OPTIMIZATION OF THE NAVAL SURFACE SHIP RESOURCE-CONSTRAINED PROJECT SCHEDULING PROBLEM

by

Eric L. J. Bertrand

Submitted in partial fulfilment of the requirements
for the degree of Masters of Applied Science

at

Dalhousie University
Halifax, Nova Scotia
June 2020

Dedication

To my wife Ashley, my daughter Evelyn, and my son Nicholas, for their support and contagious happiness.

"The greatest enemy of a good plan, is the dream of a perfect plan" – Carl Von Clausewitz, 1780-1831 Prussian general and military theorist

Table of Contents

List of Tables

List of Figures

Abstract

The research presented in this thesis focuses on proposing effective methodologies, formulations, and heuristics for scheduling and rescheduling resource-constrained project scheduling problems that are specific to naval surface ship work periods or similar maintenance projects. The network topology for these work period projects is divided into precedence-independent work orders that have precedence-dependent operations within themselves. Mixed-integer linear programming (MILP) models are developed for the initial scheduling and for the rescheduling problem. In initial scheduling, the objective is to front-load work based on priority and duration, to account for the high degree of uncertainty and scope growth that is common in these work periods. In rescheduling, the goal shifts to minimizing schedule deviation-days while incorporating urgent scope growth. Experimentation results, discussions, and insights are provided for initial and rescheduling MILP models, and for combining appropriate heuristic methods to quickly produce good feasible solutions when optimal solution times are not acceptable.

List of Abbreviations Used

ADM(Mat)	Assistant Deputy Minister (Materiel)
AEAP	As Early as Possible (scheduling priority rule)
ALAP	As Late as Possible (scheduling priority rule)
AJISS	AOPS and JSS ISS
AOPS	Arctic Offshore Patrol Ship
C.I.	Confidence Interval
CCPM	Critical Chain Project Management
CM	Corrective Maintenance
CM WO	Corrective Maintenance Work Order
CPM	Critical Path Method
DND	Department of National Defence
DRMIS	Defence Resource Management Information System
DT	Discrete time
DWP	Docking Work Period
EBST	Earliest Baseline Activity Starting Time
EC	Engineering Changes
ERP	Enterprise Resource Planning
ES	Early Start
EWP	Extended Work Period
FMFCB	Fleet Maintenance Facility Cape Breton (Esquimalt, BC)

FMFCS	Fleet Maintenance Facility Cape Scott (Halifax, NS)
HAL	Halifax (frigate ship class)
HDW	Harry De-Wolfe (patrol ship class)
ISI	Irving Shipbuilding Incorporated
ISS	In-Service Support
ISSC	In-Service Support Contractor
JSS	Joint-Support Ship (tanker ship class)
LS	Late Start
MILP	Mixed Integer Linear Programming
NSS	Naval Surface Ships
NSWPP	Naval Surface Ship Work Period Problem
OEM	Original Equipment Manufacturer
PL	Project Leader
PM	Preventive Maintenance
PM WO	Preventive Maintenance Work Order
PSP	Project Scheduling Problem
PSPC	Public Service and Procurement Canada
RCAF	Royal Canadian Air Force
RCN	Royal Canadian Navy
RCPSP	Resource-Constrained Project Scheduling Problem
SGS	Schedule Generation Scheme
SS	Ship's Staff

SWP	Short Work Period
VBA	Visual Basic for Applications
WO	Work Order, Work Package

Acknowledgments

Thank you to the Fleet Maintenance Facility Cape Scott and the Royal Canadian Navy for the full-time post-graduate sponsorship program; most adults with a family and a mortgage cannot stop active duty to pursue higher education without suffering difficult life changes. Not only was the maintenance organization funding this program, my colleagues at the facility were also helpful in assisting with research and automated-scheduling prototype usage in a real-world project.

Thank you to my advisors Drs. Claver Diallo and Alireza Ghasemi for your guidance, encouragement, and advice throughout my research. Thank you to my peers, Sanjay Prabhu and soon-to-be-Dr Zhuojun Liu for your respective assistance with parallel research at Dalhousie University and VBA to Python programming conversions.

Thank you to Thales Canada for the grant propelling this research for Dalhousie University, Université de Laval, and École Polytechnique de Montréal, as part of the “Refit Optimizer” Project, as well as for the active participation of Dr. Daniel Lafond and his team.

Chapter 1: Introduction

The Naval Surface Ship Work Period Problem (NSWPP) is a highly complex variant of the resource-constrained project scheduling problem (RCPSP) with many work orders (WOs) that are in themselves smaller projects. Planning, scheduling, and executing NSWPPs are very challenging, and a large volume of planned and scheduled work is typically never achieved for a variety of reasons.

Maintenance facilities and organizations managing these challenging projects are usually left with scheduling tools that are part of a larger enterprise resource planning (ERP) system and have disabled automated scheduling systems to prevent financial and contractual mishaps. This thesis investigates and develops new exact and heuristic solution methods for the NSWPP with the aim of contributing to the development of effective add-on software to complement naval surface ship maintenance facility ERP systems.

1.1 Background

Thales Canada, as part of multi-billion dollar in-service-support contract (ISSC) for six Harry-De-Wolfe (HDW) Class and two Protecteur Class Ships, is sponsoring Canadian Research to contribute in building NSWPP optimization software. This software is being developed over several years, in collaboration with three research teams from Dalhousie University, École Polytechnique de Montréal, and Université Laval. The software is intended to work as a sandbox standalone program, where inputs arrive from the ERP system, the schedule is optimized, then results are re-entered into the ERP system. Five major Canadian maintenance organizations are playing roles in this project as potential customers including: Thales Canada that will operate on both coasts, the Royal Canadian Navy's (RCN) Fleet Maintenance Facilities (FMF Cape Scott (FMFCS) in Halifax NS and FMF Cape Breton (FMFCB), in Esquimalt BC), Irving Ship Building Incorporated in Halifax (ISI), and Seaspan in Victoria BC. All five organizations use ERP systems where scheduling add-on systems may be used to enhance scheduling, and all five organizations currently more-or-less manually schedule work for their work periods.

This thesis is written by Lieutenant-Commander Eric Bertrand, who is sponsored by FMFCS to complete his MASc Program in Industrial Engineering at Dalhousie University. The author has been involved in the planning, scheduling, and execution of maintenance work periods in HAL Class Frigates and worked in the FMFCS Operations Department following his Head of Marine Systems Engineering Department posting in HMCS Ville de Québec from 2015 to 2017.

This thesis is the culmination of exploratory research and experimentation at several phases throughout early 2019 to mid-2020. Work for this thesis was completed jointly with another fellow graduate student, Sanjay Prabhu. Results from complementary experimentations and research were compared, and analysis of these results along with Industry clients' and sponsors' feedback guided successive experimentation and research.

1.2 Purpose

The purpose of this thesis is to propose effective formulations, methodologies, and heuristics for scheduling and re-scheduling RCPSPs that are particularized for NSWPPs or similar maintenance projects. We propose the use of two mixed-integer linear programming (MILP) formulations that better capture the practical resource-constrained project scheduling problem (RCPSP) operational objectives. These objectives are derived from several interviews and meetings with real-world scheduling users for scenarios that Thales and its potential clients may face while executing ISS contracts with the Royal Canadian Navy (RCN). For decision support, baseline comparison, and quick response times to very large problems, we propose using common and adapted heuristic techniques including priority-based serial schedule generation schemes (serial SGS) and priority-based adaptations of the shifting bottleneck procedure (SBP) with effective data-processing techniques. These are compared with experimentation using scenario-based objectives to determine which ones are more effective in each situation, and to elaborate on the pros and cons of each. These heuristics may be used to give a solver a "good" initial solution so that the user has the option of getting a solution in a reasonable timeframe for the common scenarios where short solve times are especially important. The RCPSP

scenarios used for experimentation are inspired by real-world data with many characteristics not commonly found in literature.

Chapter 2: Problem Definition

The Naval Surface Ship Work Period Problem (NSWPP) Scheduling Problem is a highly complex problem. The term "project" in this thesis refers to all work that is planned and scheduled during a period of time when a ship is available for maintenance. The length of the project from start to finish is defined as "makespan" and the end of the makespan is a point in time called the "time horizon".

Routine maintenance activity projects, such as those in a land-based power plant, may be limited in duration and scope where a particular equipment unit is safely disconnected/de-energized, disassembled, maintained, reassembled, tested, and returned to service. This also occurs in Naval Surface Ships (NSSs). However, the ship itself is often away on deployment so a repair facility has limited access to the ship, and all of these "small projects" must occur during limited time windows in a coordinated manner. The maintenance actions desired in the ship normally surpass what can feasibly be accomplished during the periods that the ship is available, and work must thus be prioritized. As long as the ship can safely sail and accomplish its next mission, it will sail and the work period will be over, even though not all desired maintenance was accomplished.

The plans for these smaller projects, such as in the power plant equipment repair example, in this thesis are referred to as "WOs" or "work packages", as these terms are used interchangeably in the NSS industry. A WO or work package consists of a number of "operations/tasks/jobs" that represent the individual tasks required to complete the work package. Note that these terms are used interchangeably in academia and industry. In this thesis, the term "activity" will be used to define these individual tasks. These activities may be sequential, meaning that they need to occur one after the other, and this is often called a "precedence" constraint where an activity cannot start until the predecessor is completed. Some activities may also occur concurrently, so that two or more of these may share the same precedence constraint. These elements represent the

main aspects of traditional project scheduling problems (PSPs) [1] and are present in NSWPPs.

The NSWPP project is therefore defined as the sum of all WOs and their associated activities, for a single ship, that are all scheduled to occur within a defined time period, called work window, where the ship will be in repair alongside a jetty near a repair facility or in a drydock within a repair facility.

When a ship is alongside a jetty, the work period may last between two weeks to several months, but since the ship is partially submerged in water, work on the submerged exterior of the ship, such as on the hull, sonar dome(s), the propellers, and the rudder is limited to cleaning and inspections. Figure 1 is an image of a ship in an alongside work period, while figure 2 is a picture of the front base of a ship in a docking work period. The docked ship is supported by numerous blocks all along the base and keel of the ship, while supporting brace pieces are placed on the sides of the ship to prevent transverse movements.

Figure 1: A ship in an "alongside" work period (source: The U.S. National Archives https://catalog.archives.gov/)

NSWPPs differ from the RCPSP in a number of ways. Firstly, not all activities need to be completed. The majority of PSP studies assume that all activities need to be completed and dummy start and end nodes are added to the project. The goal is generally to complete all activities within the makespan. As will be discussed in the following sections, the goals of the NSWPP are different. The goal is better described as: to

complete all important and critical activities within the makespan, where uncompleted important activities will not compromise the mission or can be reasonably mitigated.

Figure 2: HMCS Ville de Québec in a "docking" work period

A submarine or an aircraft, in comparison, must have all outstanding maintenance activities completed, as failures can have major and catastrophic consequences. In a surface ship, uncompleted work usually results in a lack of redundancy as most systems are highly redundant, and in a worst-case scenario, the ship floats, so people are in no immediate danger of dying except in combat scenarios. In aircraft or submarines, every system is laid on top of each other as they must be as compact as possible. This means that whenever a system needs maintenance, layers of other systems must be removed to complete repairs. Due to efficiency reasons, as systems are replaced and exposed, maintenance actions such as inspections must be performed at the right times, leading to a highly precedence-dominated project network structure as illustrated in figure 3. All activities are represented by a numbered node and the arrows represent precedence relationships between activities.

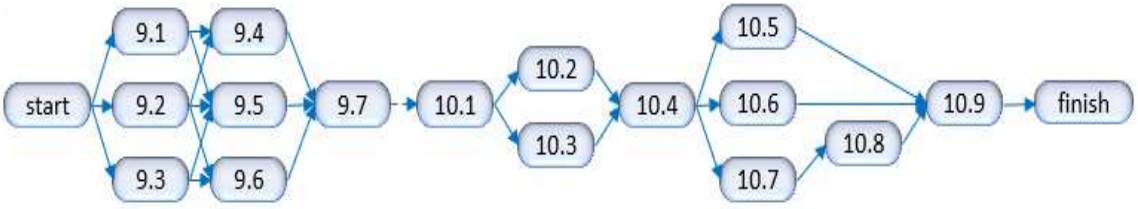


Figure 3: A standard precedence-dominated project network structure in RCPSP

In NSWPPs, although this network structure does exist between activities within the same WOs, most WOs are relatively independent of each other. Because the ship is very long and wide, and has multiple decks, equipment is separated such that one piece of equipment can be maintained while other equipment even in the same compartment may be untouched or simply shutdown. This relaxed space environment removes the need for many precedence constraints between WOs but causes the scheduling problem to have a much larger solution space as WOs may fit almost anywhere within the work window. Calculating the early start and late starts of activities is relatively trivial when considering only the precedence constraints. The harder challenges are managing the resource, same space, and exclusivity constraints (to be defined later), as well as producing robust schedules that can handle the high variability of task durations and scope growth that are inherent in NSWPPs.

Surface ships also differ from aircraft or smaller vehicles because of their sheer size. Ships hold and consume so much fuel that the unit of measure is cubic meters instead of liters. They must hold food for weeks and make potable water from seawater. Although the equipment units do not need to be on top of each other, they are only normally accessible through a limited number of passageways and ladders. For example, work in a passageway will often block access to a number of compartments. Surface ships are so big, that there are no hangars, at least in the RCN, where ships can be worked on within the hangar and therefore be sheltered from the elements (i.e. rain and snow). For example, the HAL Class Frigate has 353 spaces, four main engine rooms, a hangar for a helicopter, several small boats, and dozens of different high-technology armaments. Naval surface ships cost billions of dollars and can accommodate hundreds of people for long durations; thus, they are more comparable to small floating cities.

NSWPP projects do not only involve maintenance. Since ships must last several decades, they must remain relevant as technologies evolve in the geopolitical environment. Eventually, equipment needs to be replaced with different newer models and systems need to be able to function in an integrated way, often requiring several engineering changes (ECs) for these new equipment replacements. Weapon,

communication, and detection systems must remain state-of-the-art and are in constant need of ECs. The steel surrounding everything in a ship seems to be constantly corroding. The decks, bulkheads, and structural members need renewal. Multiple corrosion-inhibiting technologies including paints, passive and active cathodic protection systems, and sacrificial anodes are used to prevent the hull from corroding; however, the hull structure still corrodes in and around equipment, under various obstacles where people live, and is often undetected until inspections are performed. This is expected as ships are submerged in saltwater or surrounded by high salt-fog humidity for the majority of their lives. The inside of the ship is temperature and humidity controlled, but moisture does enter all spaces and colder areas are subject to frequent condensation buildup. Sailors need to work around this maintenance work, and a vast array of testing and trials needs to occur within the same projects. These projects therefore comprise of equipment maintenance, steel renewal, ECs, testing and trials, all while maintaining reasonable living arrangements within the ship. Thus, the need for an effective comprehensive maintenance, hull renewal, and EC strategy is necessary.

NSWPP problems are notoriously difficult because of the very numerous actions needed in these given work periods. There are so many things to fix and maintain that trying to fix everything in these short time frames is generally unfeasible; therefore, a common goal is to fit as much higher priority maintenance and ECs as is practical in the work window. The work periods vary in length and complexity, and they fit into a ship's lifecycle that will be described in the next section.

2.1 The lifecycle of a ship

Although figure 4 depicts a typical 5-year lifecycle of the HAL Class Frigate, it is similar to the lifecycle of the HDW Class AOPS or other Canadian NSSs. For an example of how this can differ, the Victoria Class Submarines have a 7-year life cycle, mostly because of the need to strike a balance between its long maintenance periods and deployments. During a 2015 Mess Dinner, Vice Admiral Mark Norman described a global rule of thumb when he stated: "...to have a fully-capable naval ship out at sea 365 days a year, a nation

must have three to four of these ships available at various parts of the lifecycle at all times".

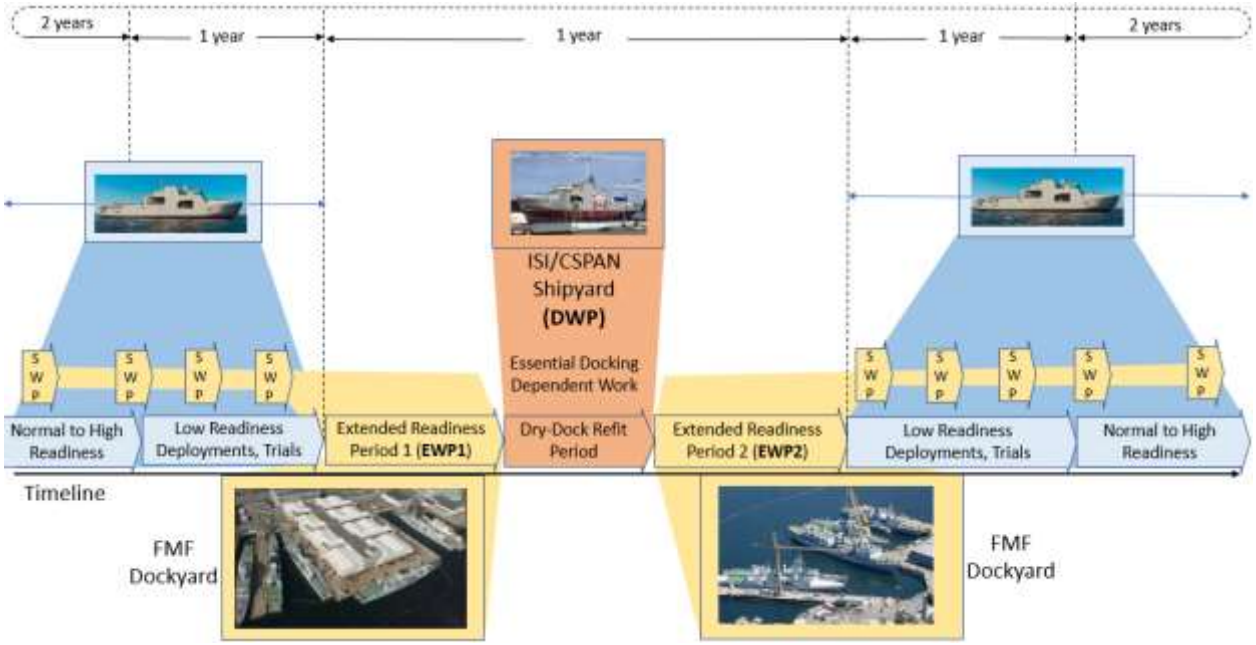


Figure 4: Illustrating a RCN ship's five-year lifecycle

As previously mentioned, there are two major work period types: docking refit work periods (DWP) and alongside work periods (EWP and SWP). Figure 4 shows the DWP at the center. This docking refit period is designed to perform heavy maintenance when the ship is out of the water. It may last 20-30 weeks when the ships are relatively new but will gradually extend to over 50 weeks as the ship ages (after 25-30 years) and more corrosion-related corrective actions are needed. Before and after a docking work period, a ship enters an extended work period (EWP). The EWPs, labelled EWP1 and EWP2 for before and after the DWP respectively, last several months and comprise docking preparation, preventive maintenance (PM), and corrective maintenance (CM) work. A ship must have all cryptographic and IT systems, special systems, and ammunition removed, and have almost no liquids remaining in its tanks before docking. After docking, systems need to be reactivated, special systems need to be reinstalled, the ship needs to have its ammunition and liquids (fuels, lubricants, etc.) replenished, and the construction

zone that is the inside of a ship in a DWP must be rendered into a clean and livable ship interior, which is why these EWPs take so long.

Outside of the EWP1-DWP-EWP2 time window, as seen in figure 4, the ship will have several SWPs. These will sum up to approximately 12-20 weeks per year for ongoing maintenance and ECs, all that can be achieved alongside with the ship in water. Most SWPs will last 3 to 10 weeks and will fit in as much maintenance as is practical within the SWP time window as doing everything desired is infeasible. Except for when the ship is in the DWP, the ship is manned 24/7 by RCN staff, whose engineering departments will continuously raise notifications for more work to be planned, scheduled, and completed as defects are found.

2.2 NSWPP Planning Process

When anything breaks or needs repair in a ship, the technical ship staff (SS), composed of two engineering departments making up almost half the ship's company, will generally try to fix the problem themselves. When the problem cannot be properly repaired, SS will raise a "notification" in the DND's Defence Resource Information Management System (DRMIS), which is an ERP system used by all four branches of the military: the Royal Canadian Navy (RCN), the Royal Canadian Army (RCA), the Royal Canadian Air Force (RCAF), and the Special Forces Branch. These notifications contain enough information for the shore repair facilities to start planning WOs for upcoming work periods. Work that can be fixed by SS is called 1st line work, referring to the first level of complexity. The 2nd line work is generally completed by shore repair facility organizations such as FMF, SNC Lavalin, or Thales, where specialized workers and tools are needed, and 3rd line work is usually associated with equipment overhauls that are completed by the original equipment manufacturers (OEMs).

It should be noted that for an ISS organization like Thales, DRMIS may not necessarily be used by SS to report problems. As is the case with SNC Lavalin and the ISS of the RCN's maritime coastal defence vessels (MCDVs), a separate defect notification system is used to notify the RCN and the responsible ISS organization. Even with a

separate notification system, the process remains relatively unchanged, where the ISS organization will take defect reports from SS, plan corrective action WOs, and schedule NSWPPs. Regardless of the ship type and the maintenance organization managing the work, PM, CM, and ECs will need to be planned, scheduled, and executed.

2.3 Automatic Scheduling in an ERP

During project execution, disruptions in a project's schedule can be repaired via manual adjustment, but the remainder of the schedule will not be automatically re-scheduled; thus, this leads to unintended future work conflicts in terms of space and personnel resources. Software suites such as Primavera P6 and MS Project are capable of auto-rescheduling, but most industries including the one being analyzed still rely on manual scheduling [1]. Once a software is used for accounting and tracking of contractual agreements, it becomes very dangerous to allow the scheduler or project leaders (PLs) to mass-modify information in the centralized system [2]. It can be so detrimental, that in all organizations reviewed in this research, mass modification of the centralized system was prohibited by schedulers or PLs. This in turn forces schedulers or PLs to modify WO and activity dates one by one, leading to a tedious process. The tediousness of this process has a secondary effect of limiting the number of WOs that are analyzed for scheduling, resulting in missed opportunities and lower than ideal worker utilization [3]. It has been observed at FMFCS that re-scheduling an entire project during project execution without considering other projects is not practical because the shop resources shared with other projects are already tied up, leading to a solution space that is very limited in how it can accept change.

2.3.1 Solution Time

Since NSS projects compete for personnel and equipment resources from a larger centralized pool, it is important for any automatic scheduling system to be able to solve relatively quickly. If the solution takes too long, the underlying data used in a calculation may change during the problem processing, as a different PL and scheduler will typically schedule at least a few WOs (with several activities in each) every day during a project. With continuously having 4-7 surface ship projects ongoing (such as at FMFCS), plus

projects that are being planned in anticipation of upcoming projects, plus additional submarine, tanker, and miscellaneous support projects being scheduled for things other than surface ships, that all use the same resource pool, the underlying resource capacity data is constantly changing and is generally shrinking. For example, a solver that takes hours to solve would be unacceptable because this would first require a download of shop availabilities, which would change during processing, usually for the worse, while the program tries to find an optimal solution. Schedulers at FMFCS were very concerned with a scheduling solution that was offline from their SAP system because of the frequent changes to the underlying shop capacity data [3]. In addition to this, and because of the stochastic nature of NSWPPs, the interviewed PLs desired a tool that was fast. The need to re-schedule is very frequent because of scope growth as defects are found, and because of the need for analyzing various options where WOs were added or removed from a project. If verifying shop capacities and dates for adding or removing WOs could be done faster with manual scheduling than with an automatic scheduling tool, then the interviewed PLs described that this tool would be a failed product and guaranteed that this would be quickly abandoned [3]. For an automatic scheduling tool to be useful in the NSWPP environments that have been surveyed for this thesis, the tool must solve within no more than a few minutes during project execution, although it may be acceptable to take a little longer during initial project planning [4] [3]. Furthermore, it is recommended, from prototype usage at FMFCS in real-world projects, that any tool should be useful for a secondary purpose: finding flaws in the underlying data. For instance, if the wrong shop was selected as part of the planning data, then a very favorable or unfavorable solution start date might result. The user can easily find the mistake by looking at the solution. Another commonly found mistake when using a prototype scheduler at FMFCS as part of this research (see section 6.5 for more detail), was incorrect shop capacity information. If certain activities become scheduled too far in the future, then the user can see this outcome and realize that the underlying data requires further investigation. A user might have a default compartment capacity that prevents two important WOs from being scheduled at the same time, while the PLs knows that these can realistically occur in

parallel; he/she will then adjust the capacity and re-schedule, etc. This is so common that it should be expected every time a new NSWPP data-set is downloaded, as well as frequently throughout a project's execution [3]. For these reasons, an automatic scheduling process can require multiple iterations every time it is used to schedule, and the user does not have time to wait for long periods at every iteration.

The interviewed project managers, PLs, and schedulers in Halifax indicated that set-up speed, ease-of-use, and computation speed are even more important than getting optimal schedules; where most users had a hard time saying the word "optimal" without rolling their eyes [3], since the underlying data is considered far from accurate.

2.4 Special Important Constraints

NSWPPs have unique combinations of constraints that may be modelled in an RCPSP. Investigations have revealed that current scheduling practices with commercial software do not model all unique NSWPP constraints, leading to avoidable conflicts and work delays [3] [4] [5]. These uncommon constraints are described in further detail in the following sub-sections.

2.4.1 Same Compartment

Due to space limitation in a navy ship, most tasks are mutually exclusive to each other (i.e. from separate WOs) and do not fit in the same compartment. For example, in figure 5, work must be performed overhead in an electrical switchboard compartment; therefore, one of the WO activities include building staging (a term used interchangeably with scaffolding but refers generally to lower platforms). The staging itself takes up the majority of the compartment's horizontal surface area and the workers will have their tools placed throughout the compartment such that they can readily access them without losing them. This "consumption" of available working area in the compartment makes it very impractical to have a second WO executed at the same time in the same compartment. In this example, several equipment units mounted on the bulkheads and access to the switchboard itself is severely limited. So, by default, a major constraint should be to have limits on parallel work in each space at the same time. This is not only related to a WO's footprint but also by the nature of the work. As previously mentioned, decks are often corroded, meaning that tiles will need

Figure 5: Staging often excludes other work in the same compartment.

Figure 6: Deck work generally makes all nearby equipment inaccessible.

to be removed, underlying steel will need to be stripped and renewed, then new tiles and under-coatings will need to be replaced. Figure 6 is a picture of all the equipment in a passageway being wrapped in plastic for deck work. It gives a good illustration of why no other unrelated concurrent work can occur in this passageway. To add to even further space and compartment issues, as a secondary effect, work in any compartment that is only easily accessible by this passageway should also be avoided. There are escape hatches that can be used if necessary, but the clearance in these are so small that many equipment pieces do not fit through the escape hatches, and interviewed PMs with decades of experience suggest that these should be avoided unless absolutely necessary [4].

Figure 7: Larger compartments may accommodate more than one concurrent WO at a time.

There are however exceptions where some compartments may accommodate more than one WO. Figure 7 shows the inside of an auxiliary engine room where maintenance is being conducted on a diesel generator. In one of these larger spaces, several WOs may occur at the same time up to a limit, determined mostly by the nature of the work, the size of equipment pieces being moved, staging, chemicals used, whether equipment requires it being operated etc. For example, when a diesel generator is in operation, it is so loud and hot in the engine space, and generates vibrations so that other WOs are not concurrently compatible.

To illustrate work capacity settings, when discussing the subject with PLs working on HAL Class Frigates, the initial default values for maximum concurrent activities (from different WOs) per space were set at 1 per space for 348 spaces, five activities per space for both main engine rooms, three activities per space for both auxiliary engine rooms, and three activities for the helicopter hangar. Depending on the work being scheduled,

the user desired the ability to change these limits as the practical limits for this depended on the nature of the work. A large WO may consume an entire compartment, but in some cases many smaller activities can occur simultaneously [3].

2.4.2 Hot Work

Hot work is work relating to activities that involve stripping coatings and rust from steel for renewal, including the cutting out of deteriorated steel sections and the re-installation of new steel sections with welding. This work produces toxic gases, is very loud, may cause damage to eyes, and can burn any flammable materials or persons. Often when hot work is being conducted, the space on the other side of the deck or bulkhead should be out of bounds. Without considering this important source of potential work conflicts, two separate WOs may be scheduled such that they are not compatible with each other. Figure 8 shows a steel section that was recently renewed in a HAL Class Frigate. The compartment on the other side and below this renewal section would need to be empty of people (other than a fire sentry) for the duration of the hot work activities. Portable ventilation units and temporary trunking will also be installed to remove toxic gases throughout the hot work activities in the main compartment and the affected adjacent compartments. These hot work activities can last a long time (weeks) as welding practices are such that only staggered small sections of an insert can be welded at a time. Welding too quickly may result in bending and warping of the surrounding steel from excess heat; therefore, hot work has long durations associated with letting the steel cool down between successive welding sessions on the same steel pieces. This in turn

Figure 8: Hot work on this corner affects three adjacent spaces, placing them out-of-bounds.

means that the adjacent space is out-of-bounds for a significant duration. A practical method of determining these conflicts beforehand is proposed in this thesis to ensure that schedules consider these potential conflict sources.

2.4.3 Emissions and Radiation

In this thesis, emissions refer to the exhaust gases that exit from funnels into the atmosphere as a result of gas/diesel-powered rotating equipment being operated. Radiation refers to non-ionizing radiation that is emitted as a result of active radars and antennas. Both emissions and radiation pose significant risks to human health. Every radar has a maximum exposure limit (MEL) that is dependent on the intensity and direction of the radiation being emitted. This MEL is essentially a safe distance to the emitter that a human can stand such that long-term exposure to this radiation would not pose a statistically significant increased risk of burning or cancer. Emitters on HAL Class Ships have a MEL ranging between 1 meter and 345 meters. Figure 9 is a picture of HMCS Ville de Québec alongside Québec City with all emitters highlighted in yellow. Naval ships require multiple emitters for detecting ships, boats, aircraft, missiles, and projectiles, guiding weapon systems and defense mechanisms, sending out jamming signals, communication, and receiving satellite television entertainment while at sea. Emission and radiation equipment is locked out using standard lock-out and tag-out procedures to ensure that any work on the upper decks, which is any work completed above the weatherdeck such as the bridge top, funnel tops, hangar top, and on the mast, poses no risk from injury as a results of active emitters.

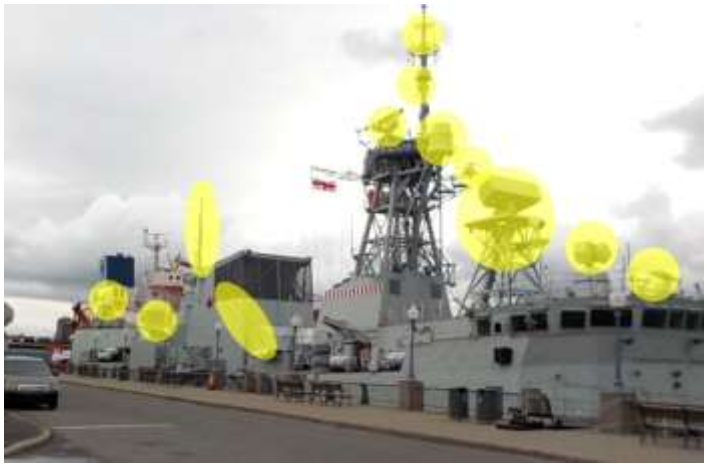


Figure 9: Naval ships and boats have many radiation emitting units (highlighted in yellow).

Above and beyond upper deck work, crane activities may also be affected by emitters. Jetty cranes, where the operator is above the ship, may become unusable when gas-powered equipment is in use and vice versa, as the crane operator may be exposed to significant exhaust gases. Mobile cranes may be used during emissions because the operator is near ground level (see figure 10), so exhaust gases should not affect him/her. Active radiation emitters however are incompatible with all types of cranes due to a public health concern of re-radiation from unintended emitters. It has been found that wires under tension from cranes may in some circumstances become unintentional receiving antennas, leading to electrical shock or burning injury in workers coming into contact with these wires [6]. For this reason, whenever cranes are in use near a ship in Canada, workers will lock-out all radiation-emitting equipment before using the crane.

Figure 10: With mobile cranes, the operator is near ground level and safe from emissions.

If the schedule building methodology being used does not consider radiation and emission aspects, two WOs may be inadvertently planned at the same time where they will in fact cause a conflict because they are mutually exclusive. Examined datasets and interviews with schedulers indicated that this is currently done by PM and scheduler knowledge, and work packages are manually planned so that they do not occur at the same time. Of course, mistakes are occasionally made and this adds to the tediousness of the scheduling process [3].

2.4.4 Miscellaneous Delays and Conflicts

Miscellaneous delays are quite common in NSWPPs. These include special circumstances such as a media event on a ship's flight deck, receptions, photo shoots, training exercises using the ship's general alarm system, diving activities, and work cancellations due to high winds, rain, or snow, that may all happen during a ship's work period.

Most of these are self-explanatory, but some may require elaboration: for instance, diving near a ship to perform inspections as part of PM WOs can be a dangerous activity. The side of the ship where the divers are in the water must remain out-of-bounds so that objects do not fall on divers. The rudder, propeller, and sonar dome must be locked out. Any equipment with water intakes and outputs must not change in their operating conditions so that divers going near an intake can first test for suction or thrust, determine a safe distance, and be confident that the suction or thrust intensity of water near the location will not change. Diving precautions will even go further by attempting to move active suction bays to parts of the ship that are far away from divers [7]. Two Canadian Navy divers died from drowning from a suction bay inlet miscommunication with the USS Pharris in 1991 [8].

As for weather, much of the repair and renewal work occurs on the outside of the ship, being on the superstructure, the upper decks, on the weather deck, and on the sides of the ships. These areas are persistently exposed to near 100% humidity with a significant salinity level in the air known as “salt fog” [9]. Due to this exposure and other environmental factors, components contacting the outside of the ship degrade faster and require more maintenance in general. Thus, a proportionately larger than normal amount of work is performed outside where it is at the mercy of the weather. This work often requires scaffolding and cranes, and is frequently unsafe to perform when it is raining, snowing, or when winds produce gusts above 42 km/hr [10]. Halifax and Esquimalt, where Canadian NSWPP occur, receive significant weather events causing delays. Halifax has on average 136 rain days and 24 snow days per year, while Vancouver (near Esquimalt) has 162 rain days and 11 snow days per year on average [11] [12]. This does not add a hard constraint, but certainly increases the variability in task duration and expected unscheduled delays that may be present in a NSWPP. During project execution, a scheduler or PL may be able to use short-term weather forecasts to re-schedule certain activities or determine the impact of upcoming weather events.

Despite these delays and their effect on variability, schedules are still needed to complete as much work as is possible; however, even if everything goes right, some WOs take much longer to complete than others and will not be possible in SWPs.

2.4.5 Duration

Looking back at figure 4 and its associated ship lifecycle descriptions, one may notice that some work periods have relatively short makespans and some have much longer makespans, so the simple goal of completing as many WOs as possible is insufficient. For example, an automatically optimized scheduled plan to complete as many WOs as possible would, by the nature of this objective function, schedule as many short and relatively "easy" WOs as possible and would completely ignore the longer duration WOs.

For instance, 60M WOs, meaning WOs that are calendar-based and occur every five years or 60 months, can usually only fit in longer DWPs or EWPs. Some ECs and larger hull repairs can take several weeks but can still fit in some SWPs with more than a three to four-week duration. It should be noted however on a practical level, that a PL who is responsible for scheduling a ship's activities in alongside work periods, should have knowledge of WO duration estimates after they are planned and may manually decide which WOs to consider for upcoming work periods. At FMFCS, the Prometheus Scheduler add-on automatic scheduling feature does not consider duration, so a practical proposed work around is the following: first manually schedule the important long-duration work packages that can really only begin near the start of work periods (because the sequence of activities takes so long), then use automatic scheduling to fit-in all the shorter WOs in a prioritized fashion until nothing else "fits" [13].

For these reasons, it is proposed that WO durations be part of the solution when considering heuristics or formulations, with an ability to "tune" the importance of duration in both methodologies.

2.4.6 Priority

As previously mentioned, a major difference between submarine/aircraft work and naval surface ship work is the concept that every activity needs to be completed. In surface ships, work is triaged into priority levels indicating how important it is to complete the work. Whenever conflicts arise, and varying priority levels are present, the higher priority WOs will be completed first as per RCN policy [14], unless rare exceptions make this impractical. Priority is generally categorized into three major groups: essential work (ESS), high opportunity work (HOPP), and normal opportunity work (NOPP), and are also called priority levels 1, 2, and 3. ESS WOs are critical and a failure to complete these mean that the ship cannot complete the entirety of its next mission as intended. For example, a radar suite capable of communicating with allied ships on a designated SECRET network requires an upgrade to be able to participate in the next mission; if this is not fitted and made to work, then the ship cannot complete the next mission, making this an ESS WO.

HOPP and NOPP work will only be considered as long as they do not impede on priority ESS work from being completed. RCN and occasionally the Assistant Deputy Minister of Materiel Group (ADM(Mat)) senior officials collectively agree on priority levels assigned to WOs, but not to individual activities. Every activity in a WO inherits the priority of the WO [15]. If a WO must be completed for another WO to be performed, then this WO will also inherit its successor's priority. A successor WO however, need not necessarily have the same priority level as its predecessor if the successor's priority is lower. In other words, if a HOPP WO needs a ESS WO to first be performed, then it does not inherit the ESS status, but if an ESS WO needs a HOPP WO to be first performed, then the HOPP WO will inherit the ESS status, thus becoming an ESS WO.

The three priority levels are also associated with operational deficiency reporting (OPDEF) levels that become active during all phases of a ship's lifecycle, except during DWPs and pre-DWP EWP1s. A category-1 OPDEF requires reporting within 4 hours, and the fleet Commander (Atlantic or Pacific, depending on the ship's home base) is informed immediately once shore authorities are informed, regardless if this happens at odd hours such as 03h00 when this person would normally be sleeping. A category-1 OPDEF is

something that renders the following upcoming mission(s) unfeasible [16]. For example, if all replenishment-at-sea (RAS) stations are not repaired and certified for re-fuelling with a tanker, where performing a RAS is part of an upcoming NATO exercise, then the lack of RAS capability is considered a CAT-1 OPDEF. Equivalently, WOs associated with repairs and certification of a preferred RAS station become a priority-1 WO. In this example, once a single RAS station is repaired (out of four), then the OPDEF will be downgraded to a CAT-2 OPDEF. Any WOs associated with repairing the other RAS stations will be priority-2 to match. A CAT-3 OPDEF will remain active, usually after a second RAS station is certified, until all four RAS stations are certified for use. The equivalent WOs for repairs of RAS stations 3 and 4 will adopt priority-3 status or (equivalently NOPP) until they are completed. Except during DWPs and EWP1s, WO priority levels generally match OPDEF levels, except where there are oversights in updating the DRMIS data linked to the associated OPDEF status.

It should be noted that precedence between WOs in SWPs is rare. Precedence between WOs is more common in EWPs but still not normal; however, in DWPs, WO precedence relationships are more common. An example of this may be deck renewals: all decks in a level may be tripped, have hot work to repair all deteriorated sections, then have the decks renewed. Since heavy equipment must be moved over these decks using rigging equipment, all heavy equipment travels that cross these decks should be done beforehand to prevent deck-rework, as these movements often damage decks [3].

2.5 High Variability

As previously eluded to, work in NSWPPs is subject to high variability in terms of activity durations and scope growth. For a relative example, the HAL Class Frigate marine system has a main control system that continuously monitors and controls equipment systems in the ship, comprising over 5000 sensors and 8000 data points [17]. Despite all these sensors, a vast proportion of equipment health cannot be determined by sensors: partial disassembly and visual inspection or testing are required. Many calendar-based preventive maintenance (PM) routines have inspection components at various stages that will incur additional maintenance if conditions are discovered that require it. To illustrate,

a five-day PM routine is scheduled, and everything fits nicely in a SWP before it begins. During project execution however, after the first day of this PM routine, a problem is found needing additional repairs. This problem may simply slightly extend the duration of the PM routine if the repair is small and required materials are present; however, if the problem needs a more invasive solution, then the PM WO may be halted while a new repair WO is planned, scheduled, and executed. This delay may mean that the PM routine itself needs to be delayed to a later work period if possible. If it cannot be delayed, then other work occurring in that space or using the same resources may need to shift, be cancelled, and/or re-scheduled. On the other hand, if the repair can wait until the next work period, then the current PM WO will be cancelled as it cannot be completed with the defect present, and a new CM WO and a new PM WO will be generated for the following work period. This would have the effect of significantly shortening the duration of the planned PM WO for the current work period.

