare the performance of these configurations operating under different coordination schemes and with different coefficient of variations for processing times. The processing times at cells are assumed to be distributed from identical shifted Weibull distribution. Using the Weibull has some advantages. The Weibull(1,β) and exponential(β) are the same. The Weibull density function can take on shapes similar to gamma densities, and as gamma, Weibull finds major application to generate time to complete some task. Further, an algorithm for generating random variables from Weibull distribution is fairly simple to implement and is reasonably efficient. Weibull(α,β) distribution has a range $[0,\infty)$; thus it allows the random variable X to take an arbitrarily small positive value. By including a so-called location parameter γ, we can shift the Weibull distribution. The shifted Weibull(α,β,γ) by an amount $\gamma>0$ has density

$$f(x)=\begin{cases}\alpha\beta^{-\alpha}(x-\gamma)^{\alpha-1}e^{-((x-\gamma)/\beta)^{\alpha}} & \text{if } x>\gamma \\ 0 & \text{otherwise}\end{cases} \qquad [9.1]$$

which has the same shape ($\alpha>0$) and scale ($\beta>0$) parameters as the Weibull(α,β) distribution but is shifted γ units to the right. The range of the shifted Weibull distribution is $[\gamma,\infty)$. Often we use shifted distributions to model situations that do not allow for the random variable X to take values less than some fixed positive number. In our case, by appropriate choice of parameters α, β and γ, we can maintain the same mean times of the processing times and change their variability. In the work reported here, we measure variability by the coefficient of variation, $cv=\sqrt{\sigma^2}/\mu$ (with variance σ^2 and mean μ).

The parameters of the shifted Weibull(α,β,γ) distribution were chosen in such a way that the coefficient of variation ranged from approximately 0 to 1 with steps of 0.1, and from approximately 0 to 5 with steps equal to 1.

In the first set of experiments, parameter α was set to 1, so the shifted Weibull(1,β,γ) distribution was the same as the shifted exponential(β,γ) distribution. We created different shifted Weibull(1,β,γ) distributions with μ equal to 10. The relevant statistics are:

Mean (μ) : $\beta+\gamma=10$

Variance(σ^2) : β^2

We want to calculate values of parameters β and γ for different values of *cv*. This is done as follows:

$$cv = \sqrt{\sigma^2} / \mu = \beta / \mu = \beta / 10 \quad \rightarrow \beta = 10\, cv; \qquad [9.2]$$

$$\mu = \beta + \gamma = 10\, cv + \gamma = 10 \quad \rightarrow \gamma = 10(1 - cv) \qquad [9.3]$$

Figure 9.1 shows plots of density functions for 11 different values of parameter β for the distribution Weibull(1,β). Table 9.1 summarizes all parameter values and values of the coefficient of variation for the shifted Weibull(1,β,γ). We used this distribution for testing model 2.

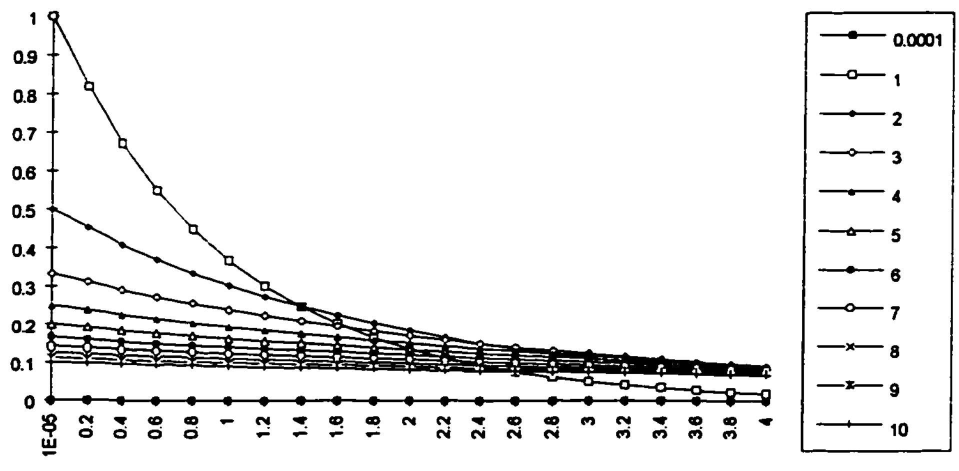


Figure 9.1 Weibull(1,β) density function for 11 different values of β.

Table 9.1 Data of shifted Weibull(1,β,γ) with μ=10

Case	1	2	3	4	5	6	7	8	9	10	11
β	0.0001	1	2	3	4	5	6	7	8	9	10
γ	10	9	8	7	6	5	4	3	2	1	0
cv	≈ 0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0

In the second set of experiments, which we applied to both models, we first create shifted Weibull distributions with μ equal always 1, to facilitate generating processing times with any mean value equal to a constant *a*. Since multiplication of a random variable by a constant value does not change the coefficient of variation. Suppose *X* is shifted Weibull random variable with μ=1 and $cv = \sqrt{\sigma^2} / \mu = \sqrt{\sigma^2}$. Let the random variable *Y* be defined as: *Y* = *a X*, where *a* is a constant representing the mean processing time at a machine. *X* has a $cv = \sigma$; and *Y* has a $cv = \sqrt{a^2 \sigma^2} / a\mu = \sigma$.

The relevant statistics for the random variable *X* are:

Mean (μ) : $\frac{\beta}{\alpha}\Gamma\left(\frac{1}{\alpha}\right) + \gamma = 1$

Variance(σ^2) : $\frac{\beta^2}{\alpha}\left\{2\Gamma\left(\frac{2}{\alpha}\right) - \frac{1}{\alpha}\left[\Gamma\left(\frac{1}{\alpha}\right)\right]^2\right\}$

We want to calculate values of parameters β and γ for different values of *cv*. The is done as follows:

$$cv = \sqrt{\sigma^2} / \mu = \sqrt{\sigma^2} / 1 = \sqrt{\frac{\beta^2}{\alpha}\left\{2\Gamma\left(\frac{2}{\alpha}\right) - \frac{1}{\alpha}\left[\Gamma\left(\frac{1}{\alpha}\right)\right]^2\right\}}$$

$$\rightarrow \beta = cv\sqrt{\frac{\alpha}{\left\{2\Gamma\left(\frac{2}{\alpha}\right) - \frac{1}{\alpha}\left[\Gamma\left(\frac{1}{\alpha}\right)\right]^2\right\}}} \qquad [9.4]$$

$$\gamma = 1 - \frac{\beta}{\alpha}\Gamma\left(\frac{1}{\alpha}\right) = 1 - \frac{cv}{\alpha}\Gamma\left(\frac{1}{\alpha}\right)\sqrt{\frac{\alpha}{\left\{2\Gamma\left(\frac{2}{\alpha}\right) - \frac{1}{\alpha}\left[\Gamma\left(\frac{1}{\alpha}\right)\right]^2\right\}}} \qquad [9.5]$$

By performing a binary search, we obtained required parameter values of this distribution. Parameter α was set to 0.28. We used 6 different Weibull distributions to vary the coefficient of variations when maintaining the mean processing times. Figure 9.2 shows plots of density functions for 6 different Weibull(0.28,β). Figure 9.3 shows plots of density functions for 6 different shifted Weibull(0.28,β,γ).

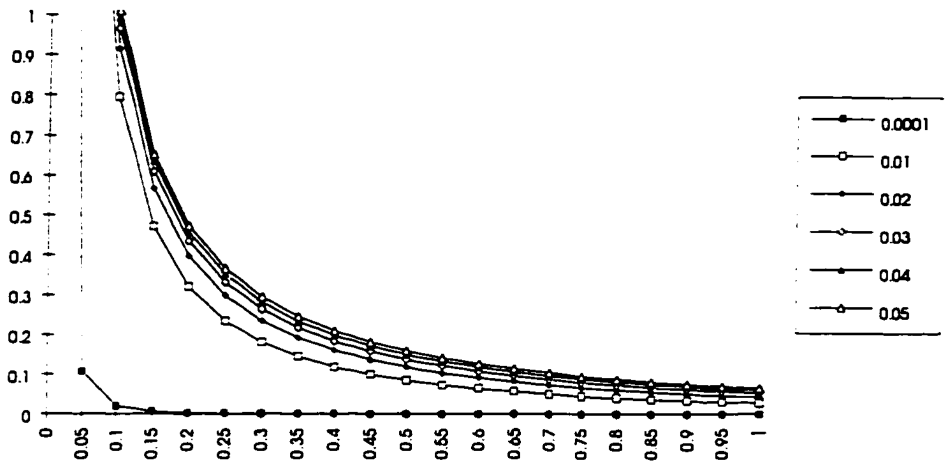


Figure 9.2 Weibull(0.28,β) density function for 6 different values of β.

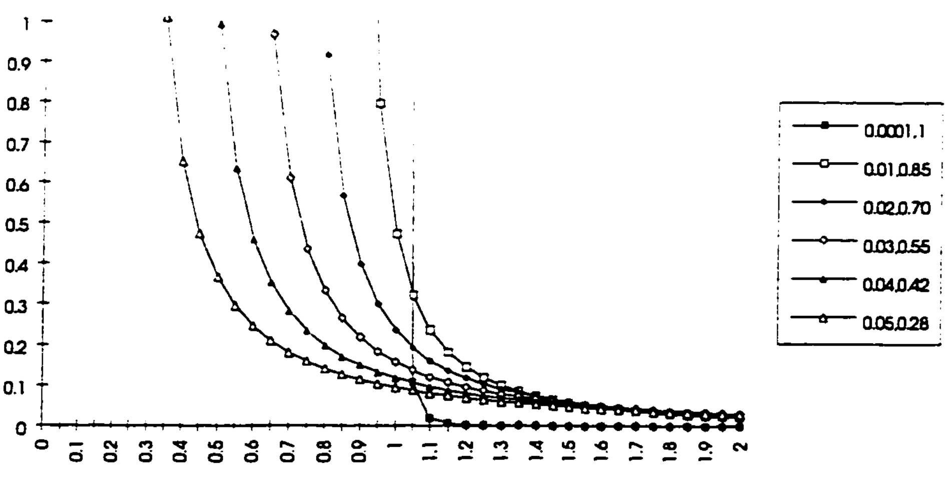


Figure 9.3 Weibull(0.28,β,γ) density function for 6 different values of β and γ.

Table 9.2 summarizes all important data for the 6 shifted Weibull(0.28,β,γ), which are used in our study.

Table 9.2 Data of shifted Weibull(0.28,β,γ) with μ=1

Case	1	2	3	4	5	6
β	0.0001	0.01	0.02	0.03	0.04	0.05
γ	1	0.85	0.70	0.56	0.42	0.28
cv	≈ 0	1	2	3	4	5

9.3 Cells in Series Layout

9.3.1 General

We first investigate model 2 as an example of a cells in series layout. The objective is to examine how the variability of the processing times changes the system performance and especially in the case of a different coordination policy. We analyse the optimization results to compare the performance of model 2 configuration operating under a different coordination scheme and with different coefficient of variations for processing times. We also want to investigate how good a system design is if it assumes a certain variability in processing times and this variability changes. For our model 2, we analyse the simulation results operating under different coordination schemes with different coefficient of variations for processing times and with fixed PAC parameters corresponding to the optimal setting to the chosen coefficient of variation.

We generate processing times using both of the shifted Weibull distributions described above. All relevant input data are as given for the variant A of model 2 (refer to Table 7.3). Each simulation run takes a total of 300 days of 24 working hour each, including 40 days of warm-up.

9.3.2 Exponential(β,γ)

To show the effects of different coefficient of variations on the system performance (total cost), eleven optimization runs with different values of coefficient of variation (*cv*) were made for each of 9 different coordination policies (PAC, MRP, PTO, PTO-$\tau \geq 0$, BSS, Kanban, LC, IC, CONWIP). The mean processing times at each cell were set to 10 minutes. In each of the eleven runs, processing times were generated using the shifted exponential(β,γ) with parameters as given in Table 9.1 for the cases referred to as 1 to 11.

Figure 9.4 shows the effect of the coefficient of variation on the total cost for the scenario with the delay costing option I (DCI) of cost calculation. Figure 9.5 shows similar results, but for the delay costing option II (DCII) scenario. Figure 9.6 presents the results for the LC policy. As *cv* varies from 0 to 1, the system processing times move from constant (deterministic) to purely exponential.

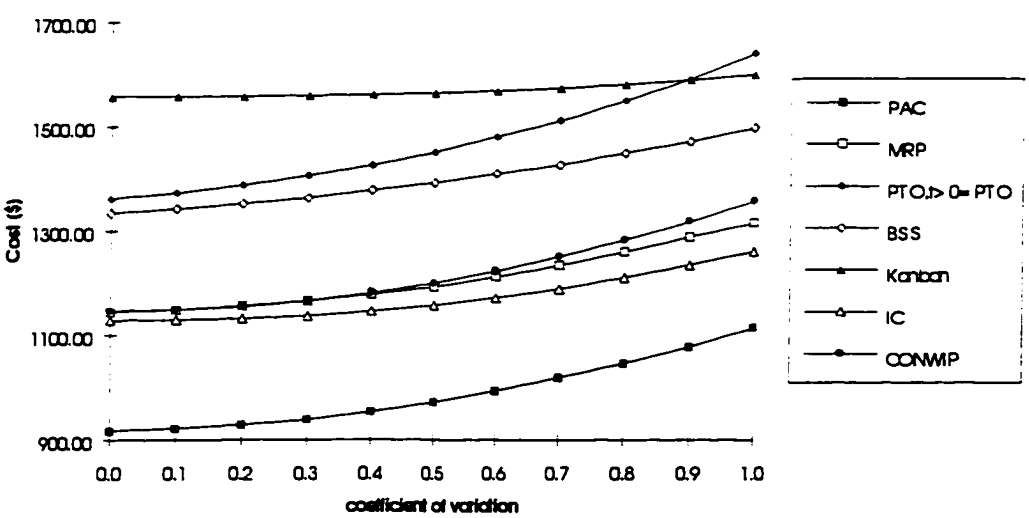


Figure 9.4 The effect of variability in processing times on the total cost: model 2, DCI, $1/\mu_j$=10 min.

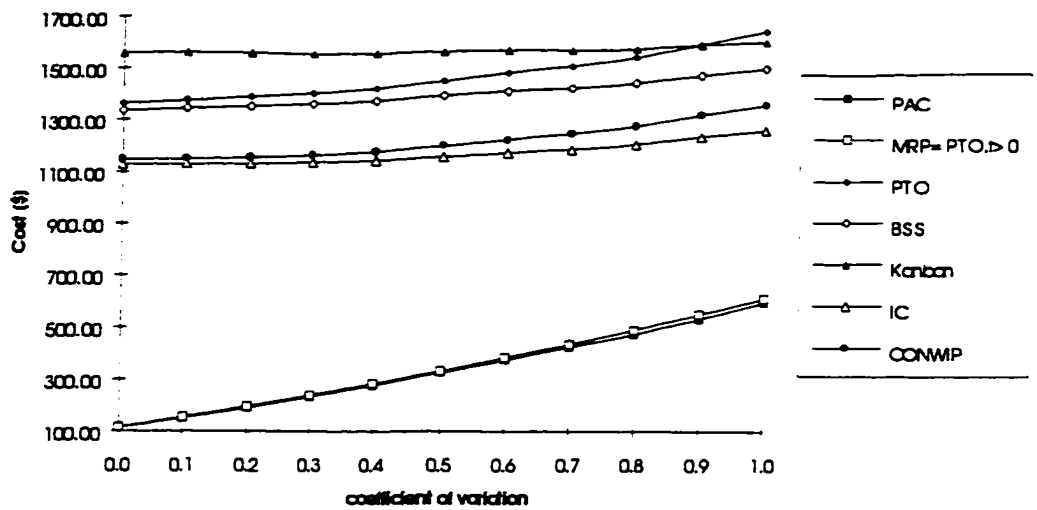


Figure 9.5 The effect of variability in processing times on the total cost: model 2, DCII, $1/\mu_j$=10 min.

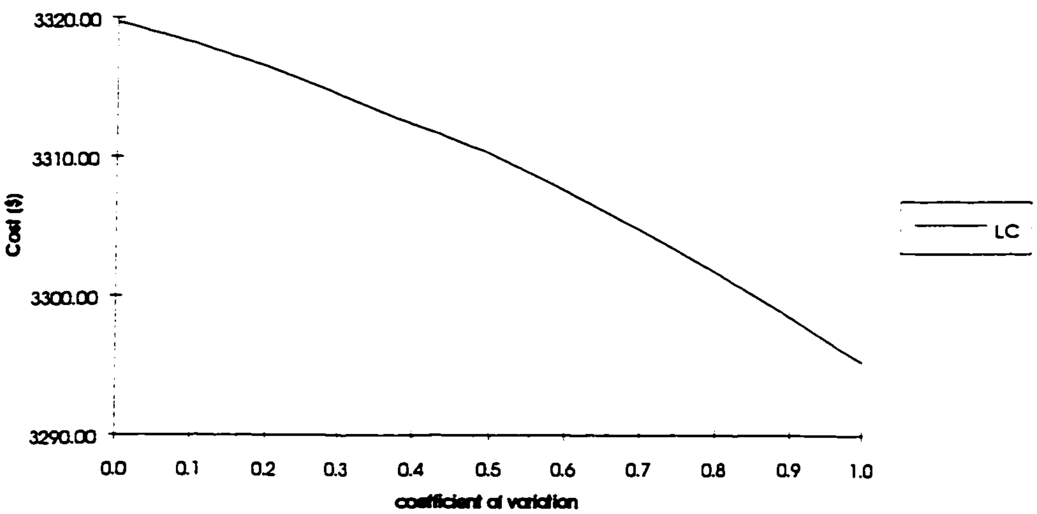


Figure 9.6 The effect of variability in processing times on the total cost: model 2, LC, $1/\mu_j$=10 min.

As expected, the total cost increases with value of cv. Generally, the policies PAC, PTO, MRP and CONWIP show the largest sensitivity to the increased cv value. Kanban is the most stable, but it is uniformly bad. The LC policy behaves totally differently, but the reason is easy to explain. The parameter setting for LC requires: $z>c$, where c indicates the number of identical servers. In our model $c=1$ which results in $z>1$, that means, $z_{min}=2$. The system tested has a very low utilization factor, and the required minimum values for the z parameters are much too high for the deterministic processing times to compensate for the high inventory costs. Larger processing times variability can better justify and use those high inventory levels. This does result in decreasing of costs, but note that these costs are still higher than anything else in Figures 9.4 and 9.5. In the next section, we will present results for higher utilization factors of model 2 (refer to 9.3.3).

For this scenario of model 2, the optimal parameter values of different policies had the same values for each of the different coefficients of variations. It suggests that for systems with lower utilization and relatively low coefficients of variations, it does not matter if we design on the deterministic or exponential assumptions. Only in the case of DCII cost calculation, and for the policies PAC and MRP (=PTO,$\tau \geq 0$), were the optimal parameter values different for each optimization run.

Ignoring machine failure, most automated production processes operate on essentially deterministic times, while the design of the system based on uncertainty assumes exponential processing. Therefore, it is interesting to see the effect of fixed parameters for both deterministic and exponential cases on the system performance under changing variability of the processing times. Figures 9.7 and 9.8 show this effect for the DCII cost scenario, for PAC and MRP respectively.

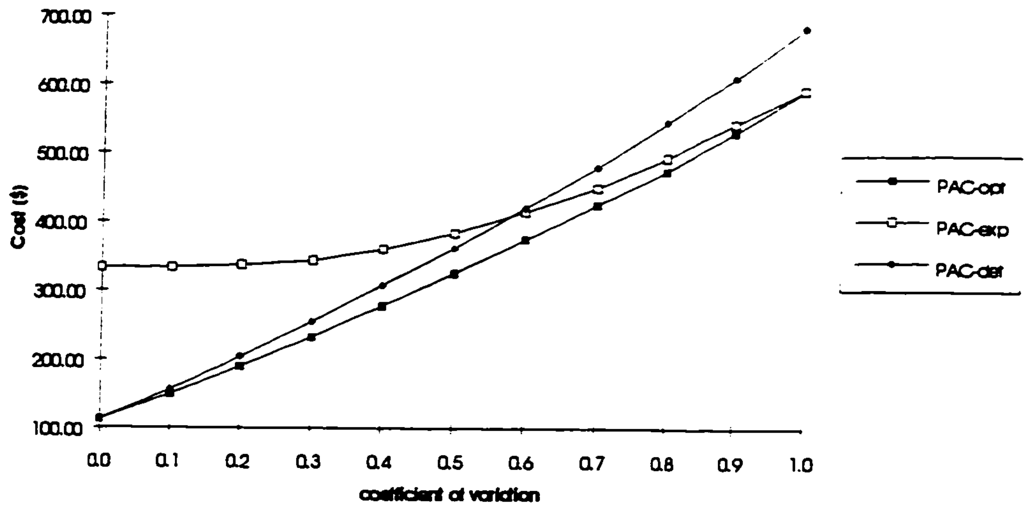


Figure 9.7 The effect of variability in processing times with fixed parameters values: model 2, DCII, PAC, $1/\mu_j$=10 min.

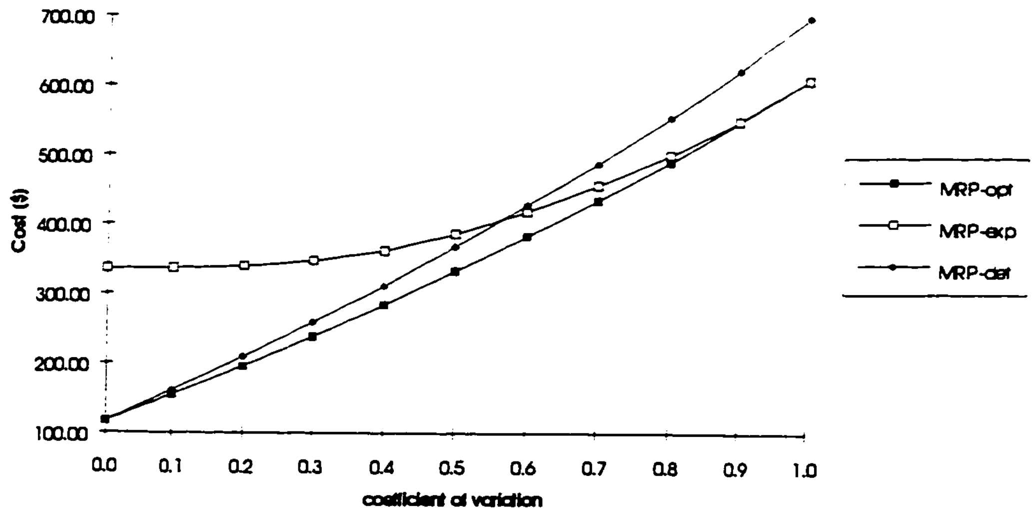


Figure 9.8 The effect of variability in processing times with fixed parameters values: model 2, DCII, MRP, $1/\mu_j$=10 min.

The "opt" curve gives the optimization results. The "exp" index corresponds to fixing the parameter settings to the optimal parameters for the case $cv=1$ and running individual simulations for each of the other cv values. Similarly, the "det" curve is obtained by performing simulations using the PAC parameter settings optimized for $cv=0$.

Relative to the "opt" curve, using the deterministic parameter setting does not increase the costs very dramatically. Up to the value of $cv=0.5$, the penalty up to 10%, is not too large, and for $cv=1$ it increases to approximately 15%. Designing on the exponential parameter setting does relatively well for cv not lower than 0.6. However, for smaller values of cv, the penalty cost increases very significantly with each lower value of cv, up to 200%.

The above results suggest that it make sense to design on the deterministic assumptions for the systems with extremely low utilization factors.

9.3.3 Weibull(0.28,β,γ)

The results in 9.3.2 advise that it is important to test the model with higher coefficients of variations and higher utilization factors. We set a mean processing time of each product ($1/\mu_j$) to 42 minutes, so the system is much more "busy". For each of 9 different policies, we want to perform 6 optimization test runs with processing times having cv values from approximately 0 to 5 with step 1. We use the previously described shifted Weibull(0.28,β,γ) distribution with parameters as given in Table 9.2 for the cases referred to as 1 to 6. This Weibull distribution generates a random variable X, and the processing time is calculated as a random variable $Y=(1/\mu_j)X=42X$.

Figure 9.9 shows the effect of coefficient of variations on the total cost for the scenario with DCI of cost calculation. Figure 9.10 shows similar results, but for DCII cost scenario. Again when cv=0, the system operates on deterministic (constant) processing times, and when cv=1, the processing times are purely exponential.

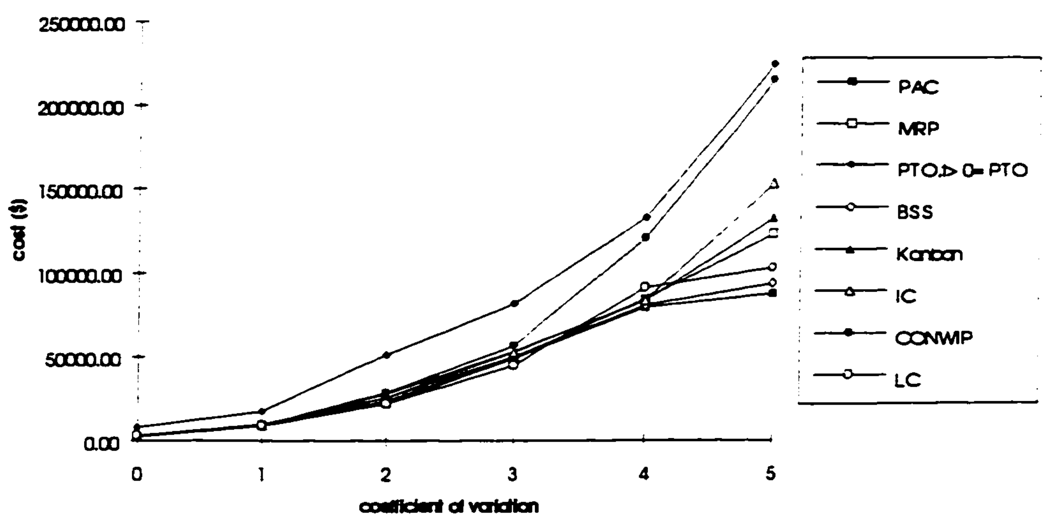


Figure 9.9 The effect of variability in processing times on the total cost: model 2, DCI, $1/\mu_j$=42 min.

Again the total cost increases with values of cv. PTO and CONWIP show the largest relative increase in costs. Kanban performs similarly to the other remaining policies; that means, it does not show the previously observed stability (compare with Figure 9.4). This is not surprising. Sarker and Fitzsimmons (1989) studied the effects of cv of the processing times on the efficiency of the 9-cell flow line performing under push and pull policy with a high system utilization. They observed that a pull system is always better at minimal WIP, but on the other hand it is less efficient than the push system, especially at higher cv. The LC policy behaves similarly to the other policies, and, as expected, does not show the previously observed decreasing of costs (refer to Figure 9.6).

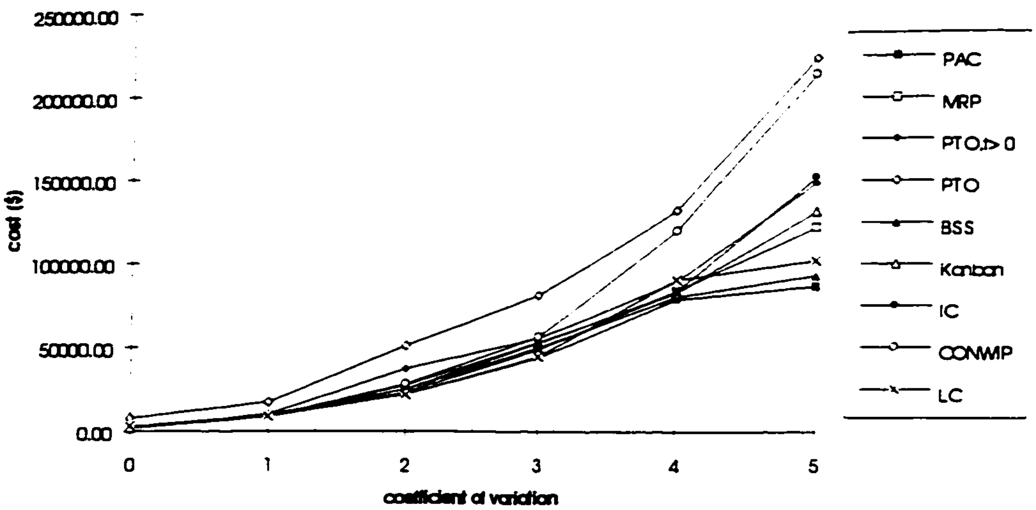


Figure 9.10 The effect of variability in processing times on the total cost: model 2, DCII, $1/\mu_j$=42 min.

To facilitate comparison of results for *cv*=0 and *cv*=1 with the previous results for the case of a low system utilization (refer to Figure 9.4 and 9.5), we construct charts on Figures 9.11 and 9.12 for the DCI and DCII of cost calculations respectively. All charts present a sequence of policies related to their costs to the first and the best PAC scheme. The total costs generated by PAC is set to 1, and the costs of all other policies are adequately proportional to the cost of the PAC policy. Charts A and B from Figure 9.11 correspond to the policies costs for a low utilization scenario; that is points *cv*=0 and *cv*=1 from Figure 9.4, and charts C and D conform to the policies costs for high utilization case, that is points *cv*=0 and *cv*=1 from Figure 9.9. Similarly, charts A and B are related to points for *cv*=0 and *cv*=1 from Figure 9.5, and charts C and D to points for *cv*=0 and *cv*=1 from Figure 9.10. In the case of DCII, MRP does well for both *cv*'s of a lower system utilization and for deterministic processing times when the utilization is high. In the case of

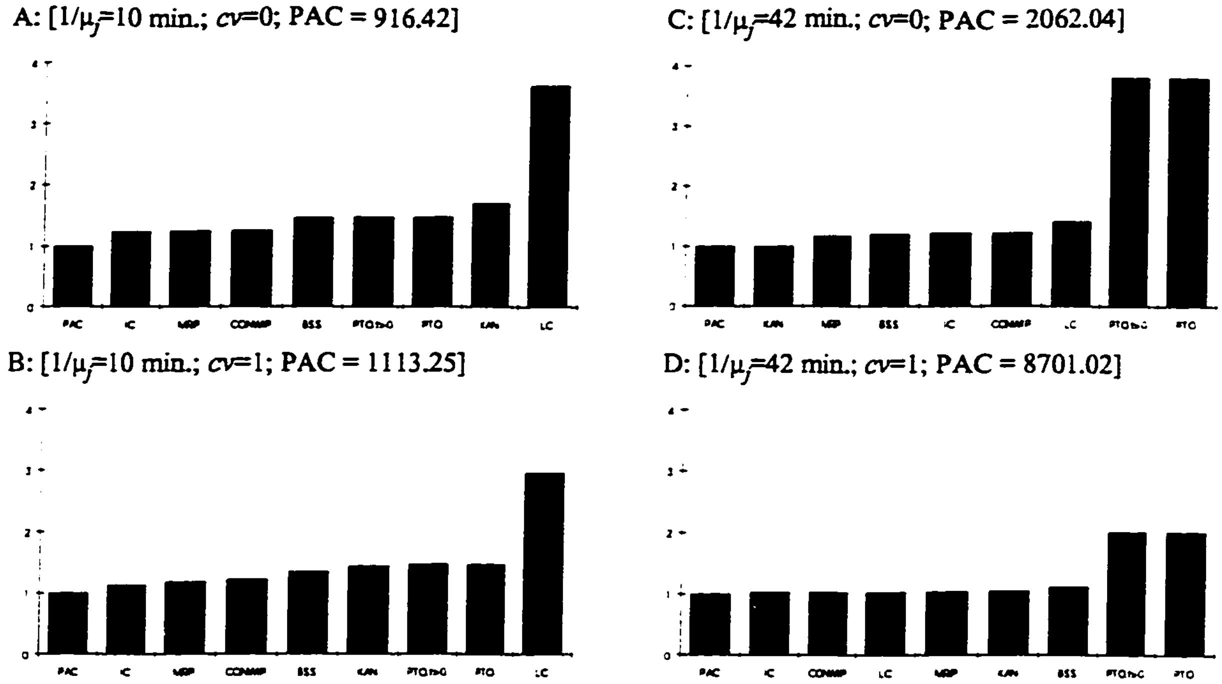


Figure 9.11 Comparison charts of relative performance: model 2, DCI.

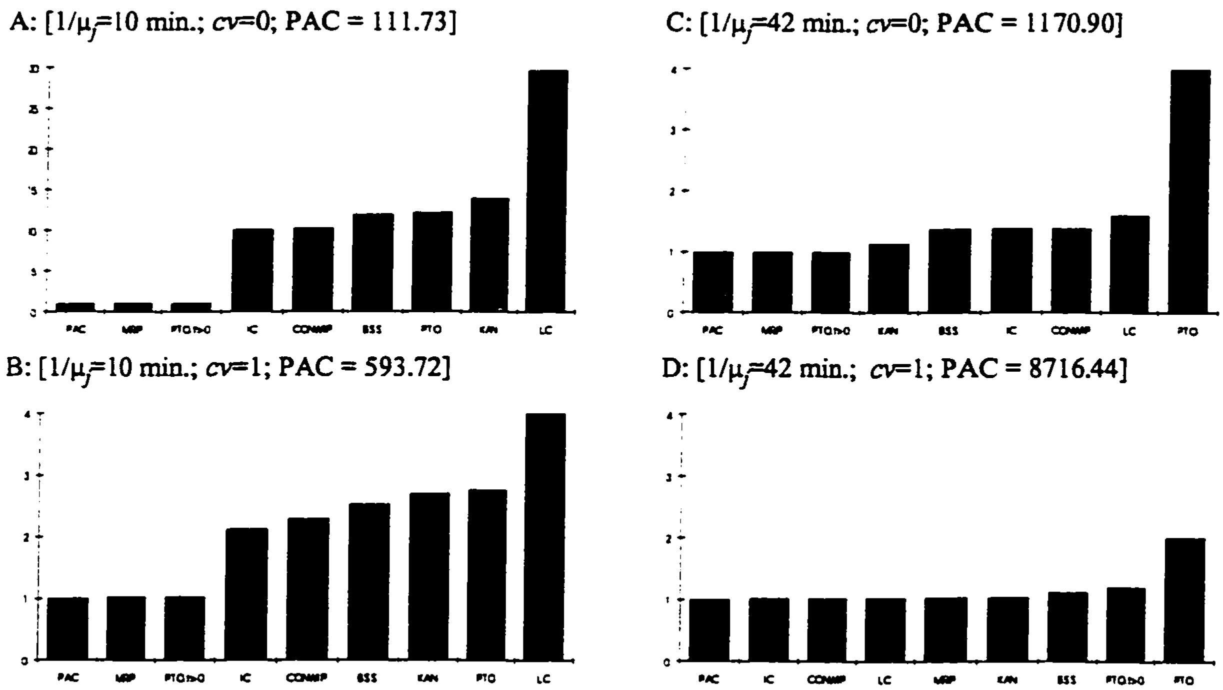


Figure 9.12 Comparison charts of relative performance: model 2, DCII.

a high utilization of the system, the PTO policy has the worst performance, while LC, which performed not so well in the case of low system utilization, does significantly better, especially for *cv*=1.

Since this scenario has much higher system utilization, as one might expect, the optimal values of parameters were different for each optimization run within each policy. Again we performed a number of simulations to examine the effects of fixing the parameters to those optimized for both extreme values of coefficient of variations, that is, for *cv*=0 and for *cv*=5, as well as those optimized for *cv*=1, the exponential case. The "opt" curve gives the optimization results. The "wei", "det" and "exp" indexes mean that we use the optimal parameter settings from the cases *cv* equal to 5, 0 and 1 respectively. Figures 9.13, 9.14, 9.15 and 9.16 show the effect of variability in processing times for the policies PAC, MRP, Kanban and CONWIP respectively.

For the PAC policy (Figure 9.13) the "wei" curve remains almost on a constant level. The "det" policy, which reaches the total cost of $2,129,775 for *cv*=5, and the "exp" policy, which reaches the total cost of $1,779,917 for *cv*=5, can perform very badly at high *cv*.

For MRP (Figure 9.14), the "det" and "exp" costs do increase, but not so significantly as it was in the case of PAC. For *cv*=5, the "det" curve reaches the total cost of $246,670 and the "exp" curve reaches the total cost of $295,425. Cost for "wei" curve decreases first and than shows a tendency to increase.

Kanban (Figure 9.15) has all three curves, "wei", "det" and "exp" with increasing costs, but for the "det" and "exp" ones the cost values reach as high as $2,191,707 and $2,079,466 respectively for *cv*=5.

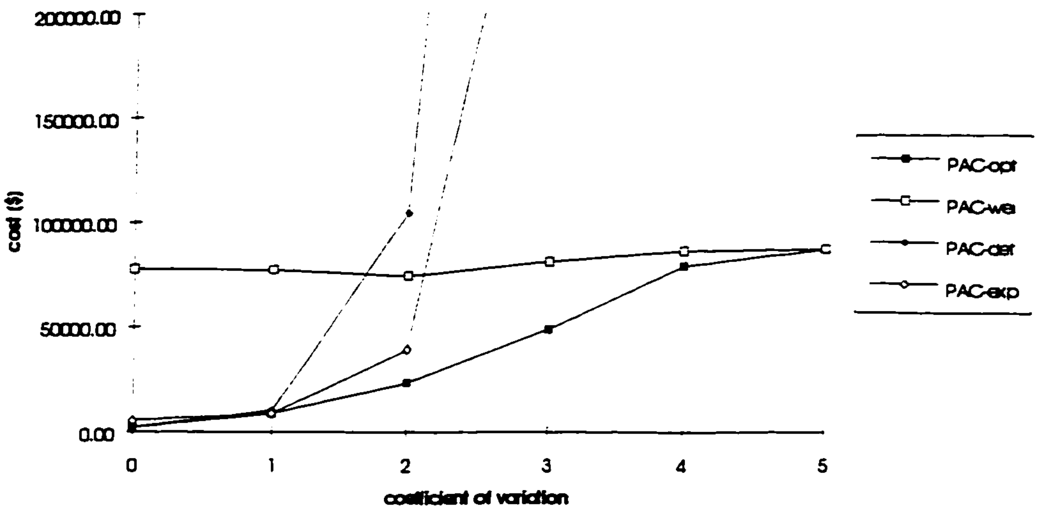


Figure 9.13 The effect of variability in processing times with fixed parameter values: model 2, PAC, DCI, $1/\mu_j$=42 min.

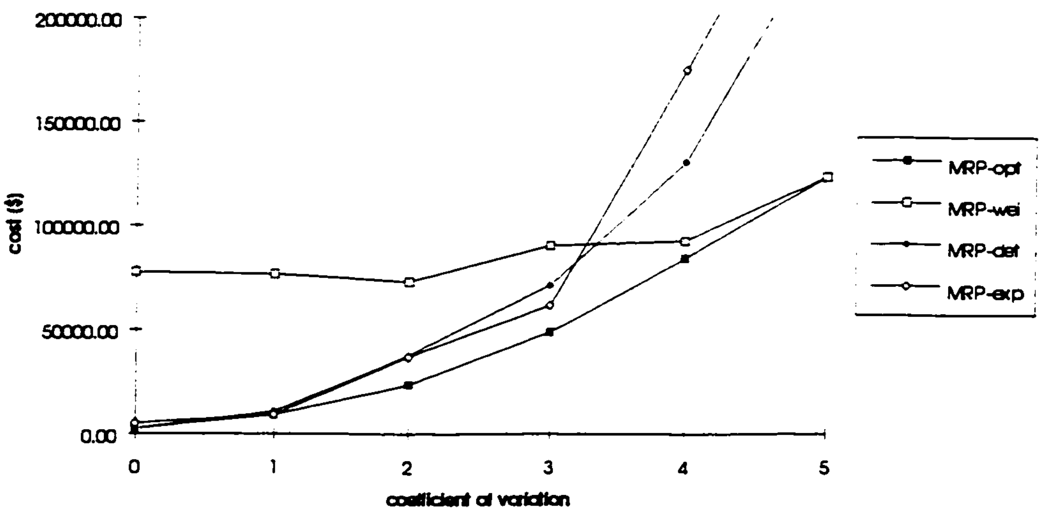


Figure 9.14 The effect of variability in processing times with fixed parameter values: model 2, MRP, DCII, $1/\mu_j$=42 min.

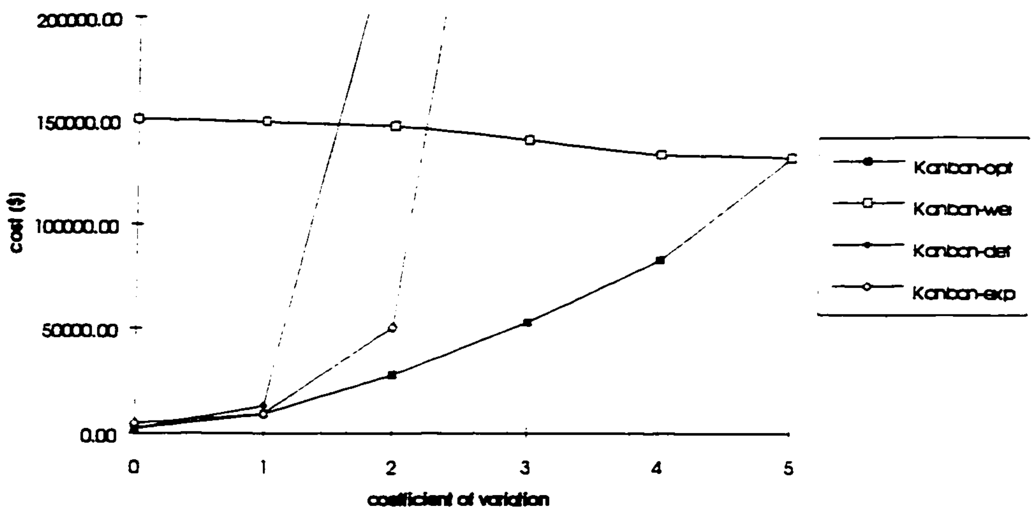


Figure 9.15 The effect of variability in processing times with fixed parameter values: model 2, Kanban, $1/\mu_j$=42 min.

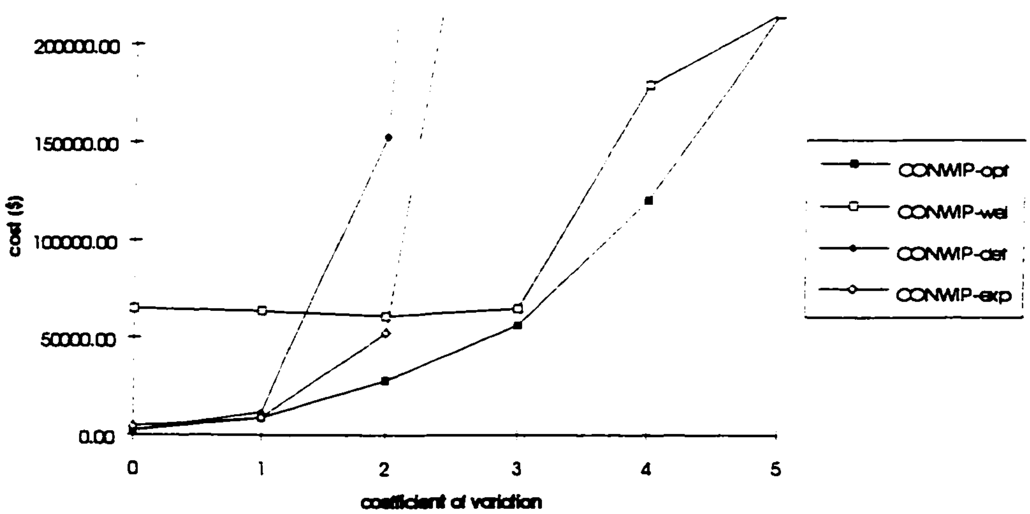


Figure 9.16 The effect of variability in processing times with fixed parameter values: model 2, CONWIP, $1/\mu_j$=42 min.

For CONWIP (Figure 9.16) and *cv*=5, the "det" curve reaches as high as $2,133,721 and the "exp" one as high as $2,027,027. The "wei" curve decreases for *cv* values equal to 4 and 3, and then it stays on an almost constant level.

Generally, fixing the PAC parameters to the "deterministic" scenario can result in a very significant growth of the costs when the *cv* of the processing times increases more than 1. Setting parameters to the optimal solution of the exponential case gives good results for *cv* not more than 2. Choosing PAC parameters for the *cv*=5 does not guarantee large decreasing of the costs in a case of decreasing variability. In the case of Kanban, it can even increase the total cost.

The obtained results suggest that for systems with high utilization, it make sense to design on the exponential assumptions, if the *cv* values are expected to be in the range between 0 and 2. For values of *cv* foreseen to be larger than 2, designing on the exponential or deterministic scenario can increase the penalty costs enormously. For *cv* values not larger than 1, both, the deterministic and the exponential suppositions, perform equally well.

9.4 More Complex Manufacturing Configuration

9.4.1 General

We investigate model 4 as an example of a more complex manufacturing configuration and analyse the simulation and optimization results to compare the performance of this configuration operating under different coordination scheme and with different coefficient of variations for processing times. The processing times at each cell are assumed to be distributed from the shifted Weibull distribution. The mean processing times ($1/\mu_j$) and all other relevant data are as for the case of the variant A of model 4 (refer to Table 7.7). In our study, we consider model 4A with setup times. Each simulation run takes a total of 300 days of 24 working hour each, including 40 days of warm-up.

9.3.2 Weibull(0.28,β,γ)

In a similar manner to that used in model 2, we analyse the effect of processing times variability on the system performance. For each of 9 different policies, we want to perform 6 optimization test runs with processing times having *cv* values from approximately 0 to 5 with step 1. Again, we use the shifted Weibull(0.28,β,γ) distribution with parameters as given in Table 9.2 for the cases referred to as 1 to 6. This Weibull distribution generates a random variable *X*, and the processing time is calculated as a random variable $Y=(1/\mu_j)X$.

Figure 9.17 shows the effect of coefficient of variations on the total cost for the DCI of cost calculation. Figure 9.18 shows similar results, but for the DCII cost scenario.

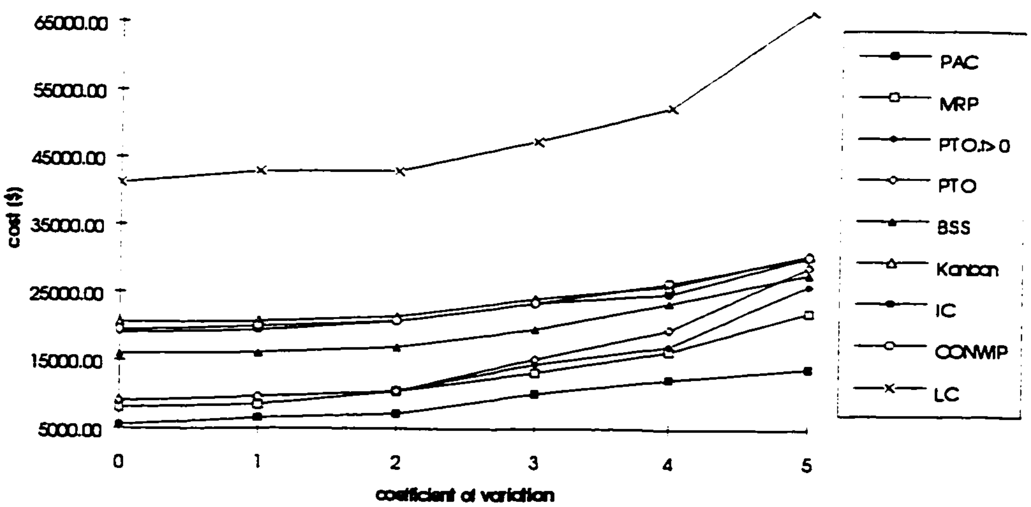


Figure 9.17 The effect of variability in processing times on the total cost: model 4, DCI.

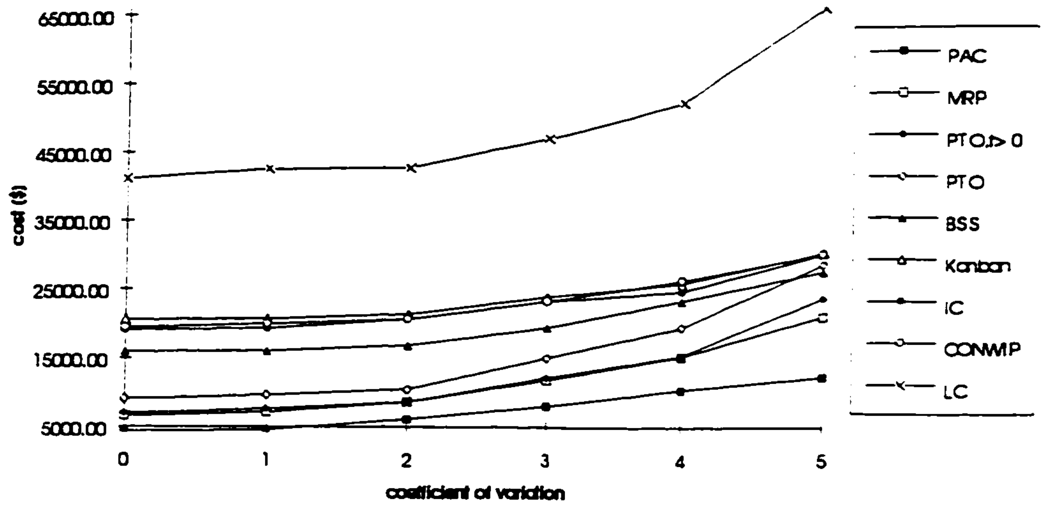


Figure 9.18 The effect of variability in processing times on the total cost: model 4, DCII.

Again when *cv*=0, the system operates on deterministic (constant) processing times, and when *cv*=1, the processing times are purely exponential. Generally, all policies increase costs with *cv* value in a similar way. However, PTO and LC show, especially for the higher value of *cv*, slightly more sensitivity.

In a similar way as for model 2, we performed a number of simulations to examine the effects of fixing the parameters obtained from optimizing both extreme values of coefficient of variations, as well as from optimizing the exponential scenario. Again, we use *cv*=0, as the deterministic ("det") case, *cv*=5, as the Weibull ("wei") case, and *cv*=1, as the exponential ("exp"). The "wei", "det" and "exp" curves are obtained by using optimal parameter settings from the case *cv* equal to 5, 0 and 1 respectively, and by performing individual simulations for each of the other *cv* value. The "opt" curve gives the optimization results. Figures 9.19, 9.20, 9.21 and 9.22 show the effect of variability in processing times with fixed parameters for the policies PAC, MRP, Kanban and CONWIP respectively.

For the PAC policy (Figure 9.19) the cost values of the "wei" curve are very close to the optimal values. For this scenario of model 4, the optimal PAC parameters for *cv*=5 perform well for a lower value of *cv*. Except for *cv*>4, the cost values of the "exp" curve are also very near the optimal ones. However, designing this system, using the PAC parameters optimized for the deterministic processing times will be not a good idea. It gives little improvement over the design at low *cv* and pays a high price at high *cv*.

For MRP (Figure 9.20), the cost values of the "wei" curve are 20% to 50% higher than the optimal values. The costs values of "det" and "exp" are so close to each other that they cannot be distinguished on the graph. Both design scenarios, "det" and "exp", perform well for *cv* values not exceeding 2, and for *cv*>2, they increase the cost by approximately 20%.

For Kanban (Figure 9.21) the "det" and "exp" curves have exactly the same cost values, and they reach as high as $45, 377 for *cv*=5. Both scenarios, "det" and "exp", have the same cost as the "opt" for *cv* equal to 0, 1 and 2, and then, for *cv*>3, increase the cost significantly. The "wei" curve decreases costs for cv=4 and 3, and then keeps them on almost the same level.

For CONWIP (Figure 9.22) the general results are quite similar as for the Kanban policy; however the "det" and "exp" curves, which have the same cost values, show relatively smaller increase of costs for *cv*>3.

Although it is dangerous to draw conclusions from a small sample (2 systems), the results here seem to indicate that, for CONWIP, Kanban, MRP and the general PAC, designing on a high *cv* assumption appears to produce good results across a wild range of *cv*'s. For the policies PAC and Kanban, designing on the deterministic assumption, *cv*=0,

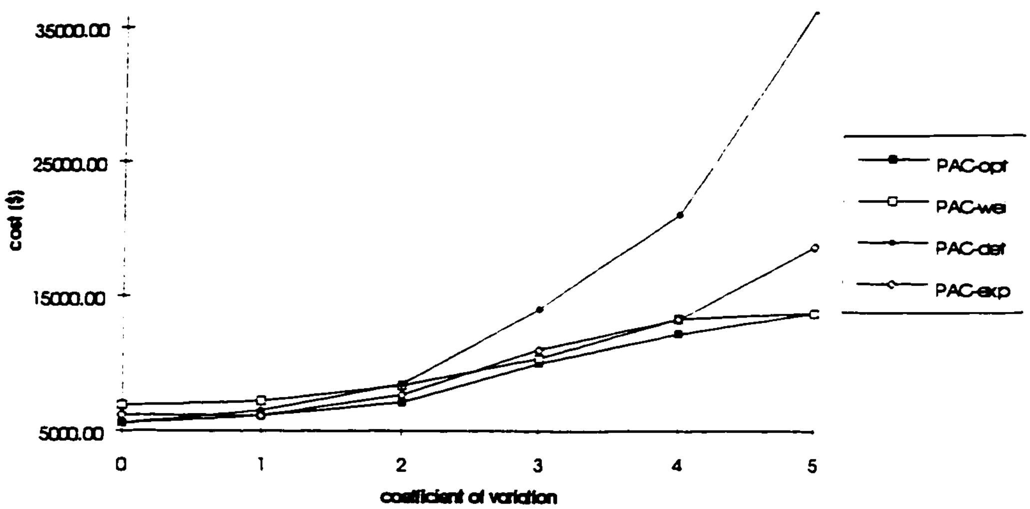
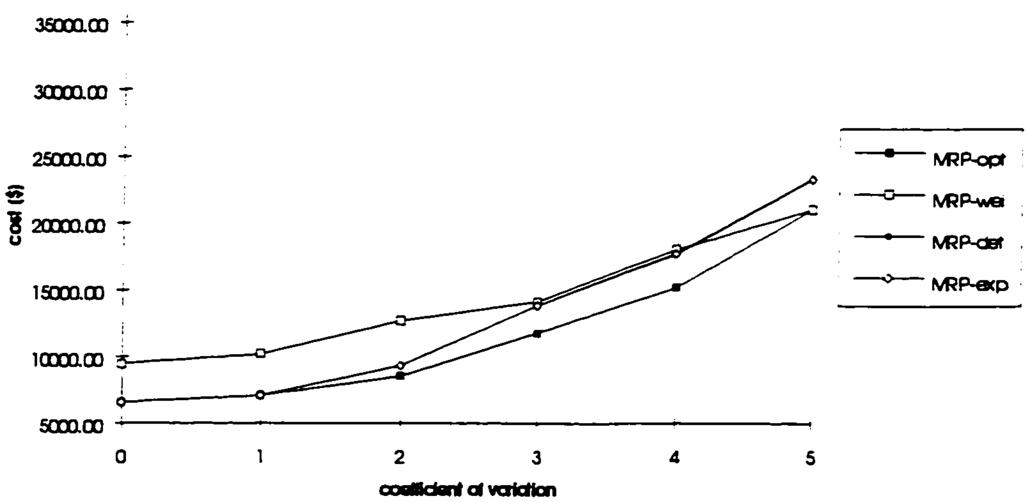
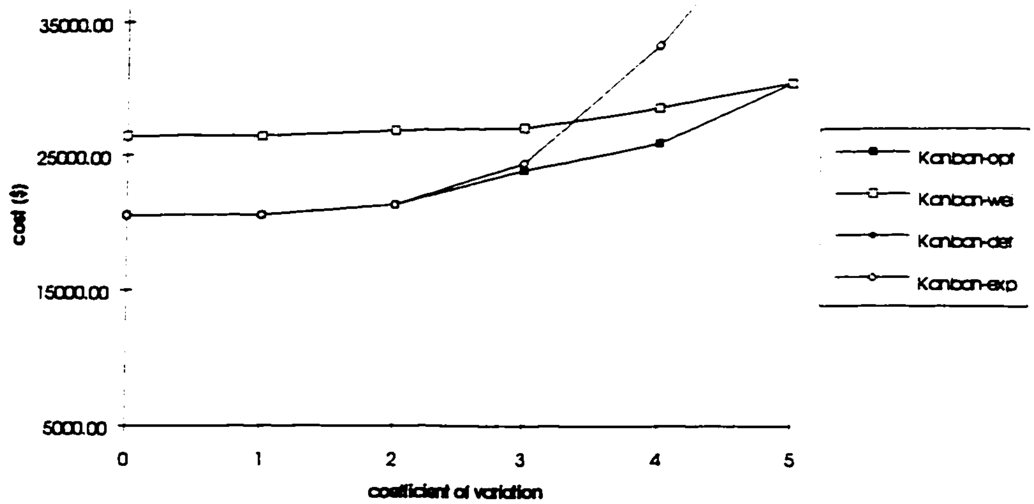


Figure 9.19 The effect of variability in processing times with fixed parameter values: model 4, DCI, PAC.



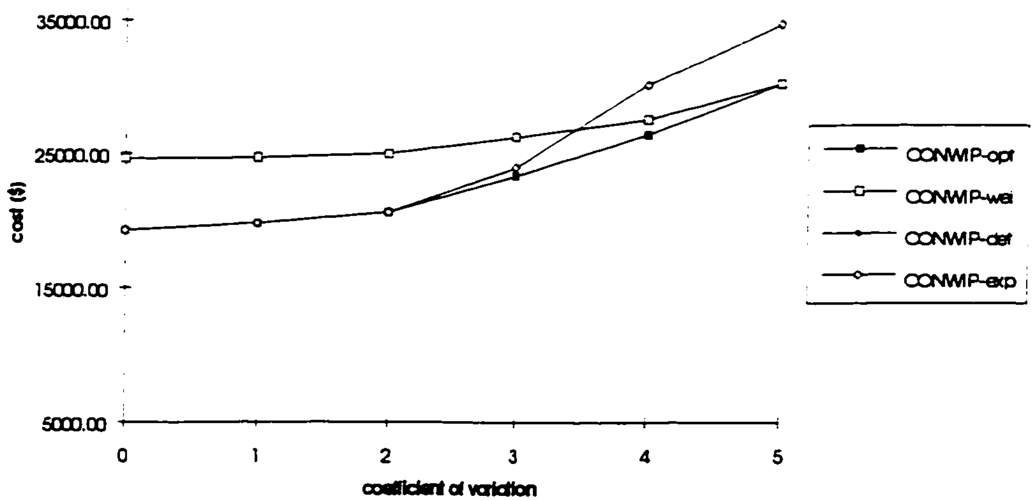
Note: The "MRP-det" and "MRP-exp" are very close to each other.

Figure 9.20 The effect of variability in processing times with fixed parameter values: model 4, DCII, MRP.



Note: The "Kanban-det" and "Kanban-exp" are equal to each other.

Figure 9.21 The effect of variability in processing times with fixed parameter values: model 4, Kanban.



Note: The "CONWIP-det" and "CONWIP-exp" are equal to each other.

Figure 9.22 The effect of variability in processing times with fixed parameter values: model 4, CONWIP.

can lead to disaster as high *cv* systems are encountered. In the case of PAC and MRP, designing on the exponential assumption, *cv*=1, is not too bad. Generally, the exponential scenario gives good results for *cv* values in ranges between 0 and 3. This suggests that modeling of processing times by exponential distribution, so commonly used in the research literature, results in robust design of the system.

9.5 Closing Remarks

We analyzed two different manufacturing configurations on the effects of processing time variability on the system performance and for the case of 9 different coordination schemes. Generally, all policies show increase of a total cost with increased values of a coefficient of variation and in quite a similar way. In all cases studied, PTO showed relatively the largest sensitivity to the increasing values of *cv*.

To design a system that is robust enough to deal with uncertainty in processing times variability is not an easy issue. Generally, we observe that small changes in *cv* (increasing or decreasing) can still be handled by most of the coordination schemes without drastically increasing the costs. The sensitivity to changes of the *cv*, when fixing the parameters to the "deterministic" scenario, seems to be much higher for a relatively simple layout with higher system utilization than for a complex and relatively not too "busy" one. In the case of a more complex production process, designing the system for the PAC parameters of "highly variable" scenario appears to work well for PAC, MRP, Kanban and CONWIP; the "deterministic" and "exponential" scenarios perform quite satisfactory for *cv* values not larger than 2, but can be highly sensitive at larger levels.

Chapter 10

CONCLUSIONS AND SUGGESTIONS FOR FURTHER RESEARCH

10.1 Summary and Conclusions

In this thesis, we have outlined the structure of a simulation model and an optimization algorithm for a multiple-cell multiple-product manufacturing system coordinated by PA cards. The simulation model permits a rapid performance evaluation for a given choice of PAC control parameters. The model implements the main features of the PAC concept. It has been developed mainly to function as a cost evaluation in the overall optimization process. Nevertheless, this model is a contribution to the research on various manufacturing issues and is essential to study the PAC control for simple and complex manufacturing systems. The optimization algorithm, PACOPT, uses the simulation model for cost function evaluation and is based on hill climbing methods similar to those of Hooke and Jeeves. Within the optimization process, we chose to evaluate the cost function by a single simulation run and started each simulation with the same values of the stream numbers for the random variables. This means that all simulations are realizations of the same system conditions, and are in fact deterministic. The single simulation run does not correspond directly to the expected function value. However, by making the length of simulation reasonably long, the values obtained are a fair approximation of the function value. To date, the simulation model has proven to be fast enough in execution to make such an optimization quite feasible for systems with a small number of products (less than 20).

We illustrated the application of the optimization algorithm on five different manufacturing configurations and for nine different coordination policies. Especially in the case of more complex production processes, the generated results are not necessarily optimal. We verified the approximate optimality of these policies by comparing the ranking of the policies for different models and scenarios and by studying in more detail the generated results.

We presented also another optimization algorithm, PACRAN, based on the random search principle. The random algorithm has demonstrated the capability to produce solutions with a slightly improved objective function value when compared to the solution obtained through the PACOPT algorithm. However, PACRAN is computationally very expensive.

The optimized solution for bulk customer arrivals resulted in significant increase of the total cost when compared to non bulk (standard Poisson) arrivals. A queueing analysis enabled us to shed some light on this phenomenon. In order to understand this phenomenon and to confirm that this is not due to a programming error, we compared simulation results with the theoretical formulations based on queueing theory. The results related to the BMAP/G/1 queue are compared with the PAC simulation results. Simulation results produce queueing behavior similar to the BMAP/G/1 analysis. This serves as an important validation of our simulation model. The more detailed analysis of one selected manufacturing layout shows how some PAC parameters filter the customer arrival process to generate arrival of raw material into the system.

We performed some additional optimization runs to study and justify the robustness of the optimization procedure, which evaluates the cost function by execution of a single simulation run. In the results obtained, the sequence of policies remains the same for other realizations of the system, and the variation in the objective function does not affect the final results or conclusions significantly.

Buzacott and Shanthikumar discussed some properties of the PAC system, which describe the impacts of certain choices of PAC parameters (z, k, r, τ) and processing times (S) on the system performance, and specifically on the characteristics of the departure process and the shipment to customers process. They developed formulas for the time when process tags and material are moved from cells to stores and stated five properties that can be verified from those formulas. We used these properties of cells in series to validate our models. We confirmed the five properties for cells in series configuration, using a three-cell layout. We also examined those properties for a more complex manufacturing configuration. The general trends given by the properties hold for the complex setting; however as expected, there are some exceptions. An examination of the impact of batching orders for products which are required for processing other products, shows quite interesting results. By setting the PA cards batch size of those products to multiples of the amount required in the bill of material, we observe a repeating pattern of costs and delay of final products to the customer. Generally, the performed experiments show the importance of the BOM issue and product characteristics on the impact of PAC parameters and processing times on the system performance and the total cost.

We have carried out a number of experiments to examine the impact of processing times variability on the system performance and the optimal parameter values for different control policies, and how we can design a system robust enough to deal with the variability of the processing times. We investigate the dependence of various policies on the different variability in processing times in the case of two models, a three-cell flow line and a more complex manufacturing layout. Although it is dangerous to draw conclusions from only two systems, the results seem to indicate that the common practice of designing on exponential assumptions produces good results, if the coefficient of variation is not larger than 2, but can be highly sensitive at larger levels. Designing on deterministic assumptions can lead to disaster if the higher variability occurs. It implies that for any manufacturing process controlled by a given coordination policy, there are processing times scenarios which are more robust than others to deal with the processing time variability. Generally, we observe that small changes in coefficient of variations can still be handled by most of the coordination schemes without drastically increasing the costs. Further, performing this kind of analysis illustrates how the developed PAC techniques can be used for studies of different issues, and how relatively easy it is to implement in the program the various modeling aspects.

The PAC control scheme was initiated by Buzacott and Shanthikumar. They indicated the need for performance evaluation and optimization models. They also made some progress using simple queueing models. However, the systems they studied are very simple and the queueing models require exponential processing times, while shifted exponential or deterministic are much more appropriate in most practical settings. Our PAC model, in contrast, enables simulation of almost any realistic situation and the optimization of not too large systems. As such, we constructed a laboratory environment which enables us to examine the behavior of many different manufacturing control systems. The created PAC model is a step forward in bringing together theoretical models to the true systems and creating possibilities for detailed studies of production processes.

10.2 Main Contributions of the Thesis

Through our work, we demonstrated the new PAC concept for material and information flow control in multicellular manufacturing systems. We developed and presented the concept of a simulation of PAC. The simulation model can represent quite complex manufacturing systems, while the program requires a small number of events. We put forward the PAC concept of numerically optimizing this simulation. We have shown that this concept is practical for systems of moderate complexity given a particular viewpoint of what it means to optimize the simulation function.

By carrying out optimization studies of the PAC system, we discovered some additional and important constraints on the PAC parameters *r* (PA cards batch size) and *k* (number of process tags). Parameter choices that do not satisfy these constraints result in an infeasible system, that is, in the system that becomes blocked in the sense that neither material nor information continues to flow. It is desirable to eliminate as many unworkable situations as possible before carrying out each time a demanding simulation. Unfortunately, there are still situations when the production process is blocked because of the specifics of the production process itself, which we have not yet been able to eliminate. However, one good aspect of the simulation runs with blocking is that they take significantly less time than "proper" ones. We specified the required PAC parameter settings for IC/CONWIP operating in a more complex manufacturing environment.

We have shown that for each specific instance of the manufacturing layout the optimal coordination scheme can depend significantly on distributional assumptions. Therefore, we cannot necessarily apply findings from studies assuming exponential processing times and Poisson arrival of demand to the systems operating under different conditions.

We examined certain issues, not considered by previous research, as variability of processing times, nature of arrival process, nature of assembly, batching and more complex routing, and their impact on the system performance and design.

The coefficient of variation of processing times, used as a measure of processing times variability, has a demonstrable effect on the design of the system. We have shown that for a given manufacturing layout operating under a given coordination policy, there are processing times scenarios which are more robust than others to deal with the processing time variability.

We have been able to give some qualification of the nature of bulk arrival of customer demand. We demonstrated how some PAC parameters filter the customer arrival process to generate arrival of raw material into the system.

We have shown how important issues of bill of material are, assembly operations and product characteristics on the impact of different PAC parameters and processing times in the case of more complex manufacturing systems.

We demonstrated that designing of the manufacturing system needs to be tailored to the specific characteristics of the system. We have shown that policies which work well for one system are not necessarily a good solution for the same system operating under different conditions.

10.3 Suggestions for Further Research

10.3.1 General

This current simulation optimization model represents an intermediate point in trying to understand the behavior of PAC controlled systems. Buzacott and Shanthikumar wished to analyze the PAC systems from the viewpoint afforded by queueing theory. Thus they were forced to make the assumption of exponential processing times and Poisson demand arrivals. Obviously, no such assumptions are necessary in a simulation model. The simulation model that we have developed to date has been used mainly as an environment to develop and test our parameter optimization routines. However, the PAC simulation model is also a starting point in the representation of realistic factory problems in studying their behavior and designing operating policies. It offers many areas for extension and modification.