This type of scope growth example is one of many things that can happen during a work period to break a schedule. Above and beyond scope growth found during PM routines, workers may become ill, materials needed for an activity may be incorrect once the packaging is opened, it might rain, something else very important may break as some equipment is used by SS throughout a work period, and the PM action itself may induce failures. Main engines and radars are stationary and not used during a work period until close to the departure date for the next mission: usually a ship's primary equipment is tested for free movement on a Friday when departure is scheduled for the following Monday morning. Rotating mechanical equipment may fail while in standby due to seizures that occur during these stationary periods, so new and especially important failures may (and often do) occur only a few days before the end of a project. The interviewed project management staff stated the same thing: "the moment work begins, a printed schedule is no longer accurate" [3].

2.6 Examination of Project Networks

In the majority of the literature reviewed for this project, such as those found in the PSLIB [18], it was found that the studied RCPSPs used classic precedence network topologies that are normally found in construction project or maintenance projects for aircraft, smaller vehicles, or submarines, where tasks have a high degree of precedence and every activity is expected to be finished. Priority is not considered because all activities are necessary. Postponing a project often represents a failure, but the project will be postponed if needed, although perhaps sped up with overtime, until all activities are completed. Figure 11 is an illustration of this type of network topology for aircraft maintenance work periods as studied by Croteau (2015) [19]:

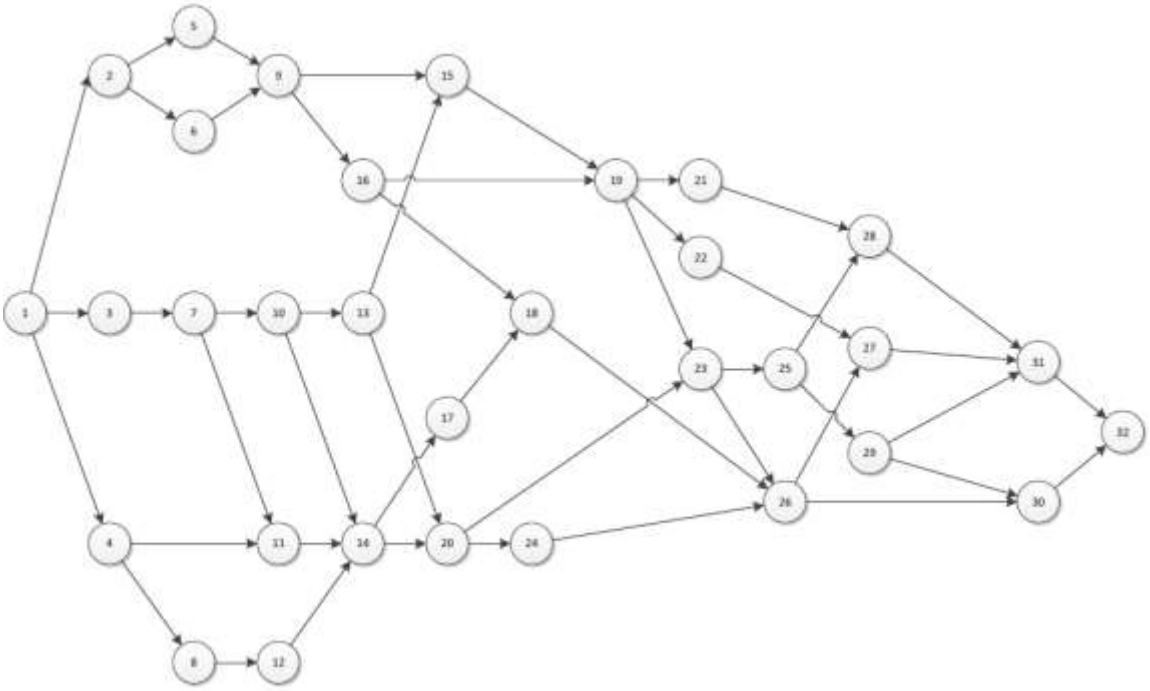


Figure 11: A precedence-dominated network structure typical for construction, aircraft, and submarine work period projects (source: Croteau (2015) [19]*)*

The network topology of NSWPP maintenance projects are however closer to the illustration shown in figure 12, where there is little precedence between WOs, but the WOs do interact with each other by competing for the same labour, compartment, and

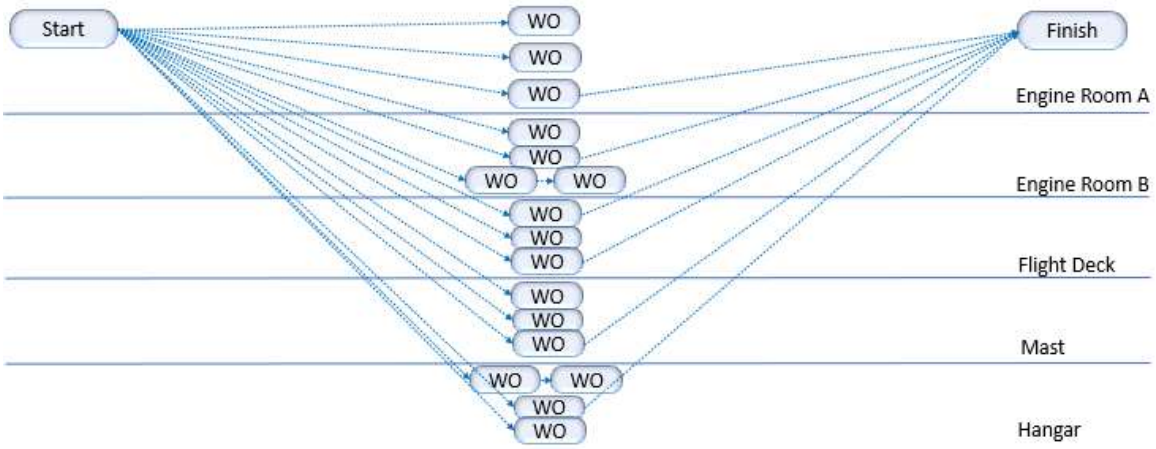


Figure 12: A resource-dominated network topology common to NSWPPs.

radiation/emission space resources. The use of a "dummy" finish node is not necessary because not all activities need to be completed or are even expected to be completed, but these may be placed to show WOs that are ESS or mandatory.

To accommodate for this unique network structure, the experimentation of scheduling methodologies in this thesis are performed using similar network topologies between WOs, as well as with varying priority levels that respect the priority conditions described earlier.

2.7 Contractor Assessment Criteria

Another aspect considered in this thesis is the primary contractor's performance assessment criteria. For example, as part of Thales' AJISS (AJISS: Arctic Offshore Patrol Ship (AOPS) Joint-Support Ship (JSS) In-Service Support) contract with Canada, it is recognized that Thales or other ISS contractors will be assessed across a number of factors [20]. Contractor assessment criteria are often not fully defined until a contractor has been performing work for a performance-measurable duration, as new ISS contracts are more commonly embracing "relational contracting", where the details of work to be performed are not fully defined by the Government of Canada until the contract is underway. Canada

requests performance objectives and higher-level outcomes, and the contractor manages the tasks needed to meet the desired performance.

The contract is guaranteed over a five-year rolling wave period, where the contractor's performance is assessed on an annual basis, and the contract is renewed for the next five years (the guaranteed next four years plus an additional year) if the performance is deemed acceptable [20]; thus, giving Industry an opportunity to plan an exit or have a guaranteed long term contract where it can safely hire personnel and invest without the threat of having the contract terminated in a very short term.

One criterion that is related to scheduling is that of meeting the agreed work completion targets for upcoming work periods. This scoring scheme will directly reflect WOs accepted for upcoming work periods, WOs completed from these accepted WOs, and the priority level of these WOs [20]. The exact scoring weights for each priority level are not included in this thesis, but an example of such weights can be: If the contractor accepts 100 priority-1 WOs, 50 priority-2 WOs, and 50 priority-3 WOs, then this becomes the basis of performance assessment for this work period. Each priority-1 WO may be worth 1 point, while priority-2 WOs may be worth 0.3 points and priority-3 WOs may be worth 0.1 points. In this scheme and example, the total project is worth 120 total points. At the end of the project, if all work is completed, the contractor will be given a 120/120 or 100% performance assessment. If only 10 priority-1 jobs are not completed, then the assessment rating will be 110/120 or 91.7%, while if only 10 priority-3 jobs are not completed, then the assessment rating will be 119/120 or 99.2%. It is therefore natural that these types of contract should consider priority.

2.8 The Scheduler's Perspective

For DWPs however, the contractor performing work for Thales such as ISI or Seaspan may not be subject to this scoring criteria and may be contracted to perform a set number of WOs, in the form of particularized maintenance repair specifications (PMRS). As scope grows throughout the project, additional work may be negotiated on a case-by-case basis. There will also likely be situations where performance assessment will

ignore certain WOs that are not completed when this is outside of the contractor's control. For example, if a serious problem is found during a maintenance routine, Thales may negotiate with Canada and propose the options of:

- fixing the new problem found but not completing the original PM routine at no fault to the contractor
- fixing the problem and completing the PM routine at a higher additional cost, or
- fixing both at normal additional costs where the project makespan is extended. How much this makespan needs to be extended and how much additional work is needed, if any, are all questions that could be answered by an automatic scheduling tool, giving quick answers to support decision-making.

A refit contractor may be consulted to negotiate project makespan, what work periods to complete and when, as well as the content of these work period projects. Ideally, a scheduling tool will need to be flexible enough to quickly schedule many variations of planning data and produce robust schedules that give reasonable makespan. A reasonable makespan is one that is long enough to comfortably complete important high priority work and as much lower priority work as possible, with the understanding that the scope will grow throughout the project and delays are inevitable. The scheduling tool needs to be fast so that back-and-forth negotiations and analysis can be performed. It is proposed that long-duration work cannot be overlooked, and priority should be considered. There should be several options for the user to manipulate data to prevent work conflicts, to find answers for negotiations, and for options analysis.

2.9 Research objectives and thesis outline

A research objective for this thesis is to study RCPSP development, current models, and RCPSP variants that could be applicable to the problem context. To encompass the problem completely, the research includes heuristic methodologies to accommodate solutions that used quick solve times to big problems, popular project management and scheduling methodologies that used buffers, and research into

overtime formulations. The literature review is not exhaustive but includes much of the most popular research in these fields.

This thesis is structured such that after the literature review, real-world inspired data is analyzed, and preliminary experimentation is performed to narrow the solutions to practical and useful formulations and heuristics. This is followed by a description of the proposed new formulations and heuristics for the NSWPPs. Finally, experimentation is conducted with these formulations on randomly generated real-world inspired problems to compare results and limitations. An automatic scheduling prototype was developed for project scheduling at FMFCS and user-feedback was collected for incorporation into solution development, and this was included as an additional qualitative experiment set. Prior to concluding the thesis, a description of future research extensions is presented. This work is intended to be complementary with similar research from other research groups sponsored by Thales Canada, for NSWPP optimization development.

Chapter 3: Literature Review

Academic research on resource-constrained project scheduling problems (RCPSPs) is very common. A 1995 survey of RCPSP research by Ozdamar and Ulusoy [21] included over 83 references. 16 years later in 2011, a survey of RCPSP research by Weglarz et. al [22] included 218 related references. Since then, the field has become so populated that surveys focused on more precise aspects of the RCPSP. A survey by Pellerin et al. [23], in 2019 listed 144 references on only the hybrid-metaheuristics for the RCPSP.

The classic RCPSP formulations first developed on the coattails of operations research that saw a huge escalation in its usage during World War II [24], and several extensions and variants have been made over the years. The literature review conducted in this thesis is one that follows these variants and extensions to the current state and includes a review of popular heuristic scheduling techniques that have been developed alongside exact methodologies. This review includes a critical look at closely related theorems, backed by preliminary experimentation. These problems are interesting due to their complexity as they are considered NP-hard, where the difficulty grows exponentially with more activities and resource demand [25] [26]. The purpose of this review is to find the RCPSP formulations and methodologies that would be most applicable to NSWPPs, and to adapt them through innovation to the particularities of this RCPSP variant, with its multiple scenario-based objectives. Due to operational requirements of a NSWPP scheduling tool, it was deemed important to review reactive scheduling research, as the re-scheduling scenario includes different objectives and associated methodologies. Papers on robust RCPSP research are also researched in depth to understand the evolution and current academic standing on methods of improving schedule robustness, meaning improving a schedule's ability to accommodate variations without needing widespread re-scheduling or not meeting a project's objectives. Finally, research of RCPSP overtime usage in formulation development is analyzed and compared to current Canadian NSWPP Industry practice at allocating overtime.

3.1 Prelude to the Classic RCPSP

The classic RCPSP derives from the Critical Path Method (CPM) that is still taught to project managers around the world and forms the foundation of project management [27]. CPM was first introduced by Kelley Jr & Walker in 1959 [24] where it competed with the program evaluation and review technique (PERT), introduced by Malcolm et al. (1959) [28], that used three-point estimates to determine activity durations. The CPM uses precedence relationships to calculate float or slack, early and late start times, early and late finish times, and the critical path which is where management must place its focus and effort (core aspects of project management). CPM alone however does not result in NP-hard problems but is insufficient to schedule projects where resources are scarce. CPM is relatively trivial to compute with modern computers [27].

In the classic RCPSP, when resource scarceness is accounted for, the CPM alone is insufficient. In a naval surface ship work period (NSWPP) like a short-work-period (SWP) for instance, almost all WOs have no predecessors; however, it would be completely impossible to start all WOs on the first day: people, equipment, and materials would have to duplicate themselves and work on top of each other occupying the same physical space. These types of situations produced impractical schedules and thus research in the resource-constrained field shortly followed the CPM. The RCPSP problem as a linear programming model was introduced by Wiest (1963) [29]. Shortly afterward in 1967 and by a need to expand the formulation to practical applications, heuristic approaches were later proposed again by Wiest (1967) [30]. The formulation was then further refined by Pritsker, Waiters, and Wolfe (1969) [31] to produce a relatively efficient discrete-time 0 - 1 linear programming model that generates optimal solutions to reasonably-sized RCPSP problems with enough time.

This author and Mr. Prabhu formulated and tested multiple replications of two formulation variants on randomly generated problems with 100 activities each to compare basic RCPSP formulations for solve times using Gurobi Solver versions 8.1 and 9.0. The disaggregate discrete time precedence-constraint modification proposed by Christofides, Alvarez-Valdez, and Tamarit (1987) [32] did not yield statistically-significant

improvement on solve times on 30 of these problems. Therefore, the discrete-time RCPSP formulation concept is used as the basis of formulation in this thesis. This formulation is presented below. It is a translation of the approach by Prisker et al. (1969) and presented in the format used in Couch (2016) [1].

3.1.1 RCPSP Model: Makespan Minimization

Indices:

1… A + 1 for activities

1…K for resources types

0…H for the time horizon

Sets:

P activity immediate predecessor pairs (*i,j*)

K resources

J activities

Parameters:

H integer, the planning horizon from the project start (0)

A integer, number of activities

d_i and d_j integer, duration of activities *i* and *j*

r_{jk} integer, activity *j* demand for resource type *k*

R_k integer, resource *k* capacity

ES_j integer, earliest start time of activity *j*

LS_j integer, latest start time of activity *j*

Decision Variables:

$$x_{jt} = \begin{cases} 1 & \text{if activity } j \text{ starts at time } t \\ 0 & \text{otherwise} \end{cases}$$

Objective function:

$$minimize \sum_{t=ES_j}^{LS_j} t \times x_{A+1,t} \qquad (1)$$

- Minimizing the dummy end node's finish time *t* (makespan)

Subject to:

$$\sum_{ES_j}^{LS_j} x_{jt} = 1 \qquad \forall\, j \in \boldsymbol{J} \qquad (2)$$

- All activities must be completed once between the activity ES and LS times.

$$\sum_{ES_j}^{LS_j} t \times x_{jt} \geq \sum_{ES_i}^{LS_i} (t + d_i) \times x_{it} \qquad \forall\, j \in \boldsymbol{J}, (i,j) \in \boldsymbol{P} \qquad (3)$$

- A successor's start time must be greater or equal to a predecessor's start time plus its duration.

$$\sum_{j=1}^{J} \sum_{b=\max\{t-d_j+1, ES_j\}}^{\min\{LS_j, t\}} r_{jk} \times x_{jb} \leq R_k \qquad \forall\, k \in \boldsymbol{K}, t \in ES_j \ldots LS_j \qquad (4)$$

- The sum of resource usage for the period of activity start times to activity end times must not exceed the resource capacity for of each resource type *k*.

$$x_{jt} \in \{0,1\} \qquad \forall\, j \in \boldsymbol{J}, t \in ES_j \ldots LS_j \tag{5}$$

- Start times are binary, so a variable exists for every time unit between each activity's ES and LS.

The early start and late start (ES and LS) portions of the above formulation are not needed to formulate these problems. Instead of ES and LS, one may use *0*…*H*. In the real-data inspired random problem generated instances used in our preliminary experimentation, the benefit in solve time by using ES and LS was almost negligible for solution time, since the NSWPP problem has lower overall precedence constraints than most RCPSPs reviewed in academic literature. This confirms observations made by Kolish et al. (1995) [33]. The benefits of calculating ES and LS are that there are fewer binary variables in the problem, but since modern solvers very quickly eliminate unfeasible binary variables from the solution space, it is theorized that the benefit of this pre-calculation was found to be negligible when problems have not many precedence relationships, such as those found in NSWPPs. Preliminary experimentation completed on a problem set provided by Pellerin (2019) [34], where all actual activities had predecessors and successors with tight ES and LS, was found to significantly decrease solve times. This benefit however was only seen when the time horizon H was relatively close to the optimal makespan. If the LS, calculated using a standard backwards pass [27], started from a relatively large time horizon H, then solution times remained relatively similar to that using 0…H, until this time horizon H began to approach the optimal solution. Figure 13 shows these preliminary results using a commercial off-the-shelf laptop (Intel Core i7 vPro, 1 TB SSD, 16 GB RAM).

Despite this observation, all other preliminary and early main experiments (see methodology in results section) were completed with formulations that calculated ES and LS from a generously large time horizon, as leaving the solution space at 0…H never shortened the solve time, and most references reviewed in literature commonly use ES and LS, recognizing the potential benefits of pre-calculating these values, such as in Couch (2016) [1], Croteau (2015) [19], and Tenera (2008) [35]. It was also later found during experimentation with larger problem sets, that although solve time does not increase, the time required for formulation software to generate linear program model files (.lp

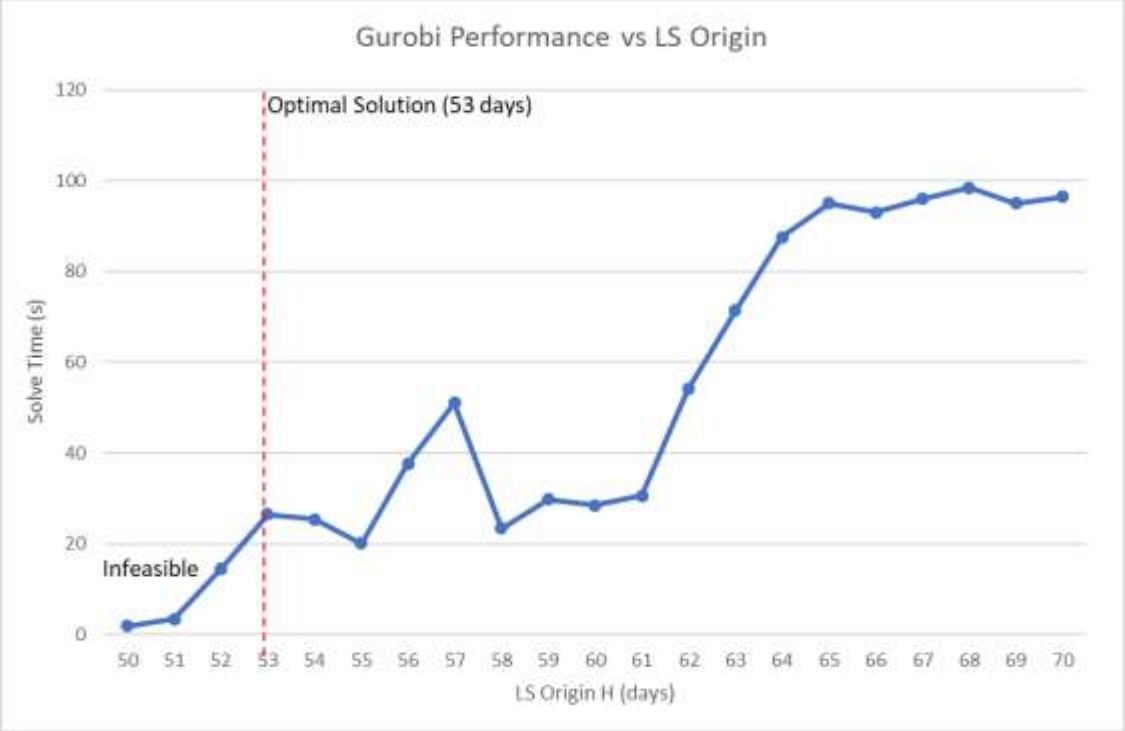


Figure 13: Gurobi software solve times improved as ES and LS calculations were assigned to task start times, but only when LS was calculated from a reasonably tight origin H.

files) decreased significantly with tighter ES-LS windows.

Many papers in RCPSP literature describe methods for determining a suitable time horizon such as using schedule generation schemes (SGSs) [36]. For a practical reference, Appendix A provides programming code in MS Excel VBA that was used for calculating ES and LS. In the code, a serial SGS sorts activities by ES, solves an initial schedule to achieve

a time horizon H, then uses H to perform a backward pass and achieve a reasonably tight ES-LS window for the scheduling activities. It was later found from Gurobi documentation that this heuristically-derived schedule can also be sent to a solver to give it an initial feasible solution, so that if the solver is interrupted due to limited time, then at least a solution can be given [37], as a perfectly optimal solution is rarely needed or desired in practice [3].

The makespan minimization formulation is however unsuitable on its own for NSWPPs. As described earlier, the huge amount of variability and expected scope growth requires all activities, not just those on the critical path, to be completed as soon as possible (front-loading [3]). Preliminary experimentation using minimize makespan formulations with Gurobi optimization solver 9.0 resulted in slack activities to be scheduled close to as late as possible (ALAP). In many industries like the new-construction industry where there is not much scope growth expected, ALAP is preferred because of people's tendency to work less efficiently when they know that they have time to spare [38]. In flexible manufacturing industries, the tendency shifts to what is more common to NSWPPs, a desire to front-load all known work to make room for expected scope growth and new work [39].

Nudstasomboon and Randhawa (1997) [40] recognized this effect in RCPSP literature and suggested the following methodology to reduce slack activity durations:

1. Solve the minimize makespan problem using the classic RCPSP and determine the optimal shortest makespan.
2. Introduce a constraint to ensure this makespan is maintained:

$$\sum_{t=ES_j}^{LS_j} t \times x_{J+1,t} \leq makespan \tag{6}$$

3. Modify the objective function to reduce the start times for all slack activities:

$$minimize \sum_{t=ES_j}^{LS_j} t \times x_{jt} \quad (7)$$

4. Solve the problem again and maintain all activity start times.

There is also the bi-objective solution, mentioned by Nudstasomboon and Randhawa [40], that has one term to minimize makespan and one term to minimize the start times of every activity.

$$minimize \sum_{ES_j}^{LS_j} t \times x_{J+1,t} + \alpha \sum_{j=2}^{J} \sum_{ES_i}^{LS_i} t \times x_{it} \quad (8)$$

Preliminary experimentation of this objective revealed a remarkable 10-fold increase in computation time for the same NSWPP-inspired 100-activity problems, when compared to a single-term minimize start-time formulation (i.e. a function that simply seeks to minimize the start time of each activity), such as using equation 7 as a starting point. From experimentation in the methodology and review sections of this thesis, it was found that using variants of equation (7) as a starting point are preferable in many ways to the classic minimizing makespan for several reasons. Firstly, in NSWPPs, minimizing the start time of all activities undertaken is generally more important than minimizing the overall makespan. The dummy end node is technically a fictitious activity and solver results that do not use a makespan constraint are very practical. These results are practical because, with trivial data processing, one can see which activities are scheduled beyond the desired project end date, indicating that there are problems with either too much scope (too many WOs) or an overly-tight makespan. The results of scheduling using equation 7 as the objective function show underlying problems relating to the assumed project makespan or work that is undertaken without needing an end-node. For example, if a project leader (PL) realizes that after scheduling, a few resource-competing activities fall outside of a project window, then the PL will gain insight in the infeasibility of his/her project and use this information to; suggest alternative execution modes, exclude certain

work from the project, complete partial components of a WO and saving the rest to be completed later, find problems with the underlying data, and may use the results to justify the use of overtime on critical activities. It should be noted, although intuitive, that a project makespan is highly correlated with any function that seeks to schedule all activities as soon as possible.

Anderson (2014) [41] suggests an RCPSP model for submarine maintenance where it modifies the objective function by adding a penalty to the start times for each activity. This thesis is related to naval work with disaggregated WOs, and Anderson recognizes that minimizing the overall makespan directly in a formulation is not required. The objective function is simplified to show the term that reduces slack of all activities by adding a 0.01 weighting penalty for starting activities later:

$$maximize \sum_{t=ES_j}^{LS_j} (1 - 0.01t) \times x_{jt} \qquad (9)$$

Note also that to convert this to a minimization problem, the start time penalty only needs to be reversed, making this identical to equation (7) if the weighting penalty value (0.01) is removed:

$$minimize \sum_{t=ES_j}^{LS_j} 0.01t \times x_{jt} \qquad (10)$$

Although this may be suitable in a submarine work period whereas previously described WOs have significant precedence relationships between them, this will still cause the problem of activity duration not being considered.

There is a considerable flaw in the solutions that are found using the above formulations in NSWPPs. In NSWPPs, where precedence relationships have much less of an effect on outcomes than resource constraints do, there will be many slack activities. The majority of activities will not be on the critical path and most activities will not even have any effect on the critical path except where they compete for resources. Through

preliminary experimentation, it was found that using the objective function from equation 7 produces solutions whereas many short-duration activities as possible become scheduled as early as possible and all the long-duration activities become scheduled later in the project. This is not good in a real NSWPP scenario [3] as longer WOs are generally the most important and complicated WOs. To illustrate this flaw, picture a scenario where completing all activities is infeasible and selecting as many activities as possible is desired by modifying the completion constraint (equation 2) to the equation below:

$$\sum_{ES_j}^{LS_j} x_{jt} \leq 1 \qquad \forall\, j \in \boldsymbol{J} \tag{11}$$

The objective function being considered may not even allow a solver to select longer duration activities at all because completing more short activities earlier has a better impact on the objective function. Figure 14 is an illustration of the solutions provided by this objective function if three WOs with different durations are scheduled in sequence where the WOs have no precedence relationships between them, but must be completed at different times due to limiting resource constraints. This effect is realized because only activity start times are considered and a sum of these constitute the objective function. Placing the longer activity at the project start will prolong the start times of all later slack activities that compete for resources, and thus produce larger objective functions, even though having the longer duration act``vities finished earlier is what is desired in the reality. This objective function therefore insufficiently models the true scheduling goals.

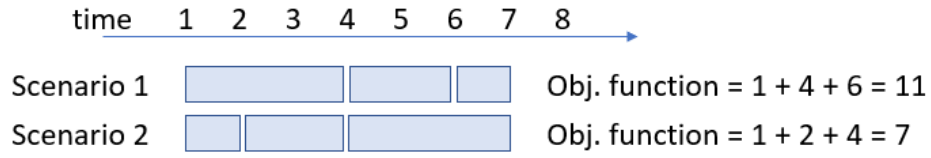


Figure 14: Scenario 1 is preferable because the longer activities are performed earlier, but the objective function that simply minimizes start times produces a better objective function for scenario 2.

It is therefore proposed that weighted activity durations should be considered in the objective function for NSWPPs. To the author's knowledge, there are no RCPSP references that use weighted activity durations in the objective function.

3.2 Formulation Modifications and Suitability with NSWPPs

Improvements to the classic RCPSP come in several forms. Christofides et al. (1987) [32] suggested to convert the discrete time precedence constraint (equation 3) to the disaggregate discrete time (DDT) form. The DDT precedence constraint is shown below:

$$\sum_{ES_j}^{LS_j} t \times x_{jt} + \sum_{b=ES_i}^{\min\{LS_i, t+d_j-1\}} x_{ib} \leq 1 \qquad \forall\, j \in J, (i,j) \in \boldsymbol{P} \qquad (12)$$

Preliminary experimentation conducted by Prabhu (2020) [42] showed no improvement in using the DDT constraint on NSWPP-inspired randomly generated problems, where a 30-set experiment of 100-activity problems solved faster with DT by 71% ± 24% at a 90% confidence interval (C.I.), assuming that the difference between solve times are normally distributed for the same problems.

Continuous-time based formulations exist for the RCPSP, as described by Applegate (1991) [43], then examined and expanded by Alvarez-Valdez (1993) [44], and Artigues et al. (2003) [45]. These require three types of additional decision variables: starting time variables, sequential binary variables determining the sequence of activities, and continuous-flow variables denoting the resource that is transferred between activities.

Other RCPSP variants include the event-based MILP formulations first proposed by Zapata et al. in 2006 [46], the on/off based formulation proposed by Koné et al. (2011) [47], and the ON-only formulations proposed by Croteau (2015) [19].

Preliminary experimentation on the makespan minimization problem were conducted using the state-of-the-art forms of these formulations on NSWPP-inspired fictitious problems and the solution times were unacceptably large. Over 10 problems

were analyzed with 100 activities, and no continuous-time or event-based formulation could finish solving in less than 8 hours, where the DT formulation solved in less than a few minutes. As stated in these continuous-time formulation papers, these formulations were considered better than their DT counterparts only when the work periods are not adequately modelled by discrete time intervals. Perhaps in industries where scheduling to the precise minute or second is practical, such as for computer programs to process routines or in manufacturing plants where increasing machine utilization is of primary importance, these continuous-time methods may be superior to DT formulations (although these can only solve problems with a small number of activities), but NSWPPs are more realistically scheduled with discrete times [3] [4].

The reason for the appropriateness of DT-modelling in NSWPP practice is that workers need time to start any activity: they need to collect tools and go to the work location, they need to inform ship staff (SS) that work is about to start, conditions need to be made safe (lock-out and tag-out), a reasonable amount of time is needed to complete the actual work, then time is needed to collect tools and return to the repair facility (RF) for meals or a shift change. Work completion is not even normally reported until the shop workers have returned to the repair facility (off the ship) and they are able to log-on and close-out their activities. Ideally this would happen immediately upon arrival from the ship, but it can take days at times. What has been gathered from interviews in FMFCS NSWPPs and as requested by PLs and PMs [3], is that a scheduling tool should suggest activity start times either at the start of each day or as precisely as at the start of each shift (i.e. morning, after lunch, and after supper for evening shifts (if an evening shift exists)). Whether there is a one, two, three, or four-hour activity, this will usually take a morning or an afternoon to complete; by the time the right parties are informed that the activity is completed and the next activity is prepared, the morning or afternoon is already expended, as workers do not have access to the ERP system in the ship (an important cyber-security measure). With such discretized time units, the DT or DDT formulations reach an optimal solution for the same problems much faster than with continuous time models. In cases examined in preliminary research, problems with 60

activities could be solved in seconds with DT and DDT models, while the continuous time models would not solve to optimality after hours of computation. To illustrate the value of having relatively larger time steps in DT and DDT models, if this work were planned such that the binary DT start-time variable x_{jt} existed for every single second in a 20-day work period with eight-hour shifts, then there would be up to 576,000 x_{jt} variables per activity. In the same illustration, by scheduling to the half-day, there are only 40 variables per activity; thus, reducing the number of binary variables by several orders of magnitude. This DT practicality was observed at FMFCS/CB, ISI, and at SNC Lavalin in dockyard work periods (SWPs and EWPs). It should be noted that Seaspan in Victoria produced a schedule from Primavera P6 showing work for a 24/7 short work period where a merchant ship required a large volume of work in only eight days. In this case, work was scheduled down to the half-hour level-of-precision, but interviewed staff indicated that the schedule was ever only loosely followed, and decisions by floor managers made throughout the work period dictated what work would progress at what time, as conflicts, duration variations, and scope growth emerged. Additionally, this project used strict precedence CPM relationships and did not consider a constrained resource pool [48]. This application of the scheduling ERP system was only used in practice to create a visual display (i.e. Gantt Chart) as no resource information, including capacities, costs, etc. were used.

Despite the simplifying nature of using DT units with a significant difference between each time step (hours to days), some real-world problems analyzed for this thesis were still too large and complex to solve to optimality in a reasonable amount of time with exact formulations and MILP solvers. Experimentation described later in this thesis shows that even the best DT formulation with exact solving methodologies are unsuitable for solving entire longer-duration NSWPP projects directly, as these projects often involve well over a few thousand activities and many data-mistakes are present, requiring the schedule to be re-solved quickly and frequently to find and correct these data mistakes iteratively. Thus, to effectively research the NSWPP problem, heuristic techniques will also be considered.

3.3 Variants and Extensions

Many papers have been written that propose RCPSP formulations with different objectives. Möhring (1984) [49] first suggested the resource availability cost problem (RACP), where the objective is to minimize the maximum per-period resource and consumption cost for each individual resource. Several other references evaluate, elaborate, and expand this problem. Coelho and Vanhoucke (2011) [50] proposed multi-mode formulations with various execution modes. Schutt et al. (2013) [51] proposed lazy-precedence clause formulations that seek to level resource usage. Schwindt and Zimmermann (2015) [52] built a 59-item table of all the RCPSP variants and objective functions. These were found to be either single or multi-objective, seeking to improve makespan, cost, resource usage, and quality. Many references seek objectives such as time-cost tradeoffs [53], resource leveling [54], net-present value objectives [55], capacity planning [56], and minimizing project risk [57].

Schedule quality and minimizing project risk, and its practical application, is closest to this thesis' objective of proposing effective methodologies, heuristics, and formulations for NSWPP scheduling and re-scheduling problems. Of note with respect to NSWPPs is the *RCPSP-t*, where studies are focused on the particularities of resource availabilities that change over time, first described in Hartmann (2012) [58]. This is interesting because the real-world NSWPP projects reviewed do in-fact often have worker availability tables that show shop capacities changing over time. For instance, at FMFCS a weekly report is derived from the ERP system DRMIS, showing how many hours of availability remain for each of the facility's 70 shops for the next 14 weeks. When data is entered as designed in their standard operating processes, worker leave intentions and holidays are entered into the ERP, and a good approximation of resource availability can be derived. PLs can then schedule their work, although manually, by first reviewing shop capacity information for each activity in a WO, before requesting dates to the schedulers. Shops availabilities are then updated in the ERP as work is scheduled. Access and use of this varying resource availability information would be essential in developing effective

automatic scheduling for organizations that track and use worker availability in their planning and scheduling.

An interviewed Thales project manager confirmed that project risk was her top objective when devising scheduling methodologies for company projects [2]. Robust RCPSP methodologies and formulations were therefore reviewed for this thesis. These include references such as Couch (2016) [1] where ideally placed project buffers were used and Van de Vonder (2006) [59] where strategies are suggested for flexible manufacturing plants. However, none were found to be ideally suitable in whole for the NSWPP RCPSP in the current Canadian Industry state due to specific problems (e.g., human factors, Student Syndrome) that will be further discussed in the next chapter.

3.4 Heuristic Scheduling Techniques

As previously mentioned, since RCPSPs are NP-hard and exact formulations are often unsuitable to handle real-world NSWPP problems [26], a literary review of heuristic techniques was conducted. These heuristics have been necessary in developing RCPSP solutions to real-world problems since the early 1960s [52]. In the late 1960s, famous Microsoft/IBM CEO and philanthropist Bill Gates and his friend John Allen were successful in selling heuristic RCPSP computer codes for his high school and several companies state-wide [60], showing the value and practical relevance of heuristic scheduling methods when the relatively weak computers of the era would have been unable to find optimal solutions to these problems. The backbone of heuristic techniques are the serial schedule generation schemes (serial SGS) and the parallel schedule generation schemes (parallel SGS), and they are generalizations of “list scheduling” from machine scheduling [61].

It is common for the serial SGS to be used to find heuristic solutions to RCPSPs [62]. These are set according to priority rules that are common to most commercial scheduling systems [63]. A closer examination of the serial SGS reveals that it accomplishes almost exactly what human schedulers accomplish when scheduling all WOs for a NSWPP project when working from a prioritized top-to-bottom list. A PL may assign all WOs in a prioritized list and schedule them one at a time, working down the list,

scheduling them as early as possible where the work is precedence and resource feasible. This requires a series of verifications at each time step until resources are available for the entire length of the activity. The serial SGS result may contain the optimal solution to any RCPSP if the order is correct and this has been demonstrated by Sprecher (1995) [64]. However, the parallel SGS may never contain the optimal solution, which may be considered a severe drawback [65].

For the serial SGS to function, early start (ES) calculations that consider only precedence relationships and activity durations are computed using the classic forward pass calculations described by Demeulemeester and Herroelen (2002) [66]. With this and relatively simple data processing, tasks may be ordered such that predecessors are always scheduled before successors. After each activity is scheduled, the precedence-related early start of the next unscheduled activity is calculated, and this becomes the first timestep to examine feasibility for this next activity. If the next activity cannot be scheduled at the ES time due to resource unfeasibility, the start time is right shifted by one time unit and it is then verified again for feasibility. The pseudo-code to schedule n activities for each activity i with scheduled start time a_i, respecting the early start constraints of each activity (ES_i), as well as the resource demand for each type m and activity k_{mi} may be summarized as follows (Visual Basic-inspired format):

1. **Do While:** each activity i , from 0… n. First_activity.Offset(i,0) <> “” ‘Keep looping through all activities
    a. **Initialize:** $start\ time\ t_i = ES_i$
    b. **Do While** $available\ k_m - k_{mi} \leq 0\ \in \{k_m \cup i\}, \{t_i \dots t_i + d_i\}$ ‘check if activity is resource unfeasible for all resource types between the start time and the start time plus its duration
        i. $t_i = t_i + 1$ ‘increment start time if unfeasible
    c. Loop ‘loop ends when activity is feasible
    d. $a_i = t_i$ ‘update activity start time
    e. $new\ k_m = old\ k_{mi} \in \{k_m \cup i\}, \{a_i \dots a_i + d_i\} \quad - k_{mi} \in \{k_m \cup i\}, \{a_i \dots a_i + d_i\}$

'calculate and update resources, may use a resource table or matrix for each resource type for entire duration of activity

f. Recalculate: ES_{i+1} 'Update the ES for the next activity

g. $i = i + 1$ 'increment the activity number

2. Loop

This is pseudo-code is implemented in VBA (see Appendix A) following the format described by Kolisch and Hartmann (1999) [65], and by Rostami et al. (2017) [36]. Some improvements to the serial SGS can be made to improve computation time. For instance, it was found with experimentation that a *GoTo* line of code, used to abandon a check in 1.b when infeasibility has been found, reduces computation time by approximately 50%. Kolisch and Hartmann (1999) [65] also recommended checking only the last time unit of a scheduling time solution for available resource capacity instead of at every time unit for an initial pass, and thus this improved the serial SGS solution time used for experimentation in this thesis by an additional 10%.

The computation time for a serial SGS is also generally faster than that of using exact solvers except where the problem is very small, as will be seen later, and the increase in computation time grows more-or-less linearly with additional activities, rather than exponentially as seen for the NP-hard problem of finding and confirming optimality in RCPSPs [26].

The other main SGS is the parallel schedule generation scheme (parallel SGS). This SGS is very popular in job-shop and machine scheduling, and variants of it such as the shifting bottleneck procedure are found to produce schedules that are very practical [67]. One downside of parallel SGSs are that they generally require more computation than the serial SGS and the typical scheduler does not readily understand how the scheme achieves results [3]. The parallel SGS is similar to the serial SGS in that it goes top to bottom through a list of activities, but it considers only a single timestep at each pass through the list. It attempts to schedule as many activities as possible at each time step, until all activities

are finished. The pseudo-code for the parallel SGS adapted from [65] and [36], is as follows:

1. **Initialize:** t=0, all activities unmarked as "scheduled"
2. **Do While:** any activity remains unmarked as "scheduled"
    a. **For:** each activity i, from 0… n. First_activity.Offset(i,0) <> "" 'Loop through entire list of activities i
        i. **If:** $start\ time\ ES_i \geq t$ **Then**
            1. **If:** $available\ k_m - k_{mi} \geq 0\ \in \{k_m \cup i\}, \{t \dots t + d_i\}$ **Then** 'check if activity is resource feasible for all resource types between the start time and the start time plus its duration
                a. **Schedule** $a_i = t$
                b. $new\ k_m = old\ k_{mi} \in \{k_m \cup i\}, \{a_i \dots a_i + d_i\} - k_{mi} \in \{k_m \cup i\}, \{a_i \dots a_i + d_i\}$ 'calculate and update resources, may use a resource table or matrix for each resource type for entire duration of activity
                c. **Mark** activity as "scheduled"
                d. **Recalculate** ES_i for next unscheduled activity (only)
            2. End If
        ii. End If
    b. Next i
    c. $t = t + 1$ 'increment time
3. **Loop**

Both the serial SGS and parallel SGS were used for experimentation in this thesis to compare with exact methodologies and to act as a basis of design for heuristic modifications, making these applicable to larger NSWPP RCPSPs where exact methodologies may not solve the problem in a reasonable time.

Many improvements to these SGSs have been found in literature such as by Stork (2001) [68], adding a priority constraint; thus, preventing an activity in the serial SGS to be scheduled if higher priority activities are not yet scheduled. Stork (2001) [68] does not explain why these activities cannot be first sorted by priority to improve computation time. Several priority rules are summarized by Alvarez-Valdez and Tamarit (1989) [69], such as the latest start time (LST), latest finish time (LFT), earliest finish time (EFT), longest processing time (LPT), activity number (AN), and select jobs randomly (RAN) rules, but the only priority rule considered to be adequate for NSWPPs is the earliest baseline activity starting time (EBST) rule, where activities are started as soon as possible, for the reasons described in the problem definition section of this thesis, relating to expected scope growth and uncertainty. Van de Vonder, Ballestin, and Demeulemeester (2007) [70] propose additional more complex priority rules that outperform these heuristic procedures, but these are stated as being more applicable to job-shop scheduling problems in the reactive scheduling scenarios.

Numerous other papers have been published that use genetic and evolutionary algorithms to sample intelligently through a large number of feasible scenarios that can be computed by serial or parallel SGSs: Mendes (2003) [71], Gonçalves (2014) [72], and Rostami (2017) [36]. More complex methodologies may combine these algorithms with additional search rules, often called metaheuristics, to deeply explore promising solution space areas, with a greater likelihood of finding high quality solutions at the cost of computation time [23]. Metaheuristic techniques for RCPSPs include: scatter search methods Berthaut et al. (2018) [73], adaptive large neighborhood search algorithms Muller (2009) [74], artificial immune system based approaches Agarwal, Colak, and Erenguc (2011) [75], tabu search methods and simulated annealing Bukata (2015) [76], particle swarm techniques Chen (2011) [77], filter-and-fan approaches as proposed by He and Chen (2016) [78], artificial bee colony algorithm approaches Nouri et al. (2013) [79], self-learning strategies Jędrzejowicz and Ratajczak-Ropel (2014) [80], and many others.

A fundamental similarity with these metaheuristics is that they are very frequently compared against each other using the famous PSPLIP library developed by Kolisch and

Sprecher in 1996 [18]. These problems include only a few resources, most activities have one or more precedence constraints, and are heavily precedence-dominated as previously discussed. None of them have the network characteristics associated with NSWPPs and these problems only go up to 120 activities. With preliminary experimentations, it was found that NSWPP-inspired problems may take well over 50 times longer to solve than PSPLIB problems with the same number of activities, even with serial SGS methods, as the solution space is much larger. The hybrid metaheuristic methods search solutions and compare results to find the best solutions among a large list of feasible schedules (several thousands), but the time required to compute even one single iteration of a real-world schedule currently is often too long for a large number of iterations to be practical, as discussed in the problem definition of this thesis. The focus in this thesis has therefore not been on these metaheuristics, because the long overall solution times needed are impractical.

3.5 Using Buffers to Produce Robust Projects

As described in Schwindt's 2015 Handbook on Project Management and Scheduling: "robust project scheduling is concerned with the problem of finding a predictive baseline schedule that still performs well in case of disruptions or adverse scenarios" [52]. This is indeed what is needed for NSWPP scheduling.

Using buffers to produce solution robust RCPSP schedules have been studied significantly in literature. It has been anecdotally indicated that in normal practice, planning staff introduce buffers at the activity level, and additional buffers would negatively reflect on management staff [3]. The use of buffers has been more formally introduced in literature with Goldratt's 1997 book "Critical Chain" [81], where using buffers at the end of the critical path or "feeding paths" is part of his CCPM system. This CCPM system includes using a large project buffer that becomes consumed throughout the project. If the consumption rate is too quick, then management must intervene. It suggests absolutely no multitasking and an as-late-as-possible (ALAP) scheduling methodology. It requires tasks to be estimated using median time estimates rather than the safer 80-90% time estimates that are supposedly traditionally used. From analysis of

his work, including follow-up papers by Leach (2000) [38], we do not recommend that the CCPM method be fully used in NSWPPs. The ALAP methodology does not perform well when significant scope growth is expected, but some concepts such as the consideration that Student Syndrome is always present and that buffers are important were considered. CCPM proposes that the concept of leaving float (slack) in individual activities is bad because of Parkinson's Law or Students' Syndrome, where activities that could have finished sooner tend to take as long as the time that is given, due to procrastination and a slower naturally-induced pace intended to keep employees busy [82]. Analysis of real-world data seen in later sections of this thesis indicates that Student's Syndrome affects NSWPPs as well.

CCPM also requires activities to be estimated using median time estimates. This means that for half of the activities, it will take longer than estimated to finish without including scope growth. This is not a normal way of estimating activity durations for most traditional project planners. According to CCPM theory, people tend to give an estimate where work will be completed within the estimated time between 80 and 90% of the time. In the case of a sub-contractor that estimates duration and is paid on an hourly rate, a median time estimate would severely hurt his/her business and would often result in working at a possible loss. Several others, such as Raz (2003) [83] and Trietsch (2005) [84]

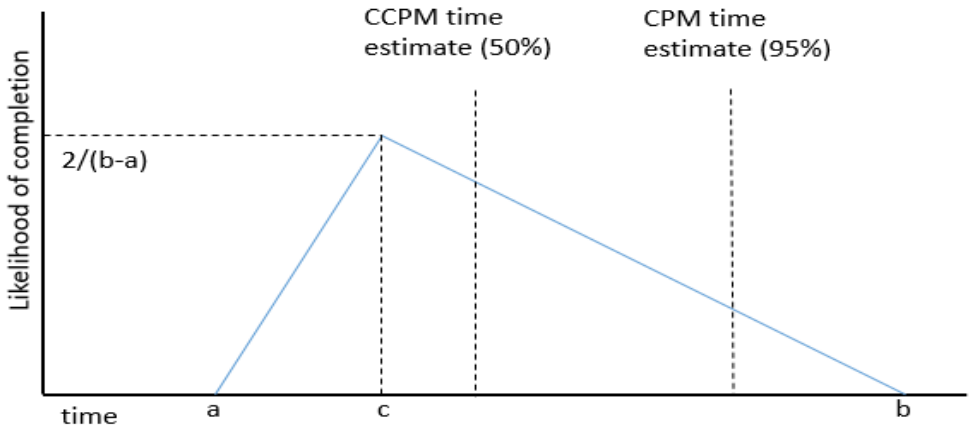


Figure 15: Common triangular distribution with three-point estimates.

have criticized CCPM concepts as being unrealistic in many environments. An illustration of these time estimates is provided in the following section in figure 15.

3.5.1 Illustrating CCPM vs CPM time estimates

In the CCPM technique, the scheduler must assign durations relating to the median completion time, or when in 50% of cases, a job would be completed. For example, if the only information available for a job was a 3-point estimate (best-case, a most likely, and a worst-case), then the probability distribution for this job could be modeled with a triangular distribution:

The CPM time estimate will be:

$$x = b - \sqrt{(0.90)(b-a)(b-c)} \qquad (13)$$

While the CCPM time estimate will be:

$$x = b - \sqrt{(0.50)(b-a)(b-c)} \qquad (14)$$

One major issue with either approach is that a 3-point estimate must be provided by a planner or scheduler. Significant bias is often introduced in this estimate as the scheduler will not often be the one estimating the job duration [3]. Most subcontractors will not typically provide a quality 3-point estimate but will tend towards providing a conservative time estimate that more closely matches the CPM 80%-90% confidence time estimate. Getting realistic minimum and maximum values is very difficult, and many workers will tend to exaggerate these numbers once they realize the implications. People do not like to be stressed or have to work in an environment where a small mistake like forgetting a tool will result in lateness [4] [83] [84]. This CCPM process may face even more difficult opposition in a unionized environment as complaints may arise from these more stressful time estimates or may cause worker dissatisfaction in a non-unionized environment where the best workers may seek employment elsewhere.

3.5.2 Buffer Sizing and Placement

Many papers have been written on predicting the best buffer sizes for use in CCPM. Although from interviews with project management staff personnel at Thales and FMFCS [3] [4], it is not believed that CCPM in whole can be directly applied to NSWPPs, because a project buffer gives the impression of poor management since activities cannot happen as planned. Moreover, the RCN is constantly seeking more time windows and opportunities to perform missions, and a project buffer would be quickly removed from a previously allocated makespan to perform additional missions. The RCN expects work to be scheduled for the entire duration of a project time window and a transformational change outside the scope of this research would be needed. It is however theorized in this thesis that buffering of the critical path where the activities are of high priority is beneficial, as will be seen from a simple experimentation with simulation. Goldratt (1997) [81] first suggested to use a "cut-and-paste" buffer strategy, where the size of the buffer should be 50% of a project's duration. This has however been criticized in later papers as being too simplistic and not relevant in all cases. For example, if a project lasted one year using deterministic median time estimates, then the buffer needed after this year would be an additional six months making the buffer 50% of the deterministic project duration, and this would be unrealistic in most projects or considered excessive.

The Root-Square Error Method (RSEM) described by Leach (2005) [85] equates the project buffer as the square-root of the sum of standard deviations in activity duration for each activity σ_i:

$$Buffer = \sqrt{\sum_i {\sigma_i}^2} \tag{15}$$

Leach (2005) [85] also suggested an addition to the buffering equation to account for scheduler bias, where it was found that some schedulers induce a bias throughout all their estimates, updating this equation to add the bias term α:

$$Buffer = \alpha + \sqrt{\sum_i {\sigma_i}^2} \quad (16)$$

Tukel (2006) [86] proposed the Adaptive Procedure with Density (ADP) where the buffer size is calculated in a similar manner, where $\sum x_{ij}$ represents the total number of precedence relationships and $\sum x_j$ represents the total number of tasks:

$$\alpha + \ (1 + max\left\{\frac{\sum x_{ij}}{\sum x_j}\right\}) \sqrt{\sum_i {\sigma_i}^2} \quad (17)$$

Tukel (2006) [86] also proposed the Adaptive Procedure with Resource Tightness (APRT), where $\sum x_{ij}$ represents the total usage of each resource and $\sum x_j$ represents the total availability of this resource.

These methods however all require either 3-point estimates (at a minimum) or the duration distribution of tasks that may give a known deviation σ_i for each activity. Unfortunately, in the naval maintenance industry, information on these distributions is very difficult to come by, and no case studies reviewed or interviews indicated the use of stochastic duration estimates [2] [3] [4] [87] [88] [89].

Interviewees indicated that deliberately placing a large buffer in a Canadian NSWPP project would be viewed as a sign of poor management, poor planning practices, and inexperience [3] [4]. The potential clients for this research indicated that the option to use buffers at the end of sequences made sense and the option to use it could be beneficial in some circumstances, but this would be of the cut and paste method, and would probably never be used for all work as project plans must be submitted to RCN managers and SS, who would naturally push for more work into project makespans or push for shorter project makespans. Despite this, the use of buffers associated with priority levels were reviewed in this research to accommodate clients and provide better solution robustness for higher priority level work, at the cost of poor solution robustness at the lower priority levels. Even without purposely adding buffers, scheduling methods

that consider priority may naturally induce buffers at different priority levels, as will be seen later in this thesis.

Some research suggested strategically placing buffers at various points along the critical path or after important activities; however, these require the assumption that the buffer itself will not affect the work duration distribution. Simulations of projects with duration distributions for each task demonstrate that placing buffers throughout a project gives good results in terms of minimizing deviations (Grey, 2007 [90] and Saihjpal and Singh, 2012 [91]). In this thesis, an analysis of real-world data and confirmation by industry schedulers, as well as points discussed by Leach and Goldratt [82] [38] [92], reveal that individual activity buffers themselves causes workers to work slower and take up the time that is given when it is feasible to do so, thus modifying the underlying duration distribution. Placing buffers after activities are actually already done in CPM time estimating and increasing these is expected to lead to more Student Syndrome slow downs [92]. Analysis of manually scheduled FMFCS data will show in this thesis that some work is occasionally scheduled to take as long as 10 times the estimated work hours, with a hope that the over-tasked shop workers can complete the work within the extra-buffered time window. This thesis adopts the view of interviewed staff and Goldratt, that a single buffer at the end of a project is easier to manage and the final important tasks can be more closely followed by management, thus reducing the potential for slowdowns caused by the buffers themselves [3] [81].

The research on buffers is closely tied however to the reactive scheduling scenario. It would be impossible to properly model NSWPPs without considering reactive scheduling for the reasons described in the problem statement. The purpose of buffers, whether purposely placed or naturally induced by a scheduling methodology, is to handle variability in task durations and changes in scope that occur throughout the execution of a project.

3.6 Reactive Scheduling

The re-scheduling scenario differs from the normal scheduling scenario for several reasons. Firstly, the user's objectives may not be the same as the ones in the initial scheduling scenario. During project execution, the tasks have already been coordinated by the project managers, resources have been assigned to tasks, and dates have been communicated to workers. In a sub-contracting framework, such as the one with a general contractor managing a work period, where the general contractor does not have complete control over the resources, adhering to previously agreed start dates or time windows for work is highly desirable. For example, a previously agreed upon start time for a certain sub-contractor may not easily be moved to another date due to prior commitments. These changes occur very frequently in NSWPPs [3]; however, reducing these changes as much as possible produces a less hectic work period with more predictable outcomes. In the framework where the manager controls the resources in large maintenance plants such as FMFCS, if the workers are constantly given different activity start dates, they tend to lose faith in the management system and will often simply pick whatever jobs are available to work on, or whatever they want to work on [3]; thus, leading to lower schedule adherence.

In this case, and as suggested by Van De Vonder (2006, 2007) [59] [70], the objective may be to reduce changes to the previous schedule as much as possible. That is, the objective is to minimize absolute new start time changes when compared to the previous schedule start times. Mathematically, this is expressed as the following deviation-days objective:

$$Minimize: \sum_{j \in J} \sum_{t \in H} \left| x_{jt} t - s_j^0 \right| \qquad (18)$$

Where x_{jt} represents the binary start times of each job *j*, *t* is the start time, and s_j^0 is the last scheduled start time for each job *j*.

The first dedicated studies on reactive scheduling with similar objectives are those of Szelke and Kerr (1994) [93]and Smith (1995) [94], although these were deemed

applicable to flexible manufacturing plants rather than RCPSPs, but the application in RCPSPs are mentioned. Since then, a large number of papers have been written on reactive scheduling such as reactive-proactive scheduling methods based on cloud computing by Norroddin and Faramarz (2018) [95] with an emphasis on computer task scheduling, where the processing times for same jobs are captured and used to build distributions over time, leading to schedules that can adapt and gradually improve. Deblaere, Demeulemeester, and Herroelen (2011) [96], proposed reactive scheduling policies to deal with stochastic activity durations in RCPSPs using the aforementioned deviation-days scoring criteria with a cost function assigning penalties and bonuses to early or late finishes.

Chaari et al. (2014) [97] described in their survey many reactive scheduling approaches in "highly perturbed environments, where the uncertainties are both frequent and large… and decisions must be made in near real-time". This relates to NSWPPs but the research in this environment is often focused on flexible manufacturing plant applications. Several reactive approaches include the *distributed* approach, where scheduling of parts of a system may be made by an autonomous agent to locally repair a schedule (see Renna, 2010 [98]): this decomposes the problem into smaller localized problems, leading to achieving quick good solutions, although not optimal overall. This is analogous to the NSWPP scheduling systems where resources on multiple projects come from a centralized pool, but each ship project is managed by a dedicated PL, such as is the case with Canadian Industry partners and the FMFs. Another approach is the *centralized* approach that uses meta-rules built from expert system knowledge, such as those found by Sun et al. (1994) [99]. The *priority rules* approach, like the ones surveyed by Rajendran and Holthaus (1999) [100], use various implementations of SGSs and schedule activities one-by-one using special sorting techniques and rules. Finally, there are *dynamic choice of priority rules* approaches, in Mouelhi and Pierreval (2010) [101] for example, that use combinations of simulation and artificial intelligence to come up with reactive scheduling solutions.

It is proposed in this thesis that NSWPP RCPSPs fall in between traditional RCPSP research and flexible manufacturing plant research where reactive scheduling systems research is dominant. In both initial scheduling and in re-scheduling however, there is a growing body of research dedicated to determining the optimal amounts and types of overtime.

3.7 Overtime Research in Initial Scheduling

As will be discussed in further detail later, despite the best formulations to re-schedule while minimizing delays, situations may arise where the reactive schedule places important work outside of a project's makespan or there are too many affected activities that now need to be re-scheduled. When this occurs, overtime may be used to recover a schedule or to ensure important work is completed. Overtime research has been therefore studied in this thesis as it is a common theme in recent RCPSP literature.

To the NSWPP RCPSP, the most direct application of overtime research is in the form of multiple activity processing modes. These processing modes may result in activity duration modes that include overtime or different execution modes. An example of overtime would be one where work normally occurs Monday to Friday between 8h00 and 16h00. An overtime shift may be applied between 16h00 and 24h00 to decrease an activity duration from three days to two days. In this case, mode one has the activity as three days duration while mode two has the activity at two days duration. There is a time-cost trade-off as mode two involves a higher hourly wage for workers performing the overtime. This type of RCPSP is called the discrete time-cost tradeoff problem (DTCTP) and has been studied by many researchers, such as Herroelen (2005) [102], Guldemond, Hubrink, and Paulus (2006) [103], Weglarz et al. (2011) [22], and Schnabel, Kellenbrink, and Helber (2018) [104]. These formulations use multiple modes for each activity as previously described and can be summarized in the discrete-time format used in this paper as presented below.

3.7.1 RCPSP Model with Overtime or DTCTP

Indices:

1… A + 1 for activities

1…K for resources types

0…H for the time horizon

1…M for varying execution modes

Sets:

P activity immediate predecessor pairs (*i,j*)

K resources

A activities

M overtime modes

Parameters:

H integer, the planning horizon from the project start (0)

A integer, number of activities

d_{im} and d_{jm} integer, duration of activities *i* and *j* for each mode *m*

r_{jk}integer, activity *j* demand for resource type *k*

$R_k$integer, resource *k* capacity

ES_j integer, earliest start time of activity *j*

LS_j integer, latest start time of activity *j*

O_j positive real number, overtime cost for one discrete time unit for activity *j*

O_k positive real number, overtime cost for one discrete time resource availability increase for resource *k*

Variables:

$$x_{jtm} = \begin{cases} 1 & \text{if activity } j \text{ starts at time } t \text{ in mode } m \\ 0 & \text{otherwise} \end{cases}$$

z_{kt} integer, additional resource capacity for a resource type k at time t. This may represent overtime or temporary hires.

Objective function:

$$minimize \sum_{j \in J} \sum_{t=ES_j}^{LS_j} (x_{jt2} O_j + 2x_{jt3} O_j) + \sum_{k \in K} \sum_{t=ES_j}^{LS_j} z_{kt}\, O_k \qquad (19)$$

- Minimizing the activities executed in modes 2 and 3. More modes may be used of course.

Subject to:

$$\sum_{m \in M} \sum_{ES_j}^{LS_j} x_{jtm} = 1 \qquad \forall\, j \in \boldsymbol{J} \qquad (20)$$

- All activities must be completed once between the activity ES and LS times, but in only one unique mode.

$$\sum_{m \in M} \sum_{ES_j}^{LS_j} t \times x_{jtm} \geq \sum_{m \in M} \sum_{ES_i}^{LS_i} (t + d_{im}) \times x_{itm} \qquad \forall\, j \in J, (i,j) \in \boldsymbol{P} \qquad (21)$$

- A successor's start time must be greater or equal to a predecessor's start time plus its duration.

$$\sum_{m \in M} \sum_{j \in J} \sum_{b=\max\{t-d_j+1, ES_j\}}^{\min\{LS_j, t\}} r_{jk} \times x_{jbm} \leq R_k + z_{kt} \qquad \forall\, k \in \boldsymbol{K}, t \in ES_j \ldots LS_j \qquad (22)$$

- The sum of resource usage for the period of activity start times to activity end times must not exceed the resource capacity for of each resource type *k*, where z_{kt} represents additional resource availability in period t, brought-on by overtime.

$$\sum_{t=ES_j}^{LS_j} t \times x_{A+1,t} \leq makespan \tag{23}$$

- Enforce the project makespan by ensuring that the last dummy activity finishes before the makespan.

$$x_{jtm} \in \{0,1\} \tag{24}$$

$$z_{kt} \geq 0 \tag{25}$$

It should be noted that this formulation includes two main additions to the classic RCPSP, one where overtime is used to reduce activity duration (left side of equation 19 and equation 21) and one where overtime is used to increase resource availability (right side of equation 19 and equation 22). Depending on the practical application, this can be interpreted differently. In a case where work occurs 24-hours a day, then the reduce-duration parts of this formulation are not applicable, as there are no real-world windows to use more time outside of the set discrete time units to reduce an activity; however, additional capacity at specific times may be modeled by z_{kt} representing temporary hires or extra workers from a different shift using overtime. Where work occurs in once-a-day discrete time units such as at the FMFs (one standard 8-hour shift per day), then both the duration reduction from overtime and the increased capacity may be used to simulate overtime. Solutions for activity start time decision variable (x_{itm}) in modes m≥2 represent activity execution modes that use overtime, where overtime represents an evening or night shift, and z_{kt} represents overtime as well, but as increased resource capacity.

The reason that both may be needed in an 8-hour shift once-a-day regime such as is the case at the FMFs, is that sometimes a makespan can be achieved by reducing certain activity durations. On the other hand, sometimes a makespan can be respected by scheduling a job earlier, where it would not otherwise be scheduled to start, thus having a certain resource work overtime during a period, effectively increasing that resource's capacity. Both methods are found in literature and may include new hires, investments in training, and temporary hires; see and Schnabel, Kellenbrink, and Helber (2018) [104] for more information. A recent paper by Beljadid et al. (2019) [105] proposes a multi-objective overtime formulation that combines duration reduction to meet a makespan while introducing resource-leveling weighting factors.

3.7.2 Notes on overtime and pricing in real-world NSWPPs

It should be noted however that the reviewed overtime formulations research suffers from a real-world problem when applying this to the NSWPP: it requires a lot of information that is not normally gathered or provided, and the real-world is not accurately modeled by simplifying assumptions for these. For example, activity durations may not always be reduced by overtime, such as paint drying or work that can only be accomplished during the day. Any aloft work (working at heights where the danger of falling is significant) cannot be performed in the dark, so an evening or nighttime shift is not applicable. True worker overtime availability is often not uniform or easy to estimate. At the FMFs or in the reviewed cases with industry partners, workers are often not forced to work overtime. Overtime may be requested, but they need to agree on dates on a case-by-case basis. Due to numerous reasons, workers may request time off instead of extra money in exchange for overtime. Time off would be in the form of extra days off illustrated by the following example that is common at the FMFs: workers may not need extra money, but would be willing to work an evening shift Wednesday, in lieu of working normal shifts Thursday and Friday, thus giving them an extra long weekend. Since overtime is often paid at 200% value compared to regular time, time off is also given at a rate of 2:1, which may negatively affect other parts of the project, or other projects for other ships [3].

The data that is inputted, such as calendar duration of WOs and resources needed, have a very large margin of error (often as much as 100%), leading to an indication that simple metrics to calculate overtime in advance is impractical. When overtime is needed, it is usually done with no more than a few days notice in anticipation of an important post-project milestone such as sailing or flooding a dry-dock. Much more than the normal work resources may be involved. For example, on a manned ship, the ship staff (SS) are required to isolate/de-isolate systems, operate equipment, etc., for the repair facility workers; this in turn requires short-term negotiations with the ship's company for additional SS overtime. Security and commissionaires, as well as shared resources like fire sentries or crane operators may be needed. It is also not realistic to think that one could pre-plan one extra day of overtime to make a 6-day activity finish in 5 days, due to high variability in calendar completion times. Commonly, the knowledge that one extra day of overtime is all that is needed to finish a 6-day activity in 5 days is only available by day 3 or 4 of that activity. Often scope growth requiring much more work, or temporary modified execution modes may be a better solution [3]. Again, these cannot normally be determined in advance of work being conducted, especially for *all* activities as is the case in the reviewed literature.

It was found from interviews with PLs, that to see the effect of applying overtime or reducing the duration of an activity by one day; the most efficient way to do so is to reduce the duration of the activity by one day and re-schedule (in a sandbox program, not directly in the ERP system) [3]. A note should be made that not all activities can be shortened with extra shifts due to complexity. Large disassembly and re-assembly activities for example, are best done by the same workers, because they will know where items have been placed, and where all the many small components are supposed to go for re-assembly. On the other hand, relatively simple activities such as scaffolding installation or equipment removals, with a small turnover between shifts, can be reduced by a different set of workers. The more complex is a work sequence, the less desirable and the more impractical it is to have different people exchanging information and working on the same tasks.

It would be prohibitively time-consuming to ask and confirm overtime availability of all workers and support agencies for every time unit throughout a work period, or to do the same for all sub-contractors. Interviewed real-world schedulers, PMs, and PLs indicated that a schedule should never include overtime to meet objectives. Overtime is used as a last resort to recover a schedule or complete activities within a makespan, and how much overtime is needed, by whom, their availabilities, etc., is determined on a case-by-case basis for work that is already in-progress where this information is better understood. In some cases where work is known to create unfeasible conflicts such as during a work period where there is significant aloft work, and diesel generators also need to be run for many hours, work may be manually scheduled to occur after hours (diesel engine run-ins), where a group of workers will perform activities during a sequence of several night shifts [3] [4]. Placing the constraints for such an exception would be more tedious than manually overriding the start times and discussing a plan with supervisors, especially since the latter must occur regardless.

In addition, several overtime formulations and methodologies exist in literature, usually with the goal of minimizing costs [40] [49] [105]. While these provide for interesting problems, the contribution of this thesis is multi-mode formulations (that can be used for overtime) that do not require cost information. Based on interviews with FMFCS, workers are not customarily asked if they can work overtime unless it becomes a necessary activity, since if workers become aware that overtime is being considered, some may slow down how quickly they work so that the expected amount of excess work needed is approximately enough to "fill" most of an overtime shift [3] [4].

Above and beyond this, in every schedule provided by DND, ISI, Seaspan, and FMFCS for this thesis, costing information is not available to the scheduler for scheduling. Using a cost minimizing function may not be practical without using some tricks to give default costing values. Moreover, the ISI and Seaspan shipyards are paid by the Department of National Defence (DND) using "cost-plus" contracts, where every work hour used is paid directly with a profit percentage. To illustrate, the profit percentage may be approximately 10% plus an administrative markup, that is added to every completed

work hour by the contractor (see PSPC cost and pricing guide for more information [106]). It should be noted that work performed at non-standard rates will not be compensated differently, so using standard rates for hours is always preferable for the contractor except where very important work may not be completed on time and previous planning for this (for evening and nighttime shifts) was not done; this will of course reduce the shipyard's overall profit. The more work that these shipyards can "fit" into a work period at a standard rate, the better it is for the shipyards (financially) and for the RCN (qualitatively), as these ships will be further along in their readiness once the RCN retakes primary care responsibilities from the shipyards after the DWPs.

Worker availability in term of hours is standard industry practice, since sharing financial information is taboo to public servants and their affiliated industries, to avoid sharing commercially confidential and sensitive information such as salaries. For these reasons, the proposed overtime formulations of this thesis allow the user to find the most impactful activity, or activities from a small set, that will recover a schedule, or achieve a desired makespan or quantity of work, to guide the user's overtime investigation, simply by placing constraints on the allowable activity reductions.

Chapter 4: FMFCS Data Analysis

Data analysis, related assumptions and discussion, preliminary experimentation for formulation and heuristic development, for both the NSWPP scheduling and rescheduling scenarios are presented in this section.

4.1 Research and Work at the Fleet Maintenance Facility Cape Scott (FMFCS)

Several meetings were held at FMFCS' Operations Department to discuss and implement the use of a trial automatic scheduling tool. In the first phase, information from historical work periods was gathered and analyzed for statistics, to confirm researched scheduling theories, to model typical NSWPP network architectures, to confirm assumptions relating to these types of work periods, to match the program with the input format, and to better understand the problem. In the second phase, project leaders (PLs), project managers (PMs), and schedulers were interviewed in progressive phases as prototype software was developed by the author of this thesis to assist in specific NSWPPs that were in the initial planning phases.

The scheduling program was developed in MS Excel and Visual Basic for Applications (VBA). Using a different program such as R or Python would require special permission by relatively high levels of the RCN and it was the author's opinion from previous attempts at software approval that the approval process would most likely extend well beyond the duration of a typical MASc Program. It was important to develop a program that could readily be used in the Defence Wide Area Network (DWAN), which has many software restrictions. DWAN is an unclassified network with need-to-know information of a sensitive nature and security software is constantly being updated to detect foreign intrusion attempts and malware. The security software is tailored to every computer's software installation, software versions, and combinations of those. Licences are managed centrally, and software acceptance is tightly controlled [107].

FMFCS Operations Department is usually understaffed and deals with a multi-project environment encompassing the largest military industrial complex in Canada. It employs over 1200 unionized civilian workers and over 140 military members [108].

FMFCS is responsible for all 2nd line maintenance projects and some 3rd line maintenance work for all HAL Class Frigates in the Atlantic (seven ships at 4500-ton displacement), 2nd-line work for SECRET-designated systems for six Maritime Coastal Defence Vessels (MCDVs), some 2nd line maintenance projects for the interim tanker MV Asterix since traversing to Lévis, Quebec is often inefficient, refits and 2nd line maintenance projects for a varying number of the four Windsor Class Submarines, as well as many smaller projects for Atlantic-based Royal Canadian Army (RCA), Royal Canadian Air Force (RCAF), Atlantic Task Force Security and Special Forces, and, occasional short-notice maintenance projects for foreign ships. Due to its responsibilities and a decade long reduction in manpower as a result of a Federal push to reduce the government footprint and increase the use of In-Service-Support (ISS) Contractors [88], the Operations Department is often too busy to answer external questions unless it has implications for them. Our approach was to offer a software program that could truly benefit the current scheduling regime, thus giving incentive to answer questions and provide valuable time for this research project in exchange.

The developed program uses a simple list scheduling heuristic or serial SGS, as this is easily understood and was specifically requested by users after the different options were presented and discussed. The primary driver was to speed-up the manual process currently employed by schedulers and PLs, so that more work could be scheduled, and better shop utilization would follow. The idea that a schedule could be re-scheduled with different sets of underlying data and space capacity limits was of great interest. It became evident that a move to automatic scheduling would be difficult as current practices were quite cemented. The serial SGS scheduler is attractive because it is easy to understand, and the schedulers still feel like they have complete control over the resulting schedule. It is theorized that for optimization to become accepted and attempted, a scheduling staff should first become more familiar with basic forms of automatic scheduling, such as the serial SGS, where this is basically a replica of human list scheduling and is easy to follow. If the milestone of trusting and using simple automatic scheduling tools cannot be

achieved, with all the resulting implications such as a need for more precise and accurate data, then how could an organization successfully adopt optimization?

4.2 Analysis of Historical FMFCS Data Background

Historical scheduling metadata of an entire naval surface ship's extended work period (EWP) was used for detailed analysis, as a large data file with 636 WOs. The level of detail provided by DND stopped at the WO level and did not include individual activities. Work in NSWPPs is characterized as work with high variability in terms or task durations, scope growth, and fluctuating resource availability [1]. FMFCS uses traditional critical path method (CPM) for scheduling activities and users individually verify if shop capacities can support work. It should be re-iterated that in NSWPPs, it is not imperative to complete all tasks such as those during an aircraft work period, a submarine work period, or a construction project. There are typically higher priority tasks that are mostly completed due to higher managerial attention, while lower priority tasks are completed if the personnel and material are available, on an opportunity basis. These lower priority tasks are not only completed when the needed shops are available, but they are only completed when the planning, scheduling, and project management staff in the operations department make time to plan and schedule these.

It should also be noted that only one shop is assigned to any one activity. In the Defence Resource Management Information System (DRMIS) scheduling planning tool, task estimates are used for billing and accounting, and two shops will very rarely spend the same number of total hours on a single activity. Each shop will be responsible for their own time keeping, so even if three shops are needed to complete essentially the same activity, this will be scheduled as three separate parallel activities.

A growth in scope for an activity generally requires a new activity to be scheduled. The implication of this is that data on hours estimated and hours billed are not direct historical indicators of true calendar task completion times or calendar time distributions. When a WO was underestimated in term of duration or shops required, it is normal to either cancel work (very few hours will be charged to this underestimated WO) and start

work on a new WO with more hours assigned, or “use-up” all the hours on the incorrect WO and raise another sub-WO to charge additional hours. The effect of this is that a comparison of hours estimated vs hours billed will not follow any standard completion time distribution with a long trail, because work that took much longer than estimated work will not be present in the data. For context of the analysis in the next paragraphs, note that in literature, maintenance task completion times are often appropriately modeled to be lognormally, Weibull, or beta distributed [109].

One focus of analysis in this section was to compare theoretical scheduler estimations such as those provided by Goldratt (1999) [92] and later criticized by Raz et al. (2003) [83] and Trietsch (2005) [84]. Goldratt and Critical Chain Project Management (CCPM) advocates claim that estimators typically estimate work such that there is an 80-90% probability that the work will be completed within the estimated time. A traditional CPM strategy places float/slack/buffers within each activity to protect the schedule from disruption. This does however allocate the float to individual tasks instead of the whole project and does not promote early completion of activities. This time is often wasted due to Student Syndrome and other human factors. Even if a job could be completed early, it is often lengthened to use-up the time provided.

To re-iterate CCPM concepts, the time estimates are tighter as they are supposed to be equal to the median time duration of activities, and this encourages work to be done with expedience [92]. It is expected in CCPM that 50% of work tasks take longer than estimated in a schedule, and this must become embedded in the organizational culture for CCPM to succeed, which has been adopted by some US, Australian, and Japanese dockyards [110]. In CCPM, project float is calculated and placed deliberately at the end of task sequences, where the float is shared throughout the entire project. Throughout the project, this float gets consumed or not, and more jobs can be added if things go well, or corrective action can be taken as the buffer becomes overly consumed.

CCPM criticism of the 80-90% assumption that is standard with the CPM [83] is that there is not enough academic evidence to support this claim, so a study on real-world

naval maintenance data was conducted to compare results. Raz states that of the case studies analyzed, work estimates were only exaggerated by 60% on average, although these were not conducted on NSWPPs [83]. Since this current analysis looks at NSWPP RCPSPs, where no real-world studies about overestimation have been published in researched academia as far as this author is aware, it is unknown where the trend lies.

4.3 Analysis of Naval Ship Data for a 2018-2019 EWP2 Project

The first dataset provided by FMFCS, approved for limited use in this thesis by DND, was analyzed in detail to determine important characteristics for NSWPP projects. This dataset was that of a ship in an extended work period (EWP) where 636 WOs (work packages) were assigned revision codes relating to this project. The data was extracted from DRMIS.

Of the 636 WOs that were scheduled for the work period along three official priority levels, only 399 WOs had any actual hours charged during the work period, representing an instance where 37.3% of the originally planned work did not occur. These WOs were not completed for a variety of reasons from missing materials, space and resource conflicts, workers being re-assigned to another higher priority ship, etc. In some cases, the original work estimate was incorrect, so the standard procedure is to cancel the WO and re-open another one that is correctly planned [14], but this realization typically only happens once the work has begun. Insufficient information was available to reliably determine statistics on the reasons for work not being completed, but Couch (2016) [1] described some interesting results from a different dataset examined for his thesis, where some information was analyzed for causation [1]; see figure 16, where only 31% of cancelled activities could be traced back to a specific causal category (16% scheduling conflict, 11% unavailable materials, 4% no longer required). In the same analysis, where discretionary work was provided, Couch (2016) [1] concluded that 71 new WOs were added to the SWP being analyzed, and the original number of WOs was 132, representing a 53.4% increase in scope growth throughout the project. 93% percent of this additional work was completed, a high percentage similar to expectations, as policy

dictates that significant new work is only accepted in the same work period that it is found if it is urgent [14].

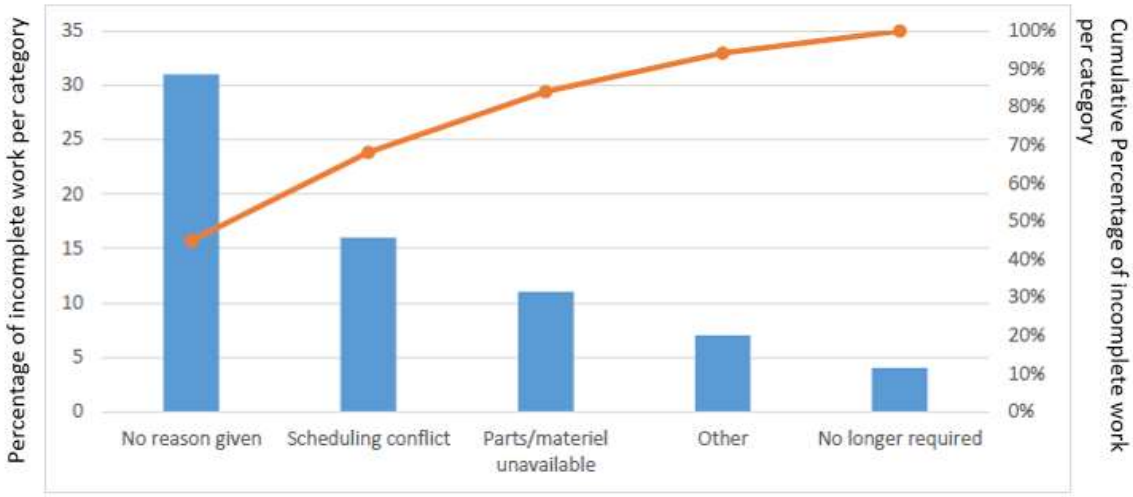


Figure 16: Reasons for uncompleted work (source: Couch, (2016) [1]*).*

It should be noted that the DRMIS instances provided to us did not include data fields that could be used to decipher the volume of work (out of the 636 WOs), that were added throughout the EWP; although from discussions and experience, it is anecdotally accepted that a large volume of work (about a third of the work) normally arises during the project execution phase [3].

As described in the previous problem definition section, it is accepted that the scheduling software being used, Prometheus Scheduler as a DRMIS add-on, has restricted mass-change capabilities via administrative rights and scheduling is performed manually. It does also allow work to be scheduled such that shop resources are scheduled well beyond a shop's work capacity, leading to partially unfeasible schedules. These are needed for urgent work scheduling for higher priority activity scheduling. For example, if a submarine needs shop resources (highest priority vessel in a dockyard), then the submarine PL will normally schedule away without considering shop resource overallocation; it will then become the lower-priority ships' PLs responsibility to cancel or modify work plans.

There is an automatic feature in Prometheus Scheduler that allows scheduling WOs at the earliest feasible time, but users did not like using the feature because it would

separate activities within a WO if shop resources could only support some activities. Unless this was impossible, users normally preferred to have all activities within a WO occur as planned following CPM early start sequencing, without significant delays between activities in the same WOs. The practical reasons for this are: that the work is complicated so workers taking long breaks between activities are more likely to make mistakes, tools and scaffolding are left around a work site causing obstructions, steel is often stripped and exposed making it more susceptible to corrosion, ventilation is often set-up using temporary trunking throughout passageways and access doors, and leaving a WO half-done is often worse than not even starting it. Note also that when the planning department plans the work using CPM precedence relationships, the activities within a work package can always occur sequentially or in parallel as planned, as no sum of parallel activities exceed FMFCS' shop capacities for any single work package [3]. When resource demands from many WOs for different projects are combined however, then resource constraints play a primary role. Moreover, it was found during the field study visits at FMFCS that several shops have capacity reporting problems, where for instance, one shop may have three workers, but the available hours are only set to reflect one or two workers. Since the underlying capacity data was incorrect, the scheduler and PL had the choice of going through the work center supervisor and requesting a capacity information adjustment, wait for the supervisor to update the system, then use the feature to schedule one WO; or, they could more efficiently (for the short term) simply accept that the data was incorrect and manually override the WO to schedule it when they "knew" (not from the DRMIS capacity information) that the work could proceed. Virtually all work was scheduled manually as keeping the underlying data perfectly accurate was a significantly difficult task; the data was used mostly as a guide.