10.3.2 Dependency on Distributional Assumptions

Arrival processes

Initially our simulation model assumed Poisson interarrival times for the customer demand, as this allowed a ready comparison with the work in Buzacott and Shanthikumar. However, no such assumptions are necessary. Previous studies reporting the impact of the demand arrival process are done by Altiok and Ranjan (1995), Gascon, Leachman and Lefrancois (1994), Albin (1984), Albin (1982), Wolff (1982) in various types of manufacturing systems. An interesting situation is the impact of bulk arrivals on system performance. We have seen substantial impacts when we replace the assumption that the arriving customer requires one item with the assumption that arriving customers require x items, where x is drawn from a geometric distribution with a mean of 1. The pertinent question is how different demand processes influence the PAC system. The future research could investigate the effect of different arrival of customer demand on the system performance.

Production processes

As seen in Gershwin (1994) and Buzacott and Shanthikumar (1992b), it is often convenient to assume exponential processing times as a model even though this assumption is usually physically unrealistic. In real-life manufacturing systems, processing times are usually better thought of as deterministic or uniform with a small variance so long as one ignores machine breakdowns and repairs. When modeling the production processes, equally important are issues of the appropriate choice of distribution for processing times, as well as machine failure, imperfect parts and rework. The simulation environment can easily deal with separate modeling of the part production and machine failure processes. We generated optimization results assuming exponentially and uniformly distributed processing times. We believe that an examination of the distributional dependence of system performance is worth some more detailed study. In addition, it would be desirable to investigate how sensitive the optimal control policies themselves are to distributional assumptions.

Setups

Setups are always difficult to analyze under queueing frameworks. In our simulation model, we use deterministic setup times but replacing these with random duration setups is obviously easy. The performance effects of setup times is a compelling issue for design stage modeling because of the possibility of investing in new technology to reduce setup time in producing an economic lot size (Hong and Hayya, 1993). We believe that it is important to develop an understanding of the impact of setups on system performance for given PAC parameter choices and their impact on the optimal PAC parameter choice.

10.3.3 Dependency on Priority Assumptions

The issue of priorities comes up in two key places in Buzacott and Shanthikumar's discussion. Priorities occur primarily in multi-product assembly situations. The first case occurs when there are parts in a store but not enough to satisfy the requisitions from two or more subsequent cells. We have to decide which cell should be supplied with parts. The second case occurs when a cell has finished producing a batch of parts and there are PA cards at the cell for more batches of more than one part type. We have to decide which part should be produced next. Buzacott and Shanthikumar assume that some sort of simple priority schemes will suffice in both cases. These issues are potentially quite complex. Even viewing these as single stage problems, the first case is related to issues of admission control to the subsequent cells and the second can be seen as a problem of queueing control (Rosa-Hatko and Gunn, 1997). In the context of the overall manufacturing system, these priority issues become even more interesting. Since it is unlikely that one can develop optimal control strategies in this larger context, it would be of interest to simulate how control policies developed for a single server setting would perform in the larger context.

10.3.4 Issues of Yield

In our investigation to date, we have not considered issues of imperfect production. However, variable yields can produce quite different system performance. Failures that only involve rework within the cell can probably be dealt with by changing the production time distributions. However failures that require scrapping a finished part, and some or all of the components included in it, imply that new PAC cards have to be created in the system. If we attempt to forecast yield, then this may also mean that more components will be produced than are actually needed. As Buzacott and Shanthikumar indicate, the simulation program will now need to deal with surplus tags and cancellation notes in addition to the other types of cards. Effects of yield losses have been studied, among others, by Ou and Wein (1995), Bitran and Dasu (1992) and Buzacott and Shanthikumar (1992b) but the context of the PAC simulation seems to provide a potentially richer environment.

10.3.5 Disassembly Operations

To date, the PAC simulation model allows for only process and assembly activity at cells. It would be possible to permit disassembly operations. Again, most of the logic for this type of situation has been discussed in Buzacott and Shanthikumar. The main simulation requirement is in the form of surplus tags since the disassembly may end up producing byproduct parts that are not yet required by any customer or cell.

10.3.6 Cost Factors

In order to carry out optimization studies, we had to combine all measures of performance generated by simulation model into one overall criterion using appropriate costs. Costing is itself, however, problematic in that it introduces yet another fundamental characteristic of the manufacturing process: the additional cost parameters. We generated some optimization results, where different cost factors were used. They show the impact of cost factors on the system performance. Studying carrying cost ratios' dependence on the choice of the coordination policy is certainly an area worth more detailed investigation.

10.3.7 Parameters Setting for the MRP Policy

The PAC parameters setting proposed by Buzacott and Shanthikumar for the MRP policy does not always seem to satisfy a "real-life" MRP scheme. The requisition delay, identified with the lead time, should be related to the flow time of a typical job through the stages of the manufacturing process. However, the results indicate that the values of delay can depend more on the cost factors than on lead times, which implies that the given MRP parameter setting is not sufficient. The issue of a proper definition of the PAC parameters setting for MRP is quite complex, but regarding its overall importance, it should be examined by much more extensive studies in the near future.

10.3.8 Optimization Algorithm

The optimization algorithm, based on hill climbing methods similar to those of Hooke and Jeeves does perform quite well, but does not guarantee a global optimum, especially in the case of a more complex manufacturing layout. Future research can be carried out on modifying the algorithm, extending it and improving its convergence rate, by applying more recent ideas. There are ways in which our approach can be seen as a form of recently widely applied *Tabu Search Techniques* (Glover and Laguna, 1992, Battiti and Tecchiolli, 1994). Implementation of some principles of this method to the PAC optimization model is worth consideration.

The CPU time required for a single simulation run is a key issue in the improving of the efficiency of the optimization procedure. SIMLIB uses a linked storage allocation for event-list processing, which can be replaced by more efficient search techniques, as *binary search* or other algorithms involving data structures other than linked lists, such as *trees*, and *heaps* (Law and Kelton, 1991). The choice of the best event-list handling technique depends mostly on the simulation type, parameters and used distributions. We could also shorten or even eliminate the warm-up period, and or shorten the total simulation time. The methods and the trade off for speeding of the simulation are worth investigation.

11 REFERENCES

Adby, P. R. and Dempster, M. A. H. (1974). *Introduction to Optimization Methods*. Chapman and Hall Ltd., London.

Albin, S. L. (1984). Approximating a Point Process by a Renewal Process, II: Superposition Arrival Processes to Queues. *Operations Research* **32**, 1133-1162.

Albin, S. L. (1982). On Poisson Approximations for Superposition Arrival Processes in Queues. *Management Science* **28**, 126-137.

Altiok, T. and Perros, H. G. (1986). Open Networks of Queues with Blocking: Split and Merge Configurations. *IIE Transactions* **18**, 251-261.

Altiok, T. and Ranjan, R. (1995). Multi-stage, pull-type production/inventory systems. *IIE Transactions* **27**, 190-200.

Altiok, T. and Stidham, Jr., S. (1983). The Allocation of Interstage Buffer Capacities in Production Lines. *IIE Transactions* **15**, 292-299.

Andradottir, S. (1996). A Scaled Stochastic Approximation Algorithm. *Management Science* **42**, 475-498.

Askin, R. G., Mitwasi, M. G. and Goldberg, J. B. (1993). Determining the Number of Kanbans in Multi-Item Just-in-Time Systems. *IIE Transactions* **25**, 89-98.

Askin, R. G. and Standridge C. R. (1993). *Modeling and Analysis of Manufacturing Systems*. J. Wiley & Sons, Inc., New York.

Bai, S. X. and Gershwin, S. B. (1994). Scheduling manufacturing Systems with Work-in-Process Inventory Control: Multiple-Part-Type Systems. *Int. J. Prod. Res.* **32**, 365-385.

Bard, J. F. and Golany, B. (1991). Determining the Number of Kanbans in a Multiproduct, Multistage Production System. *Int. J. Prod. Res.* **29**, 881-895.

Barton, R. R. and Ivey, Jr., J. S. (1996). Nelder-Mead Simplex Modifications for Simulation Optimization. *Management Science* **42**, 954-973.

Battiti, R. and Tecchiolli, G. (1994). *The Continuous Rreactive Tabu Search: Blending Combinatorial Optimization and Stochastic Search for Global Optimization*. Dipartimento di Matematica, Universita di Trento, Italy.

Bazaraa, M. S. and Shetty, C. M. (1979). *Nonlinear Programming and Network Flows*. J. Wiley & Sons, New York.

Berkley, B. J. (1994). Testing Minimum Performance Levels for Kanban-Controlled Lines. *Int. J. Prod. Res.* **32**, 93-109.

Berkley, B. J. (1993). Setting Minimum Levels for Two-Card Kanban-Controlled Lines. *Int. J. Prod. Res.* **31**, 1003-1021.

Berkley, B. J. (1992). A Review of the Kanban Production Control Research Literature. *Production and Operations Management* **1**, 393-411.

Berkley, B. J. (1991). Tandem Queues and Kanban-Controlled Lines. *Int. J. Prod. Res.* **29**, 2057-2081.

Berkley, B. J. and Kiran, A. S. (1991). A Simulation Study of Sequencing Rules in a Kanban-Controlled Flow Shop. *Decision Sciences* **22**, 559-582.

Bertsekas, D. P. (1995). *Nonlinear Programming*. Athena Scientific, Belmont, Massachusetts.

Bertsekas, D. P. (1987). *Dynamic Programming, Deterministic and Stochastic Models*. Prentice-Hall, Englewood Cliffs, NJ.

Bielecki, T. and Kumar, P. R. (1988). Optimality of Zero-Inventory Policies for Unreliable Manufacturing Systems. *Operations Research* **36**, 532-541.

Bitran, G. R. and Dasu, S. (1992). Ordering Policies in an Environment of Stochastic Yields and Substitutable Demands. *Operations Research* **40**, 999-1017.

Bitran, G. R. and Chang, L. (1987). A Mathematical Programming Approach to a Deterministic Kanban System. *Management Science* **33**, 427-441.

Box, G. E. P., Hunter, W. G. and Hunter, J. S. (1978). *Statistics for Experiments. An Introduction to Design, Data Analysis, and Model Building*. John Wiley & Sons, NY.

Bunday, B. D. (1984). *Basic Optimisation Methods*. Edward Arnold Publishers Ltd. London.

Buzacott, J. A. (1967). Automatic Transfer Lines with Buffer Stocks. *Int. J. Prod. Res.* **5**, 183-200.

Buzacott, J. A. and Hanifin, L. E. (1978). Models of Automatic Transfer Lines with Inventory Banks; A Review and Comparison. *AIIE Transactions* **10**, 197-207.

Buzacott, J. A., Price, S. M. and Shanthikumar, J. G. (1994?). The Performance of Kanban Controlled Serial Production Systems. (copy received from J.A. Buzacott).

Buzacott, J. A., Price, S. M. and Shanthikumar, J. G. (1991). Service Level in Multistage MRP and Base Stock Controlled Production Systems. *Proceedings of the Joint US/German Conference on New Directions for Operations Research in Manufacturing.*

Buzacott, J. A. and Shanthikumar, J. G. (1992a). A General Approach for Coordinating Production in Multiple-Cell Manufacturing Systems. *Production and Operations Management* **1**, 34-52.

Buzacott, J. A. and Shanthikumar, J. G. (1992b). *Stochastic Models of Manufacturing Systems.* Prentice-Hall, Englewood Cliffs, NJ.

Chang, T.-M. and Yih, Y. (1994a). Determining the Number of Kanbans and Lotsizes in a Generic Kanban System: A Simulated Annealing Approach. *Int. J. Prod. Res.* **32**, 1991-2004.

Chang, T.-M. and Yih, Y. (1994b). Generic Kanban Systems for Dynamic Environments. *Int. J. Prod. Res.* **32**, 889-902.

Cheng, D. W. (1995). Line Reversibility of Tandem Queues with General Blocking. *Management Science*, **41**, 864- 881.

Cheng, D. W. (1993). Concavity of System Throughput and Information Transfer in a PAC System. *Operations Research Letters* **14**, 143-146.

Cheng, D. W. (1992). Second Order Properties in a Tandem Queue with General Blocking. *Operations Research Letters* **12**, 139-144.

Cheng, D. W. (1991). *Tandem Queues with General Blocking: Stochastic Comparisons and Structural Properties.* Ph.D. thesis. Graduate School of Business, Colombia University, New York.

Cheng, D. W. and Yao, D. D. (1993). Tandem Queues with General Blocking: A Unified Model and Comparison Results. *Discrete Events Dynamic Systems: Theory and Applications* **2**, 207-234.

Cheng, D. W. and Zhu, Y. (1993). Optimal Order of Servers in a Tandem Queue with General Blocking. *Queueing Systems* **14**, 427-437.

Cho, K. K., Kim, K. H. and Kim, C. S. (1994). A Heuristic Lot Sizing Algorithm for a GT Cell. *Computers and Industrial Engineering* **26**, 1-9.

Clark, A. J. and Scarf, H. (1960). Optimal Policies for a Multie-echelon Inventory Problem. *Management Science* **6**, 475-490.

Co, H. C. and Jacobson, S. H. (1994). The Kanban Assignment Problem in Serial Just-in-Time Production Systems. *IIE Transactions* **26**, 76-85.

Conway, R., Maxwell, W., McClain, J. O. and Thomas, L. J. (1988). The Role of Work-in-Process Inventory in Serial Production Lines. *Operations Research* **36**, 229-241.

Davis, R. P. and Kennedy, JR, W. J. (1987). Markovian Modelling of Manufacturing Systems. *Int. J. Prod. Res.* **25**, 337-351.

Davis, W. J. and Stubitz, S. J. (1987). Configuring a Kanban System Using a Discrete Optimization of Multiple Stochastic Responses. *Int. J. Prod. Res.* **25**, 721-740.

De Koster, M. B. M. (1988). Approximation of Flow Lines with Integrally Controlled Buffers. *IIE Transactions* **20**, 374-381.

De Koster, R. and Wijngaard, J. (1989). Local and Integral Control of Workload. *Int. J. Prod. Res.* **26**, 43-52.

Deleersnyder, J. L., Hodgson, T. J., King, R. E., O'Grady, P. J. and Savva, A. (1992). Integrating Kanban Type Pull Systems and MRP Type Systems: Insights from a Markovian Model. *IIE Transactions* **24**, 43-56.

Deleersnyder, J. L., Hodgson, T. J., Muller, H. and O'Grady, P. J. (1989). Kanban Controlled Pull Systems: An Analytic Approach. *Management Science* **35**, 1079-1091.

Dixon, L. C. W. (1972). *Nonlinear Optimisation*. The English Universities Press Ltd., London.

Duenyas, I., Hopp, W. J. and Spearman, M. L. (1993). Characterizing the Output Process of a CONWIP Line with Deterministic Processing and Random Outages. *Management Science* **39**, 975-988.

Eiselt, H. A., Pederzoli, G., and Sandblom, C.-L. (1987). *Continuous Optimization Models*. Walter de Gruyter. Berlin

Foster, III, J. W. and Garcia-Diaz, A. (1983). Markovian Models for Investigating Failure and Repair Characteristics of Production Systems. *IIE Transactions* **15**, 202-209.

Fukukawa, T. and Hong, S. C. (1993). The Determination of the Optimal Number of Kanbans in a Just-in-Time Production System. *Computers and Industrial Engineering* **24**, 551-559.

Gascon, A., Leachman, R. C. and Lefrancois, P. (1994). Multi-Item, Single-Machine Scheduling Problem with Stochastic Demands: A Comparison of Heuristics. *Int. J. Prod. Res.* **32**, 583-596.

Gershwin, S. B. (1994). *Manufacturing Systems Engineering*. Prentice Hall, Enlewood Cliffs, NJ.

Gershwin, S. B. (1987). An Efficient Decomposition Method for the Approximate Evaluation of Tandem Queues with Finite Storage Space and Blocking. *Operations Research* **35**, 291-305.

Gershwin, S. B. and Berman, O. (1981). Analysis of Transfer Lines Consisting of Two Unreliable Machines with Random Processing Times and Finite Storage Buffers. *AIIE Transactions* **13**, 2-11.

Gershwin, S. B. and Schick, I. C. (1983). Modeling and Analysis of Three-Stage Transfer Lines with Unreliable Machines and Finite Buffers. *Operations Research* **31**, 354-380.

Gill, P. E., Murray, W. and Wright, M. H. (1981). *Practical Optimization*. Academic Press Inc., London

Glasserman, P. (1991). Structural Conditions for Perturbation Analysis Derivative Estimation: Finite-Time Performance Indices. *Operations Research* **39**, 724-738.

Glasserman, P. and Tayur, S. (1995). Sensitivity Analysis for Base-Stock Levels in Multiechelon Production-Inventory Systems. *Management Science* **41**, 263-281.

Glasserman, P. and Yao, D. D. (1994a). Monotone Optimal Control of Permutable GSMP. *Mathematics of Operations Research* **19**, 449-476.

Glasserman, P. and Yao, D. D. (1994b). A GSMP Framework for the Analysis of Production Lines. *Stochastic Modeling and Analysis of Manufacturing Systems*. Springer-Verlag.

Glasserman, P. and Yao, D. D. (1992a). Generalized Semi-Markov Processes: Antimatroid Structure and Second-Order Properties. *Mathematics of Operations Research* **17**, 444-469.

Glasserman, P. and Yao, D. D. (1992b). Monotonicity in Generalized Semi-Markov Processes. *Mathematics of Operations Research* **17**, 1-21.

Glasserman, P. and Yao, D. D. (1992c). Structured Buffer-Allocation Problems. (copy received from D. D. Yao).

Glassey, C. R. and Resende, M. G. C. (1988). Closed-Loop Job Release Control for VLSI Circuit Manufacturing. *IEEE Transactions on Semiconductor Manufacturing* **1**, 36-46.

Glover, F and Laguna, M. (1992). Tabu Search. A chapter in *Modern Heuristic Techniques for Combinatorial Problems.*

Goldratt, E. M., and Cox, J. (1986). *The Goal.* North River Press Inc., New York. Second Edition

Graham, I. (1992). Comparing Trigger and Kanban Control of Flow-Line Manufacture. *Int. J. Prod. Res.* **30**, 2351-2362.

Gravel, M., Kiss, L., Martel, J. M. and Price, W. L. (1994). A DSS for Production Planning. *Int. Trns. in Operational Research* **3**, 363-373.

Gravel, M. and Price, W. L. (1991). Visual Interactive Simulation Shows How to Use the Kanban Method in Small Business. *Interfaces* **21**, 22-33.

Gravel, M. and Price, W. L. (1988). Using the Kanban in a Job Shop Environment. *Int. J. Prod. Res.* **26**, 1105-1118.

Groenevelt, H. (1993). The Just-in-Time System, Chapter 12 of *Handbook in Operations Research and Management Science.* S. C. Graves et al., Editors. Elsevier Science Publishers B.V., Amsterdam.

Gupta, Y. P. and Gupta, M. C. (1989). A System Dynamics Model for a Multi-Stage Multi-Line Dual-Card JIT-Kanban System. *Int. J. Prod. Res.* **27**, 309-352.

Gursoy, M. B., Altiok, T. and Danhong, H. (1994). Control Policies for Two-Stage, Pull-Type Production/Inventory Systems with Constrained Average Cost Criterion. *Int. J. Prod. Res.* **32**, 2005-2014.

Harel, A. and Zipkin, P. H. (1987). Strong Convexity Results for Queueing Systems. *Operations Research* **35**, 405-418.

Hicks, C. R. (1982). *Fundamental Concepts in the Design of Experiments*. Holt, Rinehart and Winston, Inc. NY. Third Edition.

Ho, Y. C., Eyler, M. A. and Chien, T. T. (1983). A New Approach to Determine Parameter Sensitivities of Transfer Lines. *Management Science* **29**, 700-714.

Hodgson, T. J., King, R. E., O'Grady, P. J. and Savva, A. (1992). Integrating Kanban Type Pull Systems and MRP Type Push Systems: Insights from a Markovian Model. *IIE Transactions* **24**, 43-56.

Hong, J. D. and Hayya, J. C. (1993). Dynamic Lot Sizing with Setup Reduction. *Computers and Industrial Engineering* **24**, 209-218.

Hooke, R. and Jeeves, T. A. (1961). Direct Search Solution of Numerical and Statistical Problems. *J. Association Computer Machinery* **8**, 212-229.

Hopp, W. J. and Spearman, M. L. (1991). Throughput of a Constant Work in Process Manufacturing Line Subject to Failures. *Int. J. Prod. Res.* **29**, 635-655.

Horst, R. and Pardalos, P. M. (1995). *Handbook of Global Optimization*. Kluwer Academic Publishers, Dordrecht, The Netherlands.

Huang, P. Y., Rees, L. P. and Taylor, III, B. W. (1983). A Simulation Analysis of the Japanese Just-in-Time Technique (with Kanbans) for a Multiline, Multistage Production System. *Decision Sciences* **14**, 326-344.

Jaakola, T. H. I., and Luus, R. (1974). A Note on the Application of Nonlinear Programming to Chemical-Process Optimization. *Operations Research* **22**, 415-417.

Jacoby, S. L. S., Kowalik, J. S. and Pizzo, J. T. (1972). *Iterative Methods for Nonlinear Optimization Problems*. Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

Kamrani, A. K. and Parsaei, H. R. (1992). A Methodology for Forming Manufacturing Cells Using Manufacturing and Design Attributes. *Computers and Industrial Engineering* **23**, 73-76.

Karmarkar, U. S. (1994). *New Directions in Modelling Production and Distribution Problems*, Working Paper CMOM 94-01, Graduate School of Business Administration, The University of Rochester.

Karmarkar, U. S. (1993). Manufacturing Lead Times, Order Release and Capacity Loading, Chapter 6 of *Handbook in Operations Research and Management Science*. S. C. Graves et al., Editors. Elsevier Science Publishers B.V., Amsterdam.

Karmarkar, U. S. (1987). Lot Sizes, Lead Times and In-Process Inventories. *Management Science* **33**, 409-418.

Karmarkar, U. S. (1986). *Integrating MRP with Kanban Pull Systems*, Working Paper QM86-06, Graduate School of Management, The University of Rochester.

Karmarkar, U. S. and Kekre, S. (1990). *Capacity and Equipment Choice in a Multiproduct Manufacturing Cell*, Working Paper QM 90-04, Graduate School of Business Administration, The University of Rochester.

Karmarkar, U. S. and Kekre, S. (1987). *Batching Policy in Kanban Systems*, Technical Report QM87-06, Graduate School of Management, The University of Rochester.

Karmarkar, U. S., Kekre, S. and Kekre, S. (1985). Lotsizing in Multi-Item Multi-Machine Job Shops. *IIE Transactions* **17**, 290-298.

Karmarkar, U. S., Kekre, S., Kekre, S. and Freeman, S. (1985). Lot-Sizing and Lead-Time Performance in a Manufacturing Cell. *Interfaces* **15**, 1-9.

Kekre, S. (1987). Performance of a Manufacturing Cell with Increased Product Mix. *IIE Transactions* **19**, 329-339.

Kim, T. M. (1985). Just-in-Time Manufacturing System: A Periodic Pull System. *Int. J. Prod. Res.* **23**, 553-562.

Kimura, O. and Terada, H. (1981). Design and Analysis of Pull System, A Method of Multi-Stage Production Control. *Int. J. Prod. Res.* **19**, 241-253.

Kirkpatrick, S., Gelatt, C. D. and Vecchi, M. P. (1983). Optimization by Simulated Annealing. *Science* **220**, 671-680.

Kleijnen, J. P. C. and Van Groenendaal, W. (1992). *Simulation, A Statistical Perspective.* John Wiley & Sons, NY.

Krajewski, L. J., King, B. E., Ritzman, L. P. and Wong, D. S. (1987). Kanban, MRP, and Shaping the Manufacturing Environment. *Management Science* **33**, 39-57.

Lagodimos, A. G. (1993). Models for Evaluating the Performance of Serial and Assembly MRP Systems. *European Journal of Operational Research* **68**, 49-68.

Lambrecht, M. R., Muckstadt, J. A. and Luyten, R. (1984). Protective Stocks in Multi-Stage Production Systems. *Int. J. Prod. Res.* **22**, 1001-1025.

Law, A. M. and Carson, J. S. (1979). A Sequential Procedure for Determining the Length of a Steady-State Simulation. *Operations Research* **27**, 1011-1025.

Law, A. M. and Kelton, W. D. (1991). *Simulation Modeling and Analysis*. McGraw-Hill, Inc., Second Edition.

Law, A. M. and Kelton, W. D. (1984). Confidence Intervals for Steady-State Simulations: I. A Survey of Fixed Sample Size Procedures. *Operations Research* **32**, 1221-1239.

Law, A. M. and Kelton, W. D. (1982). Confidence Intervals for Steady-State Simulations: II. A Survey of Sequential Procedures. *Management Science* **28**, 550-562.

L'Ecuyer, P. and Glynn, P. W. (1994). Stochastic Optimization by Simulation: Convergence Proofs for the GI/G/1 Queue in Steady-State. *Management Science* **40**, 1562-1578.

Lee, Y. J. and Zipkin, P. (1992). Tandem Queues with Planned Inventories. *Operations Research* **40**, 936-947.

Li, A. and Co, H. C. (1991). A Dynamic Programming Model for the Kanban Assignment Problem in a Multistage Multiperiod Production System. *Int. J. Prod. Res.* **29**, 1-16.

Lin, L. and Chiu, F. (1993). Manufacturing Cell Operating Characteristics. *European Journal of Operational Research* **69**, 424-437.

Liu, S. C. and Lin, L. (1993). Dynamic Sequencing of Robot Moves in a Manufacturing Cell. *European Journal of Operational Research* **69**, 482-497.

Lucantoni, D.M. (1993). The BMAP/G/1 Queue: A Tutorial. *Models and Techniques for Performance Evaluation of Computer and Communication Systems*. L. Donatiello and R. Nelson Editors: Springer Verlage.

Luenberger, D. G. (1989). *Linear and Nonlinear Programming*. Addison-Wesley Publishing Company, Inc., Second Edition.

Lundrigan, R. (1986). What is Thing Called OPT? *Production and Inventory Management*, 2-12.

Luus, R. and Jaakola, T. H. I. (1973). Optimization by Direct Search and Systematic Reduction of the Size of Search Region. *AiChE Journal* **19**, 760-766.

Maxwell, W., Muckstadt, J. A., Thomas, L. J. and VanDerEecken, J. (1983). A Modeling Framework for Planning and Control of Production in Discrete Parts Manufacturing and Assembly Systems. *Interfaces* **13**, 92-104.

Melamed, B. (1986). A Note on the Reversibility and Duality of Some Tandem Blocking Queueing Systems. *Management Science* **32**, 1648-1650.

McClain, J. O. and Moodie, D. R. (1991). A Comment on Buffer Space Allocation in Automated Assembly Lines. *Operations Research* **39**, 857-860.

Mitra, D. and Mitrani, I. (1991). Analysis of a Kanban Discipline for Cell Coordination in Production Lines, II: Stochastic Demands. *Operations Research* **39**, 807-823.

Mitra, D. and Mitrani, I. (1990). Analysis of a Kanban Discipline for Cell Coordination in Production Lines. *Management Science* **36**, 1548-1566.

Mitwasi, M. G. and Askin, R. G. (1994). Production Planning for a Multi-Item, Single-Stage Kanban System. *Int. J. Prod. Res.* **32**, 1172-1195.

Montgomery, D. C. (1991). *Design and Analysis of Experiments*. John Wiley & Sons, NY. Third Edition.

Montreuil, B. and Lefrancois, P. (1996). Organizing Factories as Responsibility Networks, in *Recent Advances in Material Handling Research*. Springer-Verlag.

Muckstadt, J. A. and Tayur, S. R. (1995). A Comparison of Alternative Kamban Control Mechanisms. I. Background and Structural Results. *IIE Transactions* **27**, 140-150.

Muckstadt, J. A. and Tayur, S. R. (1995). A Comparison of Alternative Kamban Control Mechanisms. II. Experimental Results. *IIE Transactions* **27**, 151-161.

Narasimhan, R. and Melnyk, S. A. (1984). Assessing the Transient Impact of Lot Sizing Rules Following MRP Implementation. *Int. J. Prod. Res.* **22**, 759-772.

Nash, S. G. and Sofer, A. (1996). *Linear and Nonlinear Programming*. McGraw-Hill Companies, Inc., New York.

Nelder, J. A. and Mead, R. (1965). A Simplex Method for Function Minimazation-Errata. *Computer Journal* **8**, 27.

Nelder, J. A. and Mead, R. (1964). A Simplex Method for Function Minimazation. *Computer Journal* **7**, 308-313.

Nemhauser, G. L., Rinnooy Kan, A. H. G. and Todd, M. J. (1989). *Handbooks in operations Research and Management Science, Volume I, Optimization*. Elsevier Science Publishers B.V., Amsterdam, The Netherlands.

Nunen, J. A. and Puterman, M. (1983). Computing Optimal Control Limits for GI/M/1 Queueing Systems with Controlled Customer Arrivals. *Management Science* **29**, 725-734.

Ohmi, T. (1981). An Approximation for the Production Efficiency of Automatic Transfer Lines with In-Process Storages. *AIIE Transactions* **13**, 22-28.

Orlicky, J. (1975). *Material Requirements Planning*. McGraw-Hill, New York.

Ou, J. and Wein, L. M. (1995). Dynamic Scheduling of a Production/Inventory System with By-Products and Random Yield. *Management Science* **41**, 1000-1017.

Patel, N. R., Smith, R. L. and Zabinsky, Z. B. (1988) Pure Adaptive Search in Monte Carlo Optimization. *Mathematical Programming* **43**, 317-328.

Pike, R. W. (1986). *Optimization for Engineering Systems*. Van Nostrand Reinhold Company, New York.

Philipoom, P. R., Rees, L. P., Taylor, III, B. W. and Huang, P. Y. (1987). An Investigation of the factors Influencing the Number of Kanbans Required in the Implementation of the JIT Technique with Kanbans. *Int. J. Prod. Res.* **25**, 457-472.

Plenert, G. and Best, T. D. (1986). MRP, JIT, and OPT: What's Best? *Production and Inventory Management*, 22-29.

Ravindranath, L. S. and Ebeling, K. A. (1992). Distributed Control of Manufacturing Cells. *Computers and Industrial Engineering* **23**, 1-4.

Rees, L. P., Huang, P. Y. and Taylor, III, B. W. (1989). A Comparative Analysis of an MRP Lot-for-lot System and a Kanban System for a Multistage Production Operation. *Int. J. Prod. Res.* **27**, 1427-1443.

Rees, L. P., Philipoom, P. R., Taylor, III, B. W. and Huang, P. Y. (1987). Dynamically Adjusting the Number of Kanbans in a Just-in-Time Production System Using Estimated Values of Leadtime. *IIE Transactions* **19**, 199-207.

Roderick, L. M., Toland, J. and Rodriguez, F. P. (1994). A Simulation Study of CONWIP versus MRP at Westinghouse. *Computers and Industrial Engineering* **26**, 237-242.

Rosa-Hatko, W. and Gunn, E. A. (1997). Queues with Switchover - A Review and Critique. *Annals of Operations Research* **69**, 299-322.

Rosa-Hatko, W. and Gunn, E. A. (1991). Some Models of Queueing Control with Switchover. *Computers Math. Applic.* **21**, 91-110.

Ross, S. M. (1985). *Introduction to Probability Models*. Academic Press, Florida.

Sahin, I. (1979). On the Stationary Analysis of Continuous Review (s,S) Inventory Systems with Constant Lead Times. *Operations Research* **27**, 717-729.

Sarker, B. R. (1989). Simulating a Just-in-Time Production System. *Computers and Industrial Engineering* **16**, 127-137.

Sarker, B. R. and Fitzsimmons, J. A. (1989). The Performance of Push and Pull Systems: A Simulation and Comparative Study. *Int. J. Prod. Res.* **27**, 1715-1731.

Sarker, B. R. and Harris, R. D. (1988). The Effect of Imbalance in a Just-in-Time Production System: A Simulation Study. *Int. J. Prod. Res.* **26**, 1-18.

Schonberger, R. J. (1983). Applications of Single-Card and Dual-Card Kanban. *Interfaces* **13**, 56-67.

Seidmann, A. (1988). Regenerative Pull (Kanban) Production Control Policies. *European Journal of Operational Research* **35**, 401-413.

Seidmann, A. and Schweitzer, P. J. (1984). Part Selection Policy for Flexible Manufacturing Cell Feeding Several Production Lines. *IIE Transactions* **16**, 355-362.

Shanthikumar, J. G. and Buzacott, J. A. (1981). Open Queueing Network Models of Dynamic Job Shops. *Int. J. Prod. Res.* **19**, 255-266.

Shanthikumar, J. G. and Sargent, R. G. (1983). A Unifying View of Hybrid Simulation/Analytic Models and Modeling. *Operations Research* **31**, 1030-1052.

Shanthikumar, J. G., Yamazaki, G. and Sakasegawa, H. (1991). Characterization of Optimal Order of Servers in a Tandem Queue with Blocking. *Operations Research Letters* **10**, 17-22.

Siddal, J. N. (1972). *Analytical Decision-Making in Engineering Design.* Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

Siha, S. (1994). The Pull Production System: Modelling and Characteristics. *Int. J. Prod. Res.* **32**, 933-949.

Silver, E.A. and Peterson, R. (1985). *Decision Systems for Inventory Management and Production Planning.* John Wiley & Sons, Inc., New York. Second Edition.

Simon, J. T., and Hopp, W. J. (1995). Throughput and Average Inventory in Discrete Balanced Assembly Systems. *IIE Transactions* **27**, 368-373.

Sinha, R. K. and Hollier, R. H. (1984). A Review of Production Control Problems in Cellular Manufacture. *Int. J. Prod. Res.* **22**, 773-789.

Smith, J. M. and Daskalaki, S. (1988). Buffer Space Allocation in Automated Assembly Lines. *Operations Research* **36**, 343-358.

So, K. C. and Pinault, S. C. (1988). Allocating Buffer Storages in a Pull System. *Int. J. Prod. Res.* **26**, 1959-1980.

So, K. C., and Tang, C. S. (1995). Optimal Batch Sizing and Repair Strategies for Operations with Repairable Jobs. *Management Science* **41**, 894-911.

So, K. C., and Tang, C. S. (1995). Optimal Operating Policy for a Bottleneck with Random Rework. *Management Science* **41**, 620-636.

Spearman, M. L., Woodruff, D. L. and Hopp, W. J. (1990). CONWIP: A Pull Alternative to Kanban. *Int. J. Prod. Res.* **28**, 879-894.

Spearman, M. L. and Zazanis, M. A. (1992). Push and Pull Production Systems: Issues and Comparisons. *Operations Research* **40**, 521-532.

Spendley, W., Hext, G. R. and Himsworth, F. R. (1962). Sequencial Application of Simplex Designs of Optimization and Evolutionary Operations. *Technometrics* **4**, 441-461.

Stecke, K. E. and Solberg, J. J. (1985). The Optimality of Unbalancing Both Workloads and Machine Group Sizes in Closed Queueing Networks of Multiserver Queues. *Operations Research* **33**, 882-910.

Sugimori, Y., Kusunoki, K., Cho, F. and Uchikawa, S. (1977). Toyota Production System and Kanban System Materialization of Just-in-Time and Respect-For-Human System. *Int. J. Prod. Res.* **15**, 553-564.

Suri, R. and Diehl, G. W. (1986). A Variable Buffer-Size Model and Its Use in Analyzing Closed Queueing Networks with Blocking. *Management Science* **32**, 206-224.

Takahashi, K., Muramatsu, R. and Ishii, K. (1987). Feedback Method of Production Ordering System in Multi-Stage Production and Inventory Systems. *Int. J. Prod. Res.* **25**, 925-941.

Tayur, A. R. (1993). Structural Properties and a Heuristic for Kanban-Controlled Serial Lines. *Management Science* **39**, 1347-1368.

Tsiotras, G. D., Badr, H. G. and Beltrami, E. (1991). Blocking Probabilities for a Class of Two-Station Tandem Queueing Models. *IIE Transactions* **23**, 331-345.

Van Ryzin, G., Lou, X.-C. and Gershwin, S. B. (1993). Production Control for a Tandem Two-Machine System. *IIE Transactions* **25**, 5-20.

Veatch, M. H. and Wein, L. M. (1994). Optimal Control of a Two-Station Tandem Production/Inventory System. *Operations Research* **42**, 337-350.

Veatch, M. H. and Wein, L. M. (1991). Optimal Control of Multistage Production/ Inventory System, paper presented at the conference on Multi-echelon Inventory Systems at Berkeley, CA.

Vollmann, T. E., Berry, W. L. and Whybark, D. C. (1992). *Manufacturing Planning and Control Systems.* Richard D. Irvin., Inc., Boston, MA. Third Edition.

Wang, H. and Wang, H.-P. (1990). Determining the Number of Kanbans: A Step Toward Non-Stock-Production. *Int. J. Prod. Res.* **28**, 2101-2115.

Weber, R. R. (1983). A Note on Waiting Times in Single Server Queues. *Operations Research* **31**, 950-951.

Wemmerlov, U. and Hyer, N. L. (1989). Group Technology in the US Manufacturing Industry: A Survey of Current Practice. *Int. J. Prod. Res.* **27**, 1287-1304.

Wemmerlov, U. and Hyer, N. L. (1987). Research Issues in Cellular Manufacturing. *Int. J. Prod. Res.* **25**, 413-431.

Whitt, W. (1982). Approximating a Point Process by a Renewal Process, I: Two Basic Methods. *Operations Research* **30**, 125-147.

Wight, O. (1984). *Manufacturing Resource Planning: MRPII*. Oliver Wight Ltd., Essex Junction, Vt.

Wijngaard, J. (1979). The Effects of Interstage Buffer Storage on the Output of Two Unreliable Production Units in Series, with Different Production Rates. *AIIE Transactions* **11**, 42-47.

Wolff, R. W. (1982). Poisson Arrivals See Time Averages. *Operations Research* **30**, 223-231.

Yamazaki, G., Sakasegawa, H. and Shanthikumar, J. G. (1992). On Optimal Arrangement of Stations in a Tandem Queueing System with Blocking. *Management Science* **38**, 137-153.

Yao, D. D. and Buzacott, J. A. (1987). Modeling a Class of Flexible Manufacturing Systems with Reversible Routing. *Operations Research* **35**, 87-93.

Zipkin, P. H. (1995). Performance Analysis of a Multi-Item Production-Inventory System Under Alternative Policies. *Management Science* **41**, 690-703.

Zipkin, P. H. (1991). Computing Optimal Lot Sizes in the Economic Lot Scheduling Problem. *Operations Research* **39**, 56-63.

Zipkin, P. H. (1986). Models for Design and Control of Stochastic, Multi-Item Batch Production Systems. *Operations Research* **34**, 91-104.

Appendix A1

FORTRAN CODE FOR GENERAL VERSION OF SIMLIB - SIMLIBG

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C**********************************************************************C
C*     PARAM.DCL    : Parameters setting for SIMLIBG - for PAC Model      *C
C*     Author       : Krystyna Bielunska                                  *C
C*     Date         : June 1995              (example-model: model 4)*C
C**********************************************************************C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      INTEGER MA,MR,ML,ST

*     Set values for SIMLIB parameters:
*     MA        - maximum number of attributes for any record in any list
*     MR        - maximum number of all records in array MASTER
*     ML        - maximum number of different lists
*                 and the number of the event list itself
*     ST        - maximum number of SAMPST or TIMEST variables
*     MT        - ML+ST

      PARAMETER(MA=4,MR=35000,ML=84,ST=23,MT=107)


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C**********************************************************************C
C*     SIMLIBG.FOR  : Set of routines of SIMLIB - General Version         *C
C*     Author       : Krystyna Bielunska                                  *C
C*     Date         : June 1995              (RISC version)              *C
C**********************************************************************C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE INITLK
      INCLUDE 'param.dcl'

      INTEGER HEAD(ML),IOUT,LINKPR(MR),LINKSR(MR),LIST,NAR,ROW,
     &     TAIL(ML)
      REAL MASTER(MR,MA)
      COMMON /LLISTS/ HEAD,IOUT,LINKPR,LINKSR,MASTER,NAR,TAIL
      INTEGER LDECR,LEVENT,LFIRST,LINCR,LLAST,LRANK(ML),LSIZE(ML),
     &     MAXATR,NEXT
      REAL TIME,TRNSFR(MA)
      COMMON /SIMLIB/ LDECR,LEVENT,LFIRST,LINCR,LLAST,LRANK,LSIZE,
     &            MAXATR,NEXT,TIME,TRNSFR
```

```
*    Ensure integrity of variables in COMMON block LLISTS, which
*    appears for the first time in a subprogram.

      SAVE

*    Initialize links.

      DO 10 ROW = 1, MR
        LINKPR(ROW) = 0
   10 LINKSR(ROW) = ROW + 1
      LINKSR(MR) = 0

*    Initialize list attributes.

      DO 20 LIST = 1, ML
        HEAD(LIST)  = 0
        TAIL(LIST)  = 0
        LSIZE(LIST) = 0
   20 LRANK(LIST) = 0

*    Initialize mnemonics for record location in lists.

      LFIRST = 1
      LLAST  = 2
      LINCR  = 3
      LDECR  = 4

*    Initialize mnemonic for event list number.

      LEVENT = ML

*    Initialize system attributes.

      TIME        = 0.0
      NAR         = 1
      LRANK(LEVENT) = 1
      MAXATR      = MA

*    Initialize statistical routines.

      CALL SAMPST(0.0,0)
      CALL TIMEST(0.0,0)

*    Initialize output unit number for SIMLIB error messages.

      IOUT = 6

      RETURN
      END
```

```
      SUBROUTINE FILE(OPTION,LIST)
      INCLUDE 'param.dcl'

      INTEGER AHEAD,BEHIND,HEAD(ML),IHEAD,IOUT,ITAIL,ITEM,LINKPR(MR),
     &        LINKSR(MR),LIST,NAR,OPTION,ROW,TAIL(ML)
      REAL MASTER(MR,MA)
      COMMON /LLISTS/ HEAD,IOUT,LINKPR,LINKSR,MASTER,NAR,TAIL

      INTEGER LDECR,LEVENT,LFIRST,LINCR,LLAST,LRANK(ML),LSIZE(ML),
     &        MAXATR,NEXT
      REAL TIME,TRNSFR(MA)
      COMMON /SIMLIB/ LDECR,LEVENT,LFIRST,LINCR,LLAST,LRANK,LSIZE,
     &               MAXATR,NEXT,TIME,TRNSFR

*     If the master storage array is full, stop the simulation.

      IF (NAR .EQ. 0) THEN
        WRITE (IOUT,10) TIME
   10   FORMAT (' From SIMLIB: MASTER storage array overflow at time',
     &          F10.3)
        STOP
      END IF

*     If the list value is improper, stop the simulation.

      IF (LIST .LT. 1 .OR. LIST .GT. ML) THEN
        WRITE (IOUT,20) LIST, TIME
   20   FORMAT (' From SIMLIB:',I10,
     &          ' is an improper value for FILE LIST at time ',F10.3)
        STOP
      END IF

*     Increment the list size.

      LSIZE(LIST) = LSIZE(LIST) + 1

*     If the option value is improper, stop the simulation.

      IF (OPTION .LT. 1 .OR. OPTION .GT. 4) THEN
        WRITE (IOUT,30) OPTION, TIME
   30   FORMAT (' From SIMLIB:',I10,
     &          ' is an improper value for FILE OPTION at time ',F10.3)
        STOP
      END IF

*     File according to the desired option.

      GO TO (300, 200, 100, 100), OPTION

*     List is ranked. Determine item on which list is to be ranked.
```

```
 100 ITEM = LRANK(LIST)

*    If an invalid item has been specified, stop the simulation.
     IF (ITEM .LT. 1 .OR. ITEM .GT. MAXATR) THEN
       WRITE (IOUT,110) ITEM, LIST
 110    FORMAT (' From SIMLIB:',I10,
   &          ' is an improper value for ranking attribute of list',
   &          I3)
       STOP
     END IF

*    If this is not the first record in this list, continue.

     IF (LSIZE(LIST) .EQ. 1) GO TO 400

*    Search the list for the proper location.

     ROW = HEAD(LIST)
 120 IF (OPTION .EQ. 4) GO TO 130

*    Rank the list in increasing order.

     IF (TRNSFR(ITEM) .GE. MASTER(ROW,ITEM)) GO TO 150

*    The correct location has been found.

     GO TO 140

*    Rank the list in decreasing order.

 130 IF (TRNSFR(ITEM) .LE. MASTER(ROW,ITEM)) GO TO 150

*    Correct location found. Insert before last record examined.

 140 IF (ROW .EQ. HEAD(LIST)) GO TO 300

*    Insert in proper location between preceding, succeeding records.

     AHEAD = LINKSR(BEHIND)
     ROW   = NAR
     NAR   = LINKSR(ROW)
     IF (NAR .GT. 0) LINKPR(NAR) = 0
     LINKPR(ROW)    = BEHIND
     LINKSR(BEHIND) = ROW
     LINKPR(AHEAD)  = ROW
     LINKSR(ROW)    = AHEAD

*    Go to transfer the data.

     GO TO 500
```

```
*   Continue searching, consider the next row.

 150 BEHIND = ROW
   ROW    = LINKSR(BEHIND)

*   If last row considered was not the tail of the list, continue.
    IF (TAIL(LIST) .NE. BEHIND) GO TO 120

*   Insert after the last record in the list.

 200 IF (LSIZE(LIST) .EQ. 1) GO TO 400
   ROW = NAR
   NAR = LINKSR(ROW)
   IF (NAR .GT. 0) LINKPR(NAR) = 0
   ITAIL        = TAIL(LIST)
   LINKPR(ROW)  = ITAIL
   LINKSR(ITAIL) = ROW
   LINKSR(ROW)  = 0
   TAIL(LIST)   = ROW

*   Go to transfer the data.

   GO TO 500

*   Insert before the first record in the list.

 300 IF (LSIZE(LIST) .EQ. 1) GO TO 400
   ROW = NAR
   NAR = LINKSR(ROW)
   IF (NAR .GT. 0) LINKPR(NAR) = 0
   IHEAD        = HEAD(LIST)
   LINKPR(IHEAD) = ROW
   LINKSR(ROW)  = IHEAD
   LINKPR(ROW)  = 0
   HEAD(LIST)   = ROW

*   Go to transfer the data.

   GO TO 500

*   Insert the first record in the list.

 400 ROW = NAR
   NAR = LINKSR(ROW)
   IF (NAR .GT. 0) LINKPR(NAR) = 0
   LINKSR(ROW) = 0
   HEAD(LIST)  = ROW
   TAIL(LIST)  = ROW

*   Transfer the data.
```

```
 500 DO 510 ITEM = 1, MAXATR
 510 MASTER(ROW,ITEM) = TRNSFR(ITEM)

*    Update the area under the number-in-list curve.

      CALL TIMEST(FLOAT(LSIZE(LIST)),ST + LIST)

      RETURN
      END


      SUBROUTINE REMOVE(OPTION,LIST)
      INCLUDE 'param.dcl'

      INTEGER HEAD(ML),IHEAD,IOUT,ITAIL,ITEM,LINKPR(MR),LINKSR(MR),
     &        LIST,NAR,OPTION,ROW,TAIL(ML)
      REAL MASTER(MR,MA)
      COMMON /LLISTS/ HEAD,IOUT,LINKPR,LINKSR,MASTER,NAR,TAIL

      INTEGER LDECR,LEVENT,LFIRST,LINCR,LLAST,LRANK(ML),LSIZE(ML),
     &        MAXATR,NEXT
      REAL TIME,TRNSFR(MA)
      COMMON /SIMLIB/ LDECR,LEVENT,LFIRST,LINCR,LLAST,LRANK,LSIZE,
     &               MAXATR,NEXT,TIME,TRNSFR

*    If the list value is improper, stop the simulation.

      IF (LIST .LT. 1 .OR. LIST .GT. ML) THEN
        WRITE (IOUT,10) LIST, TIME
   10   FORMAT (' From SIMLIB:',I10,
     &          ' is an improper value for REMOVE LIST at time ',F10.3)
        STOP
      END IF

*    If the list is empty, stop the simulation.

      IF (LSIZE(LIST) .LE. 0) THEN
        WRITE (IOUT,20) LIST, TIME
   20   FORMAT (' From SIMLIB: underflow of list ',I2,
     &          ' at time ',F10.3)
        STOP
      END IF

*    Decrement the list size.

      LSIZE(LIST) = LSIZE(LIST) - 1

*    If the option value is improper, stop the simulation.

      IF (OPTION .NE. 1 .AND. OPTION .NE. 2) THEN
        WRITE (IOUT,30) OPTION, TIME
```

```
30   FORMAT (' From SIMLIB:',I10,
  &         ' is improper value for REMOVE OPTION at time ',F10.3)
     STOP
   END IF

*   If there is more than one record in the list, continue.

    IF (LSIZE(LIST) .EQ. 0) GO TO 300

*   Remove according to the desired option.

    GO TO (100, 200), OPTION

*   Remove the first record in the list.
 100 ROW          = HEAD(LIST)
    IHEAD        = LINKSR(ROW)
    LINKPR(IHEAD) = 0
    HEAD(LIST)   = IHEAD

*   Go to transfer the data.

    GO TO 400

*   Remove the last record in the list.

 200 ROW          = TAIL(LIST)
    ITAIL        = LINKPR(ROW)
    LINKSR(ITAIL) = 0
    TAIL(LIST)   = ITAIL

*   Go to transfer the data.

    GO TO 400

*   Remove the only record in the list.

 300 ROW        = HEAD(LIST)
    HEAD(LIST) = 0
    TAIL(LIST) = 0

*   Transfer the data.

 400 LINKSR(ROW) = NAR
    LINKPR(ROW) = 0
    NAR        = ROW
    DO 410 ITEM = 1, MAXATR
 410 TRNSFR(ITEM) = MASTER(ROW,ITEM)

*   Update the area under the number-in-list curve.

    CALL TIMEST(FLOAT(LSIZE(LIST)),ST + LIST)
```

```
      RETURN
      END


      SUBROUTINE TIMING
      INCLUDE 'param.dcl'

      INTEGER HEAD(ML),IOUT,LINKPR(MR),LINKSR(MR),NAR,TAIL(ML)
      REAL MASTER(MR,MA)
      COMMON /LLISTS/ HEAD,IOUT,LINKPR,LINKSR,MASTER,NAR,TAIL

      INTEGER LDECR,LEVENT,LFIRST,LINCR,LLAST,LRANK(ML),LSIZE(ML),
     &     MAXATR,NEXT
      REAL TIME,TRNSFR(MA)
      COMMON /SIMLIB/ LDECR,LEVENT,LFIRST,LINCR,LLAST,LRANK,LSIZE,
     &          MAXATR,NEXT,TIME,TRNSFR
*     Remove the first event from the event list.

      CALL REMOVE(LFIRST,LEVENT)

*     Check for a time reversal.

      IF (TRNSFR(1) .LT. TIME) THEN
        WRITE (IOUT,10) TRNSFR(2), TRNSFR(1), TIME
   10   FORMAT (' From SIMLIB: Attempt to schedule an event of type ',
     &        F3.0/' at time ',F10.3,' when the clock is ',F10.3)
        STOP
      END IF

*     Advance the simulation clock.

      TIME = TRNSFR(1)
      NEXT = TRNSFR(2)

      RETURN
      END


      SUBROUTINE CANCEL(ETYPE)
      INCLUDE 'param.dcl'

      INTEGER AHEAD,BEHIND,HEAD(ML),IOUT,ITEM,LINKPR(MR),LINKSR(MR),
     &     NAR,ROW,TAIL(ML)
      REAL ETYPE,HIGH,LOW,MASTER(MR,MA),VALUE
      COMMON /LLISTS/ HEAD,IOUT,LINKPR,LINKSR,MASTER,NAR,TAIL

      INTEGER LDECR,LEVENT,LFIRST,LINCR,LLAST,LRANK(ML),LSIZE(ML),
     &     MAXATR,NEXT
      REAL TIME,TRNSFR(MA)
      COMMON /SIMLIB/ LDECR,LEVENT,LFIRST,LINCR,LLAST,LRANK,LSIZE,
     &          MAXATR,NEXT,TIME,TRNSFR
```

```
*    Search the event list.

     IF (LSIZE(LEVENT) .EQ. 0) RETURN
     ROW   = HEAD(LEVENT)
     LOW   = ETYPE - 0.1
     HIGH  = ETYPE + 0.1
  10 VALUE = MASTER(ROW,2)
     IF (LOW .LT. VALUE .AND. HIGH .GT. VALUE) GO TO 20

*    Go to the next event.

     IF (ROW .EQ. TAIL(LEVENT)) RETURN
     ROW = LINKSR(ROW)
     GO TO 10

*    Cancel this event.

  20 IF (ROW .NE. HEAD(LEVENT)) GO TO 30
*    Remove the first event in the event list.

     CALL REMOVE(LFIRST,LEVENT)
     RETURN

  30 IF (ROW .NE. TAIL(LEVENT)) GO TO 40

*    Remove the last event in the event list.

     CALL REMOVE(LLAST,LEVENT)
     RETURN

*    Remove this event which is somewhere in the middle of event list.

  40 AHEAD          = LINKSR(ROW)
     BEHIND         = LINKPR(ROW)
     LINKSR(BEHIND) = AHEAD
     LINKPR(AHEAD)  = BEHIND
     LINKSR(ROW)    = NAR
     LINKPR(ROW)    = 0
     NAR            = ROW
     LSIZE(LEVENT)  = LSIZE(LEVENT) - 1

*    Place the attributes of the canceled event in the TRNSFR array.

     DO 50 ITEM = 1, MAXATR
  50 TRNSFR(ITEM) = MASTER(ROW,ITEM)

*    Update the area under the number-in-list curve.

     CALL TIMEST(FLOAT(LSIZE(LEVENT)),MT)
     RETURN
     END
```

```
      SUBROUTINE SAMPST(VALUE,VARIBL)
      INCLUDE 'param.dcl'

      INTEGER HEAD(ML),IOUT,IVAR,LINKPR(MR),LINKSR(MR),NAR,NOBS(ST),
     &        TAIL(ML),VARIBL
      REAL MASTER(MR,MA),MAX(ST),MIN(ST),SUM(ST),VALUE
      COMMON /LLISTS/ HEAD,IOUT,LINKPR,LINKSR,MASTER,NAR,TAIL

      INTEGER LDECR,LEVENT,LFIRST,LINCR,LLAST,LRANK(ML),LSIZE(ML),
     &        MAXATR,NEXT
      REAL TIME,TRNSFR(MA)
      COMMON /SIMLIB/ LDECR,LEVENT,LFIRST,LINCR,LLAST,LRANK,LSIZE,
     &               MAXATR,NEXT,TIME,TRNSFR

*     Force saving of local accumulator arrays between calls.

      SAVE MAX,MIN,NOBS,SUM

*     If the variable number is improper, stop the simulation.

      IF (VARIBL .LT. -ST .OR. VARIBL .GT. ST) THEN
        WRITE (IOUT,10) VARIBL, TIME
   10   FORMAT (' From SIMLIB:',I10,
     &          ' is improper value for SAMPST variable at time ',
     &          F10.3)
        STOP
      END IF

*     Execute the desired option.

      IF (VARIBL) 300, 100, 200

*     Initialize the routine.

  100 DO 110 IVAR = 1, ST
        SUM(IVAR)  = 0.0
        MAX(IVAR)  = -1.0E+30
        MIN(IVAR)  = 1.0E+30
  110 NOBS(IVAR) = 0
      RETURN

*     Collect data.

  200 SUM(VARIBL) = SUM(VARIBL) + VALUE
      IF (VALUE .GT. MAX(VARIBL)) MAX(VARIBL) = VALUE
      IF (VALUE .LT. MIN(VARIBL)) MIN(VARIBL) = VALUE
      NOBS(VARIBL) = NOBS(VARIBL) + 1
      RETURN

*     Report the results.
```

```
 300 IVAR     = -VARIBL
     TRNSFR(1) = 0.0
     TRNSFR(2) = NOBS(IVAR)
     TRNSFR(3) = MAX(IVAR)
     TRNSFR(4) = MIN(IVAR)
     IF (NOBS(IVAR) .EQ. 0) RETURN
     TRNSFR(1) = SUM(IVAR) / TRNSFR(2)

     RETURN
     END


     SUBROUTINE TIMEST(VALUE,VARIBL)
     INCLUDE 'param.dcl'

     INTEGER HEAD(ML),IOUT,IVAR,LINKPR(MR),LINKSR(MR),NAR,TAIL(ML),
    &        VARIBL
     REAL AREA(MT),MASTER(MR,MA),MAX(MT),MIN(MT),PREVAL(MT),TLVC(MT),
    &     VALUE,TRESET
     COMMON /LLISTS/ HEAD,IOUT,LINKPR,LINKSR,MASTER,NAR,TAIL

     INTEGER LDECR,LEVENT,LFIRST,LINCR,LLAST,LRANK(ML),LSIZE(ML),
    &        MAXATR,NEXT
     REAL TIME,TRNSFR(MA)
     COMMON /SIMLIB/ LDECR,LEVENT,LFIRST,LINCR,LLAST,LRANK,LSIZE,
    &                MAXATR,NEXT,TIME,TRNSFR

*    Force saving of local accumulator arrays between calls.

     SAVE AREA,MAX,MIN,PREVAL,TLVC

*    If the variable value is improper, stop the simulation.

     IF (VARIBL .LT. -MT .OR. VARIBL .GT. MT) THEN
       WRITE (IOUT,10) VARIBL, TIME
  10   FORMAT (' From SIMLIB:',I10,
    &          ' is improper value for TIMEST variable at time ',
    &          F10.3)
       STOP
     END IF

*    Execute the desired option.

     IF (VARIBL) 300, 100, 200

*    Initialize the routine.

 100 DO 110 IVAR = 1, MT
       AREA(IVAR)   = 0.0
       MAX(IVAR)    = -1.0E+30
       MIN(IVAR)    = 1.0E+30
```

```
      PREVAL(IVAR) = 0.0
  110 TLVC(IVAR)  = TIME
      TRESET = TIME
      RETURN

*     Collect data.

  200 AREA(VARIBL) = AREA(VARIBL) + (TIME - TLVC(VARIBL))*PREVAL(VARIBL)
      IF (VALUE .GT. MAX(VARIBL)) MAX(VARIBL) = VALUE
      IF (VALUE .LT. MIN(VARIBL)) MIN(VARIBL) = VALUE
      PREVAL(VARIBL) = VALUE
      TLVC(VARIBL) = TIME
      RETURN

*     Report the results.

  300 IVAR     = -VARIBL
      AREA(IVAR) = AREA(IVAR) + (TIME - TLVC(IVAR)) * PREVAL(IVAR)
      TLVC(IVAR) = TIME
      TRNSFR(1)  = AREA(IVAR) / (TIME - TRESET)
      TRNSFR(2)  = MAX(IVAR)
      TRNSFR(3)  = MIN(IVAR)
      RETURN
      END


      SUBROUTINE FILEST(LIST)
      INCLUDE 'param.dcl'

      INTEGER ILIST,LIST

      INTEGER LDECR,LEVENT,LFIRST,LINCR,LLAST,LRANK(ML),LSIZE(ML),
     &        MAXATR,NEXT
      REAL TIME,TRNSFR(MA)
      COMMON /SIMLIB/ LDECR,LEVENT,LFIRST,LINCR,LLAST,LRANK,LSIZE,
     &              MAXATR,NEXT,TIME,TRNSFR

*     Compute summary statistics for the list.

      ILIST = -(ST + LIST)
      CALL TIMEST(0.0,ILIST)

      RETURN
      END


      SUBROUTINE OUTSAM(UNIT,LOWVAR,HIVAR)
      INCLUDE 'param.dcl'

      INTEGER HIVAR,I,IVAR,LOWVAR,UNIT
```

```
 INTEGER LDECR,LEVENT,LFIRST,LINCR,LLAST,LRANK(ML),LSIZE(ML),
&      MAXATR,NEXT
 REAL TIME,TRNSFR(MA)
 COMMON /SIMLIB/ LDECR,LEVENT,LFIRST,LINCR,LLAST,LRANK,LSIZE,
&              MAXATR,NEXT,TIME,TRNSFR

*   Write header.

   WRITE (UNIT,10)
 10 FORMAT (/' SAMPST ',24X,'Number'/
  &      ' Variable',26X,'of'/
  &      ' Number ',6X,'Average',11X,'Values',10X,'Maximum',
  &      10X,'Minimum'/1X,76('_'))

*   Loop for desired SAMPST variable range.

   DO 20 IVAR = LOWVAR, HIVAR

*     Obtain and write summary statistics on SAMPST variable IVAR.

     CALL SAMPST(0.0,-IVAR)
 20 WRITE (UNIT,30) IVAR, (TRNSFR(I), I = 1, 4)
 30 FORMAT (/3X,I3,3X,4(1X,E15.7,1X))
   WRITE (UNIT,40)
 40 FORMAT (1X,76('_'))

   RETURN
   END

   SUBROUTINE OUTTIM(UNIT,LOWVAR,HIVAR)
   INCLUDE 'param.dcl'

   INTEGER HIVAR,I,IVAR,LOWVAR,UNIT

   INTEGER LDECR,LEVENT,LFIRST,LINCR,LLAST,LRANK(ML),LSIZE(ML),
  &      MAXATR,NEXT
   REAL TIME,TRNSFR(MA)
   COMMON /SIMLIB/ LDECR,LEVENT,LFIRST,LINCR,LLAST,LRANK,LSIZE,
  &              MAXATR,NEXT,TIME,TRNSFR

*   Write header.

   WRITE (UNIT,10)
 10 FORMAT (/' TIMEST '/
  &      ' Variable',7X,'Time'/
  &      ' Number ',6X,'Average',10X,'Maximum',10X,'Minimum'/
  &      1X,59('_'))

*   Loop for desired TIMEST variable range.

   DO 20 IVAR = LOWVAR, HIVAR
```

```
*     Obtain and write summary statistics on TIMEST variable IVAR.
      CALL TIMEST(0.0,-IVAR)
   20 WRITE (UNIT,30) IVAR, (TRNSFR(I), I = 1, 3)
   30 FORMAT (/3X,I3,3X,3(1X,E15.7,1X))
      WRITE (UNIT,40)
   40 FORMAT (1X,59('_'))

      RETURN
      END


      SUBROUTINE OUTFIL(UNIT,LOWFIL,HIFIL)
      INCLUDE 'param.dcl'

      INTEGER HIFIL,I,IFIL,LOWFIL,UNIT

      INTEGER LDECR,LEVENT,LFIRST,LINCR,LLAST,LRANK(ML),LSIZE(ML),
     &        MAXATR,NEXT
      REAL TIME,TRNSFR(MA)
      COMMON /SIMLIB/ LDECR,LEVENT,LFIRST,LINCR,LLAST,LRANK,LSIZE,
     &              MAXATR,NEXT,TIME,TRNSFR

*     Write header.

      WRITE (UNIT,10)
   10 FORMAT (/'  File ',8X,'Time'/
     &        ' Number',7X,'Average',10X,'Maximum',10X,'Minimum'/
     &        1X,59('_'))

*     Loop for desired file number range.
      DO 20 IFIL = LOWFIL, HIFIL

*     Obtain and write summary statistics on file IFIL.

      CALL FILEST(IFIL)
   20 WRITE (UNIT,30) IFIL, (TRNSFR(I), I = 1, 3)
   30 FORMAT (/3X,I3,3X,3(1X,E15.7,1X))
      WRITE (UNIT,40)
   40 FORMAT (1X,59('_'))

      RETURN
      END


      REAL FUNCTION EXPON(RMEAN,ISTRM)

      INTEGER ISTRM
      REAL RMEAN,U
      REAL RANFN
```

```
*   Generate a U(0,1) random variate from stream ISTRM.

    U = RANFN(ISTRM)

*   Generate an exponential random variate with mean RMEAN.

    EXPON = -RMEAN * LOG(U)

    RETURN
    END

    REAL FUNCTION EXPONS(RMEAN,RGAM,ISTRM)

    INTEGER ISTRM
    REAL RMEAN,U,RGAM
    REAL RANFN

*   Generate a U(0,1) random variate from stream ISTRM.

    U = RANFN(ISTRM)

*   Generate an exponential shifted random variate with mean RMEAN.

    EXPONS = -RMEAN * LOG(U) + RGAM

    RETURN
    END

    REAL FUNCTION WEIBULLS(RALFA,RBETA,RGAM,ISTRM)

    INTEGER ISTRM
    REAL RALFA,RBETA,U,RGAM
    REAL RANFN

*   Generate a U(0,1) random variate from stream ISTRM.

    U = RANFN(ISTRM)

*   Generate an weibull shifted random variate with alfa,beta and gamma.

    WEIBULLS = RBETA * (-LOG(U))**(1/RALFA) + RGAM

    RETURN
    END

    INTEGER FUNCTION IRANDI(NVALUE,PROBD,ISTRM)

    INTEGER I,ISTRM,NVALUE
    REAL PROBD(1),U
    REAL RANFN
```

```
*    Generate a U(0,1) random variate from stream ISTRM.

     U = RANFN(ISTRM)

*    Generate a random integer between 1 and NVALUE in accordance with
*    the (cumulative) distribution function PROBD.

     DO 10 I = 1, NVALUE - 1
       IF (U .LT. PROBD(I)) THEN
         IRANDI = I
         RETURN
       END IF
  10 CONTINUE
     IRANDI = NVALUE

     RETURN
     END


     REAL FUNCTION UNIFRM(A,B,ISTRM)

     INTEGER ISTRM
     REAL A,B,U
     REAL RANFN

*    Generate a U(0,1) random variate from stream ISTRM.

     U = RANFN(ISTRM)

*    Generate a U(A,B) random variate.

     UNIFRM = A + U * (B - A)

     RETURN
     END


     REAL FUNCTION RANFN(ISTRM)

*    Prime modulus multiplicative linear congruential generator
*    Z(I) = (630360016 * Z(I - 1)) (MOD(2**31 - 1)), based on Marse
*    and Roberts' portable random-number generator UNIRAN. Multiple
*    (100) streams are supported, with seeds spaced 100,000 apart.
*    Throughout, input argument ISTRM must be an INTEGER giving the
*    desired stream number.

*    Usage: (Three options)
*
*    1. To obtain the next U(0,1) random number from stream ISTRM,
*       execute
*         U = RANFN(ISTRM)
```

```
*     The REAL variable U will contain the next random number.
*
*   2. To set the seed for stream ISTRM to a desired value IZSET,
*     execute
*       CALL RANDST(IZSET,ISTRM)
*     where IZSET must be an INTEGER constant or variable set to the
*     desired seed, a number between 1 and 2147483646 (inclusive).
*     Default seeds for all 100 streams are given in the code.
*
*   3. To get the current (most recently used) integer in the
*     sequence being generated for stream ISTRM into the INTEGER
*     variable IZGET, execute
*       IZGET = IRANDG(ISTRM)

      INTEGER B2E15,B2E16,HI15,HI31,ISTRM,IZSET,LOW15,LOWPRD,
     &        MODLUS,MULT1,MULT2,OVFLOW,ZI,ZRNG(100)
      INTEGER IRANDG,RANDST

*     Force saving of ZRNG between calls.

      SAVE ZRNG

*     Define the constants.

      DATA MULT1,MULT2/24112,26143/
      DATA B2E15,B2E16,MODLUS/32768,65536,2147483647/

*     Set the default seeds for all 100 streams.