Figure 17 shows a distribution of actual work hours recorded vs estimated work hours for the 636-WO EWP described earlier, where a value of 1.00 indicates that the accounted-for actual hours were equal to the estimated hours. This same data was then

automatically distributed with Arena Input Analyzer for a best fit standard distribution exercise.

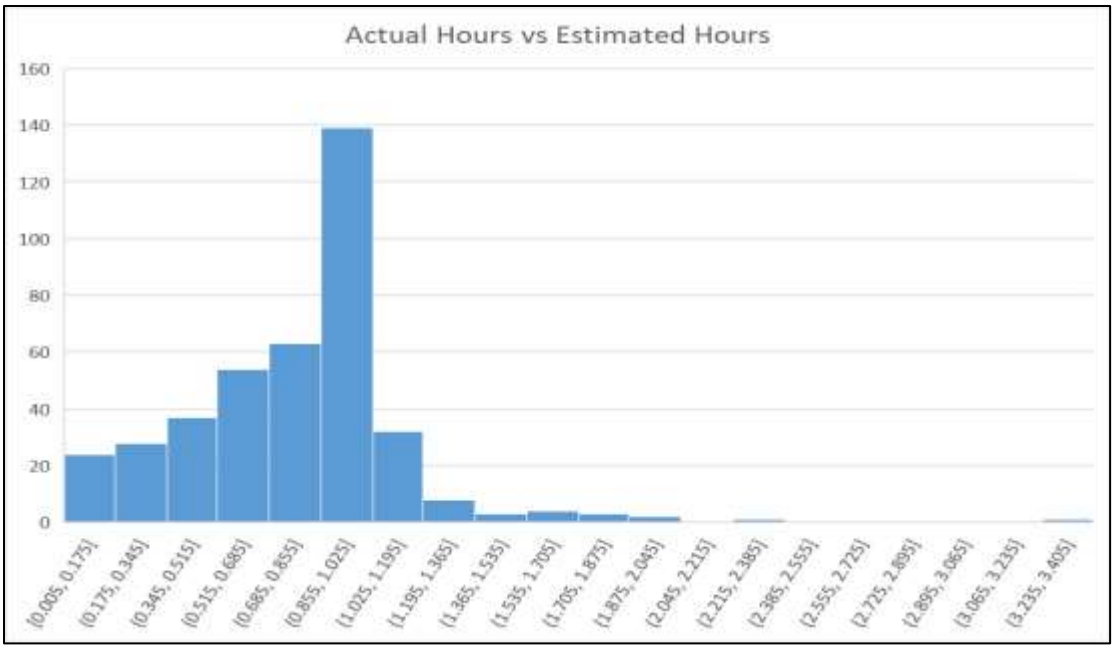


Figure 17: Actual hours vs estimated hours looks different than measurable work duration estimates in literature. Perhaps the estimate itself affects the actual work hours.

Arena Input Analyzer Software and MS Excel were used to analyze actual work charged vs work estimates. The best-fitting distribution shows that the data is closest to normally distributed (Normal(0.779,0.369)), although this poorly fits as can be seen in figure 18. The average actual hours are 78% of the average estimated hours. One standard deviation is 37.0% of estimated time or 47.4% of the actual hours average.

The distribution analysis indicates that along the cumulative distribution function (CDF), estimates only exaggerate average task duration by 28.2%, and given the high variability of data using the best-fit normal distribution, this represents 0.59 standard deviations, and 72.2% probability of completion, which is in between Goldratt (1997) [92] and Raz's claims, where Raz (2003) [83] suggests that estimators were found to only exaggerate work estimates such that 60% of work could be completed on time.

It should be noted however that 44 of 399 WOs, 11% of total WOs, finished exactly on time (the mode), indicating that perhaps with no motivation to finish early, some WOs are indeed extended longer than needed instead of being reported to having finished early, in accordance with Goldratt (1997) [92]. When using the data directly instead of the fitted distribution, 16.3% of WOs have more actual work hours billed than the work estimates. This shows that estimators do use a significantly higher than 60% confidence of completion estimate using traditional CPM (83.7%).

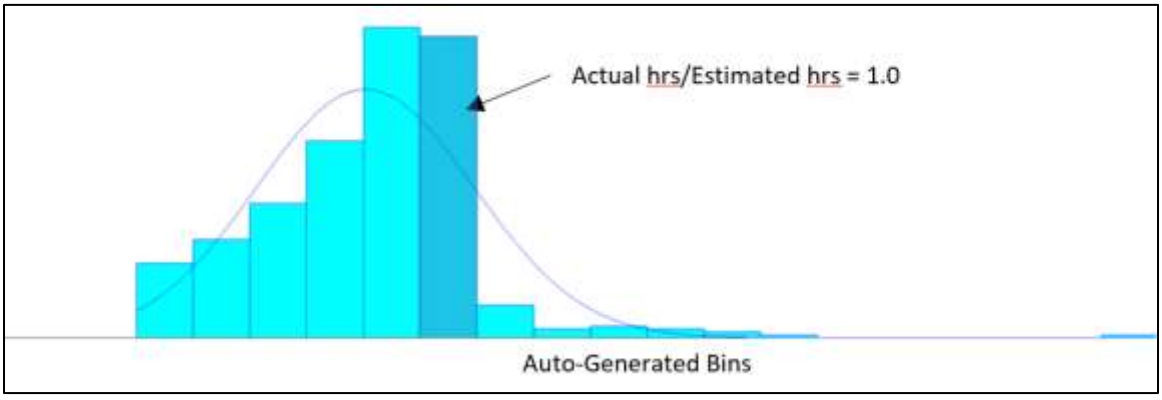


Figure 18: The best fitting distribution using Arena Input Analyzer poorly fits. Actual hours charged to WOs drastically drop after the estimate is achieved.

If experimentation is however completed with the assumption that WOs completed before the task estimates took longer than required as proposed by CCPM theory, and it is assumed that tasks that took longer than the estimate were not prolonged due to managerial pressure, then the data could be modified to better reflect task durations if they were unaffected by the estimates.

A proposed solution under the same previous assumptions is that for every task that took equal to or less than the estimated time, the task duration was extended by anywhere between 0 and 100% of its true duration, thus artificially accounting for Student Syndrome. This represents some workers working a little slower than required when there is more time left than is needed, some working as slow as half-speed at the extreme, and some working at a normal pace regardless of having more time to finish.

A uniformly distributed randomized instance of this provides the distribution generated by Arena Input Analyzer in figure 19. It can be seen from the resulting

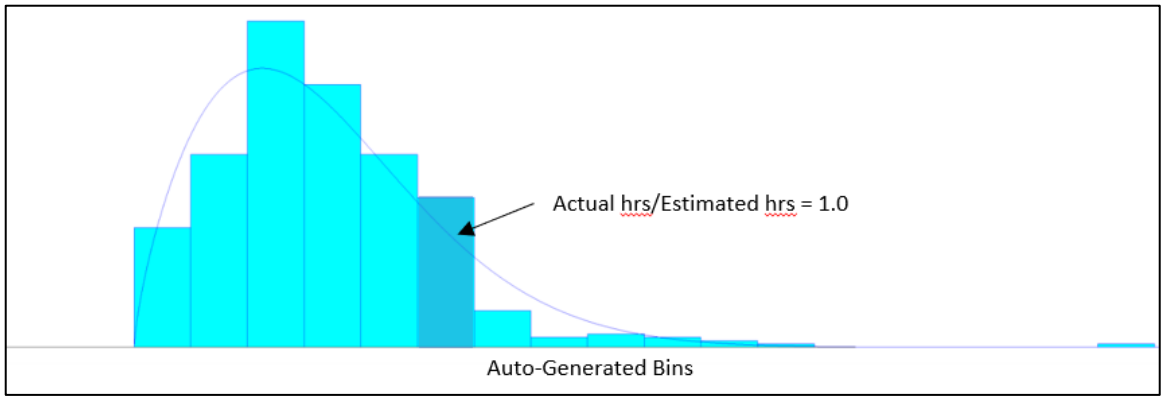


Figure 19: Accounting for Students' Syndrome with simulation, actual work hours vs estimated work hours produces a theoretical Weibull or Beta distribution with a good fit.

distribution image that this theoretical Weibull distribution fits better than the previously displayed normal distribution in figure 18. This distribution matches previous research that true task completion times are modelled well with Weibull distributions, as proposed by Rummel (2017) [111]. The Weibull distribution has a closed form CDF as seen below:

$$F(t,\beta,\theta) = 1 - e^{-\left(\frac{t}{\theta}\right)^{\beta}} \tag{26}$$

It can be computed with t=1 (the estimated time), β=1.72, and θ=0.723 (determined from Arena Analyzer), that the estimated hours for activities represent 82.6% of the probability distribution. This analysis with new sets of pseudo-random numbers, was repeated 20 times and the best fitting curves were Beta (35%), Weibull (55%), and normal (10%), with Weibull being among the top two curves at every instance. The randomized formula for modifying each task taking less than or equal to the estimate is provided below, where *x* equals the reported actual hours, and *y* represents the artificially reduced hours, removing a predicted duration extension caused by Student Syndrome, when the reported hours were equal to or less than the work estimate:

$$y = \begin{cases} x & if\ x > estimate \\ x - x \times UNIF(0{,}0.5) & otherwise \end{cases} \tag{27}$$

The analysis of overestimation times from these 20 instances indicated that estimators give completion probability duration estimates of 82.9%±0.716% with a 90% confidence interval, when the assumption of Student Syndrome is considered as previously described.

For these reasons, this author does not reject Goldratt's (1997) [92] estimates of Student Syndrome and estimator habits of scheduling, such that work can be completed 80-90% of the time, applicable to NSWPPs at FMFCS; furthermore, there is evidence that the estimate has a very large impact on work completed. In the case of FMFCS/CB, SNC Lavalin, Thales, Seaspan, ISI, and other Canadian NSWPP companies that schedule sub-contractor work or have a long history of scheduling this way, it is proposed that using median time estimates would be impractical because the estimates themselves come from the sub-contractors or workers (who do not want to be late 50% of the time) and this would require a transformational change that is beyond the scope of this research. It is also proposed that using many buffers, or additional buffers, at different places throughout the project is very likely to increase overall project duration without completing more work, as workers will be even more prone to Student Syndrome as indicated by the data. In a sense, the individual activities are already buffered most of the time, so additional direct and purposeful buffers throughout a project are not recommended.

4.4 Why the data does not reflect calendar durations

One might think that smoothing out actual hours vs estimated hours distributions such as the one in figure 17 could be used to estimate duration stochasticity; however,

Sched. start	Sched. finish	Revision	Priority	Created on	PM actual	PM planned hours
2018-10-31	2018-11-01	FNA181FR	07	2017-11-15	20.750	24.000
2018-11-16	2018-11-19	FNB181FS	43	2018-11-05	4.330	5.000
2018-11-05	2018-11-20	FND181FU	43	2018-08-08	13.000	15.000
2018-12-03	2018-12-17	FNA181FR	43	2018-11-06	90.250	104.000

Table 1: Actual calendar start-to-finish windows are much larger than the planned PM hours. There is no found metric that can reliably be used to estimate calendar durations vs planned hours.

this will not show delays resulting from inclement weather, conflicts, worker absenteeism, insufficient materials, scope growth etc.

The snapshot of maintenance data in table 1, for four WOs, shows that the scheduled start and finish dates are not related to planned PM hours. The calendar dates are generally exaggerated so that if workers are needed for another activity, they can finish that activity and return later, but still complete the activity within the time window allocated [4]. When work is delayed, for example by inclement weather or a conflict, then no hours are billed to "PM actual" hours on that WO, so the delay is not captured. The comparison of PM planned vs PM actual is useful to determine how effective the estimates for WOs are when compared to the billed hours associated with those estimated, but not for calendar durations. As well, from detailed analysis and interviews with PLs, PMs, and schedulers, there is no way to find true completion times for work from historical data [3]. It is not reflected in the DRMIS data and timekeeping for finishing activities is not maintained accurately. Modelling true variability using available data is not possible; however, one can intuitively predict that it would be much more variable than the planned PM hours vs actual hours distribution of figure 17.

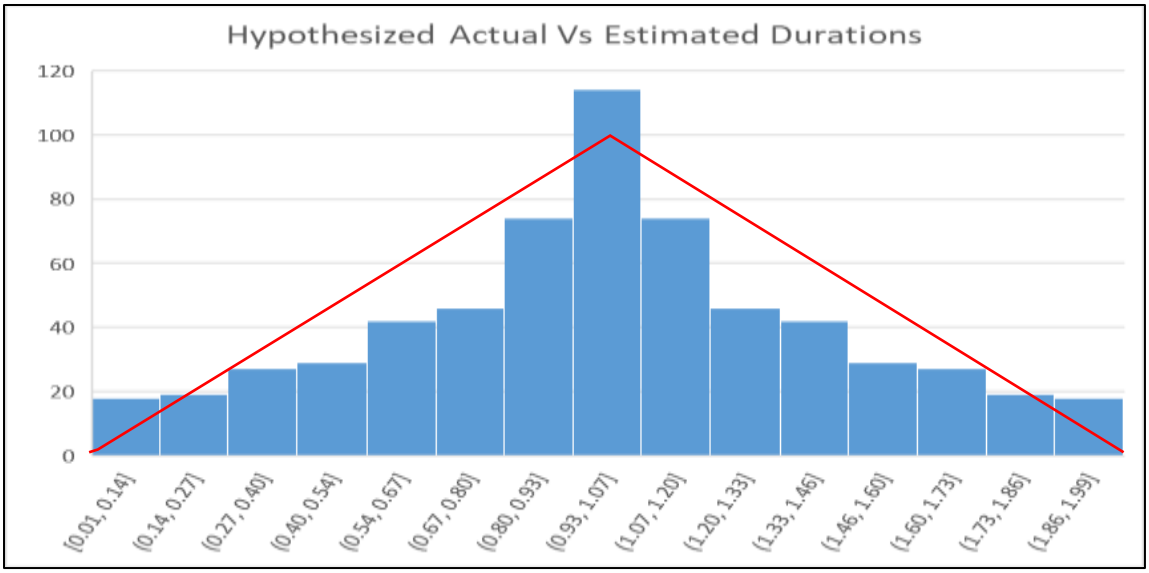


Figure 20: A triangular distribution (0,1,2) may be useful in predicting calendar durations without any better data.

Figure 20 is an example of how actual calendar completion data could be predicted if work was planned and scheduled based on estimated time durations. The left side is from the actual vs planned data, while the right is simply a mirror image, used to approximate calendar delays resulting from worker absenteeism, scope growth, weather delays, and material delays. Although probably not accurate, this stochastic distribution can be used to indicate the benefit of buffers placed at the end of critical sequences for work with high variability, and it coincidentally fits a relatively simple triangular (0,1,2) distribution. Although simulation experimentation was conducted with this variability distribution, the results were obvious and show the benefit of buffer as already described by Couch (2016) [1] in a reactive scheduling scenario with imperfect data.

4.5 Additional NSWPP Data Features

While spending time at FMFCS, an additional unscheduled dataset of 433 activities comprising of 82 WOs for work that was not yet scheduled was provided. These activities were all in the "ready to schedule" status, and an interesting trend was found that will be used in random problem generation of this thesis. Since all activities are planned by the planning department and activities precedence and parallel relationships are given without first considering resources, it is found that an earliest start date is given to all activities. For every WO, the first activity is given an earliest start date on the first day of a work period, while successor activities have earliest start dates that begin immediately after earlier activities. Some WO exceptions existed for when the work was originally planned for an earlier work period, but the work never occurred, so the start dates displayed in DRMIS were for this past work period instead of for the one being currently planned. Barring these exceptions, by only looking at these calendar dates, it appears that 92% of the work can be completed in the first eight days of the 20-day project, representing 40% of the project's makespan, since these have only been scheduled based on precedence relationships and not such that resources were considered. Using a classic serial SGS scheduling program, it was later found that only 65% of this work can even be scheduled in the 20-day work period once resource constraints are considered. Random problem generation used in later sections of this thesis will be completed using this

feature, where some activities have precedence relationships, but activities are built in small sets that all may start on day zero when considering only precedence relationships. The resource constraints of the project should be such that the makespan will be increased by 200% to 300% when these resources are introduced, to properly mimic this data set.

FMFCS did input the ship's compartment location in alphanumeric formats where work was being conducted, but this was not used in any automated way to enhance scheduling. This data could however in theory be used to model same compartment/space constraints described in Chapter 2.

To conform with the nature and variability associated with the analyzed datasets, and the qualitative goals of NSWPPs described in this thesis, formulations that more properly model the problems and their constraints are developed and described in the following chapter.

Chapter 5: New Formulations and Heuristics for the NSWPP

Based on analysis of literature, preliminary experimentation with existing discrete-time formulations and heuristics, and several iterative improvements, this chapter presents new formulations and accompanying heuristics for the initial scheduling and re-scheduling scenarios. To briefly recollect, the goal of initial scheduling is to produce a feasible and conflict-free front-loaded schedule that considers priority and duration, while the goal of re-scheduling is to produce a new schedule that deviates as little as possible from the previous schedule.

5.1 Proposed Formulations for the Initial Scheduling Problem

Based on the problem definition and the literature review, it is proposed that all common constraints including same compartment conflicts, hot work, emissions, aloft work, and radiation, as well as miscellaneous delays, can be modelled as resource constraints. For example, if the compartments in a ship are modelled as resources, then every planned activity intended for scheduling must consume this space resource, or part of a space-resource's capacity. The capacity of the compartment resources could be defaulted to one activity per space, unless there are parallel activities for the same WO. For example, the HAL Class frigate has 353 spaces, but only a dozen of these can generally "fit" more than one set of parallel activities at a time. Since these spaces are bigger however, they tend to have relatively more work scheduled into them. There is no simple metric to reliably determine how many different activities (from separate WOs) can fit in a space for many of the reasons described in the problem definition section relating to the nature of the work, so it would be practical to leave this up to the user to select a limit and a space resource consumption per activity, based on the nature of the work and his/her experience. In a real-world scheduling exercise as seen from the work at FMFCS, the user would have to iteratively re-schedule until the person is satisfied with the number and nature of concurrent activities being scheduled in the same space at the same time. The data itself does not currently include the necessary metrics to automatically detect this, but an experienced user can tell from the task description, perhaps requiring conversation with shop workers, to estimate how intrusive certain

tasks are relative to each other. Based on user feedback with FMFCS, a default value of one set of parallel activities from the same WO per space means that the scheduler only needs to modify the capacity of a space for a small number of spaces, instead of having to manually modify the compartment consumption of each activity [3]. Some spaces like the engine rooms, the hangar, the flightdeck, the boat/missile decks, or the forecastle may be defaulted to perhaps four WOs, but these capacity limits should be able to be modified easily by the user. Users desired the ability to see the resulting schedule, consider the concurrent resulting same compartment WOs, and then decide based on their experience if this mix of work is acceptable, considering makespan and the likelihood of conflicts. If the mix is not acceptable, the space capacity would be modified, and the schedule would be re-calculated.

Hot work conflicts can be accounted for by selecting the appropriate adjacent space being affected by hot work activities. A prompt could guide the user to identify a compartment that should also be "consumed" by the hot work activity during its execution. FMFCS uses an optional customized user field called "work type" in DRMIS, which alerts the PL or scheduler of special work types such as hot work, aloft, emissions, etc. There is a standard set of work types known to cause conflicts that are selectable to planners.

Emissions and radiations can also be modelled by resource constraints, by allowing the user to select an "aloft" resource. For example, if every aloft activity consumed 0.2 of the aloft resource, but an important radiation activity consumed 1.0 of the aloft resource, with a capacity of 1.0, then up to five aloft activities could be feasibly scheduled concurrently, while none could be feasibly scheduled at the same time as the radiate activity. The user would need to play around with these values depending on the number and priority of aloft, radiate, and emissions activities that are present.

Miscellaneous delays may also be modelled as resources. For example, if the flight deck and hangar are out-of-bounds for the duration of a promotion ceremony, then the user may add a dummy activity with a half-day duration that consumes all the hangar and

flight deck resources; thus, preventing work in those areas from occurring. If the entire ship is out-of-bounds during something like a “fast-cruise”, which is a ship-wide training exercise that makes use of fake smoke and the general alarms, then this calendar day can be added as a holiday in the project default information.

For duration and priority however, these aspects cannot be easily converted to any parts of the studied classic and modified RCPSP formulations in literature, so a discrete-time Priority-Duration RCPSP formulation was developed. It should be noted that this formulation is only proposed for the pre-execution (initial scheduling) phase of a NSWPP project, where the goal is to build a robust schedule that naturally causes a priority-1 buffer, by front-loading higher priority activities earlier using a weighted objective function. The duration term in this objective function is in place to prevent shorter same-priority tasks from being scheduled earlier, as described in section 2.4. This duration value can be tuned to force tie-breaking such that longer activities are scheduled earlier among the same priority level, at the cost of a potentially longer makespan. Note that duration, instead of relative resource capacity usage, was selected because this information (a time estimate) is always available. When working with sub-contractors, the number of workers and capacity of those workers is not normally provided [3].

5.1.1 Multi-Mode Discrete-Time Priority-Duration RCPSP Model

Indices:

1… *A* for activities

1…*K* for resources types

0…*H* for the time horizon

1…*M* for multiple duration modes

Sets:

P activity immediate predecessor pairs (*i,j*)

K resources

J activities

M Execution modes with various durations d_{jm}

Parameters:

H integer, the planning horizon from the project start (0)

A integer, number of activities

d_{im} and d_{jm} integer, duration of activities *i* and *j*, for every mode *m*

r_{jk}integer, activity *j* demand for resource type *k*

$R_k$integer, resource *k* capacity

ES_j integer, earliest start time of activity *j*

LS_j integer, latest start time of activity *j*

p_j integer, priority of activities (1=ESS, 2=HOPP, 3=NOPP)

θ (≥ 0), used to assign weights to activities proportionately to their priority

ε_1 (≥ 0), very small value to account for milestones that have no actual duration

ε_2 (≥ 0), very small value to penalize late scheduling of activities

α (> 0), use to give tie-breaking benefits to longer duration work (α=1.1) or making each work duration unit equal (α=1.0)

β_{M1} total alternative shift limit

β_{M2} alternative mode reduction limit

Variables:

$$x_{jtm} = \begin{cases} 1 & \text{if activity } j \text{ starts at time } t \text{ in mode } m \\ 0 & \text{otherwise} \end{cases}$$

Objective function:

$$Maximize \sum_{m\in \boldsymbol{M}} \sum_{j\in \boldsymbol{J}} \sum_{ES_j}^{LS_j} \frac{x_{jtm}}{p_j^{\theta}} (\varepsilon_1 + d_{j1})^{\alpha} (1 - \varepsilon_2 t) \quad (28)$$

- Maximize the weighted summation of activity start times, that is proportional to the activity durations, inversely proportional to an exponentially grown priority number for each activity (to outperform the duration weight), with a start time penalty that grows as the start time of every activity is increased.

Subject to:

$$\sum_{m\in \boldsymbol{M}} \sum_{ES_j}^{LS_j} x_{jtm} \leq 1 \qquad \forall\, j \in \boldsymbol{J} \quad (29)$$

- All activities must be completed between the activity ES and LS times, no more than once.

$$\sum_{m\in \boldsymbol{M}} \sum_{ES_j}^{LS_j} x_{jtm} t \geq \sum_{m\in \boldsymbol{M}} \sum_{ES_i}^{LS_i} x_{itm} (t + d_{im}) \quad \forall j \in \boldsymbol{J}, \forall (i,j) \in \boldsymbol{P} \quad (30)$$

- A successor's start time must be greater or equal to a predecessor's start time plus its duration.

$$\sum_{m\in \boldsymbol{M}} \sum_{j\in \boldsymbol{J}} \sum_{b=\max\{t-d_{jm}+1, ES_j\}}^{\min\{LS_j, t\}} x_{jbm} r_{jk} \leq R_k \quad \forall k \in \boldsymbol{K}, \forall t \in 0 \dots H \quad (31)$$

- The sum of resource usage for the period of activity start times to activity end times must not exceed the resource capacity for of each resource type *k*.

$$\sum_{j\in \boldsymbol{J}} \sum_{ES_j}^{LS_j} \left(x_{jt2} + 2x_{jt3} + \cdots + (M-1)x_{jtM}\right) \leq \beta_{M1} \quad (32)$$

- A total maximum number of additional evening and nighttime shifts, or simply reductions, can be set using the value of β_{M1}. Start times in mode three represent two additional shifts, so the start time is multiplied by two, mode four is for three additional shifts (therefore multiply by three), etc.

$$\sum_{j \in \boldsymbol{J}} \sum_{ES_j}^{LS_j} (x_{jt2} + x_{jt3} + \cdots + x_{jtM}) \leq \beta_{M2} \qquad (33)$$

- This constraint allows a limit on the number of total alternative execution modes for all WOs. It is the most practical alternative execution mode constraint when limiting the number of overtime investigations is desired.

$$x_{jtm} \in \{0,1\} \qquad \forall\, j \in \boldsymbol{J}, t \in ES_j \ldots LS_j \qquad (34)$$

- Start times are binary, thus a variable exists for every time unit between each activity's ES and LS.

This formulation should be used in scenarios where the scheduler or PL wishes to schedule all work from a list and "see" where the scheduled dates fall, as well as analyze the resulting schedule for any problems (a practical necessity with automatic scheduling). This is the most typical initial scheduling scenario. If activities have finish times that fall outside of the project makespan, then the scheduler will not schedule this work, or look at other tradeoffs that use the same shop resources. Since the formulation gives a major weight bonus to higher priority activities, it is unlikely that higher priority activities will not be scheduled within the makespan unless there are resource capacity problems, or there is too much high priority work to be scheduled. This formulation does not enforce a finish time, so a feasible solution will always be given with enough time.

In the experimentation phases of this thesis, θ was set to 5, and α was equal to 1.0 or 1.1. The reason for θ being equal to 5 has to do with solver objective function precision limitations that will be discussed in the next section of this thesis, while the reason for α values of 1.0 and 1.1 are due to adding a tie-breaking duration bias or not.

If no duration bias is added, so that α=1.0 in equation 28, then there can be several scenarios where arranging varying-duration work in different orders will produce the

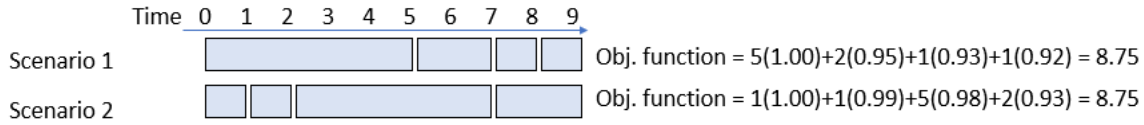


Figure 21: With no duration bias, the same volume of duration-weighted work is worth the same work with either arrangement.

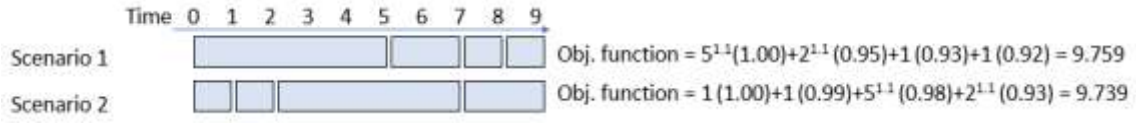


Figure 22: With a >1.0 duration bias, the same volume of duration-weighted work is arranged so that longer WOs are scheduled sooner among the same priority level.

same objective function values, as long as the priority and overall activity time units are equal, such as in figure 21. If ignoring ε_1, alternative modes, and priority, as the duration segment of the objective function is used for tie-breaking among same priority levels, and ε_2 is set to 0.01 as the time-delay penalty, then the representations in figures 21 and 22 show the effects of making α values equal to 1.0 and 1.1 respectively.

This bias thereby gives a small objective function benefit to scenario 1 from figure 22, making longer duration work scheduled earlier as a default tie-breaking rule, which does not outweigh priority. This duration bias is also suitable to handle the problem definition aspect that selecting longer duration activities should be scheduled whenever possible, rather than an equivalent work duration sum comprising of many smaller-duration activities. There may on occasion be a trade-off as will be seen in chapter 6, where applying the duration bias (α =1.1) may produce a slightly longer makespan as a result.

It was found from experimentation that it is unnecessary to enforce a finish time constraint if the late start (LS) value for each activity is determined using a backwards pass from the project makespan *H*. If the LS has not however been calculated, or has been

calculated from a large makespan H, then the makespan may be alternatively enforced by adding the following constraint:

$$\sum_{m \in \boldsymbol{M}} \sum_{ES_j}^{LS_j} x_{jtm}(t + d_{jm}) \leq H \qquad \forall\, j \in \boldsymbol{J} \tag{35}$$

This will only select the optimal combination of highest value activities in a makespan for scheduling. Appendix C shows this formulation in GLPK/Gusek, where Appendix D is the data table format used for the program. Only one mode constraint is used in the example of Appendix C.

The multi-modal aspect of this formulation was developed in response to sample work period information provided by Seaspan [112]. In the Primavera P6 file provided, it was found that users scheduled work in a variety of daily schedules. Some work was planned on a 24-hr per day schedule, some on an 8-hr per day schedule, and some in 10-hr shifts. Unlike the FMFs, evening and nighttime shifts are quite common for the larger commercial shipyards of ISI and Seaspan, and planning work in those shifts is much preferable to overtime if possible. To incorporate the need to schedule in non-standard times and to have durations that may be shortened by adding evening and nighttime shifts, the formulation was modified such that every mode after mode one represents an incremental addition of extra shifts, and a reduction of duration by one day or time unit. In theory, with standard 8-hr shifts, a reducible activity can be reduced to one third of its normal time by maximizing evening and nighttime shifts. In practice however, it is found that there is a limit

Activity	Duration Mode 1	2	3	4
1	5	4	3	3
2	3	2	2	2
3	2	1	1	1
4	4	3	2	2
5	2	1	1	1
6	1	1	1	1
7	1	1	1	1
8	5	4	3	3
9	5	4	3	3
10	5	5	5	5
11	4	3	2	2
12	4	3	2	2
13	5	5	5	5
14	3	2	2	2

Table 2: A duration table is used to determine durations associated with each mode. It differentiates between reducible and irreducible activities.

to how much of a workforce can work nighttime and evening shifts, so work in these non-standard shifts should be limited.

Another benefit of this formulation is that it uses only data that is currently being used for scheduling at Canadian shipyards, being hourly resource demands for different tasks, predecessor relationships, and activity durations.

Data preprocessing is required to set the d_{jm} parameters for certain activities that cannot be shortened, such as complex work that must be completed by the same people, or work that can only be shortened by a certain amount. In table 2, an example data table shows the underlying durations used for the multiple execution modes of activities *j*; in this case however, only evening shifts are considered, so durations cannot be reduced to more than one half of the standard activity duration. With data processing, activities' duration tables can be suited to the shift types or calendar types that the resources of those activities use. The duration units used in all experimentation in this thesis used days as a minimum duration. Inputs from hourly planning data, such as the information exported from Primavera P6, are rounded up to the nearest day; thus, providing a robust schedule that errs on the side of caution. This would of course be only suitable for planning strategies where the scheduling data is generalized into sets of parallel activities spanning at least a half-day, as seen at the FMFs and ISI, rather than long sequences of small (hourly/minute) duration activities.

5.1.2 Solver Limitations and Implications in the Priority-Duration Formulation

It was found during experimentation that a serious issue can occur with a wide-ranging synthetic objective function like the priority-duration formulation: low priority activities can be given a zero objective function weight as solvers can round relatively very small values down to zero. Many original experimentation results had to be recomputed after a mixed integer linear programming (MILP) gap limitation was found. For example, in the objective function of equation 28, the original θ value used was 10 and α was 2. Although this produces a very large difference in value between priority levels, as desired, it actually caused instability in the results and it was noticed that in some rare

circumstances, priority-3 work would not even be scheduled or would be scheduled in a poorly optimized time unit. Further research into Gurobi online documentation [113] revealed that this could occur, and that Gurobi Optimization recommends that objective function precision limits should be approximately between 10^{-3} and 10^{6}. After completing testing with various values, it was determined that the following scaling modifications to the priority-duration formulation were favorable, and no other problems with poor results from experimentation were observed. Therefore, equation 28 can be populated with θ, α, ε_1, and ε_2 constants equal to 5, 1.1, 0.001, and 0.001 respectively, along with a term multiplier of 100, producing the following equation:

$$Maximize \sum_{m \in \boldsymbol{M}} \sum_{j \in \boldsymbol{J}} \sum_{ES_j}^{LS_j} \frac{100 x_{jtm}}{p_j^5} \left(0.001 + d_{j1}\right)^{1.1} (1 - 0.001t) \qquad (36)$$

This objective function produces a value of approximately 4×10^{-4} for one single priority-3 activity of one single day duration, scheduled at day 999, representing an extreme and unrealistic objective function lower limit; furthermore, the objective function produces a value of 2.7×10^{6} for a project of 1000 activities, all with 20-day durations, all at priority-1, and all scheduled at day zero, representing a very large extreme maximum limit. Both objective function extreme limits fall within the same order of magnitudes suggested by Gurobi Optimization. Due to the solver precision limitations, this formulation is applicable to three priority levels. As more priority levels are introduced, a trade-off should occur in value differentiation between priority levels. For example, with 4 priority levels, θ may need to be reduced from 5 to 4. This may cause an unintended effect where a long-duration lower priority activity may have a larger value than a very short-duration higher priority activity. In a practical sense however, very short activities (one to two days) are easy to complete because they "fit" quite well in-between longer activities in the solution space, as they only require resources to be free for a short duration. They can even be completed "off-the-books" to take advantage of delays in other work, that frees-up workers, caused by weather or other random events, where the completion data is tallied after the fact. This relative lowering in objective function value

for these rare short high-priority activities should therefore not be a significant real-world problem.

It should be noted that with a very large amount of priority levels, the optimal solution will be closer and closer to what a prioritized list in a serial SGS would output, as the serial SGS is the ultimate front-loading priority-based scheduler, since it schedules activities in a prioritized list one-by-one as early as is feasible. Furthermore, as will be seen later, this MILP model can only be solved in a reasonable amount of time for a limited project size and complexity, meaning that if many priority levels are needed, which would be viable for only very large projects, then an exact solving methodology will be unsuitable by itself anyway.

For the experiments conducted in this thesis however, only three priority levels were used, as they accurately represent the priority-levels associated with RCN reporting and planning: ESS=priority-1, HOPP=priority-2, and NOPP=priority-3 [16].

5.1.3 Discrete-Time Priority-Duration RCPSP Model for WO Scheduling with Activity Adjacency

After completing experiment set 3 and continuing research into larger and more complex problems provided by Thales Canada, it was found that the initial scheduling formulation in section 5.1.1 would not find solutions in a reasonable amount of time. For instance, an initial 500-activity problem required 6 minutes to pre-process data even before being sent to a solver, including early start forward pass, running a serial SGS for a time horizon H, completing a backward pass from H, then formatting the data for Gusek consumption, then having Gusek build the lp file. The lp file creation itself took 65 seconds to complete. The Gurobi 9.0 solver could not solve such a large problem within 8 hours and based on discussions with schedulers mentioned throughout this thesis, improving the solution time is of great importance: even often more important than having an optimal schedule. Also mentioned in chapters three and five, is that schedulers were discouraged with automatic scheduling software features because it would often split a WO's activities, so they were no longer adjacent. The used scheduling software either did

not allow an option for adjacency or it was never used. Unless there was no other option, a WO's activities were best left adjacent to each other without breaks. Parallel tasks were normally meant to start simultaneously in a schedule, but the automatic scheduling features were not designed to facilitate this, so manual scheduling retained its dominance. The combination of these factors led to the creation of the following formulation that requires pre-calculation of activity early start times relative to WO start times, without considering resources.

This can be done because as seen in practice, a single WO will never in itself present an infeasible schedule considering resources (i.e. parallel activities are planned using resources such that they can feasibly occur in parallel). Moreover, this method was only tested on problem sets where activities were scheduled with sequential sets of single or parallel activities, without many precedence relationships between activities of different WOs, as is industry practice at the FMFs [3]. No reviewed work period data from the FMFs received for this thesis included a precedence constraint between activities of different WOs in the scheduling data of NSWPPs. If such a relationship existed, then this rare exception was accommodated with managerial intervention (i.e. schedule this work period at time *x* because that is when we expect activity *y* from this other WO to be completed). By simplifying the precedence data structure to what is seen in practice, then this method can find good solutions to larger NSWPP RCPSPs in less time, satisfying the real-world problem where time is very important and activities adjacency is respected, yet benefitting from optimization software. Precedence between WOs can still however be accommodated, but since there are much less of these relationships present, the problem is simplified. WO start times are added as additional variables. Activity start times are fixed relative to the WO start times, and parallel activities with varying durations are set to begin once the latest predecessor is completed. Since this was developed late in the research phase of this thesis, analysis and formulation development was only completed for the most common initial scheduling scenario: without multiple execution modes and no overtime. Below is the modified formulation:

Indices:

1… *A* for activities

1…*W* for WOs

1…*K* for resources types

0…*H* for the time horizon

Sets:

P activity immediate predecessor pairs (*i,j*)

K resources

J activities

W for WOs

AW (*A* cross *W*) Work package assignment for each activity

Parameters:

H integer, the planning horizon from the project start (0)

A integer, number of activities

W integer, number of WOs

d_j integer, duration of activities *j*

r_{jk}integer, activity *j* demand for resource type *k*

$R_k$integer, resource *k* capacity

ES_j integer, earliest start time of activity *j*

LS_j integer, latest start time of activity *j*

ES_w integer, earliest start time of WO *w*

LS_w integer, latest start time of WO *w*

p_j integer, priority of activities (1=ESS, 2=HOPP, 3=NOPP)

θ (≥ 0), used to assign weights to activities proportionately to their priority

ε_1 (≥ 0), very small value to account for milestones that have no actual duration

ε_2 (≥ 0), very small value to penalize late scheduling of activities

α (> 0), use to give tie-breaking benefits to longer duration work (α=1.1) or making each work duration unit equal (α=1.0)

f_j relative start time distance of activity *j* with its WO start time

Variables:

x_{jt} binary, 1 if activity *j* starts at time *t*, 0 otherwise

y_{wt} binary, 1 if WO *w* starts at time *t*, 0 otherwise

Objective function:

$$Maximize \sum_{j \in \boldsymbol{J}} \sum_{ES_j}^{LS_j} \frac{x_{jt}}{p_j^{\theta}} (\varepsilon_1 + d_j)^{\alpha} (1 - \varepsilon_2 t) \qquad (37)$$

Subject to:

$$\sum_{ES_j}^{LS_j} x_{jt} \leq 1 \qquad \forall\, j \in \boldsymbol{J} \qquad (38)$$

$$\sum_{ES_w}^{LS_w} y_{wt} \leq 1 \qquad \forall\, w \in \boldsymbol{W} \qquad (39)$$

- All activities and WOs must be completed no more than once between the respective ES and LS times.

$$\sum_{ES_j}^{LS_j} x_{jt} t \geq \sum_{ES_i}^{LS_i} x_{it}(t + d_i) \quad \forall j \in \boldsymbol{J}, \forall (i,j) \in \boldsymbol{P} \tag{40}$$

- A successor's start time must be greater or equal to a predecessor's start time plus its duration. For this to be a real benefit, this must only consider predecessors whose successor exists in another WO, determined via data pre-processing. In many projects, this constraint may not be used at all.

$$\sum_{j \in \boldsymbol{J}} \sum_{b=\max\{t-d_j+1, ES_j\}}^{\min\{LS_j, t\}} x_{jb} r_{jk} \leq R_k \quad \forall k \in \boldsymbol{K}, \forall t \in 0 \dots H \tag{41}$$

- The sum of resource usage for the period of activity start times to activity end times must not exceed the resource capacity for of each resource type *k*. This considers activities instead of WOs.

$$\sum_{ES_w}^{LS_w} t \times y_{wt} = \mathrm{t} \times x_{jt} - f_j \qquad \forall w \in \boldsymbol{W}, \forall j \cup w \in \boldsymbol{AW} \tag{42}$$

- Activity start times maintain a pre-calculated CPM start time based on respective WO start times; thus, enforcing adjacency and AEAP scheduling. Once a WO starts, it is supported until conclusion without resource-related delays.

$$x_{jt} \in \{0,1\} \qquad \forall j \in \boldsymbol{J}, t \in ES_j \dots LS_j \tag{43}$$

$$y_{wt} \in \{0,1\} \qquad \forall w \in \boldsymbol{W}, t \in ES_w \dots LS_w \tag{44}$$

- Start times are binary, thus a variable exists for every time unit between each activity's ES and LS.

Results from experimentation with this formulation are found in experiment set 4; furthermore, the formulation in Gusek and a data format example is found in Appendices E and F respectively.

5.2 Proposed Formulations for the Re-Scheduling Problem

From discussions with schedulers, PLs, and PMs [3] [4], it was confirmed that higher priority work packages are more important to keep on a firm schedule than lower priority ones, and minimizing these changes with an emphasis on priority is desirable. To incorporate this into a discrete time RCPSP formulation format, it is proposed in this thesis to add a priority term to the objective function described by Van De Vonder (2006) [59] as a common re-scheduling goal, as indicated in the following equation, where s_j^0 represents the start time of activity *j* in the initial schedule *0*:

$$Minimize: \sum_{j \in \boldsymbol{J}} \sum_{t \in \boldsymbol{H}} \frac{x_{jt}}{\boldsymbol{p_j^\theta}} \left| x_{jt} t - s_j^0 \right| \tag{45}$$

This formulation objective function is however not linear and must thereby be modified with constraints to ensure that it is linear. As suggested by Granger, Yu and Zu (2017) [114], this can be done by adding additional constraints to handle positive and negative deviations of the absolute term. This technique is applied to this model for the following key formulae:

$$Minimize: \sum_{j \in \boldsymbol{J}} \sum_{t \in \boldsymbol{H}} \frac{u_j}{p_j^\theta} \tag{46}$$

Subject to:

$$u_j \geq x_{jt}(t - S_j) \quad \forall j \in \boldsymbol{J}, \forall t \in \{ES_j \dots LS_j\} \tag{47}$$

$$u_j \geq x_{jt}(S_j - t) \quad \forall j \in \boldsymbol{J}, \forall t \in \{ES_j \dots LS_j\} \tag{48}$$

Where *t* is the time at which activities x_{jt} may begin and S_j is the original starting time of activity *j*. u_j becomes the new variable that is a maximum positive value representing the absolute change (deviation-days) between S_j and *t*.

5.2.1 Multi-Mode Re-scheduling DT Priority RCPSP Model, Version A: Deviations days

1… *A* for activities

1…*K* for resources types

0…*H* for the time horizon

Sets:

P activity immediate predecessor pairs (*i,j*)

K resources

J activities

M activity execution modes *m*

Parameters:

H integer, the planning horizon from the project start (0)

A integer, number of activities

d_{im} and d_{jm} integer, duration of activities *i* and *j*, depending on mode *m*

r_{jk}integer, activity *j* demand for resource type *k*

$R_k$integer, resource *k* capacity

ES_j integer, earliest start time of activity *j*

LS_j integer, latest start time of activity *j*

p_j integer, priority of activities (1=ESS, 2=HOPP, 3=NOPP)

θ (≥ 0), used to assign weights to activities proportionately to their priority

ε_1 (≥ 0), very small value to account for milestones that have no actual duration

ε_2 (≥ 0), very small value to penalize late scheduling of activities

α (> 0), use to give tie-breaking benefits to longer duration work (α=1.1) or making each work duration unit equal (α=1.0)

S_j Initially scheduled activity start-dates for each activity *j*

β_{M1} total alternative shift limit

β_{M2} alternative mode reduction limit

Variables:

$x_{jtm} = \begin{cases} 1 & \text{if activity } j \text{ starts at time } t \text{ in mode } m \\ 0 & \text{otherwise} \end{cases}$

u_j represents positive deviation-days for activity j

$$Minimize: \sum_{m \in \boldsymbol{M}} \sum_{j \in \boldsymbol{J}} \sum_{ES_j}^{LS_j} \frac{u_j}{p_j^{\theta}} \tag{49}$$

Subject to:

$$\sum_{m \in \boldsymbol{M}} \sum_{ES_j}^{LS_j} x_{jtm} \leq 1 \quad \forall j \in \boldsymbol{J} \tag{50}$$

$$\sum_{m \in \boldsymbol{M}} \sum_{j \in \boldsymbol{J}} \sum_{b=\max\{t-d_{jm}+1, ES_j\}}^{\min\{LS_j, t\}} x_{jbm} r_{jk} \leq R_k \quad \forall k \in \boldsymbol{K}, \forall t \in 0 \dots H \tag{51}$$

$$\sum_{m \in \boldsymbol{M}} \sum_{ES_j}^{LS_j} x_{jtm} t \geq \sum_{m \in \boldsymbol{M}} \sum_{ES_i}^{LS_i} x_{itm}(t + d_{im}) \quad \forall j \in \boldsymbol{J}, \forall (i,j) \in \boldsymbol{P} \tag{52}$$

$$u_j \geq \sum_{m \in \boldsymbol{M}} x_{jtm}(t - S_j) \quad \forall j \in \boldsymbol{J}, \forall t \in \{ES_j \dots LS_j\} \tag{53}$$

$$u_j \geq \sum_{m \in \boldsymbol{M}} x_{jtm}(S_j - t) \quad \forall j \in \boldsymbol{J}, \forall t \in \{ES_j \dots LS_j\} \tag{54}$$

$$\sum_{j \in \boldsymbol{J}} \sum_{ES_j}^{LS_j} (x_{jt2} + 2x_{jt3} + \cdots + (M-1)x_{jtM}) \leq \beta_{M1} \tag{55}$$

$$\sum_{j \in \boldsymbol{J}} \sum_{ES_j}^{LS_j} (x_{jt2} + x_{jt3} + \cdots + x_{jtM}) \leq \beta_{M2} \tag{56}$$

$$x_{jtm} \quad \in \{0,1\} \tag{57}$$

This formulation will minimize deviation days from an initial schedule after a disruption and can be very useful, but it does not directly reduce number of deviations (vs total deviation-days). Another formulation was developed and used in experiment set 2 to determine whether the deviation-days objective or the deviations objective were more suitable.

5.2.2 Multi-Mode Re-scheduling DT Priority RCPSP Model, Version B: Deviations

When speaking with scheduling staff, the question was asked: "would you prefer having one WO shifted by 20 days? Or, would you prefer having 10 WOs shifted by two days each?" Both scenarios would produce the same number of deviation days, but the interviewees indicated that it is a lot more work to re-plan, re-schedule, coordinate with shop supervisors etc. for 10 WOs than it is for one WO. If the single activity being shifted 20 days therefore maintained a feasible schedule, then this scenario would be preferred. Thus, for this case, the objective function may be modified to the following, with complimentary constraints to linearize it:

$$Minimize: \sum_{m \in \boldsymbol{M}} \sum_{j \in \boldsymbol{J}} \sum_{ES_j}^{LS_j} \frac{y_j}{p_j^{\theta}} + \varepsilon \sum_{m \in \boldsymbol{M}} \sum_{j \in \boldsymbol{J}} \sum_{ES_j}^{LS_j} x_{jtm} t \tag{58}$$

The first term will add one unit divided by the priority-term for every deviation, regardless of the deviation's magnitude. Note that the second term is necessary to ensure that any deviated activities are completed as early as possible. The second term in the objective function is the same slack reduction term used in section 3.2.1. From experimentation, it was found that without this term, the linear solvers removed problematic activities or sets of activities and re-scheduled them very far in the future. ε must be a small value to prevent the slack reduction term from outweighing the primary objective of preventing schedule changes. ε = 0.001 was used in experiment set 2, thus ensuring that this second term does not outweigh the first term. Along with the objective

function, the constraints from the complete formulation in section 5.2.1 are also modified to include the following two equations:

Subject to:

$$y_j \geq \frac{u_j}{N} \quad \forall j \in \boldsymbol{J} \tag{59}$$

$$x_{jtm},\ y_j \quad \in \{0,1\} \tag{60}$$

Using the Big *M* technique described by Hillier and Lieberman (2015) [25], a large constant *N* is used that will direct y_j to be equal to its binary form for each activity *j*, where a value of one represents any magnitude of activity change from the original schedule and a value of zero represents no changes to the original schedule. For experimentations using this formulation, *N* was given a value of 500 as no single deviation could come close to 500 since the time horizon for experimentations was limited to 240 time-units (days).

Although the formulations proposed for the re-scheduling scenarios produce the least (optimal) number of either start-time deviation-days or start-time deviations after changes have occurred to the project, by setting β_{M1} to zero, it was discovered in experiment set 2 that even optimal solutions without using alternative shifts are often poor in terms of producing too many scheduling changes. This became worse the earlier that the disruption occurred in a project. Experiment set 3 was conducted using the multi-mode formulations with β_{M1} set to one, to determine which activity will have the most impact when reduced in duration; thus, guiding a user's overtime investigation to the most worthwhile activities.

5.3 Random Problem Generation with NSWPP-Inspired Topologies

Due to the lack of NSWPP-inspired project network topologies in the literature, a fictitious ship was created, inspired by the HAL Class Frigate, where work using two types of resources were featured. The first type of resource, which is very common in literature, is that of fixed-capacity shop resources, while the second resource introduced is that of compartment resources. The fictitious ship, illustrated in figure 23, includes 40

compartments, representing a scaled-down version of the HAL Class Frigate that has 353 compartments.

SS and repair facility (RF) workers typically call compartments by their functional names, such as the “operations room” or “machinery control room”, but all spaces are also given an alphanumeric naming system that represents first the deck (numbers), then secondly the watertight bulkhead frame (letters). Not shown in figure 23, are the names of individual compartments within watertight bulkhead sections. For example, watertight compartment 4D might have multiple compartments within this watertight section, with space numbering labelled 4D01, 4D02, 4D03A, 4D03B, etc. Moreover, the fictitious ship is only drawn and used in two dimensions, but NSSs are built in three dimensions and have non-watertight compartments on either port or starboard sides that may be affected by conflict-causing activities like hot work or passageway deck repairs. Although

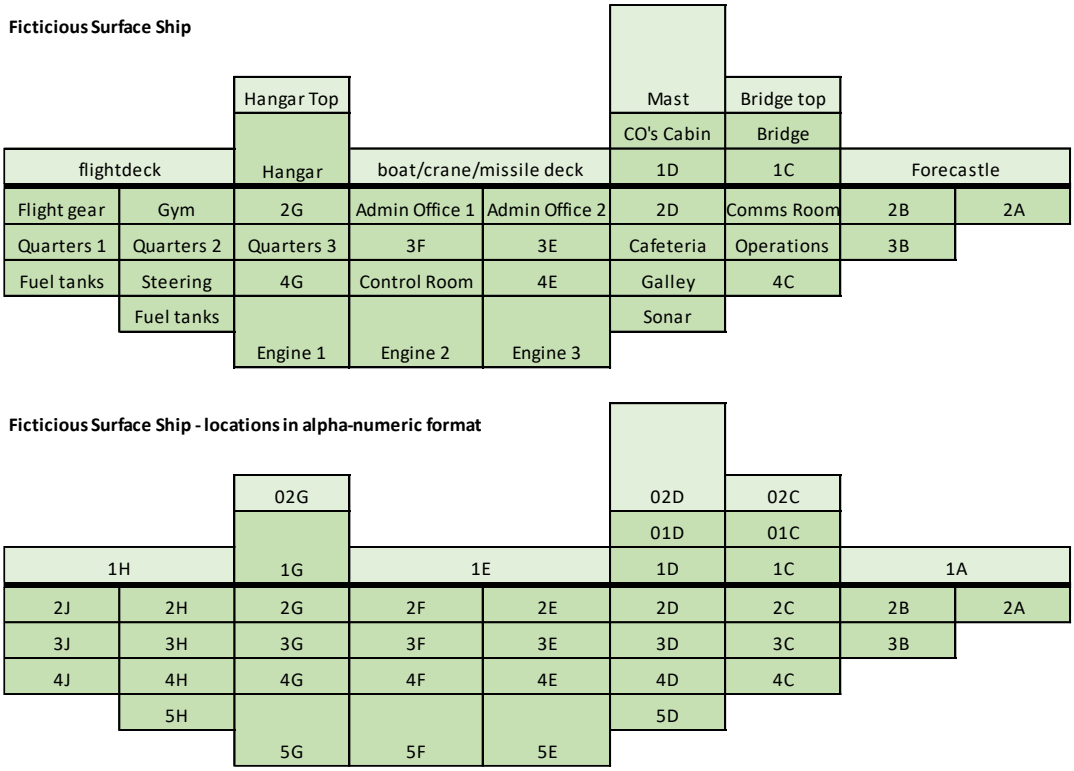


Figure 23: This fictitious ship is used to simulate compartment resources in randomly generated NSWPP-inspired RCPSPs. It is inspired by the RCN’s Halifax Class Frigate.

the model is simplified, the insights from experimentation are transferable to a larger project in three dimensions.

FMFCS has 70 shops that may provide worker resources, so a scaled-down version of this was used to mimic a simplified version where an ISS contractor is the primary agent that manages eleven additional internal and external resources, up to a certain daily capacity level, representing hours. Table 3 shows the resource table used for experiment sets 1, 2, and 3.

Activities used for this random problem generator are in the form of multiple activities estimated to occur in parallel, called a WO or work package.

To gauge problem complexity, perform experiments, and gather statistically significant trends, a problem generator was created to build numerous problems, which were then used in experiment sets 1, 2, and 3. The problems shared the following random characteristics:

- All scheduling problems were defaulted to 100 activities (as WOs) for experiment sets 1 and 2, although experiment set 3 varied the number of activities.
- Sets of WOs are created with one to four WOs, each in sequential precedence order. For each random set: 40% have one WO, 30% have two WOs in sequence, 20% have three WOs in sequence, and 10% have four WOs in sequence.
- All sets have one common main compartment for each WO within the set, as precedence constraints for work involving different primary spaces are rare in NSWPPs.

Resource	Capacity	
02C	1	
02D	1	
02G	1	
01C	1	
01D	1	
1A	1	
1C	1	
1D	1	
1E	1	
1G	1	
1H	1	
2A	1	
2B	1	
2C	1	
2D	1	
2E	1	
2F	1	
2G	1	
2H	1	
2J	1	
3B	1	
3C	1	
3D	1	
3E	1	
3F	1	
3G	1	
3H	1	
3J	1	
4C	1	
4D	1	
4E	1	
4F	1	
4G	1	
4H	1	
4J	1	
5D	1	
5E	1	
5F	1	
5G	1	
5H	1	
1001	7	Crane
1002	14	Load banks
1003	10	Jetty Spaces
1004	80	Thales team 1
1005	80	Thales team 2
1006	80	Thales team 3
1007	160	FMF1
1008	160	FMF2
1009	160	FMF3
1010	160	FMF4
1011	160	FMF5

Table 3: Resource table used for random problem generation.

- 30% of WOs have one random adjacent space; 30% of those with one adjacent space have an additional different adjacent space.
- Each set of one to four WOs have priorities assigned, and are generated as 50% priority-1, 30% priority-2, and 20% priority-3.
- Additional non-space resources were assigned randomly: 80% use one random resource of a random quantity between 0% and 50% of that resource capacity per time unit; of those, an additional resource is assigned to 50% of WOs with the same resource consumption probabilities.

5.4 Early Start and Late Start Calculation Adjustment for Multi-Mode Formulations

An important point to consider is that, as discovered via experimentation, the ES times for activities when using multiple modes that reduce duration should be adjusted. The reason for this is that if a predecessor activity finishes earlier because it is active in a duration-reduced mode, then the successor activity should be able to start earlier than in the initial ES assignment, as long as resources can support it. For experimentation in experiment set 3, the ES calculated when using multiple duration modes was set to 0 or the present day. The present day is the early start time given for all activities that have not yet started in the re-scheduling scenario. For example, if re-scheduling occurs at day-22 of a project, then it is impossible to start any activities that have not yet started any earlier than day-22, so the default ES will be 22. In the re-scheduling scenario with multiple modes, activities that are in-progress have ES and LS equal to their original start times from the initial scheduling solution, while activities that have not yet started have ES times equal to the present day. This increase in x_{jtm} values due to a more relaxed ES time did not significantly increase solve times, likely due to the low-precedence network topology of NSWPPs.

A late start (LS) based on a very conservative date in the far future will however increase the time required to generate the .lp files due to the larger number of variables created. For larger problems, there is benefit in using a serial SGS to first build a feasible schedule that creates a reasonably tight time horizon H, a forward and backward pass will

quickly determine a LS for each activity. Details on how the forward and backwards pass were calculated with Excel-VBA may be found in Appendix A, where a serial SGS is also used to calculate a reasonable time horizon H, for the backward pass to calculate a tight LS.

5.5 Experiment Assessment Criteria

Additional thought is needed to determine how to best assess the quality of schedules being produced in experimentation. In NSWPPs, due to the variability and scope growth that occurs, it is generally not good to optimally schedule all work with the simple goal of minimizing makespan. For example, a solution that produces a very small makespan might schedule long-duration and high-priority activities to finish on the last day of a project; however, makespan is still important and will be considered when evaluating projects, as a shorter makespan is a simple and classic way of evaluating solutions. It is not however the only or most important criterion. Other criteria to be used are: Priority-1 buffer, average priority-1 duration-weighted centroid (DWC), solution time, total deviations, and total deviation days.

Priority-1 buffer:

The naturally induced priority-1 buffer is defined as the percentage of the makespan where only priority-2 and priority-3 activities are scheduled, and is described mathematically in this thesis as:

$$\frac{Project\ end\ date - (finish\ time\ of\ last\ priority\_1\ activity)}{Project\ makespan} \qquad (61)$$

This buffer is an indication of project risk: a large score means that the last priority- 1 activity ends far from the end of the project meaning that there is little risk for the non-completion of a priority 1 activity. However, it might not be sufficient at describing the relative volume of high-priority activities earlier in a project. For example, as seen in figure 24, there is a fictitious project where a single priority-1 activity trails almost the full length of a project's makespan, but the bulk of priority-1 activities occur very early-on. The priority-1 buffer metric will show that the project is relatively risky, but

an experienced project manager would be very comfortable with this project when reviewing the Gantt Chart, because there is only one activity requiring significant

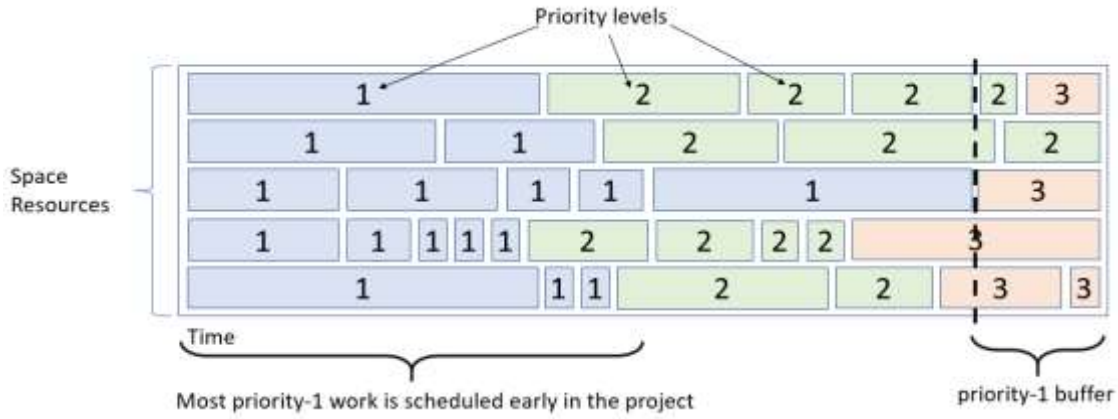


Figure 24: A priority-1 buffer is not always a suitable indication of a project's risk.

managerial attention for the last 50% of the project. Thus, the next criterion is introduced.

Average priority-1 duration-weighted centroid (DWC):

To assess how well a project places the highest priority activities earlier in a project, it is proposed in this thesis to use the average priority-1 duration-weighted centroid (DWC) evaluation criterion. This criterion can be mathematically represented by equation 62, where n_{pri_1} represents the number of priority-1 activities, x_{start} represents the priority-1 start times, x_{finish} represents the priority-1 finish times, d_j represents the duration of each activity:

$$Avg\ priority\ 1\ centroid = \sum_{j \in \boldsymbol{J}} \frac{x_{start} + x_{finish}}{2 \times n_{pri\_1}} d_j \qquad (62)$$

This criterion will give a result that can be anecdotally described as the central positioning of the average activity in a project in terms of time units and weighted by the duration of activities. For example, an average priority-1 centroid of 300 days represents that the average priority-1 activity may be approximately 10 days in duration and positioned about day-30 of a project. Figure 25 is an illustration of a sample fictitious project where the darker blue rectangle represents the average positioning and size of a priority-1 activity, along with its distance from the project start time. The distance between the project start and the centre of this average activity is the centroid. When the centroid is multiplied by this average activity's duration, this is the average pri-1 DWC.

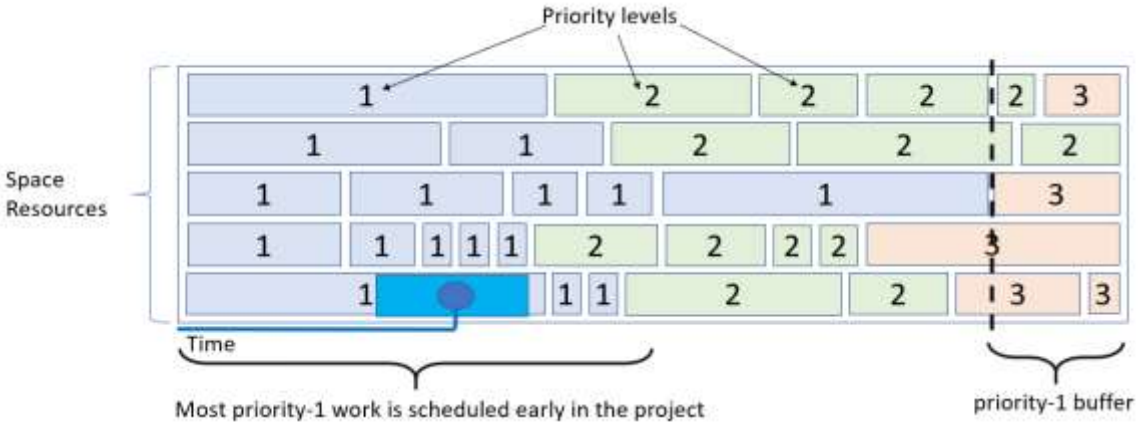


Figure 25: Illustrating the average position and size of a priority-1 activity over a project makespan.

The reason that duration should be considered in this assessment criterion is that without a duration-weight, there could be a schedule solution that appears good where several short-duration but high-priority activities are scheduled earlier, but a smaller number of longer-duration high-priority activities are scheduled later. These activities, without a duration-weight, would all be worth the same, and the resultant centroid value would be better (smaller). This could however represent a poor schedule in practice because these long-duration activities, that are typically much more complicated and difficult to manage than smaller-duration activities, would be given an equivalent weight and might be scheduled later.

The ratio between the average priority-1 DWC criterion and a project's makespan can be further used to provide a good indication of the risk that a project will not complete a priority-1 activity. In the same example above, where the priority-1 DWC is 300 days^2, if the project makespan is 100 days and recall that the average activity duration is 10 days, then it can be said that the average priority-1 activity is 10 days duration and positioned about day-30 of the project. As this value approaches day-50, the positioning of priority-1 activities is even across the project's makespan, representing a poor front-load of priority-1 activities. The lower this value, the better.

Total deviation days

For the re-scheduling scenarios however, the DCW criterion is unsuitable when the goal is to reduce disruptions. The assessment criterion primarily used in experiment sets 2 and 3 is the one proposed by Van de Vonder (2007) [70]: total deviation days between the original schedule and the new schedule after changes in a project's activities have occurred (see section 3.6). To properly assess the total number of work deviations (instead of deviation days) in experiment set 2, the assessment criterion described mathematically in the following equation is used:

$$deviations = \sum_{j \in J} \begin{cases} 1 & if \sum_{t \in H} \left| x_{jt} t - s_j^0 \right| > 0 \\ 0 & Otherwise \end{cases} \quad (63)$$

Experimentation and result analyses used to set-up subsequent experiments, using the aforementioned formulations, heuristics, and assessment criteria, are found in the next chapter.

Chapter 6: Experiments and Results Analysis

The experimentation described in this thesis was completed by iteratively performing experiments, analysing, and discussing results, then using these results and insights to set up subsequent experiments. For this reason, the results analysis, discussion, and motivation for the subsequent experiments are presented in this chapter as subsequent sections to facilitate reader comprehension.

A small preliminary experiment set was first conducted to gauge the difficulty of RCPSPs with topologies that are similar to real-world NSWPP RCPSPs. This preliminary experiment set was conducted on a large problem set with an increasing number of work packages and therefore difficulty. The same activities were present in each successive trial, such that adding more activities guarantees that the problem difficulty increases; however, no statistically significant results could be obtained because the experiment samples are correlated. This preliminary trial compared nine serial and parallel SGS sorting methods, and some obvious anecdotal results were observed. These include that the problem is indeed NP-hard and therefore solve time increases exponentially, while heuristic serial and parallel SGS methods solve much faster. Only priority-based formulations, such as the initial scheduling formulation presented in this thesis, and priority-list ordered SGS sorting methods resulted in competitive average priority-1 DWCs; thus, only these methods will be used for the main experiment sets. The makespan minimization formulation with and without slack reduction is unsuitable for the goal of minimizing pri-1 DWCs and performs poorly. This is intuitively expected since it only aims to minimize makespan and/or reduce slack but gives no importance to activity priorities.

Figure 26 shows that as constrainedness in resource demand and availability increase, by increasing the number or activities in increments of 20, exact formulations can reach long solution times while the SGS methods tested solved much faster at the cost of pri-1 DWC optimality. The serial SGS methodology appeared to perform as well as parallel SGSs, and since it more closely mimicked human manual scheduling and solved in

half the time of parallel SGSs, it was chosen for comparative purposes in future experiments.

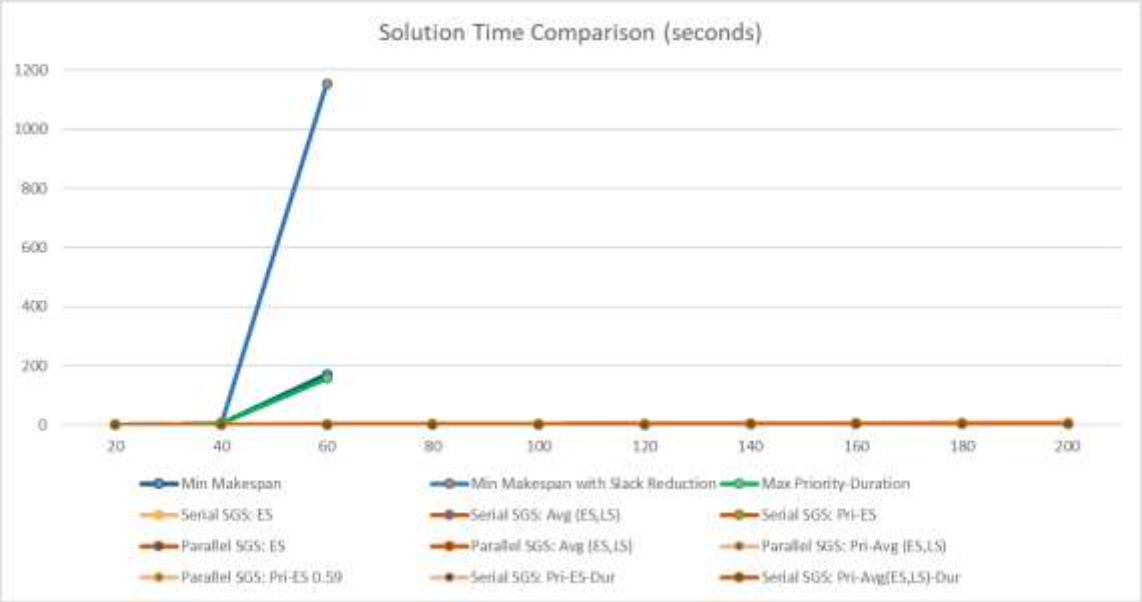


Figure 26: Solution times, in seconds, for various exact and heuristic techniques.

Although not part of the preliminary experiment, the shifting bottleneck procedure will be considered in experiment set 1, as well as two versions of the priority duration formulation that include a duration bias (α=1.1 in equation 28) or not (α=1.0). To recall the effects of the duration bias, please refer to section 5.1.1.

6.1 Numerical Experiment Set 1: Priority-Duration for Initial Scheduling with 100-Activity Sets

As described in section 5.3, random problems were created with a fictitious ship and resource capacities. 30 randomly generated problems were used to determine statistically significant advantages of formulations and heuristics when compared to each other, for those formulations and heuristics that showed potential from the results analysis of the preliminary experiments. In order to compare with the SGS heuristics that used priority sorting schemes, the formulation mode settings used for the priority-duration formulation were only the base mode (i.e. β_{M1} and β_{M2} set to zero).

In this experiment set, the shifting bottleneck procedure is added to the mix of comparable heuristics with a priority respecting parallel SGS that will only schedule lower

priority activities after higher priority activities are all scheduled. This heuristic method is titled the “PSGS(SBP)” in the experiment set.

A priority-ES-then shifting bottleneck sort method was also used with a serial SGS to see if promising results could be found. This was labelled “SSGS(ES-SBP)”. In this method, the shifting bottleneck procedure is performed successively to groups of activities with the same priority and ES.

To test for statistical significance between solution methods, a standardized paired t-test was performed, where the relative differences between solutions is assumed to be normally distributed. For more information on testing for statistical significance, please refer to Sirkin (2005) [115].

The naming convention for all 8 methods analyzed is as follows:

- The multi-mode DT priority-duration formulation (section 5.1.1) with a duration bias (α=1.1): **PD(α=1.1)**
- The multi-mode DT priority-duration formulation (section 5.1.1) without a duration bias (α=1.0): **PD(α=1.0)**
- The serial SGS, sorted 1st by priority, then 2nd by the average of both the ES and LS: **SSGS(aESLS)**
- The serial SGS, sorted 1st by priority, then 2nd by ES: **SSGS(ES)**
- The serial SGS, sorted 1st by priority, then 2nd by the average of both the ES and LS, then 3rd by duration: **SSGS(aESLS-D)**
- The serial SGS, sorted 1st by priority, then 2nd by ES, then 3rd by duration: **SSGS(ES-D)**
- The serial SGS, sorted 1st by priority, then 2nd by ES, then 3rd by the shifting bottleneck procedure: **SSGS(ES-SBP)**
- The parallel SGS, sorted 1st by priority, then 2nd by the shifting bottleneck procedure: **PSGS(SBP)**

The results in terms of makespan depicted in table 4 show no statistically significant difference between adding a duration bias or not (α=1.0 or 1.1) for the priority-duration formulation, with an average makespan difference of 0.026% in favor of removing the duration bias. The exact formulations in the first two columns produced on average a makespan that was 2.2%±1.7% smaller than the Parallel SGS: shifting bottleneck heuristic, with a 90% confidence interval.

Makespan

	PD(α=1.0)	PD(α=1.1)	SSGS(aESLS)	SSGS(ES)	SSGS(aESLS-D)	SSGS(ES-D)	SSGS(ES-SBP)	PSGS(SBP)
Trial 1	121	120	123	128	123	123	117	120
Trial 2	133	134	147	135	148	147	148	136
Trial 3	104	103	103	109	109	103	109	103
Trial 4	115	115	115	115	115	115	115	121
Trial 5	112	116	130	128	128	130	130	116
Trial 6	119	119	120	119	119	120	119	119
Trial 7	120	120	120	124	120	120	120	120
Trial 8	130	135	143	145	152	143	152	141
Trial 9	101	97	116	116	116	116	116	102
Trial 10	97	90	98	92	97	98	100	100
Trial 11	122	122	122	122	122	122	122	122
Trial 12	155	155	155	158	158	155	156	155
Trial 13	147	147	147	149	148	150	149	163
Trial 14	128	128	128	130	128	128	142	129
Trial 15	146	146	146	146	146	146	161	155
Trial 16	108	108	130	120	130	130	120	120
Trial 17	115	115	117	115	115	117	115	116
Trial 18	152	149	163	163	163	163	163	161
Trial 19	102	102	114	113	113	114	117	113
Trial 20	138	138	145	141	149	145	157	144
Trial 21	133	133	139	133	133	139	133	133
Trial 22	152	152	152	152	152	152	152	152
Trial 23	113	119	127	130	127	127	127	130
Trial 24	130	130	149	145	144	149	145	110
Trial 25	115	116	127	122	123	127	125	118
Trial 26	169	169	168	168	168	168	168	168
Trial 27	119	119	121	118	119	119	120	118
Trial 28	159	159	159	159	159	159	159	159
Trial 29	165	165	165	165	165	165	165	165
Trial 30	172	172	172	172	172	172	172	172
average	129.73	129.77	135.37	134.40	135.37	135.40	136.47	132.70

Table 4: Makespan (in days) for both new formulations and priority based SGSs. The green gradient shade shows the best performance (shortest) across the same problems.

The PSGS(SBP) was found to produce a solution makespan that was 2.3% shorter than the average Serial SGS, but due to the large variability of the results, this was found to not be statistically significant. Finally, the various serial SGSs tested showed very little difference between them on average (<1%) with no statistical significance between any two heuristic methods for makespan differences. In the very worst-case scenario (trial #23), the PSGS(SBP) produced a makespan that was 17 days longer or 15% longer than that of the Priority-Duration formulation. A summary of these results may be found in table 5.

Makespan

PD (α=1.0)	≈	PD (α=1.1)	<	PSGS(SBP)	<=	Avg SSGS
not significant			2.2 ± 1.7%		2.3%, but not significant	
Worst Case for any SGS vs Best Exact				PD (α=1.0)	<	PSGS(SBP)
					17 days or 15.0%	

Table 5: Makespan differences between formulations and priority-based SGSs (in days) summarized.

These results show that if the only criterion is makespan, then the ES priority-rules based serial and parallel SGSs that first schedule by priority are competitive with the exact formulation priority-duration model.

Table 6 presents the results obtained and used to compare the naturally induced priority-1 buffers. Standard t-tests were performed between instances and no one scheduling methods was found to produce a statistically significant different buffer when compared to each other over the sample size, considering the large variablility inherent in these randomly generated problems and resulting solutions.

As described earlier, the more priority-1 buffer, the better. These results show that the exact formulations do not produce statistically significant better priority-1

Average Naturally Induced Priority-1 Buffer (%)

	PD(α=1.0)	PD(α=1.1)	SSGS(aESLS)	SSGS(ES)	SSGS(aESLS-D)	SSGS(ES-D)	SSGS(ES-SBP)	PSGS(SBP)
Trial 1	31.4	31.4	32.5	35.2	37.6	32.5	29.1	30.0
Trial 2	13.5	26.3	21.8	14.8	22.3	21.8	22.3	15.4
Trial 3	1.0	16.3	0.0	0.0	0.0	0.0	0.0	2.8
Trial 4	8.7	8.7	8.7	8.7	8.7	8.7	8.7	13.2
Trial 5	33.0	35.3	32.3	37.5	35.9	32.3	36.9	31.9
Trial 6	49.6	49.6	46.7	42.9	42.9	46.7	44.5	49.6
Trial 7	55.8	55.0	50.8	52.4	50.0	52.5	55.0	55.8
Trial 8	35.6	35.6	39.2	39.3	42.8	39.2	42.8	38.3
Trial 9	17.8	17.0	28.4	28.4	28.4	28.4	28.4	18.6
Trial 10	25.8	32.7	26.5	10.9	25.8	26.5	28.0	28.0
Trial 11	50.8	50.8	50.8	50.8	50.8	50.8	50.8	50.8
Trial 12	25.8	27.7	25.8	26.6	26.6	25.8	26.3	24.5
Trial 13	38.8	38.8	39.5	34.9	39.8	39.3	34.9	34.6
Trial 14	14.8	14.8	14.8	12.3	14.8	14.8	23.2	14.7
Trial 15	33.6	33.6	33.6	33.6	33.6	33.6	39.8	37.4
Trial 16	17.6	17.6	31.5	25.8	31.5	31.5	25.8	25.8
Trial 17	42.6	42.6	43.6	39.1	42.6	43.6	42.6	43.1
Trial 18	49.3	48.3	52.8	52.8	52.8	52.8	52.8	52.2
Trial 19	39.2	43.1	40.4	39.8	45.1	40.4	47.0	45.1
Trial 20	29.7	29.7	28.3	29.1	32.9	28.3	38.2	32.6
Trial 21	27.8	27.8	26.6	27.8	27.8	26.6	27.8	27.8
Trial 22	44.1	44.1	44.1	44.1	44.1	44.1	44.1	44.1
Trial 23	61.1	63.0	62.2	56.9	55.9	62.2	55.9	66.2
Trial 24	34.6	34.6	43.0	41.4	41.0	43.0	41.4	22.7
Trial 25	16.5	17.2	24.4	21.3	22.0	24.4	23.2	18.6
Trial 26	53.3	53.3	53.6	53.6	53.6	53.6	53.6	53.6
Trial 27	42.9	42.9	43.8	42.4	42.9	42.9	43.3	42.4
Trial 28	19.5	19.5	19.5	19.5	19.5	19.5	19.5	19.5
Trial 29	57.6	57.6	57.6	57.6	57.6	57.6	57.6	57.6
Trial 30	31.4	31.4	31.4	31.4	31.4	31.4	31.4	31.4
average	33.44	34.88	35.14	33.70	35.36	35.16	35.83	34.28

Table 6: A summary of naturally induced priority-1 buffers for the experiment set. No method produces statistically significantly more buffer.

buffers. However, figure 27 shows that there is a correlation between average makespan and average buffer size. This is also intuitive, as with two projects having the same priority-1 activity distribution, if only the makespan changes due to having less efficiently scheduled lower priority activities, then the priority-1 buffer should grow with a larger

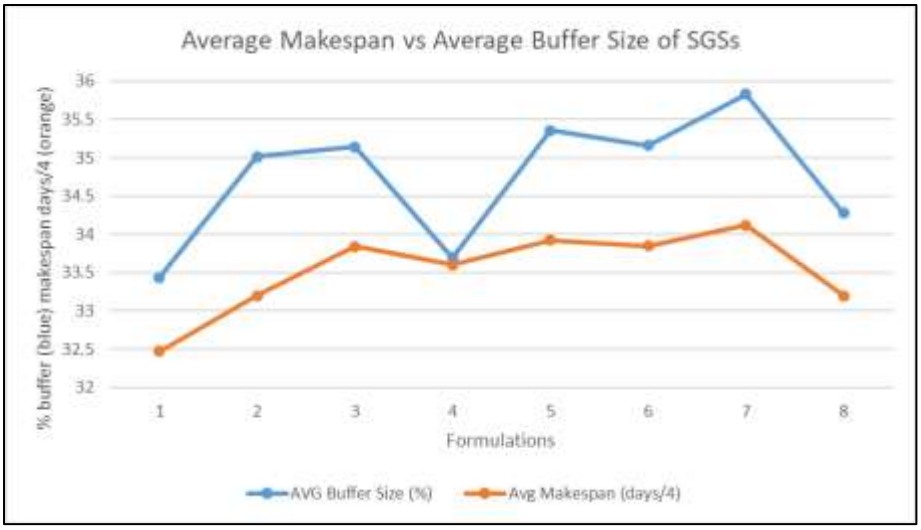


Figure 27: Makespan is correlated with priority-1 buffer size.

makespan. Another observation made from these results is that even though these projects have approximately 50% of activities being non-priority-1, there are still problems where the solutions produce only very small priority-1 buffers, see trials 3 and 4 in table 6. For this reason, if a user is interested in guaranteeing that a high-priority buffer is obtained, then care must be taken to ensure that sufficient lower priority activities are included in the project that use the same resources (spaces and shops) as for the priority-1 work. Getting a desired priority buffer is likely an iterative process requiring re-scheduling the initial schedule with different sets of available WOs and this may never produce the desired buffer if the makespan is too tight for the mix of ESS work in a project. Recall however that the best assessment criterion to determine solution robustness is the priority-1 DWC.

When assessing for priority-1 DCW, the exact priority-duration formulations produce the best priority-1 duration-weighted centroids for this problem set, as seen in table 7.