      DATA ZRNG/1973272912, 281629770,  20006270,1280689831,2096730329,
     &          1933576050, 913566091, 246780520,1363774876, 604901985,
     &          1511192140,1259851944, 824064364, 150493284, 242708531,
     &            75253171,1964472944,1202299975, 233217322,1911216000,
     &           726370533, 403498145, 993232223,1103205531, 762430696,
     &          1922803170,1385516923,  76271663, 413682397, 726466604,
     &           336157058,1432650381,1120463904, 595778810, 877722890,
     &          1046574445,  68911991,2088367019, 748545416, 622401386,
     &          2122378830, 640690903,1774806513,2132545692,2079249579,
     &            78130110, 852776735,1187867272,1351423507,1645973084,
     &          1997049139, 922510944,2045512870, 898585771, 243649545,
     &          1004818771, 773686062, 403188473, 372279877,1901633463,
     &           498067494,2087759558, 493157915, 597104727,1530940798,
     &          1814496276, 536444882,1663153658, 855503735,  67784357,
     &          1432404475, 619691088, 119025595, 880802310, 176192644,
     &          1116780070, 277854671,1366580350,1142483975,2026948561,
     &          1053920743, 786262391,1792203830,1494667770,1923011392,
     &          1433700034,1244184613,1147297105, 539712780,1545929719,
     &           190641742,1645390429, 264907697, 620389253,1502074852,
     &           927711160, 364849192,2049576050, 638580085, 547070247/

*     Generate the next random number.
```

```
      ZI     = ZRNG(ISTRM)
      HI15   = ZI / B2E16
      LOWPRD = (ZI - HI15 * B2E16) * MULT1
      LOW15  = LOWPRD / B2E16
      HI31   = HI15 * MULT1 + LOW15
      OVFLOW = HI31 / B2E15
      ZI     = (((LOWPRD - LOW15 * B2E16) - MODLUS) +
     &          (HI31 - OVFLOW * B2E15) * B2E16) + OVFLOW
      IF (ZI .LT. 0) ZI = ZI + MODLUS
      HI15   = ZI / B2E16
      LOWPRD = (ZI - HI15 * B2E16) * MULT2
      LOW15  = LOWPRD / B2E16
      HI31   = HI15 * MULT2 + LOW15
      OVFLOW = HI31 / B2E15
      ZI     = (((LOWPRD - LOW15 * B2E16) - MODLUS) +
     &          (HI31 - OVFLOW * B2E15) * B2E16) + OVFLOW
      IF (ZI .LT. 0) ZI = ZI + MODLUS
      ZRNG(ISTRM) = ZI
      RANFN  = (2 * (ZI / 256) + 1) / 16777216.0
      RETURN

*     Set the current ZRNG for stream ISTRM to IZSET.

      ENTRY RANDST(IZSET,ISTRM)
      ZRNG(ISTRM) = IZSET
      RETURN

*     Return the current ZRNG for stream ISTRM.

      ENTRY IRANDG(ISTRM)
      IRANDG = ZRNG(ISTRM)
      RETURN

      END
```

Appendix A2

PAC SIMULATION: FORTRAN CODE WITH DESCRIPTION OF VARIABLES

Main Program: PAC.FOR

Subprograms and Declaration Files:

Subprogram	Purpose
SIMLIBG	Set of subroutines of SIMLIB adjusted for PAC Model
ORDARR(NEW)	Processes arrival of an order tag, where NEW=1 if this is a new order from a customer, type 1 events, and NEW=2 if the order tag comes from a cell (called by PACARR), type 8 events; NEW is an INTEGER
REQARR*	Processes type 2 events
PACARR	Processes type 3 events
WIPARR	Processes type 4 events
DEPART	Processes type 5 events
PPARR*	Processes type 6 events
REPORT*	Generates report, called when the simulation ends

* different for DCI and DCII cost calculation approach

Declaration Files	Purpose
PAC.DCL	Declaration of COMMON variables for our MODEL and SIMLIBG
PARAM.DCL	Parameters setting for SIMLIBG
SYSTEM.DCL	Parameters setting for PAC Model

FORTRAN Variables:

Variable	Definition
Input parameters:	
ATYPES	Number of "assembly" product types
CODE	Code given for different coordination scheme (e.g. 1=Produce-to-Order, 2=BSS, 3=MRP, etc.)
COSTD(I)	Customer delay cost factor per "final" product I (in $/item/day)
COSTP(I)	Inventory cost factor in product store per product I (in $/item/day)
COSTW(I,J)	Inventory cost factor in WIP-queue per product I in cell J (in $/item/day)

FTYPES	Number of "final" product types
LENGTH	Length of the simulation, in 24-hour days
MARRVT	Mean interarrival time of demand for "final" product, in min.
MSERVT(I)	Mean service time for product type I, in min.
MTYPES	Number of "final/assembly" product types
NMACHS(I)	Number of machines in cell (unit) I
NSUBASS(I)	Number of different subassemblies per product type I
NUNITS	Number of cell/storage groups
PROBD(I)	Probability of a "final" product type ≤ I
RAWS	Number of different TRAN(I,J) on input file
RTYPES	Number of "raw materials"
SCELL(I)	Number of a production cell of a product type I
SETUP(I)	Setup time per product I in min.
SUBBATCH(I,J)	Number of units of subassembly J per product type I
SUBNAME(I,J)	Name of subassembly J per product type I
TRAN(I,J)	Transportation time per product type I from storage location to cell J (in min.)
WARMUP	Length of "warming up" period, in 24-hour days

PAC control input parameters:

KV(I)	Number of process tags per product type I
RV(I)	Number of product units in batch per product type I sent as PA cards
TV(I)	Time delay between arriving of order tag and requisition tag for product type I
ZV(I)	Initial inventory per product type I at the storage location

SIMLIBG parameters:

MA	Maximum number of attributes for any record in list (=4)
ML	Maximum number of different lists and the number of the event list itself
MR	Maximum number of all records in array MASTER
MT	ML+ST
ST	Maximum number of SAMPST or TIMEST variables

Simulated system parameters:

MG	Maximum number of all different product types incl. raw materials
MP	Maximum number of different product types without raw materials
MS	Maximum number of different subassemblies per assembly
MU	Maximum number of units (i.e. cells/stores)

Modeling variables:

ADDRESS	Number of a cell, that sends a requisition tag for a product type
BATCH(I)	Number of order tags waiting to form a batch for product type I
BATCHWIP(I,J)	Number of units of product type I in a WIP-queue in cell J
CELL	Cell of the current product
CORD	Queue with not met demand for a "final" product type
CUST	Queue with not met demand for a "final" product type
DAYS	Number of days of simulation evaluation period
DELAY	Delay in a WIP-queue for any product at a cell
FLAG	Help variable, used to set statistical accumulators to 0 after "warming up" period
FG	Finished Goods Storage; FG= m+1
INDEX	Number of SAMPST variable for delays in WIP-queue for products
LASTPR(I)	Number of product, which was produced as last one at cell I
NBUSY(I)	Number of machines in cell I that are busy
NEW	(See discussion of event routine ORRDARR(NEW))
NTYPES	Number of product types "final"+"assembly"
ORD	Queue with order tags at store
PAC	Queue with PA cards at cell
PPRD	Queue (per product type) with products at store
PROC	Queue with process tags at store
PROD	Queue with products at store
PRODTYPE	Product type
PRODA	Help variable for product type
PRODW	Help variable for product type
PSERV(I)	Number of demand met immediately per "final" product type I
PWIP	Queue (per product type) with products at cell
REQ	Queue with requisition tags at store
RDELAY	Delay in a CORD-queue of a product to customer
RINDEX	Number of SAMPST variable for delays in CORD-queue for a "final" product type
SDELAY	Delay in a CUST-queue of a product to customer
SINDEX	Number of SAMPST variable for delays in CUST-queue for a "final" product type
STORE	Store of the current product
SUM	Distribution function value for a particular product type (used by REPORT)
TIMEP	Help variable to "keep" the PA card arrival time
WIP	Queue with products at cell

Output variables:

AMDEL(I)	Average delay in queue (WIP) at cell I
APDEL(I)	Average total delay in WIP queue for product type I
AUTIL(I)	Average utilization of the machines in cell I
AVGNIQ(I)	Average number in queue (WIP) at cell I
MAX(I)	Max. delay measured in a WIP-queue for a product type I
MIN(I)	Min. delay measured in a WIP-queue for a product type I
PC1..PCX	Percentages per different costs
PND(I)	Probability an arriving demand from a customer for a product type I is met immediately
PRODI(I)	Time average inventory of parts per store
PRODINV	Total time average inventory of parts
SAPDEL(I)	Average total delay in CUST or CORD-queue for product type I
SMAX(I)	Max. delay of a product type to customer (in CUST or CORD-queue)
SMIN(I)	Min. delay of a product type to customer (in CUST or CORD-queue)
SNUM(I)	Number of products registered in CUST or CORD-queue for product type I
SOAPDEL	Overall average "final" products delay in CUST or CORD-queues
TC	Total cost (in $)
TC1..TCX	Different costs (in $)
WIPI(I)	Time average work-in-process inventory in cells
WIPINV	Total time average work-in-process inventory in cells

FORTRAN Code

Declaration Files:

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C***********************************************************************C
C*  SYSTEM.DCL : System setting -PAC Model                               *C
C*  Author     : Krystyna  Bielunska                                     *C
C*  Date       : October 1995                    (Model 4)               *C
C***********************************************************************C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      INTEGER MU,MP

*     Set values for the system:
*     MU        - maximum number of units (i.e. cells/storages)
*     MP        - maximum number of different product types
*                 without "raw materials"
*     MG        - maximum number of different product types
*                 including "raw materials"
*     MS        - maximum number of different subassembies per assembly

      PARAMETER(MU=4,MP=7,MG=11,MS=2)

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C***********************************************************************C
C*  PAC.DCL    : Declarations File for PAC Model                         *C
C*  Author     : Krystyna  Bielunska                                     *C
C*  Date       : May 1996                                                *C
C***********************************************************************C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      INCLUDE 'param.dcl'
      INCLUDE 'system.dcl'
      INTEGER NUNITS,NMACHS(MU),NTYPES,FTYPES,ATYPES,RTYPES,
     .    ZV(MP),KV(MP),RV(MP),PRODTYPE,NBUSY(MU),
     .    BATCH(MP),PSERV(MP),MTYPES,DAYS,LASTPR(MU),
     .    NSUBASS(MP),SUBNAME(MP,MS),SUBBATCH(MP,MS),
     .    SCELL(MG),BATCHWIP(MG,MU),ADDRESS,FG,TV(MP)
      REAL LENGTH,WARMUP,MARRVT,MSERVT(MP),PROBD(MP),TRAN(MG,MU)
      REAL COSTD(MP),COSTP(MP),COSTW(MG,MU),SETUP(MP)
      COMMON /MODEL/ LENGTH,MARRVT,MSERVT,NUNITS,NMACHS,DAYS,
     .         NTYPES,PROBD,ZV,KV,RV,TV,PRODTYPE,
     .         NBUSY,BATCH,WARMUP,PSERV,ATYPES,
     .         FTYPES,RTYPES,NSUBASS,SUBNAME,SUBBATCH,
     .         SCELL,BATCHWIP,ADDRESS,FG,MTYPES,TRAN,
     .         COSTD,COSTP,COSTW,SETUP,LASTPR
      INTEGER LDECR,LEVENT,LFIRST,LINCR,LLAST,LRANK(ML),LSIZE(ML),
     &    MAXATR,NEXT
      REAL TIME,TRNSFR(MA)
      COMMON /SIMLIB/ LDECR,LEVENT,LFIRST,LINCR,LLAST,LRANK,LSIZE,
     &         MAXATR,NEXT,TIME,TRNSFR
```

Main Program: PAC.FOR

In the main program, the following activities take place, in the order listed:

- open input and output files;
- read the input parameters;
- write report heading and input data on the output file (PAC.OUT);
- initialize all machines in all cells to the idle state and the last product at machines to 0;
- initialize counters of number of PA cards per product type to 0 value;
- initialize counters of number of units per product type in WIP-queues to 0 value;
- call INITLK to initialize the SIMLIB variables;
- initialize number of process tags per product type per store (k_j);
- initialize number of products per product type per store (z_j);
- initialize counters (to 0 value) for calculation of number of customer's arriving demands, which are met immediately per "final" product type;
- schedule an arrival of the first customer demand;
- schedule the end of the simulation;
- determine the length of "warming up" period in minutes;
- call TIMING to determine NEXT, the type of event to occur;
- reset the statistical accumulators to 0 values after "warming up" period;
- transfer control to the appropriate event routine (by using SIMLIB variables and routines where possible), as determined by NEXT; this is done by a computed GOTO statement, routing control to one of 7 CALL statements of the events routines;
- when the simulation ends, call a report generator;
- close all input and output files.

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C**********************************************************************C
C*  PAC.FOR        : Main Simulation Program for PAC Model            *C
C*  Author         : Krystyna  Bielunska                               *C
C*  Date           : June 1996                                         *C
C**********************************************************************C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

*   Bring in declarations file.

    INCLUDE 'pac.dcl'
    INTEGER I,J,K,STORE,CODE,PROC,PROD,PPRD,FLAG,RAWS
    REAL*4 S_TIME,E_TIME

    CALL TIME_STATS(S_TIME)
    FLAG = 0

*   Open input and output files.

    OPEN(5,FILE='pac.in')
    OPEN(6,FILE='pac.out')
*    OPEN(8,FILE='cell12.out')
*    OPEN(9,FILE='cell34.out')
*   Read input parameters.

    READ(5,*) NUNITS,FTYPES,MTYPES,ATYPES,RTYPES,MARRVT,
   .      LENGTH,WARMUP,CODE
    NTYPES=FTYPES+MTYPES+ATYPES
    FG   =NUNITS+1
    DAYS  =LENGTH-WARMUP
    READ(5,*)(NMACHS(I),I=1,NUNITS)
    DO 10 I=1,NTYPES
     READ(5,*) ZV(I),KV(I),RV(I),TV(I),MSERVT(I),SCELL(I),
   .        NSUBASS(I),(SUBNAME(I,J) ,J=1,NSUBASS(I)),
   .              (SUBBATCH(I,J),J=1,NSUBASS(I))
 10 CONTINUE
    READ(5,*)(PROBD(I),I=1,FTYPES+MTYPES)
    READ(5,*)RAWS
    DO 20 I=1,RAWS
     READ(5,*)PRODTYPE,STORE,TRAN(PRODTYPE,STORE),
   .      COSTW(PRODTYPE,STORE)
 20 CONTINUE
    DO 30 I=1,FTYPES+MTYPES
     READ(5,*)COSTD(I)
 30 CONTINUE
    DO 40 I=1,NTYPES
     READ(5,*)COSTP(I),SETUP(I)
 40 CONTINUE

*   Write report heading and input parameters.
```

```
WRITE(6,2010)
2010 FORMAT('*******************************************************',
   .    '********************'/
   .    '*   SIMULATION OF MULTIPLE-CELL SYSTEM COORDINATED BY',
   .    ' PA CARDS (PAC)   *'/
   .    '*   Developed by: Krystyna Bielunska, Ind. Eng., TUNS',
   .    ', February 1996   *'/
   .    '*******************************************************',
   .    '********************'/
   .    ' scenario: exponential service times'/
   .    '          single cost'/)
   IF(CODE.EQ.1) WRITE(6,2011)
2011 FORMAT(' COORDINATION SCHEME : Produce-to-order system'//)
   IF(CODE.EQ.2) WRITE(6,2012)
2012 FORMAT(' Coordination Scheme : Base stock system'//)
   IF(CODE.EQ.3) WRITE(6,2013)
2013 FORMAT(' Coordination Scheme : Material requirements planning'//)
   IF(CODE.EQ.4) WRITE(6,2014)
2014 FORMAT(' Coordination Scheme : KANBAN system'//)
   IF(CODE.EQ.5) WRITE(6,2015)
2015 FORMAT(' Coordination Scheme : Local control'//)
   IF(CODE.EQ.6) WRITE(6,2016)
2016 FORMAT(' Coordination Scheme : Integral control'//)
   IF(CODE.EQ.7) WRITE(6,2017)
2017 FORMAT(' Coordination Scheme : OPT'//)
   IF(CODE.EQ.8) WRITE(6,2018)
2018 FORMAT(' Coordination Scheme : CONWIP'//)
   WRITE(6,2030)NUNITS,(NMACHS(I),I=1,NUNITS)
2030 FORMAT(' INPUT DATA'/
   .    ' ----------'/
   .    ' Number of cells/stores   ',I15/
   .    ' Number of machines in each cell',3X,8I5)
   WRITE(6,2060)FTYPES,MTYPES,ATYPES,RTYPES
2060 FORMAT( ' Number of "final"          products',I9/
   .    ' Number of "final/assembly" products',I9/
   .    ' Number of "assembly"       products',I9/
   .    ' Number of "raw materials"          ',I9)
   WRITE(6,2070)(PROBD(I),I=1,FTYPES+MTYPES)
2070 FORMAT(' Distr.funct. of "final" product types',8F5.2)
   WRITE(6,2080)MARRVT,LENGTH,WARMUP
2080 FORMAT( ' Mean interarr.time of all "final" prod. ',F7.2,' min.'
   .    /' Length of the simulation',F22.1,'  24-hours days'
   .    /' Length of "warming up"  ',F22.1,'  24-hours days'/ )
   WRITE(6,2120)
2120 FORMAT( ' Parameters setting for the coordination scheme:'//
   .    ' Product  Cell/Store  Mean service time  z-value ',
   .    ' k-value  r-value  t-value'/
   .    ' type                 (in min.)'/
   .    ' =======  =========  =================  =======',
   .    ' =======  =======  ========')
   DO 60 I=1,FTYPES+MTYPES+ATYPES
```

```
      WRITE(6,2130)I,SCELL(I),MSERVT(I),ZV(I),KV(I),RV(I),TV(I)
 2130    FORMAT(I4,8X,I3,10X,F6.2,9X,I5,4X,I5,4X,I5,5X,I6)
  60 CONTINUE
     WRITE(6,2132)
 2132 FORMAT(/'Product type   No. of subassemblies   Subassembly name',
   .        '  Units of subassembly'/
   .       '============  ===================  ==================',
   .        '  ====================')
     DO 80 I=1,NTYPES
      DO 70 J=1,NSUBASS(I)
       IF(J.EQ.1) THEN
        WRITE(6,2134)I,NSUBASS(I),SUBNAME(I,J),SUBBATCH(I,J)
       ELSE
        WRITE(6,2136)SUBNAME(I,J),SUBBATCH(I,J)
       END IF
 2134    FORMAT(I7,13X,I7,13X,I7,13X,I7)
 2136    FORMAT(40X,I7,13X,I7)
  70  CONTINUE
  80 CONTINUE
     WRITE(6,2138)
 2138 FORMAT(/'Product    Cell    Time to transport a unit of product',
   .       '   WIP Cost'/
   .       ' type             from storage to cell (in min.)',
   .       '   ($/day/item)'/
   .       '=======  ====  ========================================',
   .       '  ===============')
     DO 90 I=1,NTYPES+RTYPES
      DO 85 J=1,NUNITS
       IF(TRAN(I,J).GT.0.0) THEN
        WRITE(6,2139) I,J,TRAN(I,J),COSTW(I,J)
 2139     FORMAT(I4,7X,I3,19X,F6.2,20X,F6.2)
       END IF
  85  CONTINUE
  90 CONTINUE
     WRITE(6,2140)
 2140 FORMAT(/'Product          Customer Service Cost'/
   .       ' type          Delay Cost ($/item/day)'/
   .       '==============================================')
     DO 92 I=1,FTYPES+MTYPES
      WRITE(6,2142)I,COSTD(I)
 2142  FORMAT(I4,20X,F10.2)
  92 CONTINUE
     WRITE(6,2144)
 2144 FORMAT(/'Product     PROD Cost       Setup'/
   .       ' type     ($/day/item)    (in min.)'/
   .       '==========================================')
     DO 94 I=1,NTYPES
      WRITE(6,2146)I,COSTP(I),SETUP(I)
 2146  FORMAT(I4,12X,F6.2,9X,F6.2)
  94 CONTINUE
     WRITE(6,2160)
```

```
 2160 FORMAT(//' SIMULATION RESULTS'/
     .        '------------------')

*    Initialize all machines in all cells to the idle state and the last
*    product at machines to zero.

      DO 100 I=1,NUNITS
        NBUSY(I) =0
        LASTPR(I)=0
  100 CONTINUE

*    Initialize all batches of PA cards per product per store.

      DO 110 I=1,NTYPES
         BATCH(I)=0
  110 CONTINUE

*    Initialize all "subassembly" batches per product per WIP-queues.

      DO 120 I=1,NTYPES+RTYPES
       DO 115 J=1,NUNITS
         BATCHWIP(I,J)=0
  115  CONTINUE
  120 CONTINUE

*    Initialize SIMLIBG.

      CALL INITLK

*    Initialize number of process tags per product type per store.

      DO 150 I=1,NTYPES
       STORE=SCELL(I)
       IF(KV(I).NE.0) THEN
         PROC=4+(STORE-1)*6
         DO 130 K=1,KV(I)
           TRNSFR(1)=0.0
           TRNSFR(2)=I
           CALL FILE(LLAST,PROC)
  130    CONTINUE
       END IF
  150 CONTINUE

*    Initialize number of products per product type per store.
*    Initialize counter for probability an arriving demand
*    is met immediately.

      DO 190 I=1,NTYPES
       PSERV(I) =0
       STORE=SCELL(I)
       IF(ZV(I).NE.0) THEN
```

```
      PROD=3+(STORE-1)*6
      PPRD=NUNITS*6+FTYPES+MTYPES+I
      DO 170 K=1,ZV(I)
       TRNSFR(1)=0.0
       TRNSFR(2)=I
       CALL FILE(LLAST,PROD)
       CALL FILE(LLAST,PPRD)
 170   CONTINUE
     END IF
 190 CONTINUE

*   Schedule the arrival of the first job.

    TRNSFR(1)=EXPON(MARRVT,1)
    TRNSFR(2)=1.0
    TRNSFR(4)=FG
    CALL FILE(LINCR,LEVENT)

*   Schedule the end of the simulation.

    TRNSFR(1)=1440*LENGTH
    TRNSFR(2)=7.0
    CALL FILE(LINCR,LEVENT)

*   Determine the length of "warming up" period in minutes.

    WARMUP = 1440*WARMUP

*   Schedule the time of recording machine/cell states.

*   TRNSFR(1)=WARMUP
*   TRNSFR(2)=7.0
*   CALL FILE(LINCR,LEVENT)

*   Determine the next event.

 200 CALL TIMING

*   Reset the statistical accumulators to 0 after "warming up" period.

    IF(TIME.GE.WARMUP.AND.FLAG.EQ.0) THEN
     CALL SAMPST(0.0,0)
     DO 202 I=1,NUNITS
      CALL TIMEST(FLOAT(NBUSY(I)),I)
 202  CONTINUE
     FLAG = 1
     DO 205 I=1,NTYPES
      PSERV(I)=0
 205  CONTINUE
    END IF
```

```
*    Call the appropriate event routine.

      GOTO(210,220,230,240,250,260,270),NEXT
 210  CALL ORDARR(1)
      GOTO 200
 220  CALL REQARR
      GOTO 200
 230  CALL PACARR
      GOTO 200
 240  CALL WIPARR
      GOTO 200
 250  CALL DEPART
      GOTO 200
 260  CALL PPARR
      GOTO 200
* 270 CALL STATUS
*     GOTO 200
 270  CALL REPORT

      CALL TIME_STATS(E_TIME)
      WRITE(6,*)'CPU time:',E_TIME-S_TIME
      CLOSE(5)
      CLOSE(6)
*     CLOSE(8)
*     CLOSE(9)

      STOP
      END

      SUBROUTINE TIME_STATS(ELAPSED)
      TYPE TB_TYPE
        SEQUENCE
        REAL*4 USRTIME
        REAL*4 SYSTIME
      END TYPE
      TYPE (TB_TYPE) ETIME_STRUCT
      REAL*4 ELAPSED
      ELAPSED = ETIME_(ETIME_STRUCT)
      RETURN
      END
```

Subprograms:

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C**********************************************************************C
C*  ORDARR.FOR : Order Tag Arrival Routine - PAC Model                *C
C*  Author     : Krystyna  Bielunska                                  *C
C*  Date       : September 1996                                       *C
C**********************************************************************C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE ORDARR(NEW)

      INCLUDE 'pac.dcl'
      INTEGER I,K,NEW,STORE,PROC,ORD,CORD

*     If this is a new order (from a customer), generate the time
*     of the next order arrival and determine the product type
*     of the arriving job, and place it in the CORD-queue.

      IF(NEW.EQ.1) THEN
       TRNSFR(1)=TIME+EXPON(MARRVT,1)
       TRNSFR(2)=1.0
       TRNSFR(4)=FG
       CALL FILE(LINCR,LEVENT)
       PRODTYPE=IRANDI(FTYPES+MTYPES,PROBD,2)

       CORD=NUNITS*6+FTYPES+MTYPES+NTYPES+NUNITS*(NTYPES+RTYPES)
     .   +PRODTYPE
       TRNSFR(1)=TIME
       CALL FILE(LLAST,CORD)
      END IF

*     Determine the store for this order.

      STORE=SCELL(PRODTYPE)

*     Generate the arrival time of the requisition tag of this order
*     to this store.

      TRNSFR(1)=TIME+TV(PRODTYPE)
      TRNSFR(2)=2.0
      TRNSFR(3)=PRODTYPE
      IF(NEW.EQ.1) THEN
       TRNSFR(4)=FG
      ELSE
       TRNSFR(4)=ADDRESS
      END IF
      CALL FILE(LINCR,LEVENT)
```

```
*     If process tags are available: accumulate arriving order tag to
*     a required batch of PA cards or generate PA card(s) if batch is formed;
*     otherwise: place this order tag in a ORD queue.

      PROC=4+(STORE-1)*6
      IF(LSIZE(PROC).NE.0) THEN
       K=LSIZE(PROC)
       DO 10 I=1,K
        CALL REMOVE(LFIRST,PROC)
        IF(TRNSFR(2).EQ.PRODTYPE) THEN
         GOTO 20
        ELSE
         CALL FILE(LLAST,PROC)
        END IF
 10    CONTINUE
       GOTO 50
 20    BATCH(PRODTYPE)=BATCH(PRODTYPE)+1
       IF(BATCH(PRODTYPE).EQ.RV(PRODTYPE)) THEN
        DO 30 I=1,RV(PRODTYPE)
         TRNSFR(1)=TIME
         TRNSFR(2)=3.0
         TRNSFR(3)=PRODTYPE
         CALL FILE(LINCR,LEVENT)
 30      CONTINUE
        IF(RV(PRODTYPE).NE.1) THEN
         ORD=1+(STORE-1)*6
         K=LSIZE(ORD)
         DO 40 I=1,K
          CALL REMOVE(LFIRST,ORD)
          IF(TRNSFR(2).NE.PRODTYPE) THEN
           CALL FILE(LLAST,ORD)
          END IF
 40       CONTINUE
        END IF
        BATCH(PRODTYPE)=0
       ELSE
        ORD=1+(STORE-1)*6
        TRNSFR(1)=TIME
        TRNSFR(2)=PRODTYPE
        TRNSFR(3)=1.0
        CALL FILE(LLAST,ORD)
       END IF
      ELSE
       ORD=1+(STORE-1)*6
       TRNSFR(1)=TIME
       TRNSFR(2)=PRODTYPE
       TRNSFR(3)=0.0
       CALL FILE(LLAST,ORD)
      END IF
      GOTO 100
 50   ORD=1+(STORE-1)*6
```

```
      TRNSFR(1)=TIME
      TRNSFR(2)=PRODTYPE
      TRNSFR(3)=0.0
      CALL FILE(LLAST,ORD)

  100 RETURN
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C**********************************************************************C
C*  REQARR.FOR : Requisition Tag Arrival Routine - PAC Model          *C
C*  Author     : Krystyna  Bielunska                                  *C
C*  Date       : September 1996                (DCI)                  *C
C**********************************************************************C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE REQARR

      INCLUDE 'pac.dcl'
      INTEGER I,K,STORE,PROD,PPRD,REQ,CUST,SINDEX,CORD,RINDEX
      REAL SDELAY,RDELAY

*     Determine product of the arriving requisition tag and the cell
*     it should be delivered.

      PRODTYPE=TRNSFR(3)
      ADDRESS =TRNSFR(4)

*     Determine the store for this requisition tag.

      STORE=SCELL(PRODTYPE)

*     In case of "final" product, place it in CUST-queue for this product.

      IF(ADDRESS.EQ.FG) THEN
       CUST=NUNITS*6+PRODTYPE
       TRNSFR(1)=TIME
       CALL FILE(LLAST,CUST)
      END IF

*     If required product is at this store: remove it from PROD queue;
*     otherwise: place this requisition tag in a REQ queue.

      PPRD=NUNITS*6+FTYPES+MTYPES+PRODTYPE
      IF(LSIZE(PPRD).NE.0) THEN
       CALL REMOVE(LFIRST,PPRD)
       PROD=3+(STORE-1)*6
       K=LSIZE(PROD)
       DO 10 I=1,K
        CALL REMOVE(LFIRST,PROD)
```

```
      IF(TRNSFR(2).EQ.PRODTYPE) THEN
       GOTO 50
      ELSE
       CALL FILE(LLAST,PROD)
      END IF
   10 CONTINUE
    ELSE
     REQ=2+(STORE-1)*6
     TRNSFR(1)=TIME
     TRNSFR(2)=PRODTYPE
     TRNSFR(3)=ADDRESS
     CALL FILE(LLAST,REQ)
     GOTO 100
    END IF

*    If the current departing product is an subassembly, generate its
*    arrival to the next cell; otherwise "deliver it to customer"
*    and calculate service level.

   50 IF(ADDRESS.NE.FG) THEN
      TRNSFR(1)=TIME+TRAN(PRODTYPE,ADDRESS)
      TRNSFR(2)=4.0
      TRNSFR(3)=PRODTYPE
      TRNSFR(4)=ADDRESS
      CALL FILE(LINCR,LEVENT)
     ELSE
      CUST=NUNITS*6+PRODTYPE
      CALL REMOVE(LFIRST,CUST)
      SDELAY=TIME-TRNSFR(1)
      IF(SDELAY.EQ.0) THEN
       PSERV(PRODTYPE)=PSERV(PRODTYPE)+1
      END IF
      SINDEX=NUNITS+NTYPES+RTYPES+PRODTYPE
      CALL SAMPST(SDELAY,SINDEX)
      CORD=NUNITS*6+FTYPES+MTYPES+NTYPES+NUNITS*(NTYPES+RTYPES)
     .    +PRODTYPE
      CALL REMOVE(LFIRST,CORD)
      RDELAY=TIME-TRNSFR(1)
      RINDEX=NUNITS+NTYPES+RTYPES+FTYPES+MTYPES+PRODTYPE
      CALL SAMPST(RDELAY,RINDEX)
     END IF

  100 RETURN
      END
```

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C**********************************************************************C
C*  REQARR.FOR : Requisition Tag Arrival Routine - PAC Model          *C
C*  Author     : Krystyna  Bielunska                                  *C
C*  Date       : September 1996              (DCII)                   *C
C**********************************************************************C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE REQARR

      INCLUDE 'pac.dcl'
      INTEGER I,K,STORE,PROD,PPRD,REQ,CUST,SINDEX,CORD,RINDEX
      REAL SDELAY,RDELAY

*     Determine product of the arriving requisition tag and the cell
*     it should be delivered.

      PRODTYPE=TRNSFR(3)
      ADDRESS =TRNSFR(4)

*     Determine the store for this requisition tag.

      STORE=SCELL(PRODTYPE)

*     In case of "final" product, place it in CUST-queue for this product,
*     and remove it from CORD-queue.

      IF(ADDRESS.EQ.FG) THEN
       CUST=NUNITS*6+PRODTYPE
       TRNSFR(1)=TIME
       CALL FILE(LLAST,CUST)
       CORD=NUNITS*6+FTYPES+MTYPES+NTYPES+NUNITS*(NTYPES+RTYPES)
     .     +PRODTYPE
       CALL REMOVE(LFIRST,CORD)
       RDELAY=TIME-TRNSFR(1)
       RINDEX=NUNITS+NTYPES+RTYPES+FTYPES+MTYPES+PRODTYPE
       CALL SAMPST(RDELAY,RINDEX)
      END IF

*     If required product is at this store: remove it from PROD queue;
*     otherwise: place this requisition tag in a REQ queue.

      PPRD=NUNITS*6+FTYPES+MTYPES+PRODTYPE
      IF(LSIZE(PPRD).NE.0) THEN
       CALL REMOVE(LFIRST,PPRD)
       PROD=3+(STORE-1)*6
       K=LSIZE(PROD)
       DO 10 I=1,K
        CALL REMOVE(LFIRST,PROD)
        IF(TRNSFR(2).EQ.PRODTYPE) THEN
         GOTO 50
```

```
      ELSE
        CALL FILE(LLAST,PROD)
      END IF
 10  CONTINUE
    ELSE
     REQ=2+(STORE-1)*6
     TRNSFR(1)=TIME
     TRNSFR(2)=PRODTYPE
     TRNSFR(3)=ADDRESS
     CALL FILE(LLAST,REQ)
     GOTO 100
    END IF

*    If the current departing product is an subassembly, generate its
*    arrival to the next cell; otherwise "deliver it to customer"
*    and calculate service level.

 50 IF(ADDRESS.NE.FG) THEN
      TRNSFR(1)=TIME+TRAN(PRODTYPE,ADDRESS)
      TRNSFR(2)=4.0
      TRNSFR(3)=PRODTYPE
      TRNSFR(4)=ADDRESS
      CALL FILE(LINCR,LEVENT)
    ELSE
      CUST=NUNITS*6+PRODTYPE
      CALL REMOVE(LFIRST,CUST)
      SDELAY=TIME-TRNSFR(1)
      IF(SDELAY.EQ.0) THEN
        PSERV(PRODTYPE)=PSERV(PRODTYPE)+1
      END IF
      SINDEX=NUNITS+NTYPES+RTYPES+PRODTYPE
      CALL SAMPST(SDELAY,SINDEX)
    END IF

100 RETURN
    END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C*********************************************************************C
C*  PACARR.FOR : PA Card Arrival Routine - PAC Model                  *C
C*  Author      : Krystyna  Bielunska                                 *C
C*  Date        : October 1995                                        *C
C*********************************************************************C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

    SUBROUTINE PACARR

    INCLUDE 'pac.dcl'
    INTEGER I,J,K,CELL,PAC,PRODA
```

```
*   Determine product of the arriving PA card.

    PRODTYPE=TRNSFR(3)

*   Determine the cell for this PA card.

    CELL=SCELL(PRODTYPE)

*   Place this PA card in PAC queue of this cell.

    PAC=6+(CELL-1)*6
    TRNSFR(1)=TIME
    TRNSFR(2)=PRODTYPE
    CALL FILE(LLAST,PAC)

*   Generate an order tag if you require product from
*   another unit (cell/store).
*   Generate an arrival directly to WIP queue of this cell
*   if you require material from the Raw Material store.

    PRODA=PRODTYPE
    DO 50 I=1,NSUBASS(PRODA)
     IF(SUBNAME(PRODA,I).LE.NTYPES) THEN
      DO 10 J=1,SUBBATCH(PRODA,I)
       PRODTYPE=SUBNAME(PRODA,I)
       ADDRESS =CELL
       CALL ORDARR(2)
 10    CONTINUE
     ELSE
      DO 20 K=1,SUBBATCH(PRODA,I)
       TRNSFR(1)=TIME+TRAN(SUBNAME(PRODA,I),CELL)
       TRNSFR(2)=4.0
       TRNSFR(3)=SUBNAME(PRODA,I)
       TRNSFR(4)=CELL
       CALL FILE(LINCR,LEVENT)
 20    CONTINUE
     END IF
 50 CONTINUE

    RETURN
    END
```

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C***********************************************************************C
C*   WIPARR.FOR  : Product to Cell Arrival Routine - PAC Model           *C
C*   Author      : Krystyna  Bielunska                                   *C
C*   Date        : May 1996                                              *C
C***********************************************************************C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE WIPARR

      INCLUDE 'pac.dcl'
      INTEGER I,II,J,JJ,K,KK,L,M,MM,CELL,WIP,PWIP,INDEX,PRODA,PRODW,PAC
      REAL DELAY

*     Determine product type of the arriving product.

      PRODTYPE=TRNSFR(3)

*     Determine the cell to which this product arrives and its WIP-queue.
*     Register arrival of this product and place this product
*     at the end of WIP-queue.

      CELL=TRNSFR(4)
      WIP =5+(CELL-1)*6
      PWIP=NUNITS*6+FTYPES+MTYPES+NTYPES+
     .   (CELL-1)*(NTYPES+RTYPES)+PRODTYPE
      BATCHWIP(PRODTYPE,CELL)=BATCHWIP(PRODTYPE,CELL)+1
      TRNSFR(1)=TIME
      TRNSFR(2)=PRODTYPE
      CALL FILE(LLAST,WIP)
      CALL FILE(LLAST,PWIP)

*     If all machines in the cell are busy go to the next "event".

      IF(NBUSY(CELL).EQ.NMACHS(CELL)) THEN
       GOTO 500
      END IF

*     There is a machine capacity available, so start processing
*     the first product in PAC-queue for which a sufficient batch
*     is formed in a WIP-queue.

      PAC=6+(CELL-1)*6
      K=LSIZE(PAC)
      M=0
      DO 400 J=1,K
       CALL REMOVE(LFIRST,PAC)
       TIMEP=TRNSFR(1)
       PRODA=TRNSFR(2)
       IF(M.EQ.0) THEN
        GOTO 50
```

```
      ELSE
       GOTO 300
      END IF

  50  DO 100 I=1,NSUBASS(PRODA)
       IF(BATCHWIP(SUBNAME(PRODA,I),CELL).LT.SUBBATCH(PRODA,I)) THEN
        GOTO 300
       END IF
 100   CONTINUE

*     There is a product in PAC-queue with a sufficient batch in
*     a WIP-queue, so start processing  it on a machine and
*     schedule end of processing for this product at this cell.

      NBUSY(CELL)=NBUSY(CELL)+1
      CALL TIMEST(FLOAT(NBUSY(CELL)),CELL)
      IF(LASTPR(CELL).EQ.0.OR.LASTPR(CELL).EQ.PRODA) THEN
       TRNSFR(1)=TIME+EXPON(MSERVT(PRODA),3)
      ELSE
       TRNSFR(1)=TIME+SETUP(PRODA)+EXPON(MSERVT(PRODA),3)
      END IF
      TRNSFR(2)=5.0
      TRNSFR(3)=PRODA
      CALL FILE(LINCR,LEVENT)
      M=1

*     Remove all product componenets from WIP-queue, which are used for
*     just scheduled processing.

      DO 220 I=1,NSUBASS(PRODA)
       L=0
       MM=0
       KK=LSIZE(WIP)
       DO 210 JJ=1,KK
        CALL REMOVE(LFIRST,WIP)
        IF(MM.EQ.0) THEN
         IF(TRNSFR(2).EQ.SUBNAME(PRODA,I)) THEN
          BATCHWIP(SUBNAME(PRODA,I),CELL)=
     .         BATCHWIP(SUBNAME(PRODA,I),CELL)-1
          L=L+1
          DELAY=TIME-TRNSFR(1)
          CALL SAMPST(DELAY,CELL)
          INDEX=NUNITS+TRNSFR(2)
          CALL SAMPST(DELAY,INDEX)
         ELSE
          CALL FILE(LLAST,WIP)
         END IF
         IF(L.EQ.SUBBATCH(PRODA,I)) THEN
          MM=1
         END IF
        ELSE
```

```
        CALL FILE(LLAST,WIP)
       END IF
  210   CONTINUE
  220  CONTINUE

     DO 250 I=1,NSUBASS(PRODA)
      PRODW=SUBNAME(PRODA,I)
      PWIP =NUNITS*6+FTYPES+MTYPES+NTYPES+
     .    (CELL-1)*(NTYPES+RTYPES)+PRODW
      DO 240 II=1,SUBBATCH(PRODA,I)
       CALL REMOVE(LFIRST,PWIP)
  240   CONTINUE
  250  CONTINUE

  300  TRNSFR(1)=TIMEP
     TRNSFR(2)=PRODA
     CALL FILE(LLAST,PAC)
  400 CONTINUE

  500 RETURN
     END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C*********************************************************************C
C*  DEPART.FOR : Product from Cell Departure Routine - PAC Model      *C
C*  Author     : Krystyna  Bielunska                                   *C
C*  Date       : May 1996                                              *C
C*********************************************************************C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

     SUBROUTINE DEPART

     INCLUDE 'pac.dcl'
     INTEGER CELL,WIP,PWIP,PAC,I,II,JJ,K,KK,L,M,MM,PRODA,PRODW
     REAL DELAY,TIMEP

*    Determine product type of departing product.

     PRODTYPE=TRNSFR(3)

*    Determine the cell from which this product is departing.

     CELL=SCELL(PRODTYPE)

*    Update the name of the last departing product for this cell.

     LASTPR(CELL)=PRODTYPE

*    Remove PA card from PAC queue for the departing product.
```

```
      PAC=6+(CELL-1)*6
      K=LSIZE(PAC)
      DO 10 I=1,K
        CALL REMOVE(LFIRST,PAC)
        IF(TRNSFR(2).EQ.PRODTYPE) THEN
          GO TO 20
        ELSE
          CALL FILE(LLAST,PAC)
        END IF
   10 CONTINUE

*     Schedule arrival of this product together with process tag to the store.

   20 TRNSFR(1)=TIME
      TRNSFR(2)=6.0
      TRNSFR(3)=PRODTYPE
      CALL FILE(LINCR,LEVENT)

*     Check to see whether the queue for this cell is empty.

      WIP=5+(CELL-1)*6
      IF(LSIZE(WIP).EQ.0) THEN

*     The queue is empty, so make a machine in this cell idle.

        NBUSY(CELL)=NBUSY(CELL)-1
        CALL TIMEST(FLOAT(NBUSY(CELL)),CELL)
        GOTO 500
      END IF

*     The queue is not empty, so start processing the first product
*     in PAC-queue for which a sufficient batch is formed in a WIP-queue.

      M=0
      K=LSIZE(PAC)
      DO 400 J=1,K
        CALL REMOVE(LFIRST,PAC)
        TIMEP=TRNSFR(1)
        PRODA=TRNSFR(2)
        IF(M.EQ.0) THEN
          GOTO 50
        ELSE
          GOTO 300
        END IF

   50 DO 100 I=1,NSUBASS(PRODA)
        IF(BATCHWIP(SUBNAME(PRODA,I),CELL).LT.SUBBATCH(PRODA,I)) THEN
          GOTO 300
        END IF
  100 CONTINUE
```

```
*     There is a product in PAC-queue with a sufficient batch, so
*     schedule end of processing for this product at this cell.

      IF(PRODA.EQ.PRODTYPE) THEN
       TRNSFR(1)=TIME+EXPON(MSERVT(PRODA),3)
      ELSE
       TRNSFR(1)=TIME+SETUP(PRODA)+EXPON(MSERVT(PRODA),3)
      END IF
      TRNSFR(2)=5.0
      TRNSFR(3)=PRODA
      CALL FILE(LINCR,LEVENT)
      M=1

*     Remove all product componenets from WIP-queue, which are used for
*     just scheduled processing.

      DO 220 I=1,NSUBASS(PRODA)
       L=0
       MM=0
       KK=LSIZE(WIP)
       DO 210 JJ=1,KK
        CALL REMOVE(LFIRST,WIP)
        IF(MM.EQ.0) THEN
         IF(TRNSFR(2).EQ.SUBNAME(PRODA,I)) THEN
          BATCHWIP(SUBNAME(PRODA,I),CELL)=
     .        BATCHWIP(SUBNAME(PRODA,I),CELL)-1
          L=L+1
          DELAY=TIME-TRNSFR(1)
          CALL SAMPST(DELAY,CELL)
          INDEX=NUNITS+TRNSFR(2)
          CALL SAMPST(DELAY,INDEX)
         ELSE
          CALL FILE(LLAST,WIP)
         END IF
         IF(L.EQ.SUBBATCH(PRODA,I)) THEN
          MM=1
         END IF
        ELSE
         CALL FILE(LLAST,WIP)
        END IF
210    CONTINUE
220   CONTINUE

      DO 250 I=1,NSUBASS(PRODA)
       PRODW=SUBNAME(PRODA,I)
       PWIP =NUNITS*6+FTYPES+MTYPES+NTYPES+
     .     (CELL-1)*(NTYPES+RTYPES)+PRODW
       DO 240 II=1,SUBBATCH(PRODA,I)
        CALL REMOVE(LFIRST,PWIP)
240    CONTINUE
250   CONTINUE
```

```
 300 TRNSFR(1)=TIMEP
     TRNSFR(2)=PRODA
     CALL FILE(LLAST,PAC)
 400 CONTINUE

*    There is no product in PAC-queue, which can be processed (no enough
*    subassemblies in WIP-queue), so make a machine in this cell idle.

      IF(M.EQ.0) THEN
       NBUSY(CELL)=NBUSY(CELL)-1
       CALL TIMEST(FLOAT(NBUSY(CELL)),CELL)
      END IF

 500 RETURN
     END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C***********************************************************************C
C*   PPARR.FOR   : Product/Process Tag Arrival Routine - PAC Model      *C
C*   Author      : Krystyna  Bielunska                                  *C
C*   Date        : September 1996                    (DCI)              *C
C***********************************************************************C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE PPARR

      INCLUDE 'pac.dcl'
      INTEGER I,K,STORE,PROC,REQ,PROD,PPRD,ORD,CUST,SINDEX,CORD,RINDEX
      REAL SDELAY,RDELAY

*     Determine product type of arriving product/process tag.

      PRODTYPE=TRNSFR(3)

*     Determine the store to which this product arrives.

      STORE=SCELL(PRODTYPE)

*     If an order tag is in ORD queue: accumulate this order tag to
*     a required batch of PA cards or generate PA card(s) if batch is formed;
*     otherwise: place the process tag in PROC queue.

      ORD=1+(STORE-1)*6
      IF(LSIZE(ORD).NE.0) THEN
       K=LSIZE(ORD)
       DO 10 I=1,K
        CALL REMOVE(LFIRST,ORD)
        IF(TRNSFR(2).EQ.PRODTYPE.AND.TRNSFR(3).EQ.0) THEN
         GOTO 20
```

```
      ELSE
       CALL FILE(LLAST,ORD)
      END IF
 10  CONTINUE
     PROC=4+(STORE-1)*6
     TRNSFR(1)=TIME
     TRNSFR(2)=PRODTYPE
     CALL FILE(LLAST,PROC)
     GOTO 100
 20  BATCH(PRODTYPE)=BATCH(PRODTYPE)+1
     IF(BATCH(PRODTYPE).EQ.RV(PRODTYPE)) THEN
      DO 30 I=1,RV(PRODTYPE)
       TRNSFR(1)=TIME
       TRNSFR(2)=3.0
       TRNSFR(3)=PRODTYPE
       CALL FILE(LINCR,LEVENT)
 30    CONTINUE
      BATCH(PRODTYPE)=0
      IF(RV(PRODTYPE).NE.1) THEN
       K=LSIZE(ORD)
       DO 40 I=1,K
        CALL REMOVE(LFIRST,ORD)
        IF(TRNSFR(2).NE.PRODTYPE) THEN
         CALL FILE(LLAST,ORD)
        END IF
 40     CONTINUE
      END IF
     ELSE
      TRNSFR(1)=TIME
      TRNSFR(2)=PRODTYPE
      TRNSFR(3)=1.0
      CALL FILE(LLAST,ORD)
     END IF
    ELSE
     PROC=4+(STORE-1)*6
     TRNSFR(1)=TIME
     TRNSFR(2)=PRODTYPE
     CALL FILE(LLAST,PROC)
    END IF

*    If a requisition tag for this product is at this store:
*    remove it from REQ queue and generate arrival of this product
*    to next cell or to the customer and calculate service level;
*    otherwise: place this product in a PROD queue.

 100 REQ=2+(STORE-1)*6
    IF(LSIZE(REQ).NE.0) THEN
     K=LSIZE(REQ)
     DO 110 I=1,K
      CALL REMOVE(LFIRST,REQ)
      IF(TRNSFR(2).EQ.PRODTYPE) THEN
```

```
      ADDRESS=TRNSFR(3)
      GOTO 130
     ELSE
      CALL FILE(LLAST,REQ)
     END IF
 110  CONTINUE
   ELSE
    GOTO 120
   END IF
 120 PROD=3+(STORE-1)*6
   PPRD=NUNITS*6+FTYPES+MTYPES+PRODTYPE
   TRNSFR(1)=TIME
   TRNSFR(2)=PRODTYPE
   CALL FILE(LLAST,PROD)
   CALL FILE(LLAST,PPRD)
   GOTO 200
 130 IF(ADDRESS.NE.FG) THEN
    TRNSFR(1)=TIME+TRAN(PRODTYPE,ADDRESS)
    TRNSFR(2)=4.0
    TRNSFR(3)=PRODTYPE
    TRNSFR(4)=ADDRESS
    CALL FILE(LINCR,LEVENT)
   ELSE
    CUST=NUNITS*6+PRODTYPE
    CALL REMOVE(LFIRST,CUST)
    SDELAY=TIME-TRNSFR(1)
    IF(SDELAY.EQ.0) THEN
     PSERV(PRODTYPE)=PSERV(PRODTYPE)+1
    END IF
    SINDEX=NUNITS+NTYPES+RTYPES+PRODTYPE
    CALL SAMPST(SDELAY,SINDEX)
    CORD=NUNITS*6+FTYPES+MTYPES+NTYPES+NUNITS*(NTYPES+RTYPES)
  .    +PRODTYPE
    CALL REMOVE(LFIRST,CORD)
    RDELAY=TIME-TRNSFR(1)
    RINDEX=NUNITS+NTYPES+RTYPES+FTYPES+MTYPES+PRODTYPE
    CALL SAMPST(RDELAY,RINDEX)
   END IF

200 RETURN
   END
```

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C*********************************************************************C
C*  PPARR.FOR   : Product/Process Tag Arrival Routine - PAC Model     *C
C*  Author      : Krystyna  Bielunska                                 *C
C*  Date        : September 1996               (DCII)                 *C
C*********************************************************************C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE PPARR

      INCLUDE 'pac.dcl'
      INTEGER I,K,STORE,PROC,REQ,PROD,PPRD,ORD,CUST,SINDEX
      REAL SDELAY

*     Determine product type of arriving product/process tag.

      PRODTYPE=TRNSFR(3)

*     Determine the store to which this product arrives.

      STORE=SCELL(PRODTYPE)

*     If an order tag is in ORD queue: accumulate this order tag to
*     a required batch of PA cards or generate PA card(s) if batch is formed;
*     otherwise: place the process tag in PROC queue.

      ORD=1+(STORE-1)*6
      IF(LSIZE(ORD).NE.0) THEN
       K=LSIZE(ORD)
       DO 10 I=1,K
        CALL REMOVE(LFIRST,ORD)
        IF(TRNSFR(2).EQ.PRODTYPE.AND.TRNSFR(3).EQ.0) THEN
         GOTO 20
        ELSE
         CALL FILE(LLAST,ORD)
        END IF
 10   CONTINUE
      PROC=4+(STORE-1)*6
      TRNSFR(1)=TIME
      TRNSFR(2)=PRODTYPE
      CALL FILE(LLAST,PROC)
      GOTO 100
 20   BATCH(PRODTYPE)=BATCH(PRODTYPE)+1
      IF(BATCH(PRODTYPE).EQ.RV(PRODTYPE)) THEN
       DO 30 I=1,RV(PRODTYPE)
        TRNSFR(1)=TIME
        TRNSFR(2)=3.0
        TRNSFR(3)=PRODTYPE
        CALL FILE(LINCR,LEVENT)
 30    CONTINUE
       BATCH(PRODTYPE)=0
```

```
        IF(RV(PRODTYPE).NE.1) THEN
         K=LSIZE(ORD)
         DO 40 I=1,K
          CALL REMOVE(LFIRST,ORD)
          IF(TRNSFR(2).NE.PRODTYPE) THEN
           CALL FILE(LLAST,ORD)
          END IF
 40       CONTINUE
        END IF
       ELSE
        TRNSFR(1)=TIME
        TRNSFR(2)=PRODTYPE
        TRNSFR(3)=1.0
        CALL FILE(LLAST,ORD)
       END IF
      ELSE
       PROC=4+(STORE-1)*6
       TRNSFR(1)=TIME
       TRNSFR(2)=PRODTYPE
       CALL FILE(LLAST,PROC)
      END IF

*     If a requisition tag for this product is at this store:
*     remove it from REQ queue and generate arrival of this product
*     to next cell or to the customer and calculate service level;
*     otherwise: place this product in a PROD queue.

 100 REQ=2+(STORE-1)*6
     IF(LSIZE(REQ).NE.0) THEN
      K=LSIZE(REQ)
      DO 110 I=1,K
       CALL REMOVE(LFIRST,REQ)
       IF(TRNSFR(2).EQ.PRODTYPE) THEN
        ADDRESS=TRNSFR(3)
        GOTO 130
       ELSE
        CALL FILE(LLAST,REQ)
       END IF
 110  CONTINUE
     ELSE
      GOTO 120
     END IF
 120 PROD=3+(STORE-1)*6
     PPRD=NUNITS*6+FTYPES+MTYPES+PRODTYPE
     TRNSFR(1)=TIME
     TRNSFR(2)=PRODTYPE
     CALL FILE(LLAST,PROD)
     CALL FILE(LLAST,PPRD)
     GOTO 200
 130 IF(ADDRESS.NE.FG) THEN
      TRNSFR(1)=TIME+TRAN(PRODTYPE,ADDRESS)
```

```
     TRNSFR(2)=4.0
     TRNSFR(3)=PRODTYPE
     TRNSFR(4)=ADDRESS
     CALL FILE(LINCR,LEVENT)
    ELSE
     CUST=NUNITS*6+PRODTYPE
     CALL REMOVE(LFIRST,CUST)
     SDELAY=TIME-TRNSFR(1)
     IF(SDELAY.EQ.0) THEN
      PSERV(PRODTYPE)=PSERV(PRODTYPE)+1
     END IF
     SINDEX=NUNITS+NTYPES+RTYPES+PRODTYPE
     CALL SAMPST(SDELAY,SINDEX)
    END IF

 200 RETURN
     END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C*******************************************************************C
C*  REPORT.FOR  : Report Routine - PAC Model                        *C
C*  Author      : Krystyna  Bielunska                               *C
C*  Date        : September 1996            (DCI)                   *C
C*******************************************************************C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC


      SUBROUTINE REPORT

      INCLUDE 'pac.dcl'
      INTEGER I,INDEX,WIP,ORD,REQ,PROD,PROC,PAC,CUST,SINDEX,PPRD,PWIP
      INTEGER CORD,RINDEX
      REAL APDEL(MG),AMDEL(MU),AUTIL(MU),AVGNIQ(MU),SUM
      REAL SAPDEL(MP),SOAPDEL,SNUM(MP)
      REAL SMAX(MP),SMIN(MP),MAX(MG),MIN(MG),PC1,PC2,PC3,PCX
      REAL PRODINV,PRODI(MU),WIPINV,WIPI(MU),PND(MP),TC1,TC2,TC3,TCX

      TC1=0.0
      TC2=0.0
      TC3=0.0
      TCX=0.0

      SOAPDEL=0.0
      SUM   =0.0
      DO 10 I=1,FTYPES+MTYPES
       SINDEX=NUNITS+NTYPES+RTYPES+I
       CALL SAMPST(0.0,-SINDEX)
       SAPDEL(I)=TRNSFR(1)
       SNUM(I)  =TRNSFR(2)
       SMAX(I)  =TRNSFR(3)
```

```
      SMIN(I)  =TRNSFR(4)
      IF(I.LE.FTYPES+MTYPES) THEN
       SOAPDEL  =SOAPDEL+(PROBD(I)-SUM)*SAPDEL(I)
       SUM      =PROBD(I)
      END IF
      PND(I)  =PSERV(I)/SNUM(I)
   10 CONTINUE

      WRITE(6,2010)
 2010 FORMAT( /' SERVICE LEVEL'/
     .       ' -------------'/
     .       '(calculated from time of arrival of requisition tag)'//
     .       ' Product          Probability an arriving demand'/
     .       '  type          from a customer is met immediately')
      DO 15 I=1,FTYPES+MTYPES
        WRITE(6,2015) I,PND(I)
   15 CONTINUE
 2015 FORMAT(/I5,F32.3)
      WRITE(6,2016)
 2016 FORMAT(//' Product       Delay in meeting a customer demand ',
     .       '  Number of'/
     .       ' type          Average    Max        Min    ',
     .       '    units')
      DO 20 I=1,FTYPES+MTYPES
        WRITE(6,2020) I,SAPDEL(I),SMAX(I),SMIN(I),SNUM(I)
   20 CONTINUE
 2020 FORMAT(/I5,F20.3,F12.3,F12.3,F13.3)
      WRITE(6,2030) SOAPDEL
 2030 FORMAT(/' Overall average product total delay to customer   =',
     .       F10.3/)

      DO 25 I=1,FTYPES+MTYPES
        TC1=TC1 + SNUM(I)*SAPDEL(I)*COSTD(I)/1440
   25 CONTINUE

      SOAPDEL=0.0
      SUM    =0.0
      DO 26 I=1,FTYPES+MTYPES
       RINDEX=NUNITS+NTYPES+RTYPES+FTYPES+MTYPES+I
       CALL SAMPST(0.0,-RINDEX)
       SAPDEL(I)=TRNSFR(1)
       SNUM(I)  =TRNSFR(2)
       SMAX(I)  =TRNSFR(3)
       SMIN(I)  =TRNSFR(4)
       IF(I.LE.FTYPES+MTYPES) THEN
        SOAPDEL  =SOAPDEL+(PROBD(I)-SUM)*SAPDEL(I)
        SUM      =PROBD(I)
       END IF
   26 CONTINUE
      WRITE(6,2031)
 2031 FORMAT( /' SERVICE LEVEL'/
```

```
     .      '--------------'/
     .      '(calculated from time of arrival of order tag)')
      WRITE(6,2016)
      DO 27 I=1,FTYPES+MTYPES
        WRITE(6,2020) I,SAPDEL(I),SMAX(I),SMIN(I),SNUM(I)
   27 CONTINUE
      WRITE(6,2030) SOAPDEL
      DO 28 I=1,FTYPES+MTYPES
        TCX=TCX + SNUM(I)*SAPDEL(I)*COSTD(I)/1440
   28 CONTINUE

      DO 30 I=1,NTYPES+RTYPES
        INDEX=NUNITS+I
        IF(I.GT.FTYPES+MTYPES) THEN
          CALL SAMPST(0.0,-INDEX)
          MAX(I)  =TRNSFR(3)
          MIN(I)  =TRNSFR(4)
          APDEL(I)=TRNSFR(1)
        END IF
   30 CONTINUE
      DO 40 I=1,NUNITS
        CALL SAMPST(0.0,-I)
        AMDEL(I)=TRNSFR(1)
        WIP=5+(I-1)*6
        CALL FILEST(WIP)
        AVGNIQ(I)=TRNSFR(1)
        CALL TIMEST(0.0,-I)
        AUTIL(I)=TRNSFR(1)/NMACHS(I)
   40 CONTINUE
      WRITE(6,2040)
 2040 FORMAT(/' Product        Delay (waiting time) in WIP -queue'/
     .      ' type          Average    Max        Min')
      DO 50 I=FTYPES+1,NTYPES+RTYPES
        WRITE(6,2050) I,APDEL(I),MAX(I),MIN(I)
   50 CONTINUE
 2050 FORMAT(/I5,F20.3,F12.3,F12.3)
      WRITE(6,2060)
 2060 FORMAT(//' Machines    Average number      Average',7X,
     .      'Average delay'/
     .      ' in cell       in queue       utilization',5X,
     .      '  in queue')
      DO 60 I=1,NUNITS
        WRITE(6,2070) I,AVGNIQ(I),AUTIL(I),AMDEL(I)
   60 CONTINUE
 2070 FORMAT(/I5,3F17.3)
 2100 FORMAT(/I5,9X,F8.3,9X,F8.3,9X,F8.3)
      WRITE(6,2110)
 2110 FORMAT(//' INVENTORY LEVELS'/
     .      '----------------')
      WRITE(6,2150)
 2150 FORMAT(' PROD-queues  Time average      Maximum',
```

```
     .        '        Minimum'/
     .     ' (per store)')
      PRODINV=0.0
      DO 70 I=1,NUNITS
        PROD=3+(I-1)*6
        CALL FILEST(PROD)
        WRITE(6,2100) I,TRNSFR(1),TRNSFR(2),TRNSFR(3)
        PRODI(I)=TRNSFR(1)
        PRODINV=PRODINV+PRODI(I)
   70 CONTINUE
      WRITE(6,2155)PRODINV
 2155 FORMAT(/' Time average inventory of parts in stores      :',F10.3)
      WRITE(6,2156)
 2156 FORMAT(/' PPRD-queues  Time average     Maximum',
     .        '        Minimum'/
     .     ' (per product)')
      DO 75 I=1,NTYPES
        PPRD=NUNITS*6+FTYPES+MTYPES+I
        CALL FILEST(PPRD)
        WRITE(6,2100) I,TRNSFR(1),TRNSFR(2),TRNSFR(3)
        TC2=TC2+DAYS*TRNSFR(1)*COSTP(I)
   75 CONTINUE
      WRITE(6,2160)
 2160 FORMAT(//' WIP -queues  Time average     Maximum',
     .        '        Minimum'/
     .     ' (per cell)')
      WIPINV=0.0
      DO 80 I=1,NUNITS
        WIP=5+(I-1)*6
        CALL FILEST(WIP)
        WRITE(6,2100) I,TRNSFR(1),TRNSFR(2),TRNSFR(3)
        WIPI(I)=TRNSFR(1)
        WIPINV=WIPINV+WIPI(I)
   80 CONTINUE
      WRITE(6,2165)WIPINV
 2165 FORMAT(/' Time average work-in-process inventory in cells:',F10.3)
      WRITE(6,2166)
 2166 FORMAT(/' PWIP-queues  Time average     Maximum',
     .        '        Minimum'/
     .     ' product cell')
      DO 85 J=1,NTYPES+RTYPES
        DO 83 I=1,NUNITS
          PWIP=NUNITS*6+FTYPES+MTYPES+NTYPES+(I-1)*(NTYPES+RTYPES)+J
          CALL FILEST(PWIP)
          IF(TRNSFR(1).GT.0.0000000001) THEN
            WRITE(6,2167) J,I,TRNSFR(1),TRNSFR(2),TRNSFR(3)
 2167       FORMAT(/I5,2X,I5,2X,F8.3,9X,F8.3,9X,F8.3)
            TC3=TC3+DAYS*TRNSFR(1)*COSTW(J,I)
          END IF
   83   CONTINUE
   85 CONTINUE
```

```
      WRITE(6,2169)
 2169 FORMAT(//' STATISTICAL SUMMARY DATA ON RECORDS IN QUEUES'/
     .       '--------------------------------------')
      WRITE(6,2170)
 2170 FORMAT(' ORD -queues  Time average     Maximum',
     .       '          Minimum'/
     .       ' (per store)')
      DO 90 I=1,NUNITS
        ORD=1+(I-1)*6
        CALL FILEST(ORD)
        WRITE(6,2100) I,TRNSFR(1),TRNSFR(2),TRNSFR(3)
   90 CONTINUE
      WRITE(6,2180)
 2180 FORMAT(//' REQ -queues  Time average     Maximum',
     .       '          Minimum'/
     .       ' (per store)')
      DO 100 I=1,NUNITS
        REQ=2+(I-1)*6
        CALL FILEST(REQ)
        WRITE(6,2100) I,TRNSFR(1),TRNSFR(2),TRNSFR(3)
  100 CONTINUE
      WRITE(6,2190)
 2190 FORMAT(//' PROC-queues  Time average     Maximum',
     .       '          Minimum'/
     .       ' (per store)')
      DO 110 I=1,NUNITS
        PROC=4+(I-1)*6
        CALL FILEST(PROC)
        WRITE(6,2100) I,TRNSFR(1),TRNSFR(2),TRNSFR(3)
  110 CONTINUE
      WRITE(6,2200)
 2200 FORMAT(//' PAC -queues  Time average     Maximum',
     .       '          Minimum'/
     .       ' (per cell)')
      DO 120 I=1,NUNITS
        PAC=6+(I-1)*6
        CALL FILEST(PAC)
        WRITE(6,2100) I,TRNSFR(1),TRNSFR(2),TRNSFR(3)
  120 CONTINUE
      WRITE(6,2210)
 2210 FORMAT(//' CUST-queues Time average     Maximum',
     .       '          Minimum'/
     .       ' (per product)')
      DO 130 I=1,FTYPES+MTYPES
        CUST=NUNITS*6+I
        CALL FILEST(CUST)
        WRITE(6,2100) I,TRNSFR(1),TRNSFR(2),TRNSFR(3)
  130 CONTINUE
      WRITE(6,2215)
 2215 FORMAT(//' CORD-queues Time average     Maximum',
     .       '          Minimum'/
```

```
     .      ' (per product)')
      DO 140 I=1,FTYPES+MTYPES
        CORD=NUNITS*6+FTYPES+MTYPES+NTYPES+NUNITS*(NTYPES+RTYPES)+I
        CALL FILEST(CORD)
        WRITE(6,2100) I,TRNSFR(1),TRNSFR(2),TRNSFR(3)
  140 CONTINUE

      TC=TCX+TC2+TC3
      PC1=TC1/TC*100
      PCX=TCX/TC*100
      PC2=TC2/TC*100
      PC3=TC3/TC*100
      WRITE(6,2220)TC1,PC1,TCX,PCX,TC2,PC2,TC3,PC3,TC
 2220 FORMAT(/'CUST. "REQ" SERV.COST [$] =',F20.2,'    (',F5.1,' %)'/
     .     '==================================================',
     .     '==========='/
     .     'CUST. "ORD" SERV.COST [$] =',F20.2,'    (',F5.1,' %)'/
     .     'PROD. INVENTORY  COST [$] =',F20.2,'    (',F5.1,' %)'/
     .     'WIP.  INVENTORY  COST [$] =',F20.2,'    (',F5.1,' %)'/
     .     '==================================================',
     .     '==========='/
     .     '        TOTAL COST [$] =',F20.2,'    (100.0 %)')

      RETURN
      END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C*****************************************************************C
C*  REPORT.FOR  : Report Routine - PAC Model                      *C
C*  Author      : Krystyna  Bielunska                             *C
C*  Date        : September 1996                  (DCII)          *C
C*****************************************************************C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE REPORT

      INCLUDE 'pac.dcl'
      INTEGER I,INDEX,WIP,ORD,REQ,PROD,PROC,PAC,CUST,SINDEX,PPRD,PWIP
      INTEGER CORD,RINDEX
      REAL APDEL(MG),AMDEL(MU),AUTIL(MU),AVGNIQ(MU),SUM
      REAL SAPDEL(MP),SOAPDEL,SNUM(MP)
      REAL SMAX(MP),SMIN(MP),MAX(MG),MIN(MG),PC1,PC2,PC3,PCX
      REAL PRODINV,PRODI(MU),WIPINV,WIPI(MU),PND(MP),TC1,TC2,TC3,TCX

      TC1=0.0
      TC2=0.0
      TC3=0.0
      TCX=0.0

      SOAPDEL=0.0
```

```
      SUM   =0.0
      DO 10 I=1,FTYPES+MTYPES
       SINDEX=NUNITS+NTYPES+RTYPES+I
       CALL SAMPST(0.0,-SINDEX)
       SAPDEL(I)=TRNSFR(1)
       SNUM(I)  =TRNSFR(2)
       SMAX(I)  =TRNSFR(3)
       SMIN(I)  =TRNSFR(4)
       IF(I.LE.FTYPES+MTYPES) THEN
        SOAPDEL  =SOAPDEL+(PROBD(I)-SUM)*SAPDEL(I)
        SUM      =PROBD(I)
       END IF
       PND(I)  =PSERV(I)/SNUM(I)
   10 CONTINUE

      WRITE(6,2010)
 2010 FORMAT( /' SERVICE LEVEL'/
     .       ' -------------'/
     .       '(calculated from time of arrival of requisition tag)'//
     .       ' Product         Probability an arriving demand'/
     .       '  type          from a customer is met immediately')
      DO 15 I=1,FTYPES+MTYPES
       WRITE(6,2015) I,PND(I)
   15 CONTINUE
 2015 FORMAT(/I5,F32.3)
      WRITE(6,2016)
 2016 FORMAT(//' Product       Delay in meeting a customer demand ',
     .       '  Number of'/
     .       ' type          Average    Max       Min    ',
     .       '    units')
      DO 20 I=1,FTYPES+MTYPES
       WRITE(6,2020) I,SAPDEL(I),SMAX(I),SMIN(I),SNUM(I)
   20 CONTINUE
 2020 FORMAT(/I5,F20.3,F12.3,F12.3,F13.3)
      WRITE(6,2030) SOAPDEL
 2030 FORMAT(/' Overall average product total delay to customer  =',
     .       F10.3/)
      DO 25 I=1,FTYPES+MTYPES
       TC1=TC1 + SNUM(I)*SAPDEL(I)*COSTD(I)/1440
   25 CONTINUE
      SOAPDEL=0.0
      SUM   =0.0
      DO 26 I=1,FTYPES+MTYPES
       RINDEX=NUNITS+NTYPES+RTYPES+FTYPES+MTYPES+I
       CALL SAMPST(0.0,-RINDEX)
       SAPDEL(I)=TRNSFR(1)
       SNUM(I)  =TRNSFR(2)
       SMAX(I)  =TRNSFR(3)
       SMIN(I)  =TRNSFR(4)
       IF(I.LE.FTYPES+MTYPES) THEN
        SOAPDEL  =SOAPDEL+(PROBD(I)-SUM)*SAPDEL(I)
```