Priority 1-Duration-Weighted Centroid

	PD(α=1.0)	PD(α=1.1)	SSGS(aESLS)	SSGS(ES)	SSGS(aESLS-D)	SSGS(ES-D)	SSGS(ES-SBP)	PSGS(SBP)
Trial 1	231.4	236.9	240.5	250.7	246.1	240.5	238.4	239.2
Trial 2	292.4	292.4	302.9	318.8	309.0	302.9	314.5	294.1
Trial 3	323.8	325.5	328.3	335.6	335.5	328.3	337.5	328.4
Trial 4	283.2	283.9	294.9	288.2	295.4	286.9	288.2	291.2
Trial 5	302.5	303.3	327.2	336.9	360.7	327.2	347.5	325.6
Trial 6	186.0	186.2	214.7	224.8	227.0	214.7	234.1	201.5
Trial 7	171.9	173.0	179.7	189.8	194.5	179.0	183.1	177.0
Trial 8	268.7	270.9	283.6	287.8	288.2	283.6	286.2	288.0
Trial 9	260.7	262.0	298.2	291.0	318.3	307.8	289.0	281.0
Trial 10	229.1	229.1	252.7	262.7	256.7	252.7	268.3	255.2
Trial 11	219.0	219.9	230.3	228.9	232.7	226.3	224.4	223.7
Trial 12	311.8	311.8	357.4	337.3	331.8	357.4	333.4	324.8
Trial 13	336.5	336.5	342.1	352.2	349.8	346.3	352.2	348.2
Trial 14	342.9	343.7	357.0	387.7	349.1	357.0	359.6	362.4
Trial 15	284.3	284.3	346.8	316.5	316.6	346.8	304.9	290.9
Trial 16	208.2	208.3	222.1	220.8	221.7	222.1	228.6	210.1
Trial 17	163.3	164.7	176.5	179.6	175.6	176.5	186.6	172.5
Trial 18	177.9	178.0	187.2	182.9	188.2	187.2	187.8	187.8
Trial 19	200.2	200.5	230.4	227.0	220.2	230.4	227.8	208.4
Trial 20	283.3	288.7	315.2	322.4	331.3	315.2	330.3	301.7
Trial 21	263.7	265.5	291.2	311.6	291.0	291.2	311.6	273.3
Trial 22	332.5	332.5	335.8	336.6	335.8	335.8	336.6	336.6
Trial 23	135.4	135.4	143.4	147.4	147.4	143.4	147.4	135.4
Trial 24	213.1	214.5	220.5	227.6	231.2	220.5	227.6	220.0
Trial 25	327.5	329.6	346.0	362.6	352.6	346.0	351.4	342.9
Trial 26	194.7	194.7	208.4	210.6	212.3	208.4	210.6	208.9
Trial 27	229.5	229.9	241.6	263.3	231.6	259.5	255.1	248.5
Trial 28	416.9	418.5	437.8	448.8	445.2	437.8	448.8	437.0
Trial 29	222.8	222.8	224.0	226.4	222.8	224.0	222.9	222.9
Trial 30	283.1	284.8	288.1	301.8	301.3	288.1	300.0	287.0
Average	256.54	257.59	274.15	279.27	277.31	274.78	277.82	267.47

Table 7: Average Priority-1 DWC in days2 for the experiment set. The green color coding shows how well the new formulations perform against priority based SGS heuristics.

This is expected since the formulation is designed to almost directly find an optimal solution for this. Analysis of the results indicate that the priority-duration exact formulation without a duration bias (α=1.0) produces a smaller priority-1 DWC by 0.4%±0.2% than that of the same formulation with a duration bias (α=1.1). This is intuitively expected because without the bias, small benefits can be achieved by being able to arrange activities in various orders to achieve a better centroid, instead of having an exponential bonus for the higher durations being scheduled earlier. The PSGS(SBP)

produces a priority-1 DWC that is 3.6%±0.8% greater (worse) than the priority-duration exact formulation with duration bias, while the average serial SGS (with no statistically significant benefit among either sorting method) performs 3.5%±1.1% worse than that of the PSGS(SBP), as summarized in table 8.

It is further noted that the priority-duration exact formulation without a duration

Pri-1 DWC Comparison

PD (α=1.0)	<	PD (α=1.1)	<	PSGS(SBP)	<	Avg SSGS
	0.4 ± 0.2%		3.6 ± 0.8%		3.5 ± 1.1%	
Worst Case for PSGS(SBP)				PD (α=1)	<	SBP-P
					26.1 duration-days (11%)	

Table 8: Statistical ranking of priority-1 DWC for the experiment set #1.

bias (α=1.0), produces the best priority-1 DCW centroid in all instances, followed closely by the same formulation with a duration bias (α=1.1). For the heuristic methods analyzed, in no case does one heuristic solving method perform consistently better than the others with this assessment criterion.

The exact solution methods of the multi-mode DT priority-duration RCPSP model produce the most solution-robust projects when considering priority and duration. It should however be noted that many other factors are involved in project scheduling. For instance, the underlying data may be highly stochastic in terms of activity durations as well as in terms of even having the correct activities planned. It should be noted that in the reviewed FMFCS NSWPP data, an approximate 30% growth in scope was observed. The interviewed industry members commonly said things like: “the moment a schedule is printed, it becomes obsolete” [3]. This then puts into question whether a single-digit percentile differences in makespan and pri-1 DCW are enough to warrant exact optimization solutions, especially if these take longer to complete the multiple scheduling iterations needed.

6.2 Numerical Experiment Set 2: Deviations and Deviation-Days DT Priority RCPSP formulations

Experiments were conducted to determine how effective the formulations from section 5.2.1 and 5.2.2 may be in re-scheduling activities in the 40-space fictitious ship (inspired by the HAL Class Frigate) used in previous experiments. The re-scheduling exact formulation used was the multi-mode re-scheduling DT priority RCPSP deviation days model (DT-R-DD), detailed in subsection 5.2.1, and the multi-mode re-scheduling DT priority RCPSP deviations model (DT-R-D), detailed in subsection 5.2.2, and this was compared to select serial and parallel SGS scheduling methods from the previous experiment.

It was found that a more effective way to use the serial SGS for re-scheduling was to re-order activities by the initial schedule start times, then to re-schedule using the serial SGS in that order. The re-scheduling serial SGS procedure functions along the following sequence:

1. Sort activities by previous schedule start times.
2. Remove completed activities and their precedence relationships.
3. “Lock-in” in-progress activities and remove their resource consumptions from the resource field at their current start dates.
4. Perform serial SGS scheduling for remaining unscheduled activities.

Experiments were conducted with the exact formulation and compared to the serial SGS methodology described above, to determine how effective this may be in re-scheduling a project during mid-execution. The experiment re-uses the first 10 of 30 previously used 100-activity initial scheduling problems from experiment set 1.

The naming convention for 3 re-scheduling methods used and analyzed is as follows:

- The re-scheduling serial SGS procedure described above: **SSGS-R**

- The multi-mode re-scheduling DT priority RCPSP formulation with deviation days (section 5.2.1): **DT-R-DD**
- The multi-mode re-scheduling DT priority RCPSP formulation with deviations (section 5.2.2): **DT-R-D**

The naming convention for 4 initial scheduling methods used and analyzed in this experiment is as follows:

- The best heuristic from experiment set 1, the parallel SGS, sorted 1st by priority, then 2nd by the shifting bottleneck procedure: **PSGS(SBP)**
- The heuristic most understood and liked by FMFCS schedulers from experiment set 1, the serial SGS, sorted 1st by priority, then 2nd by ES, then 3rd by duration: **SSGS(ES-D)**
- The multi-mode DT priority-duration formulation (section 5.1.1) with a duration bias (α=1.1), from experiment set 1: **PD(α=1.1)**
- The multi-mode DT priority-duration formulation (section 5.1.1) without a duration bias (α=1.0), also from experiment set 1: **PD(α=1.0)**

In the first part of the experiment, all 4 initial scheduling methods described above are used, followed by a re-scheduling step at day 22, using only the SSGS-R, the same scheduling method used in the initial schedule, and DT-R-DD for re-scheduling. DT-R-D was not used for re-scheduling in this part because the analysis is strictly focused on measuring the number of deviation days, and the DT-R-DD will give the optimal smallest number of deviation days. This optimal re-scheduling method is then compared with the SSGS-R, as well as with the same initial scheduling methods, to show the pitfalls of re-scheduling during a project with the same method used for initial scheduling. Prior to re-scheduling at day 22, the activities from the initial schedule are re-ordered by start date, and the first 5 in-progress activities on the list (i.e. started on or before day 22 but have not yet finished) are disrupted by having their durations extended by 3 days. All activities completed before day 22 are no longer part of the re-scheduling problem, and activities that have already started (in-progress) are pre-processed to ensure that they continue

until completion as they will not be re-scheduled. Only the activities that have not yet started, because their start day from the initial schedule was after day 22, are considered free for re-scheduling.

Deviation Days

Initial:	PSGS(SBP)			SSGS(ES-D)			PD (α=1.1)			PD (α=1.0)		
Re-schedule:	PSGS(SBP)	SSGS-R	DT-R-DD	PD (α=1.1)	SSGS-R	DT-R-DD	PD (α=1.1)	SSGS-R	DT-R-DD	PD (α=1.0)	SSGS-R	DT-R-DD
Trial 1	186	88	54	632	53	53	46	46	46	136	50	41
Trial 2	291	90	90	800	133	91	41	37	28	155	99	85
Trial 3	107	59	48	856	139	69	56	56	42	52	24	24
Trial 4	476	86	75	437	82	51	15	15	15	167	32	32
Trial 5	45	18	18	434	24	24	42	30	30	120	68	52
Trial 6	294	118	98	286	179	55	107	73	72	136	73	64
Trial 7	115	45	42	292	63	34	193	79	70	208	79	70
Trial 8	166	72	53	347	157	73	145	74	74	109	75	75
Trial 9	305	142	101	436	96	80	137	74	74	95	59	59
Trial 10	131	96	49	285	81	81	21	10	10	46	46	32
Average:	211.6	81.4	62.8	480.5	100.7	61.1	80.3	49.4	46.1	122.4	60.5	53.4

Table 9: Deviation days after first scheduling and then re-scheduling following disruptions at day 22. Color coding draws attention towards lowest deviation days (green) and towards highest deviation days (red) along the same initial scheduling trial number.

Table 9 shows the results of this experiment for all its 120 trials. It should be noted that although the re-scheduling problems are the same problem for the same trial and initial scheduling method, they are not the same problem when comparing between different initial schedules and the same trial (experiment) number. For illustration, the PD(α=1.0) and PD(α=1.1) are better at scheduling more activities earlier than the SSGS, as seen from experiment set 1, so the re-scheduling problem at day-22 is a simpler one. This explains why the deviation days are generally higher after re-scheduling when heuristic initial schedules are used, and it is a testament to how well the PD(α=1.0) and PD(α=1.1) "front-load" the activities.

Statistics obtained from this experiment set using standard t-tests at 90% C.I. show that formulations or heuristics that do not consider the original schedule (PD(α=1.1), PD(α=1.0), and PSGS(SBP)) perform poorly. Depending on the problem complexity at day-22, these scheduling formulations produce between 94% and 773% more deviation-days (or deviation time-units) than the SSGS-R. The DT-R-DD results in minimal deviations as

and this acts as base optimal values for statistical comparisons shown in table 10. The heuristic re-scheduler (SSGS-R) performs very close to optimal with the simpler problems (5%±17%) but performs worse as the problem complexity and size increases (up to 65%±46%).

Re-scheduling Results

Initial:	SSGS(ES-D)		
Re-schedule:	PD (α=1.1)	SSGS-R	DT-R-DD
Relative Results:	773%(±522) > 65%(±46) > 0%		

Initial:	PSGS(SBP)		
Re-schedule:	PSGS(SBP)	SSGS-R	DT-R-DD
Relative Results:	223%(±136) > 30%(±18) > 0%		

Initial:	PD (α=1.0)		
Re-schedule:	PD (α=1.0)	SSGS-R	DT-R-DD
Relative Results:	144%(±97) > 14%(±13) > 0%		

Initial:	PD (α=1.1)		
Re-schedule:	PD (α=1.1)	SSGS-R	DT-R-DD
Relative Results:	94%(±73) > 5%(±17) > 0%		

Table 10: Relative deviation days for the re-scheduling methods and initial scheduling methods.

An interesting observation is that the PD(α=1.1) performs significantly better than the PD(α=1.0), when using its developed initial schedule, then re-using the same method to re-schedule, with 29.4%±15.0% fewer deviation days (90% C.I.). This becomes intuitive after some thought because the duration bias (α=1.1) would more predictably order activities than with not placing the bias (α=1.0) (see subsection 5.1.1 for more detail). Although, outcomes on either side of this parameter change produce similarly ranked initial schedules in terms of priority-1 DWC (see table 7 in subsection 6.1), the duration bias (α=1.1) is more likely to re-position activities closer to the initial schedule following disruptions., when used for re-scheduling.

For the second part of this experiment set, the same randomly generated scheduling scenarios were used to determine how effective the priority-weighted deviations-only re-scheduling formulation (DT-R-D) might be in re-scheduling at different times throughout a project.

In the second part of the experiment, only 3 initial scheduling methods described above are used (PSGS(SBP), SSGS(ES-D), and PD(α=1.1)), followed by a re-scheduling step at day 22, using the SSGS-R, the same scheduling method used in the initial schedule, the PD(α=1.1), and the DT-R-D for re-scheduling. DT-R-DD was not used for re-scheduling in

this part because the analysis is strictly focused on measuring the number of deviations, and the DT-R-D will give the optimal smallest number. This optimal re-scheduling method is then compared with the SSGS-R, as well as with the same initial scheduling methods, to again show the pitfalls of re-scheduling during a project with the same method used for initial scheduling. Prior to re-scheduling at day 22, the activities from the initial schedule are again re-ordered by start date, and the first 5 in-progress activities on the list (i.e. started on or before day 22 but have not yet finished) are disrupted by having their durations extended by 3 days. All activities completed before day 22 are no longer part of the re-scheduling problem, and activities that have already started (in-progress) are pre-processed to ensure that they continue until completion as they will not be re-scheduled. Only the activities that have not yet started, because their start day from the initial schedule was after day 22, are considered available for re-scheduling.

	Deviations								
Initial:	PSGS(SBP)			SSGS(ES-D)			PD (α=1.1)		
Re-schedule:	PSGS(SBP)	SSGS-R	DT-R-D	PD (α=1.1)	SSGS-R	DT-R-D	PD (α=1.1)	SSGS-R	DT-R-D
Trial 1	24	18	18	49	25	25	16	16	16
Trial 2	36	30	30	46	40	29	18	16	4
Trial 3	23	20	18	46	24	19	17	17	11
Trial 4	38	26	25	30	19	17	5	5	5
Trial 5	6	6	6	34	8	8	12	10	10
Trial 6	32	35	31	30	23	23	26	25	24
Trial 7	15	19	14	30	24	11	26	27	17
Trial 8	20	21	9	40	32	16	36	35	35
Trial 9	43	39	29	46	42	36	29	29	29
Trial 10	19	22	15	34	27	27	4	4	3
Average:	25.6	23.6	19.5	38.5	26.4	21.1	18.9	18.4	15.4

Table 11: Deviations after first scheduling and then re-scheduling following a disruption at day 22. Color coding draws attention towards lowest deviation days (green) and towards highest deviation days (red) along the same initial scheduling trial number.

Statistics obtained from this experiment set (results in table 11) using standard t-tests at 90% C.I. show again that formulations or heuristics that do not consider the original schedule (PD(α=1.1) and PSGS(SBP)) perform the worst among tested methods. Depending on the problem complexity at day-22, these scheduling formulations produce between 4% and 70% more deviations than the SSGS-R. The DT-R-DD results in minimal

deviations as and this acts as base optimal values for statistical comparisons shown in table 12. The heuristic re-scheduler (SSGS-R) performs worse than the optimal DT-R-D with between 28%(±24%) and 45%(±54%) more deviations on average, although this is not always significant due to the high variability in results. As expected, in no case does a heuristic method perform better than the DT-R-D.

Re-scheduling Results

Initial:	PSGS(SBP)		
Re-schedule:	PSGS(SBP)	SSGS-R	DT-R-D
Relative Results:	8%(±12) > 28%(±24) > 0%		

Initial:	SSGS(ES-D)		
Re-schedule:	PD (α=1.1)	SSGS-R	DT-R-D
Relative Results:	70%(±55) > 31%(±25) > 0%		

Initial:	PD (α=1.1)		
Re-schedule:	PD (α=1.1)	SSGS-R	DT-R-D
Relative Results:	4%(±4) > 45%(±54) > 0%		

Table 12: Relative deviations for the re-scheduling methods and initial scheduling methods.

A third part to this experiment, inspired by the previous two parts, is then conducted to show the general effect of using the re-scheduling methods at different points in a project. Figure 28 shows the total number of deviations resulting from re-scheduling five scenarios each where the initial project was scheduled using the SSGS, then re-scheduled at days 30, 22, 15, and 7 where the number and complexity of the remaining project activities grows larger as the project is closer to its infancy. This can be seen graphically in the figure, as the sooner a disruption occurs in a project, the more impact it will have to the rest of the project, requiring more overall deviations and deviation-days without considering alternative execution modes that reduce the duration of either the current in-progress activities or the duration of future activities.

It is seen that the DT-R-D formulation performs better at reducing the overall number of deviations. In a practical sense, this translates into a scheduler or PL doing less leg work to re-schedule activities.

There is however a frequent drawback from using this formulation. The largest problem, when considering the real-world implications, is that it does not matter how much later an activity is scheduled from its original date, and although a project manager might not mind if an activity is re-scheduled two days later or four days later, a project

manager will most likely mind if it is very much later, such as 40 days later or outside the agreed project timeframe.

When reviewing the solution schedules from the experiment set, it was observed

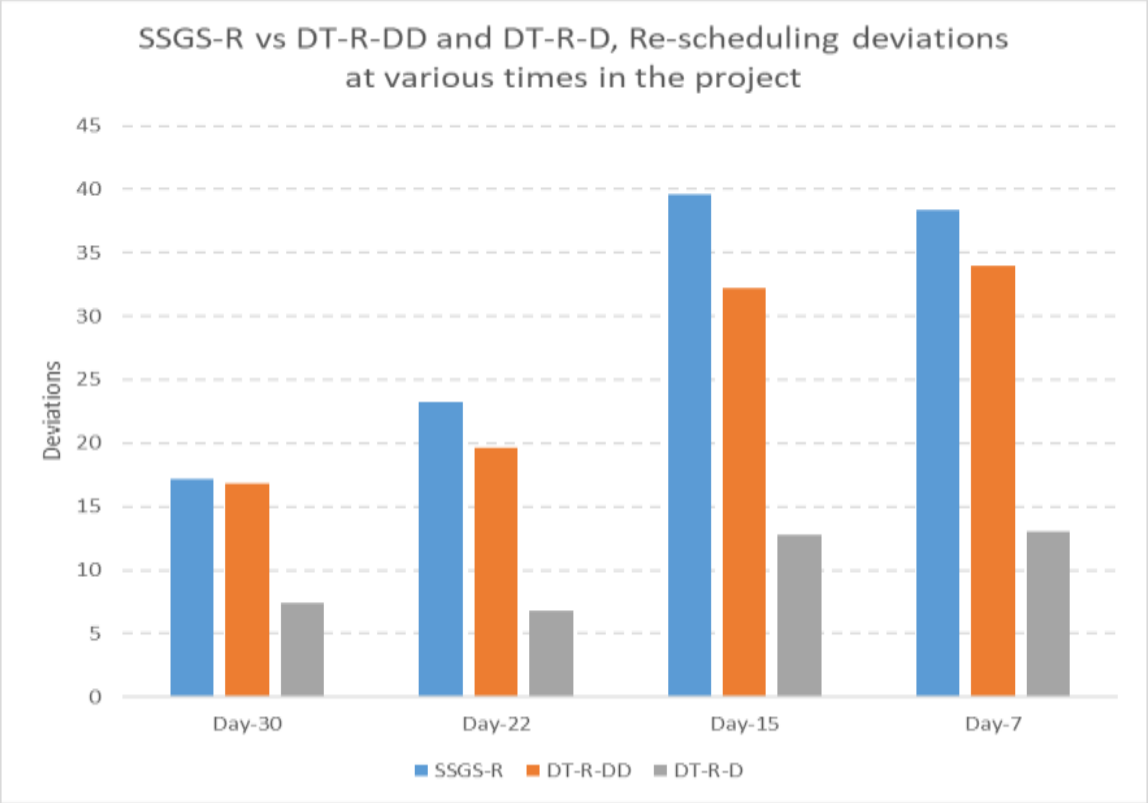


Figure 28: Absolute deviations using re-scheduling methodologies.

that many of the changes are priority-1 activities being moved to the end of the other project's activities as soon as resources were free. This has the effect of completely removing the previously mentioned priority-1 buffers that naturally occur from using the SSGS(ES-D) heuristic or the PD(α=1.0) and PD(α=1.1) initial scheduling formulations.

A secondary negative effect is that this may also increase the project makespan, as seen in table 13. Analysis of the results show that the SSGS-R method and the DT-R-DD method only slightly increased the project's makespan by an average of 0.6 days, 1.6 days,

1.8 days, and 2.4 days when rescheduling at day-30, day-22, day-15, and day-7 respectively; however, the DT-R-D formulation produced makespans that were increased by and average of 5.0, 4.0, 5.0, and 4.8 days for the scenarios where the project was re-scheduled at day-30, day-22, day-15, and day-7 respectively. It should be noted that in real-world occasions, this new schedule may or may not be acceptable because of this makespan increase.

Average Re-Scheduled Makespan (days)			
Disruption Point	Re-schedule		
	DT-R-DD	SSGS-R	DT-R-D
Day-30	125.2	125.2	129.6
Day-22	126.2	126.2	128.2
Day-15	126.4	126.4	129.6
Day-7	127	127	129.4
Initial Average Makespan 124.6 days			

Table 13: Re-scheduling after delays (disruptions) using any method potentially increases a project's makespan, but more-so using the DT-R-D.

The re-scheduling situation is difficult in NSWPPs because it is not trivial to re-work plans, and important key stakeholders may want to know that the PL selected the best option. The cost-time-scope trade-off decision is often not the decision of the person using the scheduling tool, so different options should be presented to decision makers. In some cases, having only a few priority-1 activities left at the end of a project might not be so bad because they will have closer managerial attention, especially if the technical risk of those activities is lower and if the makespan is acceptable. As was seen in numerical experiment set 2, even the optimal re-scheduling formulations may still produce too many deviation-days and deviations, and therefore using the multi-mode features of the re-scheduling formulations may be useful in guiding management's investigations. Numerical experiment set 3 explores the multi-mode feature of the DT-R-DD formulation.

6.3 Numerical Experiment Set 3: Re-scheduling with the multi-mode feature - limited job targeting

The purpose of the experiment set is to show how many iterations are needed to reduce deviation-days to a reasonable value when this formulation is used iteratively. This is conceptually how this formulation is expected to be used, where alternative execution modes and overtime is carefully determined on a case-by-case basis. Secondarily, its

purpose is to gauge how much benefit is seen at each iteration, and to gauge solve-time with problem size, and to set expected limits on the number of activities to include in the model.

In this experiment set, the fictitious 40-space ship encountered a 3-day delay to the first five in-progress activities at day-22. Note that a delay will normally represent new urgent tasks found during an inspection that must be completed before other work can finish in that space. It is normal practice to leave non-urgent arising work until the next NSWPP [3]. In every case, the schedule was first solved using the SSGS(ES-D). The multi-mode re-scheduling DT priority RCPSP deviation-days (DT-R-DD) formulation described in subsection 5.2.1 was then used in a 1st round to re-calculate a new schedule after the disruption, without using any duration reductions. In the second round of the same experiment set, the parameter β_{M2} is set to one, since this models an iterative process where the user wishes to find the single most impactful activity to reduce, thus leading to less re-scheduling, but adhering to the reality that overtime or alternate execution modes are normally determined for one WO at a time. Setting β_{M2} equal to one will normally show a single activity reduced by three days because the disruptions are three days in duration. For the subsequent iterations in the same trial, the selected activity in the previous trial is duration reduced as prescribed by the previous experiment round's solution. The reduced activity's alternative modes are fixed at that reduced duration so that further reductions for the same activity in duration are prevented from being a viable solution.

It is practical to set β_{M2} to a small number because it simulates more closely how a scheduler may be expected to use this formulation in a real-world context where overtime or additional shifts are difficult to plan and approve. In this context, only a small amount of overtime should be analyzed, being the most beneficial overtime/additional shifts. The idea is that the scheduler and PL do not know if overtime is feasible for any work and determining if overtime is feasible for all work at all times is impractical. By setting β_{M2} to a small number and solving a schedule that produces the smallest number of deviation-days, the user can find the most impactful activities to investigate for

overtime feasibility. This process might be iterative. A decision algorithm is shown in figure 29 to illustrate how using this formulation as part of an optimizer tool can be used for this iterative process. For example, after the DT-R-DD formulation is first used, the

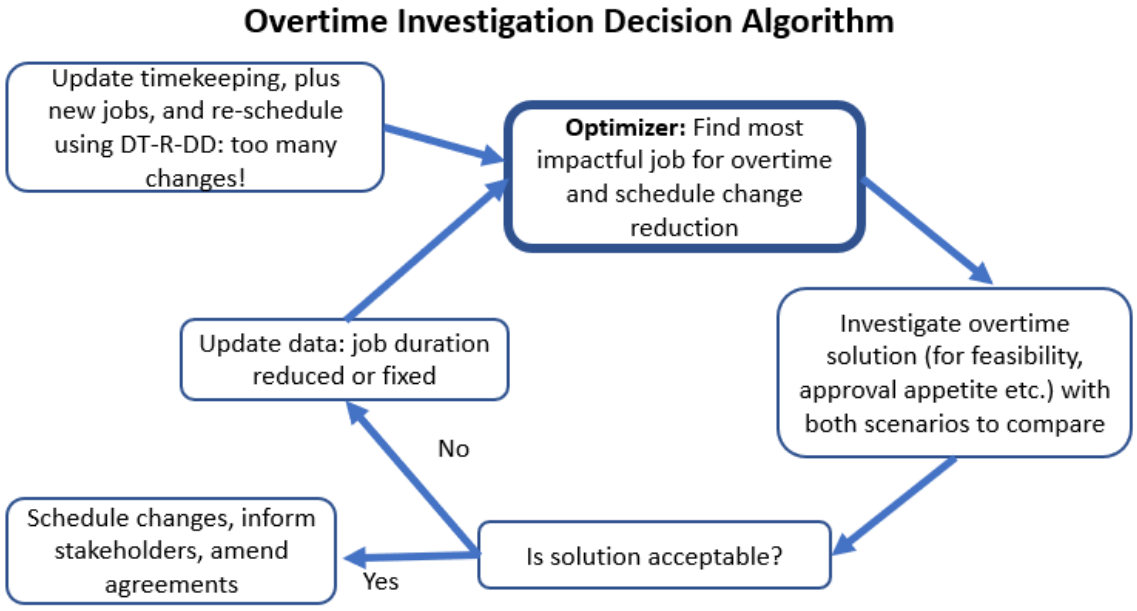


Figure 29: A decision algorithm shows where the DT-R-DD formulation is used with a single duration-reduced activity.

analyzed activity, determined by activity *j* in the solution where x_{jtm} is equal to 1 and *m* is not mode 1 (i.e. 2, 3, or 4), might not be able to be shortened after conducting his/her overtime investigation. The user can then adjust the duration table to indicate that alternative modes *m* for this activity do not produce a reduced duration and re-use the formulation. This will find the next most beneficial activity for reduction that will cause the smallest number of deviation-days.

As expected, solution time increases exponentially with the number of activities in the initial scheduling problem. The solution time also has a high variability, and figure 30 displays the results using a box and whisker chart to show the spread and distribution of solution times.

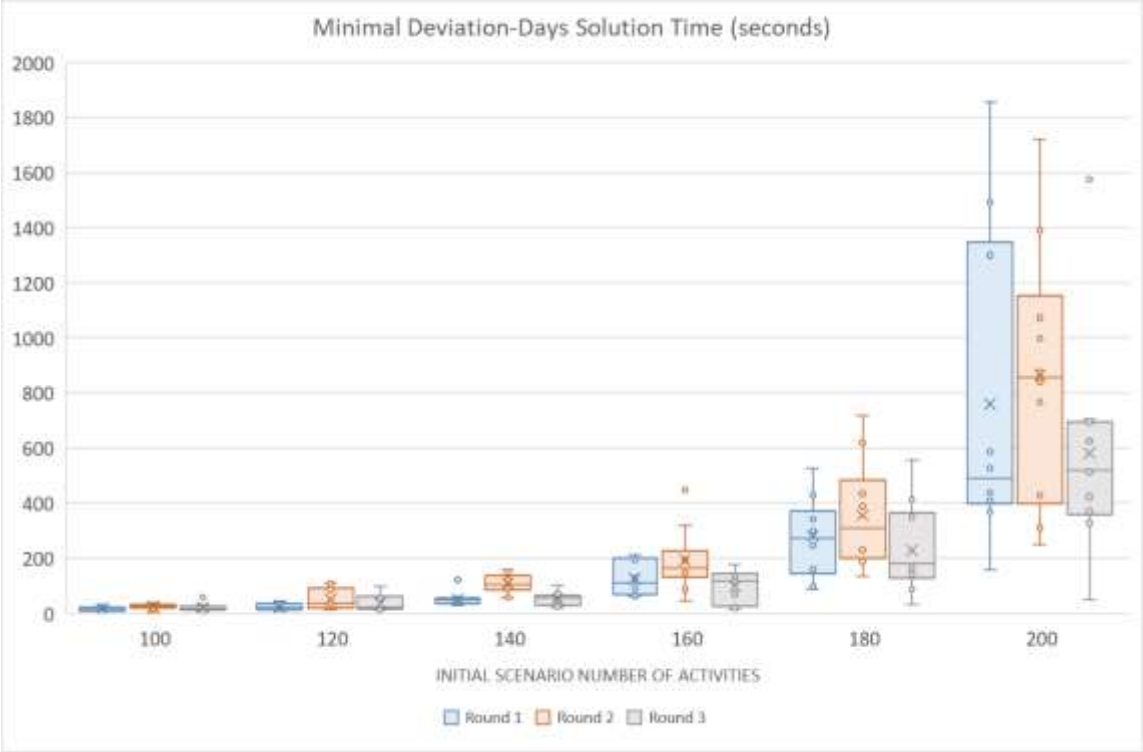


Figure 30: Solution time of the the DT-R-DD formulation as the number of activities increases.

To review the source experiment data, please refer to Appendix B. A less intuitive observation is that the solution time increases from the initial re-scheduling scenario, where β_{M2} is equal to zero, to the first iteration with reduced duration modes where β_{M2} equals one. As reduced duration modes are accepted and fixed, the problem complexity diminishes in subsequent rounds and solution times improve. For the second iteration (the first round with β_{M2}=1), an average relative increase in computation time of 71.9% ± 17.1% (90% C.I.) is seen. In the next iteration after the first activity is duration-reduced, the solution time then decreases relatively by 35.1% ± 16.5% (90% C.I.). When a second activity is duration reduced, the next iteration experiences an average decreased solution time of another relative 46.2% ± 13.3% (90% C.I.). Finally, with three activities duration-reduced, another relative average decrease in computation time of 52.0% ± 14.7% (90% C.I.) (not shown in figure 39) is observed. Analysis was not performed for 5 total activities as it is intuitively obvious that by reducing the five initially disrupted activities, the exact

same schedule can be achieved with no deviations and optimization is meaningless in a case where all five of these activities could be returned to their original durations.

When a single reduced duration can completely correct the schedule (at any round), then the solution time is drastically smaller as the solver reaches a lower bound and need not investigate further. The average solution time for the final round that completely corrects a schedule was 4.9, 8.8, 14.0, 19.0, 25.0, and 35.6 seconds for the experimentation sets with 100, 120, 140, 160, 180, and 200 initial schedule activities, respectively.

Of the 60 randomly generated schedules used in this experiment set, only one schedule (1.7%) could be completely corrected with a single reduced activity, six schedules (10.0%) could be completely corrected with two reduced activities, 14 schedules (23.3%) were completely corrected with three reduced activities, 26 schedules (43.3%) were completely corrected with four reduced activities, and 10 schedules (16.7%) were completely corrected with five reduced activities. In three schedules used (5%), a complete correction could not be achieved with five reduced-activities as a later activity (uncommonly, not one of the original five disrupted activities) was duration-reduced in an earlier round. Note that this iterative approach (with one reduced activity at a time) cannot be guaranteed to give as optimal of a solution as if multiple activities are allowed to be reduced, but the practical approach of this formulation is not to completely correct a schedule as this is practically impossible in reality (i.e. the probability that all five activities will actually be reduced by as much as desired is impractically low and not worth investigating as a whole).

As expected, in terms of improving the re-scheduled schedule quality with reducing activity-durations, the largest improvements are observed in the first round, followed by the second round, then the third rounds etc. The relative reduction of deviation days does increase at each round however, by a relative average of 56.5% ± 9.3%, 57.3% ± 11.5%, 67.1% ± 13.2%, and 86.7% ± 8.0% (90% C.I. for all), for rounds two,

three, four, and five respectively. Figures 31 and 32 illustrate the number of deviation

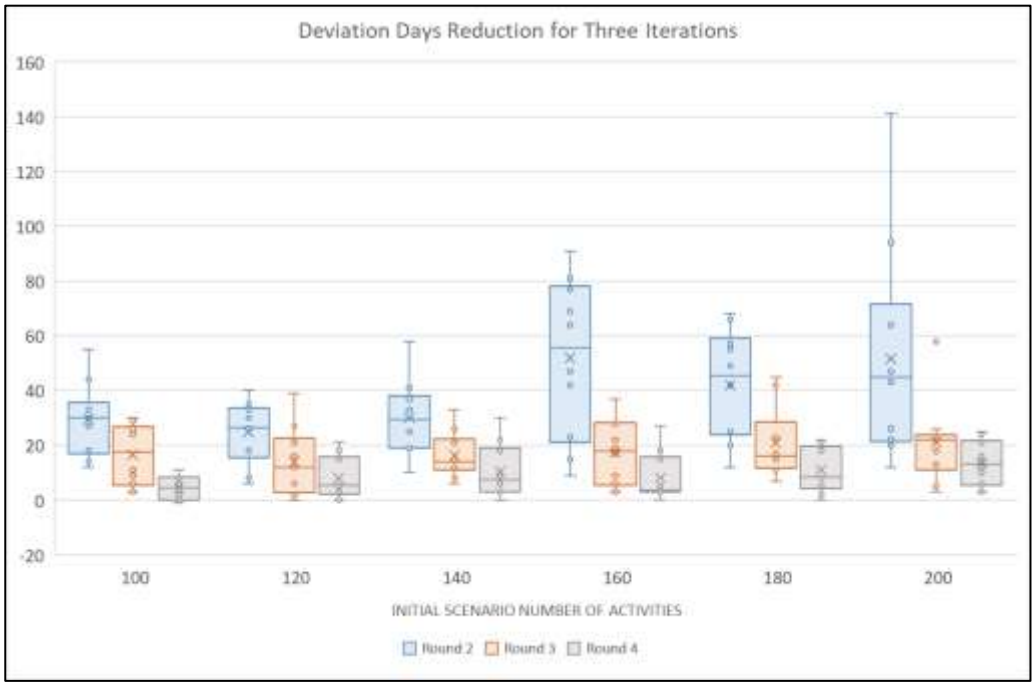


Figure 31: Deviation-days reduction using the re-scheduling formulation. The most absolute benefit is seen after the first round, then it decreases with successive rounds.

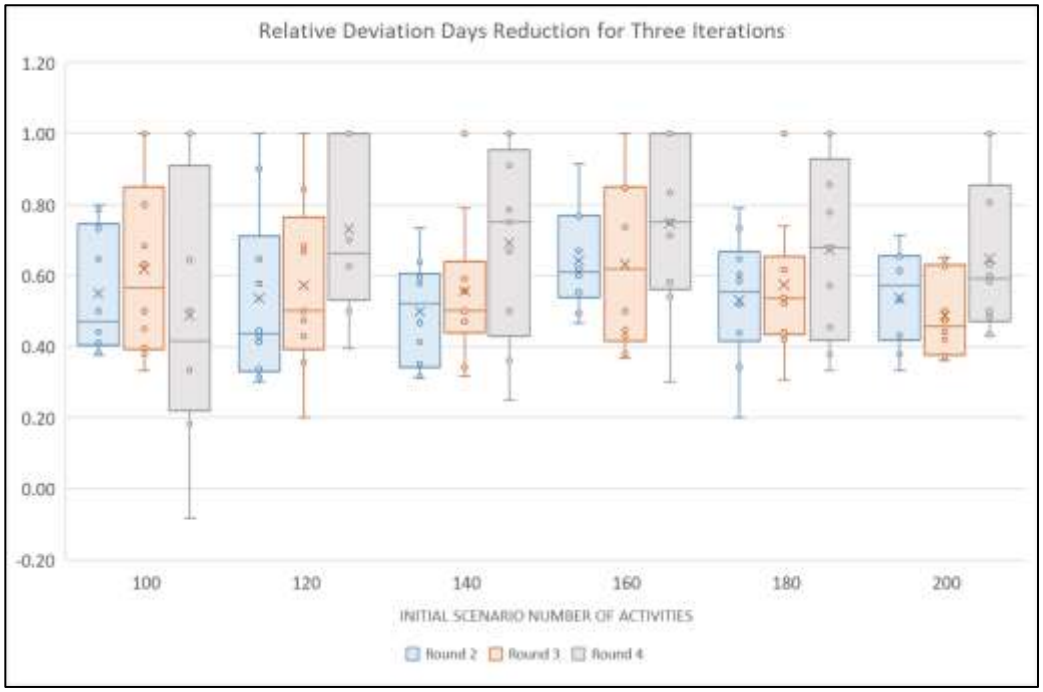


Figure 32: Deviation-days reduction using the re-scheduling formulation. The relative benefit in reduction (% of remaining deviation days) tends to increase with successive rounds.

days and relative deviation days reduced in the first three iterative rounds that considered a single reduced activity.

The relative reduction of deviations (not deviation days) is also shown to increase at each round and summarized is in table 14. Figures 33 and 34 show the relative reduction at the first three iterative rounds that reduce an activity.

Average relative deviation reduction 90% C.I.	
Round 1	Initial deviations baseline
Round 2	49.6% ± 25.2%
Round 3	56.9% ± 22.3%
Round 4	64.2% ± 19.8%
Round 5	83.3% ± 13.1%

Table 14: Relative reduction in deviations by reducing the single-most troublesome activity to regain a schedule at each round.

Note that the formulation weights deviation-days by priority levels and the experimentation analysis does not consider the underlying priority of an activity;

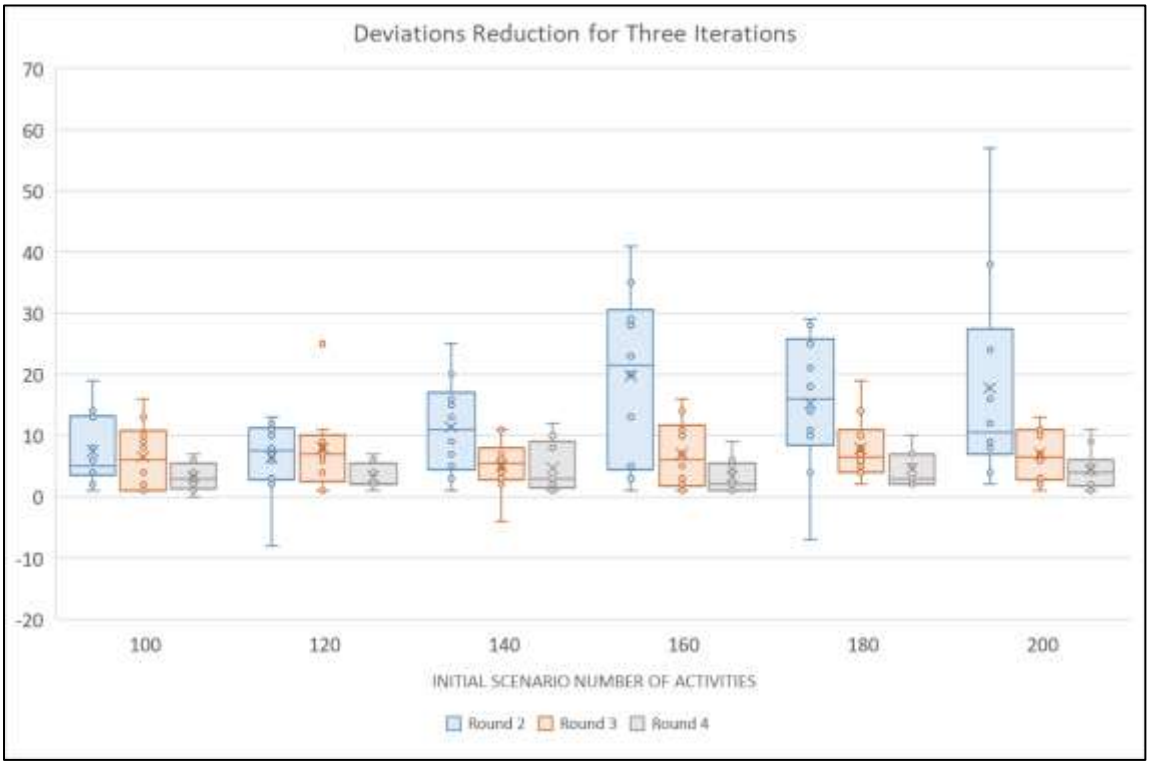


Figure 33: Deviations reduction using the re-scheduling formulation. The most absolute benefit is seen after the first round with reductions, then it decreases with successive rounds.

therefore, in one case of 284, there was actually a one deviation-day increase in an iteration, while total deviations increased three times in the 284 experiment rounds.

Also note that in this specific experiment, a single iteration representing a single WO reduced with overtime/alternative execution/night shift, may decrease total deviation-days and deviations by approximately 50%; however, the magnitude of a

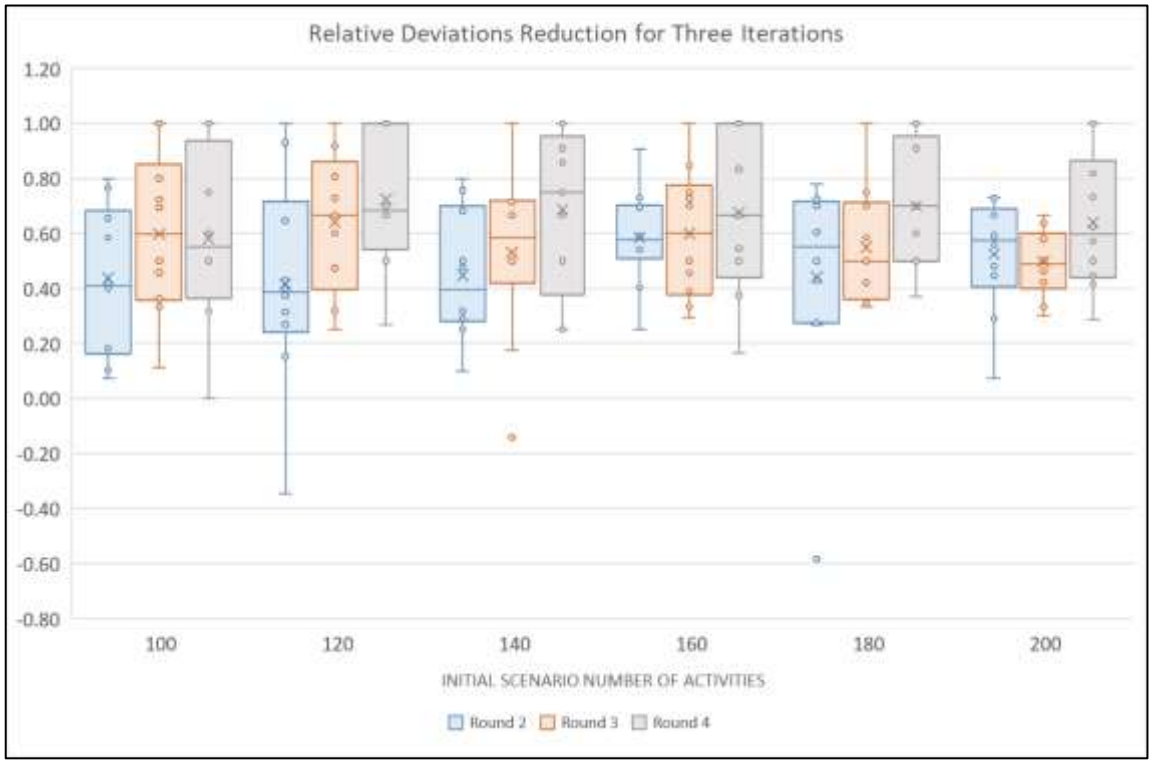


Figure 34: Relative deviations reduction using the re-scheduling formulation. The relative benefit increases with successive rounds.

schedule improvement will be reliant on the number and nature of schedule disruptions that occurred before re-scheduling, but this formulation can at least be trusted to find the most impactful activity.

To truly gauge the value and practical benefits of this formulation, it is proposed that this be enabled as a user-friendly option (as much as possible) in a refit optimizer and user-feedback is collected to expand on default settings, options, how to capture an initial schedule for minimizing deviation-days, and to find practical limits on how many activities should be used in the formulation. For example, a practical default setting might be that only the nearest 150 activities are optimized for reduced deviation-days, even though the increased project stability will have positive impacts beyond those 150 activities.

6.4 Numerical Experiment Set 4: Discrete-Time Priority-Duration RCPSP Model for WO Scheduling with Activity Adjacency

The primary purposes of this formulation were to enforce adjacency among activities/activities in the same WO/work packages, and to improve solution time from the DT priority-duration formulation. In this experiment set, small sets of problems were solved with the DT priority-duration formulation and then again with the DT priority-duration-adjacency formulation. These were compared for solution time, makespan, and priority-1 DWC. The random problem generator used in this experiment phase was

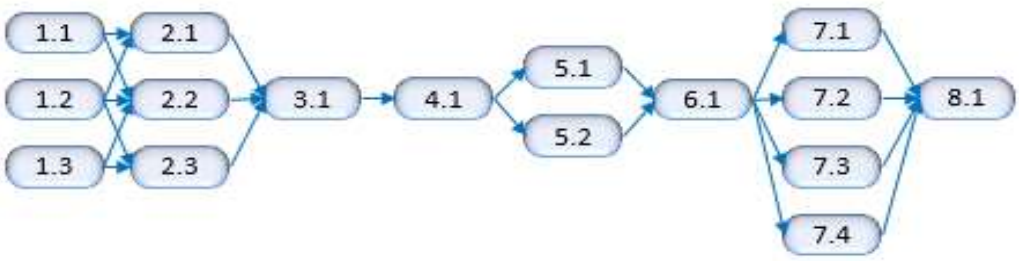


Figure 35: WOs were built with subsequent sets of parallel activities, to mimic real-world NSWPP scheduling structure.

developed to have WOs with 10 to 50 activities each, in sequential sets of one to five parallel activities as a data structure. Figure 35 is a depiction of this data structure for one single work package.

These new problem sets where generated in the Primavera P6 output format provided by the research partner Thales Canada, with many of the nuances that come with the data. For instance, a single compartment resource demand was generated for each WO's activities, but it is not possible to assigned multiple values for this resource (i.e. capacity). These were assigned as "resources". One can however populated a field called "max workers" in "resources". For shop resources, these are added as "roles" assigned to each activity, and the capacity of each role can be user-modified, so can the number of "roles" used by an activity. Data pre-processing is then needed to sum the "roles number" for each role used in an activity, then use this sum as the numerical resource demand of "max workers" from the compartment resource for the same activity.

Since the random problem generator generated entire work packages, the problem sets were parsed into bins with a range of activity numbers, rather than a fixed number of activities. For example, problem set 1 has between 100 to 110 activities, problem set 2 has between 140 to 150 activities, etc.

The naming convention for the methods analyzed in this experiment set is as follows:

- The multi-mode DT priority-duration formulation (section 5.1.1) with a duration bias (α=1.1): **PD(α=1.1)**
- The multi-mode DT priority-duration formulation (section 5.1.3) with the adjacency constraint. Note that α=1.1 in this experiment as well: **PD(adj)**

It was observed that the serial SGS solution that does not consider same-WO-adjacency constraints will occasionally produce a better makespan than the optimal solution with adjacency, leading to an unfeasible problem for the solver if the LS is calculated from the time horizon H produced by the serial SGS. To work around this problem, the serial SGS was modified to check that all activities within a work package can "fit"; if not, the work package and start times of its activities were shifted by one day, then the check repeated etc. Once the work package fits in the resource table, the work package and all its activities are scheduled, the resource table is updated, new ES values for the next work package are updated, and then the next work package is checked for fit starting at the ES etc. This modification produced a time horizon H to be used for LS calculations, that always presented feasible solutions. The results of the experiment are tabulated in table 15.

Using a standard t-test, the .lp file creation time is reduced by 65.8%±0.85% (90% C.I.) by introducing the adjacency constraints. The same analysis cannot however be done for solver solution times as the differences change dynamically with increased number of activities. Figure 36 is a depiction of average solution times amongst the four bins, showing that with simpler problems, adjacency constraints lead to longer (yet

operationally acceptable) solution times; however, once the activity size reaches 180-190 activities and larger, the solution time with adjacency does not grow as quickly.

				PD(α=1.1)	PD(adj)	PD(α=1.1)	PD(adj)	PD(α=1.1)	PD(adj)	PD(α=1.1)	PD(adj)
Bins Set	Trial Number	Number of WOs	Number of Activities	Solution Time (s) lp file creation		Solution Time (s) Solver (Gurobi)		Makespan (days)		Avg DWC (days^2)	
100-110	1	5	103	10.9	3.8	1.2	3.4	75	73	104.16	113.05
100-110	2	7	107	10.7	3.7	2.3	0.4	63	78	80.71	109.15
100-110	3	6	104	9.7	3.4	0.5	2.8	61	65	73.66	100.60
100-110	4	6	106	10.0	3.6	0.4	2.8	74	74	82.64	90.80
100-110	5	6	110	10.5	3.8	0.8	5.4	51	61	71.03	95.50
140-150	6	8	145	15.2	5.1	2.8	5.5	72	82	152.60	187.06
140-150	7	10	145	16.1	4.9	3.2	0.5	69	63	141.38	167.86
140-150	8	9	147	14.2	5.0	0.6	9.6	62	63	133.69	155.93
140-150	9	11	150	15.3	5.5	10.2	16.9	58	69	116.49	146.47
140-150	10	9	144	17.4	5.2	2.3	9.5	87	88	141.25	151.71
180-190	11	13	189	24.2	9.6	49.9	74.0	84	85	145.14	191.33
180-190	12	14	184	21.7	7.8	154.7	74.1	71	77	118.95	165.99
180-190	13	11	182	20.0	6.8	143.9	24.3	74	89	134.26	157.77
180-190	14	14	185	20.4	7.5	109.7	110.3	59	70	129.98	178.74
180-190	15	11	186	22.5	7.5	61.9	27.4	72	101	159.36	233.40
220-230	16	12	225	24.6	9.1	1340.0	143.4	70	99	154.46	216.78
220-230	17	19	221	28.4	10.5	30000.0	737.0	88	92	176.25	185.42
220-230	18	15	222	25.8	9.1	4466.0	316.1	63	92	138.73	194.19
220-230	19	16	228	29.4	8.8	30000.0	99.2	79	90	127.00	179.95
220-230	20	15	227	27.4	11.2	611.9	909.8	65	80	142.49	213.39

Table 15: Solution times (seconds) per number-of-activity bin with and without adjacency constraint.

When considering the combination of lp file creation and solution times, the overall solve time for adjacency constraints was lower in 18 out of 20 cases.

This suggests that scheduling with same-order adjacency constraints may be an acceptable way to perform aggregate planning for larger problems since this reduces overall solution time. With adjacency constraints however, note that this represents a different problem, and the optimal solutions in terms of makespan and DW centroid is often poorer. In the experiment set, adding same-work package adjacency constraints increased the makespan on average by 7.4%±4.1% with a 90% C.I., while the average pri-1 DWC was increased on average by 22.4%±3.9% with a 90% C.I.

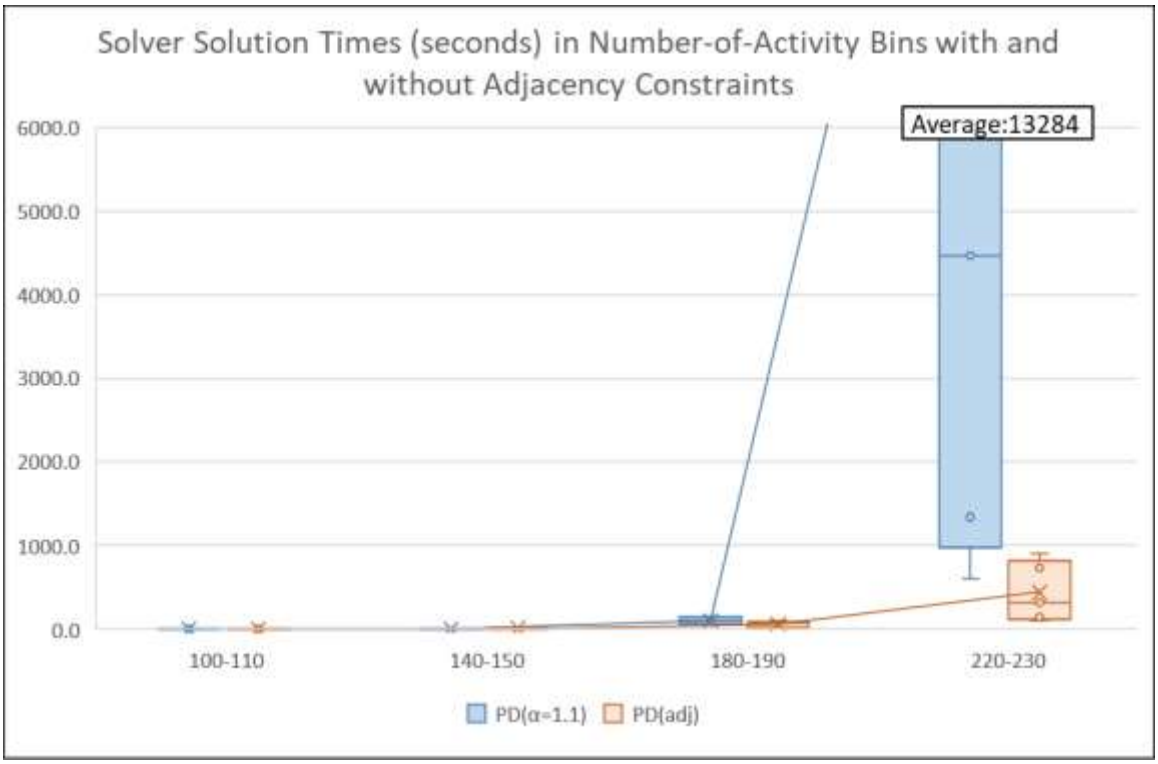


Figure 36: Solver solution times do not increase as quickly with adjacency constraints.

Note also that the number of activities may not translate directly to problem difficulty. As was found with experimentation, if the resource capacities were larger, this solver could easily schedule many more activities. The problem difficulty arises from having an increasing number of bottleneck resources. As was found from working at FMFCS, a few critical shops with low worker capacity can completely cripple a PL's ability to schedule large work packages that required dozens of healthily capacitated shops, resulting in poor overall utilization for those shops in the work period analyzed. This is indirectly an insight into the importance of having sufficient resource capacity (i.e. shop workers); however, compartment space capacity is not so easily grown, and it is theorized that this represents the most important bottleneck when sufficient worker resources are made available. As a single trial using the same random problem generator that produces Primavera P6 files, the worker capacity was increased from 5 units per shop to 20 units per shop, and no changes to space capacities were made. The generator produced a 506-activity problem with 36 work packages. With the adjacency formulation, Gusek produced an lp file in 15.6 seconds, and Gurobi 9.0 solved the problem in 4.31 seconds, which is

much faster than the previous average 220-230 activity instances used in the experiment set, where the average time was 2079 seconds.

The results of these experiments, showing improvements in optimized scheduling methodologies vs priority-rule SGS methods, motivated a qualitative experiment set in this thesis: a series of interviews alongside prototype development at FMFCS using only a serial SGS methodology, for a real NSWPP project. It is proposed that for an organization to embrace optimization, the obstacles to automatic scheduling themselves should first be understood, so that they may be overcome.

6.5 Qualitative Experiment Set 5: Development and Testing of a Heuristic-based Scheduler at FMFCS

Although user-feedback information is more qualitative than quantitative, it is proposed that this be used when considering NSWPP scheduling software development, as the users are the ones who will ultimately decide if a such a program is useful. With information provided by FMFCS as a result of interviews and meetings [3], a prototype scheduler was built in Excel and VBA using a serial SGS. Features were added to import and translate weekly shop capacity information into half-day capacities. This program considered holidays and special occasions that would prevent work, it allowed the user to manually adjust concurrence assumptions with previous activities, and to override activity start times. Note that in this section, to conform with SAP nomenclature present in real-data figures, the term “activity” is replaced with “operation”, as an “activity” in SAP represents a group of parallel “operations”, and a “suboperation” number represents the unique identifier of an operation within the same activity group. For example, every line in figure 46 represents an “activity” in the rest of this thesis, but every line in figure 46 represents an “operation” in this section.

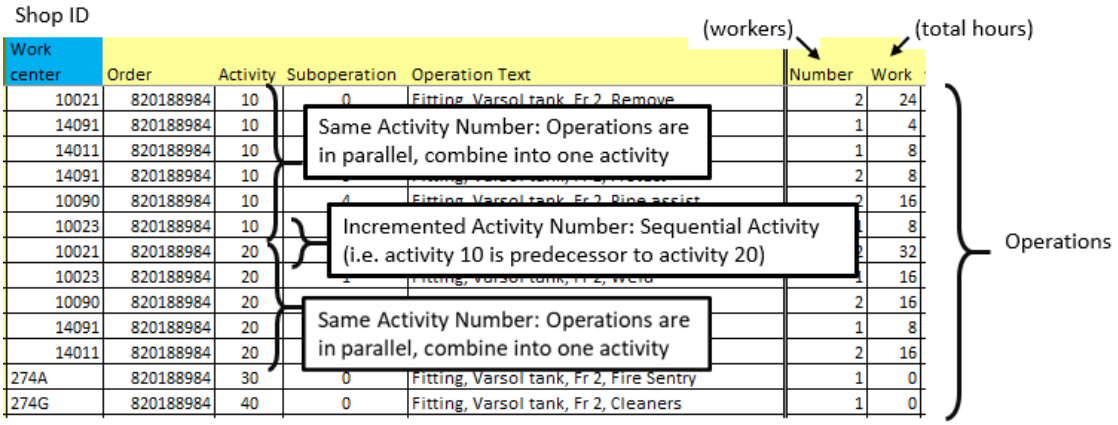
The program took up to 6 parallel operations (the most observed in practice), and combined them into a single activity for scheduling, which used the duration of the longest parallel operation. This was done to accommodate the reality of planning precision. Although in most plans, several operations were scheduled to occur in parallel,

it was rare to see these parallel operations have the same number of hours assigned. For example, an installation activity might consist of contributions from four shops: electrical, mechanical, rigging, and engineering. Rather than going through the tedium of scheduling each operation as multiple activities occurring at 0.5 to 1 hour intervals in a complicated sequence, prior knowledge was used to simply set all activities to occur in parallel, with varying hourly contributions from each shop that are derived from shop discussions or historical information on similar tasks. The precision of when shops can arrive for work in a ship is approximately a half-day to a day, so each shop will arrive at approximately the same time (i.e. the start of a work day), then work together throughout the morning; as shops have completed their contribution, they will return to FMFCS, then move on to the next operation. If the operation takes several days and a shop's contribution is actually several small steps over a period, then shops will work on concurrent operations from other projects until they receive indication that their next contribution is ready for action. This causes unintended delays, and all may occur within the span of a single operation's start and end date. Because of this, a practical adjustment to the prototype was to distribute a shop's hourly contributions to a parallel set of operations to persist for the duration of the longest parallel operation in the set. The prototype scheduler also assumes that shop availability after the shop capacity report forecast, up to 200 workdays in the future, was equal to the completely unloaded capacities for each shop.

It was found with the operations list that several inconsistencies were present in the data. From the text descriptions, it was found that the sequencing of operations was not always correct, as for example, some work that was clearly intended to occur after the previous work was often listed as a parallel operation. No precedence information existed between WOs from SAP data extracts, but it was clear that some precedence was desired and programmed in Prometheus Scheduler for activities within a same WO. Due to this, a default setting was implemented that assumed any numerically increasing number in the "activity" field of an operation within a WO were sequential, but a feature was added to make an operation concurrent with the previous one by the click of a button, so that a user could modify the planning data. The DRMIS data structure is

formatted with unique characteristics. Figure 37 is a sample of this operation structure, showing sequential and parallel relationships.

For data related to compartment information, an adjacent compartment field was added for the user to select an adjacent space affect by hot work. No hard coding for this is in the data except for the "Work Type" field that may list "Hot Work". The adjacent compartment field could also be used to make operations exclusive by having two operations "consume" a compartment resource that is surely not part of a work period, such as the rudder, the tip of the bow, a void space near a steering compartment etc. This



Work center	Order	Activity	Suboperation	Operation Text	Number	Work
10021	820188984	10	0	Fitting, Varsol tank, Fr 2, Remove	2	24
14091	820188984	10			1	4
14011	820188984	10			1	8
14091	820188984	10			2	8
10090	820188984	10			2	16
10023	820188984	10			1	8
10021	820188984	20			2	32
10023	820188984	20			1	16
10090	820188984	20			2	16
14091	820188984	20			1	8
14011	820188984	20			2	16
274A	820188984	30	0	Fitting, Varsol tank, Fr 2, Fire Sentry	1	0
274G	820188984	40	0	Fitting, Varsol tank, Fr 2, Cleaners	1	0

Figure 37: The SAP-DRMIS operation data structure used for scheduling

is how exclusivity between radiate, emissions, and aloft work types could be considered in the prototype. "Radiate" and "emissions" were also added as resources for this purpose.

If no additional input for space information was provided, then the default value would be that any scheduled operation, including combined operations described earlier as a concurrent activity, consumes an entire space, making no two sets of combined operations from separate WOs able to be scheduled at the same time in the same compartment. For certain, spaces such as the engine rooms or the hangar, multiple operations from separate WOs may occur in parallel. If this is the case, the user could set the WO capacity per compartment. For instance, the PL using the prototype desired that the engine rooms had a default capacity of four WOs. In a few cases, a WO that was very invasive, required scaffolding and many interference removals, could be set to consume

three or four of the engine space resource capacity units. These options were developed after a few options were used with different options being less successful. The user preferred any option that required less set-up, less work, and was more efficient. After a few iterations with user feedback, default values were set for the normal tendencies and only exceptions required manual intervention.

Since the prototype scheduler used a serial SGS heuristic methodology, the solve time was very fast. It was found that problems with the planning data were usually only evident once scheduled start and end dates were given and reviewed. These problems may have been incorrect sequential information between operations, concurrence between operations, or having the wrong compartment selected for the operations, etc. The user could more easily work with this imperfect data by changing a parameter, then re-scheduling, one issue at a time and checking the new schedule, until no issues were evident in the planning data. For this to be practical, the solve time between schedules after partially "cleaning" the data must be reasonably fast. For the 400 operations, the serial SGS solved the problem in 1.3 seconds on average between data "cleaning" iterations.

The PL insisted that he wanted the ability to sort all WOs (and underlying operations) in a top-to-bottom priority list, above and beyond the three official priority levels. Even within a priority level, some WOs were more pressing and important to start early than others.

After describing optimization, parallel SGS, and serial SGS scheduling schemes, the interviewed users preferred the serial SGS methodology because it more closely mimicked their scheduling tendencies, where the user schedules from a priority-ordered list placing WOs as soon as are feasible in a makespan, until the entire list was scheduled. They were more interested in automatic scheduling to save time rather than to have a more efficient schedule.

Upon a post-work period interview, the PL using the prototype indicated that although the tool was useful to plan an initial large list of WOs, he did not use it when

requirements emerged to re-schedule only a few WOs. He stated that whatever was quicker was better, and that he could manually navigate DRMIS to check shop capacities at various weeks and manually schedule these WOs, faster than he could use the serial SGS prototype scheduler. The biggest "pain" was felt when downloading the shop capacity report, which took a few minutes to download and translate into a resource table. In this case, even a delay of a few minutes was enough for him to abandon automatic scheduling during project execution.

Another important observation was that although the planners had not indicated such in the planning data, schedulers, PLs, and PMs by default always wanted all parallel operation sets to occur within a WO, one after the other until they were all completed, unless this was practically impossible. There is a real-world reason for this in NSSs, where space is limited, and is also being occupied by ship staff (SS). If tools and scaffolding are left in a compartment, or steel is left exposed (contributing to rapid corrosion), then there is a risk of tools being lost or misplaced, it prevents other work from beginning, and it causes managers at the FMF repair facility to be constantly asked when the work is going to be finished by SS. Unless it was infeasible to do so, management always preferred to leave no significant time gaps between sequential operations. These were only considered when there was no choice but to split a WO because the sum of sequential operation sets could not fit in a single SWP. For example, in one SWP, FMFCS might remove multiple interference items and expose a structural joint to make it ready for a hot work welding operation. In this first SWP, additional work is added to prime the exposed steel with a small protection coat, cleanup, and remove the scaffolding. In the next SWP, the scaffolding is replaced, the primer coat is removed, and the WO can continue. This is undesired since the additional set-up and cleanup is repeated, but it is done in rare occasions if needed. This was the motivation for the adjacency constraint formulation in subsection 5.1.3 and numerical experiment set 4.

The major takeaways from this experiment were many instances of user feedback and documented interviews with planning and scheduling staff used throughout this thesis. It is evident that software solve time was more important than an "optimal"

solution because the underlying planning data was constantly changing and because the SGS solver could be frequently used to correct inaccurate planning data. Having good planning data is the most important characteristic of a scheduling exercise. Users preferred the serial SGS methodology because it scheduled each activity in the user-preferred order, giving them more control over the resultant schedule, and it was more readily understood by the scheduler. A note should be made that this perspective came from users that were accustomed to manual scheduling. It is hypothesized that if users become comfortable with automatic scheduling, then perhaps optimization to compare with easily understood scheduling schemes like the serial SGS may be attractive to users.

The re-scheduling scenario was found to be more complicated and re-scheduling with FMFCS DRMIS data was impractical without a major change in how business is performed. This is because the original WOs have already been scheduled and their associated resources have already been used. Due to this, attempting to re-schedule all jobs again (with a few new ones) without un-scheduling the already-scheduled activities will produce a poor result because the original resources are already "consumed". To re-schedule during project execution, a macro would have to be used to "un-consume" the resources used in the original schedule prior to re-scheduling, but in the near term, since all projects compete from the same resource pool and critical resources are already fully consumed, this will cause an available resource profile that matches only the activities that were "un-consumed", meaning that this work will not be able to be scheduled elsewhere than where it is already scheduled. Moreover, since PLs from different ships compete for the same centralized resources, actually un-scheduling operations in DRMIS would pose a significant risk where another ship's PL could schedule operations from a separate project and consume the recently released shop resources, before the re-scheduling PL could re-schedule his/her WOs.

Due to the discovered increased challenges in re-scheduling, additional consideration was placed to research the NSWPP re-scheduling scenario as done in sections 5.2, 6.2 and 6.3. It is proposed that to successfully make use of optimization software for NSWPPs, a centralized approach to using resources and re-scheduling should

be adopted, such as re-scheduling all active and upcoming projects at the same time on a weekly basis.

Chapter 7: Future Research Extensions

Throughout this research, many ideas and potential avenues for research arose. However, there was only a limited amount of time to focus efforts. Below are many of the ideas and research potentials that were found but not significantly explored.

The average of early start (ES) and late start (LS) as a serial SGS sorting method for precedence dominated projects, such as in the PSPLIB, could be analyzed for potential benefits as a serial and/or parallel SGS pre-sorting method. This analysis should really be completed for larger projects. Small PSLIB problems with 120 activities highly precedence-dominated are relatively trivial to complete. Combining many small problems, modeled as work packages, with a shared resource pool, for a total of thousands of activities, is desirable. Metaheuristics that calculate small problems thousands of times with SGSs will no longer work well when each iteration takes several seconds instead of microseconds. Different methods that intelligently suggest sorting methods and select the best one, with only 20-30 schedules calculated by SGS, may be operationally effective with these larger problems.

Analysis of real-world post-work period data could be completed to determine if the average priority-1 duration weighted centroid (DWC), in relation to project makespan, is a worthwhile assessment criterion of overall risk. Although, this was suggested in this thesis, this was not analyzed in depth and no experiments were performed (perhaps with simulation) to confirm project completion probabilities. It is however possible that due to the high variability of scope growth, two random projects with the same ratio could be very different in terms of complexity and difficulty, where this ratio is insufficient or a good indicator of overall risk.

The re-scheduling overtime formulation that increases appropriate resources up to a maximum value may be re-formulated and examined further with experimentation or simulation, not to shorten activity durations, but to schedule work earlier in time periods where work could not otherwise begin.

Meta-heuristics, where a very large number of serial SGS solutions are examined by alternating the activity list order, may be examined to solve problems with the NSWPP architecture, where many WOs, having no precedence relationships between each other, have in themselves many activities with precedence relationships and resources come from a centralized pool. These could be compared to other solving methods used in this research. These metaheuristics could have a special feature where only entire WOs are moved around and the activities within the WOs have adjacency constraints to prevent them from being separated; this is how our interviewed schedulers, PLs, and PMs wanted their work arranged after-all.

All the formulations and heuristics may be extended to include varying resource capacities, as the real-world projects from FMFCS showed a need to schedule in a varying resource capacity environment. Anecdotally, the varying resource capacity environment may lend itself well to re-scheduling overtime formulations where additional resources (or overtime of same resources) may allow activities to be scheduled in discrete time intervals where they otherwise would not be able to fit.

Further research may be done in the multi-project scheduling and re-scheduling environment where resources come from a shared pool. The centralized multi-project approach could be just what a repair facility like FMFCS needs to increase worker utilisation. Even a serial SGS-based scheduling program, used in a multi-project centralized approach, particularized for a shipyard/dockyard and its ships/boats, could improve the current state significantly.

Since the NSWPP environment is one with many space/compartment resources (300+) and many capacity-limited specialty shops and/or sub-contracting resources (70+), analysis of this data may show that WOs have no resource connections with each other may be made. Furthering this potential finding, there may be a way to divide a problem into resource-unrelated subsets, or subsets that barely compete for resources (perhaps only for resources that are far from being bottleneck resources), and having these subsets

solved to optimality, leading to very good NSWPP RCPSP solutions within a reasonable solution time.

Prabhu (2020) deals with the same type of NSWPP RCPSP problems by studying the benefits of decomposition math-heuristics, where only a subset of a project's activities may be scheduled to optimality in a multi-step approach, leading to projects that are almost optimal, but require much less computation time than solving an entire NSWPP RCPSP in one single pass. This research may open the door to comparison with meta-heuristic approaches or other promising approaches that place a high importance on having good schedules solved in a reasonable amount of time.

Combining the math-heuristic work from Prabhu (2020) with the serial SGS findings of this thesis may be a way to perform the NSWPP RCPSP re-scheduling scenario in a heuristic method that leads to solution optimality. During project execution, if a schedule is disrupted and needs to be repaired, either by adding new activities and/or by increasing the duration of activities, then by locally solving the solution space to optimality (perhaps for 100 related near-present term activities) with overtime or extra evening/nighttime shifts, this may produce absolutely no schedule disruptions outside of the solution space, leading to no longer-term schedule changes. Since the solver is used to solve only a subset of the entire problem size, then it can quickly produce a solution that is optimal.

From experimentation and study of NSWPP RCPSP scenarios, there is anecdotal evidence to suggest that, perhaps even more importantly than optimal scheduling, is the need to have the best resource capacity across resource types. Certain bottleneck resources can cripple a project and it is in management's best interest to ensure that critical resources are grown or limited to match the needed capacity of a fleet, and that this capacity information is kept accurate. For example, this author has already discussed with naval officers a potential future program to use automatic scheduling to compare scheduling scenarios with varying resources, such as an additional resource per shop in

each scenario. This comparison can be used to quantitatively assess the benefit or hiring more of each type of resource, and target management's hiring efforts.

Experiment set 4 showed interesting results in terms of reducing solution times while incorporating mandatory adjacency constraints into the formulation. It would be interesting to develop an aggregate planning scheduling scheme where resources from all activities in a WO were spread out and fractionalized over the entire duration of the WO. This would cause a WO to be scheduled as a single activity with multiple resources assigned for the entire duration; thus, significantly reducing problem complexity. For example, the upcoming or most important (priority or prioritized list) 300 activities-worth of WOs may be scheduled using the multi-mode priority-duration formulation or the adjacency variant, while the next 2000 activities beyond the 300 may be scheduled with a spreading-and-fractionalizing scheme. Comparisons with a serial SGS that mimicked human scheduling could be appropriate.

Additionally, it would be interesting to investigate a combined formulation with adjacency constraints into this problem type, but not necessarily with all activities in each work packages. Perhaps slack activities may have a user-defined field that allows them to be scheduled more freely, thus combining adjacency and non-adjacency constraints. This can be further complicated with multi-modes for duration reduction, a multi-project environment, and with varying resource capacities over time to align more appropriately with the real-world problem that repair facilities may face.

A combination of these exact formulations with heuristics to improve scheduling time, even at the cost of solution quality, would likely be most beneficial since a short solution time is very important.

When analyzing Primavera P6 activity relationship constraints, it became obvious that to build add-on software that accommodated any output file, all constraints types would have to be formulated, if these were to be used in a discrete time optimization format compatible with the priority-duration formulations presented in this thesis. All activity dates constraints can be handled with pre-processing to manipulate the ES and LS

of activities, with the exception of “As-Late-As-Possible”, this would require a variant objective function that inverses the time penalty: $(1 - \varepsilon_2 t)$. The classic predecessor constraint is called a “finish to start” constraint in primavera P6; however, three more activity relationship constraints exist: “finish to finish” (the successor activity cannot finish until its predecessor finishes), “start to start” (the successor activity cannot start until its predecessor starts), and “start to finish” (the successor activity cannot finish until its predecessor starts). Modelling these into an enhanced version of the priority-duration formulation would allow it to be more compatible with available scheduling software. It is proposed that these might be modeled with three additional sets, and with each their own additional constraint equation, similar to the standard predecessor constraint. These can have relationships *i* and *j* for predecessors and successors, with constraints relating to their durations and start times x_{jtm}.

Near the end of the research phase, an important observation for users of NSWPP optimization was made in terms of how WOs may be spread-out over the length of an entire project by using various optimization methods. As part of this research, Thales Canada provided a 500-activity problem with significant precedence complexity, such that heuristic methods were necessary to solve the problem in a reasonable amount of time. Figure 38 shows the network diagram of this larger problem built directly into Primavera P6. Following from left to right, nodes 1, 2, 3, 4, and 9 at the start of the project are zero-duration milestones, as well as nodes 5, 7, 10, 8, and 11 at the end of the project. These zero-duration milestones consume no resources and adding these precedence constraints to zero duration milestones are rarely seen in NSWPPs as a planner would have to add these relationships. At the FMFs, the planners build the WOs and precedence relationships are set within WOs. The scheduler and PL do not have administrative rights to set precedence relationships, so this extra tedium is not even permissible by either party. It also further allows for incorrect priority rules application as discussed in subsection 2.4.6: A predecessor cannot have a lower priority than a successor. If this is done, such as is the case below for the WO with activities 216-233 (priority 2) and the WO

with activities 199-215 (priority 1), then the priority rules are broken and activities cannot be sorted first by priority.

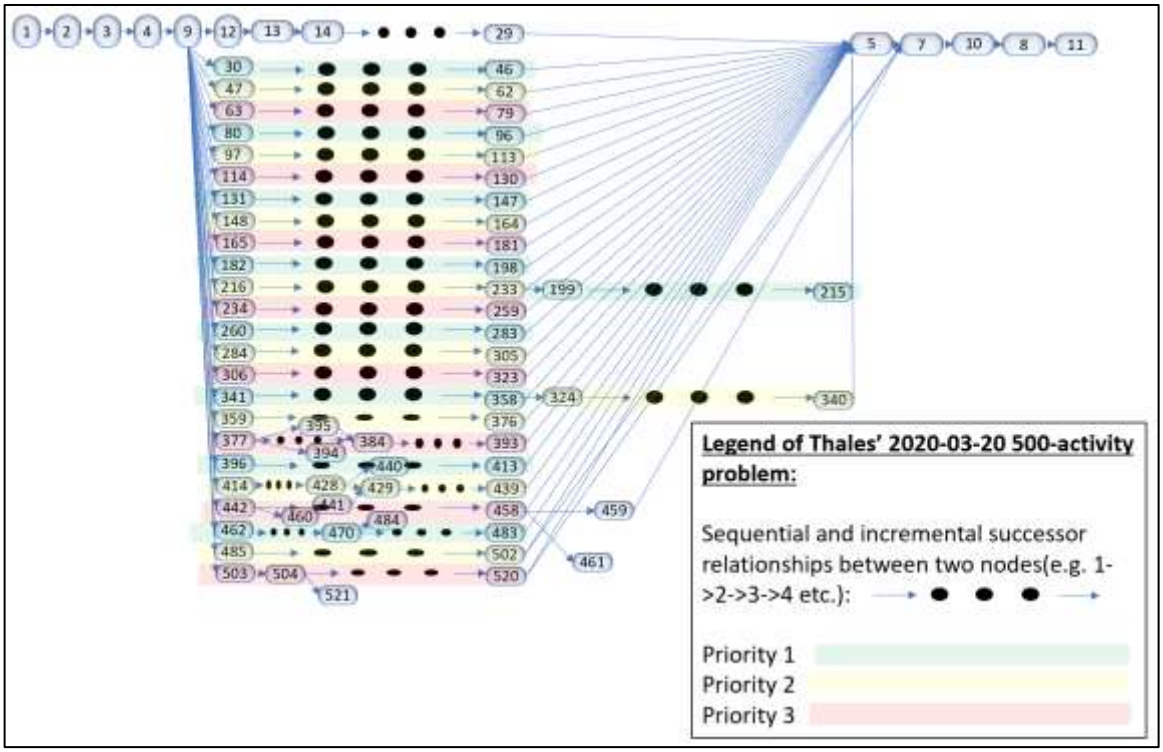


Figure 38: A large practice problem with priority-rule errors and many precedence relationships.

This has negative effects: in the program submitted to Thales Canada by Dalhousie for the refit optimizer, a check had to be included that verified if priority rules were followed, and if not, then the SSGS that gave the solver an initial solution, was forced to first sort activities by ES instead of priority. Otherwise, a precedence relationship could be broken, causing the solver to receive an infeasible initial solution that would therefore be ignored. Since the problem is very complicated, the solver only slightly improves upon this SSGS solution within 10 minutes and the SSGS solution is very close to the final schedule produced (SSGS-PD(α=1.1) combination). Because of ES 1st sorting, the first and last bulk set of activities to be scheduled in the SSGS are from almost all WOs, causing the WOs to stretch over the entire project length. Figure 39 is a simplified depiction of what the Gantt Chart could look like and this would be PL's nightmare, because he/she would

have to manage and verify progress on almost all WO work sites at almost all times throughout the entire length of the project.

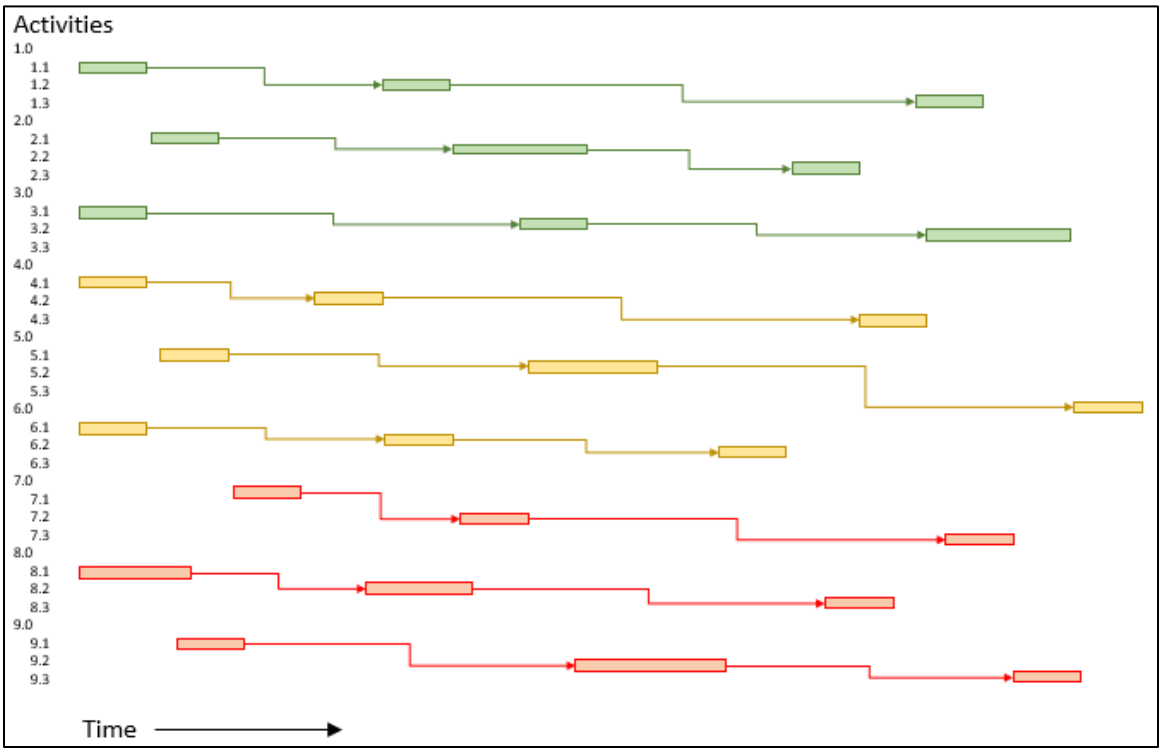


Figure 39: Optimization methods that focus solely on makespan or priority-based methods that resort to ES-ordered SSGS or ES-ordered decomposition produce stretched-out WOs, resulting in schedules that are impractical to manage.