```
        SUM    =PROBD(I)
       END IF
   26 CONTINUE
      WRITE(6,2031)
 2031 FORMAT( /' SERVICE LEVEL'/
     .      ' -----------'/
     . '(calculated between time of arrival of ord. tag and req. tag)')
      WRITE(6,2032)
 2032 FORMAT(//' Product       Waiting of an entity in CORD-queue ',
     .      '  Number of'/
     .      ' type            Average    Max        Min  ',
     .      '    units')
      DO 27 I=1,FTYPES+MTYPES
       WRITE(6,2020) I,SAPDEL(I),SMAX(I),SMIN(I),SNUM(I)
   27 CONTINUE
      WRITE(6,2033) SOAPDEL
 2033 FORMAT(/' Overall average entity waiting time in CORD-queues=',
     .       F10.3/)
      DO 28 I=1,FTYPES+MTYPES
       TCX=TCX + SNUM(I)*SAPDEL(I)*COSTD(I)*0.05/1440
   28 CONTINUE
      DO 30 I=1,NTYPES+RTYPES
       INDEX=NUNITS+I
       IF(I.GT.FTYPES+MTYPES) THEN
        CALL SAMPST(0.0,-INDEX)
        MAX(I)  =TRNSFR(3)
        MIN(I)  =TRNSFR(4)
        APDEL(I)=TRNSFR(1)
       END IF
   30 CONTINUE
      DO 40 I=1,NUNITS
       CALL SAMPST(0.0,-I)
       AMDEL(I)=TRNSFR(1)
       WIP=5+(I-1)*6
       CALL FILEST(WIP)
       AVGNIQ(I)=TRNSFR(1)
       CALL TIMEST(0.0,-I)
       AUTIL(I)=TRNSFR(1)/NMACHS(I)
   40 CONTINUE
      WRITE(6,2040)
 2040 FORMAT(/' Product       Delay (waiting time) in WIP -queue'/
     .      ' type            Average    Max        Min')
      DO 50 I=FTYPES+1,NTYPES+RTYPES
       WRITE(6,2050) I,APDEL(I),MAX(I),MIN(I)
   50 CONTINUE
 2050 FORMAT(/I5,F20.3,F12.3,F12.3)
      WRITE(6,2060)
 2060 FORMAT(//' Machines    Average number      Average',7X,
     .      'Average delay'/
     .      ' in cell       in queue        utilization',5X,
     .      '  in queue')
```

```
      DO 60 I=1,NUNITS
        WRITE(6,2070) I,AVGNIQ(I),AUTIL(I),AMDEL(I)
   60 CONTINUE
 2070 FORMAT(/I5,3F17.3)
 2100 FORMAT(/I5,9X,F8.3,9X,F8.3,9X,F8.3)
      WRITE(6,2110)
 2110 FORMAT(//' INVENTORY LEVELS'/
     .      ' ---------------')
      WRITE(6,2150)
 2150 FORMAT(' PROD-queues  Time average      Maximum',
     .      '        Minimum'/
     .      ' (per store)')
      PRODINV=0.0
      DO 70 I=1,NUNITS
        PROD=3+(I-1)*6
        CALL FILEST(PROD)
        WRITE(6,2100) I,TRNSFR(1),TRNSFR(2),TRNSFR(3)
        PRODI(I)=TRNSFR(1)
        PRODINV=PRODINV+PRODI(I)
   70 CONTINUE
      WRITE(6,2155)PRODINV
 2155 FORMAT(/' Time average inventory of parts in stores      :',F10.3)
      WRITE(6,2156)
 2156 FORMAT(/' PPRD-queues  Time average      Maximum',
     .      '        Minimum'/
     .      ' (per product)')
      DO 75 I=1,NTYPES
        PPRD=NUNITS*6+FTYPES+MTYPES+I
        CALL FILEST(PPRD)
        WRITE(6,2100) I,TRNSFR(1),TRNSFR(2),TRNSFR(3)
        TC2=TC2+DAYS*TRNSFR(1)*COSTP(I)
   75 CONTINUE
      WRITE(6,2160)
 2160 FORMAT(//' WIP -queues  Time average      Maximum',
     .      '        Minimum'/
     .      ' (per cell)')
      WIPINV=0.0
      DO 80 I=1,NUNITS
        WIP=5+(I-1)*6
        CALL FILEST(WIP)
        WRITE(6,2100) I,TRNSFR(1),TRNSFR(2),TRNSFR(3)
        WIPI(I)=TRNSFR(1)
        WIPINV=WIPINV+WIPI(I)
   80 CONTINUE
      WRITE(6,2165)WIPINV
 2165 FORMAT(/' Time average work-in-process inventory in cells:',F10.3)
      WRITE(6,2166)
 2166 FORMAT(/' PWIP-queues  Time average      Maximum',
     .      '        Minimum'/
     .      ' product cell')
      DO 85 J=1,NTYPES+RTYPES
```

```
        DO 83 I=1,NUNITS
         PWIP=NUNITS*6+FTYPES+MTYPES+NTYPES+(I-1)*(NTYPES+RTYPES)+J
         CALL FILEST(PWIP)
         IF(TRNSFR(1).GT.0.0000000001) THEN
          WRITE(6,2167) J,I,TRNSFR(1),TRNSFR(2),TRNSFR(3)
 2167      FORMAT(/I5,2X,I5,2X,F8.3,9X,F8.3,9X,F8.3)
          TC3=TC3+DAYS*TRNSFR(1)*COSTW(J,I)
         END IF
  83  CONTINUE
  85 CONTINUE
      WRITE(6,2169)
 2169 FORMAT(//' STATISTICAL SUMMARY DATA ON RECORDS IN QUEUES'/
     .       '-----------------------------------------')
      WRITE(6,2170)
 2170 FORMAT(' ORD -queues  Time average     Maximum',
     .       '          Minimum'/
     .       ' (per store)')
      DO 90 I=1,NUNITS
       ORD=1+(I-1)*6
       CALL FILEST(ORD)
       WRITE(6,2100) I,TRNSFR(1),TRNSFR(2),TRNSFR(3)
  90 CONTINUE
      WRITE(6,2180)
 2180 FORMAT(//' REQ -queues  Time average     Maximum',
     .       '          Minimum'/
     .       ' (per store)')
      DO 100 I=1,NUNITS
       REQ=2+(I-1)*6
       CALL FILEST(REQ)
       WRITE(6,2100) I,TRNSFR(1),TRNSFR(2),TRNSFR(3)
 100 CONTINUE
      WRITE(6,2190)
 2190 FORMAT(//' PROC-queues  Time average     Maximum',
     .       '          Minimum'/
     .       ' (per store)')
      DO 110 I=1,NUNITS
       PROC=4+(I-1)*6
       CALL FILEST(PROC)
       WRITE(6,2100) I,TRNSFR(1),TRNSFR(2),TRNSFR(3)
 110 CONTINUE
      WRITE(6,2200)
 2200 FORMAT(//' PAC -queues  Time average     Maximum',
     .       '          Minimum'/
     .       ' (per cell)')
      DO 120 I=1,NUNITS
       PAC=6+(I-1)*6
       CALL FILEST(PAC)
       WRITE(6,2100) I,TRNSFR(1),TRNSFR(2),TRNSFR(3)
 120 CONTINUE
      WRITE(6,2210)
 2210 FORMAT(//' CUST-queues Time average     Maximum',
```

```
     .        '          Minimum'/
     .        ' (per product)')
      DO 130 I=1,FTYPES+MTYPES
        CUST=NUNITS*6+I
        CALL FILEST(CUST)
        WRITE(6,2100) I,TRNSFR(1),TRNSFR(2),TRNSFR(3)
  130 CONTINUE
      WRITE(6,2215)
 2215 FORMAT(//' CORD-queues Time average     Maximum',
     .        '          Minimum'/
     .        ' (per product)')
      DO 140 I=1,FTYPES+MTYPES
        CORD=NUNITS*6+FTYPES+MTYPES+NTYPES+NUNITS*(NTYPES+RTYPES)+I
        CALL FILEST(CORD)
        WRITE(6,2100) I,TRNSFR(1),TRNSFR(2),TRNSFR(3)
  140 CONTINUE

      TC =TC1+TCX+TC2+TC3
      PC1=TC1/TC*100
      PCX=TCX/TC*100
      PC2=TC2/TC*100
      PC3=TC3/TC*100
      WRITE(6,2220)TC1,PC1,TCX,PCX,TC2,PC2,TC3,PC3,TC
 2220 FORMAT(/'CUST. "REQ" SERV.COST [$] =',F20.2,'    (',F5.1,' %)'/
     .      'CUST. "ORD-REQ"  COST [$] =',F20.2,'    (',F5.1,' %)'/
     .      'PROD. INVENTORY  COST [$] =',F20.2,'    (',F5.1,' %)'/
     .      'WIP.  INVENTORY  COST [$] =',F20.2,'    (',F5.1,' %)'/
     .      '==================================================',
     .      '=========='/
     .      '         TOTAL COST [$] =',F20.2,'    (100.0 %)')

      RETURN

      END
```

Appendix A3

SIMULATION REPORT

Subroutine REPORT is used to create the output of the simulation results. The simulation report of the PAC model contains four main parts:

Service Level

- for each (*f*+*fa*) product type: probability that an arriving demand for a product type is met immediately, calculated from the arrival time of a requisition tag;
- for each (*f*+*fa*) product type: an average, maximum and minimum value of a delay in meeting a customer demand, the total number of units delivered to the customer during the simulation run; and an overall product delay to customer, calculated from the arrival time of a requisition tag;
- for each (*f*+*fa*) product type:
  - DCI: an average, maximum and minimum value of a delay in meeting a customer demand, the total number of units delivered to the customer during the simulation run; and an overall product delay to customer, calculated from the arrival time of an order tag;
  - DCII: an average, maximum and minimum value of a waiting time of entities in the CORD-queues, the total number of units registered in CORD-queues, calculated between the arrival time of an order tag and a requisition tag;
- for each (*fa*+*a*+*r*) product type: an average, maximum, and minimum value of a delay (i.e. waiting time) in all WIP-queues the product is delivered to;
- for each machine group (machines in a cell): time average of work-in-process queue (WIP-queue), average machine utilization and average waiting time in WIP-queue.

Inventory Levels

- for each store: a time average, maximum and minimum length of PROD-queue with a total time average of all parts in all stores;
- for each product type: a time average, maximum and minimum length of PPRD-queue;
- for each cell: a time average, maximum and minimum length of WIP-queue with a total time average of all parts in work-in-process inventory in all cells;
- for each (*fa*+*a*+*r*) product type and each cell, where this product is used as a component: a time average, maximum and minimum length of PWIP-queue.

Statistical Summary Data on Records in Queues

- for each unit: a time average, maximum and minimum of ORD, REQ, PROC, PAC-queues;
- for each (*f*+*fa*) product type: a time average, maximum and minimum of CUST and CORD-queues.

Costs with Percentages and the CPU Time

- Customer delay cost(s):
- Inventory cost in product stores;
- Inventory cost in cells;
- Total cost;
- CPU time.

Example of a Complete Simulation Report:

```
**************************************************************************************
* SIMULATION OF MULTIPLE-CELL SYSTEM COORDINATED BY PA CARDS (PAC)
* Developed by: Krystyna Bielunska, Industrial Engineering Department, TUNS, February 1996
**************************************************************************************
```

scenario: exponential service times
DCII (5%, 100%)

Coordination Scheme : Material Requirements Planning

INPUT DATA

Number of cells/stores	4
Number of machines in each cell	1 1 1 1
Number of "final" products	3
Number of "final/assembly" products	1
Number of "assembly" products	3
Number of "raw materials"	4
Distr.funct. of "final" product types	.50 .60 .90 1.00
Mean interarr.time of all "final" prod.	180.00 min.
Length of the simulation	300.0 24-hours days
Length of "warming up"	40.0 24-hours days

Parameters setting for the coordination scheme:

Product type	Cell/Store	Mean service time (in min.)	z-value	k-value	r-value	t-value (in min.)
1	1	42.00	2	200	5	1440
2	4	18.00	1	200	4	1440
3	4	36.00	1	200	3	240
4	2	18.00	1	200	3	240
5	2	24.00	0	200	4	60
6	3	18.00	0	200	3	60
7	3	30.00	0	200	2	60

Product type	No. of subassemblies	Subassembly name	Units of subassembly
1	2	4	1
		5	1
2	1	5	3
3	2	5	1
		6	2
4	1	8	1
5	2	7	3
		9	2
6	1	11	1
7	1	10	1

Product type	Cell	Time to transport a unit of product from storage to cell (in min.)	WIP Cost ($/item/day)
4	1	12.00	3.00
5	1	12.00	12.00
5	4	18.00	12.00
6	4	6.00	3.00
7	2	18.00	3.00
8	2	24.00	2.00
9	2	24.00	1.00
10	3	18.00	2.00
11	3	18.00	2.00

Product type	Customer Service Cost Delay Cost ($/item/day)
1	10.00
2	8.00
3	6.00
4	5.00

Product type	PROD Cost ($/item/day)	Setup (in min.)
1	16.00	.00
2	40.00	7.00
3	20.00	1.00
4	3.00	1.00
5	12.00	9.00
6	3.00	1.00
7	3.00	8.00

SIMULATION RESULTS

SERVICE LEVEL

(calculated from time of arrival of requisition tag)

Product type	Probability an arriving demand from a customer is met immediately
1	.800
2	.453
3	.090
4	.700

Product type	Delay in meeting a customer demand (in min.) Average	Max	Min	Number of units
1	87.717	2422.547	.000	1021.000
2	886.060	7929.672	.000	212.000
3	780.952	6005.344	.000	636.000
4	240.068	2620.875	.000	210.000

Overall average product total delay to customer = 390.757

SERVICE LEVEL

(calculated between time of arrival of ord. tag and req. tag)

Product type	Waiting of an entity in CORD-queue (in min.) Average	Max	Min	Number of units
1	1440.000	1440.016	1439.992	1021.000
2	1440.000	1440.000	1439.984	212.000
3	240.000	240.000	240.000	635.000
4	240.000	240.000	240.000	210.000

Overall average entity waiting time in CORD-queues= 960.000

Product type	Delay (waiting time) in WIP -queue (in min.) Average	Max	Min
4	.000	.000	.000
5	51.862	2763.359	.000
6	582.749	2884.375	.000
7	36.287	355.562	.000
8	44.088	397.000	.000
9	605.359	2272.719	54.000
10	526.465	2239.406	.000
11	300.265	1704.281	.000

Machines in cell	Average number in queue	Average utilization	Average delay in queue (in min.)
1	1.671	.116	296.173
2	8.260	.215	242.593
3	10.844	.615	491.122
4	2.187	.070	320.994

INVENTORY LEVELS

PROD-queues (per store)	Time average	Maximum	Minimum
1	1.653	8.000	.000
2	.745	8.000	.000
3	.020	6.000	.000
4	.330	3.000	.000

Time average inventory of parts in stores: 2.748

PPRD-queues (per product)	Time average	Maximum	Minimum
1	1.653	8.000	.000
2	.223	2.000	.000
3	.107	2.000	.000
4	.745	8.000	.000
5	.000	********	********
6	.010	5.000	.000
7	.010	6.000	.000

WIP -queues (per cell)	Time average	Maximum	Minimum
1	1.671	15.000	.000
2	8.260	46.000	.000
3	10.844	72.000	.000
4	2.187	13.000	.000

Time average work-in-process inventory in cells: 22.962

PWIP-queues product/ cell		Time average	Maximum	Minimum
4	1	1.559	14.000	.000
5	1	.112	4.000	.000
5	4	.194	4.000	.000
6	4	1.993	12.000	.000
7	2	.654	11.000	.000
8	2	.144	6.000	.000
9	2	7.462	42.000	.000
10	3	9.759	60.000	.000
11	3	1.085	12.000	.000

STATISTICAL SUMMARY DATA ON RECORDS IN QUEUES

ORD -queues (per store)	Time average	Maximum	Minimum
1	2.023	4.000	.000
2	2.581	5.000	.000
3	.000	2.000	.000
4	2.446	5.000	.000

REQ -queues (per store)	Time average	Maximum	Minimum
1	.263	7.000	.000
2	5.477	22.000	.000
3	10.565	73.000	.000
4	1.876	8.000	.000

PROC-queues (per store)	Time average	Maximum	Minimum
1	195.417	200.000	1.000
2	393.103	400.000	1.000
3	388.149	400.000	1.000
4	395.245	400.000	1.000

PAC -queues (per cell)	Time average	Maximum	Minimum
1	2.560	15.000	.000
2	4.316	26.000	.000
3	11.852	74.000	.000
4	2.309	10.000	.000

CUST-queues (per product)	Time average	Maximum	Minimum
1	.263	7.000	.000
2	.513	2.000	.000
3	1.363	7.000	.000
4	.130	1.000	.000

CORD-queues (per product)	Time average	Maximum	Minimum
1	3.973	12.000	.000
2	.797	5.000	.000
3	.413	5.000	.000
4	.134	3.000	.000

CUST. "REQ" DELAY COST [$]=	3910.09	(14.6 %)
CUST. "ORD-REQ" COST [$] =	635.80	(2.4 %)
PROD. INVENTORY COST [$] =	10350.52	(38.6 %)
WIP. INVENTORY COST [$] =	11890.86	(44.4 %)
TOTAL COST [$]=	26787.27	(100.0 %)

CPU time: 4.05 (seconds) on IBM RISC/6000 Model Computer

Appendix A4

MACHINE/CELL STATE ROUTINE

Subroutine STATUS help us to collect data about state of the machine groups and length of the WIP-queues. The machine group in a cell can be in two states: busy (NBUSY(I)=1) or idle (NBUSY(I)=0). The length of any WIP-queue is kept by SIMLIB variable LSIZE(LIST). This routine has to be adjusted for any new manufacturing configuration, as it highly depends on number of units of the simulated system.

In case of our example-model we collected data for the first 10 hours after the "warming up'" time (=40 hours) was finished, and with a time interval of 0.1 hour (i.e. every 6 minutes). The data were written in two separate files in such a format, that it was easy to read them directly into EXCEL and create useful charts.

Modeling variables:

ITAB	ASCII- code for "TAB"-key
TAB	"TAB"-key
TIME1	Help variable for simulation time

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C************************************************************************C
C*  STATUS.FOR  : Machine/Cell State Routine - PAC Model                *C
C*  Author      : Krystyna  Bielunska                                    *C
C*  Date        : October 1995                                           *C
C************************************************************************C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE STATUS

      INCLUDE 'pac.dcl'

      REAL TIME1
      CHARACTER*1 TAB
      INTEGER*1  ITAB
      EQUIVALENCE(TAB,ITAB)
      ITAB=9

*     Register state of machines and length of the WIP-queues

      WRITE(8,2120) TIME,TAB,NBUSY(1),TAB,LSIZE(5),TAB,
     .        TIME,TAB,NBUSY(2),TAB,LSIZE(11)
      WRITE(9,2120) TIME,TAB,NBUSY(3),TAB,LSIZE(17),TAB,
     .        TIME,TAB,NBUSY(4),TAB,LSIZE(23)

 2120 FORMAT(F8.1,A1,I3,A1,I3,A1,F8.1,A1,I3,A1,I3)

*     Schedule the next event of registration of machine/cell state.

      TIME1=TIME-WARMUP
      IF(TIME1.LT.10.0) THEN
       TRNSFR(1)=TIME+0.1
       TRNSFR(2)=7.0
       CALL FILE(LINCR,LEVENT)
      END IF

      RETURN
      END
```

Examples of charts for machines and the WIP-queue:

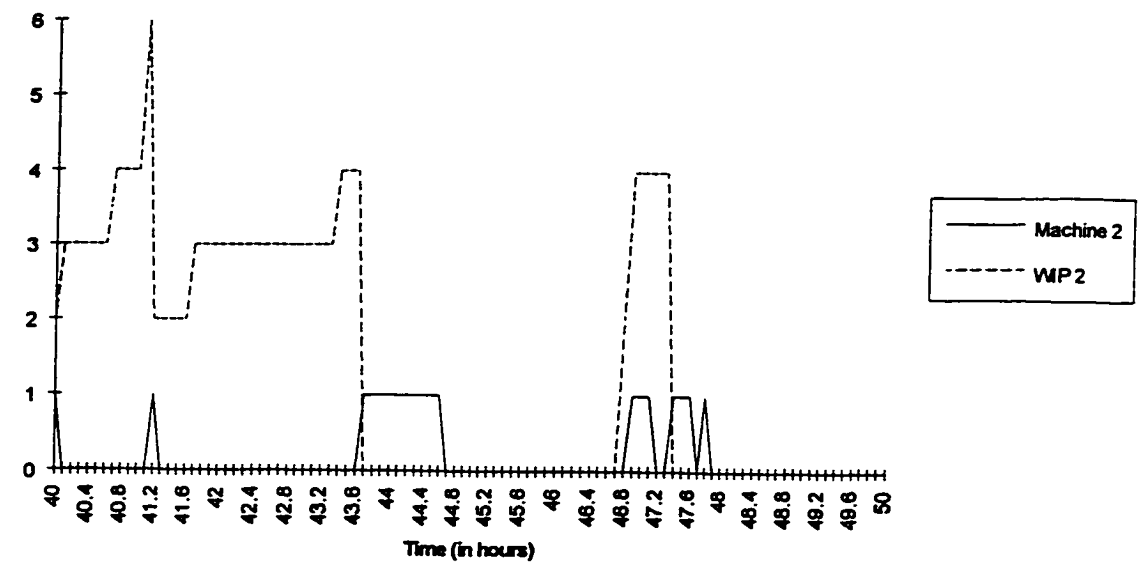


Figure a5.1 Machine/cell state for cell 2.

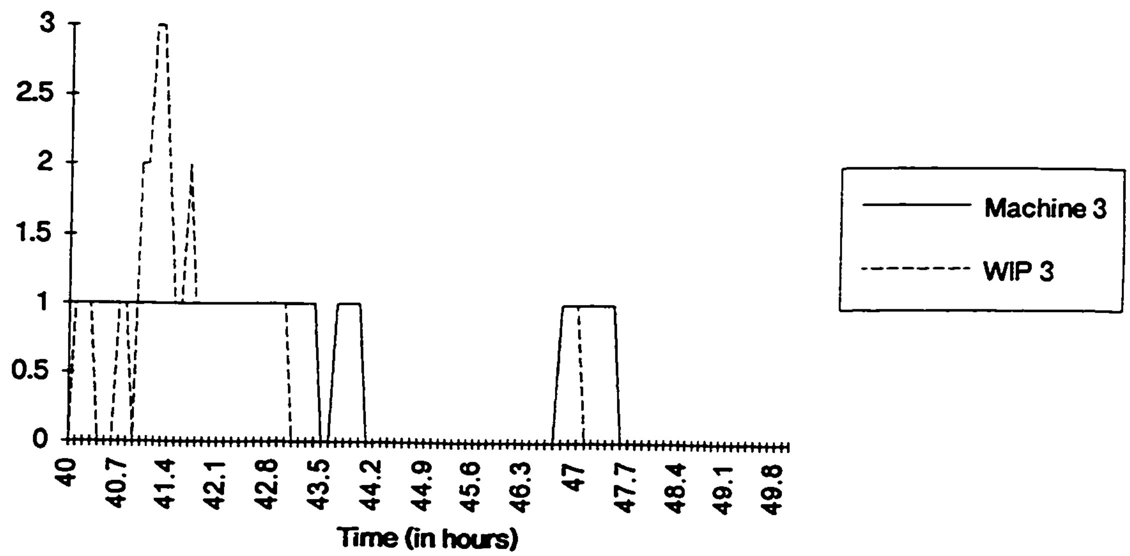


Figure a5.2 Machine/cell state for cell 3.

Appendix B1

PAC OPTIMIZATION: FORTRAN CODE WITH DESCRIPTION OF VARIABLES

Main Programs: **OPT.FOR, OPK.FOR, OPLFOR**

Subprograms and Declaration Files:

Subprogram	Purpose
OSIMLIBG	Set of subroutines of SIMLIBG adjusted for PAC optimization model
OPAC	Main routine for PAC simulation, i.e. cost evaluation of the described by a given setting of PAC parameters manufacturing configuration.
OORDARR(NEW)[1]	Processes arrival of an order tag, where NEW=1 if this is a new order from a customer, type 1 events, and NEW=2 if the order tag comes from a cell (called by OPACARR), type 8 events; NEW is an INTEGER
OREQARR[1][2]	Processes type 2 events
OPACARR[1]	Processes type 3 events
OWIPARR[1]	Processes type 4 events
ODEPART[1]	Processes type 5 events
OPPARR[1][2]	Processes type 6 events
OREPORT[2]	Calculates the total cost per each simulation run. Generates simulation report for the best solution found, when the optimization ends.

[1] diroutine is exactly the same as the original PAC simulation routine, except it uses *opac.dcl* instead of *pac.dcl*

[2] different for DCI and DCII cost calculation approach

Declaration Files	Purpose
OPAC.DCL	Declaration of COMMON variables for our MODEL and OSIMLIBG
PARAM.DCL	Parameters setting for OSIMLIBG
OSYSTEM.DCL	Parameters setting for PAC Model

FORTRAN Variables (additional):

Variable	Definition
Input parameters:	
DATE	Date of the performed optimization, e.g. Dec.05.1996
MAXEV	Max. number of function evaluations (=PAC simulations)
MODEL	Number of tested model, e.g. 1,2,3, etc.
POLICY	Name of policy, e.g. General, MRP, Kanban,etc.
SI(I)	Step length of the parameter I
STEP	Step length of the parameter *z*
XMX	Maximum value of the parameter *z*
XMAX(I)	Maximum value of the parameter I
XMIN(I)	Minimum value of the parameter I
XK(I)	Initial number of process tags per product type I
XR(I)	Initial number of product units in batch per product type I sent as PA cards
XT(I)	Initial value of time delay between arriving of order tag and requisition tag for product type I
XZ(I)	Initial value of the inventory per product type I at the store
Simulated system parameters:	
MX	Total number of PAC parameters : 4*MP
Modeling variables:	
ASSBATCH(I,J)	Number of units of assembly J per product type I
ASSNAME(I,J)	Name of assembly J per product type I
B(I)	Help variable for parameter values at base point
BS	Base point help variable
COND	Help variable to calculate r-k condition
FB	Help variable for total cost value
FE	Current number of iterations
FI	Help variable for total cost value
GENNUM	Help variable for starting with the same random variable on each new PAC simulation
KPAR(I)	Help variable for parameter *k*
KPRM(I)	Help variable for parameter *k*
N	Total number of parameters for optimization
NASS(I)	Number of different assembly per product type I
P(I)	Help variable for parameter values at pattern point

PP	Help variable for product number
PR	Help variable for product number
PS	Pattern point help variable
RPAR	Help variable for parameter r
RR	Help variable for parameter r
S(I)	SI(I) as REAL variable
TAU	Help variable for parameter τ
X(I)	Help variable for parameter values
XMAXB(I)	Help variable for maximum value of the parameter I
XMAXS(I)	Maximum value of the parameter I used for simulation during optimization process
XMINB(I)	Help variable for minimum value of the parameter I
XMINS(I)	Minimum value of the parameter I used for simulation during optimization process
XX(I)	Help variable for parameter values
Y(I)	Help variable for parameter values
Z	Help variable for total cost value
ZZ	Help variable for total cost value

FORTRAN Code

Declaration Files:

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C******************************************************************C
C*  OSYSTEM.DCL      : System setting -PAC Model                   *C
C*  Author           : Krystyna  Bielunska                         *C
C*  Date             : May 1996                (Model 4)           *C
C******************************************************************C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      INTEGER MU,MP,MG,MS,MB,MX

*     Set values for the system:
*     MU      - maximum number of units (i.e. cells/storages)
*     MP      - maximum number of different product types
*               without "raw materials"
*     MG      - maximum number of different product types
*               including "raw materials"
*     MS      - maximum number of different subassembies per assembly
*     MB      - maximum number of different assemblies per subassembly
*               (but only for products, not for "raw materials"
*     MX      - total number of PAC parameters : 4*MP

      PARAMETER(MU=4,MP=7,MG=11,MS=2,MB=3,MX=28)
```

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C*******************************************************************C
C*   OPAC.DCL   : Declarations File for PAC Model/Optimization        *C
C*                (General, BSS, MRP, PTO, LC)                        *C
C*   Author     : Krystyna  Bielunska                                 *C
C*   Date       : May 1996                                            *C
C*******************************************************************C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

$INCLUDE: 'param.dcl'
$INCLUDE: 'osystem.dcl'
      INTEGER NUNITS,NMACHS(MU),NTYPES,FTYPES,ATYPES,RTYPES,
     .        ZV(MP),KV(MP),RV(MP),PRODTYPE,NBUSY(MU),
     .        BATCH(MP),PSERV(MP),MTYPES,DAYS,LASTPR(MU),
     .        NSUBASS(MP),SUBNAME(MP,MS),SUBBATCH(MP,MS),
     .        SCELL(MG),BATCHWIP(MG,MU),ADDRESS,FG,TV(MP)
      REAL LENGTH,WARMUP,MARRVT,MSERVT(MP),PROBD(MP),TRAN(MG,MU),
     .  COSTD(MP),COSTP(MP),COSTW(MG,MU),TC,SETUP(MP),MSA(MP),MSB(MP)
      COMMON /MODEL/ LENGTH,MARRVT,MSERVT,NUNITS,NMACHS,DAYS,
     .           NTYPES,PROBD,ZV,KV,RV,TV,PRODTYPE,
     .           NBUSY,BATCH,WARMUP,PSERV,ATYPES,
     .           FTYPES,RTYPES,NSUBASS,SUBNAME,SUBBATCH,
     .           SCELL,BATCHWIP,ADDRESS,FG,MTYPES,TRAN,
     .           COSTD,COSTP,COSTW,TC,SETUP,LASTPR,MSA,MSB
      INTEGER LDECR,LEVENT,LFIRST,LINCR,LLAST,LRANK(ML),LSIZE(ML),
     &        MAXATR,NEXT
      REAL TIME,TRNSFR(MA)
      COMMON /SIMLIB/ LDECR,LEVENT,LFIRST,LINCR,LLAST,LRANK,LSIZE,
     &            MAXATR,NEXT,TIME,TRNSFR

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C*******************************************************************C
C*   OPT.FOR    : Main Optimization Program for PAC Model             *C
C*                (General, BSS, MRP, PTO, LC)                        *C
C*   Author     : Krystyna  Bielunska                                 *C
C*   Date       : June 1996                                           *C
C*******************************************************************C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      INCLUDE 'opac.dcl'
      INTEGER I,J,K,N,MAXEV,PS,BS,FE,XMIN(MX),XMAX(MX),SI(MX)
      INTEGER X(MX),P(MX),Y(MX),B(MX),XX(MX),XMINB(MX),XMAXB(MX)
      INTEGER RAWS,MODEL,XMINS(MX),XMAXS(MX),NIR,PR,II,RR,KK
      INTEGER NASS(MP),ASSNAME(MP,MB),ASSBATCH(MP,MB)
      INTEGER XZ(MP),XK(MP),XR(MP),XT(MP),PP,III,COND
      REAL Z,FI,FB,ZZ,S(MX)
      REAL*4 S_TIME,E_TIME
      CHARACTER*11 DATE
      CHARACTER*8 POLICY
      COMMON/OPT/FE,Z
      COMMON/PAR/N,X
```

```
      CALL TIME_STATS(S_TIME)

*     Open input and output files.

      OPEN(15,FILE='opt.in')
      OPEN(16,FILE='opt.out')
      FE = 0
      N  = MX

*     Read input parameters for optimization.

      READ(15,FMT='(1A11)') DATE
      READ(15,FMT='(1A8)') POLICY
      READ(15,*) MODEL,MAXEV
      READ(15,*) XMIN(1),XMAX(1),SI(1)
      S(1)=FLOAT(SI(1))
      READ(15,*) XMIN(2),XMAX(2),SI(2)
      S(2)=FLOAT(SI(2))
      READ(15,*) XMIN(3),XMAX(3),SI(3)
      S(3)=FLOAT(SI(3))
      READ(15,*) XMIN(4),XMAX(4),SI(4)
      S(4)=FLOAT(SI(4))

*     Read input parameters for simulation.

      READ(15,*) NUNITS,FTYPES,MTYPES,ATYPES,RTYPES,MARRVT,
     .       LENGTH,WARMUP
      NTYPES=FTYPES+MTYPES+ATYPES
      FG   =NUNITS+1
      DAYS =LENGTH-WARMUP
      READ(15,*)(NMACHS(I),I=1,NUNITS)
      DO 10 I=1,NTYPES
       READ(15,*) XZ(I),XK(I),XR(I),XT(I),MSERVT(I),SCELL(I),
     .        NSUBASS(I),(SUBNAME(I,J) ,J=1,NSUBASS(I)),
     .              (SUBBATCH(I,J),J=1,NSUBASS(I))
       MSA(I)=MSERVT(I)-1.0
       MSB(I)=MSERVT(I)+1.0
   10 CONTINUE
      READ(15,*)(PROBD(I),I=1,FTYPES+MTYPES)
      READ(15,*)RAWS
      DO 15 I=1,RAWS
       READ(15,*)PRODTYPE,STORE,TRAN(PRODTYPE,STORE),
     .       COSTW(PRODTYPE,STORE)
   15 CONTINUE
      DO 20 I=1,FTYPES+MTYPES
       READ(15,*)COSTD(I)
   20 CONTINUE
      DO 25 I=1,NTYPES
       READ(15,*)COSTP(I),SETUP(I)
   25 CONTINUE
```

```
      DO 30 I=1,NTYPES
        NASS(I)=0
   30 CONTINUE
      DO 40 I=1,NTYPES
        DO 35 J=1,NSUBASS(I)
          IF(SUBNAME(I,J).LE.NTYPES) THEN
            NASS(SUBNAME(I,J))=NASS(SUBNAME(I,J))+1
            ASSNAME(SUBNAME(I,J),NASS(SUBNAME(I,J)))=I
            ASSBATCH(SUBNAME(I,J),NASS(SUBNAME(I,J)))=SUBBATCH(I,J)
          END IF
   35   CONTINUE
   40 CONTINUE
      DO 50 I=5,N,4
        XMIN(I)= XMIN(1)
        XMAX(I)= XMAX(1)
        S(I)  = S(1)
        SI(I) = SI(1)
   50 CONTINUE
      DO 60 I=6,N,4
        XMIN(I)= XMIN(2)
        XMAX(I)= XMAX(2)
        S(I)  = S(2)
        SI(I) = SI(2)
   60 CONTINUE
      DO 70 I=7,N,4
        XMIN(I)= XMIN(3)
        XMAX(I)= XMAX(3)
        S(I)  = S(3)
        SI(I) = SI(3)
   70 CONTINUE
      DO 80 I=8,N,4
        XMIN(I)= XMIN(4)
        XMAX(I)= XMAX(4)
        S(I)  = S(4)
        SI(I) = SI(4)
   80 CONTINUE
      K=1
      DO 90 I=1,N,4
        X(I)   = XZ(K)
        X(I+1) = XK(K)
        X(I+2) = XR(K)
        X(I+3) = XT(K)
        K      = K+1
   90 CONTINUE
      DO 100 I=1,N
        XMINB(I)=XMIN(I)
        XMAXB(I)=XMAX(I)
        Y(I)   =X(I)
        P(I)   =X(I)
        B(I)   =X(I)
        XMAXS(I)=X(I)
```

```
     XMINS(I)=X(I)
  100 CONTINUE
    WRITE(16,2010)
 2010 FORMAT('*************************************************************',
    .    '*******************'/
    .    '*  OPTIMIZATION OF MULTIPLE-CELL SYSTEM COORDINATED BY',
    .    ' PA CARDS (PAC)   *'/
    .    '*  Developed by: Krystyna Bielunska, Ind. Eng., TUN',
    .    'S, May 1996        *'/
    .    '*******************************************************',
    .    '*******************'/
    .    ' scenario: exponential service times'/
    .    '          single cost'/)
    K=N/4
    WRITE(16,2011)MODEL,POLICY,DATE,K
 2011 FORMAT( ' Parameters setting',' Model: ',I2,5X,'Policy: ',A8,4X,
    .     ' Date: ',A11//
    .     ' Product: 1 - ',I5,12X,
    .     ' z-value  k-value  r-value  t-value'/
    .     ' ===================',12X,
    .     ' =======  =======  =======  ======')
    WRITE(16,2013) XMIN(1),XMIN(2),XMIN(3),XMIN(4)
 2013 FORMAT(/' Minimum values:',15X,I5,4X,I5,4X,I5,4X,I5)
    WRITE(16,2014) XMAX(1),XMAX(2),XMAX(3),XMAX(4)
 2014 FORMAT(/' Maximum values:',15X,I5,4X,I5,4X,I5,4X,I5)
    WRITE(16,2015) SI(1),SI(2),SI(3),SI(4)
 2015 FORMAT(/' Step length:   ',15X,I5,4X,I5,4X,I5,4X,I5)
    WRITE(16,2016)
 2016 FORMAT(/' Initial values per product')
    K=1
    DO 110 I=1,N,4
     WRITE(16,2017) K,X(I),X(I+1),X(I+2),X(I+3)
 2017  FORMAT(22X,I3,7X,I5,4X,I5,4X,I5,4X,I5)
     K=K+1
  110 CONTINUE
    WRITE(16,2018)MAXEV
 2018 FORMAT(/' Maximum number of cost evaluations:',I10/)
    CALL PACPAR
    CALL FUNEVAL
    DO 170 I=1,N
     IF(X(I).GT.XMAXS(I)) THEN
      XMAXS(I) = X(I)
     END IF
     IF(X(I).LT.XMINS(I)) THEN
      XMINS(I) = X(I)
     END IF
  170 CONTINUE
    FI=Z
    WRITE(16,2019)FI
 2019 FORMAT(/' Initial total cost              :',F20.2//
    . '=====================================================================')
```

```
     .' OPTIMIZATION STEPS:'/
     .'=================================================================')
      DO 180 I=1,N
        XX(I) = X(I)
  180 CONTINUE
      ZZ = Z
      PS=0
      BS=1

*     Explore about Base Point

      J  = 1
      FB  = FI
  200 IF(FE.GE.MAXEV) THEN
        GOTO 1000
      END IF
      X(J) = Y(J)+S(J)
      IF(X(J).LE.XMAX(J)) THEN

*     Extra check for r-k feasibility conditions.

       IF(POLICY.EQ.'General') THEN
        DO 204 I=3,N,4
         IF(J.EQ.I) THEN
          PR=INT(J/4)+1
          DO 202 II=1,NSUBASS(PR)
           PP=SUBNAME(PR,II)
           IF(NASS(PP).EQ.1) THEN
            RR=SUBNAME(PR,II)*4-1
            IF(RR.GT.N) THEN
             GOTO 202
            END IF
            IF(X(J-1)+INT(X(J)/X(RR))*X(RR).LT.2*X(J)) THEN
             Z=FI+100
             GOTO 210
            END IF
           ELSE
            COND=0
            DO 201 III=1,NASS(PP)
             RR=ASSNAME(PP,III)*4-1
             KK=ASSNAME(PP,III)*4-2
             COND=COND+INT(X(KK)/X(RR))*X(RR)
  201       CONTINUE
            RR=PP*4-1
            IF(COND.LT.X(RR)) THEN
             Z=FI+100
             GOTO 210
            END IF
           END IF
  202     CONTINUE
          IF(PR.GT.FTYPES) THEN
```

```
          IF(NASS(PR).EQ.1) THEN
           RR=ASSNAME(PR,1)*4-1
           KK=ASSNAME(PR,1)*4-2
           IF(X(KK)+INT(X(RR)/X(J))*X(J).LT.2*X(RR)) THEN
            Z=FI+100
            GOTO 210
           END IF
          ELSE
           COND=0
           DO 203 II=1,NASS(PR)
            RR=ASSNAME(PR,II)*4-1
            KK=ASSNAME(PR,II)*4-2
            COND=COND+INT(X(KK)/X(RR))*X(RR)
 203       CONTINUE
           IF(COND.LT.X(J)) THEN
            Z=FI+100
            GOTO 210
           END IF
          END IF
         END IF
        END IF
 204    CONTINUE
      END IF

*     End of specific r-k conditions.

      CALL PACPAR
      CALL FUNEVAL

      DO 205 I=1,N
       IF(X(I).GT.XMAXS(I)) THEN
        XMAXS(I) = X(I)
       END IF
       IF(X(I).LT.XMINS(I)) THEN
        XMINS(I) = X(I)
       END IF
 205  CONTINUE
     ELSE
      Z = FI + 100
     END IF
 210 IF(Z.LT.FI) THEN
      GOTO 280
     END IF
     X(J) = Y(J)-S(J)
     IF(X(J).GE.XMIN(J)) THEN

*    Extra check for r-k feasibility conditions.

      IF(POLICY.EQ.'General') THEN
       DO 214 I=3,N,4
        IF(J.EQ.I) THEN
```

```
          PR=INT(J/4)+1
          DO 212 II=1,NSUBASS(PR)
           PP=SUBNAME(PR,II)
           IF(NASS(PP).EQ.1) THEN
            RR=SUBNAME(PR,II)*4-1
            IF(RR.GT.N) THEN
             GOTO 212
            END IF
            IF(X(J-1)+INT(X(J)/X(RR))*X(RR).LT.2*X(J)) THEN
             Z=FI+100
             GOTO 220
            END IF
           ELSE
            COND=0
            DO 211 III=1,NASS(PP)
             RR=ASSNAME(PP,III)*4-1
             KK=ASSNAME(PP,III)*4-2
             COND=COND+INT(X(KK)/X(RR))*X(RR)
211         CONTINUE
            RR=PP*4-1
            IF(COND.LT.X(RR)) THEN
             Z=FI+100
             GOTO 220
            END IF
           END IF
212       CONTINUE
          IF(PR.GT.FTYPES) THEN
           IF(NASS(PR).EQ.1) THEN
            RR=ASSNAME(PR,1)*4-1
            KK=ASSNAME(PR,1)*4-2
            IF(X(KK)+INT(X(RR)/X(J))*X(J).LT.2*X(RR)) THEN
             Z=FI+100
             GOTO 220
            END IF
           ELSE
            COND=0
            DO 213 II=1,NASS(PR)
             RR=ASSNAME(PR,II)*4-1
             KK=ASSNAME(PR,II)*4-2
             COND=COND+INT(X(KK)/X(RR))*X(RR)
213         CONTINUE
            IF(COND.LT.X(J)) THEN
             Z=FI+100
             GOTO 220
            END IF
           END IF
          END IF
        END IF
214   CONTINUE
      DO 218 I=2,N,4
       IF(J.EQ.I) THEN
```

```
          PR=INT(J/4)+1
          DO 217 II=1,NSUBASS(PR)
           PP=SUBNAME(PR,II)
           IF(NASS(PP).EQ.1) THEN
            RR=SUBNAME(PR,II)*4-1
            IF(RR.GT.N) THEN
             GOTO 217
            END IF
            IF(X(J)+INT(X(J+1)/X(RR))*X(RR).LT.2*X(J+1)) THEN
             Z=FI+100
             GOTO 220
            END IF
           ELSE
            COND=0
            DO 216 III=1,NASS(PP)
             RR=ASSNAME(PP,III)*4-1
             KK=ASSNAME(PP,III)*4-2
             COND=COND+INT(X(KK)/X(RR))*X(RR)
 216        CONTINUE
            RR=PP*4-1
            IF(COND.LT.X(RR)) THEN
             Z=FI+100
             GOTO 220
            END IF
           END IF
 217     CONTINUE
        END IF
 218  CONTINUE
      END IF

*    End of specific r-k conditions.

      CALL PACPAR
      CALL FUNEVAL
      DO 219 I=1,N
       IF(X(I).GT.XMAXS(I)) THEN
        XMAXS(I) = X(I)
       END IF
       IF(X(I).LT.XMINS(I)) THEN
        XMINS(I) = X(I)
       END IF
 219  CONTINUE
     ELSE
      Z = FI + 100
     END IF
 220 IF(Z.LT.FI) THEN
      GOTO 280
     END IF
     X(J) = XX(J)
     Z = ZZ
     GOTO 290
```

```
 280 Y(J) = X(J)
 290 FI  = Z
     WRITE(16,2020)Z,FE
 2020 FORMAT( ' Exploration Step    Cost=',F20.2,5X,'No. of eval.=',I6)
*   . 21X,' Product  z-value  k-value  r-value  t-value'/
*   . 21X,' =======  =======  =======  =======  =======')
*     K=0
*     DO 310 I=1,N,4
*      K=K+1
*      WRITE(16,2112)K,X(I),X(I+1),X(I+2),X(I+3)
 2112  FORMAT(21X,I5,5X,I5,4X,I5,4X,I5,4X,I5)
* 310 CONTINUE
     DO 320 I=1,N
      XX(I) = X(I)
 320 CONTINUE
     ZZ = Z
     IF(J.EQ.N) THEN
      DO 325 I=1,N
       XMIN(I)=XMINB(I)
       XMAX(I)=XMAXB(I)
 325  CONTINUE
      GOTO 360
     END IF
     J = J+1
     DO 330 I=1,N,4
      IF(J.EQ.(I+1).AND.(X(J)-S(J)).LT.(X(J+1))) THEN
       XMINB(J)=XMIN(J)
       XMIN(J) = X(J+1)
      END IF
      IF(J.EQ.(I+2).AND.(X(J)+S(J)).GT.(X(J-1))) THEN
       XMAXB(J)=XMAX(J)
       XMAX(J) = X(J-1)
      END IF
 330 CONTINUE
     GOTO 200

*    After 360 make a pattern move if function has been reduced.

 360 IF(FI.LT.(FB-(1.0E-08))) THEN
      GOTO 540
     END IF

*    But if exploration was about a pattern point
*    and no reduction was made change base at 420,
*    otherwise reduce step length at 490.

     IF(PS.EQ.1.AND.BS.EQ.0) THEN
      GOTO 420
     END IF
     GOTO 490
```

```
420 DO 430 I=1,N
     P(I) = B(I)
     Y(I) = B(I)
     X(I) = B(I)
430 CONTINUE
    CALL PACPAR
    CALL FUNEVAL
    DO 435 I=1,N
     IF(X(I).GT.XMAXS(I)) THEN
      XMAXS(I) = X(I)
     END IF
     IF(X(I).LT.XMINS(I)) THEN
      XMINS(I) = X(I)
     END IF
435 CONTINUE
    BS = 1
    PS = 0
    FI = Z
    FB = Z
    WRITE(16,2030)Z,FE
2030 FORMAT( ' Base Change        Cost=',F20.2,5X,'No. of eval.=',I6/
   .  21X,' Product  z-value  k-value  r-value  t-value'/
   .  21X,' =======  =======  =======  =======  =======')
    K=0
    DO 440 I=1,N,4
     K=K+1
     WRITE(16,2112)K,X(I),X(I+1),X(I+2),X(I+3)
440 CONTINUE
    DO 450 I=1,N
     XX(I) = X(I)
450 CONTINUE
    ZZ = Z
    J = 1
    GOTO 200
490 K = 0
    DO 495 I=1,N
     IF(SI(I).EQ.1) THEN
      K=K+1
     END IF
495 CONTINUE
    IF(K.EQ.N) THEN
     GOTO 700
    END IF
    DO 500 I=1,N
     S(I)=S(I)/SI(I)
500 CONTINUE
    WRITE(16,2040)
2040 FORMAT(' Contract Step Length')
    DO 510 I=1,N
     IF(S(I).LT.1) THEN
      GOTO 700
```

```
      END IF
  510 CONTINUE
      J = 1
      GOTO 200

*     Pattern Move

  540 DO 600 I=1,N
       P(I)=2*Y(I)-B(I)
       IF(P(I).GT.XMAX(I)) THEN
        P(I) = XMAX(I)
       END IF
       IF(P(I).LT.XMIN(I)) THEN
        P(I) = XMIN(I)
       END IF
       DO 550 J=1,N,4
        IF(P(J+1).LT.P(J+2)) THEN
         P(J+2)=P(J+1)
        END IF
 550   CONTINUE
       B(I)=Y(I)
       X(I)=P(I)
       Y(I)=X(I)
 600 CONTINUE
     CALL PACPAR
     CALL FUNEVAL
     DO 605 I=1,N
      IF(X(I).GT.XMAXS(I)) THEN
       XMAXS(I) = X(I)
      END IF
      IF(X(I).LT.XMINS(I)) THEN
       XMINS(I) = X(I)
      END IF
 605 CONTINUE
     FB = FI
     PS = 1
     BS = 0
     FI = Z
     WRITE(16,2050)Z,FE
2050 FORMAT( ' Pattern Move        Cost=',F20.2,5X,'No. of eval.=',I6/
   .  21X,' Product  z-value  k-value  r-value  t-value'/
   .  21X,' =======  =======  =======  =======  =======')
     K=0
     DO 610 I=1,N,4
      K=K+1
      WRITE(16,2112)K,X(I),X(I+1),X(I+2),X(I+3)
 610 CONTINUE
     DO 620 I=1,N
      XX(I) = X(I)
 620 CONTINUE
     ZZ = Z
```

```
*   Then explore about latest pattern point.

      J = 1
      GOTO 200
  700 WRITE(16,2060)FB,FE
 2060 FORMAT(/' Minimum found:      Cost=',F20.2,5X,'No. of eval.=',
     . I6//,21X,' Product  z-value  k-value  r-value  t-value'/
     .  21X,' =======  =======  =======  =======  =======')
      DO 710 I=1,N
       X(I)=P(I)
  710 CONTINUE
      DO 720 I=1,N
       IF(X(I).GT.XMAXS(I)) THEN
        XMAXS(I)=X(I)
       END IF
       IF(X(I).LT.XMINS(I)) THEN
        XMINS(I) = X(I)
       END IF
  720 CONTINUE
      K=0
      DO 750 I=1,N,4
       K=K+1
       WRITE(16,2112)K,X(I),X(I+1),X(I+2),X(I+3)
  750 CONTINUE
 1000 WRITE(16,2070)
 2070 FORMAT(/' Max. values of parameters used in simulation:',
     .  //,21X,' Product  z-value  k-value  r-value  t-value'/
     .  21X,' =======  =======  =======  =======  =======')
      K=0
      DO 1010 I=1,N,4
       K=K+1
       WRITE(16,2112)K,XMAXS(I),XMAXS(I+1),XMAXS(I+2),XMAXS(I+3)
 1010 CONTINUE
      WRITE(16,2080)
 2080 FORMAT(/' Min. values of parameters used in simulation:',
     .  //,21X,' Product  z-value  k-value  r-value  t-value'/
     .  21X,' =======  =======  =======  =======  =======')
      K=0
      DO 1020 I=1,N,4
       K=K+1
       WRITE(16,2112)K,XMINS(I),XMINS(I+1),XMINS(I+2),XMINS(I+3)
 1020 CONTINUE
      FE=-1
      CALL PACPAR
      CALL FUNEVAL
      CALL TIME_STATS(E_TIME)
      WRITE(16,*)'CPU time:', E_TIME-S_TIME
      CLOSE(15)
      CLOSE(16)
```

```
      STOP
      END

      SUBROUTINE PACPAR

      INCLUDE 'opac.dcl'
      INTEGER I,K
      INTEGER N,X(MX)
      COMMON/PAR/N,X

      K = 0
      DO 10 I=1,N,4
       K   =K+1
       ZV(K)=X(I)
       KV(K)=X(I+1)
       RV(K)=X(I+2)
       TV(K)=X(I+3)
   10 CONTINUE

      RETURN
      END

      SUBROUTINE TIME_STATS(ELAPSED)
      TYPE TB_TYPE
        SEQUENCE
        REAL*4 USRTIME
        REAL*4 SYSTIME
      END TYPE
      TYPE (TB_TYPE) ETIME_STRUCT
      REAL*4 ELAPSED
      ELAPSED = ETIME_(ETIME_STRUCT)
      RETURN
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C*****************************************************************************C
C*   OPK.FOR      : Main Optimization Program for PAC Model               *C
C*                 (Kanban)                                               *C
C*   Author       : Krystyna  Bielunska                                   *C
C*   Date         : June 1996                                             *C
C*****************************************************************************C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      INCLUDE 'opac.dcl'
      INTEGER I,J,K,N,MAXEV,PS,BS,FE,XMIN(MX),XMAX(MX),SI(MX)
      INTEGER X(MX),P(MX),Y(MX),B(MX),XX(MX),XMINB(MX),XMAXB(MX)
      INTEGER RAWS,MODEL,XMINS(MX),XMAXS(MX),NIR,PR,II,RR,KK
      INTEGER NASS(MP),ASSNAME(MP,MB),ASSBATCH(MP,MB),TAU
      INTEGER XZ(MP),XR(MP),PP,III,COND
      REAL Z,FI,FB,ZZ,S(MX)
      REAL*4 S_TIME,E_TIME
```

```
      CHARACTER*11 DATE
      CHARACTER*8 POLICY
      COMMON/OPT/FE,Z
      COMMON/PAR/N,X

      CALL TIME_STATS(S_TIME)

*     Open input and output files.

      OPEN(15,FILE='opk.in')
      OPEN(16,FILE='opk.out')
      FE = 0
      N  = MX/2
      TAU= 0

*     Read input parameters for optimization.

      READ(15,FMT='(1A11)') DATE
      READ(15,FMT='(1A8)') POLICY
      READ(15,*) MODEL,MAXEV
      READ(15,*) XMIN(1),XMAX(1),SI(1)
      S(1)=FLOAT(SI(1))
      READ(15,*) XMIN(2),XMAX(2),SI(2)
      S(2)=FLOAT(SI(2))

*     Read input parameters for simulation.

      READ(15,*) NUNITS,FTYPES,MTYPES,ATYPES,RTYPES,MARRVT,
     .     LENGTH,WARMUP
      NTYPES=FTYPES+MTYPES+ATYPES
      FG   =NUNITS+1
      DAYS =LENGTH-WARMUP
      READ(15,*)(NMACHS(I),I=1,NUNITS)
      DO 10 I=1,NTYPES
       READ(15,*) XZ(I),XR(I),MSERVT(I),SCELL(I),
     .       NSUBASS(I),(SUBNAME(I,J) ,J=1,NSUBASS(I)),
     .             (SUBBATCH(I,J),J=1,NSUBASS(I))
       MSA(I)=MSERVT(I)-1.0
       MSB(I)=MSERVT(I)+1.0
   10 CONTINUE
      READ(15,*)(PROBD(I),I=1,FTYPES+MTYPES)
      READ(15,*)RAWS
      DO 15 I=1,RAWS
       READ(15,*)PRODTYPE,STORE,TRAN(PRODTYPE,STORE),
     .     COSTW(PRODTYPE,STORE)
   15 CONTINUE
      DO 20 I=1,FTYPES+MTYPES
       READ(15,*)COSTD(I)
   20 CONTINUE
      DO 25 I=1,NTYPES
       READ(15,*)COSTP(I),SETUP(I)
```

```
 25 CONTINUE
    DO 30 I=1,NTYPES
      NASS(I)=0
 30 CONTINUE
    DO 40 I=1,NTYPES
      DO 35 J=1,NSUBASS(I)
        IF(SUBNAME(I,J).LE.NTYPES) THEN
          NASS(SUBNAME(I,J))=NASS(SUBNAME(I,J))+1
          ASSNAME(SUBNAME(I,J),NASS(SUBNAME(I,J)))=I
          ASSBATCH(SUBNAME(I,J),NASS(SUBNAME(I,J)))=SUBBATCH(I,J)
        END IF
 35   CONTINUE
 40 CONTINUE
    DO 45 I=1,NTYPES
      DO 44 J=1,NASS(I)
 44   CONTINUE
 45 CONTINUE
    DO 50 I=3,N,2
      XMIN(I)= XMIN(1)
      XMAX(I)= XMAX(1)
      S(I)  = S(1)
      SI(I) = SI(1)
 50 CONTINUE
    DO 60 I=4,N,2
      XMIN(I)= XMIN(2)
      XMAX(I)= XMAX(2)
      S(I)  = S(2)
      SI(I) = SI(2)
 60 CONTINUE
    K=1
    DO 90 I=1,N,2
      X(I)   = XZ(K)
      X(I+1) = XR(K)
      K      = K+1
 90 CONTINUE
    DO 100 I=1,N
      XMINB(I)=XMIN(I)
      XMAXB(I)=XMAX(I)
      Y(I)   =X(I)
      P(I)   =X(I)
      B(I)   =X(I)
      XMAXS(I)=X(I)
      XMINS(I)=X(I)
100 CONTINUE
    WRITE(16,2010)
2010 FORMAT('************************************************************',
   .     '*******************'/
   .     '*  OPTIMIZATION OF MULTIPLE-CELL SYSTEM COORDINATED BY',
   .     ' PA CARDS (PAC)  *'/
   .     '*  Developed by: Krystyna Bielunska, Ind. Eng., TUN',
   .     'S, May 1996        *'/
```

```
     .      '*****************************************************',
     .      '*******************'/
     .      ' scenario: exponential service times'/
     .      '          :single cost'/)
      K=N/2
      WRITE(16,2011)MODEL,POLICY,DATE,K
 2011 FORMAT( ' Parameters setting',' Model: ',I2,5X,'Policy: ',A8,4X,
     .      ' Date: ',A11//
     .      ' Product: 1 - ',I5,12X,
     .      ' z-value  k-value  r-value  t-value'/
     .      ' ==================',12X,
     .      ' ======= ======= ======= =======')
      WRITE(16,2013) XMIN(1),XMIN(1),XMIN(2),TAU
 2013 FORMAT(/' Minimum values:',15X,I5,4X,I5,4X,I5,4X,I5)
      WRITE(16,2014) XMAX(1),XMAX(1),XMAX(2),TAU
 2014 FORMAT(/' Maximum values:',15X,I5,4X,I5,4X,I5,4X,I5)
      WRITE(16,2015) SI(1),SI(1),SI(2),TAU
 2015 FORMAT(/' Step length:   ',15X,I5,4X,I5,4X,I5,4X,I5)
      WRITE(16,2016)
 2016 FORMAT(/' Initial values per product')
      K=1
      DO 110 I=1,N,2
       WRITE(16,2017) K,X(I),X(I),X(I+1),TAU
 2017  FORMAT(22X,I3,7X,I5,4X,I5,4X,I5,4X,I5)
       K=K+1
  110 CONTINUE
      WRITE(16,2018)MAXEV
 2018 FORMAT(/' Maximum number of cost evaluations:',I10/)
      CALL PACPAR
      CALL FUNEVAL
      DO 170 I=1,N
       IF(X(I).GT.XMAXS(I)) THEN
        XMAXS(I) = X(I)
       END IF
       IF(X(I).LT.XMINS(I)) THEN
        XMINS(I) = X(I)
       END IF
  170 CONTINUE
      FI=Z
      WRITE(16,2019)FI
 2019 FORMAT(/' Initial total cost              :',F20.2//
     . '=================================================================='/
     . ' OPTIMIZATION STEPS:'/
     . '==================================================================')
      DO 180 I=1,N
       XX(I) = X(I)
  180 CONTINUE
      ZZ = Z

      PS=0
      BS=1
```

```
*     Explore about Base Point

      J   = 1
      FB  = FI
 190  IF(FE.GE.MAXEV) THEN
        GOTO 1000
      END IF
      DO 200 I=1,N,2
       IF(J.EQ.(I).AND.(X(J)-S(J)).LT.(X(J+1))) THEN
        XMINB(J)=XMIN(J)
        XMIN(J) = X(J+1)
       END IF
       IF(J.EQ.(I+1).AND.(X(J)+S(J)).GT.(X(J-1))) THEN
        XMAXB(J)=XMAX(J)
        XMAX(J) = X(J-1)
       END IF
 200  CONTINUE
      X(J) = Y(J)+S(J)
      IF(X(J).LE.XMAX(J)) THEN

*     Extra check for r-k feasibility conditions.

       IF(POLICY.EQ.'Kanban') THEN
        DO 204 I=2,N,2
         IF(J.EQ.I) THEN
          PR=INT(J/2)+1
          DO 202 II=1,NSUBASS(PR)
           PP=SUBNAME(PR,II)
           IF(NASS(PP).EQ.1) THEN
            RR=SUBNAME(PR,II)*2
            IF(RR.GT.N) THEN
             GOTO 202
            END IF
            IF(X(J-1)+INT(X(J)/X(RR))*X(RR).LT.2*X(J)) THEN
             Z=FI+100
             GOTO 210
            END IF
           ELSE
            COND=0
            DO 201 III=1,NASS(PP)
             RR=ASSNAME(PP,III)*2
             KK=ASSNAME(PP,III)*2-1
             COND=COND+INT(X(KK)/X(RR))*X(RR)
 201        CONTINUE
            RR=PP*2
            IF(COND.LT.X(RR)) THEN
             Z=FI+100
             GOTO 210
            END IF
           END IF
```

```
 202      CONTINUE
          IF(PR.GT.FTYPES) THEN
           IF(NASS(PR).EQ.1) THEN
            RR=ASSNAME(PR,1)*2
            KK=ASSNAME(PR,1)*2-1
            IF(X(KK)+INT(X(RR)/X(J))*X(J).LT.2*X(RR)) THEN
             Z=FI+100
             GOTO 210
            END IF
           ELSE
            COND=0
            DO 203 II=1,NASS(PR)
             RR=ASSNAME(PR,II)*2
             KK=ASSNAME(PR,II)*2-1
             COND=COND+INT(X(KK)/X(RR))*X(RR)
 203        CONTINUE
            IF(COND.LT.X(J)) THEN
             Z=FI+100
             GOTO 210
            END IF
           END IF
          END IF
         END IF
 204    CONTINUE
       END IF

*     End of specific r-k conditions.

       CALL PACPAR
       CALL FUNEVAL

       DO 205 I=1,N
        IF(X(I).GT.XMAXS(I)) THEN
         XMAXS(I) = X(I)
        END IF
        IF(X(I).LT.XMINS(I)) THEN
         XMINS(I) = X(I)
        END IF
 205   CONTINUE
      ELSE
       Z = FI + 100
      END IF
 210 IF(Z.LT.FI) THEN
       GOTO 280
      END IF
      X(J) = Y(J)-S(J)
      IF(X(J).GE.XMIN(J)) THEN

*     Extra check for r-k feasibility conditions.

       IF(POLICY.EQ.'Kanban') THEN
```

```
      DO 214 I=2,N,2
       IF(J.EQ.I) THEN
        PR=INT(J/2)+1
        DO 212 II=1,NSUBASS(PR)
         PP=SUBNAME(PR,II)
         IF(NASS(PP).EQ.1) THEN
          RR=SUBNAME(PR,II)*2
          IF(RR.GT.N) THEN
           GOTO 212
          END IF
          IF(X(J-1)+INT(X(J)/X(RR))*X(RR).LT.2*X(J)) THEN
           Z=FI+100
           GOTO 220
          END IF
         ELSE
          COND=0
          DO 211 III=1,NASS(PP)
           RR=ASSNAME(PP,III)*2
           KK=ASSNAME(PP,III)*2-1
           COND=COND+INT(X(KK)/X(RR))*X(RR)
 211      CONTINUE
          RR=PP*2
          IF(COND.LT.X(RR)) THEN
           Z=FI+100
           GOTO 220
          END IF
         END IF
 212    CONTINUE
        IF(PR.GT.FTYPES) THEN
         IF(NASS(PR).EQ.1) THEN
          RR=ASSNAME(PR,1)*2
          KK=ASSNAME(PR,1)*2-1
          IF(X(KK)+INT(X(RR)/X(J))*X(J).LT.2*X(RR)) THEN
           Z=FI+100
           GOTO 220
          END IF
         ELSE
          COND=0
          DO 213 II=1,NASS(PR)
           RR=ASSNAME(PR,II)*2
           KK=ASSNAME(PR,II)*2-1
           COND=COND+INT(X(KK)/X(RR))*X(RR)
 213      CONTINUE
          IF(COND.LT.X(J)) THEN
           Z=FI+100
           GOTO 220
          END IF
         END IF
        END IF
       END IF
 214  CONTINUE
```

```
        DO 218 I=1,N,2
         IF(J.EQ.I) THEN
          PR=INT(J/2)+1
          DO 217 II=1,NSUBASS(PR)
           PP=SUBNAME(PR,II)
           IF(NASS(PP).EQ.1) THEN
            RR=SUBNAME(PR,II)*2
            IF(RR.GT.N) THEN
             GOTO 217
            END IF
            IF(X(J)+INT(X(J+1)/X(RR))*X(RR).LT.2*X(J+1)) THEN
             Z=FI+100
             GOTO 220
            END IF
           ELSE
            COND=0
            DO 216 III=1,NASS(PP)
             RR=ASSNAME(PP,III)*2
             KK=ASSNAME(PP,III)*2-1
             COND=COND+INT(X(KK)/X(RR))*X(RR)
 216        CONTINUE
            RR=PP*2
            IF(COND.LT.X(RR)) THEN
             Z=FI+100
             GOTO 220
            END IF
           END IF
 217      CONTINUE
         END IF
 218  CONTINUE
      END IF

*     End of specific r-k conditions.

      CALL PACPAR
      CALL FUNEVAL
      DO 219 I=1,N
       IF(X(I).GT.XMAXS(I)) THEN
        XMAXS(I) = X(I)
       END IF
       IF(X(I).LT.XMINS(I)) THEN
        XMINS(I) = X(I)
       END IF
 219  CONTINUE
     ELSE
      Z = FI + 100
     END IF
 220 IF(Z.LT.FI) THEN
      GOTO 280
     END IF
     X(J) = XX(J)
```

```
      Z = ZZ
      GOTO 290
  280 Y(J) = X(J)
  290 FI  = Z
      WRITE(16,2020)Z,FE
 2020 FORMAT( ' Exploration Step     Cost=',F20.2,5X,'No. of eval.=',I6)
*   .  21X,' Product  z-value  k-value  r-value  t-value'/
*   .  21X,' ======  ======  ======  ======  ======')
*    K=0
*    DO 310 I=1,N,2
*     K=K+1
*     WRITE(16,2112)K,X(I),X(I),X(I+1),TAU
 2112  FORMAT(21X,I5,5X,I5,4X,I5,4X,I5,4X,I5)
* 310 CONTINUE
      DO 320 I=1,N
       XX(I) = X(I)
  320 CONTINUE
      ZZ = Z
      IF(J.EQ.N) THEN
       DO 325 I=1,N
        XMIN(I)=XMINB(I)
        XMAX(I)=XMAXB(I)
  325  CONTINUE
       GOTO 360
      END IF
      J = J+1
      GOTO 190

*    After 360 make a pattern move if function has been reduced.

  360 IF(FI.LT.(FB-(1.0E-08))) THEN
       GOTO 540
      END IF

*    But if exploration was about a pattern point
*    and no reduction was made change base at 420,
*    otherwise reduce step length at 490.

      IF(PS.EQ.1.AND.BS.EQ.0) THEN
       GOTO 420
      END IF
      GOTO 490
  420 DO 430 I=1,N
       P(I) = B(I)
       Y(I) = B(I)
       X(I) = B(I)
  430 CONTINUE
      CALL PACPAR
      CALL FUNEVAL
      DO 435 I=1,N
       IF(X(I).GT.XMAXS(I)) THEN
```

```
        XMAXS(I) = X(I)
       END IF
       IF(X(I).LT.XMINS(I)) THEN
        XMINS(I) = X(I)
       END IF
 435 CONTINUE
     BS = 1
     PS = 0
     FI = Z
     FB = Z
     WRITE(16,2030)Z,FE
2030 FORMAT( ' Base Change        Cost=',F20.2,5X,'No. of eval.=',I6/
    .  21X,' Product  z-value  k-value  r-value  t-value'/
    .  21X,' =======  =======  =======  =======  =======')
     K=0
     DO 440 I=1,N,2
      K=K+1
      WRITE(16,2112)K,X(I),X(I),X(I+1),TAU
 440 CONTINUE
     DO 450 I=1,N
      XX(I) = X(I)
 450 CONTINUE
     ZZ = Z
     J = 1
     GOTO 190
 490 K = 0
     DO 495 I=1,N
      IF(SI(I).EQ.1) THEN
       K=K+1
      END IF
 495 CONTINUE
     IF(K.EQ.N) THEN
      GOTO 700
     END IF
     DO 500 I=1,N
      S(I)=S(I)/SI(I)
 500 CONTINUE
     WRITE(16,2040)
2040 FORMAT(' Contract Step Length')
     DO 510 I=1,N
      IF(S(I).LT.1) THEN
       GOTO 700
      END IF
 510 CONTINUE
     J = 1
     GOTO 190

*    Pattern Move

 540 DO 600 I=1,N
      P(I)=2*Y(I)-B(I)
```

```
      IF(P(I).GT.XMAX(I)) THEN
       P(I) = XMAX(I)
      END IF
      IF(P(I).LT.XMIN(I)) THEN
       P(I) = XMIN(I)
      END IF
      DO 550 J=1,N,4
       IF(P(J+1).LT.P(J+2)) THEN
        P(J+2)=P(J+1)
       END IF
 550  CONTINUE
      B(I)=Y(I)
      X(I)=P(I)
      Y(I)=X(I)
 600 CONTINUE
    CALL PACPAR
    CALL FUNEVAL
    DO 605 I=1,N
     IF(X(I).GT.XMAXS(I)) THEN
      XMAXS(I) = X(I)
     END IF
     IF(X(I).LT.XMINS(I)) THEN
      XMINS(I) = X(I)
     END IF
 605 CONTINUE
    FB = FI
    PS = 1
    BS = 0
    FI = Z
    WRITE(16,2050)Z,FE
2050 FORMAT( ' Pattern Move        Cost=',F20.2,5X,'No. of eval.=',I6/
   .  21X,' Product  z-value  k-value  r-value  t-value'/
   .  21X,' =======  =======  =======  =======  =======')
    K=0
    DO 610 I=1,N,2
     K=K+1
     WRITE(16,2112)K,X(I),X(I),X(I+1),TAU
 610 CONTINUE

    DO 620 I=1,N
     XX(I) = X(I)
 620 CONTINUE
    ZZ = Z

*   Then explore about latest pattern point.

    J = 1
    GOTO 190
 700 WRITE(16,2060)FB,FE
2060 FORMAT(/' Minimum found:      Cost=',F20.2,5X,'No. of eval.=',
   .  I6//,21X,' Product  z-value  k-value  r-value  t-value'/
```

```
     .  21X,' ====== ====== ====== ====== ======')
      DO 710 I=1,N
       X(I)=P(I)
  710 CONTINUE
      DO 720 I=1,N
       IF(X(I).GT.XMAXS(I)) THEN
        XMAXS(I)=X(I)
       END IF
       IF(X(I).LT.XMINS(I)) THEN
        XMINS(I) = X(I)
       END IF
  720 CONTINUE
      K=0
      DO 750 I=1,N,2
       K=K+1
       WRITE(16,2112)K,X(I),X(I),X(I+1),TAU
  750 CONTINUE
 1000 WRITE(16,2070)
 2070 FORMAT(/' Max. values of parameters used in simulation:',
     .  //,21X,' Product  z-value  k-value  r-value  t-value'/
     .  21X,' ====== ====== ====== ====== ======')
      K=0
      DO 1010 I=1,N,2
       K=K+1
       WRITE(16,2112)K,XMAXS(I),XMAXS(I),XMAXS(I+1),TAU
 1010 CONTINUE
      WRITE(16,2080)
 2080 FORMAT(/' Min. values of parameters used in simulation:',
     .  //,21X,' Product  z-value  k-value  r-value  t-value'/
     .  21X,' ====== ====== ====== ====== ======')
      K=0
      DO 1020 I=1,N,2
       K=K+1
       WRITE(16,2112)K,XMINS(I),XMINS(I),XMINS(I+1),TAU
 1020 CONTINUE
      FE=-1
      CALL PACPAR
      CALL FUNEVAL
      CALL TIME_STATS(E_TIME)
      WRITE(16,*)'CPU time:', E_TIME-S_TIME

      CLOSE(15)
      CLOSE(16)

      STOP
      END

      SUBROUTINE PACPAR

      INCLUDE 'opac.dcl'
      INTEGER I,K
```

```
      INTEGER N,X(MX)
      COMMON/PAR/N,X

      K = 0
      DO 10 I=1,N,2
       K   =K+1
       ZV(K)=X(I)
       KV(K)=X(I)
       RV(K)=X(I+1)
       TV(K)=0
   10 CONTINUE

      RETURN
      END

      SUBROUTINE TIME_STATS(ELAPSED)
      (see OPT.FOR)


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C*********************************************************************C
C*   OPI.FOR       : Main Optimization Program for PAC Model          *C
C*                  (IC, CONWIP)                                       *C
C*   Author        : Krystyna  Bielunska                               *C
C*   Date          : July 1996                                         *C
C*********************************************************************C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      INCLUDE 'opac.dcl'
      INTEGER I,J,N,MAXEV,PS,BS,FE,XMIN(MP),XMAX(MP),SI(MP),XMINS(MP)
      INTEGER X(MP),P(MP),Y(MP),B(MP),XMINB(MP),XMAXB(MP),XMAXS(MP)
      INTEGER KPAR(MP),KPRM(MP),RPAR,MODEL,XX(MP),TAU,XMX,STEP,JJ
      INTEGER NASS(MP),ASSNAME(MP,MB),ASSBATCH(MP,MB)
      REAL Z,FI,FB,ZZ,S(MP)
      REAL*4 S_TIME,E_TIME
      CHARACTER*11 DATE
      CHARACTER*8 POLICY
      COMMON/OPT/FE,Z
      COMMON/PAR/N,X,NASS,ASSNAME,ASSBATCH

      CALL TIME_STATS(S_TIME)

*     Open input and output files.

      OPEN(15,FILE='opi.in')
      OPEN(16,FILE='opi.out')
      FE  = 0
      TAU = 0
      RPAR= 1
      N   = MP
```

```
*   Read input parameters for optimization.
    READ(15,FMT='(1A11)') DATE
    READ(15,FMT='(1A8)') POLICY
    READ(15,*)MODEL,MAXEV
    READ(15,*) XMX,STEP

*   Read input parameters for simulation.
    READ(15,*) NUNITS,FTYPES,MTYPES,ATYPES,RTYPES,MARRVT,
   .      LENGTH,WARMUP
    NTYPES=FTYPES+MTYPES+ATYPES
    FG   =NUNITS+1
    DAYS =LENGTH-WARMUP
    READ(15,*)(NMACHS(I),I=1,NUNITS)
    DO 10 I=1,NTYPES
     READ(15,*) X(I),MSERVT(I),SCELL(I),
   .       NSUBASS(I),(SUBNAME(I,J) ,J=1,NSUBASS(I)),
   .            (SUBBATCH(I,J),J=1,NSUBASS(I))
     MSA(I)=MSERVT(I)-1.0
     MSB(I)=MSERVT(I)+1.0
 10 CONTINUE
    READ(15,*)(PROBD(I),I=1,FTYPES+MTYPES)
    READ(15,*)RAWS
    DO 15 I=1,RAWS
     READ(15,*)PRODTYPE,STORE,TRAN(PRODTYPE,STORE),
   .      COSTW(PRODTYPE,STORE)
 15 CONTINUE
    DO 20 I=1,FTYPES+MTYPES
     READ(15,*)COSTD(I)
 20 CONTINUE
    DO 25 I=1,NTYPES
     READ(15,*)COSTP(I),SETUP(I)
 25 CONTINUE
    DO 30 I=1,NTYPES
     NASS(I)=0
 30 CONTINUE
    DO 40 I=1,NTYPES
     DO 35 J=1,NSUBASS(I)
      IF(SUBNAME(I,J).LE.NTYPES) THEN
       NASS(SUBNAME(I,J))=NASS(SUBNAME(I,J))+1
       ASSNAME(SUBNAME(I,J),NASS(SUBNAME(I,J)))=I
       ASSBATCH(SUBNAME(I,J),NASS(SUBNAME(I,J)))=SUBBATCH(I,J)
      END IF
 35  CONTINUE
 40 CONTINUE
    DO 50 I=1,FTYPES+MTYPES
     XMIN(I)= 1
     XMAX(I)= XMX
     SI(I)  = STEP
     S(I)   = FLOAT(STEP)
 50 CONTINUE
    DO 60 I=FTYPES+MTYPES+1,N
```

```
      XMIN(I)= 0
      IF(POLICY.EQ.'IC') THEN
       XMAX(I)= XMX
      ELSE
       XMAX(I)= 0
      END IF
      SI(I)  = STEP
      S(I)   = FLOAT(STEP)
   60 CONTINUE
     DO 100 I=1,N
      XMINB(I)=XMIN(I)
      XMAXB(I)=XMAX(I)
      Y(I)   =X(I)
      P(I)   =X(I)
      B(I)   =X(I)
      XMAXS(I)=X(I)
      XMINS(I)=X(I)
  100 CONTINUE
     WRITE(16,2010)
 2010 FORMAT('***********************************************************',
     .     '********************'/
     .     '*  OPTIMIZATION OF MULTIPLE-CELL SYSTEM COORDINATED BY',
     .     ' PA CARDS (PAC)   *'/
     .     '*  Developed by: Krystyna Bielunska, Ind. Eng., TUN',
     .     'S, May 1996          *'/
     .     '***********************************************************',
     .     '********************'/
     .     ' scenario: exponential service times'/
     .     '           single cost'/)
     WRITE(16,2011)MODEL,POLICY,DATE
 2011 FORMAT( ' Parameters setting',' Model: ',I2,5X,'Policy: ',A8,4X,
     .     ' Date: ',A11//
     .     ' Initial values:    ',
     .     ' Product  z-value  k-value  r-value  t-value'/
     .  21X,' =======  =======  =======  =======  =======')
     DO 110 I=1,N
      KPAR(I)=X(I)
      KPRM(I)=XMIN(I)
  110 CONTINUE
     DO 130 I=FTYPES+1,N
      DO 120 JJ=1,NASS(I)
       KPAR(I)=KPAR(I)+KPAR(ASSNAME(I,JJ))*ASSBATCH(I,JJ)
       KPRM(I)=KPRM(I)+KPRM(ASSNAME(I,JJ))*ASSBATCH(I,JJ)
  120 CONTINUE
  130 CONTINUE
     DO 150 I=1,N
      WRITE(16,2012)I,X(I),KPAR(I),RPAR,TAU
 2012 FORMAT(21X,I5,5X,I5,4X,I5,4X,I5,4X,I5)
  150 CONTINUE
     WRITE(16,2013)
 2013 FORMAT(/' Minimum values:    ',
```

```
     .      'Product  z-value  k-value  r-value  t-value'/
     .   21X,' =======  =======  =======  =======  =======')
        DO 160 I=1,N
          WRITE(16,2012)I,XMIN(I),KPRM(I),RPAR,TAU
  160   CONTINUE
        WRITE(16,2014)
 2014 FORMAT(/' Maximum values:    ',
     .      'Product  z-value  k-value  r-value  t-value'/
     .   21X,' =======  =======  =======  =======  =======')
        WRITE(16,2112)N,XMX
 2112 FORMAT(22X,'1-',I5,2X,I5)
        WRITE(16,2015)
 2015 FORMAT(/' Step length:       ',
     .      'Product  z-value  k-value  r-value  t-value'/
     .   21X,' =======  =======  =======  =======  =======')
        WRITE(16,2016)N,STEP
 2016 FORMAT(22X,'1-',I5,2X,I5)
        WRITE(16,2018)MAXEV
 2018 FORMAT(/' Maximum number of cost evaluations:',I20/)
        CALL PACPAR
        CALL FUNEVAL
        DO 175 I=1,N
          IF(X(I).GT.XMAXS(I)) THEN
            XMAXS(I) = X(I)
          END IF
          IF(X(I).LT.XMINS(I)) THEN
            XMINS(I) = X(I)
          END IF
  175   CONTINUE
        FI=Z
        WRITE(16,2019)FI
 2019 FORMAT(/' Initial total cost                :',F20.2//
     .   '=========================================================================='/
     .   ' OPTIMIZATION STEPS:'/
     .   '==========================================================================')
        DO 180 I=1,N
          XX(I) = X(I)
  180   CONTINUE
        ZZ = Z
        PS=0
        BS=1

*       Explore about Base Point

        J   = 1
        FB  = FI
  200 IF(FE.GE.MAXEV) THEN
          GOTO 1000
        END IF
        X(J) = Y(J)+S(J)
        IF(X(J).LE.XMAX(J)) THEN
```

```
      CALL PACPAR
      CALL FUNEVAL
      DO 205 I=1,N
       IF(X(I).GT.XMAXS(I)) THEN
        XMAXS(I) = X(I)
       END IF
       IF(X(I).LT.XMINS(I)) THEN
        XMINS(I) = X(I)
       END IF
 205  CONTINUE
     ELSE
      Z = FI + 100
     END IF
 210 IF(Z.LT.FI) THEN
      GOTO 280
     END IF
     X(J) = Y(J)-S(J)
     IF(X(J).GE.XMIN(J)) THEN
      CALL PACPAR
      CALL FUNEVAL
      DO 215 I=1,N
       IF(X(I).GT.XMAXS(I)) THEN
        XMAXS(I) = X(I)
       END IF
       IF(X(I).LT.XMINS(I)) THEN
        XMINS(I) = X(I)
       END IF
 215  CONTINUE
     ELSE
      Z = FI + 100
     END IF
 220 IF(Z.LT.FI) THEN
      GOTO 280
     END IF
     X(J) = XX(J)
     Z = ZZ
     GOTO 290
 280 Y(J) = X(J)
 290 FI  = Z
     DO 292 I=1,N
      KPAR(I)=X(I)
 292 CONTINUE
     DO 296 I=FTYPES+1,N
      DO 294 JJ=1,NASS(I)
       KPAR(I)=KPAR(I)+KPAR(ASSNAME(I,JJ))*ASSBATCH(I,JJ)
 294  CONTINUE
 296 CONTINUE
     WRITE(16,2020)Z,FE
2020 FORMAT( ' Exploration Step    Cost=',F20.2,5X,'No. of eval.=',I6)
*   .   21X,' Product  z-value  k-value  r-value  t-value'/
*   .   21X,' =======  =======  =======  =======  =======')
```

```
*     DO 310 I=1,N
*       WRITE(16,2012)I,X(I),KPAR(I),RPAR,TAU
* 310 CONTINUE
      DO 320 I=1,N
        XX(I) = X(I)
  320 CONTINUE
      ZZ = Z
      IF(J.EQ.N) THEN
        DO 325 I=1,N
          XMIN(I)=XMINB(I)
          XMAX(I)=XMAXB(I)
 325    CONTINUE
        GOTO 360
      END IF
      J = J+1
      GOTO 200

*     After 360 make a pattern move if function has been reduced.

 360 IF(FI.LT.(FB-(1.0E-08))) THEN
       GOTO 540
     END IF

*     But if exploration was about a pattern point
*     and no reduction was made change base at 420,
*     otherwise reduce step length at 490.

     IF(PS.EQ.1.AND.BS.EQ.0) THEN
      GOTO 420
     END IF
     GOTO 490
 420 DO 430 I=1,N
       P(I) = B(I)
       Y(I) = B(I)
       X(I) = B(I)
 430 CONTINUE
     CALL PACPAR
     CALL FUNEVAL
     DO 435 I=1,N
       IF(X(I).GT.XMAXS(I)) THEN
         XMAXS(I) = X(I)
       END IF
       IF(X(I).LT.XMINS(I)) THEN
         XMINS(I) = X(I)
       END IF
 435 CONTINUE
     BS = 1
     PS = 0
     FI = Z
     FB = Z
```

```
      DO 436 I=1,N
       KPAR(I)=X(I)
  436 CONTINUE
      DO 438 I=FTYPES+1,N
       DO 437 JJ=1,NASS(I)
        KPAR(I)=KPAR(I)+KPAR(ASSNAME(I,JJ))*ASSBATCH(I,JJ)
  437  CONTINUE
  438 CONTINUE
      WRITE(16,2030)Z,FE
 2030 FORMAT( ' Base Change        Cost=',F20.2,5X,'No. of eval.=',I6/
     .  21X,' Product  z-value  k-value  r-value  t-value'/
     .  21X,' =======  =======  =======  =======  =======')
      DO 440 I=1,N
       WRITE(16,2012)I,X(I),KPAR(I),RPAR,TAU
  440 CONTINUE
      DO 450 I=1,N
       XX(I) = X(I)
  450 CONTINUE
      ZZ = Z
      J = 1
      GOTO 200
  490 K = 0
      DO 495 I=1,N
       IF(SI(I).EQ.1) THEN
        K=K+1
       END IF
  495 CONTINUE
      IF(K.EQ.N) THEN
       GOTO 700
      END IF
      DO 500 I=1,N
       S(I)=S(I)/SI(I)
  500 CONTINUE
      WRITE(16,2040)
 2040 FORMAT(' Contract Step Length')
      DO 510 I=1,N
       IF(S(I).LT.1) THEN
        GOTO 700
       END IF
  510 CONTINUE
      J = 1
      GOTO 200

*     Pattern Move

  540 DO 600 I=1,N
       P(I)=2*Y(I)-B(I)
       IF(P(I).GT.XMAX(I)) THEN
        P(I) = XMAX(I)
       END IF
       IF(P(I).LT.XMIN(I)) THEN
```

```
      P(I) = XMIN(I)
     END IF
     B(I)=Y(I)
     X(I)=P(I)
     Y(I)=X(I)
 600 CONTINUE

    CALL PACPAR
    CALL FUNEVAL
    DO 605 I=1,N
     IF(X(I).GT.XMAXS(I)) THEN
      XMAXS(I) = X(I)
     END IF
     IF(X(I).LT.XMINS(I)) THEN
      XMINS(I) = X(I)
     END IF
 605 CONTINUE
    FB = FI
    PS = 1
    BS = 0
    FI = Z
    DO 606 I=1,N
     KPAR(I)=X(I)
 606 CONTINUE
    DO 608 I=FTYPES+1,N
     DO 607 JJ=1,NASS(I)
      KPAR(I)=KPAR(I)+KPAR(ASSNAME(I,JJ))*ASSBATCH(I,JJ)
 607  CONTINUE
 608 CONTINUE
    WRITE(16,2050)Z,FE
2050 FORMAT( ' Pattern Move       Cost=',F20.2,5X,'No. of eval.=',I6/
   .  21X,' Product  z-value  k-value  r-value  t-value'/
   .  21X,' =======  =======  =======  =======  =======')
    DO 610 I=1,N
     WRITE(16,2012)I,X(I),KPAR(I),RPAR,TAU
 610 CONTINUE
    DO 620 I=1,N
     XX(I) = X(I)
 620 CONTINUE
    ZZ = Z

*   Then explore about latest pattern point.

    J = 1
    GOTO 200
 700 WRITE(16,2060)FB,FE
2060 FORMAT(/' Minimum found:      Cost=',F20.2,5X,'No. of eval.=',
   .  I6//,21X,' Product  z-value  k-value  r-value  t-value'/
   .  21X,' =======  =======  =======  =======  =======')
    DO 710 I=1,N
     X(I)=P(I)
```

```
710 CONTINUE
    DO 720 I=1,N
      IF(X(I).GT.XMAXS(I)) THEN
        XMAXS(I)=X(I)
      END IF
      IF(X(I).LT.XMINS(I)) THEN
        XMINS(I) = X(I)
      END IF
720 CONTINUE
    DO 730 I=1,N
      KPAR(I)=X(I)
730 CONTINUE
    DO 745 I=FTYPES+1,N
      DO 740 JJ=1,NASS(I)
        KPAR(I)=KPAR(I)+KPAR(ASSNAME(I,JJ))*ASSBATCH(I,JJ)
740   CONTINUE
745 CONTINUE
    DO 750 I=1,N
      WRITE(16,2012)I,X(I),KPAR(I),RPAR,TAU
750 CONTINUE
1000 WRITE(16,2070)
2070 FORMAT(/' Max. values of parameters used in simulation:',
   .  //,21X,' Product  z-value  k-value  r-value  t-value'/
   .  21X,' =======  =======  =======  =======  =======')
    DO 1010 I=1,N
      WRITE(16,2212)I,XMAXS(I)
2212  FORMAT(21X,I5,5X,I5)
1010 CONTINUE
    WRITE(16,2080)
2080 FORMAT(/' Min. values of parameters used in simulation:',
   .  //,21X,' Product  z-value  k-value  r-value  t-value'/
   .  21X,' =======  =======  =======  =======  =======')
    DO 1020 I=1,N
      WRITE(16,2012)I,XMINS(I)
1020 CONTINUE
    FE=-1
    CALL PACPAR
    CALL FUNEVAL
    CALL TIME_STATS(E_TIME)
    WRITE(16,*)'CPU time:', E_TIME-S_TIME

    CLOSE(15)
    CLOSE(16)

    STOP
    END
```

```
      SUBROUTINE PACPAR

      INCLUDE 'opac.dcl'
      INTEGER I,J
      INTEGER N,X(MP),NASS(MP),ASSNAME(MP,MB),ASSBATCH(MP,MB)
      COMMON/PAR/N,X,NASS,ASSNAME,ASSBATCH

      DO 10 I=1,N
       ZV(I)=X(I)
       KV(I)=X(I)
       RV(I)=1
       TV(I)=0
   10 CONTINUE
      DO 30 I=FTYPES+1,N
       DO 20 J=1,NASS(I)
        KV(I)=KV(I)+KV(ASSNAME(I,J))*ASSBATCH(I,J)
   20  CONTINUE
   30 CONTINUE

      RETURN
      END

      SUBROUTINE TIME_STATS(ELAPSED)
      (see OPT.FOR)


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C********************************************************************C
C*  OPAC.FOR      : Main Simulation Program for PAC Model -           *C
C*                  used for any optimization procedure               *C
C*  Author        : Krystyna  Bielunska                               *C
C*  Date          : May 1996                                          *C
C********************************************************************C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE FUNEVAL

*     Bring in declarations file.

      INCLUDE 'opac.dcl'
      INTEGER I,J,K,STORE,PROC,PROD,PPRD,FLAG
      INTEGER FE
      REAL Z
      COMMON/OPT/FE,Z
      INTEGER GENNUM
      COMMON /GEN/ GENNUM

      FLAG  = 0
      TIME  = 0.0

*     Write model input parameters.
```

```
      IF(FE.EQ.0) THEN
        WRITE(16,2030)NUNITS,(NMACHS(I),I=1,NUNITS)
 2030   FORMAT(' MODEL INPUT DATA'/
     .   ' ----------------'/
     .   ' Number of cells/stores  ',I15/
     .   ' Number of machines in each cell',3X,8I5)
        WRITE(16,2060)FTYPES,MTYPES,ATYPES,RTYPES
 2060   FORMAT(' Number of "final"          products',I9/
     .    ' Number of "final/assembly" products',I9/
     .    ' Number of "assembly"       products',I9/
     .    ' Number of "raw materials"           ',I9)
        WRITE(16,2070)(PROBD(I),I=1,FTYPES+MTYPES)
 2070   FORMAT(' Distr.funct. of "final" product types',8F5.2)
        WRITE(16,2080)MARRVT,LENGTH,WARMUP
 2080   FORMAT(' Mean interarr.time of all "final" prod. ',F7.2,' min.'
     .   /' Length of the simulation',F22.1,' 24-hours days'
     .   /' Length of "warming up"  ',F22.1,' 24-hours days'/)
        WRITE(16,2090)
 2090   FORMAT( ' Product  Cell/Store  Mean service time'/
     .    ' type                (in min.)'/
     .    ' =======  ==========  =================')
        DO 50 I=1,FTYPES+MTYPES+ATYPES
          WRITE(16,2100)I,SCELL(I),MSERVT(I)
 2100     FORMAT(I4,8X,I3,10X,F6.2)
   50   CONTINUE
        WRITE(16,2132)
 2132   FORMAT(/' Product type  No. of subassemblies  Subassembly ',
     .   'name  Units of subassembly'/
     .    ' ============  ====================  =================',
     .    ' ====================')
        DO 80 I=1,NTYPES
          DO 70 J=1,NSUBASS(I)
            IF(J.EQ.1) THEN
              WRITE(16,2134)I,NSUBASS(I),SUBNAME(I,J),SUBBATCH(I,J)
            ELSE
              WRITE(16,2136)SUBNAME(I,J),SUBBATCH(I,J)
            END IF
 2134       FORMAT(I8,13X,I7,13X,I7,13X,I7)
 2136       FORMAT(41X,I7,13X,I7)
   70     CONTINUE
   80   CONTINUE
        WRITE(16,2138)
 2138   FORMAT(/' Product   Cell   Time to transport a unit of ',
     .  'product   WIP Cost'/
     .    ' type           from storage to cell (in min.)',
     .    '  ($/day/item)'/
     .    ' =======  ====  ==============================',
     .    ' ==============')
        DO 90 I=1,NTYPES+RTYPES
          DO 85 J=1,NUNITS
```

```
        IF(TRAN(I,J).GT.0.0) THEN
         WRITE(16,2139) I,J,TRAN(I,J),COSTW(I,J)
 2139      FORMAT(I5,7X,I3,19X,F6.2,20X,F6.2)
        END IF
  85    CONTINUE
  90   CONTINUE
      WRITE(16,2140)
 2140  FORMAT(/' Product          Customer Service Cost'/
     .    ' type          Delay Cost ($/item/day)'/
     .    ' ========      ======================')
      DO 92 I=1,FTYPES+MTYPES
       WRITE(16,2142)I,COSTD(I)
 2142    FORMAT(I5,20X,F10.2)
  92   CONTINUE
      WRITE(16,2144)
 2144  FORMAT(/'Product     PROD Cost      Setup'/
     .    ' type      ($/day/item)    (in min.)'/
     .    '======================================')
      DO 94 I=1,NTYPES
       WRITE(16,2146)I,COSTP(I),SETUP(I)
 2146    FORMAT(I4,12X,F6.2,9X,F6.2)
  94   CONTINUE
      END IF

      IF(FE.EQ.-1) THEN
       WRITE(16,2150)
 2150  FORMAT('*******************************************************',
     .    '********************'/
     .    '*  SIMULATION OF MULTIPLE-CELL SYSTEM COORDINATED BY',
     .    ' PA CARDS (PAC)   *'/
     .    '*  Developed by: Krystyna Bielunska, Ind. Eng., TUN',
     .    'S, April 1996      *'/
     .    '*******************************************************',
     .    '********************'/)
      WRITE(16,2152)
 2152  FORMAT( ' Parameters setting for the coordination scheme:'//
     .    ' Product  z-value ',
     .    ' k-value  r-value  t-value'/
     .    ' =======  =======',
     .    ' =======  =======  =======')
      DO 96 I=1,FTYPES+MTYPES+ATYPES
       WRITE(16,2154)I,ZV(I),KV(I),RV(I),TV(I)
 2154    FORMAT(I4,6X,I5,4X,I5,4X,I5,3X,I6)
  96   CONTINUE
      WRITE(16,2160)
 2160  FORMAT(//' SIMULATION RESULTS OF THE FINAL OPTIMIZATION STEP'/
     .    ' ------------------------------------')
      END IF

*     Initialize all machines in all cells to the idle state and the last
*     product at machines to zero.
```

```
      DO 100 I=1,NUNITS
        NBUSY(I) =0
        LASTPR(I)=0
  100 CONTINUE

*     Initialize all batches of PA cards per product per store.

      DO 110 I=1,NTYPES
        BATCH(I)=0
  110 CONTINUE

*     Initialize all "subassembly" batches per product per WIP-queues.

      DO 120 I=1,NTYPES+RTYPES
       DO 115 J=1,NUNITS
        BATCHWIP(I,J)=0
  115  CONTINUE
  120 CONTINUE

*     Initialize SIMLIBG.

      CALL INITLK

*     Initialize number of process tags per product type per store.

      DO 150 I=1,NTYPES
       STORE=SCELL(I)
       IF(KV(I).NE.0) THEN
        PROC=4+(STORE-1)*6
        DO 130 K=1,KV(I)
         TRNSFR(1)=0.0
         TRNSFR(2)=I
         CALL FILE(LLAST,PROC)
  130   CONTINUE
       END IF
  150 CONTINUE

*     Initialize number of products per product type per store.
*     Initialize counter for probability an arriving demand
*     is met immediately.

      DO 190 I=1,NTYPES
       PSERV(I) =0
       STORE=SCELL(I)
       IF(ZV(I).NE.0) THEN
        PROD=3+(STORE-1)*6
        PPRD=NUNITS*6+FTYPES+MTYPES+I
        DO 170 K=1,ZV(I)
         TRNSFR(1)=0.0
         TRNSFR(2)=I
```

```
          CALL FILE(LLAST,PROD)
          CALL FILE(LLAST,PPRD)
  170   CONTINUE
        END IF
  190 CONTINUE

*   Schedule the arrival of the first job.

      GENNUM=0
      TRNSFR(1)=EXPON(MARRVT,1)
      GENNUM=1
      TRNSFR(2)=1.0
      TRNSFR(4)=FG
      CALL FILE(LINCR,LEVENT)

*   Schedule the end of the simulation.

      TRNSFR(1)=1440*LENGTH
      TRNSFR(2)=7.0
      CALL FILE(LINCR,LEVENT)

*   Determine the length of "warming up" period in minutes.

      IF(FE.EQ.0) THEN
       WARMUP = 1440*WARMUP
      END IF

*   Schedule the time of recording machine/cell states.

*     TRNSFR(1)=WARMUP
*     TRNSFR(2)=7.0
*     CALL FILE(LINCR,LEVENT)

*   Determine the next event.

  200 CALL TIMING

*   Reset the statistical accumulators to 0 after "warming up" period.

      IF(TIME.GE.WARMUP.AND.FLAG.EQ.0) THEN
       CALL SAMPST(0.0,0)
       DO 202 I=1,NUNITS
        CALL TIMEST(FLOAT(NBUSY(I)),I)
  202  CONTINUE
       FLAG = 1
       DO 205 I=1,NTYPES
        PSERV(I)=0
  205  CONTINUE
      END IF

*   Call the appropriate event routine.
```

```
      GOTO(210,220,230,240,250,260,270),NEXT
  210   CALL OORDARR(1)
        GOTO 200
  220   CALL OREQARR
        GOTO 200
  230   CALL OPACARR
        GOTO 200
  240   CALL OWIPARR
        GOTO 200
  250   CALL ODEPART
        GOTO 200
  260   CALL OPPARR
        GOTO 200
  270   CALL OREPORT

      Z  = TC
      FE = FE+1
      IF(Z.EQ.1.0E+30) THEN
       WRITE(16,*)'!!!! This simulation run was infeasible !!!!'
      END IF
*     WRITE(*,2300)FE,Z
*2300 FORMAT(' Sim. no. =',I6,'       Cost =',F20.2)
      IF(FE.GT.1) THEN
       WRITE(16,2310)FE,Z
 2310  FORMAT(' Sim. no.= ',I6,'    Cost=',F20.2)
      END IF

      RETURN
      END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C**********************************************************************C
C*  OSIMLIBG.FOR : Set of routines of SIMLIB - General Version        *C
C*  Author        : Krystyna  Bielunska                                *C
C*  Date          : February 1996                                      *C
C**********************************************************************C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      (as SIMLIBG.FOR)

      REAL FUNCTION RANFN(ISTRM)

      INTEGER B2E15,B2E16,HI15,HI31,ISTRM,IZSET,LOW15,LOWPRD,
     &        MODLUS,MULT1,MULT2,OVFLOW,ZI,ZRNG(100),ZRN1(100)
      INTEGER IRANDG,RANDST
      INTEGER GENNUMI,I
      COMMON /GEN/ GENNUM

*     Force saving of ZRNG between calls.
```

```
      SAVE ZRNG
      IF(GENNUM.EQ.0) THEN
       DO 10 I=1,100
        ZRNG(I)=ZRN1(I)
   10  CONTINUE
      END IF

*     Define the constants.

      DATA MULT1,MULT2/24112,26143/
      DATA B2E15,B2E16,MODLUS/32768,65536,2147483647/

*     Set the default seeds for all 100 streams.

      DATA ZRNG/1973272912, 281629770, 20006270,1280689831,2096730329,
     &       1933576050, 913566091, 246780520,1363774876, 604901985,
     &       1511192140,1259851944, 824064364, 150493284, 242708531,
     &        75253171,1964472944,1202299975, 233217322,1911216000,
     &        726370533, 403498145, 993232223,103205531, 762430696,
     &       1922803170,1385516923, 76271663, 413682397, 726466604,
     &        336157058,1432650381,1120463904, 595778810, 877722890,
     &       1046574445, 68911991,2088367019, 748545416, 622401386,
     &       2122378830, 640690903,1774806513,2132545692,2079249579,
     &        78130110, 852776735,1187867272,1351423507,1645973084,
     &       1997049139, 922510944,2045512870, 898585771, 243649545,
     &       1004818771, 773686062, 403188473, 372279877,1901633463,
     &        498067494,2087759558, 493157915, 597104727,1530940798,
     &       1814496276, 536444882,1663153658, 855503735, 67784357,
     &       1432404475, 619691088, 119025595, 880802310, 176192644,
     &       1116780070, 277854671,1366580350,1142483975,2026948561,
     &       1053920743, 786262391,1792203830,1494667770,1923011392,
     &       1433700034,1244184613,1147297105, 539712780,1545929719,
     &        190641742,1645390429, 264907697, 620389253,1502074852,
     &        927711160, 364849192,2049576050, 638580085, 547070247/

      DATA ZRN1/1973272912, 281629770, 20006270,1280689831,2096730329,
     &       1933576050, 913566091, 246780520,1363774876, 604901985,
     &       1511192140,1259851944, 824064364, 150493284, 242708531,
     &        75253171,1964472944,1202299975, 233217322,1911216000,
     &        726370533, 403498145, 993232223,1103205531, 762430696,
     &       1922803170,1385516923, 76271663, 413682397, 726466604,
     &        336157058,1432650381,1120463904, 595778810, 877722890,
     &       1046574445, 68911991,2088367019, 748545416, 622401386,
     &       2122378830, 640690903,1774806513,2132545692,2079249579,
     &        78130110, 852776735,1187867272,1351423507,1645973084,
     &       1997049139, 922510944,2045512870, 898585771, 243649545,
     &       1004818771, 773686062, 403188473, 372279877,1901633463,
     &        498067494,2087759558, 493157915, 597104727,1530940798,
     &       1814496276, 536444882,1663153658, 855503735, 67784357,
     &       1432404475, 619691088, 119025595, 880802310, 176192644,
     &       1116780070, 277854671,1366580350,1142483975,2026948561,
```

```
     &       1053920743, 786262391,1792203830,1494667770,1923011392,
     &       1433700034,1244184613,1147297105, 539712780,1545929719,
     &        190641742,1645390429, 264907697, 620389253,1502074852,
     &        927711160, 364849192,2049576050, 638580085, 547070247/

*    Generate the next random number.

      ZI     = ZRNG(ISTRM)
      HI15   = ZI / B2E16
      LOWPRD = (ZI - HI15 * B2E16) * MULT1
      LOW15  = LOWPRD / B2E16
      HI31   = HI15 * MULT1 + LOW15
      OVFLOW = HI31 / B2E15
      ZI     = (((LOWPRD - LOW15 * B2E16) - MODLUS) +
     &         (HI31 - OVFLOW * B2E15) * B2E16) + OVFLOW
      IF (ZI .LT. 0) ZI = ZI + MODLUS
      HI15   = ZI / B2E16
      LOWPRD = (ZI - HI15 * B2E16) * MULT2
      LOW15  = LOWPRD / B2E16
      HI31   = HI15 * MULT2 + LOW15
      OVFLOW = HI31 / B2E15
      ZI     = (((LOWPRD - LOW15 * B2E16) - MODLUS) +
     &         (HI31 - OVFLOW * B2E15) * B2E16) + OVFLOW
      IF (ZI .LT. 0) ZI = ZI + MODLUS
      ZRNG(ISTRM) = ZI
      RANFN  = (2 * (ZI / 256) + 1) / 16777216.0
      RETURN

*    Set the current ZRNG for stream ISTRM to IZSET.

      ENTRY RANDST(IZSET,ISTRM)
      ZRNG(ISTRM) = IZSET
      RETURN

*    Return the current ZRNG for stream ISTRM.

      ENTRY IRANDG(ISTRM)
      IRANDG = ZRNG(ISTRM)
      RETURN

      END
```

Appendix B2

PAC OPTIMIZATION: EXAMPLES OF THE OPTIMIZATION REPORTS

Example of the report generated by *opt*

```
*******************************************************************************
* OPTIMIZATION OF MULTIPLE-CELL SYSTEM COORDINATED BY PA CARDS (PAC) *
* Developed by: Krystyna Bielunska, Ind. Eng., TUNS, May 1996          *
*******************************************************************************
```

scenario: exponential service times
DCI

Parameters setting   Model: 4   Policy: General   Date: Dec.05.1996

Product: 1 - 7	z-value	k-value	r-value	t-value
Minimum values:	0	1	1	0
Maximum values:	50	100	1	2000
Step length:	2	2	1	20

Initial values per product

1	0	1	1	0
2	0	1	1	0
3	0	1	1	0
4	0	1	1	0
5	0	1	1	0
6	0	1	1	0
7	0	1	1	0

Maximum number of cost evaluations: 10000

MODEL INPUT DATA

Number of cells/stores	4
Number of machines in each cell	1 1 1 1
Number of "final" products	3
Number of "final/assembly" products	1
Number of "assembly" products	3
Number of "raw materials"	4
Distr.funct. of "final" product types	.50 .60 .90 1.00
Mean interarr.time of all "final" prod.	180.00 min.
Length of the simulation	300.0 24-hour days
Length of "warming up"	40.0 24-hour days

Product type	Cell/Store	Mean service time (in min.)
1	1	42.00
2	4	18.00
3	4	36.00
4	2	18.00
5	2	24.00
6	3	18.00
7	3	30.00

Product type	No. of subassemblies	Subassembly name	Units of subassembly
1	2	4	1
		5	1
2	1	5	3
3	2	5	1
		6	2
4	1	8	1
5	2	7	3
		9	2
6	1	11	1
7	1	10	1

Product type	Cell	Time to transport a unit of product from storage to cell (in min.)	WIP Cost ($/day/item)
4	1	12.00	3.00
5	1	12.00	12.00
5	4	18.00	12.00
6	4	6.00	3.00
7	2	18.00	3.00
8	2	24.00	2.00
9	2	24.00	1.00
10	3	18.00	2.00
11	3	18.00	2.00

Product type	Customer Service Cost Delay Cost ($/item/day)
1	10.00
2	8.00
3	6.00
4	5.00

Product type	PROD Cost ($/day/item)	Setup (in min.)
1	16.00	.00
2	40.00	.00
3	20.00	.00
4	3.00	.00
5	12.00	.00
6	3.00	.00
7	3.00	.00

Initial total cost: 75166.24

OPTIMIZATION STEPS:

Sim. no.= 2	Cost=	77270.10		
Exploration Step	Cost=	75166.24	No. of eval.=	2
Sim. no.= 3	Cost=	79187.58		
Exploration Step	Cost=	75166.24	No. of eval.=	3
Exploration Step	Cost=	75166.24	No. of eval.=	3
Sim. no.= 4	Cost=	75166.24		

......

Exploration Step	Cost=	16192.31	No. of eval.=	22
Sim. no.= 23	Cost=	72957.19		
Pattern Move	Cost=	72957.19	No. of eval.=	23

Product	z-value	k-value	r-value	t-value
1	0	1	1	0
2	4	1	1	0
3	0	1	1	0
4	0	1	1	0
5	0	5	1	40
6	0	1	1	0
7	0	1	1	0

......

Contract Step Length

Sim. no.= 322	Cost=	8173.87		
Exploration Step	Cost=	6512.92	No. of eval.=	322
Sim. no.= 323	Cost=	6512.92		

......

Sim. no.= 383	Cost=	6458.49		
Base Change	Cost=	6458.49	No. of eval.=	383

Product	z-value	k-value	r-value	t-value
1	0	7	1	0
2	0	9	1	0
3	0	15	1	0
4	0	9	1	20
5	0	1	1	20
6	0	11	1	0
7	3	3	1	0

Sim. no.= 384	Cost=	8010.12		
Exploration Step	Cost=	6458.49	No. of eval.=	384
......				
Exploration Step	Cost=	6248.88	No. of eval.=	447
Exploration Step	Cost=	6248.88	No. of eval.=	447
Sim. no.= 448	Cost=	6248.88		
Exploration Step	Cost=	6248.88	No. of eval.=	448
Contract Step Length				

Minimum found: Cost= 6248.88 No. of eval.= 448

Product	z-value	k-value	r-value	t-value
1	0	2	1	0
2	0	2	1	0
3	0	11	1	0
4	0	7	1	20
5	0	2	1	20
6	0	9	2	20
7	3	3	1	0

Max. values of parameters used in simulation:

Product	z-value	k-value	r-value	t-value
1	2	9	2	20
2	2	8	2	20
3	2	13	2	20
4	2	9	1	60
5	2	4	1	40
6	2	11	3	80
7	5	5	1	20

Min. values of parameters used in simulation:

Product	z-value	k-value	r-value	t-value
1	0	1	1	0
2	0	1	1	0
3	0	1	1	0
4	0	1	1	0
5	0	1	1	0
6	0	1	1	0
7	0	1	1	0

....

Here comes complete output of the simulation run with the final optimal parameters setting

Example of the report generated by *opk*

****

* OPTIMIZATION OF MULTIPLE-CELL SYSTEM COORDINATED BY PA CARDS (PAC) *

* Developed by: Krystyna Bielunska, Ind. Eng., TUNS, May 1996 *

****

scenario: exponential service times
DCI

Parameters setting Model: 4 Policy: Kanban Date: Dec.05.1996

Product: 1 - 7		z-value	k-value	r-value	t-value
Minimum values:		1	1	1	0
Maximum values:		100	100	20	0
Step length:		2	2	1	0
Initial values per product					
	1	1	1	1	0
	2	1	1	1	0
	3	1	1	1	0
	4	1	1	1	0
	5	1	1	1	0
	6	1	1	1	0
	7	1	1	1	0

Maximum number of cost evaluations: 10000

MODEL INPUT DATA

(the same as for the *opt* output)

Initial total cost: 47329.10

OPTIMIZATION STEPS:

Sim. no.= 2 Cost= 28381.92
Exploration Step Cost= 28381.92 No. of eval.= 2
Sim. no.= 3 Cost= 176917.06
......
Pattern Move Cost= 27168.59 No. of eval.= 61

Product	z-value	k-value	r-value	t-value
1	4	4	1	0
2	1	1	1	0
3	1	1	1	0
4	1	1	1	0
5	2	2	1	0
6	1	1	1	0
7	1	1	1	0

Sim. no.=	62	Cost=	30736.13		
Sim. no.=	63	Cost=	23907.48		

......

Exploration Step	Cost=	23907.48	No. of eval.=	83
Exploration Step	Cost=	23907.48	No. of eval.=	83
Contract Step Length				

Minimum found:	Cost=	23907.48	No. of eval.=	83

Product	z-value	k-value	r-value	t-value
1	3	3	1	0
2	1	1	1	0
3	1	1	1	0
4	1	1	1	0
5	2	2	1	0
6	1	1	1	0
7	1	1	1	0

Max. values of parameters used in simulation:

Product	z-value	k-value	r-value	t-value
1	7		2	
2	3		1	
3	3		1	
4	3		2	
5	7		1	
6	3		1	
7	3		1	

Min. values of parameters used in simulation:

Product	z-value	k-value	r-value	t-value
1	1		1	
2	1		1	
3	1		1	
4	1		1	
5	1		1	
6	1		1	
7	1		1	

....

Here comes complete output of the simulation run with the final optimal parameters setting

Example of the report generated by *opi*

```
*******************************************************************************
* OPTIMIZATION OF MULTIPLE-CELL SYSTEM COORDINATED BY PA CARDS (PAC) *
* Developed by: Krystyna Bielunska, Ind. Eng., TUNS, May 1996               *
*******************************************************************************
```

scenario: exponential service times
DCI

Parameters setting Model: 4 Policy: IC Date: Dec.05.1996

Initial values:

Product	z-value	k-value	r-value	t-value
1	1	1	1	0
2	1	1	1	0
3	1	1	1	0
4	1	2	1	0
5	0	3	1	0
6	0	1	1	0
7	0	3	1	0

Minimum values:

Product	z-value	k-value	r-value	t-value
1	1	1	1	0
2	1	1	1	0
3	1	1	1	0
4	1	2	1	0
5	0	3	1	0
6	0	1	1	0
7	0	3	1	0

Maximum values:

Product	z-value	k-value	r-value	t-value
1- 7	50			

Step length:

Product	z-value	k-value	r-value	t-value
1- 7	2			

Maximum number of cost evaluations: 10000

MODEL INPUT DATA

(the same as for the *opt* output)

Initial total cost: 19307.02

OPTIMIZATION STEPS:

Sim. no.= 2 Cost= 23956.57

Exploration Step	Cost=	19307.02	No. of eval.=	2
Sim. no.= 3	Cost=	39695.70		
Exploration Step	Cost=	19307.02	No. of eval.=	3
Sim. no.= 4	Cost=	29550.19		

......

Exploration Step	Cost=	18616.32	No. of eval.=	51
Contract Step Length				
Minimum found:	Cost=	18616.32	No. of eval.=	51

Product	z-value	k-value	r-value	t-value
1	1	1	1	0
2	1	1	1	0
3	1	1	1	0
4	1	2	1	0
5	0	3	1	0
6	0	1	1	0
7	1	4	1	0

Max. values of parameters used in simulation:

Product	z-value	k-value	r-value	t-value
1	3			
2	3			
3	3			
4	3			
5	2			
6	2			
7	6			

Min. values of parameters used in simulation:

Product	z-value	k-value	r-value	t-value
1	1			
2	1			
3	1			
4	1			
5	0			
6	0			
7	0			

....

Here comes complete output of the simulation run with the final optimal parameters setting

Appendix B3

RANDOM OPTIMIZATION ALGORITHM

Main Programs: **ROP.FOR**

Subprograms and Declaration Files:

as in case of OPT.FOR optimization program - refer to Appendix B1

FORTRAN Variables (additional):

Variable	Definition
Input parameters:	
BOXN	Maximum number of boxes
EVRED	Reduction of number of function evaluations (=PAC simulations) per each new box
MAXEVB	Maximum number of function evaluations for the first box
RANGE(I)	Range value for PAC parameter I
RMI(I)	Range minimum for PAC parameter I
SIB(I)	Range reduction for PAC parameter I
Modeling variables:	
NB	Help variable for number of boxes
NEB	Help variable for number of function evaluations per box
XO(I)	Help variable for optimum value of PAC parameter I

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C************************************************************************C
C*    ROP.FOR      : Main Random Optimization Program for PAC Model      *C
C*                   (General, BSS, MRP, PTO, LC)                        *C
C*    Author       : Krystyna Bielunska                                  *C
C*    Date         : August 1996                                         *C
C************************************************************************C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      INCLUDE 'opac.dcl'
      INTEGER I,K,L,N,MAXEVB,FE,SIB(MX)
      INTEGER X(MX),XO(MX),XMINS(MX),XMAXS(MX),EVRED
      INTEGER RAWS,MODEL,NIR,PR,II,RR,BOXN,NB,NEB
      INTEGER NASS(MP),ASSNAME(MP,MB),ASSBATCH(MP,MB)
      INTEGER XZ(MP),XK(MP),XR(MP),XT(MP),RMI(MX)
      REAL XMIN(MX),XMAX(MX),RANGE(MX),RMIN(MX)
      REAL Z,FI,XX(MX)
      REAL*4 S_TIME,E_TIME
      CHARACTER*11 DATE
      CHARACTER*8 POLICY
      COMMON/OPT/FE,Z
      COMMON/PAR/N,X

      CALL TIME_STATS(S_TIME)

*     Open input and output files.

      OPEN(15,FILE='rop.in')
      OPEN(16,FILE='rop.out')
      FE = 0
      NB = 1
      NEB= 0
      N  = MX

*     Read input parameters for optimization.

      READ(15,FMT='(1A11)') DATE
      READ(15,FMT='(1A8)') POLICY
      READ(15,*) MODEL,MAXEVB,EVRED,BOXN
      READ(15,*) RANGE(1),SIB(1),RMI(1)
      RMIN(1)=FLOAT(RMI(1))
      READ(15,*) RANGE(2),SIB(2),RMI(2)
      RMIN(2)=FLOAT(RMI(2))
      READ(15,*) RANGE(3),SIB(3),RMI(3)
      RMIN(3)=FLOAT(RMI(3))
      READ(15,*) RANGE(4),SIB(4),RMI(4)
      RMIN(4)=FLOAT(RMI(4))

*     Read input parameters for simulation.

      READ(15,*) NUNITS,FTYPES,MTYPES,ATYPES,RTYPES,MARRVT,
```

```
     .      LENGTH,WARMUP
      NTYPES=FTYPES+MTYPES+ATYPES
      FG   =NUNITS+1
      DAYS =LENGTH-WARMUP
      READ(15,*)(NMACHS(I),I=1,NUNITS)
      DO 10 I=1,NTYPES
       READ(15,*) XZ(I),XK(I),XR(I),XT(I),MSERVT(I),SCELL(I),
     .      NSUBASS(I),(SUBNAME(I,J) ,J=1,NSUBASS(I)),
     .          (SUBBATCH(I,J),J=1,NSUBASS(I))
   10 CONTINUE
      READ(15,*)(PROBD(I),I=1,FTYPES+MTYPES)
      READ(15,*)RAWS
      DO 15 I=1,RAWS
       READ(15,*)PRODTYPE,STORE,TRAN(PRODTYPE,STORE),
     .      COSTW(PRODTYPE,STORE)
   15 CONTINUE
      DO 20 I=1,FTYPES+MTYPES
       READ(15,*)COSTD(I)
   20 CONTINUE
      DO 25 I=1,NTYPES
       READ(15,*)COSTP(I),SETUP(I)
   25 CONTINUE

      DO 30 I=1,NTYPES
       NASS(I)=0
   30 CONTINUE
      DO 40 I=1,NTYPES
       DO 35 J=1,NSUBASS(I)
        IF(SUBNAME(I,J).LE.NTYPES) THEN
         NASS(SUBNAME(I,J))=NASS(SUBNAME(I,J))+1
         ASSNAME(SUBNAME(I,J),NASS(SUBNAME(I,J)))=I
         ASSBATCH(SUBNAME(I,J),NASS(SUBNAME(I,J)))=SUBBATCH(I,J)
        END IF
   35   CONTINUE
   40 CONTINUE

      DO 50 I=5,N,4
       RANGE(I)=RANGE(1)
       RMI(I)  =RMI(1)
       SIB(I)  =SIB(1)
   50 CONTINUE
      DO 60 I=6,N,4
       RANGE(I)=RANGE(2)
       RMI(I)  =RMI(2)
       SIB(I)  =SIB(2)
   60 CONTINUE
      DO 70 I=7,N,4
       RANGE(I)=RANGE(3)
       RMI(I)  =RMI(3)
       SIB(I) = SIB(3)
   70 CONTINUE
```

```
      DO 80 I=8,N,4
        RANGE(I)=RANGE(4)
        RMI(I) =RMI(4)
        SIB(I) = SIB(4)
   80 CONTINUE
      K=1
      DO 90 I=1,N,4
        X(I)   = XZ(K)
        X(I+1) = XK(K)
        X(I+2) = XR(K)
        X(I+3) = XT(K)
        K    = K+1
   90 CONTINUE
      DO 95 I=1,N
        XMAXS(I)=X(I)
        XMINS(I)=X(I)
   95 CONTINUE
      DO 96 I=1,N
        XO(I) =X(I)
   96 CONTINUE
      WRITE(16,2010)
 2010 FORMAT('**********************************************************',
     .      '********************'/
     .      '*  OPTIMIZATION OF MULTIPLE-CELL SYSTEM COORDINATED BY',
     .      ' PA CARDS (PAC)   *'/
     .      '*  Developed by: Krystyna Bielunska, Ind. Eng., TUN',
     .      'S, July 1996 (random)*'/
     .      '**********************************************************',
     .      '********************'/
     .      ' scenario: exponential service times'/)
      K=N/4
      WRITE(16,2011)MODEL,POLICY,DATE,K
 2011 FORMAT( ' Parameters setting',' Model: ',I2,5X,'Policy: ',A8,4X,
     .      ' Date: ',A11//
     .      ' Product: 1 - ',I5,12X,
     .      ' z-value  k-value  r-value  t-value'/
     .      ' ==================',12X,
     .      ' =======  =======  =======  =======')
      WRITE(16,2012) RANGE(1),RANGE(2),RANGE(3),RANGE(4)
 2012 FORMAT(/'  Range          ',15X,F5.1,4X,F5.1,4X,F5.1,4X,F5.1)
      WRITE(16,2013) SIB(1),SIB(2),SIB(3),SIB(4)
 2013 FORMAT(/'  Range reduction',15X,I5,4X,I5,4X,I5,4X,I5)
      WRITE(16,2014) RMI(1),RMI(2),RMI(3),RMI(4)
 2014 FORMAT(/'  Minimum range  ',15X,I5,4X,I5,4X,I5,4X,I5)
      WRITE(16,2015)
 2015 FORMAT(/'  Initial values per product')
      K=1
      DO 97 I=1,N,4
        WRITE(16,2016) K,X(I),X(I+1),X(I+2),X(I+3)
 2016   FORMAT(22X,I3,7X,I5,4X,I5,4X,I5,4X,I5)
        K=K+1
```

```
   97 CONTINUE
      WRITE(16,2017)MAXEVB,EVRED,BOXN
 2017 FORMAT(/' Maximum number of cost evaluations per box:',I6/
     .        ' Reduction per number of cost evaluations  :',I6/
     .        ' Maximum number of boxes (iterations)       :',I6/)
      DO 98 I=1,N
          XMAX(I)  = XO(I) + RANGE(I)*0.5
          XMIN(I)  = XO(I) - RANGE(I)*0.5
   98 CONTINUE

      CALL PACPAR
      CALL FUNEVAL
      FI=Z
      WRITE(16,2019)FI
 2019 FORMAT(/' Initial total cost          :',F20.2//
     .'==================================================================='/
     .' OPTIMIZATION STEPS:'/
     .'===================================================================')
      WRITE(16,2040)NB
      K=0
      DO 99 I=1,N,4
       K=K+1
       WRITE(16,2042)K,XMIN(I),XMIN(I+1),XMIN(I+2),XMIN(I+3)
       WRITE(16,2043)K,XMAX(I),XMAX(I+1),XMAX(I+2),XMAX(I+3)
   99 CONTINUE

  100 DO 120 I=2,N,4
  110   XX(I) = UNF(XMIN(I),XMAX(I),1)
        X(I)  = INT(XX(I))
        IF(X(I).LT.1) GOTO 110
  120 CONTINUE
      DO 140 I=3,N,4
  130   XX(I) = UNF(XMIN(I),XMAX(I),1)
        X(I)  = INT(XX(I))
        IF(X(I).LT.1.OR.X(I).GT.X(I-1)) GOTO 130
  140 CONTINUE

*     Extra check for r-k feasibility conditions.

*     End of specific r-k conditions.

      DO 230 I=1,N,4
  220   XX(I) = UNF(XMIN(I),XMAX(I),1)
        X(I)  = INT(XX(I))
        IF(X(I).LT.0) GOTO 220
  230 CONTINUE
      DO 250 I=4,N,4
  240   XX(I) = UNF(XMIN(I),XMAX(I),1)
        X(I)  = INT(XX(I))
        IF(X(I).LT.0) GOTO 240
  250 CONTINUE
```

```
      CALL PACPAR
      CALL FUNEVAL
      IF(Z.EQ.1.0E+30) THEN
       GOTO 100
      END IF
      DO 260 I=1,N
       IF(X(I).GT.XMAXS(I)) THEN
        XMAXS(I) = X(I)
       END IF
       IF(X(I).LT.XMINS(I)) THEN
        XMINS(I) = X(I)
       END IF
  260 CONTINUE
      NEB=NEB+1
      IF(Z.LT.FI) THEN
       DO 300 I=1,N
        XO(I) = X(I)
        FI   = Z
  300  CONTINUE
      END IF
*     WRITE(16,2020)Z,NEB
*2020 FORMAT( ' Random Step      Cost=',F20.2,5X,'No. of box eval.=',I5/
*    .  21X,' Product  z-value  k-value  r-value  t-value'/
*    .  21X,' ======== ======== ======== ======== ========')
*     K=0
*     DO 310 I=1,N,4
*      K=K+1
*      WRITE(16,2112)K,X(I),X(I+1),X(I+2),X(I+3)
 2112  FORMAT(21X,I5,5X,I5,4X,I5,4X,I5,4X,I5)
* 310 CONTINUE
      IF(NEB.LT.MAXEVB) THEN
       GOTO 100
      ELSE
       NEB = 0
       NB  = NB+1
       MAXEVB = MAXEVB-EVRED
      END IF
      IF(NB.GT.BOXN.OR.MAXEVB.LE.0) THEN
       GOTO 700
      END IF
      DO 500 I=1,N
          RANGE(I) = RANGE(I)-SIB(I)
          IF(RANGE(I).LT.RMIN(I)) THEN
           RANGE(I)=RMIN(I)
          END IF
      XMAX(I)  = XO(I) + RANGE(I)*0.5
      XMIN(I)  = XO(I) - RANGE(I)*0.5
  500 CONTINUE

*     WRITE(16,2040) NB
 2040 FORMAT(' Box number:    ',I5/
```

```
     .  30X,' z-value  k-value  r-value  t-value'/
     .  30X,' ======= ======= ======= =======')
*    K=0
*    DO 600 I=1,N,4
*     K=K+1
*     WRITE(16,2042)K,XMIN(I),XMIN(I+1),XMIN(I+2),XMIN(I+3)
 2042  FORMAT(' Box minimum-:',I3,13X,F8.1,1X,F8.1,1X,F8.1,1X,F8.1)
*     WRITE(16,2043)K,XMAX(I),XMAX(I+1),XMAX(I+2),XMAX(I+3)
 2043  FORMAT(' Box maximum-:',I3,13X,F8.1,1X,F8.1,1X,F8.1,1X,F8.1)
* 600 CONTINUE
     GOTO 100
 700 WRITE(16,2060)FI,FE
2060 FORMAT(/' Minimum found:      Cost=',F20.2,5X,'No. of eval.=',
    .  I6//,21X,' Product  z-value  k-value  r-value  t-value'/
    .  21X,' ======= ======= ======= ======= =======')
     DO 710 I=1,N
      X(I)=XO(I)
 710 CONTINUE
     K=0
     DO 750 I=1,N,4
      K=K+1
      WRITE(16,2112)K,X(I),X(I+1),X(I+2),X(I+3)
 750 CONTINUE
1000 WRITE(16,2070)
2070 FORMAT(/' Max. values of parameters used in simulation:',
    .  //,21X,' Product  z-value  k-value  r-value  t-value'/
    .  21X,' ======= ======= ======= ======= =======')
     K=0
     DO 1010 I=1,N,4
      K=K+1
      WRITE(16,2112)K,XMAXS(I),XMAXS(I+1),XMAXS(I+2),XMAXS(I+3)
1010 CONTINUE
     WRITE(16,2080)
2080 FORMAT(/' Min. values of parameters used in simulation:',
    .  //,21X,' Product  z-value  k-value  r-value  t-value'/
    .  21X,' ======= ======= ======= ======= =======')
     K=0
     DO 1020 I=1,N,4
      K=K+1
      WRITE(16,2112)K,XMINS(I),XMINS(I+1),XMINS(I+2),XMINS(I+3)
1020 CONTINUE
     FE=-1
     CALL PACPAR
     CALL FUNEVAL
     CALL TIME_STATS(E_TIME)
     WRITE(16,*)'CPU time:', E_TIME-S_TIME

     CLOSE(15)
     CLOSE(16)
     STOP
     END
```

```
      SUBROUTINE PACPAR

      INCLUDE 'opac.dcl'
      INTEGER I,K
      INTEGER N,X(MX)
      COMMON/PAR/N,X

      K = 0
      DO 10 I=1,N,4
       K   =K+1
       ZV(K)=X(I)
       KV(K)=X(I+1)
       RV(K)=X(I+2)
       TV(K)=X(I+3)
   10 CONTINUE

      RETURN
      END


      REAL FUNCTION UNF(A,B,ISTRM)

      INTEGER ISTRM
      REAL A,B,U
      REAL GRAND

*     Generate a U(0,1) random variate from stream ISTRM.

      U = GRAND(ISTRM)

*     Generate a U(A,B) random variate.

      UNF = A + U * (B - A)

      RETURN
      END

      REAL FUNCTION GRAND(ISTRM)

      (as FUNCTION RAND(ISTRM)  in SIMLIBG.FOR  refer to Appendix A1)
```

```
**************************************************************************
* OPTIMIZATION OF MULTIPLE-CELL SYSTEM COORDINATED BY PA CARDS (PAC) *
* Developed by: Krystyna Bielunska, Ind. Eng., TUNS, July 1996 (random)   *
**************************************************************************
```

scenario: exponential service times
double customer delay cost

Parameters setting Model: 4 Policy: MRP Date: Nov.26.1996

Product: 1 - 2		z-value	k-value	r-value	t-value
Range		10.0	.0	6.0	400.0
Range reduction		2	0	1	50
Minimum range		2	0	2	10
Initial values per product					
	1	0	220	1	0
	2	0	220	1	0
	3	0	220	1	0
	4	0	220	1	0
	5	0	220	1	0
	6	0	220	1	0
	7	0	220	1	0

Maximum number of cost evaluations per box: 3
Reduction per number of cost evaluations : 1
Maximum number of boxes: 2

MODEL INPUT DATA

(the same as for the *opt* output - refer to Appendix B2)

Initial total cost: 7407.44

OPTIMIZATION STEPS:

Box number: 1

	z-value	k-value	r-value	t-value
Box minimum-: 1	-5.0	220.0	-2.0	-200.0
Box maximum-: 1	5.0	220.0	4.0	200.0
Box minimum-: 2	-5.0	220.0	-2.0	-200.0
Box maximum-: 2	5.0	220.0	4.0	200.0
Box minimum-: 3	-5.0	220.0	-2.0	-200.0
Box maximum-: 3	5.0	220.0	4.0	200.0
Box minimum-: 4	-5.0	220.0	-2.0	-200.0
Box maximum-: 4	5.0	220.0	4.0	200.0
Box minimum-: 5	-5.0	220.0	-2.0	-200.0
Box maximum-: 5	5.0	220.0	4.0	200.0
Box minimum-: 6	-5.0	220.0	-2.0	-200.0
Box maximum-: 6	5.0	220.0	4.0	200.0
Box minimum-: 7	-5.0	220.0	-2.0	-200.0
Box maximum-: 7	5.0	220.0	4.0	200.0

Random Step Cost= 34096.42 No. of box eval.= 1

Product	z-value	k-value	r-value	t-value
1	1	220	3	62
2	2	220	1	197
3	0	220	2	139
4	0	220	3	158
5	2	220	3	146
6	0	220	2	0
7	0	220	2	140

Random Step Cost= 64342.80 No. of box eval.= 2

Product	z-value	k-value	r-value	t-value
1	0	220	3	65
2	4	220	1	137
3	2	220	3	122
4	0	220	2	30
5	4	220	2	15
6	0	220	1	150
7	1	220	1	79

Random Step Cost= 30236.66 No. of box eval.= 3

Product	z-value	k-value	r-value	t-value
1	4	220	3	165
2	0	220	3	50
3	2	220	1	3
4	3	220	3	120
5	0	220	1	52
6	1	220	2	42
7	2	220	1	22

Box number: 2

	z-value	k-value	r-value	t-value
Box minimum-: 1	-4.0	220.0	-1.5	-175.0
Box maximum-: 1	4.0	220.0	3.5	175.0
Box minimum-: 2	-4.0	220.0	-1.5	-175.0
Box maximum-: 2	4.0	220.0	3.5	175.0
Box minimum-: 3	-4.0	220.0	-1.5	-175.0
Box maximum-: 3	4.0	220.0	3.5	175.0
Box minimum-: 4	-4.0	220.0	-1.5	-175.0
Box maximum-: 4	4.0	220.0	3.5	175.0
Box minimum-: 5	-4.0	220.0	-1.5	-175.0
Box maximum-: 5	4.0	220.0	3.5	175.0
Box minimum-: 6	-4.0	220.0	-1.5	-175.0
Box maximum-: 6	4.0	220.0	3.5	175.0
Box minimum-: 7	-4.0	220.0	-1.5	-175.0
Box maximum-: 7	4.0	220.0	3.5	175.0

Random Step Cost= 28243.98 No. of box eval.= 1

Product	z-value	k-value	r-value	t-value
1	2	220	1	66
2	0	220	3	14
3	2	220	1	62
4	0	220	1	119
5	1	220	1	67
6	2	220	1	112
7	3	220	1	32

Random Step Cost= 35535.07 No. of box eval.= 2

Product	z-value	k-value	r-value	t-value
1	0	220	1	96
2	2	220	1	54
3	2	220	2	148
4	1	220	3	92
5	0	220	1	173
6	3	220	1	162
7	0	220	2	41

Minimum found: Cost= 9162.75 No. of eval.= 6

Product	z-value	k-value	r-value	t-value
1	0	220	1	0
2	0	220	1	0
3	0	220	1	0
4	0	220	1	0
5	0	220	1	0
6	0	220	1	0
7	0	220	1	0

Max. values of parameters used in simulation:

Product	z-value	k-value	r-value	t-value
1	4	220	3	165
2	4	220	3	197
3	2	220	3	148
4	3	220	3	158
5	4	220	3	173
6	3	220	2	162
7	3	220	2	140

Min. values of parameters used in simulation:

Product	z-value	k-value	r-value	t-value
1	0	220	1	0
2	0	220	1	0
3	0	220	1	0
4	0	220	1	0
5	0	220	1	0
6	0	220	1	0
7	0	220	1	0

....

Here comes complete output of the simulation run with the final optimal parameters setting

Appendix C1

OPTIMIZATION RESULTS FOR MODEL 1

Table c1-1 **Model 1** case: ρ_2 =0.1 ($1/\mu_2$ = 6 min.)
DCI [single cost; from arrival of ORD-tag]

Best Solu -tion	exponential service time		uniform service time		exponential + bulk service time demand		uniform + bulk service time demand	
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7	0.1	0.7
1	**PAC** 595.5 - 34 z k r s 0 3 1 0 0 1 1 0	**PAC** 4239.66 - 74 z k r s 3 2 1 0 0 1 1 0	**PAC** 542.54 - 34 z k r s 0 3 1 0 0 1 1 0	**PAC** 2252.14 - 55 z k r s 2 2 1 0 0 1 1 0	**PAC** 1072.27 - 51 z k r s 0 11 1 0 0 1 1 0	**PAC** * 8239.69 - 51 z k r s 7 1 1 0 1 1 1 0	**PAC** 860.12 - 43 z k r s 0 5 1 0 0 1 1 0	**PAC** 7501.97 - 64 z k r s 5 2 1 0 0 1 1 0
2	**MRP** 600.63 - 13 z k r s 0 M 1 0 0 M 1 0	**LC** 4407.07 - 21 z k r s 3 1 1 0 2 1 1 0	**MRP** 545.46 - 13 z k r s 0 M 1 0 0 M 1 0	**IC** 2253.65 - 21 z k r s 2 2 1 0 0 2 1 0	**MRP** 1148.33 - 13 z k r s 0 M 1 0 0 M 1 0	**LC** 8566.64 - 18 z k r s 6 1 1 0 2 1 1 0	**MRP** 921.76 - 13 z k r s 0 M 1 0 0 M 1 0	**LC** 7938.51 - 25 z k r s 5 1 1 0 2 1 1 0
3	**PTO, τ≥0** 600.63 - 5 z k r s 0 M 1 0 0 M 1 0	**IC** 4705.03 - 14 z k r s 3 3 1 0 0 3 1 0	**PTO, τ≥0** 545.46 - 5 z k r s 0 M 1 0 0 M 1 0	**CONWIP** 2253.65 - 15 z k r s 2 2 1 0 0 2 1 0	**PTO, τ≥0** 1148.33 - 5 z k r s 0 M 1 0 0 M 1 0	**IC** 9748.77 - 18 z k r s 5 5 1 0 0 5 1 0	**PTO, τ≥0** 921.76 - 5 z k r s 0 M 1 0 0 M 1 0	**IC** 8651.54 - 21 z k r s 4 4 1 0 0 4 1 0
4	**PTO** 600.63 - 1 z k r s 0 M 1 0 0 M 1 0	**CONWIP** 4705.03 - 10 z k r s 3 3 1 0 0 3 1 0	**PTO** 545.46 - 1 z k r s 0 M 1 0 0 M 1 0	**MRP** 2700.38 - 30 z k r s 2 M 1 0 0 M 1 0	**PTO** 1148.33 - 1 z k r s 0 M 1 0 0 M 1 0	**CONWIP** 9748.77 - 13 z k r s 5 5 1 0 0 5 1 0	**PTO** 921.76 - 1 z k r s 0 M 1 0 0 M 1 0	**CONWIP** 8651.54 - 15 z k r s 4 4 1 0 0 4 1 0
5	**IC** 933.36 - 5 z k r s 1 1 1 0 0 1 1 0	**Kanban** 4805.23 - 21 z k r s 2 2 1 0 1 1 1 0	**IC** 897.21 - 5 z k r s 1 1 1 0 0 1 1 0	**Kanban** 2717.04 - 5 z k r s 1 1 1 0 1 1 1 0	**IC** 1617.75 - 5 z k r s 1 1 1 0 0 1 1 0	**Kanban** 10159.07 - 23 z k r s 5 5 1 0 1 1 1 0	**BSS** 1449.91 - 22 z k r s 1 M 2 0 1 M 1 0	**Kanban** 9074.22 - 27 z k r s 4 4 1 0 1 1 1 0
6	**CONWIP** 933.36 - 3 z k r s 1 1 1 0 0 1 1 0	**BSS** 5017.97 - 25 z k r s 3 M 1 0 1 M 2 0	**CONWIP** 897.21 - 3 z k r s 1 1 1 0 0 1 1 0	**LC** 2821.14 - 5 z k r s 2 1 1 0 2 1 1 0	**CONWIP** 1617.75 - 3 z k r s 1 1 1 0 0 1 1 0	**MRP** 11970.28 - 57 z k r s 8 M 4 0 0 M 1 0	**IC** 1571.18 - 5 z k r s 1 1 1 0 0 1 1 0	**MRP** 10630.88 - 52 z k r s 5 M 1 0 0 M 1 0
7	**BSS** 1125.41 - 22 z k r s 1 M 2 0 1 M 1 0	**MRP** 5243.12 - 53 z k r s 3 M 1 0 0 M 1 0	**BSS** 1077.36 - 22 z k r s 1 M 2 0 1 M 1 0	**BSS** 2915.86 - 38 z k r s 2 M 1 0 1 M 2 0	**BSS** 1620.75 - 22 z k r s 1 M 2 0 1 M 1 0	**BSS** 12784.29 - 47 z k r s 8 M 2 0 1 M 3 0	**CONWIP** 1571.18 - 3 z k r s 1 1 1 0 0 1 1 0	**BSS** 10841.12 - 32 z k r s 5 M 1 0 1 M 2 0
8	**Kanban** 1427.71 - 5 z k r s 1 1 1 0 1 1 1 0	**PTO, τ≥0** 8023.74 - 5 z k r s 0 M 1 0 0 M 1 3	**Kanban** 1416.67 - 5 z k r s 1 1 1 0 1 1 1 0	**PTO, τ≥0** 4866.93 - 5 z k r s 0 M 1 0 0 M 1 0	**Kanban** 1833.41 - 5 z k r s 1 1 1 0 1 1 1 0	**PTO, τ≥0** 17398.84 - 14 z k r s 0 M 1 0 0 M 1 10	**Kanban** 1725.42 - 5 z k r s 1 1 1 0 1 1 1 0	**PTO, τ≥0** 14790.07 - 5 z k r s 0 M 1 0 0 M 1 10
9	**LC** 2948.86 - 5 z k r s 2 1 1 0 2 1 1 0	**PTO** 8023.74 - 1 z k r s 0 M 1 0 0 M 1 0	**LC** 2956.78 - 5 z k r s 2 1 1 0 2 1 1 0	**PTO** 4866.93 - 1 z k r s 0 M 1 0 0 M 1 0	**LC** 3087.60 - 5 z k r s 2 1 1 0 2 1 1 0	**PTO** 18243.87 - 1 z k r s 0 M 1 0 0 M 1 0	**LC** 3073.20 - 5 z k r s 2 1 1 0 2 1 1 0	**PTO** 14790.07 - 1 z k r s 0 M 1 0 0 M 1 0

* by using other policy best solution

Table c1-2 **Model 1** case: ρ_2 =0.1 ($1/\mu_2$ = 6 min.)
DCII [double cost; ORD-REQ]

Best Solu-tion	exponential service time		uniform service time		exponential + bulk service time demand		uniform + bulk service time demand	
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7	0.1	0.7
1	PAC 256.81 - 78 z k r s 0 3 1 16 0 1 1 4	PAC 4180.27 - 142 z k r s 2 2 1 69 0 1 1 0	PAC 59.51 - 102 z k r s 0 3 1 13 0 1 1 0	PAC 2191.83 - 60 z k r s 2 2 1 19 0 1 1 0	PAC 519.68 - 129 z k r s 0 11 1 27 0 1 1 1	PAC 7718.23 - 417 z k r s 0 3 1 340 0 2 1 11	PAC 275.26 - 56 z k r s 0 5 1 29 0 1 1 1	PAC 6196.71 - 96 z k r s 0 2 1 280 0 1 1 0
2	MRP 262.43 - 63 z k r s 0 M 1 16 0 M 1 4	LC 4407.07	MRP 62.38 -83 z k r s 0 M 1 13 0 M 1 0	MRP * 2215.86 -25 z k r s 0 M 1 103 0 M 1 0	MRP 591.93 -74 z k r s 0 M 1 27 0 M 1 0	LC 8566.64	MRP 337.39 -30 z k r s 0 M 1 20 0 M 1 0	LC 7938.51
3	PTO, $\tau\geq0$ 262.43 - 31 z k r s 0 M 1 16 0 M 1 4	IC 4705.03	PTO, $\tau\geq0$ 62.38 - 24 z k r s 0 M 1 13 0 M 1 0	PTO, $\tau\geq0$ 2215.86 - 31 z k r s 0 M 1 103 0 M 1 0	PTO, $\tau\geq0$ 591.93 - 34 z k r s 0 M 1 27 0 M 1 0	IC 9748.77	PTO, $\tau\geq0$ 337.39 -18 z k r s 0 M 1 20 0 M 1 0	IC 8651.54
4	PTO 600.63	CONWIP 4705.03	PTO 545.46	IC 2253.65	PTO 1148.33	CONWIP 9748.77	PTO 921.76	CONWIP 8651.54
5	IC 933.36	Kanban 4805.23	IC 897.21	CONWIP 2253.65	IC 1617.75	Kanban 10159.07	BSS 1449.91	Kanban 9074.22
6	CONWIP 933.36	MRP * 4991.14 - 25 z k r s 2 M 1 70 0 M 1 0	CONWIP 897.21	Kanban 2717.04	CONWIP 1617.75	MRP 11080.10-145 z k r s 0 M 2 361 0 M 1 20	IC 1571.18	MRP 9326.04- 77 z k r s 0 M 1 281 0 M 1 0
7	BSS 1125.41	PTO, $\tau\geq0$ 4991.14 - 55 z k r s 0 M 1 168 0 M 1 3	BSS 1077.36	LC 2821.14	BSS 1620.75	PTO, $\tau\geq0$ 11445.36 - 65 z k r s 0 M 1 360 0 M 1 19	CONWIP 1571.18	PTO, $\tau\geq0$ 9326.04- 39 z k r s 0 M 1 281 0 M 1 0
8	Kanban 1427.71	BSS 5017.97	Kanban 1416.67	BSS 2915.86	Kanban 1833.4	BSS 12784.29	Kanban 1725.42	BSS 10841.12
9	LC 2948.86	PTO 8023.74	LC 2956.78	PTO 4866.93	LC 3087.60	PTO 18243.87	LC 3073.20	PTO 14790.07

* by using other policy best solution

Table c1-3 **Model 1** case: ρ_2 =0.7 ($1/\mu_2$ = 42 min.)
DCI [single cost; from arrival of ORD-tag]

Best Solu-tion	exponential service time		uniform service time		exponential + bulk service time demand		uniform + bulk service time demand	
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7	0.1	0.7
1	**PAC** 2622.80 - 85 z k r s 1 4 1 0 3 1 1 11	**PAC** 6006.30 - 120 z k r s 5 15 1 0 2 1 1 32	**PAC *** 1535.69 - 61 z k r s 0 1 1 0 3 1 1 0	**PAC *** 2013.00 - 25 z k r s 2 2 1 0 1 1 1 0	**PAC** 6063.78 - 81 z k r s 0 2 1 0 10 1 1 3	**PAC *** 12715.33- 42 z k r s 7 2 1 0 9 1 1 0	**PAC** 5541.76 - 51 z k r s 0 1 1 0 8 1 1 0	**PAC** 7151.33 - 94 z k r s 5 2 1 0 1 1 1 0
2	**LC** 3372.18 - 13 z k r s 2 1 1 0 3 1 1 0	**LC** 6226.76 - 22 z k r s 4 1 1 0 6 1 1 0	**Kanban** 1732.98 - 5 z k r s 1 1 1 0 1 1 1 0	**Kanban** 2013.00 - 27 z k r s 2 2 1 0 1 1 1 0	**LC** 6296.30 - 23 z k r s 2 1 1 0 6 1 1 0	**LC** 12788.70- 33 z k r s 7 1 1 0 9 1 1 0	**LC** 5952.61 - 23 z k r s 2 1 1 0 6 1 1 0	**Kanban** 7151.33- 23 z k r s 5 5 1 0 1 1 1 0
3	**MRP** 3399.93 - 119 z k r s 1 M 1 0 4 M 1 9	**Kanban** 6618.27 - 37 z k r s 4 4 1 0 3 3 1 0	**IC** 2066.26 - 21 z k r s 2 2 1 0 0 2 1 0	**IC** 2335.74 - 14 z k r s 3 3 1 0 0 3 1 0	**Kanban** 7380.04 - 42 z k r s 1 1 1 0 5 5 1 0	**Kanban** 13093.55- 38 z k r s 10 10 1 0 1 1 1 0	**Kanban** 6831.28- 33 z k r s 4 4 1 0 1 1 1 0	**LC** 7609.63 - 25 z k r s 5 1 1 0 2 1 1 0
4	**Kanban** 3410.14 - 30 z k r s 1 1 1 0 3 3 1 0	**IC** 7245.95 - 22 z k r s 3 3 1 0 4 7 1 0	**CONWIP** 2066.26 - 15 z k r s 2 2 1 0 0 2 1 0	**CONWIP** 2335.74 - 10 z k r s 3 3 1 0 0 3 1 0	**IC** 7662.88 - 30 z k r s 1 1 1 0 7 8 1 0	**IC** 14682.68- 39 z k r s 10 10 1 0 0 10 1 0	**IC** 6878.66 - 24 z k r s 1 1 1 0 6 7 1 0	**IC** 8091.59- 28 z k r s 5 5 1 0 1 6 1 0
5	**BSS** 3661.84 - 53 z k r s 1 M 2 0 4 M 1 0	**CONWIP** 7291.88 - 17 z k r s 6 6 1 0 0 6 1 0	**BSS** 2066.68 - 21 z k r s 1 M 1 0 2 M 1 0	**LC** 2499.11 - 5 z k r s 2 1 1 0 2 1 1 0	**BSS** 8445.31 - 44 z k r s 1 M 2 0 9 M 1 0	**CONWIP** 14682.68- 20 z k r s 10 10 1 0 0 10 1 0	**MRP** 7424.83 - 47 z k r s 0 M 1 0 8 M 1 0	**CONWIP** 8215.31- 13 z k r s 5 5 1 0 0 5 1 0
6	**IC** 3863.63 - 28 z k r s 2 2 1 0 1 3 1 0	**BSS** 7427.41 - 51 z k r s 4 M 1 0 3 M 1 0	**MRP** 2289.70 -30 z k r s 2 M 1 0 0 M 1 0	**MRP** 2535.90 - 45 z k r s 3 M 1 0 0 M 1 0	**MRP** 8742.13 -90 z k r s 1 M 1 0 9 M 1 18	**MRP** 15455.51- 89 z k r s 9 M 2 0 4 M 1 40	**BSS** 7578.54- 73 z k r s 1 M 1 0 7 M 1 0	**MRP** 9016.04- 79 z k r s 5 M 1 0 1 M 1 0
7	**CONWIP** 4091.68 - 10 z k r s 3 3 1 0 0 3 1 0	**MRP** 7446.06 - 66 z k r s 4 M 2 0 4 M 1 1	**LC** 2291.04 - 5 z k r s 2 1 1 0 2 1 1 0	**BSS** 2648.96 - 25 z k r s 3 M 2 0 1 M 1 0	**CONWIP** 9898.16 - 13 z k r s 5 5 1 0 0 5 1 0	**BSS** 15565.21- 54 z k r s 9 M 2 0 5 M 4 0	**CONWIP** 7897.08- 13 z k r s 5 5 1 0 0 5 1 0	**BSS** 9016.04- 28 z k r s 5 M 1 0 1 M 1 0
8	**PTO, $\tau \geq 0$** 7407.44 - 5 z k r s 0 M 1 0 0 M 1 0	**PTO, $\tau \geq 0$** 14660.46 - 36 z k r s 0 M 1 0 0 M 1 13	**PTO, $\tau \geq 0$** 4454.76 - 5 z k r s 0 M 1 0 0 M 1 0	**PTO, $\tau \geq 0$** 6077.12 - 5 z k r s 0 M 1 0 0 M 1 0	**PTO, $\tau \geq 0$** 15609.64 - 5 z k r s 0 M 1 0 0 M 1 0	**PTO, $\tau \geq 0$** 25443.59 - 5 z k r s 0 M 1 0 0 M 1 0	**PTO, $\tau \geq 0$** 12993.23 - 5 z k r s 0 M 1 0 0 M 1 0	**PTO, $\tau \geq 0$** 14693.34 - 5 z k r s 0 M 1 0 0 M 1 0
9	**PTO** 7407.44 - 1 z k r s 0 M 1 0 0 M 1 0	**PTO** 14739.34 - 1 z k r s 0 M 1 0 0 M 1 0	**PTO** 4454.76 - 1 z k r s 0 M 1 0 0 M 1 0	**PTO** 6077.12 - 1 z k r s 0 M 1 0 0 M 1 0	**PTO** 15609.64 - 1 z k r s 0 M 1 0 0 M 1 0	**PTO** 25443.59 - 1 z k r s 0 M 1 0 0 M 1 0	**PTO** 12993.23 - 1 z k r s 0 M 1 0 0 M 1 0	**PTO** 14693.34 - 1 z k r s 0 M 1 0 0 M 1 0

* by using other policy best solution

Table c1-4 **Model 1** case: $\rho_2 = 0.7$ ($1/\mu_2 = 42$ min.)
DCII [double cost; ORD-REQ]

Best Solu -tion	exponential service time		uniform service time		exponential + bulk service time demand		uniform + bulk service time demand	
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7	0.1	0.7
1	PAC 2526.58 - 174 z k r s 0 4 1 57 2 1 1 12	PAC 5972.84 - 152 z k r s 4 15 1 28 2 1 1 32	PAC * 1590.07 - 88 z k r s 1 3 1 38 1 1 1 32	PAC 1503.28 - 86 z k r s 0 9 1 140 0 1 1 0	PAC 5419.06 - 131 z k r s 0 6 1 64 8 1 1 38	PAC 12126.58- 430 z k r s 0 7 1 439 4 1 1 60	PAC 5101.11 - 188 z k r s 0 47 1 379 0 1 1 360	PAC 5884.77 - 157 z k r s 0 3 1 277 1 1 1 0
2	MRP 3274.99 -164 z k r s 0 M 1 13 4 M 1 1	LC 6226.76	Kanban 1732.98	MRP 1913.68 - 69 z k r s 3 M 1 140 0 M 1 0	LC 6296.30	LC 12788.70	LC 5952.61	Kanban 7151.33
3	LC 3372.18	Kanban 6618.27	MRP * 1805.65 - 25 z k r s 0 M 1 103 0 M 1 0	PTO, τ≥0 1913.68 - 31 z k r s 0 M 1 140 0 M 1 0	Kanban 7380.04	Kanban 13093.55	MRP 6523.94 - 364 z k r s 0 M 1 371 0 M 1 351	LC 7609.63
4	Kanban 3410.14	MRP 7104.55 -194 z k r s 5 M 2 40 2 M 1 50	PTO, τ≥0 1805.65 - 31 z k r s 0 M 1 103 0 M 1 0	Kanban 2013.00	IC 7662.88	MRP 14418.85- 194 z k r s 0 M 2 526 2 M 1 40	Kanban 6831.28	MRP 7686.46- 106 z k r s 0 M 1 319 0 M 1 0
5	PTO, τ≥0 3430.57 - 44 z k r s 0 M 1 221 0 M 1 209	IC 7245.95	IC 2066.26	IC 2335.74	MRP 7803.50 -197 z k r s 0 M 1 63 9 M 1 40	PTO, τ≥0 14400.48 - 47 z k r s 0 M 1 538 0 M 1 2	IC 6878.66	PTO, τ≥0 7686.46- 54 z k r s 0 M 1 319 0 M 1 0
6	BSS 3661.84	CONWIP 7291.88	CONWIP 2066.26	CONWIP 2335.74	BSS 8445.31	IC 14682.68	PTO, τ≥0 7527.20 - 27 z k r s 0 M 1 280 0 M 1 0	IC 8091.59
7	IC 3863.63	BSS 7427.41	BSS 2066.68	LC 2499.11	CONWIP 9898.16	CONWIP 14682.68	BSS 7578.54	CONWIP 8215.31
8	CONWIP 4091.68	PTO, τ≥0 7436.17 - 66 z k r s 0 M 1 334 0 M 1 30	LC 2291.04	BSS 2648.96	PTO, τ≥0 10020.50 - 56 z k r s 0 M 1 325 0 M 1 5	BSS 15565.21	CONWIP 7897.08	BSS 9016.04
9	PTO 7407.44	PTO 14739.34	PTO 4454.76	PTO 6077.12	PTO 15609.64	PTO 25443.59	PTO 12993.23	PTO 14693.34

* by using other policy best solution

Appendix C2

OPTIMIZATION RESULTS FOR MODEL 2

Table c2-1 **Model 2A** case: ρ_2 =0.1, ρ_3 =0.1 ($1/\mu_2$ = 6 min., $1/\mu_3$= 6 min.)
DCI [single cost; from arrival of ORD-tag]

Best Solution	exponential service time		uniform service time		exponential + bulk service time demand	
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7
1	PAC 799.54 - 138 z k r s 0 7 1 0 1 5 1 0 0 7 1 0	PAC 4199.06 - 187 z k r s 3 3 1 0 0 5 1 0 0 5 1 0	PAC 739.53 - 97 z k r s 0 5 1 0 1 5 1 0 0 1 1 0	PAC 2255.41 - 73 z k r s 2 2 1 0 0 1 1 0 0 1 1 0	PAC 1424.24 - 136 z k r s 0 7 1 0 0 6 1 0 1 1 1 0	PAC 8866.01 - 95 z k r s 6 1 1 0 2 1 1 0 0 1 1 0
2	MRP 802.31 - 41 z k r s 0 M 1 0 1 M 1 0 0 M 1 0	IC 4204.23 - 18 z k r s 3 3 1 0 0 3 1 0 0 3 1 0	MRP 740.81 - 41 z k r s 0 M 1 0 1 M 1 0 0 M 1 0	IC 2260.32 - 27 z k r s 2 2 1 0 0 2 1 0 0 2 1 0	MRP 1478.46 - 41 z k r s 0 M 1 0 1 M 1 0 0 M 1 0	LC 10066.62 - 30 z k r s 6 1 1 0 2 1 1 0 2 1 1 0
3	PTO, τ≥0 894.79 - 7 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	CONWIP 4204.23 - 10 z k r s 3 3 1 0 0 3 1 0 0 3 1 0	PTO, τ≥0 807.93 - 7 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	CONWIP 2260.32 - 15 z k r s 2 2 1 0 0 2 1 0 0 2 1 0	PTO, τ≥0 1502.70 - 7 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	IC 10442.41 - 26 z k r s 4 4 1 0 0 4 1 0 0 4 1 0
4	PTO 894.79 - 1	MRP 4897.23 - 154 z k r s 4 M 1 0 0 M 1 50 0 M 1 10	PTO 807.93 - 1	MRP 2749.71 - 42 z k r s 2 M 1 0 0 M 1 0 0 M 1 0	PTO 1502.70 - 1	CONWIP 10442.41 - 14 z k r s 4 4 1 0 0 4 1 0 0 4 1 0
5	IC 948.92 - 7 z k r s 1 1 1 0 0 1 1 0 0 1 1 0	Kanban 4949.02 - 16 z k r s 2 2 1 0 1 1 1 0 1 1 1 0	IC 896.66 - 7 z k r s 1 1 1 0 0 1 1 0 0 1 1 0	Kanban 2950.96 - 20 z k r s 3 3 1 0 1 1 1 0 1 1 1 0	BSS 1865.22 - 30 z k r s 1 M 2 0 1 M 1 0 1 M 1 0	Kanban 10890.72 - 22 z k r s 3 3 1 0 1 1 1 0 1 1 1 0
6	CONWIP 948.92 - 7 z k r s 1 1 1 0 0 1 1 0 0 1 1 0	LC 4977.36 - 17 z k r s 3 1 1 0 2 1 1 0 2 1 1 0	CONWIP 896.66 - 3 z k r s 1 1 1 0 0 1 1 0 0 1 1 0	BSS 3134.34 - 50 z k r s 2 M 1 0 1 M 2 0 1 M 1 0	IC 1920.24 - 27 z k r s 1 1 1 0 0 1 1 0 1 2 1 0	MRP 11641.74 - 108 z k r s 7 M 2 0 0 M 1 13 0 M 1 0
7	BSS 1347.32 - 30 z k r s 1 M 2 0 1 M 1 0 1 M 1 0	BSS 5141.39 - 33 z k r s 3 M 1 0 1 M 2 0 1 M 1 0	BSS 1297.80 - 30 z k r s 1 M 2 0 1 M 1 0 1 M 1 0	LC 3315.01 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	Kanban 2089.32 - 7 z k r s 1 1 1 0 1 1 1 0 1 1 1 0	PTO, τ≥0 17771.60 - 17 z k r s 0 M 1 0 0 M 1 10 0 M 1 0
8	Kanban 1658.18 - 7 z k r s 1 1 1 0 1 1 1 0 1 1 1 0	PTO, τ≥0 7656.56 - 21 z k r s 0 M 1 0 0 M 1 0 0 M 1 3	Kanban 1650.11 - 7 z k r s 1 1 1 0 1 1 1 0 1 1 1 0	PTO, τ≥0 5122.42 - 7 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	CONWIP 2113.05 - 3 z k r s 1 1 1 0 0 1 1 0 0 1 1 0	BSS 12573.79 - 58 z k r s 6 M 1 0 1 M 1 0 1 M 2 0
9	LC 3440.59 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	PTO 7862.95 - 1	LC 3450.34 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	PTO 5122.42 - 1	LC 3562.64 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	PTO 18480.51 - 1

Table c2-2 **Model 2A** case: ρ_2 =0.1, ρ_3 =0.1 ($1/\mu_2$ = 6 min., $1/\mu_3$= 6 min.)
DCII [double cost; ORD-REQ]

Best Solution	exponential service time		uniform service time		exponential + bulk service time demand	
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7
1	**PAC** 316.75 - 174 z k r s 0 5 1 24 0 7 1 11 0 1 1 4	**PAC** 4077.81 - 233 z k r s 2 3 1 75 0 7 1 0 0 7 1 10	**PAC** 76.17 - 99 z k r s 0 4 1 19 0 3 1 0 0 1 1 0	**PAC** 2036.44 - 229 z k r s 2 2 1 54 0 1 1 0 0 1 1 35	**PAC** 604.29 - 185 z k r s 0 11 1 38 0 9 1 11 0 1 1 3	**PAC** 7802.97 - 191 z k r s 0 1 1 322 2 1 1 0 0 1 1 0
2	**MRP** * 321.02 - 135 z k r s 0 M 1 25 0 M 1 12 0 M 1 6	**IC** 4204.23	**MRP** 77.61 - 63 z k r s 0 M 1 19 0 M 1 0 0 M 1 0	**MRP** * 2222.78 - 56 z k r s 0 M 1 109 0 M 1 0 0 M 1 0	**MRP** * 653.34 - 160 z k r s 0 M 1 38 0 M 1 3 0 M 1 7	**LC** 10066.62
3	**PTO, τ≥0** 321.02 - 78 z k r s 0 M 1 25 0 M 1 12 0 M 1 6	**CONWIP** 4204.23	**PTO, τ≥0** 77.61 - 32 z k r s 0 M 1 19 0 M 1 0 0 M 1 0	**PTO, τ≥0** 2222.78 - 48 z k r s 0 M 1 109 0 M 1 0 0 M 1 0	**PTO, τ≥0** 653.34 - 76 z k r s 0 M 1 38 0 M 1 3 0 M 1 7	**IC** 10442.41
4	**PTO** 894.79	**MRP** * 4548.16 - 54 z k r s 0 M 1 161 0 M 1 3 0 M 1 3	**PTO** 807.93	**IC** 2260.32	**PTO** 1502.70	**CONWIP** 10442.41
5	**IC** 948.92	**PTO, τ≥0** 4548.16 - 65 z k r s 0 M 1 161 0 M 1 3 0 M 1 3	**IC** 896.66	**CONWIP** 2260.32	**BSS** 1865.22	**MRP** 10547.47 - 325 z k r s 0 M 2 369 0 M 1 13 0 M 1 0
6	**CONWIP** 948.92	**Kanban** 4949.02	**CONWIP** 896.66	**Kanban** 2950.96	**IC** 1920.24	**Kanban** 10890.72
7	**BSS** 1347.32	**LC** 4977.36	**BSS** 1297.80	**BSS** 3134.34	**Kanban** 2089.32	**PTO, τ≥0** 10992.38 - 112 z k r s 0 M 1 413 0 M 1 60 0 M 1 84
8	**Kanban** 1658.18	**BSS** 5141.39	**Kanban** 1650.11	**LC** 3315.01	**CONWIP** 2113.05	**BSS** 12573.79
9	**LC** 3440.59	**PTO** 7862.95	**LC** 3450.34	**PTO** 5122.42	**LC** 3562.64	**PTO** 18480.51

* by using other policy best solution

Table c2-3 **Model 2A** case: ρ_2 =0.1, ρ_3 =0.7 ($1/\mu_2$ = 6 min., $1/\mu_3$= 42 min.)
DCI [single cost; from arrival of ORD-tag]

Best Solution	exponential service time		uniform service time		exponential + bulk	service time demand
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7
1	PAC 2288.41 - 199 z k r s 3 5 1 0 1 3 3 12 4 1 1 1	PAC 5313.83 - 193 z k r s 4 3 1 0 2 3 2 10 5 1 1 20	PAC 1436.02 - 376 z k r s 4 7 1 0 0 3 2 60 3 1 1 0	PAC * 2159.31 - 60 z k r s 3 3 1 0 0 3 1 0 0 1 1 0	PAC 4458.99 - 189 z k r s 0 11 1 0 1 1 1 2 11 1 1 0	PAC 11701.46 - 150 z k r s 8 3 1 0 2 1 1 18 4 1 1 0
2	Kanban 2772.79 - 22 z k r s 1 1 1 0 3 3 1 0 1 1 1 0	LC 5781.23 - 55 z k r s 3 1 1 0 3 1 1 0 4 1 1 0	IC 1574.99 - 33 z k r s 1 1 1 0 0 1 1 0 2 3 1 0	IC 2257.86 - 18 z k r s 3 3 1 0 0 3 1 0 0 3 1 0	LC 5615.80 - 43 z k r s 2 1 1 0 2 1 1 0 9 1 1 0	LC 11850.91 - 57 z k r s 7 1 1 0 3 1 1 0 10 1 1 0
3	BSS 2871.00 - 51 z k r s 1 M 2 0 1 M 1 0 5 M 1 0	MRP 6123.39 - 172 z k r s 4 M 1 0 0 M 1 22 3 M 1 20	Kanban 1587.22 - 7 z k r s 1 1 1 0 1 1 1 0 1 1 1 0	CONWIP 2257.86 - 10 z k r s 3 3 1 0 0 3 1 0 0 3 1 0	Kanban 5625.04 - 76 z k r s 1 1 1 0 1 1 1 0 10 10 1 0	Kanban 12879.58 - 38 z k r s 5 5 1 0 1 1 1 0 5 5 1 0
4	MRP 2887.89 - 77 z k r s 2 M 1 0 0 M 2 50 4 M 1 0	Kanban 6234.67 - 52 z k r s 4 4 1 0 1 1 1 0 2 2 1 0	MRP 1635.97 - 79 z k r s 1 M 1 0 0 M 1 0 2 M 1 3	MRP 2381.92 - 48 z k r s 2 M 1 0 0 M 1 0 2 M 2 0	BSS 5856.26- 110 z k r s 1 M 2 0 1 M 1 0 13 M 3 0	IC 13333.88 - 29 z k r s 7 7 1 0 2 9 1 0 6 15 1 0
5	LC 3163.67 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	IC 6559.72 - 24 z k r s 3 3 1 0 2 5 1 0 2 7 1 0	BSS 1792.44- 13 z k r s 1 M 1 0 1 M 1 0 1 M 1 0	Kanban 2560.91 - 20 z k r s 2 2 1 0 1 1 1 0 1 1 1 0	MRP 6624.43 - 143 z k r s 1 M 1 0 6 M 2 10 4 M 1 49	CONWIP 13505.40 - 19 z k r s 12 12 1 0 0 12 1 0 0 12 1 0
6	IC 3764.87 - 18 z k r s 3 3 1 0 0 3 1 0 0 3 1 0	BSS 6896.25- 57 z k r s 4 M 2 0 3 M 1 0 1 M 2 0	CONWIP 2016.64 - 15 z k r s 2 2 1 0 0 2 1 0 0 2 1 0	BSS 2979.90- 33 z k r s 3 M 2 0 1 M 1 0 1 M 1 0	IC 6689.17 - 45 z k r s 1 1 1 0 6 7 1 0 4 11 1 0	MRP 13627.91 - 118 z k r s 8 M 1 0 4 M 1 21 0 M 1 21
7	CONWIP 3764.87 - 10 z k r s 3 3 1 0 0 3 1 0 0 3 1 0	CONWIP 7280.26 - 12 z k r s 7 7 1 0 0 7 1 0 0 7 1 0	LC 2975.00 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	LC 3153.86 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	CONWIP 9016.10 - 18 z k r s 6 6 1 0 0 6 1 0 0 6 1 0	BSS 13653.85- 82 z k r s 7 M 1 0 1 M 1 0 5 M 3 0
8	PTO, τ≥0 6833.57 - 44 z k r s 0 M 1 0 0 M 1 17 0 M 1 18	PTO, τ≥0 14641.81 - 17 z k r s 0 M 1 0 0 M 1 0 0 M 1 20	PTO, τ≥0 4512.88 - 7 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	PTO, τ≥0 6145.75 - 7 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	PTO, τ≥0 14754.31 - 7 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	PTO, τ≥0 24845.78 - 137 z k r s 0 M 1 0 0 M 1 60 0 M 1 21
9	PTO 6966.56 - 1	PTO 14844.80 - 1	PTO 4512.88 - 1	PTO 6145.75 - 1	PTO 14754.31 - 1	PTO 25455.48 - 1

* by using other policy best solution

Table c2-4 **Model 2A** case: ρ_2 =0.1, ρ_3 =0.7 ($1/\mu_2$ = 6 min., $1/\mu_3$= 42 min.)
DCII [double cost; ORD-REQ]

Best Solution	exponential service time		uniform service time		exponential + bulk service time demand	
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7
1	**PAC *** 2509.58 - 296 z k r s 4 11 1 232 2 7 2 304 0 1 1 210	**PAC** 5497.71 - 265 z k r s 2 3 1 105 2 7 3 20 4 1 1 30	**PAC *** 1412.04 - 176 z k r s 1 3 1 18 0 4 1 0 2 1 1 20	**PAC *** 1670.32 - 230 z k r s 0 5 1 132 2 9 3 9 0 1 1 0	**PAC** 3710.00 - 353 z k r s 0 29 1 74 0 9 1 40 10 1 1 40	**PAC** 10759.86 - 668 z k r s 0 3 1 432 2 4 1 31 5 1 1 20
2	**MRP *** 2630.70 - 22 z k r s 0 M 1 318 0 M 1 160 0 M 1 300	**LC** 5781.23	**IC** 1574.99	**MRP** 1725.78 - 137 z k r s 0 M 1 146 0 M 1 0 0 M 1 0	**MRP** 4845.57 - 206 z k r s 0 M 1 55 0 M 1 20 10 M 1 20	**LC** 11850.91
3	**PTO, $\tau \geq 0$** 2630.70 - 88 z k r s 0 M 1 318 0 M 1 160 0 M 1 300	**Kanban** 6234.67	**Kanban** 1587.22	**PTO, $\tau \geq 0$** 1725.78 - 52 z k r s 0 M 1 146 0 M 1 0 0 M 1 0	**PTO, $\tau \geq 0$** 5260.87 - 105 z k r s 0 M 1 475 0 M 1 240 0 M 1 440	**MRP** 12662.24 - 311 z k r s 0 M 1 405 4 M 1 20 0 M 1 21
4	**Kanban** 2772.79	**MRP** 6268.10 - 157 z k r s 2 M 1 78 2 M 2 0 3 M 1 40	**MRP *** 1610.98 - 22 z k r s 0 M 1 109 0 M 1 0 0 M 1 0	**IC** 2257.86	**LC** 5615.80	**PTO, $\tau \geq 0$** 12782.40 - 95 z k r s 0 M 1 546 0 M 1 60 0 M 1 21
5	**BSS** 2871.00	**IC** 6559.72	**PTO, $\tau \geq 0$** 1610.98 - 48 z k r s 0 M 1 109 0 M 1 0 0 M 1 0	**CONWIP** 2257.86	**Kanban** 5625.04	**Kanban** 12879.58
6	**LC** 3163.67	**PTO, $\tau \geq 0$** 6694.13 - 96 z k r s 0 M 1 411 0 M 1 0 0 M 1 220	**BSS** 1792.44	**Kanban** 2560.91	**BSS** 5856.26	**IC** 13333.88
7	**IC** 3764.87	**BSS** 6896.25	**CONWIP** 2016.64	**BSS** 2979.90	**IC** 6689.17	**CONWIP** 13505.40
8	**CONWIP** 3764.87	**CONWIP** 7280.26	**LC** 2975.00	**LC** 3153.86	**CONWIP** 9016.10	**BSS** 13653.85
9	**PTO** 6966.56	**PTO** 14844.80	**PTO** 4512.88	**PTO** 6145.75	**PTO** 14754.31	**PTO** 25455.48

* by using other policy best solution

Table c2-5 **Model 2A** case: ρ_2 =0.7, ρ_3 =0.1 ($1/\mu_2$ = 42 min., $1/\mu_3$ = 6 min.)
DCI [single cost; from arrival of ORD-tag]

Best Solution	exponential service time		uniform service time		exponential + bulk service time demand	
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7
1	PAC * 3137.51 - 155 z k r s 1 5 1 0 4 3 1 9 0 1 1 1	PAC 6191.76 - 250 z k r s 4 4 1 0 3 2 1 19 0 5 1 2	PAC 1730.94 - 94 z k r s 0 1 1 0 3 2 1 0 0 1 1 0	PAC 2233.52 - 109 z k r s 2 2 1 0 1 2 1 0 0 1 1 0	PAC 7596.01 - 222 z k r s 4 8 1 0 4 5 1 64 0 7 3 5	PAC 12768.14 - 186 z k r s 8 6 1 0 4 4 1 44 0 1 1 1
2	IC 3362.38 - 53 z k r s 1 1 1 0 2 3 1 0 0 3 1 0	LC 6601.90 - 37 z k r s 4 1 1 0 5 1 1 0 2 1 1 0	Kanban 1965.56 - 7 z k r s 1 1 1 0 1 1 1 0 1 1 1 0	Kanban 2246.45 - 33 z k r s 2 2 1 0 1 1 1 0 1 1 1 0	IC 7674.06 - 43 z k r s 1 1 1 0 8 9 1 0 1 10 1 0	Kanban 13011.69 - 46 z k r s 10 10 1 0 1 1 1 0 1 1 1 0
3	MRP 3534.58 - 88 z k r s 2 M 1 0 2 M 1 39 0 M 1 1	Kanban 6654.13 - 34 z k r s 5 5 1 0 1 1 1 0 1 1 1 0	IC 2098.27 - 27 z k r s 2 2 1 0 0 2 1 0 0 2 1 0	IC 2347.94 - 18 z k r s 3 3 1 0 0 3 1 0 0 3 1 0	LC 7760.84 - 42 z k r s 3 1 1 0 4 1 1 0 2 1 1 0	LC 13566.34 - 35 z k r s 8 1 1 0 7 1 1 0 2 1 1 0
4	LC 3607.68 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	IC 6973.92 - 38 z k r s 4 4 1 0 4 8 1 0 0 8 1 0	CONWIP 2098.27 - 15 z k r s 2 2 1 0 0 2 1 0 0 2 1 0	CONWIP 2347.94 - 10 z k r s 3 3 1 0 0 3 1 0 0 3 1 0	Kanban 7857.71 - 67 z k r s 5 5 1 0 1 1 1 0 2 2 2 0	IC 14513.30 - 40 z k r s 8 8 1 0 4 12 1 0 1 13 1 0
5	BSS 3659.71 - 58 z k r s 1 M 2 0 4 M 1 0 1 M 2 0	CONWIP 7014.13 - 17 z k r s 6 6 1 0 0 6 1 0 0 6 1 0	BSS 2189.91 - 48 z k r s 1 M 1 0 2 M 1 0 1 M 2 0	MRP 2770.22 - 56 z k r s 4 M 2 0 0 M 1 0 0 M 1 0	MRP 7981.54 - 114 z k r s 1 M 2 0 8 M 1 0 0 M 1 0	CONWIP 14908.24 - 14 z k r s 13 13 1 0 0 13 1 0 0 13 1 0
6	Kanban 3775.63 - 27 z k r s 2 2 1 0 1 1 1 0 1 1 1 0	MRP 7277.13 - 88 z k r s 4 M 1 0 4 M 4 0 0 M 1 1	MRP 2343.48 - 42 z k r s 2 M 1 0 0 M 1 0 0 M 1 0	BSS 2870.98 - 33 z k r s 3 M 2 0 1 M 1 0 1 M 1 0	BSS 8346.63 - 62 z k r s 1 M 1 0 8 M 1 0 1 M 1 0	MRP 14946.94 - 85 z k r s 8 M 1 0 4 M 1 20 0 M 1 0
7	CONWIP 4222.90 - 10 z k r s 3 3 1 0 0 3 1 0 0 3 1 0	BSS 7580.77 - 53 z k r s 5 M 2 0 3 M 1 0 1 M 1 0	LC 2783.64 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	LC 2992.80 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	CONWIP 9473.37 - 13 z k r s 5 5 1 0 0 5 1 0 0 5 1 0	BSS 15522.30 - 94 z k r s 9 M 4 0 4 M 1 0 2 M 1 0
8	PTO, τ≥0 7574.52 - 22 z k r s 0 M 1 0 0 M 1 20 0 M 1 0	PTO, τ≥0 14367.47 - 29 z k r s 0 M 1 0 0 M 1 10 0 M 1 0	PTO, τ≥0 4717.33 - 7 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	PTO, τ≥0 6339.24 - 7 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	PTO, τ≥0 15900.28 - 7 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	PTO, τ≥0 25414.62 - 21 z k r s 0 M 1 0 0 M 1 0 0 M 1 2
9	PTO 7725.34 - 1	PTO 14715.94 - 1	PTO 4717.33 - 1	PTO 6339.24 - 1	PTO 15900.28 - 1	PTO 26161.46 - 1

* by using other policy best solution

Table c2-6 **Model 2A** case: ρ_2 =0.7, ρ_3 =0.1 ($1/\mu_2$ = 42 min., $1/\mu_3$= 6 min.)
DCII [double cost; ORD-REQ]

Best Solution	exponential service time		uniform service time		exponential + bulk service time demand	
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7
1	PAC * 3252.00 - 196 z k r s 1 2 1 11 2 3 1 0 0 2 1 1	PAC 6173.48 - 161 z k r s 6 7 1 0 1 3 1 101 0 9 1 21	PAC 1433.53 - 234 z k r s 0 5 1 41 2 2 1 34 0 1 1 0	PAC 1859.07 - 164 z k r s 0 1 1 97 2 5 1 0 0 1 1 40	PAC 6672.62 - 316 z k r s 0 17 1 148 8 4 1 120 0 9 4 6	PAC 12333.59 - 374 z k r s 0 5 1 413 6 5 1 42 0 1 1 0
2	IC 3362.38	LC 6601.90	MRP * 1816.40 - 48 z k r s 0 M 1 109 0 M 1 0 0 M 1 0	MRP 1924.44 - 137 z k r s 0 M 1 146 0 M 1 0 0 M 1 0	IC 7674.06	Kanban 13011.69
3	MRP 3479.61 - 198 z k r s 1 M 1 24 4 M 2 32 0 M 1 3	Kanban 6654.13	PTO, $\tau\geq0$ 1816.40 - 48 z k r s 0 M 1 109 0 M 1 0 0 M 1 0	PTO, $\tau\geq0$ 1924.44 - 52 z k r s 0 M 1 146 0 M 1 0 0 M 1 0	LC 7760.84	LC 13566.34
4	PTO, $\tau\geq0$ 3481.76 - 103 z k r s 0 M 1 167 0 M 1 140 0 M 1 0	MRP * 6756.57 - 22 z k r s 0 M 1 309 0 M 1 40 0 M 1 1	Kanban 1965.56	Kanban 2246.45	MRP 7932.68 - 133 z k r s 2 M 3 49 8 M 1 30 0 M 1 0	MRP 13965.53 - 505 z k r s 0 M 1 449 4 M 1 60 0 M 1 60
5	LC 3607.68	PTO, $\tau\geq0$ 6756.57 - 106 z k r s 0 M 1 309 0 M 1 40 0 M 1 1	IC 2098.27	IC 2347.94	Kanban 7857.71	PTO, $\tau\geq0$ 13804.64 - 128 z k r s 0 M 1 565 0 M 1 0 0 M 1 2
6	BSS 3659.71	IC 6973.92	CONWIP 2098.27	CONWIP 2347.94	BSS 8346.63	IC 14513.30
7	Kanban 3775.63	CONWIP 7014.13	BSS 2189.91	BSS 2870.98	PTO, $\tau\geq0$ 9901.80 - 56 z k r s 0 M 1 339 0 M 1 0 0 M 1 0	CONWIP 14908.24
8	CONWIP 4222.90	BSS 7580.77	LC 2783.64	LC 2992.80	CONWIP 9473.37	BSS 15522.30
9	PTO 7725.34	PTO 14715.94	PTO 4717.33	PTO 6339.24	PTO 15900.28	PTO 26161.46

* by using other policy best solution

Table c2-7 **Model 2A** case: ρ_2 =0.7, ρ_3 =0.7 ($1/\mu_2$ = 42 min., $1/\mu_3$ = 42 min.)
DCI [single cost; from arrival of ORD-tag]

Best Solution	exponential service time		uniform service time		exponential + bulk service time demand	
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7
1	PAC 4351.06 - 184 z k r s 0 15 1 0 5 16 1 0 5 1 1 0	PAC * 7430.23 - 23 z k r s 4 1 1 0 6 1 1 0 5 1 1 0	PAC 1623.74 - 233 z k r s 0 5 1 0 3 5 1 0 1 1 1 0	PAC 2095.42 - 89 z k r s 2 2 1 0 1 1 1 0 1 1 1 0	PAC * 8809.83 - 165 z k r s 0 1 1 0 11 12 1 0 6 1 1 0	PAC * 14856.25 - 85 z k r s 8 1 1 0 10 1 1 0 9 1 1 0
2	Kanban 4543.98 - 51 z k r s 1 1 1 0 6 6 1 0 1 1 1 0	LC 7430.23 - 43 z k r s 4 1 1 0 6 1 1 0 5 1 1 0	Kanban 1742.74 - 19 z k r s 1 1 1 0 2 2 1 0 1 1 1 0	Kanban 2095.42 - 33 z k r s 2 2 1 0 1 1 1 0 1 1 1 0	BSS 10339.26 - 94 z k r s 1 M 1 0 9 M 2 0 6 M 1 0	LC 14856.25 - 53 z k r s 8 1 1 0 10 1 1 0 9 1 1 0
3	IC 4800.41 - 45 z k r s 1 1 1 0 3 4 1 0 4 8 1 0	Kanban 7884.03 - 52 z k r s 3 3 1 0 6 6 1 0 1 1 1 0	MRP 1829.29 - 141 z k r s 0 M 1 0 3 M 1 0 1 M 1 0	MRP 2398.42 - 45 z k r s 2 M 1 0 2 M 1 0 0 M 1 0	LC 10401.33 - 43 z k r s 4 1 1 0 2 1 1 0 12 1 1 0	Kanban 15771.36 - 59 z k r s 9 9 1 0 3 3 1 0 7 7 1 0
4	LC 4928.11 - 50 z k r s 2 1 1 0 5 1 1 0 5 1 1 0	IC 8162.18 - 31 z k r s 5 5 1 0 2 7 1 0 4 11 1 0	BSS 1948.28 - 29 z k r s 1 M 1 0 2 M 1 0 1 M 1 0	BSS 2463.92 - 33 z k r s 3 M 2 0 1 M 1 0 1 M 1 0	MRP 10568.50 - 213 z k r s 2 M 2 0 7 M 1 6 8 M 1 0	IC 16630.96 - 41 z k r s 8 8 1 0 6 14 1 0 5 19 1 0
5	MRP 5118.53 - 100 z k r s 2 M 2 0 4 M 1 31 4 M 1 21	MRP 8263.80 - 117 z k r s 5 M 1 0 2 M 1 0 4 M 2 0	IC 2247.21 - 18 z k r s 3 3 1 0 0 3 1 0 0 3 1 0	IC 2483.79 - 27 z k r s 4 4 1 0 0 4 1 0 0 4 1 0	IC 10806.33 - 48 z k r s 3 3 1 0 6 9 1 0 8 17 1 0	MRP 16672.09 - 168 z k r s 12 M 3 0 0 M 3 31 6 M 1 0
6	BSS 5501.71 - 67 z k r s 2 M 2 0 5 M 4 0 1 M 1 0	BSS 8537.28 - 55 z k r s 5 M 1 0 3 M 1 0 3 M 1 0	CONWIP 2247.21 - 10 z k r s 3 3 1 0 0 3 1 0 0 3 1 0	CONWIP 2483.79 - 15 z k r s 4 4 1 0 0 4 1 0 0 4 1 0	Kanban 11390.21 - 59 z k r s 5 5 1 0 5 5 2 0 7 7 1 0	BSS 16714.25 - 70 z k r s 9 M 1 0 5 M 2 0 3 M 2 0
7	CONWIP 6283.48 - 12 z k r s 7 7 1 0 0 7 1 0 0 7 1 0	CONWIP 8779.64 - 20 z k r s 10 10 1 0 0 10 1 0 0 10 1 0	LC 2622.75 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	LC 2833.00 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	CONWIP 13451.74 - 15 z k r s 11 11 1 0 0 11 1 0 0 11 1 0	CONWIP 17694.63 - 14 z k r s 13 13 1 0 0 13 1 0 0 13 1 0
8	PTO, $\tau \geq 0$ 13478.88 - 46 z k r s 0 M 1 0 0 M 1 79 0 M 1 61	PTO, $\tau \geq 0$ 20302.18 - 17 z k r s 0 M 1 0 0 M 1 0 0 M 1 10	PTO, $\tau \geq 0$ 6129.65 - 7 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	PTO, $\tau \geq 0$ 7735.75 - 7 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	PTO, $\tau \geq 0$ 24581.98 - 31 z k r s 0 M 1 0 0 M 1 0 0 M 1 29	PTO, $\tau \geq 0$ 35158.61 - 7 z k r s 0 M 1 0 0 M 1 0 0 M 1 0
9	PTO 13958.96 - 1	PTO 20539.25 - 1	PTO 6129.65 - 1	PTO 7735.75 - 1	PTO 25074.59 - 1	PTO 35158.61 - 1

* by using other policy best solution

Table c2-8 **Model 2A** case: ρ_2 =0.7, ρ_3 =0.7 ($1/\mu_2$ = 42 min., $1/\mu_3$= 42 min.)
DCII [double cost; ORD-REQ]

Best Solution	exponential service time		uniform service time		exponential + bulk service time demand	
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7
1	PAC 4084.60 - 271 z k r s 0 15 1 59 3 12 1 43 4 1 1 0	PAC * 7429.91 - 66 z k r s 4 1 1 3 6 1 1 0 5 1 1 0	PAC * 1605.18 - 116 z k r s 0 3 1 146 0 10 1 0 0 10 1 0	PAC * 1642.26 - 115 z k r s 0 3 1 184 0 10 1 0 0 10 1 0	PAC 8130.92 - 317 z k r s 0 25 1 185 10 17 1 161 0 1 1 0	PAC 14184.52 - 434 z k r s 0 5 1 456 6 5 1 19 4 1 1 0
2	Kanban 4543.98	LC 7430.23	MRP 1714.98 - 137 z k r s 0 M 1 146 0 M 1 0 0 M 1 0	MRP * 1813.40 - 22 z k r s 0 M 1 183 0 M 1 0 0 M 1 0	BSS 10339.26	LC 14856.25
3	IC 4800.41	Kanban 7884.03	PTO, τ≥0 1714.98 - 52 z k r s 0 M 1 146 0 M 1 0 0 M 1 0	PTO, τ≥0 1813.40 - 61 z k r s 0 M 1 183 0 M 1 0 0 M 1 0	LC 10401.33	MRP 15323.26 - 662 z k r s 0 M 2 551 4 M 3 30 4 M 1 0
4	MRP * 4913.44 - 22 z k r s 0 M 1 412 0 M 1 400 0 M 1 101	MRP 8115.25 - 309 z k r s 3 M 1 71 3 M 1 10 4 M 2 0	Kanban 1742.74	Kanban 2095.42	MRP 10749.30 - 198 z k r s 2 M 3 22 8 M 1 0 10 M 2 21	PTO, τ≥0 15344.10 - 155 z k r s 0 M 1 821 0 M 1 359 0 M 1 130
5	PTO, τ≥0 4913.44 - 98 z k r s 0 M 1 412 0 M 1 400 0 M 1 101	IC 8162.18	BSS 1948.28	BSS 2463.929	IC 10806.33	Kanban 15771.36
6	LC 4928.11	PTO, τ≥0 8365.04 - 118 z k r s 0 M 1 466 0 M 1 0 0 M 1 58	IC 2247.21	IC 2483.79	Kanban 11390.21	IC 16630.96
7	BSS 5501.71	BSS 8537.28	CONWIP 2247.21	CONWIP 2483.79	PTO, τ≥0 12158.46 - 128 z k r s 0 M 1 738 0 M 1 0 0 M 1 369	BSS 16714.25
8	CONWIP 6283.48	CONWIP 8779.64	LC 2622.75	LC 2833.00	CONWIP 13451.74	CONWIP 17694.63
9	PTO 13958.96	PTO 20539.25	PTO 6129.65	PTO 7735.75	PTO 25074.59	PTO 35158.61

* by using other policy best solution

Table c2-9 **Model 2B** case: ρ_2 =0.1, ρ_3 =0.1 ($1/\mu_2$ = 6 min., $1/\mu_3$= 6 min.)
DCI [single cost; from arrival of ORD-tag]

Best Solution	exponential service time		uniform service time		exponential + bulk service time demand	
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7
1	PAC 97.83 - 69 z k r s 0 2 1 0 0 1 1 0 0 1 1 0	PAC 1128.55 - 133 z k r s 0 2 1 0 0 2 1 7 0 1 1 1	PAC 81.42 - 107 z k r s 0 4 1 0 0 2 1 0 0 1 1 0	PAC 753.72 - 79 z k r s 0 2 1 0 0 1 1 0 0 1 1 0	PAC 186.09 - 68 z k r s 0 2 1 0 0 1 1 0 0 1 1 0	PAC 2133.50 - 142 z k r s 1 3 1 0 0 3 2 0 0 2 2 3
2	MRP 109.31 - 19 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	IC 1187.21 - 27 z k r s 1 1 1 0 1 2 1 0 0 2 1 0	MRP 82.96 - 19 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	IC 839.29 - 27 z k r s 1 1 1 0 0 1 1 0 1 2 1 0	MRP 252.92 - 19 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	Kanban 2162.09 - 7 z k r s 1 1 1 0 1 1 1 0 1 1 1 0
3	PTO, $\tau \geq 0$ 109.31 - 7 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	Kanban 1424.43 - 7 z k r s 1 1 1 0 1 1 1 0 1 1 1 0	PTO, $\tau \geq 0$ 82.96 - 7 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	Kanban 1199.99 - 7 z k r s 1 1 1 0 1 1 1 0 1 1 1 0	PTO, $\tau \geq 0$ 252.92 - 7 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	IC 2297.44 - 27 z k r s 2 2 1 0 0 2 1 0 0 2 1 0
4	PTO 109.31 - 1	CONWIP 1486.38 - 9 z k r s 2 2 1 0 0 2 1 0 0 2 1 0	PTO 82.96 - 1	CONWIP 1245.59 - 9 z k r s 2 2 1 0 0 2 1 0 0 2 1 0	PTO 252.92 - 1	CONWIP 2297.44 - 15 z k r s 2 2 1 0 0 2 1 0 0 2 1 0
5	IC 746.94 - 0 z k r s 1 1 1 0 0 1 1 0 0 1 1 0	MRP 2220.25 - 50 z k r s 0 M 1 0 0 M 1 0 0 M 1 3	IC 741.61 - 7 z k r s 1 1 1 0 0 1 1 0 0 1 1 0	MRP 1245.96 - 19 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	IC 864.17 - 7 z k r s 1 1 1 0 0 1 1 0 0 1 1 0	LC 3154.55 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0
6	CONWIP 746.94 - 3 z k r s 1 1 1 0 0 1 1 0 0 1 1 0	PTO, $\tau \geq 0$ 2220.25 - 20 z k r s 0 M 1 0 0 M 1 0 0 M 1 3	CONWIP 741.61 - 3 z k r s 1 1 1 0 0 1 1 0 0 1 1 0	PTO, $\tau \geq 0$ 1245.96 - 7 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	CONWIP 864.17 - 3 z k r s 1 1 1 0 0 1 1 0 0 1 1 0	MRP 5358.33 - 165 z k r s 1 M 2 0 0 M 1 0 0 M 1 0
7	BSS 873.09 - 53 z k r s 1 M 4 0 1 M 3 0 1 M 2 0	PTO 2300.92 - 1	BSS 791.56 - 53 z k r s 1 M 4 0 1 M 3 0 1 M 2 0	PTO 1245.96 - 1	BSS 971.46 - 53 z k r s 1 M 4 0 1 M 3 0 1 M 2 0	PTO, $\tau \geq 0$ 5650.39 - 17 z k r s 0 M 1 0 0 M 1 10 0 M 1 0
8	Kanban 1626.59 - 7 z k r s 1 1 1 0 1 1 1 0 1 1 1 0	LC 2589.93 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	Kanban 1636.60 - 7 z k r s 1 1 1 0 1 1 1 0 1 1 1 0	LC 2497.57 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	Kanban 1634.48 - 7 z k r s 1 1 1 0 1 1 1 0 1 1 1 0	PTO 5965.26 - 1
9	LC 3437.99 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	BSS 2717.88 - 30 z k r s 1 M : 0 1 M 2 0 1 M 1 0	LC 3449.81 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	BSS 1766.62 - 30 z k r s 1 M 1 0 1 M 2 0 1 M 1 0	LC 3366.12 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	BSS 5995.70 - 39 z k r s 1 M 1 0 1 M 1 0 1 M 2 0

Table c2-10 **Model 2B** case: ρ_2 =0.1, ρ_3 =0.1 ($1/\mu_2$ = 6 min., $1/\mu_3$= 6 min.)
DCII [double cost; ORD-REQ]

Best Solution	exponential service time		uniform service time		exponential + bulk service time demand	
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7
1	PAC 68.65 - 76 z k r s 0 2 1 10 0 1 1 1 0 1 1 1	PAC 1003.92 - 229 z k r s 0 2 1 46 0 2 1 10 0 1 1 3	PAC 12.41 - 165 z k r s 0 3 1 17 0 5 1 0 0 1 1 4	PAC 532.03 - 190 z k r s 0 2 1 54 0 1 1 0 0 1 1 0	PAC 143.95 - 100 z k r s 0 2 1 15 0 1 1 5 0 1 1 0	PAC 1988.48 - 296 z k r s 0 3 1 116 0 11 2 3 0 11 1 6
2	MRP 79.99 - 108 z k r s 0 M 1 11 0 M 1 5 0 M 1 1	IC 1187.21	MRP 13.91 - 74 z k r s 0 M 1 17 0 M 1 0 0 M 1 0	IC 839.29	MRP 210.39 - 103 z k r s 0 M 1 15 0 M 1 7 0 M 1 1	Kanban 2162.09
3	PTO, τ≥0 79.99 - 48 z k r s 0 M 1 11 0 M 1 5 0 M 1 1	Kanban 1424.43	PTO, τ≥0 13.91 - 32 z k r s 0 M 1 17 0 M 1 0 0 M 1 0	MRP 1024.27 - 95 z k r s 0 M 1 54 0 M 1 0 0 M 1 0	PTO, τ≥0 210.39 - 49 z k r s 0 M 1 15 0 M 1 7 0 M 1 1	IC 2297.44
4	PTO 109.31	CONWIP 1486.38	PTO 82.96	PTO, τ≥0 1024.27 - 41 z k r s 0 M 1 54 0 M 1 0 0 M 1 0	PTO 252.92	CONWIP 2297.44
5	IC 746.94	MRP 2109.52 - 85 z k r s 0 M 1 43 0 M 1 0 0 M 1 3	IC 741.61	Kanban 1199.99	IC 864.17	LC 3154.55
6	CONWIP 746.94	PTO, τ≥0 2109.52 - 37 z k r s 0 M 1 43 0 M 1 0 0 M 1 3	CONWIP 741.61	CONWIP 1245.59	CONWIP 864.17	MRP 5190.92 - 113 z k r s 0 M 2 111 0 M 1 13 0 M 1 0
7	BSS 873.09	PTO 2300.92	BSS 791.56	PTO 1245.96	BSS 971.46	PTO, τ≥0 5612.37 - 69 z k r s 0 M 1 88 0 M 1 20 0 M 1 20
8	Kanban 1626.59	LC 2589.93	Kanban 1636.60	LC 2497.57	Kanban 1634.48	PTO 5965.26
9	LC 3437.99	BSS 2717.88	LC 3449.81	BSS 1766.62	LC 3366.12	BSS 5995.70

Table c2-11 **Model 2B** case: ρ_2 =0.1, ρ_3 =0.7 ($1/\mu_2$ = 6 min., $1/\mu_3$= 42 min.)
DCI [single cost; from arrival of ORD-tag]

Best Solution	exponential service time		uniform service time		exponential + bulk service time demand	
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7
1	**PAC** 721.90 - 132 z k r s 3 7 1 0 0 5 3 32 0 1 1 9	**PAC** 2070.49 - 147 z k r s 1 3 1 0 2 7 1 2 0 1 1 40	**PAC** 368.79 - 130 z k r s 0 3 1 0 1 3 2 0 1 1 1 0	**PAC** 578.71 - 75 z k r s 2 5 1 0 0 5 2 0 0 1 1 0	**PAC** 1422.62 - 212 z k r s 1 8 1 0 2 11 4 46 0 1 1 10	**PAC** 3357.64 - 160 z k r s 0 2 1 0 1 1 1 4 5 1 1 3
2	**MRP** 985.46 - 132 z k r s 0 M 1 0 1 M 1 1 0 M 1 24	**Kanban** 2109.88 - 32 z k r s 1 1 1 0 6 6 5 0 1 1 1 0	**MRP** 528.88 - 63 z k r s 0 M 1 0 0 M 1 0 1 M 1 0	**MRP** 721.92 - 41 z k r s 1 M 1 0 0 M 1 0 0 M 1 0	**Kanban** 1701.13 - 7 z k r s 1 1 1 0 1 1 1 0 1 1 1 0	**Kanban** 3469.01 - 70 z k r s 1 1 1 0 1 1 1 0 5 5 2 0
3	**PTO, τ≥0** 1078.29 - 54 z k r s 0 M 1 0 0 M 1 16 0 M 1 14	**LC** 2263.96 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	**PTO, τ≥0** 635.93 - 7 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	**IC** 733.56 - 27 z k r s 2 2 1 0 0 2 1 0 0 2 1 0	**IC** 1942.17 - 27 z k r s 2 2 1 0 0 2 1 0 0 2 1 0	**LC** 3587.05 - 27 z k r s 2 1 1 0 3 1 1 0 2 1 1 0
4	**IC** 1044.69 - 27 z k r s 1 1 1 0 1 2 1 0 0 2 1 0	**IC** 2648.06 - 32 z k r s 2 2 1 0 2 4 1 0 0 4 1 0	**PTO** 635.93- 1	**CONWIP** 733.56 - 15 z k r s 2 2 1 0 0 2 1 0 0 2 1 0	**CONWIP** 1942.17 - 15 z k r s 2 2 1 0 0 2 1 0 0 2 1 0	**IC** 3682.04 - 50 z k r s 1 1 1 0 1 2 1 0 5 7 1 0
5	**PTO** 1109.02- 1	**CONWIP** 2852.90 - 18 z k r s 4 4 1 0 0 4 1 0 0 4 1 0	**IC** 777.48 - 27 z k r s 1 1 1 0 0 1 1 0 1 2 1 0	**PTO, τ≥0** 814.61 - 7 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	**MRP** 2411.80 - 78 z k r s 0 M 1 0 2 M 1 0 0 M 1 2	**CONWIP** 4727.69 - 13 z k r s 5 5 1 0 0 5 1 0 0 5 1 0
6	**Kanban** 1123.78 - 7 z k r s 1 1 1 0 1 1 1 0 1 1 1 0	**MRP** 3067.28 - 118 z k r s 1 M 1 0 0 M 1 10 3 M 1 15	**BSS** 963.18- 30 z k r s 1 M 2 0 1 M 1 0 1 M 1 0	**PTO** 814.61- 1	**LC** 2541.80 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	**MRP** 5514.20 - 84 z k r s 2 M 1 0 0 M 1 60 0 M 1 21
7	**CONWIP** 1171.12 - 9 z k r s 2 2 1 0 0 2 1 0 0 2 1 0	**PTO, τ≥0** 3470.26 - 17 z k r s 0 M 1 0 0 M 1 0 0 M 1 20	**CONWIP** 1000.37 - 9 z k r s 2 2 1 0 0 2 1 0 0 2 1 0	**BSS** 915.45- 38 z k r s 1 M 1 0 1 M 3 0 1 M 1 0	**PTO, τ≥0** 2553.67 - 7 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	**PTO, τ≥0** 5653.25 - 36 z k r s 0 M 1 0 0 M 1 60 0 M 1 21
8	**BSS** 1367.18- 30 z k r s 1 M 2 0 1 M 1 0 1 M 1 0	**BSS** 3027.55- 65 z k r s 1 M 1 0 1 M 2 0 2 M 1 0	**Kanban** 1131.74 - 7 z k r s 1 1 1 0 1 1 1 0 1 1 1 0	**Kanban** 1035.32 - 7 z k r s 1 1 1 0 1 1 1 0 1 1 1 0	**PTO** 2553.67- 1	**BSS** 5850.37- 13 z k r s 1 M 1 0 1 M 1 0 1 M 1 0
9	**LC** 2654.23 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	**PTO** 3535.46- 1	**LC** 2559.85 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	**LC** 2336.42 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	**BSS** 2604.97- 48 z k r s 1 M 2 0 1 M 1 0 2 M 1 0	**PTO** 5966.44- 1

Table c2-12 **Model 2B** case: ρ_2 =0.1, ρ_3 =0.7 ($1/\mu_2$ = 6 min., $1/\mu_3$= 42 min.)
DCII [double cost; ORD-REQ]

Best Solution	exponential service time		uniform service time		exponential + bulk	service time demand
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7
1	PAC 483.16 - 204 z k r s 0 11 1 87 0 7 1 83 0 1 1 72	PAC 1978.86 - 185 z k r s 0 3 1 56 2 7 1 2 0 1 1 40	PAC 335.94 - 257 z k r s 0 5 1 52 2 7 3 46 0 1 1 0	PAC 355.22 - 279 z k r s 0 5 1 131 1 5 3 42 0 1 1 0	PAC 1002.41 - 381 z k r s 0 12 1 146 0 13 1 130 0 1 1 131	PAC 3130.44 - 274 z k r s 0 3 1 109 0 2 1 10 5 1 1 20
2	MRP * 880.88 - 25 z k r s 0 M 1 125 0 M 1 0 0 M 1 118	Kanban 2109.88	MRP 414.25 - 95 z k r s 0 M 1 54 0 M 1 0 0 M 1 0	MRP 443.36 - 117 z k r s 0 M 1 90 0 M 1 0 0 M 1 0	Kanban 1701.13	Kanban 3469.01
3	PTO, $\tau \geq 0$ 880.88 - 82 z k r s 0 M 1 125 0 M 1 0 0 M 1 118	LC 2263.96	PTO, $\tau \geq 0$ 414.25 - 41 z k r s 0 M 1 54 0 M 1 0 0 M 1 0	PTO, $\tau \geq 0$ 443.36 - 51 z k r s 0 M 1 90 0 M 1 0 0 M 1 0	IC 1942.17	LC 3587.05
4	IC 1044.69	IC 2648.06	PTO 635.93	IC 733.56	CONWIP 1942.17	IC 3682.04
5	PTO 1109.02	MRP 2771.44 - 349 z k r s 0 M 1 91 0 M 1 20 1 M 1 34	IC 777.48	CONWIP 733.56	MRP 2185.75 - 230 z k r s 0 M 1 72 2 M 1 63 0 M 3 41	CONWIP 4727.69
6	Kanban 1123.78	CONWIP 2852.90	BSS 963.18	PTO 814.61	PTO, $\tau \geq 0$ 2249.19 - 78 z k r s 0 M 1 132 0 M 1 125 0 M 1 58	MRP 5148.30 - 132 z k r s 0 M 1 181 0 M 1 60 0 M 1 21
7	CONWIP 1171.12	PTO, $\tau \geq 0$ 2932.25 - 69 z k r s 0 M 1 156 0 M 1 0 0 M 1 100	CONWIP 1000.37	BSS 915.45	LC 2541.80	PTO, $\tau \geq 0$ 5148.30 - 63 z k r s 0 M 1 181 0 M 1 60 0 M 1 21
8	BSS 1367.18	BSS 3027.55	Kanban 1131.74	Kanban 1035.32	PTO 2553.67	BSS 5850.37
9	LC 2654.23	PTO 3535.46	LC 2559.85	LC 2336.42	BSS 2604.97	PTO 5966.44

* by using other policy best solution

Table c2-13 **Model 2B** case: ρ_2 =0.7, ρ_3 =0.1 ($1/\mu_2$ = 42 min., $1/\mu_3$= 6 min.)
DCI [single cost; from arrival of ORD-tag]

Best Solution	exponential service time		uniform service time		exponential + bulk service time demand	
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7
1	PAC 909.31 - 154 z k r s 1 5 1 0 0 1 1 32 0 1 1 0	PAC 2512.56 - 101 z k r s 1 3 1 0 3 1 1 8 0 1 1 1	PAC 531.89 - 103 z k r s 0 1 1 0 1 1 1 0 0 1 1 0	PAC 665.49 - 74 z k r s 1 3 1 0 0 1 1 0 0 1 1 0	PAC 1761.57 - 182 z k r s 2 5 1 0 2 3 2 18 0 1 1 1	PAC 4158.55 - 97 z k r s 1 3 1 0 2 3 1 10 0 1 1 1
2	IC 1165.21 - 27 z k r s 1 1 1 0 1 2 1 0 0 2 1 0	Kanban 2690.84 - 22 z k r s 3 3 1 0 1 1 1 0 1 1 1 0	MRP 786.56 - 41 z k r s 0 M 1 0 1 M 1 0 0 M 1 0	IC 742.97 - 27 z k r s 2 2 1 0 0 2 1 0 0 2 1 0	Kanban 1847.99 - 7 z k r s 1 1 1 0 1 1 1 0 1 1 1 0	LC 4215.00 - 30 z k r s 2 1 1 0 4 1 1 0 2 1 1 0
3	Kanban 1185.64 - 7 z k r s 1 1 1 0 1 1 1 0 1 1 1 0	LC 2702.24 - 7 z k r s 2 1 1 0 3 1 1 0 2 1 1 0	IC 839.29 - 27 z k r s 1 1 1 0 0 1 1 0 1 2 1 0	CONWIP 742.97 - 15 z k r s 2 2 1 0 0 2 1 0 0 2 1 0	IC 2109.55 - 27 z k r s 2 2 1 0 0 2 1 0 0 2 1 0	Kanban 4489.39 - 22 z k r s 3 3 1 0 1 1 1 0 1 1 1 0
4	CONWIP 1304.04 - 9 z k r s 2 2 1 0 0 2 1 0 0 2 1 0	IC 2806.43 - 32 z k r s 2 2 1 0 2 4 1 0 0 4 1 0	PTO, $\tau\geq0$ 839.63 - 7 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	Kanban 886.29 - 7 z k r s 1 1 1 0 1 1 1 0 1 1 1 0	CONWIP 2109.55 - 15 z k r s 2 2 1 0 0 2 1 0 0 2 1 0	IC 4739.25 - 21 z k r s 3 3 1 0 2 5 1 0 0 5 1 0
5	MRP 1515.09 - 63 z k r s 0 M 1 0 1 M 1 0 0 M 1 0	CONWIP 3150.50 - 18 z k r s 4 4 1 0 0 4 1 0 0 4 1 0	PTO 839.63 - 1	MRP 922.28 - 41 z k r s 1 M 1 0 0 M 1 0 0 M 1 0	LC 2630.09 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	CONWIP 5233.08 - 18 z k r s 4 4 1 0 0 4 1 0 0 4 1 0
6	PTO, $\tau\geq0$ 1551.55 - 18 z k r s 0 M 1 0 0 M 1 20 0 M 1 0	MRP 3384.87 - 133 z k r s 1 M 1 0 0 M 1 14 0 M 1 2	Kanban 1048.46 - 7 z k r s 1 1 1 0 1 1 1 0 1 1 1 0	PTO, $\tau\geq0$ 1014.63 - 7 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	MRP 3316.29 - 83 z k r s 0 M 1 0 2 M 4 0 0 M 1 0	MRP 6627.31 - 75 z k r s 1 M 1 0 0 M 1 0 0 M 1 2
7	PTO 1586.14 - 1	BSS 3672.01 - 30 z k r s 1 M 1 0 1 M 1 0 1 M 2 0	CONWIP 1082.14 - 9 z k r s 2 2 1 0 0 2 1 0 0 2 1 0	PTO 1014.63 - 1	PTO, $\tau\geq0$ 3635.42 - 7 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	PTO, $\tau\geq0$ 6725.03 - 21 z k r s 0 M 1 0 0 M 1 0 0 M 1 2
8	BSS 1978.63 - 13 z k r s 1 M 1 0 1 M 1 0 1 M 1 0	PTO, $\tau\geq0$ 3696.77 - 29 z k r s 0 M 1 0 0 M 1 11 0 M 1 1	BSS 1248.56 - 30 z k r s 1 M 2 0 1 M 1 0 1 M 1 0	BSS 1307.54 - 30 z k r s 1 M 1 0 1 M 2 0 1 M 1 0	PTO 3635.42 - 1	PTO 6977.95 - 1
9	LC 2402.34 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	PTO 3808.89 - 1	LC 2633.81 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	LC 2175.37 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	BSS 3770.68 - 13 z k r s 1 M 1 0 1 M 1 0 1 M 1 0	BSS 7240.39 - 63 z k r s 2 M 3 0 1 M 1 0 1 M 2 0

Table c2-14 **Model 2B** case: ρ_2 =0.7, ρ_3 =0.1 ($1/\mu_2$ = 42 min., $1/\mu_3$= 6 min.)
DCII [double cost; ORD-REQ]

Best Solution	exponential service time		uniform service time		exponential + bulk service time demand	
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7
1	**PAC** 755.39 - 186 z k r s 0 15 1 95 0 1 1 90 0 1 1 0	**PAC** 2508.82 - 216 z k r s 1 3 1 11 3 1 1 8 0 1 1 1	**PAC** 394.64 - 314 z k r s 0 8 1 60 0 2 1 54 0 1 1 0	**PAC** 507.75 - 280 z k r s 0 3 1 48 1 1 1 5 0 1 1 0	**PAC** 1550.69 - 242 z k r s 0 3 1 61 2 3 2 42 0 9 1 0	**PAC** 3931.99 - 264 z k r s 0 3 1 130 2 3 1 10 0 1 1 1
2	**IC** 1165.21	**Kanban** 2690.84	**MRP** 617.96 - 95 z k r s 0 M 1 54 0 M 1 0 0 M 1 0	**MRP** 643.46 - 117 z k r s 0 M 1 90 0 M 1 0 0 M 1 0	**Kanban** 1847.99	**LC** 4215.00
3	**Kanban** 1185.64	**LC** 2702.24	**PTO, $\tau \geq 0$** 617.96 - 41 z k r s 0 M 1 54 0 M 1 0 0 M 1 0	**PTO, $\tau \geq 0$** 643.46 - 51 z k r s 0 M 1 90 0 M 1 0 0 M 1 0	**IC** 2109.55	**Kanban** 4489.39
4	**CONWIP** 1304.04	**IC** 2806.43	**IC** 839.29	**IC** 742.97	**CONWIP** 2109.55	**IC** 4739.25
5	**MRP *** 1383.23 - 25 z k r s 0 M 1 82 0 M 1 78 0 M 1 0	**CONWIP** 3150.50	**PTO** 839.63	**CONWIP** 742.97	**LC** 2630.09	**CONWIP** 5233.08
6	**PTO, $\tau \geq 0$** 1383.23 - 46 z k r s 0 M 1 82 0 M 1 78 0 M 1 0	**MRP** 3280.29 - 365 z k r s 0 M 1 74 1 M 1 14 0 M 1 2	**Kanban** 1048.46	**Kanban** 886.29	**MRP** 3187.35 - 158 z k r s 0 M 1 67 2 M 4 10 0 M 1 0	**MRP** 6291.81 - 180 z k r s 0 M 1 170 0 M 1 0 0 M 1 2
7	**PTO** 1586.14	**PTO, $\tau \geq 0$** 3376.39 - 78 z k r s 0 M 1 120 0 M 1 11 0 M 1 1	**CONWIP** 1082.14	**PTO** 1014.63	**PTO, $\tau \geq 0$** 3463.37 - 55 z k r s 0 M 1 73 0 M 1 0 0 M 1 0	**PTO, $\tau \geq 0$** 6291.81 - 82 z k r s 0 M 1 170 0 M 1 0 0 M 1 2
8	**BSS** 1978.63	**BSS** 3672.01	**BSS** 1248.56	**BSS** 1307.54	**PTO** 3635.42	**PTO** 6977.95
9	**LC** 2402.34	**PTO** 3808.89	**LC** 2933.81	**LC** 2175.37	**BSS** 3770.68	**BSS** 7240.39

* by using other policy best solution

Table c2-15 **Model 2B** case: ρ_2 =0.7, ρ_3 =0.7 ($1/\mu_2$ = 42 min., $1/\mu_3$= 42 min.)
DCI [single cost; from arrival of ORD-tag]

Best Solution	exponential service time		uniform service time		exponential + bulk service time demand	
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7
1	**PAC** 1601.88 - 156 z k r s 1 3 1 0 0 3 1 16 2 1 1 0	**PAC *** 2983.27 - 65 z k r s 1 1 1 0 4 3 1 0 3 1 1 0	**PAC** 599.80 - 307 z k r s 2 5 1 0 3 5 4 87 0 1 1 0	**PAC** 655.30 - 187 z k r s 2 7 1 0 3 7 3 43 0 1 1 0	**PAC** 2877.76 - 122 z k r s 0 3 1 0 0 7 1 0 4 1 1 1	**PAC** 5051.78 - 177 z k r s 1 3 1 0 3 3 1 1 4 1 1 1
2	**Kanban** 2039.52 - 62 z k r s 1 1 1 0 6 6 5 0 1 1 1 0	**LC** 3049.72 - 45 z k r s 2 1 1 0 4 1 1 0 3 1 1 0	**MRP** 654.71 - 63 z k r s 0 M 1 0 1 M 1 0 0 M 1 0	**Kanban** 728.25 - 7 z k r s 1 1 1 0 1 1 1 0 1 1 1 0	**LC** 3328.42 - 23 z k r s 2 1 1 0 2 1 1 0 6 1 1 0	**LC** 5101.52 - 45 z k r s 2 1 1 0 6 1 1 0 5 1 1 0
3	**LC** 2160.21 - 27 z k r s 2 1 1 0 2 1 1 0 3 1 1 0	**Kanban** 3249.59 - 78 z k r s 1 1 1 0 6 6 2 0 1 1 1 0	**IC** 732.65 - 27 z k r s 2 2 1 0 0 2 1 0 0 2 1 0	**MRP** 789.17 - 65 z k r s 1 M 1 0 0 M 1 0 1 M 1 0	**Kanban** 3406.02 - 43 z k r s 1 1 1 0 1 1 1 0 7 7 3 0	**IC** 5479.52 - 45 z k r s 1 1 1 0 6 7 1 0 1 8 1 0
4	**MRP** 2196.20 - 167 z k r s 0 M 1 0 1 M 1 32 1 M 1 6	**MRP** 3980.60 - 127 z k r s 3 M 1 0 0 M 1 35 0 M 1 34	**CONWIP** 732.65 - 15 z k r s 2 2 1 0 0 2 1 0 0 2 1 0	**BSS** 927.14 - 30 z k r s 1 M 2 0 1 M 1 0 1 M 1 0	**IC** 3687.20 - 43 z k r s 1 1 1 0 4 5 1 0 1 6 1 0	**Kanban** 5618.06 - 73 z k r s 3 3 1 0 7 7 3 0 1 1 1 0
5	**BSS** 2265.23 - 30 z k r s 1 M 1 0 1 M 1 0 1 M 2 0	**IC** 4041.52 - 24 z k r s 3 3 1 0 2 5 1 0 2 7 1 0	**PTO, $\tau \geq 0$** 803.96 - 7 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	**PTO, $\tau \geq 0$** 973.03 - 7 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	**MRP** 4169.66 - 129 z k r s 0 M 1 0 2 M 1 0 0 M 1 40	**CONWIP** 6844.71 - 17 z k r s 8 8 1 0 0 8 1 0 0 8 1 0
6	**IC** 2365.24 - 32 z k r s 2 2 1 0 2 4 1 0 0 4 1 0	**BSS** 4420.67 - 33 z k r s 1 M 2 0 3 M 1 0 1 M 1 0	**PTO** 803.96 - 1	**PTO** 973.03 - 1	**BSS** 4373.35 - 13 z k r s 1 M 1 0 1 M 1 0 1 M 1 0	**MRP** 6947.08 - 126 z k r s 3 M 3 0 2 M 2 40 0 M 1 19
7	**CONWIP** 2434.52 - 10 z k r s 4 4 1 0 0 4 1 0 0 4 1 0	**CONWIP** 4465.85 - 12 z k r s 7 7 1 0 0 7 1 0 0 7 1 0	**BSS** 844.36 - 30 z k r s 1 M 2 0 1 M 1 0 1 M 1 0	**IC** 1010.77 - 18 z k r s 3 3 1 0 0 3 1 0 0 3 1 0	**CONWIP** 4467.04 - 18 z k r s 6 6 1 0 0 6 1 0 0 6 1 0	**BSS** 7009.18 - 29 z k r s 2 M 1 0 1 M 1 0 1 M 1 0
8	**PTO, $\tau \geq 0$** 2571.43 - 45 z k r s 0 M 1 0 0 M 1 60 0 M 1 4	**PTO, $\tau \geq 0$** 4560.15 - 26 z k r s 0 M 1 0 0 M 1 0 0 M 1 5	**Kanban** 887.00 - 7 z k r s 1 1 1 0 1 1 1 0 1 1 1 0	**CONWIP** 1010.77 - 10 z k r s 3 3 1 0 0 3 1 0 0 3 1 0	**PTO, $\tau \geq 0$** 4744.55 - 31 z k r s 0 M 1 0 0 M 1 0 0 M 1 29	**PTO, $\tau \geq 0$** 7745.89 - 36 z k r s 0 M 1 0 0 M 1 0 0 M 1 58
9	**PTO** 2617.91 - 1	**PTO** 4621.58 - 1	**LC** 2472.91 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	**LC** 2015.56 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	**PTO** 4861.77 - 1	**PTO** 8105.99 - 1

* by using other policy best solution

Table c2-16 **Model 2B** case: ρ_2 =0.7, ρ_3 =0.7 ($1/\mu_2$ = 42 min., $1/\mu_3$= 42 min.)
DCII [double cost; ORD-REQ]

Best Solution	exponential service time		uniform service time		exponential + bulk service time demand	
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7
1	**PAC** 1471.11 - 254 z k r s 0 3 1 71 0 3 1 56 2 1 1 0	**PAC** * 2983.08 - 66 z k r s 1 1 1 3 4 3 1 0 3 1 1 0	**PAC** 410.67 - 184 z k r s 4 11 1 89 0 7 2 50 0 1 1 0	**PAC** 403.52 - 316 z k r s 6 9 1 125 0 3 2 70 0 1 1 0	**PAC** 2583.09 - 330 z k r s 0 3 1 107 0 13 1 61 3 1 1 1	**PAC** 4860.35 - 361 z k r s 0 3 1 112 3 3 1 1 4 1 1 1
2	**MRP** * 1964.52 - 25 z k r s 0 M 1 148 0 M 1 140 0 M 1 59	**LC** 3049.72	**MRP** 432.83 - 117 z k r s 0 M 1 90 0 M 1 0 0 M 1 0	**MRP** * 452.35 - 25 z k r s 0 M 1 126 0 M 1 0 0 M 1 0	**LC** 3328.42	**LC** 5101.52
3	**PTO, $\tau \geq 0$** 1964.52 - 120 z k r s 0 M 1 148 0 M 1 140 0 M 1 59	**Kanban** 3249.59	**PTO, $\tau \geq 0$** 432.83 - 51 z k r s 0 M 1 90 0 M 1 0 0 M 1 0	**PTO, $\tau \geq 0$** 452.35 - 43 z k r s 0 M 1 126 0 M 1 0 0 M 1 0	**Kanban** 3406.02	**IC** 5479.52
4	**Kanban** 2039.52	**MRP** * 3744.06 - 25 z k r s 0 M 1 216 0 M 1 109 0 M 1 42	**IC** 732.65	**Kanban** 728.25	**IC** 3687.20	**Kanban** 5618.06
5	**LC** 2160.21	**PTO, $\tau \geq 0$** 3744.06 - 121 z k r s 0 M 1 216 0 M 1 109 0 M 1 42	**CONWIP** 732.65	**BSS** 927.14	**MRP** 4090.65 - 281 z k r s 0 M 1 78 2 M 1 53 0 M 1 59	**MRP** 6586.45 - 191 z k r s 0 M 3 297 2 M 2 40 0 M 1 19
6	**BSS** 2265.23	**IC** 4041.52	**PTO** 803.96	**PTO** 973.03	**PTO, $\tau \geq 0$** 4271.78 - 90 z k r s 0 M 1 216 0 M 1 0 0 M 1 120	**CONWIP** 6844.71
7	**IC** 2365.24	**BSS** 4420.67	**BSS** 844.36	**IC** 1010.77	**BSS** 4373.35	**BSS** 7009.18
8	**CONWIP** 2434.52	**CONWIP** 4465.85	**Kanban** 887.00	**CONWIP** 1010.77	**CONWIP** 4467.04	**PTO, $\tau \geq 0$** 7094.44 - 83 z k r s 0 M 1 380 0 M 1 20 0 M 1 160
9	**PTO** 2617.91	**PTO** 4621.58	**LC** 2472.91	**LC** 2015.56	**PTO** 4861.77	**PTO** 8105.99

* by using other policy best solution

Appendix C3

OPTIMIZATION RESULTS FOR MODEL 3

Table c3-1 **Model 3A** case: ρ_2 =0.1, ρ_3 =0.1 ($1/\mu_2 = 6$ min., $1/\mu_3 = 6$ min.)
DCI [single cost; from arrival of ORD-tag]

Best Solution	exponential service time		uniform service time		exponential + bulk service time demand	
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7
1	**PAC** 1085.54 - 74 z k r s 0 3 1 0 0 1 1 1 0 1 1 1	**PAC** 4866.10 - 148 z k r s 4 2 1 0 0 3 1 1 0 1 1 1	**PAC** 987.44 - 73 z k r s 0 2 1 0 0 1 1 11 0 1 1 0	**PAC** 2697.78 - 104 z k r s 2 2 1 0 0 1 1 12 0 1 1 0	**PAC** 1865.35 - 98 z k r s 0 6 1 0 0 1 1 2 0 1 1 2	**PAC** * 10022.84 - 132 z k r s 6 3 1 0 0 1 1 4 0 1 1 4
2	**IC** 1106.47 - 7 z k r s 1 1 1 0 0 1 1 0 0 1 1 0	**IC** 5203.43 - 18 z k r s 3 3 1 0 0 3 1 0 0 3 1 0	**MRP** 1006.68 - 63 z k r s 0 M 1 0 0 M 1 11 0 M 1 0	**IC** 2742.10 - 27 z k r s 2 2 1 0 0 2 1 0 0 2 1 0	**MRP** 2155.32 - 55 z k r s 0 M 1 0 0 M 1 2 0 M 1 2	**LC** 10166.20 - 32 z k r s 7 1 1 0 2 1 1 0 2 1 1 0
3	**CONWIP** 1106.47 - 3 z k r s 1 1 1 0 0 1 1 0 0 1 1 0	**CONWIP** 5203.43 - 10 z k r s 3 3 1 0 0 3 1 0 0 3 1 0	**PTO, $\tau\geq 0$** 1006.68 - 27 z k r s 0 M 1 0 0 M 1 11 0 M 1 0	**CONWIP** 2742.10 - 15 z k r s 2 2 1 0 0 2 1 0 0 2 1 0	**PTO, $\tau\geq 0$** 2155.32 - 25 z k r s 0 M 1 0 0 M 1 2 0 M 1 2	**IC** 11534.50 - 27 z k r s 4 4 1 0 0 4 1 0 0 4 1 0
4	**MRP** 1114.16 - 43 z k r s 0 M 1 0 0 M 1 1 0 M 1 1	**LC** 5462.51 - 27 z k r s 3 1 1 0 2 1 1 0 2 1 1 0	**PTO** 1040.48 - 1	**Kanban** 3186.22 - 7 z k r s 1 1 1 0 1 1 1 0 1 1 1 0	**PTO** 2156.04 - 1	**CONWIP** 11534.50 - 15 z k r s 4 4 1 0 0 4 1 0 0 4 1 0
5	**PTO, $\tau\geq 0$** 1114.16 - 19 z k r s 0 M 1 0 0 M 1 1 0 M 1 1	**Kanban** 6004.71 - 17 z k r s 2 2 1 0 1 1 1 0 1 1 1 0	**IC** 1054.24 - 7 z k r s 1 1 1 0 0 1 1 0 0 1 1 0	**MRP** 3668.58 - 68 z k r s 2 M 1 0 0 M 1 11 0 M 1 0	**IC** 2269.42 - 7 z k r s 1 1 1 0 0 1 1 0 0 1 1 0	**Kanban** 11854.75 - 33 z k r s 2 2 1 0 1 1 1 0 1 1 1 0
6	**PTO** 1114.51 - 1	**MRP** 6290.71 - 94 z k r s 3 M 1 0 0 M 1 11 0 M 1 11	**CONWIP** 1054.24 - 3 z k r s 1 1 1 0 0 1 1 0 0 1 1 0	**LC** 3808.78 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	**CONWIP** 2269.42 - 3 z k r s 1 1 1 0 0 1 1 0 0 1 1 0	**MRP** 16186.58 - 52 z k r s 6 M 1 0 0 M 1 0 0 M 1 0
7	**BSS** 1736.19 - 30 z k r s 1 M 2 0 1 M 1 0 1 M 1 0	**BSS** 6594.93 - 32 z k r s 3 M 1 0 1 M 2 0 1 M 1 0	**BSS** 1660.36 - 32 z k r s 1 M 2 0 1 M 2 0 1 M 1 0	**BSS** 4246.20 - 53 z k r s 2 M 1 0 1 M 2 0 1 M 1 0	**Kanban** 2542.41 - 7 z k r s 1 1 1 0 1 1 1 0 1 1 1 0	**BSS** 16397.71 - 51 z k r s 6 M 1 0 1 M 1 0 1 M 1 0
8	**Kanban** 1912.63 - 7 z k r s 1 1 1 0 1 1 1 0 1 1 1 0	**PTO, $\tau\geq 0$** 9289.61 - 31 z k r s 0 M 1 0 0 M 1 11 0 M 1 11	**Kanban** 1894.56 - 7 z k r s 1 1 1 0 1 1 1 0 1 1 1 0	**PTO, $\tau\geq 0$** 6042.38 - 27 z k r s 0 M 1 0 0 M 1 11 0 M 1 0	**BSS** 2654.09 - 30 z k r s 1 M 2 0 1 M 1 0 1 M 1 0	**PTO, $\tau\geq 0$** 20881.30 - 21 z k r s 0 M 1 0 0 M 1 10 0 M 1 10
9	**LC** 3920.07 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	**PTO** 9518.92 - 1	**LC** 3931.76 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	**PTO** 6075.86 - 1	**LC** 4059.81 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	**PTO** 20947.04 - 1

* by using other policy best solution

Table c3-2 **Model 3A** case: ρ_2 =0.1, ρ_3 =0.1 ($1/\mu_2$ = 6 min., $1/\mu_3$ = 6 min.)
DCII [double cost; ORD-REQ]

Best Solution	exponential service time		uniform service time		exponential + bulk service time demand	
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7
1	PAC 495.13 - 132 z k r s 0 3 1 24 0 1 1 9 0 1 1 9	PAC 4762.01 - 309 z k r s 2 2 1 84 0 3 1 1 0 1 1 1	PAC 246.01 - 129 z k r s 0 2 1 19 0 1 1 12 0 1 1 0	PAC 2592.99 - 152 z k r s 2 2 1 25 0 1 1 12 0 1 1 0	PAC 937.67 - 126 z k r s 0 4 1 42 0 1 1 10 0 1 1 10	PAC 9727.60 - 575 z k r s 0 4 1 341 0 1 1 9 0 1 1 9
2	MRP 524.19 - 115 z k r s 0 M 1 24 0 M 1 9 0 M 1 9	IC 5203.43	MRP 265.48 - 112 z k r s 0 M 1 19 0 M 1 12 0 M 1 0	IC 2742.10	MRP 1228.65 - 135 z k r s 0 M 1 43 0 M 1 10 0 M 1 10	LC 10166.20
3	PTO, $\tau \geq 0$ 524.19 - 60 z k r s 0 M 1 24 0 M 1 9 0 M 1 9	CONWIP 5203.43	PTO, $\tau \geq 0$ 265.48 - 52 z k r s 0 M 1 19 0 M 1 12 0 M 1 0	CONWIP 2742.10	PTO, $\tau \geq 0$ 1228.65 - 52 z k r s 0 M 1 43 0 M 1 10 0 M 1 10	IC 11534.50
4	IC 1106.47	LC 5462.51	PTO 1040.48	MRP * 3141.68 - 23 z k r s 0 M 1 109 0 M 1 11 0 M 1 0	PTO 2156.04	CONWIP 11534.50
5	CONWIP 1106.47	Kanban 6004.71	IC 1054.24	PTO, $\tau \geq 0$ 3141.68 - 57 z k r s 0 M 1 109 0 M 1 11 0 M 1 0	IC 2269.42	Kanban 11854.75
6	PTO 1114.51	MRP * 6103.91 - 25 z k r s 0 M 1 166 0 M 1 11 0 M 1 11	CONWIP 1054.24	Kanban 3186.22	CONWIP 2269.42	MRP 15146.21 - 159 z k r s 0 M 1 325 0 M 1 0 0 M 1 0
7	BSS 1736.19	PTO, $\tau \geq 0$ 6103.91 - 104 z k r s 0 M 1 166 0 M 1 11 0 M 1 11	BSS 1660.36	LC 3808.78	Kanban 2542.41	PTO, $\tau \geq 0$ 15146.21 - 61 z k r s 0 M 1 325 0 M 1 0 0 M 1 0
8	Kanban 1912.63	BSS 6594.93	Kanban 1894.56	BSS 4246.20	BSS 2654.09	BSS 16397.71
9	LC 3920.07	PTO 9518.92	LC 3931.76	PTO 6075.86	LC 4059.81	PTO 20947.04

* by using other policy best solution

Table c3-3 **Model 3A** case: ρ_2 =0.1, ρ_3 =0.7 ($1/\mu_2$ = 6 min., $1/\mu_3$= 42 min.)
DCI [single cost; from arrival of ORD-tag]

Best Solution	exponential service time		uniform service time		exponential + bulk service time demand	
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7
1	**PAC** 5147.54 - 168 z k r s 2 3 1 0 3 1 1 31 4 1 1 31	**PAC** 8693.37 - 195 z k r s 5 6 1 0 2 1 1 25 3 1 1 25	**PAC** 3301.52 - 167 z k r s 2 2 1 0 1 1 1 20 2 1 1 20	**PAC** 3520.70 - 147 z k r s 3 2 1 0 0 3 1 20 1 1 1 20	**PAC** 12519.92 - 122 z k r s 8 7 1 0 2 1 1 70 2 1 1 70	**PAC** 17350.84 - 152 z k r s 12 7 1 0 2 1 1 40 3 1 1 40
2	**LC** 5681.62 - 30 z k r s 2 1 1 0 3 1 1 0 4 1 1 0	**LC** 8785.16 - 60 z k r s 5 1 1 0 5 1 1 0 6 1 1 0	**LC** 3669.26 - 17 z k r s 2 1 1 0 2 1 1 0 3 1 1 0	**LC** 3909.71 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	**LC** 12908.92 - 60 z k r s 8 1 1 0 2 1 1 0 3 1 1 0	**Kanban** 17509.04 - 43 z k r s 12 12 1 0 1 1 1 0 1 1 1 0
3	**Kanban** 6025.33 - 27 z k r s 4 4 1 0 1 1 1 0 1 1 1 0	**Kanban** 10131.88 - 31 z k r s 8 8 1 0 1 1 1 0 1 1 1 0	**Kanban** 3856.48 - 33 z k r s 2 2 1 0 1 1 1 0 1 1 1 0	**Kanban** 3988.44 - 22 z k r s 3 3 1 0 1 1 1 0 1 1 1 0	**Kanban** 14116.43 - 44 z k r s 10 10 1 0 1 1 1 0 1 1 1 0	**LC** 18599.01 - 60 z k r s 13 1 1 0 6 1 1 0 6 1 1 0
4	**IC** 6639.97 - 30 z k r s 3 3 1 0 0 3 1 0 1 4 1 0	**IC** 11373.73 - 41 z k r s 6 6 1 0 1 7 1 0 2 9 1 0	**IC** 3895.51 - 35 z k r s 1 1 1 0 1 2 1 0 2 3 1 0	**IC** 4066.29 - 41 z k r s 2 2 1 0 0 2 1 0 1 3 1 0	**IC** 16653.69 - 39 z k r s 6 6 1 0 0 6 1 0 1 7 1 0	**IC** 22347.88 - 48 z k r s 9 9 1 0 4 13 1 0 4 13 1 0
5	**CONWIP** 7336.47 - 10 z k r s 3 3 1 0 0 3 1 0 0 3 1 0	**CONWIP** 11591.02 - 41 z k r s 8 8 1 0 0 8 1 0 0 8 1 0	**CONWIP** 4637.57 - 15 z k r s 2 2 1 0 0 2 1 0 0 2 1 0	**CONWIP** 4731.47 - 10 z k r s 3 3 1 0 0 3 1 0 0 3 1 0	**CONWIP** 16848.55 - 12 z k r s 7 7 1 0 0 7 1 0 0 7 1 0	**CONWIP** 23843.94 - 14 z k r s 13 13 1 0 0 13 1 0 0 13 1 0
6	**MRP** 8519.13 - 110 z k r s 5 M 1 0 0 M 2 40 0 M 1 40	**MRP** 11990.24 - 191 z k r s 6 M 1 0 0 M 1 23 0 M 1 23	**BSS** 4996.58 - 50 z k r s 2 M 1 0 1 M 2 0 1 M 1 0	**BSS** 5121.13 - 32 z k r s 3 M 1 0 1 M 2 0 1 M 1 0	**BSS** 19058.50 - 112 z k r s 9 M 2 0 2 M 4 0 2 M 2 0	**MRP** 24437.88 - 119 z k r s 11 M 1 0 2 M 1 0 3 M 1 0
7	**BSS** 8609.14 - 50 z k r s 4 M 1 0 1 M 2 0 1 M 1 0	**BSS** 12424.61 - 77 z k r s 6 M 1 0 1 M 2 0 1 M 1 0	**MRP** 5049.11 - 110 z k r s 3 M 1 0 0 M 2 54 0 M 1 0	**MRP** 5914.54 - 89 z k r s 4 M 1 0 2 M 4 59 0 M 1 0	**MRP** 20104.47 - 118 z k r s 6 M 1 0 6 M 3 40 8 M 6 40	**BSS** 25037.91 - 44 z k r s 13 M 1 0 1 M 1 0 1 M 1 0
8	**PTO, τ≥0** 13353.92 - 37 z k r s 0 M 1 0 0 M 1 7 0 M 1 7	**PTO, τ≥0** 20739.39 - 135 z k r s 0 M 1 0 0 M 1 40 0 M 1 40	**PTO, τ≥0** 8792.50 - 41 z k r s 0 M 1 0 0 M 1 47 0 M 1 0	**PTO, τ≥0** 10368.33 - 41 z k r s 0 M 1 0 0 M 1 47 0 M 1 0	**PTO, τ≥0** 29092.45 - 19 z k r s 0 M 1 0 0 M 1 3 0 M 1 0	**PTO, τ≥0** 42966.63 - 25 z k r s 0 M 1 0 0 M 1 5 0 M 1 0
9	**PTO** 13356.13 - 1	**PTO** 21280.38 - 1	**PTO** 8889.20 - 1	**PTO** 10465.04 - 1	**PTO** 29092.91 - 1	**PTO** 42966.63 - 1

Table c3-4 **Model 3A** case: ρ_2 =0.1, ρ_3 =0.7 ($1/\mu_2$ = 6 min., $1/\mu_3$= 42 min.)
DCII [double cost; ORD-REQ]

Best Solution	exponential service time		uniform service time		exponential + bulk service time demand	
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7
1	PAC 4822.50 - 409 z k r s 0 9 1 51 3 1 1 32 4 1 1 32	PAC * 8688.09 - 159 z k r s 5 6 1 10 2 1 1 25 3 1 1 25	PAC 3159.01 - 148 z k r s 1 1 1 36 1 1 1 20 2 1 1 20	PAC 3415.90 - 389 z k r s 2 2 1 66 0 3 1 10 1 1 1 10	PAC 11948.64 - 375 z k r s 0 7 1 428 2 1 1 70 2 1 1 70	PAC 16827.08 - 710 z k r s 0 7 1 616 2 1 1 40 3 1 1 40
2	LC 5681.62	LC 8785.16	LC 3669.26	LC 3909.71	LC 12908.92	Kanban 17509.04
3	Kanban 6025.33	Kanban 10131.88	Kanban 3856.48	Kanban 3988.44	Kanban 14116.43	LC 18599.01
4	MRP 6264.88 - 201 z k r s 2 M 1 61 0 M 1 9 0 M 1 9	IC 11373.73	IC 3895.51	IC 4066.29	IC 16653.69	IC 22347.88
5	IC 6639.97	CONWIP 11591.02	CONWIP 4637.57	CONWIP 4731.47	CONWIP 16848.55	MRP 23579.34 - 557 z k r s 0 M 1 594 2 M 1 0 3 M 1 0
6	CONWIP 7336.47	MRP 12176.28 - 169 z k r s 8 M 3 22 0 M 1 50 0 M 1 50	MRP 4882.80 - 157 z k r s 2 M 1 76 0 M 2 54 0 M 1 0	MRP 4963.40 - 197 z k r s 2 M 1 112 0 M 2 54 0 M 1 0	BSS 19058.50	CONWIP 23843.94
7	BSS 8609.14	BSS 12424.61	PTO, $\tau \geq 0$ 4938.44 - 72 z k r s 0 M 1 168 0 M 1 51 0 M 1 0	PTO, $\tau \geq 0$ 5017.78 - 84 z k r s 0 M 1 204 0 M 1 49 0 M 1 0	MRP * 19610.66 - 626 z k r s 3 M 4 291 2 M 2 30 4 M 4 30	BSS 25037.91
8	PTO, $\tau \geq 0$ 8651.45 - 83 z k r s 0 M 1 280 0 M 1 183 0 M 1 183	PTO, $\tau \geq 0$ 12579.06 - 70 z k r s 0 M 1 391 0 M 1 14 0 M 1 14	BSS 4996.58	BSS 5121.13	PTO, $\tau \geq 0$ 19610.66 - 75 z k r s 0 M 1 660 0 M 1 10 0 M 1 10	PTO, $\tau \geq 0$ 27194.27 - 95 z k r s 0 M 1 783 0 M 1 8 0 M 1 0
9	PTO 13356.13	PTO 21280.38	PTO 8889.20	PTO 10465.04	PTO 29092.91	PTO 42966.63

* by using other policy best solution

Table c3-5 **Model 3A** case: ρ_2 =0.7, ρ_3 =0.1 ($1/\mu_2$ = 42 min., $1/\mu_3$= 6 min.)
DCI [single cost; from arrival of ORD-tag]

Best Solution	exponential service time		uniform service time		exponential + bulk service time demand	
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7
1	**PAC** 5323.34 - 102 z k r s 5 2 1 0 0 1 1 31 0 1 1 31	**PAC** * 9113.16 - 186 z k r s 8 10 1 0 0 1 1 160 0 1 1 160	**PAC** * 3406.16 - 26 z k r s 2 2 1 0 1 1 1 0 1 1 1 0	**PAC** * 3504.41 - 26 z k r s 3 1 1 0 1 1 1 5 1 1 1 5	**PAC** 12419.01 - 137 z k r s 8 5 1 0 2 1 1 100 2 1 1 100	**PAC** 17514.36 - 138 z k r s 11 9 1 0 2 1 1 0 3 1 1 0
2	**LC** 5373.34 - 17 z k r s 3 1 1 0 2 1 1 0 2 1 1 0	**LC** 9284.76 - 59 z k r s 5 1 1 0 5 1 1 0 6 1 1 0	**Kanban** 3406.16 - 33 z k r s 2 2 1 0 1 1 1 0 1 1 1 0	**Kanban** 3538.12 - 22 z k r s 3 3 1 0 1 1 1 0 1 1 1 0	**LC** 13099.73 - 22 z k r s 8 1 1 0 2 1 1 0 2 1 1 0	**Kanban** 17884.00 - 32 z k r s 8 8 1 0 1 1 1 0 1 1 1 0
3	**Kanban** 5502.72 - 27 z k r s 4 4 1 0 1 1 1 0 1 1 1 0	**Kanban** 9694.62 - 31 z k r s 8 8 1 0 1 1 1 0 1 1 1 0	**LC** 3485.13 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	**LC** 3748.18 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	**Kanban** 13465.46 - 44 z k r s 10 10 1 0 1 1 1 0 1 1 1 0	**LC** 18783.75 - 105 z k r s 12 1 1 0 7 1 1 0 8 1 1 0
4	**IC** 7467.41 - 27 z k r s 4 4 1 0 0 4 1 0 0 4 1 0	**IC** 11542.54 - 54 z k r s 6 6 1 0 1 7 1 0 2 9 1 0	**IC** 4322.31 - 27 z k r s 2 2 1 0 0 2 1 0 0 2 1 0	**IC** 4416.21 - 18 z k r s 3 3 1 0 0 3 1 0 0 3 1 0	**IC** 17132.47 - 60 z k r s 5 5 1 0 2 7 1 0 2 7 1 0	**IC** 21315.15 - 49 z k r s 9 9 1 0 2 11 1 0 3 12 1 0
5	**CONWIP** 7467.41 - 3 z k r s 1 1 1 0 0 1 1 0 0 1 1 0	**CONWIP** 11711.94 - 10 z k r s 3 3 1 0 0 3 1 0 0 3 1 0	**CONWIP** 4322.31 - 15 z k r s 2 2 1 0 0 2 1 0 0 2 1 0	**CONWIP** 4416.21 - 10 z k r s 3 3 1 0 0 3 1 0 0 3 1 0	**CONWIP** 18773.52 - 17 z k r s 8 8 1 0 0 8 1 0 0 8 1 0	**BSS** 25356.46 - 44 z k r s 13 M 1 0 1 M 1 0 1 M 1 0
6	**BSS** 8326.59 - 72 z k r s 4 M 2 0 1 M 2 0 1 M 1 0	**BSS** 12777.45 - 39 z k r s 7 M 2 0 1 M 1 0 1 M 1 0	**BSS** 5053.40 - 45 z k r s 2 M 1 0 1 M 1 0 1 M 1 0	**MRP** 5184.63 - 53 z k r s 4 M 1 0 0 M 1 0 0 M 1 0	**MRP** 19223.54 - 150 z k r s 9 M 1 0 2 M 2 10 2 M 2 10	**CONWIP** 25537.19 - 22 z k r s 16 16 1 0 0 16 1 0 0 16 1 0
7	**MRP** 8374.54 - 95 z k r s 5 M 1 0 0 M 1 30 0 M 1 30	**MRP** 13121.58 - 65 z k r s 6 M 1 0 2 M 2 0 2 M 2 0	**MRP** 5056.65 - 74 z k r s 3 M 1 0 0 M 1 0 0 M 1 0	**BSS** 5185.36 - 30 z k r s 3 M 1 0 1 M 1 0 1 M 1 0	**BSS** 19315.03 - 118 z k r s 9 M 3 0 2 M 1 0 2 M 2 0	**MRP** 25568.88 - 213 z k r s 13 M 2 0 0 M 2 30 0 M 1 30
8	**PTO, $\tau \geq 0$** 13747.98 - 17 z k r s 0 M 1 0 0 M 1 1 0 M 1 0	**PTO, $\tau \geq 0$** 22765.18 - 41 z k r s 0 M 1 0 0 M 1 42 0 M 1 42	**PTO, $\tau \geq 0$** 8573.93 - 7 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	**PTO, $\tau \geq 0$** 10149.78 - 7 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	**PTO, $\tau \geq 0$** 31938.89 - 22 z k r s 0 M 1 0 0 M 1 2 0 M 1 0	**PTO, $\tau \geq 0$** 44838.82 - 29 z k r s 0 M 1 0 0 M 1 9 0 M 1 0
9	**PTO** 13748.00 - 1	**PTO** 23356.17 - 1	**PTO** 8573.93 - 1	**PTO** 10149.78 - 1	**PTO** 31938.91 - 1	**PTO** 44862.30 - 1

* by using other policy best solution

Table c3-6 **Model 3A DCII** case: ρ_2 =0.7, ρ_3 =0.1 ($1/\mu_2$ = 42 min., $1/\mu_3$= 6 min.) [double cost; ORD-REQ]

Best Solution	exponential service time		uniform service time		exponential + bulk service time demand	
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7
1	PAC 5310.20 - 170 z k r s 4 2 1 42 0 1 1 31 0 1 1 31	PAC * 9113.06 - 77 z k r s 8 10 1 2 5 1 1 160 6 1 1 160	PAC 3265.25 - 147 z k r s 2 7 1 72 0 1 1 0 0 1 1 0	PAC 3530.60 - 135 z k r s 4 5 1 11 0 1 1 0 0 1 1 0	PAC 11796.73 - 289 z k r s 0 5 1 437 2 1 1 100 2 1 1 100	PAC 16747.57 - 346 z k r s 0 9 1 589 2 1 1 0 3 1 1 0
2	LC 5373.34	LC 9284.76	Kanban 3406.16	Kanban 3538.12	LC 13099.73	Kanban 17884.00
3	Kanban 5502.72	Kanban 9694.62	LC 3485.13	LC 3748.18	Kanban 13465.46	LC 18783.75
4	IC 7467.41	IC 11542.54	IC 4322.31	IC 4416.21	IC 17132.47	IC 21315.15
5	CONWIP 7467.41	CONWIP 11711.94	CONWIP 4322.31	CONWIP 4416.21	MRP 17274.18 - 345 z k r s 6 M 2 461 2 M 2 122 2 M 2 122	MRP 24506.16 - 890 z k r s 0 M 2 770 0 M 2 103 0 M 1 103
6	MRP * 8147.60 - 23 z k r s 0 M 1 254 0 M 1 183 0 M 1 183	MRP * 12699.17 - 23 z k r s 0 M 1 429 0 M 1 160 0 M 1 160	MRP * 4723.51 - 23 z k r s 0 M 1 168 0 M 1 0 0 M 1 0	MRP * 4803.01 - 23 z k r s 0 M 1 204 0 M 1 0 0 M 1 0	CONWIP 18773.52	BSS 25356.46
7	PTO, $\tau \geq 0$ 8147.60 - 88 z k r s 0 M 1 254 0 M 1 183 0 M 1 183	PTO, $\tau \geq 0$ 12699.17 - 108 z k r s 0 M 1 429 0 M 1 160 0 M 1 160	PTO, $\tau \geq 0$ 4723.51 - 62 z k r s 0 M 1 168 0 M 1 0 0 M 1 0	PTO, $\tau \geq 0$ 4803.01 - 57 z k r s 0 M 1 204 0 M 1 0 0 M 1 0	BSS 19315.03	CONWIP 25537.19
8	BSS 8326.59	BSS 12777.45	BSS 5053.40	BSS 5185.36	PTO, $\tau \geq 0$ 21654.06 - 88 z k r s 0 M 1 567 0 M 1 300 0 M 1 300	PTO, $\tau \geq 0$ 26388.71 - 177 z k r s 0 M 1 778 0 M 1 150 0 M 1 150
9	PTO 13748.00	PTO 23356.17	PTO 8573.93	PTO 10149.78	PTO 31938.91	PTO 44862.30

* by using other policy best solution

Table c3-7 **Model 3B DCI** case: ρ_2 =0.1, ρ_3 =0.1 ($1/\mu_2$ = 6 min., $1/\mu_3$= 6 min.) [single cost; from arrival of ORD-tag]

Best Solution	exponential service time		uniform service time		exponential + bulk service time demand	
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7
1	**PAC** 1241.05 - 140 z k r s 1 2 1 0 0 1 1 14 0 3 1 14	**PAC** 5230.25 - 97 z k r s 4 2 1 0 0 1 1 20 0 1 1 20	**PAC** 1143.15 - 131 z k r s 1 2 1 0 0 1 1 18 0 3 1 0	**PAC** 2715.33 - 118 z k r s 2 2 1 0 0 1 1 18 0 1 1 0	**PAC** 2687.66 - 124 z k r s 1 4 1 0 0 1 1 12 0 1 1 12	**PAC** * 9990.06 - 123 z k r s 7 1 1 0 1 1 1 0 2 1 1 0
2	**MRP** 1314.74 - 41 z k r s 1 M 1 0 0 M 1 0 0 M 1 0	**LC** 5263.71 - 27 z k r s 3 1 1 0 2 1 1 0 2 1 1 0	**MRP** 1205.91 - 58 z k r s 1 M 1 0 0 M 1 0 0 M 1 0	**IC** 2804.12 - 27 z k r s 2 2 1 0 0 2 1 0 0 2 1 0	**MRP** 2735.27 - 102 z k r s 1 M 2 0 0 M 2 0 0 M 1 4	**LC** 10534.22 - 31 z k r s 7 1 1 0 2 1 1 0 2 1 1 0
3	**IC** 1316.18 - 7 z k r s 1 1 1 0 0 1 1 0 0 1 1 0	**IC** 5762.50 - 718 z k r s 3 3 1 0 0 3 1 0 0 3 1 0	**IC** 1254.44 - 7 z k r s 1 1 1 0 0 1 1 0 0 1 1 0	**CONWIP** 2804.12 - 15 z k r s 2 2 1 0 0 2 1 0 0 2 1 0	**Kanban** 2929.37 - 7 z k r s 1 1 1 0 1 1 1 0 1 1 1 0	**IC** 11992.70 - 23 z k r s 5 5 1 0 0 5 1 0 0 5 1 0
4	**CONWIP** 1316.18 - 3 z k r s 1 1 1 0 0 1 1 0 0 1 1 0	**CONWIP** 5762.50 - 10 z k r s 3 3 1 0 0 3 1 0 0 3 1 0	**CONWIP** 1254.44 - 3 z k r s 1 1 1 0 0 1 1 0 0 1 1 0	**Kanban** 3124.91 - 7 z k r s 1 1 1 0 1 1 1 0 1 1 1 0	**BSS** 2966.61 - 33 z k r s 1 M 2 0 1 M 2 0 1 M 1 0	**CONWIP** 11992.70 - 13 z k r s 5 5 1 0 0 5 1 0 0 5 1 0
5	**PTO, τ≥0** 1682.72 - 21 z k r s 0 M 1 0 0 M 1 10 0 M 1 10	**Kanban** 5776.98 - 33 z k r s 2 2 1 0 1 1 1 0 1 1 1 0	**PTO, τ≥0** 1569.20 - 37 z k r s 0 M 1 0 0 M 1 18 0 M 1 0	**LC** 3747.47 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	**IC** 3025.95 - 17 z k r s 2 2 1 0 0 2 1 0 0 2 1 0	**Kanban** 12236.88 - 32 z k r s 4 4 1 0 1 1 1 0 1 1 1 0
6	**PTO** 1700.56 - 1	**MRP** 7002.71 - 84 z k r s 3 M 1 0 0 M 1 20 0 M 1 20	**PTO** 1631.89 - 1	**MRP** 3774.35 - 92 z k r s 2 M 1 0 0 M 1 18 0 M 1 0	**CONWIP** 3025.95 - 9 z k r s 2 2 1 0 0 2 1 0 0 2 1 0	**MRP** 15975.03 - 103 z k r s 9 M 4 0 0 M 1 20 0 M 1 20
7	**BSS** 1765.89 - 32 z k r s 1 M 2 0 1 M 2 0 1 M 1 0	**BSS** 7050.64 - 32 z k r s 3 M 1 0 1 M 2 0 1 M 1 0	**BSS** 1689.32 - 32 z k r s 1 M 2 0 1 M 2 0 1 M 1 0	**BSS** 4242.91 - 49 z k r s 2 M 1 0 1 M 2 0 1 M 1 0	**PTO, τ≥0** 3474.97 - 31 z k r s 0 M 1 0 0 M 1 11 0 M 1 11	**BSS** 16408.06 - 52 z k r s 6 M 1 0 1 M 1 0 1 M 1 0
8	**Kanban** 1886.16 - 7 z k r s 1 1 1 0 1 1 1 0 1 1 1 0	**PTO, τ≥0** 10303.10 - 7 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	**Kanban** 1863.84 - 7 z k r s 1 1 1 0 1 1 1 0 1 1 1 0	**PTO, τ≥0** 6373.68 - 40 z k r s 0 M 1 0 0 M 1 18 0 M 1 0	**PTO** 3483.81 - 1	**PTO, τ≥0** 23006.84 - 21 z k r s 0 M 1 0 0 M 1 10 0 M 1 10
9	**LC** 3826.71 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	**PTO** 10303.10 - 1	**LC** 3839.10 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	**PTO** 6436.13 - 1	**LC** 4088.60 - 7 z k r s 2 1 1 0 2 1 1 0 2 1 1 0	**PTO** 23016.44 - 1

* by using other policy best solution

Table c3-8 **Model 3B** case: ρ_2 =0.1, ρ_3 =0.1 ($1/\mu_2$ = 6 min., $1/\mu_3$= 6 min.)
DCII [double cost; ORD-REQ]

Best Solution	exponential service time		uniform service time		exponential + bulk service time demand	
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7
1	PAC 668.38 - 176 z k r s 0 7 1 35 0 1 1 20 0 1 1 20	PAC 5222.33 - 208 z k r s 4 2 1 11 0 1 1 20 0 1 1 20	PAC 439.04 - 164 z k r s 0 2 1 27 0 1 1 19 0 3 1 0	PAC 2544.49 - 124 z k r s 2 2 1 32 0 1 1 19 0 1 1 0	PAC 1448.28 - 184 z k r s 0 4 1 67 0 1 1 30 0 1 1 30	PAC 9832.43 - 354 z k r s 0 3 1 377 0 1 1 30 0 2 1 30
2	MRP 737.30 - 110 z k r s 0 M 1 35 0 M 1 20 0 M 1 20	LC 5263.71	MRP 523.07 - 148 z k r s 0 M 1 27 0 M 1 20 0 M 1 20	IC 2804.12	MRP * 1992.73 - 23 z k r s 0 M 1 66 0 M 1 20 0 M 1 20	LC 10534.22
3	PTO, $\tau \geq 0$ 737.30 - 56 z k r s 0 M 1 35 0 M 1 20 0 M 1 20	IC 5762.50	PTO, $\tau \geq 0$ 523.07 - 76 z k r s 0 M 1 27 0 M 1 20 0 M 1 20	CONWIP 2804.12	PTO, $\tau \geq 0$ 1992.73 - 53 z k r s 0 M 1 66 0 M 1 20 0 M 1 20	IC 11992.70
4	IC 1316.18	CONWIP 5762.50	IC 1254.44	Kanban 3124.91	Kanban 2929.37	CONWIP 11992.70
5	CONWIP 1316.18	Kanban 5776.98	CONWIP 1254.44	MRP * 3181.55 - 23 z k r s 0 M 1 116 0 M 1 18 0 M 1 0	BSS 2966.61	Kanban 12236.88
6	PTO 1700.56	MRP * 6743.84 - 23 z k r s 0 M 1 182 0 M 1 20 0 M 1 20	PTO 1631.89	PTO, $\tau \geq 0$ 3181.55 - 73 z k r s 0 M 1 116 0 M 1 18 0 M 1 0	IC 3025.95	MRP * 14533.84 - 23 z k r s 0 M 1 369 0 M 1 70 0 M 1 70
7	BSS 1765.89	PTO, $\tau \geq 0$ 6743.84 - 57 z k r s 0 M 1 182 0 M 1 20 0 M 1 20	BSS 1689.32	LC 3747.47	CONWIP 3025.95	PTO, $\tau \geq 0$ 14533.84 - 114 z k r s 0 M 1 369 0 M 1 70 0 M 1 70
8	Kanban 1886.16	BSS 7050.64	Kanban 1863.84	BSS 4242.91	PTO 3483.81	BSS 16408.06
9	LC 3826.71	PTO 10303.10	LC 3839.10	PTO 6436.13	LC 4088.60	PTO 23016.44

* by using other policy best solution

Table c3-9 **Model 3B** case: ρ_2 =0.1, ρ_3 =0.7 ($1/\mu_2$ = 6 min., $1/\mu_3$= 42 min.)
DCI [single cost; from arrival of ORD-tag]

Best Solution	exponential service time		uniform service time		exponential + bulk	service time demand
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7
1	**PAC** 6346.23 - 136 z k r s 4 3 1 0 2 3 3 31 2 1 1 31	**PAC** 11284.66 - 179 z k r s 7 7 1 0 0 3 1 32 4 1 1 32	**PAC *** 6021.22 - 206 z k r s 2 4 1 0 3 10 4 20 2 10 1 20	**PAC *** 4942.50 - 418 z k r s 5 4 1 0 1 10 4 110 0 1 1 110	**PAC** 17936.05 - 231 z k r s 14 6 1 0 4 1 1 70 10 7 6 70	**PAC** 21208.52 - 271 z k r s 12 12 1 0 4 9 4 55 7 1 1 55
2	**MRP** 9437.11 - 213 z k r s 6 M 2 0 1 M 4 75 0 M 2 75	**Kanban** 12274.70 - 48 z k r s 7 7 1 0 5 5 2 0 1 1 1 0	**BSS** 6356.92 - 92 z k r s 1 M 1 0 4 M 4 0 3 M 1 0	**BSS** 6922.20 - 82 z k r s 4 M 1 0 3 M 4 0 1 M 1 0	**MRP** 21543.14 - 231 z k r s 11 M 2 0 4 M 8 103 2 M 4 103	**Kanban** 26046.98 - 95 z k r s 11 11 1 0 5 5 3 0 1 1 1 0
3	**BSS** 10122.88 - 63 z k r s 6 M 1 0 1 M 4 0 1 M 3 0	**LC** 13522.63 - 74 z k r s 9 1 1 0 7 1 1 0 8 1 1 0	**MRP** 6656.68 - 102 z k r s 4 M 1 0 2 M 5 81 0 M 1 0	**MRP** 7596.04 - 58 z k r s 6 M 1 0 0 M 2 60 0 M 1 0	**BSS** 22403.87 - 59 z k r s 9 M 1 0 7 M 7 0 5 M 3 0	**LC** 27350.97 - 65 z k r s 18 1 1 0 12 1 1 0 12 1 1 0
4	**LC** 10999.98 - 40 z k r s 8 1 1 0 3 1 1 0 4 1 1 0	**MRP** 13934.77 - 177 z k r s 7 M 1 0 4 M 6 61 3 M 1 61	**LC** 8740.24 - 55 z k r s 5 1 1 0 3 1 1 0 4 1 1 0	**LC** 8798.98 - 42 z k r s 5 1 1 0 3 1 1 0 4 1 1 0	**LC** 26739.69 - 69 z k r s 24 1 1 0 3 1 1 0 3 1 1 0	**BSS** 27429.44 - 56 z k r s 13 M 1 0 5 M 6 0 3 M 1 0
5	**Kanban** 14444.24 - 33 z k r s 11 11 1 0 1 1 1 0 1 1 1 0	**BSS** 14158.79 - 63 z k r s 9 M 1 0 1 M 4 0 1 M 3 0	**Kanban** 9435.03 - 38 z k r s 8 8 1 0 1 1 1 0 1 1 1 0	**Kanban** 9476.67 - 44 z k r s 8 8 1 0 1 1 1 0 1 1 1 0	**Kanban** 30678.16 - 43 z k r s 27 27 1 0 1 1 1 0 1 1 1 0	**MRP** 27964.66 - 118 z k r s 13 M 1 0 5 M 4 1 3 M 1 1
6	**IC** 15112.63 - 53 z k r s 4 4 1 0 2 6 1 0 3 7 1 0	**IC** 17439.51 - 36 z k r s 9 9 1 0 2 11 1 0 2 11 1 0	**IC** 12228.67 - 55 z k r s 5 5 1 0 1 6 1 0 2 7 1 0	**IC** 12301.45 - 54 z k r s 5 5 1 0 1 6 1 0 2 7 1 0	**IC** 38795.48 - 66 z k r s 12 12 1 0 6 18 1 0 7 19 1 0	**IC** 38656.63 - 72 z k r s 20 20 1 0 0 20 1 0 0 20 1 0
7	**PTO, $\tau\geq 0$** 32952.70 - 31 z k r s 0 M 1 0 0 M 1 50 0 M 1 50	**CONWIP** 18210.89 - 15 z k r s 11 11 1 0 0 11 1 0 0 11 1 0	**CONWIP** 12943.20 - 17 z k r s 6 6 1 0 0 6 1 0 0 6 1 0	**CONWIP** 12978.27 - 12 z k r s 7 7 1 0 0 7 1 0 0 7 1 0	**CONWIP** 40507.69 - 21 z k r s 20 20 1 0 0 20 1 0 0 20 1 0	**CONWIP** 38656.63 - 21 z k r s 20 20 1 0 0 20 1 0 0 20 1 0
8	**CONWIP** 17686.63 - 15 z k r s 9 9 1 0 0 9 1 0 0 9 1 0	**PTO, $\tau\geq 0$** 35920.33 - 42 z k r s 0 M 1 0 0 M 1 19 0 M 1 19	**PTO, $\tau\geq 0$** 24269.40 - 49 z k r s 0 M 1 0 0 M 1 54 0 M 1 0	**PTO, $\tau\geq 0$** 25842.15 - 49 z k r s 0 M 1 0 0 M 1 54 0 M 1 0	**PTO, $\tau\geq 0$** 74701.12 - 31 z k r s 0 M 1 0 0 M 1 12 0 M 1 0	**PTO, $\tau\geq 0$** 73197.16 - 25 z k r s 0 M 1 0 0 M 1 30 0 M 1 30
9	**PTO** 33167.10 - 1	**PTO** 35940.31 - 1	**PTO** 24314.13 - 1	**PTO** 25886.88 - 1	**PTO** 74702.78 - 1	**PTO** 73212.23 - 1

* by using other policy best solution

Table c3-10 **Model 3B** case: ρ_2 =0.1, ρ_3 =0.7 ($1/\mu_2$ = 6 min., $1/\mu_3$= 42 min.)
DCII [double cost; ORD-REQ]

Best Solution	exponential service time		uniform service time		exponential + bulk service time demand	
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7
1	**PAC** 6346.23 - 155 z k r s 4 3 1 0 2 3 3 31 2 1 1 31	**PAC** 11298.03 - 376 z k r s 6 7 1 29 0 3 1 32 4 1 1 32	**PAC** * 6073.49 - 607 z k r s 3 4 1 25 2 10 4 71 2 10 2 71	**PAC** * 5489.20 - 272 z k r s 6 3 1 36 0 10 2 103 0 10 2 0	**PAC** 17259.41 - 1071 z k r s 0 6 1 709 4 1 1 70 10 7 6 70	**PAC** 20970.02 - 1184 z k r s 4 13 1 699 5 7 6 101 3 1 1 101
2	**MRP** 9434.65 - 393 z k r s 5 M 2 40 1 M 4 77 0 M 2 77	**Kanban** 12274.70	**BSS** 6356.92	**MRP** 6912.29 - 162 z k r s 4 M 1 45 2 M 4 70 0 M 1 0	**MRP** 21026.18 - 861 z k r s 1 M 2 553 2 M 6 85 2 M 2 85	**Kanban** 26046.98
3	**BSS** 10122.88	**MRP** 13328.02 - 247 z k r s 6 M 1 8 3 M 4 20 2 M 1 20	**MRP** 6717.65 - 155 z k r s 4 M 1 37 0 M 3 69 0 M 1 0	**BSS** 6922.20	**BSS** 22403.87	**MRP** 26490.77 - 822 z k r s 0 M 2 805 4 M 6 100 1 M 1 100
4	**LC** 10999.98	**LC** 13522.63	**LC** 8740.24	**LC** 8798.98	**LC** 26739.69	**LC** 27350.97
5	**Kanban** 14444.24	**BSS** 14158.79	**Kanban** 9435.03	**Kanban** 9476.67	**Kanban** 30678.16	**BSS** 27429.44
6	**IC** 15112.63	**IC** 17439.51	**IC** 12228.67	**IC** 12301.45	**IC** 38795.48	**IC** 38656.63
7	**PTO, $\tau \geq 0$** 17487.13 - 93 z k r s 0 M 1 502 0 M 1 340 0 M 1 340	**CONWIP** 18210.89	**CONWIP** 12943.20	**CONWIP** 12978.27	**CONWIP** 40507.69	**CONWIP** 38656.63
8	**CONWIP** 17686.63	**PTO, $\tau \geq 0$** 19971.53 - 113 z k r s 0 M 1 709 0 M 1 420 0 M 1 420	**PTO, $\tau \geq 0$** 15320.85 - 100 z k r s 0 M 1 458 0 M 1 73 0 M 1 0	**PTO, $\tau \geq 0$** 15396.33 - 95 z k r s 0 M 1 494 0 M 1 63 0 M 1 0	**PTO, $\tau \geq 0$** 47702.64 - 120 z k r s 0 M 1 1483 0 M 1 54 0 M 1 54	**PTO, $\tau \geq 0$** 42809.47 - 189 z k r s 0 M 1 1299 0 M 1 541 0 M 1 541
9	**PTO** 33167.10	**PTO** 35940.31	**PTO** 24314.13	**PTO** 25886.88	**PTO** 74702.78	**PTO** 73212.23

* by using other policy best solution

Table c3-11 **Model 3B** case: ρ_2 =0.7, ρ_3 =0.1 ($1/\mu_2$ = 42 min., $1/\mu_3$= 6 min.)
DCI [single cost; from arrival of ORD-tag]

Best Solution	exponential service time		uniform service time		exponential + bulk service time demand	
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7
1	**PAC** 8034.19 - 151 z k r s 7 3 1 0 0 7 3 61 0 1 1 61	**PAC** 10365.52 - 217 z k r s 6 5 1 0 4 1 1 60 3 5 5 60	**PAC *** 6110.73 - 93 z k r s 6 4 1 0 0 10 3 0 0 10 2 152	**PAC *** 5993.12 - 161 z k r s 6 4 1 0 0 10 3 0 0 10 1 139	**PAC** 18435.55 - 243 z k r s 17 9 1 0 0 5 5 131 4 1 1 131	**PAC** 20739.80 - 193 z k r s 14 9 1 0 6 1 1 42 4 7 4 42
2	**MRP** 9988.91 - 158 z k r s 7 M 4 0 2 M 4 81 0 M 1 81	**LC** 13452.07 - 59 z k r s 10 1 1 0 7 1 1 0 7 1 1 0	**MRP** 7721.14 - 156 z k r s 6 M 1 0 0 M 3 0 0 M 2 142	**MRP** 7965.70 - 96 z k r s 6 M 1 0 0 M 3 0 0 M 1 120	**MRP** 22034.12 - 121 z k r s 12 M 4 0 2 M 4 80 4 M 7 80	**LC** 26124.41 - 788 z k r s 13 1 1 0 16 1 1 0 17 1 1 0
3	**BSS** 10287.18 - 83 z k r s 6 M 1 0 1 M 3 0 1 M 2 0	**Kanban** 14063.89 - 32 z k r s 13 13 1 0 1 1 1 0 1 1 1 0	**BSS** 7769.15 - 45 z k r s 5 M 1 0 1 M 3 0 1 M 2 0	**BSS** 7984.00 - 42 z k r s 5 M 1 0 1 M 3 0 1 M 1 0	**BSS** 22920.52 - 114 z k r s 12 M 4 0 3 M 4 0 1 M 1 0	**Kanban** 27522.49 - 51 z k r s 17 17 1 0 1 1 1 0 5 5 1 0
4	**LC** 10968.54 - 40 z k r s 8 1 1 0 3 1 1 0 4 1 1 0	**BSS** 14670.64 - 69 z k r s 6 M 1 0 3 M 2 0 3 M 1 0	**LC** 8878.90 - 32 z k r s 7 1 1 0 2 1 1 0 2 1 1 0	**LC** 8915.13 - 31 z k r s 7 1 1 0 2 1 1 0 2 1 1 0	**Kanban** 27006.69 - 54 z k r s 25 25 1 0 1 1 1 0 1 1 1 0	**MRP** 28433.71 - 262 z k r s 18 M 1 0 2 M 8 240 2 M 2 240
5	**Kanban** 11624.13 - 44 z k r s 10 10 1 0 1 1 1 0 1 1 1 0	**MRP** 14953.64 - 98 z k r s 8 M 1 0 2 M 3 60 2 M 2 60	**Kanban** 8881.26 - 38 z k r s 8 8 1 0 1 1 1 0 1 1 1 0	**Kanban** 8922.90 - 44 z k r s 8 8 1 0 1 1 1 0 1 1 1 0	**LC** 27221.40 - 44 z k r s 26 1 1 0 2 1 1 0 2 1 1 0	**BSS** 28579.43 - 128 z k r s 14 M 4 0 5 M 8 0 4 M 1 0
6	**IC** 15996.26 - 761 z k r s 6 6 1 0 2 8 1 0 2 8 1 0	**IC** 18337.93 - 49 z k r s 10 10 1 0 4 14 1 0 5 15 1 0	**IC** 12628.24 - 31 z k r s 6 6 1 0 0 6 1 0 0 6 1 0	**IC** 12663.32 - 22 z k r s 7 7 1 0 0 7 1 0 0 7 1 0	**IC** 38544.11 - 83 z k r s 16 16 1 0 1 17 1 0 1 17 1 0	**IC** 41216.89 - 48 z k r s 21 21 1 0 0 21 1 0 0 21 1 0
7	**CONWIP** 21119.56 - 15 z k r s 9 9 1 0 0 9 1 0 0 9 1 0	**CONWIP** 18482.64 - 14 z k r s 13 13 1 0 0 13 1 0 0 13 1 0	**CONWIP** 12628.24 - 17 z k r s 6 6 1 0 0 6 1 0 0 6 1 0	**CONWIP** 12663.32 - 12 z k r s 7 7 1 0 0 7 1 0 0 7 1 0	**CONWIP** 47787.38 - 21 z k r s 22 22 1 0 0 22 1 0 0 22 1 0	**CONWIP** 41216.89 - 16 z k r s 21 21 1 0 0 21 1 0 0 21 1 0
8	**PTO, τ≥0** 37593.95 - 25 z k r s 0 M 1 0 0 M 1 10 0 M 1 10	**PTO, τ≥0** 36911.92 - 21 z k r s 0 M 1 0 0 M 1 10 0 M 1 10	**PTO, τ≥0** 23999.18 - 7 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	**PTO, τ≥0** 25571.93 - 7 z k r s 0 M 1 0 0 M 1 0 0 M 1 0	**PTO, τ≥0** 83929.34 - 25 z k r s 0 M 1 0 0 M 1 30 0 M 1 30	**PTO, τ≥0** 77584.83 - 41 z k r s 0 M 1 0 0 M 1 81 0 M 1 81
9	**PTO** 37594.25 - 1	**PTO** 36912.28 - 1	**PTO** 23999.18 - 1	**PTO** 25571.93 - 1	**PTO** 83937.40 - 1	**PTO** 78050.55 - 1

* by using other policy best solution

Table c3-12 **Model 3B** case: ρ_2 =0.7, ρ_3 =0.1 ($1/\mu_2$ = 42 min., $1/\mu_3$= 6 min.)
DCII [double cost; ORD-REQ]

Best Solution	exponential service time		uniform service time		exponential + bulk service time demand	
ρ_1	0.1	0.7	0.1	0.7	0.1	0.7
1	**PAC** 8127.55 - 219 z k r s 6 3 1 55 0 5 3 40 0 1 1 40	**PAC** 10362.86 - 288 z k r s 6 5 1 7 4 1 1 60 3 5 5 60	**PAC** * 6240.98 - 340 z k r s 4 5 1 141 0 10 4 0 0 10 3 196	**PAC** * 5926.08 - 225 z k r s 6 4 1 33 0 10 3 0 0 10 1 148	**PAC** 17897.53 - 930 z k r s 5 9 1 638 0 5 5 131 4 1 1 131	**PAC** 20419.95 - 997 z k r s 6 11 1 589 6 1 1 150 4 5 4 150
2	**MRP** 9946.50 - 231 z k r s 6 M 4 28 2 M 4 81 0 M 1 81	**LC** 13452.07	**MRP** 7616.12 - 167 z k r s 4 M 2 143 0 M 4 0 0 M 3 240	**MRP** 7920.54 - 194 z k r s 6 M 1 25 0 M 3 0 0 M 1 127	**MRP** 20371.21 - 692 z k r s 2 M 3 436 2 M 3 160 4 M 5 160	**LC** 26124.41
3	**BSS** 10287.18	**Kanban** 14063.89	**BSS** 7769.15	**BSS** 7984.00	**BSS** 22920.52	**MRP** 27440.84 - 949 z k r s 0 M 4 941 2 M 11 280 2 M 4 300
4	**LC** 10968.54	**BSS** 14670.64	**LC** 8878.90	**LC** 8915.13	**Kanban** 27006.69	**Kanban** 27522.49
5	**Kanban** 11624.13	**MRP** 14947.57 - 269 z k r s 8 M 2 91 0 M 4 130 0 M 1 130	**Kanban** 8881.26	**Kanban** 8922.90	**LC** 27221.40	**BSS** 28579.43
6	**IC** 15996.26	**IC** 18337.93	**IC** 12628.24	**IC** 12663.32	**IC** 38544.11	**IC** 41216.89
7	**CONWIP** 21119.56	**CONWIP** 18482.64	**CONWIP** 12628.24	**CONWIP** 12663.32	**PTO, $\tau \geq 0$** 41134.63 - 266 z k r s 0 M 1 1078 0 M 1 940 0 M 1 940	**CONWIP** 41216.89
8	**PTO, $\tau \geq 0$** 24938.16 - 126 z k r s 0 M 1 668 0 M 1 280 0 M 1 280	**PTO, $\tau \geq 0$** 21291.81 - 135 z k r s 0 M 1 702 0 M 1 210 0 M 1 210	**PTO, $\tau \geq 0$** 15056.74 - 66 z k r s 0 M 1 458 0 M 1 0 0 M 1 0	**PTO, $\tau \geq 0$** 15135.45 - 71 z k r s 0 M 1 494 0 M 1 0 0 M 1 0	**CONWIP** 47787.38	**PTO, $\tau \geq 0$** 49620.03 - 151 z k r s 0 M 1 1447 0 M 1 230 0 M 1 230
9	**PTO** 37594.25	**PTO** 36912.28	**PTO** 23999.18	**PTO** 25571.93	**PTO** 83937.40	**PTO** 78050.55

* by using other policy best solution

Appendix C4

OPTIMIZATION RESULTS FOR MODEL 4

Table c4-1 **Model 4A**

DCI [single cost; from arrival of ORD-tag]

Best Solu -tion	exponential service time no setup	exponential service time with setup	uniform service time no setup	uniform service time with setup	exponential + bulk service time demand no setup	exponential + bulk service time demand with setup	uniform + bulk service time demand no setup	uniform + bulk service time demand with setup
1	**PAC** 6248.88 - 448 z k r s 0 2 1 0 0 2 1 0 0 11 1 0 0 7 1 20 0 2 1 20 0 9 1 20 3 3 1 3	**PAC** 6551.64 - 610 z k r s 0 15 1 0 0 17 1 0 0 1 1 0 0 1 1 1 0 1 1 1 0 15 1 22 3 3 1 0	**PAC** 5244.95 - 387 z k r s 0 2 1 0 0 5 1 0 0 1 1 0 0 11 1 1 0 1 1 1 0 13 1 0 3 2 1 0	**PAC** 5545.96 - 328 z k r s 0 9 1 0 0 1 1 0 0 1 1 0 0 2 1 0 0 1 1 0 0 9 1 0 3 2 1 0	**PAC** 9522.24 - 436 z k r s 0 17 1 0 0 27 1 0 0 1 1 0 0 2 1 0 0 2 1 22 0 23 1 20 3 3 1 0	**PAC** 9794.36 - 602 z k r s 0 14 1 0 0 1 1 0 0 1 1 0 0 1 1 16 0 2 1 26 0 3 1 0 4 3 1 0	**PAC** 7996.11 - 366 z k r s 0 2 1 0 0 1 1 0 0 1 1 0 0 11 1 22 0 1 1 41 0 5 1 0 3 2 1 0	**PAC** 8121.55 - 581 z k r s 0 2 1 0 0 2 1 0 0 1 1 0 0 2 1 34 0 1 1 35 0 7 1 40 3 2 1 0
2	**MRP** 8130.20 - 251 z k r s 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 25 0 M 1 20 3 M 3 3	**MRP** 8657.25 - 196 z k r s 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 1 0 M 1 3 0 M 1 0 4 M 2 18	**MRP** 7698.78 - 116 z k r s 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 4 M 2 0	**MRP** 8084.90 - 116 z k r s 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 4 M 2 0	**MRP** 17199.17 - 303 z k r s 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 19 0 M 1 0 3 M 4 15	**MRP** 17310.56 - 243 z k r s 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 22 0 M 1 0 5 M 2 22	**MRP** 16580.97 - 235 z k r s 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 26 0 M 1 40 0 M 1 0 4 M 2 4	**MRP** 16963.48 - 199 z k r s 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 18 0 M 1 39 0 M 1 0 4 M 2 0
3	**PTO, τ≥0** 9162.75 - 15 z k r s 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0	**PTO, τ≥0** 9965.52 - 15 z k r s 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0	**PTO, τ≥0** 8598.26 - 15 z k r s 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0	**PTO, τ≥0** 9062.69 - 69 z k r s 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 1 0 M 1 1 0 M 1 0 0 M 1 0	**PTO, τ≥0** 18343.18 - 42 z k r s 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 2 0 M 1 0 0 M 1 0	**PTO, τ≥0** 18411.15 - 67 z k r s 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 26 0 M 1 0 0 M 1 0	**PTO, τ≥0** 17172.29 - 51 z k r s 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 21 0 M 1 0 0 M 1 0	**PTO, τ≥0** 17587.20 - 101 z k r s 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 14 0 M 1 35 0 M 1 0 0 M 1 0
4	**PTO** 9162.75 - 1	**PTO** 9965.52 - 1	**PTO** 8598.26 - 1	**PTO** 9070.59 - 1	**PTO** 18627.24 - 1	**PTO** 19058.04 - 1	**PTO** 17932.77 - 1	**PTO** 18300.52 - 1
5	**BSS** 16297.58 - 70 z k r s 1 M 2 0 1 M 2 0 1 M 2 0 1 M 2 0 1 M 1 0 1 M 1 0 1 M 1 0	**BSS** 16537.00 - 70 z k r s 1 M 2 0 1 M 2 0 1 M 2 0 1 M 2 0 1 M 1 0 1 M 1 0 1 M 1 0	**BSS** 15780.78 - 71 z k r s 1 M 2 0 1 M 2 0 1 M 2 0 1 M 2 0 1 M 1 0 1 M 1 0 1 M 1 0	**BSS** 15911.86 - 71 z k r s 1 M 2 0 1 M 2 0 1 M 2 0 1 M 2 0 1 M 1 0 1 M 1 0 1 M 1 0	**IC** 24151.10 - 51 z k r s 1 1 1 0 1 1 1 0 1 1 1 0 1 2 1 0 1 6 1 0 0 2 1 0 0 18 1 0	**Kanban** 24729.80 - 58 z k r s 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 3 3 1 0	**Kanban** 23353.19 - 53 z k r s 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 2 2 1 0	**Kanban** 23604.80 - 55 z k r s 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 2 2 1 0
6	**IC** 19367.25 - 33 z k r s 1 1 1 0 1 1 1 0 1 1 1 0 1 2 1 0 0 5 1 0 0 2 1 0 1 16 1 0	**IC** 19498.24 - 51 z k r s 1 1 1 0 1 1 1 0 1 1 1 0 1 2 1 0 0 5 1 0 0 2 1 0 1 16 1 0	**IC** 18937.40 - 33 z k r s 1 1 1 0 1 1 1 0 1 1 1 0 1 2 1 0 0 5 1 0 0 2 1 0 1 16 1 0	**IC** 19111.61 - 33 z k r s 1 1 1 0 1 1 1 0 1 1 1 0 1 2 1 0 0 5 1 0 0 2 1 0 1 16 1 0	**Kanban** 24423.73 - 78 z k r s 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 3 3 1 0	**BSS** 24780.57 - 138 z k r s 1 M 2 0 1 M 3 0 1 M 3 0 1 M 2 0 1 M 2 0 1 M 1 0 4 M 1 0	**IC** 23599.66 - 51 z k r s 1 1 1 0 1 1 1 0 1 1 1 0 1 2 1 0 0 5 1 0 0 2 1 0 3 18 1 0	**IC** 24020.77 - 51 z k r s 1 1 1 0 1 1 1 0 1 1 1 0 1 2 1 0 1 6 1 0 0 2 1 0 0 18 1 0
7	**CONWIP** 19516.47 - 9 z k r s 1 1 1 0 1 1 1 0 1 1 1 0 1 2 1 0 0 5 1 0 0 2 1 0 0 15 1 0	**CONWIP** 20038.72 - 9 z k r s 1 1 1 0 1 1 1 0 1 1 1 0 1 2 1 0 0 5 1 0 0 2 1 0 0 15 1 0	**CONWIP** 18997.24 - 9 z k r s 1 1 1 0 1 1 1 0 1 1 1 0 1 2 1 0 0 5 1 0 0 2 1 0 0 15 1 0	**CONWIP** 19371.16 - 9 z k r s 1 1 1 0 1 1 1 0 1 1 1 0 1 2 1 0 0 5 1 0 0 2 1 0 0 15 1 0	**BSS** 24656.81 - 98 z k r s 1 M 2 0 1 M 3 0 1 M 3 0 1 M 2 0 1 M 2 0 1 M 1 0 3 M 1 0	**IC** 25009.82 - 51 z k r s 2 2 1 0 1 1 1 0 1 1 1 0 1 2 1 0 0 6 1 0 0 2 1 0 0 18 1 0	**BSS** 24143.78 - 116 z k r s 1 M 2 0 1 M 3 0 1 M 3 0 1 M 2 0 1 M 2 0 1 M 1 0 3 M 1 0	**BSS** 24352.44 - 96 z k r s 1 M 2 0 1 M 3 0 1 M 3 0 1 M 2 0 1 M 2 0 1 M 1 0 3 M 1 0
8	**Kanban** 23907.48 - 83 z k r s 3 3 1 0 1 1 1 0 1 1 1 0 1 1 1 0 2 2 1 0 1 1 1 0 1 1 1 0	**Kanban** 24106.66 - 73 z k r s 3 3 1 0 1 1 1 0 1 1 1 0 1 1 1 0 2 2 1 0 1 1 1 0 1 1 1 0	**Kanban** 21752.77 - 57 z k r s 2 2 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0	**Kanban** 22832.37 - 38 z k r s 3 3 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0	**CONWIP** 24873.01 - 21 z k r s 2 2 1 0 1 1 1 0 1 1 1 0 1 2 1 0 0 6 1 0 0 2 1 0 0 18 1 0	**CONWIP** 25009.82 - 33 z k r s 2 2 1 0 1 1 1 0 1 1 1 0 1 2 1 0 0 6 1 0 0 2 1 0 0 18 1 0	**CONWIP** 24458.80 - 21 z k r s 2 2 1 0 1 1 1 0 1 1 1 0 1 3 1 0 0 6 1 0 0 2 1 0 0 18 1 0	**CONWIP** 24648.10 - 21 z k r s 2 2 1 0 1 1 1 0 1 1 1 0 1 3 1 0 0 6 1 0 0 2 1 0 0 18 1 0
9	**LC** 43310.17 - 34 z k r s 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 4 1 1 0 2 1 1 0 2 1 1 0	**LC** 45881.72 - 33 z k r s 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 6 1 1 0 2 1 1 0 2 1 1 0	**LC** 40975.25 - 15 z k r s 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0	**LC** 42295.15 - 15 z k r s 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0	**LC** 58988.26 - 80 z k r s 2 1 1 0 2 1 1 0 2 1 1 0 3 1 1 0 4 1 1 0 2 1 1 0 4 1 1 0	**LC** 66486.85 - 57 z k r s 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 10 1 1 0 2 1 1 0 6 1 1 0	**LC** 55955.90 - 100 z k r s 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 4 1 1 0 2 1 1 0 8 1 1 0	**LC** 60693.10 - 57 z k r s 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 27 1 1 0

Table c4-2 **Model 4A**
DCII [double cost; ORD-REQ]

Best Solu -tion	exponential service time		uniform service time		exponential + bulk service time demand		uniform + bulk service time demand	
	no setup	with setup	no setup	with setup	no setup	with setup	no setup	with setup
1	PAC 5012.81 - 672 z k r s 0 3 1 184 0 11 1 408 0 3 1 207 0 3 1 20 0 3 1 20 0 1 1 0 0 3 1 0	PAC 5224.04 - 811 z k r s 0 2 1 143 0 2 1 301 0 1 1 119 0 1 1 0 0 2 1 0 0 3 1 79 3 3 1 6	PAC 4003.59 - 631 z k r s 0 2 1 144 0 9 1 322 0 1 1 179 0 15 1 0 0 1 1 0 0 17 1 0 3 2 1 0	PAC 4257.44 - 953 z k r s 0 2 1 154 0 11 1 379 0 1 1 188 0 11 1 41 0 1 1 41 0 5 1 0 2 2 1 0	PAC 8364.27 - 604 z k r s 0 10 1 134 0 19 1 344 0 1 1 156 0 1 1 0 0 1 1 21 0 15 1 0 4 3 1 0	PAC 8344.88 - 656 z k r s 0 14 1 198 0 1 1 347 0 1 1 168 0 1 1 16 0 2 1 26 0 3 1 0 4 3 1 0	PAC 6627.09 - 1043 z k r s 0 2 1 188 0 2 1 404 0 2 1 215 0 11 1 26 0 1 1 40 0 5 1 0 2 2 1 0	PAC 6650.91 - 963 z k r s 0 2 1 200 0 2 1 374 0 1 1 125 0 2 1 21 0 1 1 39 0 7 1 60 3 2 1 0
2	MRP 7360.96 - 429 z k r s 0 M 1 181 0 M 1 377 0 M 1 236 0 M 1 0 0 M 1 0 0 M 1 141 0 M 1 0	MRP 7663.56 - 376 z k r s 0 M 1 155 0 M 1 246 0 M 1 118 0 M 1 0 0 M 1 60 0 M 1 0 4 M 2 22	MRP 6360.20 - 490 z k r s 0 M 1 121 0 M 1 273 0 M 1 102 0 M 1 0 0 M 1 0 0 M 1 0 4 M 2 0	MRP 6669.92 - 442 z k r s 0 M 1 131 0 M 1 275 0 M 1 103 0 M 1 0 0 M 1 0 0 M 1 0 4 M 2 0	MRP * 15661.73 - 447 z k r s 0 M 1 232 0 M 1 409 0 M 1 311 0 M 1 1 0 M 1 12 0 M 1 122 1 M 1 52	MRP 15883.20 - 799 z k r s 0 M 1 203 0 M 1 261 0 M 1 171 0 M 1 20 0 M 1 20 0 M 1 0 5 M 2 9	MRP * 14753.37 - 53 z k r s 0 M 1 207 0 M 1 463 0 M 1 347 0 M 1 34 0 M 1 39 0 M 1 0 0 M 1 0	MRP 15104.85 - 1228 z k r s 0 M 1 219 0 M 1 463 0 M 1 353 0 M 1 28 0 M 1 41 0 M 1 0 0 M 3 61
3	PTO, $\tau \geq 0$ 7360.96 - 202 z k r s 0 M 1 181 0 M 1 377 0 M 1 236 0 M 1 0 0 M 1 0 0 M 1 141 0 M 1 0	PTO, $\tau \geq 0$ 7750.98 - 259 z k r s 0 M 1 341 0 M 1 370 0 M 1 277 0 M 1 60 0 M 1 69 0 M 1 238 0 M 1 0	PTO, $\tau \geq 0$ 6587.83 - 195 z k r s 0 M 1 302 0 M 1 448 0 M 1 241 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0	PTO, $\tau \geq 0$ 7014.20 - 204 z k r s 0 M 1 219 0 M 1 374 0 M 1 252 0 M 1 57 0 M 1 1 0 M 1 0 0 M 1 0	PTO, $\tau \geq 0$ 16111.03 - 329 z k r s 0 M 1 258 0 M 1 482 0 M 1 354 0 M 1 1 0 M 1 2 0 M 1 127 0 M 1 43	PTO, $\tau \geq 0$ 16305.26 - 252 z k r s 0 M 1 305 0 M 1 462 0 M 1 353 0 M 1 22 0 M 1 40 0 M 1 99 0 M 1 0	PTO, $\tau \geq 0$ 14753.37 - 292 z k r s 0 M 1 207 0 M 1 463 0 M 1 347 0 M 1 34 0 M 1 39 0 M 1 0 0 M 1 0	PTO, $\tau \geq 0$ 15181.05 - 246 z k r s 0 M 1 221 0 M 1 463 0 M 1 351 0 M 1 21 0 M 1 39 0 M 1 0 0 M 1 0
4	PTO 9162.75	PTO 9965.52	PTO 8598.26	PTO 9070.59	PTO 18627.24	PTO 19058.04	PTO 17932.77	PTO 18300.52
5	BSS 16297.58	BSS 16537.00	BSS 15780.78	BSS 15911.86	IC 24151.10	Kanban 24729.80	Kanban 23353.19	Kanban 23604.80
6	IC 19367.25	IC 19498.24	IC 18937.40	IC 19111.61	Kanban 24423.73	BSS 24780.57	IC 23599.66	IC 24020.77
7	CONWIP 19516.47	CONWIP 20038.72	CONWIP 18997.24	CONWIP 19371.16	BSS 24656.81	IC 25009.82	BSS 24143.78	BSS 24352.44
8	Kanban 23907.48	Kanban 24106.66	Kanban 21752.77	Kanban 22832.37	CONWIP 24873.01	CONWIP 25009.82	CONWIP 24458.80	CONWIP 24648.10
9	LC 43310.17	LC 45881.72	LC 40975.25	LC 42295.15	LC 58988.26	LC 66486.85	LC 55955.90	LC 60693.10

* by using other policy best solution

Table c4-3 **Model 4B**
DCI [single cost; from arrival of ORD-tag]

Best Solu-tion	exponential service time		uniform service time		exponential + bulk service time demand		uniform + bulk service time demand	
	no setup	with setup	no setup	with setup	no setup	with setup	no setup	with setup
1	**PAC** 14495.85 - 773 z k r s 1 11 2 0 0 2 1 0 0 3 1 0 0 4 1 0 0 1 1 23 0 11 1 145 14 3 1 19	**PAC** 17794.54 - 305 z k r s 0 15 1 0 0 12 1 0 0 53 1 0 0 14 1 0 0 31 1 20 0 11 1 0 0 3 1 0	**PAC** 12549.27 - 393 z k r s 0 3 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 1 0 7 1 0 12 2 1 0	**PAC** 14638.29 - 423 z k r s 2 23 3 0 0 1 1 0 0 1 1 0 0 1 1 220 0 1 1 392 0 13 1 0 14 2 1 1	**PAC** 27022.04 - 411 z k r s 0 6 1 0 0 8 1 0 0 2 1 0 0 1 1 0 1 3 1 2 3 1 1 3 5 3 1 0	**PAC** 28263.31 - 413 z k r s 1 3 2 0 0 14 1 0 0 1 1 0 0 1 1 0 0 4 1 0 2 5 1 0 16 3 1 1	**PAC** 25360.13 - 321 z k r s 0 9 1 0 0 1 1 0 0 1 1 0 0 1 1 61 0 1 1 79 0 5 1 80 20 2 1 0	**PAC** 28725.12 - 549 z k r s 0 20 1 0 0 1 1 0 0 1 1 0 0 1 1 90 4 1 1 91 0 9 1 80 2 9 1 2
2	**Kanban** 24479.36 - 71 z k r s 3 3 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 3 3 1 0	**Kanban** 27495.04 - 100 z k r s 3 3 1 0 1 1 1 0 1 1 1 0 1 1 1 0 4 4 1 0 1 1 1 0 3 3 1 0	**Kanban** 22986.14 - 79 z k r s 3 3 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 2 2 1 0	**Kanban** 24505.20 - 57 z k r s 3 3 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 3 3 1 0	**Kanban** 35666.05 - 48 z k r s 3 3 1 0 1 1 1 0 1 1 1 0 1 1 1 0 3 3 1 0 1 1 1 0 3 3 1 0	**Kanban** 37986.14 - 98 z k r s 3 3 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 2 2 1 0 9 9 2 0	**Kanban** 36078.66 - 72 z k r s 4 4 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 2 2 1 0	**Kanban** 38921.61 - 96 z k r s 4 4 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 2 2 1 0 3 3 1 0
3	**IC** 34068.12 - 65 z k r s 4 4 1 0 1 1 1 0 2 2 1 0 1 5 1 0 0 9 1 0 0 4 1 0 0 27 1 0	**MRP** 34875.52 - 229 z k r s 1 M 1 0 0 M 1 0 1 M 2 0 0 M 1 0 4 M 2 0 0 M 1 0 0 M 1 0	**IC** 28746.14 - 37 z k r s 3 3 1 0 1 1 1 0 1 1 1 0 1 4 1 0 0 7 1 0 2 4 1 0 0 21 1 0	**IC** 29820.32 - 32 z k r s 2 2 1 0 1 1 1 0 1 1 1 0 1 3 1 0 0 6 1 0 2 4 1 0 3 21 1 0	**IC** 42605.89 - 77 z k r s 3 3 1 0 1 1 1 0 1 1 1 0 1 4 1 0 1 8 1 0 2 4 1 0 0 24 1 0	**IC** 44744.64 - 90 z k r s 3 3 1 0 1 1 1 0 1 1 1 0 1 4 1 0 0 7 1 0 2 4 1 0 4 25 1 0	**IC** 41820.72 - 137 z k r s 3 3 1 0 1 1 1 0 1 1 1 0 1 4 1 0 0 7 1 0 2 4 1 0 3 24 1 0	**IC** 43854.61 - 103 z k r s 3 3 1 0 1 1 1 0 1 1 1 0 1 4 1 0 0 7 1 0 2 4 1 0 3 24 1 0
4	**CONWIP** 34068.12 - 44 z k r s 4 4 1 0 1 1 1 0 2 2 1 0 1 5 1 0 0 9 1 0 0 4 1 0 0 27 1 0	**IC** 34897.77 - 75 z k r s 4 4 1 0 1 1 1 0 2 2 1 0 1 5 1 0 0 9 1 0 0 4 1 0 0 27 1 0	**CONWIP** 29160.93 - 33 z k r s 2 2 1 0 1 1 1 0 1 1 1 0 1 3 1 0 0 6 1 0 0 2 1 0 0 18 1 0	**CONWIP** 31961.45 - 33 z k r s 2 2 1 0 1 1 1 0 1 1 1 0 1 3 1 0 0 6 1 0 0 2 1 0 0 18 1 0	**CONWIP** 50828.73 - 37 z k r s 2 2 1 0 1 1 1 0 1 1 1 0 2 4 1 0 0 6 1 0 0 2 1 0 0 18 1 0	**CONWIP** 48930.98 - 58 z k r s 4 4 1 0 1 1 1 0 2 2 1 0 1 5 1 0 0 9 1 0 0 4 1 0 0 27 1 0	**CONWIP** 49106.12 - 41 z k r s 4 4 1 0 1 1 1 0 2 2 1 0 1 5 1 0 0 9 1 0 0 4 1 0 0 27 1 0	**CONWIP** 51804.41 - 44 z k r s 5 5 1 0 1 1 1 0 2 2 1 0 1 6 1 0 0 10 1 0 0 4 1 0 0 30 1 0
5	**MRP** 34711.05 - 91 z k r s 1 M 1 0 0 M 1 0 1 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0	**CONWIP** 34897.77 - 51 z k r s 4 4 1 0 1 1 1 0 2 2 1 0 1 5 1 0 0 9 1 0 0 4 1 0 0 27 1 0	**MRP** 31028.31 - 266 z k r s 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 2 M 1 39 18 M 1 1	**MRP** 34581.64 - 223 z k r s 0 M 1 0 0 M 1 0 2 M 3 0 0 M 1 0 0 M 1 0 0 M 1 0 18 M 1 0	**MRP** 68635.27 - 371 z k r s 1 M 1 0 0 M 1 0 1 M 2 0 0 M 1 20 6 M 1 20 2 M 1 58 0 M 1 101	**MRP** 74489.88 - 259 z k r s 3 M 1 0 1 M 4 0 0 M 1 0 0 M 3 0 2 M 1 40 2 M 1 0 2 M 3 0	**MRP** 73340.83 - 423 z k r s 0 M 1 0 0 M 2 0 0 M 1 0 0 M 1 17 3 M 1 38 4 M 1 0 9 M 1 22	**MRP** 77911.16 - 324 z k r s 0 M 1 0 0 M 2 0 3 M 5 0 0 M 1 19 5 M 2 40 2 M 1 40 6 M 2 0
6	**PTO, τ≥0** 35738.34 - 15 z k r s 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0	**PTO, τ≥0** 42008.68 - 134 z k r s 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 20 0 M 1 0 0 M 1 0	**PTO, τ≥0** 35319.98 - 15 z k r s 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0	**PTO, τ≥0** 40508.96 - 15 z k r s 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0	**BSS** 79302.39 - 138 z k r s 3 M 2 0 1 M 2 0 1 M 1 0 1 M 2 0 3 M 3 0 1 M 1 0 5 M 1 0	**BSS** 82984.86 - 134 z k r s 4 M 2 0 1 M 1 0 1 M 1 0 1 M 3 0 1 M 1 0 1 M 1 0 5 M 1 0	**PTO, τ≥0** 77603.82 - 72 z k r s 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 17 0 M 1 41 0 M 1 0 0 M 1 0	**PTO, τ≥0** 81028.48 - 63 z k r s 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 21 0 M 1 39 0 M 1 0 0 M 1 0
7	**PTO** 35738.34 - 1	**BSS** 42837.00 - 113 z k r s 2 M 2 0 1 M 2 0 1 M 1 0 1 M 1 0 1 M 2 0 2 M 1 0 10 M 1 0	**PTO** 35319.98 - 1	**PTO** 40508.96 - 1	**PTO, τ≥0** 79740.22 - 33 z k r s 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 1 0 M 1 21 0 M 1 0 0 M 1 0	**PTO, τ≥0** 84327.51 - 106 z k r s 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 10 0 M 1 30 0 M 1 0 0 M 1 0	**BSS** 82111.72 - 254 z k r s 1 M 2 0 1 M 2 0 1 M 1 0 1 M 2 0 1 M 1 0 8 M 1 0 34 M 1 0	**BSS** 84281.04 - 371 z k r s 2 M 2 0 1 M 2 0 1 M 3 0 1 M 2 0 2 M 4 0 8 M 1 0 27 M 1 0
8	**BSS** 37545.49 - 219 z k r s 2 M 2 0 1 M 2 0 1 M 2 0 1 M 2 0 1 M 2 0 1 M 1 0 7 M 1 0	**PTO** 44496.42 - 1	**BSS** 38205.46 - 206 z k r s 1 M 1 0 1 M 2 0 1 M 1 0 1 M 2 0 1 M 1 0 2 M 1 0 10 M 1 0	**BSS** 41685.76 - 114 z k r s 2 M 1 0 1 M 1 0 1 M 2 0 1 M 2 0 1 M 2 0 4 M 1 0 4 M 1 0	**PTO** 93354.42 - 1	**PTO** 94874.47 - 1	**PTO** 87783.98 - 1	**PTO** 92905.05 - 1
9	**LC** 252143.08 - 110 z k r s 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 95 1 1 0 2 1 1 0 153 1 1 0	**LC** 275241.19 - 124 z k r s 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 102 1 1 0 2 1 1 0 252 1 1 0	**LC** 259415.22 - 118 z k r s 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 542 1 1 0	**LC** 263396.16 - 89 z k r s 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 562 1 1 0	**LC** 250117.17 - 123 z k r s 4 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 92 1 1 0 3 1 1 0 62 1 1 0	**LC** 251825.12 - 111 z k r s 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 523 1 1 0	**LC** 279530.94 - 150 z k r s 49 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 48 1 1 0 2 1 1 0 35 1 1 0	**LC** 293538.88 - 134 z k r s 50 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 48 1 1 0 2 1 1 0 17 1 1 0

Table c4-4 **Model 4B**
DCII [double cost; ORD-REQ]

Best Solu -tion	exponential service time		uniform service time		exponential + bulk service time demand		uniform + bulk service time demand	
	no setup	with setup	no setup	with setup	no setup	with setup	no setup	with setup
1	**PAC** 13371.82 - 2032 z k r s 1 21 2 157 0 2 1 357 0 3 1 214 0 17 1 0 0 1 1 23 0 11 1 145 14 3 1 19	**PAC** 16582.80 - 678 z k r s 0 15 1 193 0 1 1 53 0 5 1 188 0 1 1 0 0 3 1 0 0 11 1 0 0 3 1 0	**PAC** 10807.52 - 1345 z k r s 0 19 1 206 0 1 1 680 0 1 1 179 0 1 1 0 0 1 1 0 0 11 1 0 7 2 1 0	**PAC** 12718.88 - 1663 z k r s 1 23 2 172 0 2 1 411 0 1 1 188 0 1 1 42 0 1 1 43 0 13 1 127 14 2 1 0	**PAC** 26421.67 - 1498 z k r s 0 3 1 393 0 7 1 60 0 3 1 362 0 1 1 0 6 3 1 0 1 1 1 0 0 3 1 0	**PAC** 26055.62 - 1109 z k r s 0 3 1 649 0 11 1 1176 0 3 1 313 0 1 1 0 0 1 1 0 0 7 1 0 8 3 1 1	**PAC** 22508.00 - 2347 z k r s 0 21 1 437 0 1 1 933 0 1 1 305 0 1 1 85 0 1 1 98 0 5 1 30 9 2 1 22	**PAC** 24456.81 - 2345 z k r s 0 19 1 476 0 1 1 1573 0 1 1 347 0 1 1 92 0 1 1 98 0 13 1 99 5 2 1 47
2	**Kanban** 24479.36	**Kanban** 27495.04	**Kanban** 22986.14	**Kanban** 24505.20	**Kanban** 35666.05	**Kanban** 37986.14	**Kanban** 36078.66	**Kanban** 38921.61
3	**MRP** * 30371.16 - 114 z k r s 0 M 1 337 0 M 1 1068 0 M 1 637 0 M 1 0 0 M 1 0 0 M 1 420 0 M 1 121	**MRP** 33583.64 - 978 z k r s 0 M 1 250 0 M 1 300 1 M 2 230 0 M 1 0 4 M 2 0 0 M 1 0 0 M 1 0	**IC** 28746.14	**IC** 29820.32	**IC** 42605.89	**IC** 44744.64	**IC** 41820.72	**IC** 43854.61
4	**PTO, τ≥0** 30371.16 - 337 z k r s 0 M 1 337 0 M 1 1068 0 M 1 637 0 M 1 0 0 M 1 0 0 M 1 420 0 M 1 121	**IC** 34897.77	**CONWIP** 29160.93	**CONWIP** 31961.45	**CONWIP** 50828.73	**CONWIP** 48930.98	**CONWIP** 49106.12	**CONWIP** 51804.41
5	**IC** 34068.12	**CONWIP** 34897.77	**MRP** 29621.18 - 518 z k r s 0 M 1 170 0 M 1 177 0 M 1 134 0 M 1 1 0 M 1 1 2 M 1 27 21 M 1 1	**MRP** 33219.93 - 636 z k r s 0 M 1 208 0 M 1 183 2 M 1 91 0 M 1 0 0 M 1 0 0 M 1 0 18 M 1 0	**MRP** 66939.30 - 967 z k r s 0 M 1 481 0 M 1 91 0 M 2 777 0 M 1 20 6 M 1 20 2 M 1 62 0 M 1 0	**MRP** 68562.38 - 2341 z k r s 2 M 1 398 1 M 4 1223 0 M 1 393 0 M 3 23 2 M 1 40 4 M 1 0 2 M 3 104	**MRP** 71322.64 - 1402 z k r s 0 M 2 491 0 M 2 1985 0 M 1 867 0 M 1 22 0 M 1 40 2 M 1 586 6 M 1 241	**MRP** 73999.77 - 3090 z k r s 0 M 2 403 1 M 4 792 1 M 5 1758 0 M 1 24 0 M 1 42 0 M 1 0 18 M 3 20
6	**CONWIP** 34068.12	**PTO, τ≥0** 35368.88 - 355 z k r s 0 M 1 388 0 M 1 1441 0 M 1 654 0 M 1 0 0 M 1 20 0 M 1 0 0 M 1 141	**PTO, τ≥0** 31655.93 - 267 z k r s 0 M 1 295 0 M 1 1450 0 M 1 606 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0	**PTO, τ≥0** 36460.50 - 317 z k r s 0 M 1 313 0 M 1 1666 0 M 1 709 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0	**PTO, τ≥0** 69261.82 - 543 z k r s 0 M 1 488 0 M 1 1381 0 M 1 1172 0 M 1 4 0 M 1 21 0 M 1 0 0 M 1 100	**PTO, τ≥0** 78259.98 - 862 z k r s 0 M 1 777 0 M 1 1518 0 M 1 1503 0 M 1 10 0 M 1 30 0 M 1 0 0 M 1 0	**PTO, τ≥0** 73068.84 - 304 z k r s 0 M 1 414 0 M 1 1603 0 M 1 1340 0 M 1 22 0 M 1 40 0 M 1 0 0 M 1 0	**PTO, τ≥0** 76125.24 - 359 z k r s 0 M 1 415 0 M 1 1602 0 M 1 1444 0 M 1 21 0 M 1 39 0 M 1 0 0 M 1 0
7	**PTO** 35738.34	**BSS** 42837.00	**PTO** 35319.98	**PTO** 40508.96	**BSS** 79302.39	**BSS** 82984.86	**BSS** 82111.72	**BSS** 84281.04
8	**BSS** 37545.49	**PTO** 44496.42	**BSS** 38205.46	**BSS** 41685.76	**PTO** 93354.42	**PTO** 94874.47	**PTO** 87783.98 - 1	**PTO** 92905.05 - 1
9	**LC** 252143.08	**LC** 275241.19	**LC** 259415.22	**LC** 263396.16	**LC** 250117.17	**LC** 251825.12	**LC** 279530.94	**LC** 293538.88

* by using other policy best solution

Appendix C5

OPTIMIZATION RESULTS FOR MODEL 5

Table c5-1 **Model 5A**
DCI [single cost; from arrival of ORD-tag]

Best Solution	exponential service time		uniform service time		exponential + bulk service time demand	
	no setup	with setup	no setup	with setup	no setup	with setup
1	PAC * 14362.40 - 634 z k r s 1 2 1 0 0 2 1 0 0 5 1 39 0 8 1 0 0 3 1 0 0 5 1 4 0 5 1 40 0 5 1 0 0 4 1 0 0 5 1 40 2 3 1 21	PAC * 18013.95 - 1510 z k r s 1 7 1 0 0 2 1 0 1 11 2 21 6 10 1 1 0 6 1 8 0 5 1 13 0 5 1 60 0 7 1 63 0 3 1 14 0 6 1 83 3 5 3 15	PAC * 10346.86 - 941 z k r s 1 2 1 0 0 7 1 0 0 2 1 78 0 2 1 115 0 7 1 115 0 1 1 105 0 5 1 36 0 7 1 0 0 5 1 0 0 9 1 0 3 7 1 1	PAC* 11480.00 - 1061 z k r s 1 2 1 0 0 2 1 0 0 2 1 103 4 7 1 120 0 7 1 19 0 5 1 6 0 5 1 37 0 5 1 0 0 4 1 0 0 7 1 0 2 4 3 34	PAC * 28975.41 - 688 z k r s 1 3 1 0 0 2 1 0 0 9 1 61 1 10 1 20 1 4 1 39 0 2 1 2 0 2 1 3 0 7 1 40 0 5 1 0 0 4 1 21 0 6 1 0	PAC * 30235.73 - 338 z k r s 1 3 1 0 0 2 1 0 0 3 1 0 3 6 1 20 1 4 1 20 0 2 1 1 1 2 1 0 0 7 1 0 0 5 1 21 0 4 1 20 0 6 1 0
2	MRP* 16507.13 - 358 z k r s 1 M 1 0 0 M 1 0 0 M 1 39 0 M 1 0 0 M 1 0 0 M 1 2 0 M 1 40 0 M 1 0 0 M 1 20 0 M 1 60 2 M 1 22	MRP 20292.25 - 280 z k r s 2 M 2 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 20 0 M 1 60 0 M 1 20 0 M 1 60 2 M 3 0	MRP 12300.50 - 446 z k r s 1 M 1 0 0 M 1 0 0 M 1 120 2 M 1 120 0 M 1 120 0 M 1 80 0 M 1 40 0 M 1 0 0 M 1 0 0 M 1 0 3 M 1 0	MRP 15483.21 - 689 z k r s 1 M 1 0 0 M 1 0 0 M 1 43 2 M 1 99 2 M 1 0 0 M 1 3 0 M 1 60 0 M 1 0 0 M 1 0 0 M 1 0 4 M 3 0	IC 34363.38 - 80 z k r s 3 3 1 0 2 2 1 0 0 5 1 0 0 12 1 0 0 6 1 0 0 2 1 0 0 2 1 0 0 14 1 0 0 15 1 0 0 6 1 0 0 12 1 0	IC 37264.88 - 86 z k r s 3 3 1 0 2 2 1 0 0 5 1 0 0 12 1 0 0 6 1 0 0 2 1 0 0 2 1 0 0 14 1 0 0 15 1 0 0 6 1 0 0 12 1 0
3	PTO, τ≥0 17468.48 - 215 z k r s 0 M 1 0 0 M 1 0 0 M 1 40 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 40 0 M 1 0 0 M 1 0 0 M 1 40 0 M 1 22	IC 19227.35 - 160 z k r s 2 2 1 0 1 1 1 0 0 3 1 0 0 7 1 0 0 4 1 0 1 1 1 0 0 2 1 0 0 10 1 0 0 9 1 0 0 4 1 0 3 10 1 0	PTO, τ≥0 13654.80 - 80 z k r s 0 M 1 0 0 M 1 0 0 M 1 9 0 M 1 3 0 M 1 3 0 M 1 78 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0	PTO, τ≥0 16270.67 - 128 z k r s 0 M 1 0 0 M 1 0 0 M 1 2 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 105 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0	CONWIP 34363.38 - 26 z k r s 3 3 1 0 2 2 1 0 0 5 1 0 0 12 1 0 0 6 1 0 0 2 1 0 0 2 1 0 0 14 1 0 0 15 1 0 0 6 1 0 0 12 1 0	CONWIP 37264.88 - 26 z k r s 3 3 1 0 2 2 1 0 0 5 1 0 0 12 1 0 0 6 1 0 0 2 1 0 0 2 1 0 0 14 1 0 0 15 1 0 0 6 1 0 0 12 1 0
4	IC 18594.70 - 80 z k r s 2 2 1 0 1 1 1 0 0 3 1 0 2 9 1 0 0 4 1 0 0 1 1 0 0 1 1 0 0 8 1 0 0 9 1 0 0 4 1 0 0 9 1 0	PTO, τ≥0 20592.18 - 180 z k r s 0 M 1 0 0 M 1 0 0 M 1 1 0 M 1 0 0 M 1 4 0 M 1 0 0 M 1 20 0 M 1 60 0 M 1 40 0 M 1 0 0 M 1 0	PTO 13876.95 - 1	IC 16446.19 - 86 z k r s 2 2 1 0 1 1 1 0 0 3 1 0 0 7 1 0 0 4 1 0 0 1 1 0 0 1 1 0 0 8 1 0 0 9 1 0 0 4 1 0 0 7 1 0	MRP 35590.03 - 585 z k r s 0 M 1 0 0 M 1 0 0 M 1 2 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 21 0 M 1 19 0 M 1 23 2 M 1 1 4 M 3 0	MRP* 43305.64 - 294 z k r s 1 M 3 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 1 M 1 22 0 M 1 20 0 M 1 0 0 M 1 24 0 M 1 0

5	**PTO** 18789.94 - 1	**PTO** 21639.23 - 1	**IC** 14198.81 - 84 z k r s 1 1 1 0 1 1 1 0 0 2 1 0 2 7 1 0 2 4 1 0 0 1 1 0 0 1 1 0 0 6 1 0 2 8 1 0 0 4 1 0 0 7 1 0	**CONWIP** 16446.19 - 21 z k r s 2 2 1 0 1 1 1 0 0 3 1 0 0 7 1 0 0 4 1 0 0 1 1 0 0 1 1 0 0 8 1 0 0 9 1 0 0 4 1 0 0 7 1 0	**PTO, τ≥0** 38627.20 - 63 z k r s 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 40 0 M 1 0	**PTO, τ≥0** 44261.10 - 88 z k r s 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 22 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0
6	**CONWIP** 19016.02 - 21 z k r s 2 2 1 0 1 1 1 0 0 3 1 0 0 7 1 0 0 4 1 0 0 1 1 0 0 1 1 0 0 8 1 0 0 9 1 0 0 4 1 0 0 7 1 0	**CONWIP** 23601.55 - 21 z k r s 2 2 1 0 1 1 1 0 0 3 1 0 0 7 1 0 0 4 1 0 0 1 1 0 0 1 1 0 0 8 1 0 0 9 1 0 0 4 1 0 0 7 1 0	**CONWIP** 15734.52 - 21 z k r s 2 2 1 0 1 1 1 0 0 3 1 0 0 7 1 0 0 4 1 0 0 1 1 0 0 1 1 0 0 8 1 0 0 9 1 0 0 4 1 0 0 7 1 0	**PTO** 16539.88 - 1	**PTO** 39053.00 - 1	**PTO** 49148.04 - 1
7	**Kanban** 22960.03 - 58 z k r s 2 2 1 0 1 1 1 0 3 3 2 0 2 2 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0	**BSS** 24985.45 - 100 z k r s 1 M 1 0 1 M 1 0 1 M 2 0 1 M 1 0 1 M 1 0 1 M 2 0 1 M 1 0 1 M 1 0 1 M 1 0 1 M 1 0 3 M 1 0	**Kanban** 18560.31 - 55 z k r s 1 1 1 0 1 1 1 0 3 3 2 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0	**Kanban** 19467.83 - 95 z k r s 1 1 1 0 1 1 1 0 3 3 2 0 2 2 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0	**Kanban** 42514.61 - 147 z k r s 2 2 1 0 1 1 1 0 3 3 1 0 4 4 1 0 2 2 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 2 2 2 0 1 1 1 0	**BSS** 50093.80 - 192 z k r s 2 M 3 0 1 M 1 0 1 M 2 0 3 M 1 0 3 M 1 0 1 M 2 0 1 M 1 0 3 M 1 0 1 M 1 0 1 M 1 0 1 M 1 0
8	**BSS** 23301.18 - 103 z k r s 1 M 1 0 1 M 2 0 1 M 1 0 3 M 1 0 1 M 1 0 1 M 1 0 1 M 1 0 1 M 1 0 1 M 2 0 1 M 1 0 1 M 2 0	**Kanban** 27019.80 - 146 z k r s 2 2 1 0 1 1 1 0 4 4 2 0 2 2 1 0 2 2 2 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0	**BSS** 20342.15 - 100 z k r s 1 M 1 0 1 M 2 0 1 M 2 0 3 M 1 0 1 M 1 0 1 M 1 0 1 M 1 0 1 M 1 0 1 M 1 0 1 M 1 0 1 M 1 0	**BSS** 21572.92 - 105 z k r s 1 M 1 0 1 M 2 0 1 M 2 0 3 M 2 0 1 M 1 0 1 M 2 0 1 M 1 0 1 M 1 0 1 M 1 0 1 M 1 0 1 M 1 0	**BSS** 43033.66 - 130 z k r s 1 M 2 0 1 M 2 0 1 M 2 0 3 M 1 0 3 M 1 0 1 M 1 0 1 M 3 0 1 M 1 0 1 M 1 0 1 M 1 0 1 M 1 0	**Kanban** 57227.79 - 158 z k r s 2 2 1 0 1 1 1 0 4 4 1 0 2 2 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 3 3 1 0 1 1 1 0
9	**LC** 36625.12 - 138 z k r s 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 4 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0	**LC** 39936.02 - 221 z k r s 2 1 1 0 2 1 1 0 4 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 4 1 1 0 8 1 1 0 2 1 1 0 8 1 1 0	**LC** 35371.50 - 67 z k r s 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0	**LC** 34760.14 - 23 z k r s 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0	**LC** 53830.52 - 412 z k r s 2 1 1 0 2 1 1 0 3 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 5 1 1 0 6 1 1 0 2 1 1 0 9 1 1 0	**LC** 71069.88 - 276 z k r s 2 1 1 0 2 1 1 0 6 1 1 0 6 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 6 1 1 0 6 1 1 0 4 1 1 0 6 1 1 0

* by using other policy best solution

Table c5-2 **Model 5A**
DCII [double cost; ORD-REQ]

Best Solution	exponential service time		uniform service time		exponential + bulk service time demand	
	no setup	with setup	no setup	with setup	no setup	with setup
1	**PAC *** 13071.58 - 627 z k r s 0 3 1 266 0 5 1 282 0 5 1 219 0 7 1 0 0 5 1 0 0 5 1 0 0 3 1 62 0 5 1 100 0 5 1 0 0 5 1 80 0 5 1 80	**PAC *** 15342.74 - 977 z k r s 1 3 1 154 0 3 1 317 0 7 1 183 0 11 1 40 0 7 1 20 0 7 1 18 0 5 1 58 0 7 2 80 0 7 1 18 0 7 1 40 0 5 1 20	**PAC*** 7185.83 - 830 z k r s 0 2 1 165 0 1 1 184 0 2 1 111 2 4 1 61 0 5 2 62 0 5 1 58 0 5 1 36 0 11 1 0 0 9 1 0 0 5 1 0 0 5 1 0	**PAC *** 8438.03 - 1097 z k r s 0 2 1 182 0 1 1 205 0 7 1 50 0 9 1 0 0 7 1 0 0 5 1 120 0 5 1 69 0 13 1 0 0 4 1 0 0 7 2 0 2 3 3 21	**PAC *** 26900.41 - 343 z k r s 0 2 1 0 0 2 1 0 1 3 1 1 1 4 1 0 0 2 1 0 0 1 1 18 1 2 1 0 0 5 1 21 2 5 1 0 2 4 1 0 2 6 1 0	**PAC*** 24559.95 - 2776 z k r s 0 4 1 367 0 2 1 387 0 3 1 5 2 8 1 17 1 4 1 20 0 2 1 5 0 2 1 27 0 7 1 0 0 5 1 0 0 4 1 20 0 6 2 54
2	**MRP*** 14522.29 - 81 z k r s 0 M 1 261 0 M 1 284 0 M 1 219 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 122 0 M 1 100 0 M 1 0 0 M 1 80 0 M 1 80	**MRP*** 16297.47 - 314 z k r s 1 M 1 256 0 M 1 277 0 M 1 163 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 58 0 M 1 80 0 M 1 0 0 M 1 39 2 M 1 0	**MRP** 9248.70 - 763 z k r s 0 M 1 162 0 M 1 179 0 M 1 52 2 M 1 61 0 M 1 62 0 M 1 37 0 M 1 56 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0	**MRP*** 11479.19 - 615 z k r s 0 M 1 183 0 M 1 193 0 M 1 30 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 71 0 M 1 0 0 M 1 0 0 M 1 0 2 M 1 20	**MRP** 32419.06 - 916 z k r s 0 M 1 291 0 M 1 280 0 M 1 20 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 54 4 M 1 18 0 M 1 44 0 M 1 60 6 M 3 0	**IC** 37264.88
3	**PTO, τ≥0** 14522.29 - 218 z k r s 0 M 1 261 0 M 1 284 0 M 1 219 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 122 0 M 1 100 0 M 1 0 0 M 1 80 0 M 1 80	**PTO, τ≥0** 16586.24 - 301 z k r s 0 M 1 260 0 M 1 278 0 M 1 143 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 78 0 M 1 80 0 M 1 0 0 M 1 39 0 M 1 0	**PTO, τ≥0** 10020.86 - 179 z k r s 0 M 1 195 0 M 1 216 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 80 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0	**PTO, τ≥0** 12283.72 - 287 z k r s 0 M 1 231 0 M 1 230 0 M 1 2 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 105 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0	**PTO, τ≥0** 32738.88 - 239 z k r s 0 M 1 358 0 M 1 352 0 M 1 40 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 140 0 M 1 100 0 M 1 40 0 M 1 140 0 M 1 0	**CONWIP** 37264.88
4	**IC** 18594.70	**IC** 19227.35	**PTO** 13876.95	**IC** 16446.19	**IC** 34363.38	**MRP *** 38827.20 - 418 z k r s 0 M 1 367 0 M 1 373 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 22 0 M 1 20 0 M 1 0 0 M 1 34 0 M 1 0
5	**PTO** 18789.94	**PTO** 21639.23	**IC** 14198.81	**CONWIP** 16446.19	**CONWIP** 34363.38	**PTO, τ≥0** 39206.44 - 234 z k r s 0 M 1 365 0 M 1 379 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 22 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0
6	**CONWIP** 19016.02	**CONWIP** 23601.55	**CONWIP** 15734.52	**PTO** 16539.88	**PTO** 39053.00	**PTO** 49148.04
7	**Kanban** 22960.03	**BSS** 24985.45	**Kanban** 18560.31	**Kanban** 19467.83	**Kanban** 42514.61	**BSS** 50093.80
8	**BSS** 23301.18	**Kanban** 27019.80	**BSS** 20342.15	**BSS** 21572.92	**BSS** 43033.66	**Kanban** 57227.79

9	LC 36625.12	LC 39936.02	LC 35371.50	LC 34760.14	LC 53830.52	LC 71069.88

* by using other policy best solution

Table c5-3 **Model 5B**

DCI [single cost; from arrival of ORD-tag]

Best Solution	exponential service time		uniform service time		exponential + bulk service time demand	
	no setup	with setup	no setup	with setup	no setup	with setup
1	PAC * 23467.99 - 421 z k r s 2 3 1 0 0 2 1 0 0 5 1 21 0 6 1 0 0 2 1 1 0 2 1 0 0 1 1 0 0 7 1 0 0 4 1 0 2 4 1 0 2 6 1 0	PAC * 24025.54 - 506 z k r s 1 2 1 0 1 1 1 0 2 2 1 0 2 1 1 0 1 1 1 1 0 2 1 0 2 2 1 1 2 2 2 0 1 2 1 0 1 1 1 0 1 1 1 0	PAC * 14532.95 - 1023 z k r s 1 2 1 0 0 1 1 0 0 3 1 62 1 2 1 0 0 2 1 0 0 1 1 124 0 1 1 0 0 8 1 0 1 2 1 22 0 2 1 0 2 6 2 14	PAC * 17334.44 - 895 z k r s 1 2 1 0 0 2 1 0 0 2 1 0 2 4 1 140 0 3 2 0 0 2 1 100 0 1 1 60 0 8 1 0 0 10 1 0 0 2 1 0 1 3 1 40	PAC * 30646.87 - 509 z k r s 3 4 1 0 0 1 1 0 3 3 4 0 3 3 2 2 3 3 1 4 3 3 3 0 1 3 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0	PAC * 29127.15 - 288 z k r s 3 5 1 0 1 1 1 0 3 3 3 0 3 3 2 0 3 3 1 0 3 3 3 0 1 3 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0
2	IC 28015.51 - 80 z k r s 2 2 1 0 2 2 1 0 0 5 1 0 0 12 1 0 0 6 1 0 0 2 1 0 0 2 1 0 0 14 1 0 0 15 1 0 0 6 1 0 0 12 1 0	Kanban 30396.79 - 96 z k r s 4 4 1 0 1 1 1 0 2 2 1 0 2 2 1 0 1 1 1 0 2 2 1 0 2 2 1 0 2 2 2 0 1 1 1 0 1 1 1 0 1 1 1 0	IC 18297.23 - 97 z k r s 2 2 1 0 1 1 1 0 0 3 1 0 1 8 1 0 1 5 1 0 0 1 1 0 0 1 1 0 0 8 1 0 0 9 1 0 0 5 1 0 1 9 1 0	IC 21371.68 - 95 z k r s 2 2 1 0 1 1 1 0 0 3 1 0 3 10 1 0 0 4 1 0 1 2 1 0 0 2 1 0 0 10 1 0 0 9 1 0 0 4 1 0 0 10 1 0	Kanban* 44206.20 - 158 z k r s 6 6 1 0 1 1 1 0 5 5 4 0 3 3 2 0 3 3 1 0 3 3 3 0 3 3 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0	Kanban 40507.43 - 194 z k r s 5 5 1 0 1 1 1 0 3 3 3 0 3 3 2 0 3 3 1 0 3 3 3 0 3 3 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0
3	CONWIP 28015.51 - 26 z k r s 2 2 1 0 2 2 1 0 0 5 1 0 0 12 1 0 0 6 1 0 0 2 1 0 0 2 1 0 0 14 1 0 0 15 1 0 0 6 1 0 0 12 1 0	IC 30642.41 - 87 z k r s 3 3 1 0 2 2 1 0 0 5 1 0 0 12 1 0 0 6 1 0 0 2 1 0 0 2 1 0 1 15 1 0 0 15 1 0 0 6 1 0 2 14 1 0	MRP 21273.14 - 561 z k r s 1 M 1 0 0 M 1 0 0 M 1 80 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 81 0 M 1 0 0 M 1 0 0 M 1 0 3 M 1 20	CONWIP 24747.27 - 23 z k r s 2 2 1 0 2 2 1 0 0 4 1 0 0 10 1 0 0 4 1 0 0 2 1 0 0 2 1 0 0 12 1 0 0 12 1 0 0 4 1 0 0 10 1 0	IC 48939.51 - 161 z k r s 3 3 1 0 2 2 1 0 0 5 1 0 2 14 1 0 2 8 1 0 0 2 1 0 0 2 1 0 0 14 1 0 1 16 1 0 0 8 1 0 2 16 1 0	IC 63116.73 - 173 z k r s 4 4 1 0 3 3 1 0 0 7 1 0 0 17 1 0 2 10 1 0 0 3 1 0 0 3 1 0 1 21 1 0 0 21 1 0 0 10 1 0 1 18 1 0
4	Kanban* 28624.91 - 119 z k r s 2 2 1 0 1 1 1 0 2 2 1 0 2 2 1 0 4 4 4 0 1 1 1 0 1 1 1 0 1 1 1 0 3 3 1 0 1 1 1 0 4 4 1 0	CONWIP 31651.83 - 26 z k r s 3 3 1 0 2 2 1 0 0 5 1 0 0 12 1 0 0 6 1 0 0 2 1 0 0 2 1 0 0 14 1 0 0 15 1 0 0 6 1 0 0 12 1 0	CONWIP 21395.07 - 23 z k r s 2 2 1 0 1 1 1 0 0 3 1 0 0 7 1 0 0 4 1 0 0 1 1 0 0 1 1 0 0 8 1 0 0 9 1 0 0 4 1 0 0 7 1 0	MRP 28876.12 - 492 z k r s 1 M 1 0 0 M 1 0 0 M 1 40 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 83 0 M 1 0 0 M 1 0 0 M 1 0 4 M 3 0	CONWIP 49203.03 - 44 z k r s 3 3 1 0 2 2 1 0 0 5 1 0 0 12 1 0 0 6 1 0 0 2 1 0 0 2 1 0 0 14 1 0 0 15 1 0 0 6 1 0 0 12 1 0	CONWIP 64625.04 - 25 z k r s 5 5 1 0 3 3 1 0 0 8 1 0 0 19 1 0 0 10 1 0 0 3 1 0 0 3 1 0 0 22 1 0 0 24 1 0 0 10 1 0 0 19 1 0
5	MRP 30050.36 - 337 z k r s 1 M 1 0 0 M 1 0 0 M 1 0 2 M 1 0 0 M 1 20 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 21 0 M 1 34	MRP 35461.66 - 281 z k r s 2 M 1 0 1 M 1 0 0 M 2 0 6 M 4 0 0 M 1 1 0 M 1 0 0 M 2 0 0 M 1 21 0 M 1 0 0 M 1 20 0 M 1 40	Kanban 21577.02 - 119 z k r s 1 1 1 0 1 1 1 0 3 3 2 0 2 2 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0	BSS 32030.57 - 218 z k r s 1 M 1 0 1 M 1 0 1 M 2 0 3 M 1 0 2 M 1 0 1 M 2 0 1 M 1 0 1 M 1 0 1 M 1 0 1 M 1 0 3 M 2 0	MRP 72555.24 - 305 z k r s 4 M 4 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 40 0 M 3 21 0 M 1 0 2 M 1 2	BSS 96482.91 - 301 z k r s 2 M 2 0 1 M 2 0 1 M 1 0 5 M 1 0 1 M 1 0 1 M 3 0 1 M 1 0 1 M 1 0 1 M 1 0 2 M 1 0 3 M 4 0
6	PTO, τ≥0 31719.91 - 154 z k r s 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 39 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 40 0 M 1 47	BSS 38413.70 - 207 z k r s 1 M 1 0 1 M 1 0 1 M 2 0 3 M 1 0 3 M 1 0 1 M 2 0 1 M 1 0 1 M 1 0 1 M 1 0 1 M 1 0 1 M 1 0	PTO, τ≥0 25129.51 - 23 z k r s 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0	PTO, τ≥0 32945.17 - 64 z k r s 0 M 1 0 0 M 1 0 0 M 1 20 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 60 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0	PTO, τ≥0 79026.00 - 266 z k r s 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 20 0 M 1 62 0 M 1 0 0 M 1 20 0 M 1 56 0 M 1 80 0 M 1 0	MRP 96664.92 - 183 z k r s 4 M 3 0 2 M 3 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0

7	**PTO** 33907.47 - 1	**PTO, $\tau\geq0$** 41282.78 - 133 z k r s 0 M 1 0 0 M 1 0 0 M 1 41 0 M 1 0 0 M 1 1 0 M 1 0 0 M 1 1 0 M 1 0 0 M 1 41 0 M 1 40 0 M 1 0	**PTO** 25129.51 - 1	**PTO** 33228.11 - 1	**BSS** 81483.64 - 264 z k r s 2 M 2 0 1 M 1 0 1 M 2 0 5 M 1 0 5 M 1 0 1 M 1 0 1 M 2 0 5 M 1 0 1 M 1 0 1 M 1 0 1 M 3 0	**PTO, $\tau\geq0$** 104337.67 - 121 z k r s 0 M 1 0 0 M 1 0 0 M 1 19 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 39 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0
8	**BSS** 34103.85 - 266 z k r s 1 M 1 0 1 M 1 0 1 M 1 0 7 M 3 0 3 M 1 0 1 M 2 0 1 M 1 0 1 M 1 0 1 M 3 0 1 M 1 0 1 M 1 0	**PTO** 45892.11 - 1	**BSS** 28346.95 - 154 z k r s 1 M 1 0 1 M 1 0 1 M 2 0 3 M 1 0 1 M 1 0 1 M 2 0 1 M 1 0 1 M 1 0 1 M 1 0 1 M 1 0 2 M 1 0	**Kanban** 36602.46 - 134 z k r s 2 2 1 0 1 1 1 0 3 3 1 0 2 2 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0	**PTO** 90940.54 - 1	**PTO** 115614.69 - 1
9	**LC** 49050.56 - 152 z k r s 3 1 1 0 2 1 1 0 2 1 1 0 4 1 1 0 4 1 1 0 2 1 1 0 2 1 1 0 8 1 1 0 20 1 1 0 2 1 1 0 10 1 1 0	**LC** 177852.72 - 212 z k r s 52 1 1 0 2 1 1 0 10 1 1 0 12 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 6 1 1 0 10 1 1 0 2 1 1 0 14 1 1 0	**LC** 33711.03 - 23 z k r s 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0	**LC** 47482.74 - 135 z k r s 5 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 2 1 1 0 6 1 1 0 2 1 1 0 2 1 1 0	**LC** 109704.66 - 256 z k r s 10 1 1 0 2 1 1 0 6 1 1 0 2 1 1 0 7 1 1 0 2 1 1 0 2 1 1 0 3 1 1 0 16 1 1 0 4 1 1 0 6 1 1 0	**LC** 288919.56 - 368 z k r s 75 1 1 0 2 1 1 0 8 1 1 0 2 1 1 0 6 1 1 0 2 1 1 0 2 1 1 0 13 1 1 0 14 1 1 0 2 1 1 0 13 1 1 0

* by using other policy best solution

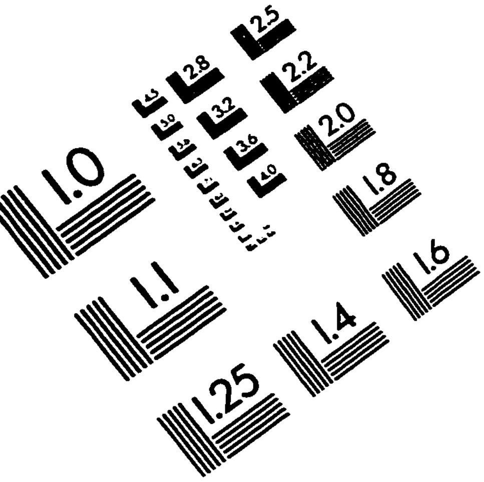
Table c5-4 **Model 5B**
DCII [double cost; ORD-REQ]

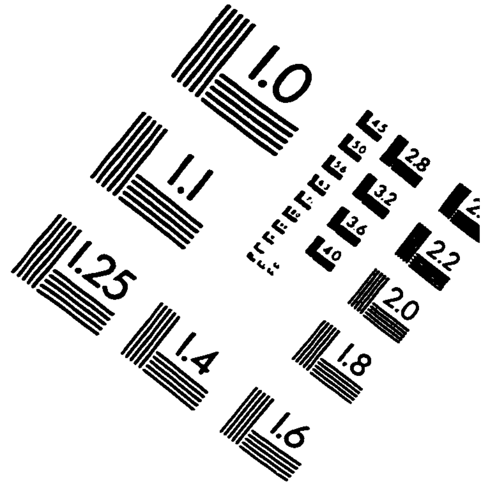
Best Solution	exponential service time		uniform service time		exponential + bulk service time demand	
	no setup	with setup	no setup	with setup	no setup	with setup
1	**PAC *** 20051.80 - 1365 z k r s 0 3 1 345 0 8 1 348 0 3 1 0 2 7 1 1 0 5 1 21 0 4 1 43 0 5 1 1 0 5 1 20 0 3 1 0 0 5 1 40 0 3 1 40	**PAC *** 24025.54 - 508 z k r s 1 2 1 0 1 1 1 0 2 2 1 0 2 1 1 0 1 1 1 1 0 2 1 0 2 2 1 1 2 2 2 0 1 2 1 0 1 1 1 0 1 1 1 0	**PAC *** 12130.18 - 2580 z k r s 0 2 1 233 0 1 1 268 0 3 1 80 2 2 1 0 4 2 2 30 0 1 1 80 0 1 1 20 0 14 1 0 1 2 1 35 0 2 1 0 2 6 3 1	**PAC *** 15228.78 - 2652 z k r s 0 2 1 303 0 2 1 251 0 2 1 100 2 4 1 22 14 5 2 39 0 2 1 161 0 1 1 57 0 8 1 0 0 6 1 0 0 2 1 0 1 3 1 44	**PAC *** 30646.87 - 510 z k r s 3 4 1 0 0 1 1 0 5 5 4 0 3 3 2 2 3 3 1 4 3 3 3 0 1 3 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0	**PAC *** 29127.15 - 288 z k r s 3 5 1 0 1 1 1 0 3 3 3 0 3 3 2 0 3 3 1 0 3 3 3 0 1 3 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0
2	**MRP*** 26779.68 - 77 z k r s 0 M 1 356 0 M 1 367 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 41 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 40 0 M 1 60	**Kanban** 30396.79	**IC** 18297.23	**IC** 21371.68	**Kanban** 44206.20	**Kanban** 40507.43
3	**PTO, τ≥0** 26779.68 - 292 z k r s 0 M 1 356 0 M 1 367 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 41 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 40 0 M 1 60	**IC** 30642.41	**MRP** 19847.67 - 1166 z k r s 1 M 1 97 0 M 1 246 0 M 1 177 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 65 0 M 1 0 0 M 1 0 0 M 1 0 3 M 1 23	**CONWIP** 24747.27	**IC** 48939.51	**IC** 63116.73
4	**IC** 28015.51	**CONWIP** 31651.83	**PTO, τ≥0** 20519.20 - 168 z k r s 0 M 1 283 0 M 1 319 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0	**MRP** 26005.12 - 704 z k r s 0 M 1 297 0 M 1 328 0 M 1 259 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 82 0 M 1 0 0 M 3 0 0 M 1 0 3 M 1 1	**CONWIP** 49203.03	**CONWIP** 62625.04
5	**CONWIP** 28015.51	**MRP** 33533.51 - 1730 z k r s 1 M 1 240 0 M 1 494 0 M 2 0 3 M 4 0 0 M 1 1 0 M 1 0 0 M 2 0 0 M 1 21 0 M 1 0 0 M 1 20 0 M 1 40	**CONWIP** 21395.07	**PTO, τ≥0** 27492.92 - 292 z k r s 0 M 1 352 0 M 1 369 0 M 1 200 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 82 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0	**MRP** 70290.51 - 933 z k r s 0 M 3 737 0 M 1 565 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 20 0 M 1 0 0 M 1 40 4 M 1 0	**MRP** 91426.38 - 895 z k r s 0 M 3 963 0 M 3 1037 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 0
6	**Kanban** 28624.91	**PTO, τ≥0** 38071.74 - 375 z k r s 0 M 1 485 0 M 1 492 0 M 1 420 0 M 1 80 0 M 1 100 0 M 1 100 0 M 1 60 0 M 1 80 0 M 1 0 0 M 1 140 0 M 1 0	**Kanban** 21577.02	**BSS** 32030.57	**PTO, τ≥0** 72181.67 - 526 z k r s 0 M 1 543 0 M 1 564 0 M 1 0 0 M 1 20 0 M 1 1 0 M 1 81 0 M 1 1 0 M 1 30 0 M 1 0 0 M 1 60 0 M 1 0	**PTO, τ≥0** 92562.05 - 416 z k r s 0 M 1 737 0 M 1 671 0 M 1 81 0 M 1 0 0 M 1 0 0 M 1 0 0 M 1 60 0 M 1 60 0 M 1 0 0 M 1 0 0 M 1 0

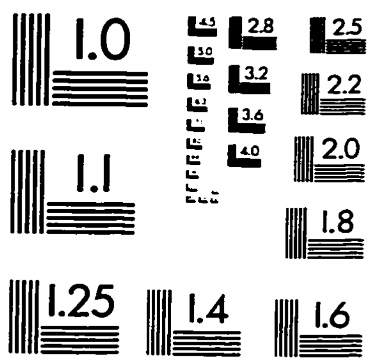
7	PTO 33907.47	BSS 38413.70	PTO 25129.51	PTO 33228.11	BSS 81483.64	BSS 96482.91
8	BSS 34103.85	PTO 45892.11	BSS 28346.95	Kanban 36602.46	PTO 90940.54	PTO 115614.69
9	LC 49050.56	LC 177852.72	LC 33711.03	LC 47482.74	LC 109704.66	LC 288919.56

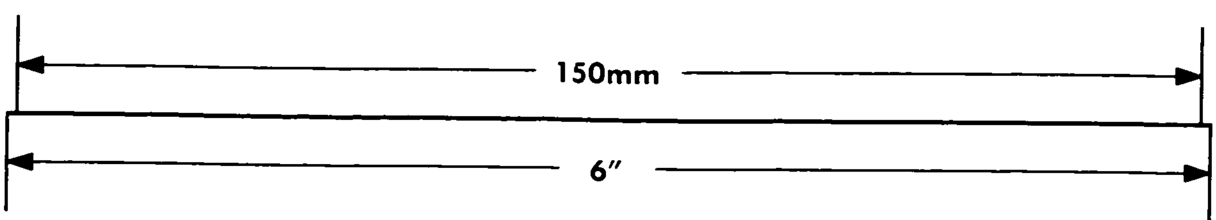
* by using other policy best solution

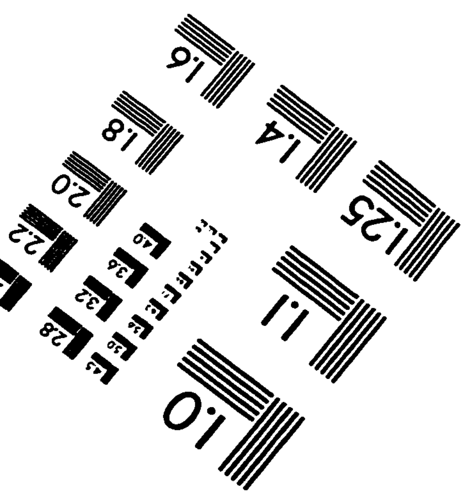
IMAGE EVALUATION TEST TARGET (QA-3)

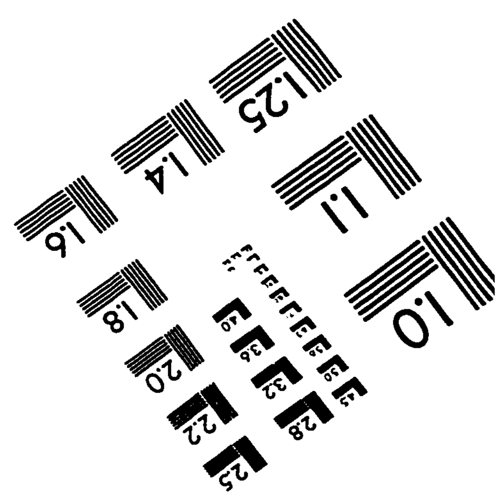












APPLIED IMAGE . Inc
1653 East Main Street
Rochester, NY 14609 USA
Phone: 716/482-0300
Fax: 716/288-5989