The decomposition method produced by Prabhu (2020) [42] also suffered from this same feature: the priority-rule violation causes the decomposition method to default to ES sorting (rather than priority sorting), to ensure precedence-feasible schedules are produced; thus, although the final solution had a very short makespan, almost every WO was stretched over the bulk of the project makespan, which would lead to PL frustration (or returning to manual scheduling). To work around this problem, the PD(adj) formulation might be necessary, although the makespan and pri-1 DWC could suffer from inflexibility in finding better solutions, as slight WO stretching may improve bottleneck resource utilisation. If the priority-rules are corrected however, then the resulting project Gantt Chart for the SSGS-PD(α=1.1) combination method will be more palatable to PLs, because it will only bunch the scheduling of activities by priority level, resulting in much fewer WOs to track during project execution. Figure 40 illustrates what this project output

could look like. In contrast to figure 39, this project is easier to manage because the PL only needs to manage a reasonable number of work sites in the ship at any given time.

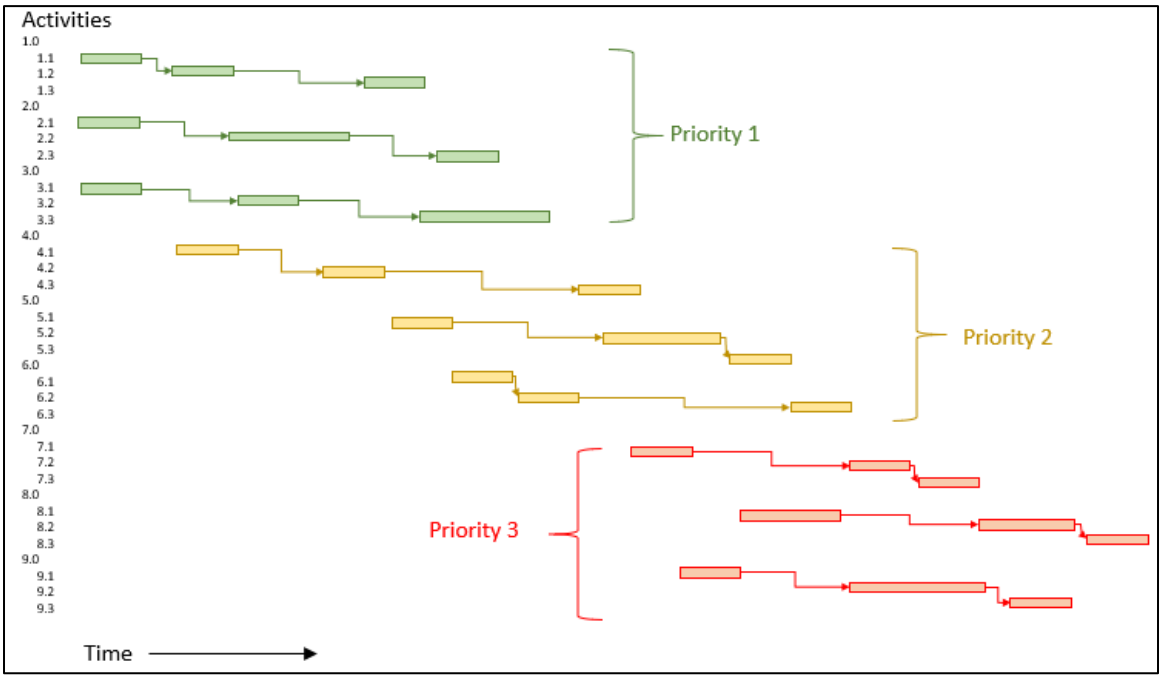


Figure 40: With priority-rules respected (or automatically corrected as proposed), the resulting schedule Gantt Charts are more appealing to PLs, because there will be fewer work sites to manage at any given time.

It is recommended that in order to take full advantage of priority-based formulations (and the heuristics that accompany them for large problems), special care by users should be made to respect the priority precedence rule (see section 2.4.6), which is made much simpler by setting priority at the WO level and not placing precedence relationships between WOs. The ability to modify the priority level of WOs in a refit optimizer interface would add program robustness, and methods to automatically detect and correct priority-rules violations are proposed for future research.

Chapter 8: Conclusions

The research presented in this thesis is that of the Naval Surface Ship Work Period (NSWPP) RCPSP and experiments conducted in this thesis are on RCPSP-topologies that more closely mimic NSWPP problems. These projects consist of many WOs that are mostly un-related by precedence, but each is normally comprised of several activities that are precedence-related within WOs. The WOs conflict with each other by using a shared multi-project centralized worker pool, while using a single project shared compartment resource pool. It is proposed that the majority of the various conflicts can be modeled by resource constraints, ensuring that solution-robust schedules are created.

The multi-mode discrete-time priority-duration RCPSP model presented in this paper finds the optimal solution, in terms of priority-duration weighted front-loading, to initial NSWPP RCPSPs, by scheduling as many activities as possible as early as possible in a project, while considering the important aspects of priority and duration. From experimentation, it was demonstrated that this NP-hard problem will not solve to optimality within a reasonable period of time for large projects, especially when considering the size of real-world problems. For larger problems were adjacency between activities in the same WOs makes sense, the adjacency version of this formulation will produce results for even larger problems in less time. Despite this and based on several interviews with current NSWPP schedulers, project leaders, and project managers, it is proposed that any optimization software has a fast-built-in heuristic scheduler (such as a serial SGS) that allows users to sort all activities by their preferred priorities, and schedule them very quickly and conveniently, while being easy to model all resource constraints. This introduces users to automatic-scheduling and makes the leap to optimization a smaller one, where optimal results can be compared to conventional ones. Afterall, the user of this add-on software may very likely not have access to automatic scheduling in their ERP system, and a standalone add-on program that works in a sandbox could be a great place to experiment, even if it cannot perfectly optimize a schedule.

As found from introducing automatic scheduling at FMFCS, users are busy and they require analysis of a large number of scenarios to effectively weigh options against each other, where the underlying data may be modified at each iteration. For this to be possible, NSWPP scheduling software must be able to produce feasible solutions very quickly, even if these are not optimal, and must be able to conveniently model the conflicts and preferences that should be considered in project scheduling.

For the re-scheduling scenario, where the schedule was disrupted, the goal may be to reduce total deviations or deviation-days. In the real-world situations observed, costs are not included in scheduling information, and the overtime and alternative execution modes are not known; moreover, acquiring this data is often impractical for all activities and at all time periods throughout a project. To aid the scheduler in investigating alternative execution modes or overtime, or to find the most undisturbed schedule that is possible without reduced-duration modes, the multi-mode re-scheduling discrete-time deviation-days priority RCPSP model is proposed. This model determines the activities that will give the most benefit to the project in terms of minimizing deviation days.

References

[1] J. Couch, "Performance Robust Project Scheduling Policies for Naval Ship Maintenance," Dalhousie University, Halifax, 2016.

[2] J. MacLean, Interviewee, Interview with Thales Head ERP Manager. [Interview]. 22 May 2019.

[3] R. Luciano, T. Doucette, A. Mitchell, M. Noel, D. Horne, S. Larade, J. Derby and S. Aaron, Interviewees, Interviews and Prototype Work Meetings with Project Managers and Project Leaders on Real Projects at FMFCS. [Interview]. 28 October, 4 November, 29 November, 9 December,12 December 2019.

[4] P. Jeffery, Interviewee, Interview with Thales Project Manager. [Interview]. 13 May 2019.

[5] S. VanderByl, Interviewee, Interview Q&A with FMFCB Project Manager. [Interview]. 12 June 2019.

[6] V. Javorr, "Electromagnetic Interference between Cranes and Broadcasting Antennas," International Journal of Antennas and Propagation, vol. 2015, no. URL: https://www.hindawi.com /journals/ijap/2015/452962/, pp. 1-10, 2015.

[7] "MARLANT HQ N52 Operational Readiness: CFCD 102 (L) RCN Combat Readiness Requirements," Royal Canadian Navy Internal Document, Halifax, 2016.

[8] "The Wednesday Report: Questions Remain After Canadian Navy Diver's Deaths," 13 February 1991. [Online]. Available: http://navydiver.ca/Documents/News/ Margaree_Deaths.pdf. [Accessed 31 January 2020].

[9] Corrosionpedia, "Salt Fog," Corrosionpedia, [Online]. Available: https://www.corrosionpedia.com/definition/1000/salt-fog. [Accessed 5 November 2019].

[10] GoC, "Canadian Occupational health and Safety Regulations SOR/86-304 Canada Labour Code," Ottawa, 1986.

[11] "The climat and weather in Halifax, Nova Scotia," livingin-canada.com, 2019. [Online]. Available: https://www.livingin- canada.com/climate-halifax.html. [Accessed 16 08 2019].

[12] "Weather & Climate Vancouver British Columbia," 2019. [Online]. Available: https://weather-and-climate.com/average-monthly-Rainy-days,vancouver,Canada. [Accessed 13 November 2019].

[13] "Best Practices for Planners and Schedulers Whitepaper," Prometheus Group, 2019.

[14] F. FMFCS, "Work Period Planning Process (WP3)," Royal Canadian Navy, Halifax and Esquimalt, 2014.

[15] "Commander RCN under QR&O 4.12; NAVORD 3120-2 RCN Fleet Scheduling Process," Royal Canadian Navy Internal Document, Halifax, 2013.

[16] R. C. Navy, "Operational Deficiency Reporting System Standard Operating Procedures (OPDEF SOPs)," Maritime Command Atlantic, Halifax, 2015.

[17] D. 3, "Operation and Maintenance Propulsion Operating Manual applicable to Halifax Class," ADM(Mat), Ottawa, 2014.

[18] R. Kolisch and A. Sprecher, "PSLIB - A project scheduling problem library," European Journal of Operations Research, vol. 96, no. 1, pp. 205-216, 1996.

[19] E. Croteau, "An On-Event Based Model for Resource Constrained Scheduling of Aircraft Heavy Maintenance Tasks," Dalhousie University, Halifax, 2015.

[20] L. Durocher, "AJISS Performance Work Statement," Assistant Deputy Minister of Materiel, Ottawa, 2016.

[21] L. Ozdamar and G. Ulusoy, "A Survey on the Resource-Constrained Project Scheduling Problem," IIE Transactions - October 1995, vol. 27, pp. 574-586, 1995.

[22] J. Weglarz, J. Jozefowska, M. Mike and G. Waligora, "Project scheduling with finite or infonite number of activity processing modes - A survey," European Journal of Operational Research, vol. 208, pp. 177-206, 2011.

[23] R. Pellerin, N. Perrier and f. Berthaut, "A survey of hybrid metaheuristics for the resource-constrained project scheduling problem," European Journal of Operational Research, vol. 280, pp. 395-416, 2020.

[24] J. E. Kelley and M. R. Walker, "Critical-Path Planning and Scheduling," 1959 Proceedings of the Eastern Joint Computer Conference, pp. 160-173, 1959.

[25] F. S. Hillier and G. J. Lieberman, Introduction to Operations Research Tenth Edition, Standford University, New York: McGraw Hill Education, 2015.

[26] J. Blazewicz, J. K. Lenstra and K. A. H. G. Rinnooy, "Scheduling subject to resource constraints," Discrete Applied Mathematics, vol. 5, pp. 11-24, 1983.

[27] S. W. Kramer and J. L. Jenkins, "Understanding the basics of CPM calculations," in Project Management Institute, Seattle WA, 2006.

[28] D. G. Malcolm, J. H. Roseboom, C. E. Clark and W. Fazar, "Application of a technique for research and development program evaluation," Operations Research, vol. 7, no. 5, pp. 646-669, 1959.

[29] J. D. Wiest, The Scheduling of Large Projects with Limited Resources, Carnegie Institue of Technology, 1963.

[30] J. D. Wiest, "A heuristic model for scheduling large projects with limited resources," Management Science, vol. 13, no. 6, pp. B359-B377, 1967.

[31] A. A. Pritsker, L. J. Waiters and P. M. Wolfe, "Multiproject scheduling with limited resources: A zero-one programming approach," Management Science, vol. 16, no. 1, pp. 93-108, 1969.

[32] N. Christofides, R. Alvarez-Valdez and J. M. Tamarit, "Project scheduling with resource constraints: A branch and bound approach," European Journal of Operational Research, vol. 29, no. 3, pp. 262-273, 1987.

[33] R. Kolisch, A. Sprecher and A. Drexl, "Characterization and Generation of a General Class of Resource-Constrained Project Scheduling Problems," Management Science, vol. 41, no. 10, pp. 1693-1703, 1995.

[34] NSWPP RCPSP Initial Practice Set Created by Dr. Robert Pellerin, 2019.

[35] A. B. Tenera, "Critical chain buffer sizing: a comparative study," in PMI® Research Conference: Defining the Future of Project Management, Warsaw, Poland, 2008.

[36] S. Rostami, S. Creemers and R. Leus, "New strategies for stochastic resource-constrained project scheduling," Springer Science+Business Media New York, vol. 21, pp. 349-365, 2017.

[37] G. Optimization, "Gurobi Optimization Documentation: MIP starts," [Online]. Available: https://www.gurobi.com/documentation/9.0/examples/mip_starts.html. [Accessed 3 March 2020].

[38] L. Leach, Critical Chain Project Management, Norwood: EUA: Artech House, 2000.

[39] S. Smith, "Reactive Scheduling Systems," Carnegie Melton University, Pittsburgh, 1999.

[40] N. Nudtasomboon and S. U. Randhawa, "Resource-constrained project scheduling with renewable and non-renewable resources and time-resource tradeoffs," Computers & Industrial Engineering, vol. 32, no. 1, pp. 227-242, 1997.

[41] C. K. Anderson, "Optimization of Continuous Maintenance Availability Schedulig," Naval Post Graduate School, Monterey California, 2014.

[42] S. Prabhu, "Matheuristic approaches for large scale resource constrained scheduling," Dalhousie University, Halifax, 2020.

[43] D. C. W. Applegate, "A computational study of job-shop scheduling," ORSA Journal on Computing, vol. 3, no. 2, pp. 149-56, 1991.

[44] R. Alvarez-Valdez and J. M. Tamarit, "The project scheduling polyhedron, facets and lifting theorems: dimension, facets and lifting theorems," European Journal of Operational Research, vol. 67, no. 2, pp. 204-220, 1993.

[45] C. Artigues, P. Michelon and S. Reusser, "Insertion techniques for static and dynamic resource-constrained project scheduling," European Journal of Operational Research, vol. 149, no. 2, pp. 249-267, 2003.

[46] J. C. Zapata, B. M. Hodge and R. G. V, "The multimode resource constrained multiproject scheduling problem: alternative formulations," INFORMS journal on Computing, vol. 18, no. 3, pp. 377-390, 2006.

[47] O. Koné, C. Artigues, P. Lopez and M. Mongeau, "Event-based MILP models for resource-constrained project scheduling problems," Computers & Operations Research, vol. 28, no. 2011, pp. 3-13, 2011.

[48] S. Employees, Interviewee, Interview with SeaSpan Employees. [Interview]. 2017.

[49] R. H. Möhring, "Minimizing costs of resource requirements in project networks subject to a fixed completion time," Operations Research, vol. 32, no. 1, pp. 89-120, 1984.

[50] J. Coelho and M. Vanhoucke, "Multi-mode resource-constrained project scheduling using RCPSP and SAT solvers," European Journal of Operational Research, vol. 213, no. 1, pp. 73-82, 2011.

[51] A. Schutt, T. Feydy, P. J. Stuckey and M. G. Wallace, "Solving RCPSP/max by lazy clause generation," Journal of Scheduling, vol. 16, no. 3, pp. 273-289, 2013.

[52] C. Schwindt and J. Zimmermann, Handbook on Project Management and Scheduling Vol.1, Springer International Publishing, 2015.

[53] Y. S. Wang, C. Y. Liang and Y. Z. Ju, "Model and resolution approach for problem of multi-project time-cost trade-off," Computer Engineering Applications, vol. 46, pp. 237-240, 2010.

[54] A. Kastor and K. Sirakoulis, "The effectiveness of resource levelling tools for resource constraint project scheduling problem," International Joint Project Management, vol. 27, pp. 493-500, 2009.

[55] K. Neumann and J. Zimmermann, "Procedures for resource leveling and net present value problems in project scheduling with general temporal and resource constraints," European Journal of Operations Research, vol. 127, no. 2, pp. 425-443, 2000.

[56] G. Gatica, L. G. Papageorgiou and N. Shah, "Capacity planning under uncertainty for the pharmaceutical industry," Chemical Engineering Research, vol. 28, no. 8, pp. 665-678, 2003.

[57] S. Creemers, S. Van de Vonder and E. Demeulemeester, "2014," Annals of Operations Research, vol. 213, no. 1, pp. 27-65, 2014.

[58] S. Hartmann, "Project scheduling with resource capacities and requests varying with time: a case study," Flexible Services Manufacturing, Springer Science+Business Media, LCC 2012, 2012.

[59] S. Van de Vonder, "Proactive-reactive procedures for robust project scheduling," Ph.D. dissertation, KU Leuven, Leuven, 2006.

[60] B. Gates, "Bill Gates Press Room Speech," 23 September 2005. [Online]. Available: https://www.gatesfoundation.org/media-center/speeches/2005/09/bill-gates-lakeside-school. [Accessed 21 November 2019].

[61] J. Blazewicz, K. Ecker, E. Pesch, G. Schmidt and J. Weglarz, Handbook on Scheduling, Berlin: Springer, 2007.

[62] R. Kolisch, "Serial and parallel resource-constrained project scheduling methods revisited: theory and computation," European Journal of Operations Research, vol. 90, pp. 320-333, 1996.

[63] H. Fahimi, "Efficient algorithms to solve scheduling problems with a variety of optimization criteria," Universite de Laval, Montreal, 2016.

[64] A. Sprecher, R. Kolisch and A. Drexl, "Semi-active, active, and non-delay schedules for the resource-constrained project scheduling problem," European Journal of Operations Research, vol. 80, pp. 94-102, 1995.

[65] R. Kolisch and S. Hartmann, "Heuristic algorithms for the resource-constrained project scheduling problem: classification and computational analysis," Project Scheduling: recent models, algorithms and applications, pp. 147-178, 1999.

[66] E. Demeulemeester and W. Herroelen, Project scheduling: a research handbook, Boston: Kluwer, 2002.

[67] J. Dorn, "Evaluating Reactive Scheduling Systems," in IEEE/WIC/ACM International Conference on Intelligent Agent Technology, 2004.

[68] F. Stork, "Stochastic resource-constrained project scheduling," PhD Thesis, Technical University of Berlin, Berlin, 2001.

[69] R. Alvarez-Valdes and J. Tamarit, "Heuristic algorithms for resource-constrained project scheduling: a review and an empirical analysis," in Advances in Project Scheduling, 1989, pp. 113-134.

[70] S. Van de Vonder, F. Ballestin and W. Demeulemeester, "Heuristic procedures for reactive project scheduling," Computers & Industrial Engineering, vol. 52, pp. 11-12, 2007.

[71] J. Mendes, "Sistema de apoio à decisão para planeamento de sistemas de produção do tipo," Ph.D. dissertation, English translation, Universidade do Porto, Porto, 2003.

[72] J. F. Gonçalves and M. G. Resend, "An extended akers graphical method with a biased random-key genetic algorithm for job-shop scheduling," International Trans Operations Research, vol. 21, no. 2, pp. 215-246, 2014.

[73] F. Berthaut, R. Pellerin, A. Hajji and N. Perrier, "A path relinking-based scatter serach for the resource-constrained project scheduling problem," International Journal of Project Organisation and Management, vol. 10, no. 1, pp. 1-36, 2018.

[74] L. F. Muller, "An adaptive large neighborhood search algorithm for the resource-constrained project scheduling problem," in Proceedings of the VIII metaheuristics international conference, MIC, 2009.

[75] R. Argawal, M. K. Tiwari and S. K. Mukherjee, "Artificial immune system based approach for solving resource constraint project scheduling problem," The International Journal of Advanced Manufacturing Technology, vol. 38, no. 1, pp. 44-50, 2007.

[76] L. Bukata, P. Sucha and Z. Hanzalek, "Solving the resource constrained project scheduling problem using the parallel tabu search designed for the CUDA platform," Journal of Parallel and Distributed Computing, vol. 77, pp. 58-68, 2015.

[77] R. M. Chen, "Particle swarm optimization with justification and designed mechanisms for resource-constrained project scheduling problem," Expert Systems with Applications, vol. 38, pp. 7102-7111, 2011.

[78] J. He and X. Chen, "A filter-and-fan approach with adaptive neighborhood switching for resource-constrained project scheduling," Computer & Operations Research, vol. 71, pp. 71-81, 2016.

[79] N. Nouri, S. Krichen and T. Ladhari, "A discrete artificial bee colony algorithm for resource-constrained project scheduling problem," In Proceedings of the 5th international conference on modeling, simulation and applied optimization, vol. ICMSAO:IEEE, p. 106, 2013.

[80] P. Jędrzejowicz and E. Ratajczak-Ropel, "Reinforcement learning strategies for A-team solving the resource-constrained project scheduling problem," Neurocomputing, vol. 146, pp. 301-307, 2014.

[81] E. Goldratt, Critical Chain, Great Barrington: The North River Press, 1997.

[82] L. Leach, Critical Chain Project Management Improves Project Performance, Idaho Falls: Advanced Project Institute, 2000.

[83] T. Raz, R. Barnes and D. Dvir, "A Critical Look at Critical Chain Project Management," Project Management Journal, no. 34, pp. 24-32, 2003.

[84] D. Trietsch, "Why a critical path by any other name would smell less sweet?," Project Management Journal, vol. 36, pp. 27-36, 2005.

[85] L. P. Leach, Lean Project Management: 8 Principles for Success, Boise: Advanced Projects, 2005.

[86] O. I. Tukel, W. O. Rom and S. D. Eksioglu, "An investigation of buffer sizing techniques in critical chain scheduling," European Journal of Operational Research, vol. 172, pp. 401-416, 2006.

[87] L. C. D. H. 2004 and S. L. S. E. S. T. P. Issue, "Shipyard Log," 8 Apri 2004. [Online]. Available: www.phnsy.navy.mil. [Accessed 15 June 2019].

[88] "Report on Transformation 2011," Government of Canada, 23 August 2016. [Online]. Available: https://www.canada.ca/en/department-national-defence/corporate/reports-publications/report-on-transformation-2011.html. [Accessed 5 November 2019].

[89] D. Pinhas and R. Ahluwailia, "Decision Support System for the Ship Repair Industry," in Proceedings of the 2nd Annual World Conference of the Society for Industrial and Systems Engineering, Las Vegas, 2013.

[90] J. Grey, "Buffer Techniques for Stochastic Resource Constrained Project Scheduling with Stochastic Task Insertions Problems," University of Central Florida, Orlando, 2007.

[91] V. Saihjpal and S. B. Singh, "New Placement Strategy for Buffers in Critical Chain," in Proceedings of the Second International Conference on Soft Computing for Problem Solving, 2012.

[92] E. Goldratt, Critical Chain, Great Barrington: The North River Press, 1997.

[93] E. Szelke and R. Kerr, "Knowledge-bsed reactive scheduling," Production Planning & Control, vol. 5, no. 2, p. 124, 1994.

[94] S. F. Smith, "Reactive scheduling systems," in D.E. Brown, & W. T. Scherer, Intelligent Scheduling Systems, Springer UX, 1995, pp. 155-192.

[95] A. Noroddin and S.-E. Faramarz, "RePro-Active: a reactive-proactive scheduling method based on simulation in cloud computing," Supercomputing, vol. 74, pp. 801-829, 2018.

[96] F. Deblaere, E. Demeulemeester and W. Herroelen, "Proactive policies for the stochastic resource-constrained project scheduling problem," European Journal of Operational Research, vol. 214, pp. 308-316, 2011.

[97] T. Chaari, S. Chaabane, N. Aissani and D. Trentesaux, "Scheduling under uncertainty: survey and research directions," in International Conference on Advanced Logistics and Transport, Valenciennes, 2014.

[98] P. Renna, "Job shop scheduling by pheromone approach in a dynamic environment," International Journal of Computer Integrated Manufacturing, vol. 23, no. 5, pp. 412-424, 2010.

[99] T. C. Sun, K. K. Lai, K. Lam and K. P. So, "A study of heuristics for bidirectional multi-hoist production scheduling systems," International Journal of Production Economics, vol. 33, pp. 207-214, 1994.

[100] C. Rajendran and O. Holthaus, "A comparative study of dispatching rules in dynamic flow shops and job shops," European Journal of Operational Research, vol. 116, no. 1, pp. 156-170, 1999.

[101] W. Mouelhi and H. Pierreval, "Training a neural network to select dispatching rules in real time," Computers & Industrial Engineering, vol. 58, no. 2, pp. 249-256, 2010.

[102] W. Herroelen and R. Leus, "Identification and illumination of popular misconceptions about project scheduling and time buffering in a resource-

constrained environment," Journal of the Operational Research Society, pp. 102-109, 2005.

[103] T. A. Guldemond, J. L. Hubrink and J. J. Paulus, "Time-constrained project scheduling," BETA publicatie: working papers, vol. 180, 2006.

[104] A. Schnabel, C. Kellenbrink and S. Helber, "Profit-oriented scheduling of resource-constrained projects with flexible capacity constraints," Business Research, vol. 11, pp. 329-356, 2018.

[105] A. Beljadid, H. Delmaires, I. Hallaoui, M. Gamache, R. Pellerin, C. Rocha and F. Soumis, "Refit Optimizer: multicriteria optimization," Université Polytechnique de Montréal, Montréal, 2019.

[106] G. o. Canada, "Chapter 10 Cost and Pricing," Public Works Government Services Canada, 14 July 2012. [Online]. Available: https://buyandsell.gc.ca/policy-and-guidelines/supply-manual/section/10. [Accessed 9 December 2019].

[107] "Department of National Defence," Government of Canada, 31 October 2019. [Online]. Available: https://www.canada.ca/en/department-national-defence.html. [Accessed October October 2019].

[108] "FMF Cape Scott (Halifax)," Wikimapia, [Online]. Available: http://wikimapia.org/23361128/FMF-Cape-Scott. [Accessed 31 October 2019].

[109] B. Ashtiani, G.-R. Jalali, M.-B. Aryanezhad and A. Makui, "A new approach for buffer sizing in Critical Chain Scheduling," in IEEE IEEM, Tehran, 2007.

[110] M. P. Prajapati and N. B. Yadav, "Buffer Based CCPM Scheduling: A Modern," Kalpa Publications in Civil Engineering, vol. 1, pp. 291-300, 2017.

[111] B. Rummel, "Beyond average: Weibull analysis of task completion times," Journal of Usability Studies, vol. 12, no. 2, pp. 56-72, 2017.

[112] G. George, "Sample Primavera Scheduling File Folder," Seaspan Victoria Shipyards, Victoria, 2019.

[113] G. Optimization, "Gurobi Guidelines: Advanced User Scaling," [Online]. Available: https://www.gurobi.com/documentation/8.1 /refman/numerics_advanced_user_sca.html. [Accessed 15 September 2019].

[114] B. Granger, M. Yu and K. Zhou, "Optimization with absolute values," 3 October 2017. [Online]. Available: https://optimization.mccormick.northwestern.edu /index.php/Optimization _with_absolute_values. [Accessed 24 October 2019].

[115] M. R. Sirkin, ""Two-sample t tests"," in Statistics for the Social Sciences (3rd ed.), Thousand Oaks, CA, SAGE Publications, Inc., 2005, pp. 271-316.

[116] Alexandra, "Critical Chain Buffer Sizing: a Comparative Study," in Paper presented at PMI® Research Conference: Defining the Future of Project Management, Warsaw, Poland, 2008.

[117] J. Caprace, C. Petcu, M. Velarde and P. Rigo, "Optimization of Shipyard Space Allocation and Scheduling Using a Heuristic Algorithm," Journal or Marine Science and Technology, vol. 18, no. 3, pp. 404-417, 2013.

[118] DMSS4/DGMEPM, "Canadian Navy Damage Control Information Book For Halifax Class Ships," Royal Canadian Navy, Ottawa, 2011.

[119] C. Garcia and G. Rabadi, "An Optimization Model for Scheduling Problems with Two-Dimensional Spatial Resource Constraints," Old Dominion University, Hanoi, 2010.

[120] W. Herroelen and R. Leus, "On the Merits and Pitfalls of Critical Chain Scheduling," Katholieke Universiteit, Leuven, Belgium, 2001.

[121] S. Martello, D. Pisinger and D. Vigo, "Three Dimentional Bin Packing Problem," Open Research, 2000.

[122] R. C. Newbold, Project management in the fast lane: applying the theory of constraints, Boca Raton: The St. Lucie Press/APICS series on constraints management, 1998.

[123] V. H. Nguyen, "Optimal Ship Maintenance Scheduling Under Restricted Conditions and Constrained Resources," Old Dominion University, Hanoi, 2017.

[124] S. A. Oke, A. Oluleye, F. Oyawale and C. Owaba, "Sensitivity Analysis of a prenventive maintenance scheduling model," International Journal of Industrial and System Engineering, vol. 3, pp. 298-323, 2008.

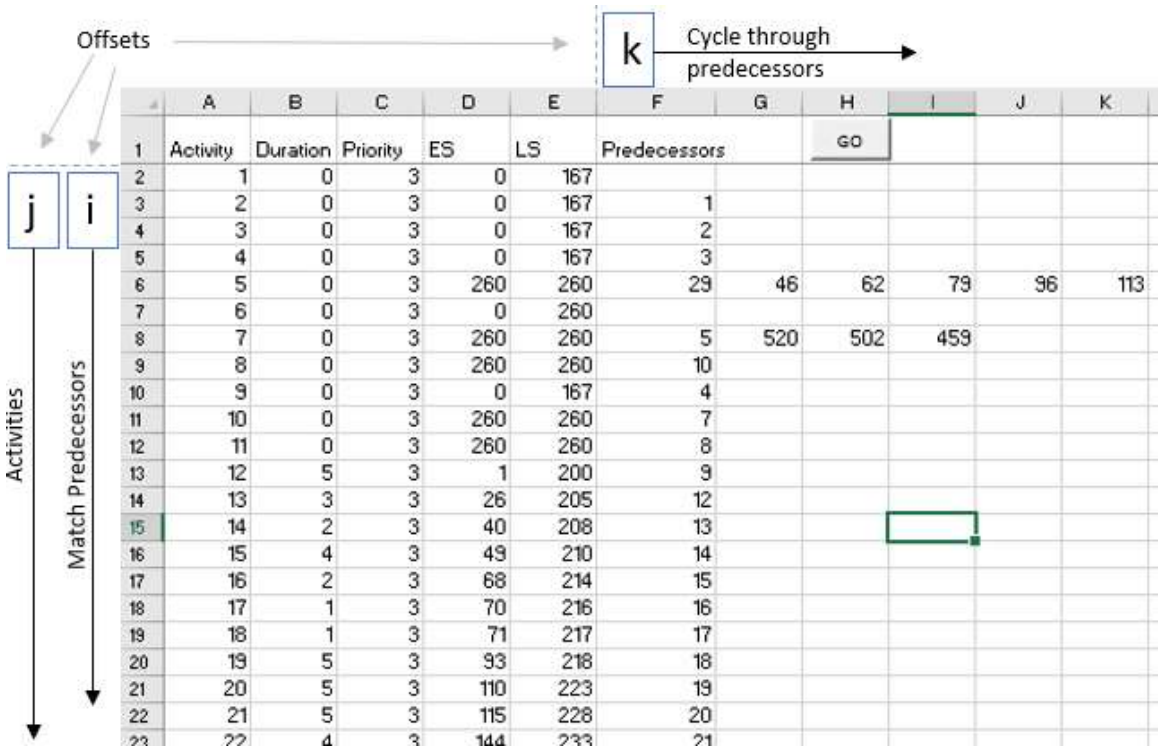
[125] D. C. Pinha and R. S. Ahluwalia, "Flexible resource management and its effect on project cost and duration," Journal of Industrial Engineering International, 2018.

Appendices

8.1 Appendix A – Excel VBA Code for Early-Start, Serial SGS, and Late Start Calculations used for Refit Optimizer Trial

This code reads sheets that have data converted from Primavera P6 into sheets "1" through "12". The code converts data from these 12 sheets into the "SGS" sheet. Data processing then begins on sheet "SGS" and uses sheet "RES" for updating the resource plan in the serial SGS. This is a suitable method with Excel VBA to calculate a forward ES pass, a serial SGS, and a backward LS pass. Many Excel VBA efficiency and error handling improvements are included, that may assist future researchers using Excel VBA.

ES Calculation (Visual Depiction of SGS Sheet):



	A	B	C	D	E	F	G	H	I	J	K
1	Activity	Duration	Priority	ES	LS	Predecessors		GO			
2	1	0	3	0	167						
3	2	0	3	0	167	1					
4	3	0	3	0	167	2					
5	4	0	3	0	167	3					
6	5	0	3	260	260	29	46	62	79	96	113
7	6	0	3	0	260						
8	7	0	3	260	260	5	520	502	459		
9	8	0	3	260	260	10					
10	9	0	3	0	167	4					
11	10	0	3	260	260	7					
12	11	0	3	260	260	8					
13	12	5	3	1	200	9					
14	13	3	3	26	205	12					
15	14	2	3	40	208	13					
16	15	4	3	43	210	14					
17	16	2	3	68	214	15					
18	17	1	3	70	216	16					
19	18	1	3	71	217	17					
20	19	5	3	93	218	18					
21	20	5	3	110	223	19					
22	21	5	3	115	228	20					
23	22	4	3	144	233	21					

VBA Code for ES Forward Pass:

'Calculate ES Loop**********

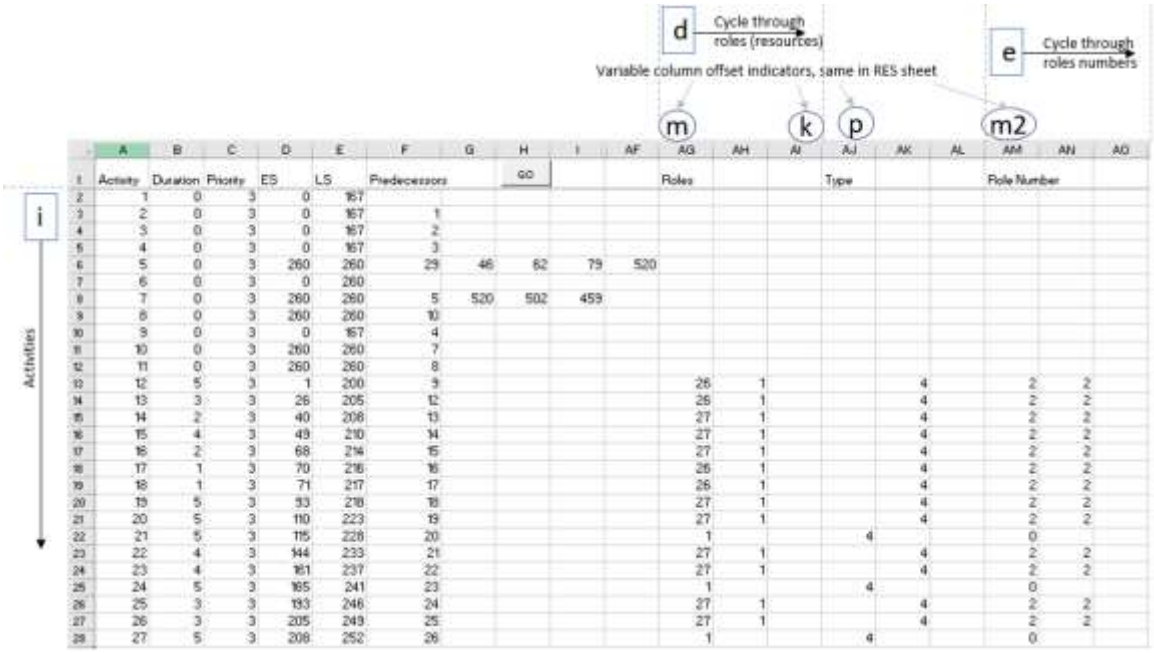
Line1:

j = 0

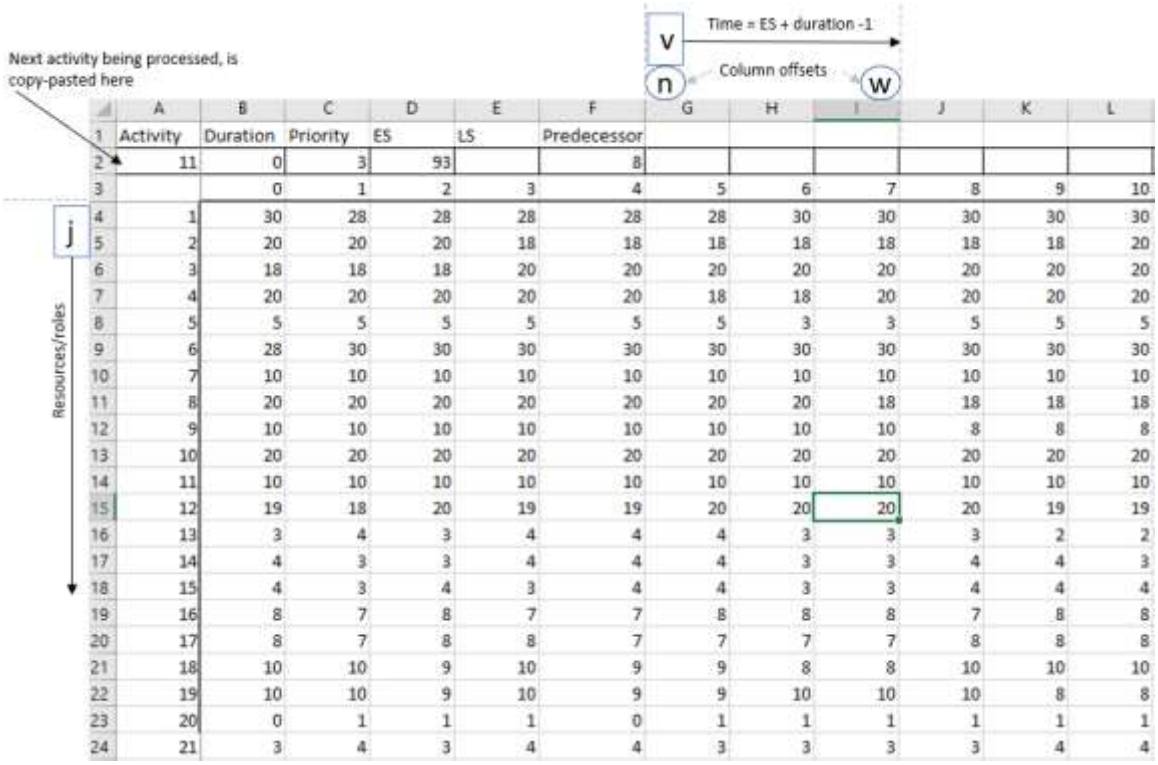
```
        i4 = 0

Do While Sheets("SGS").Range("A2").Offset(j, 0) <> "" 'cycle through all activities j
  If Sheets("SGS").Range("F2").Offset(j, 0) = "" Then
    Sheets("SGS").Range("D2").Offset(j, 0) = 0 'no predecessor means ES is 0
    GoTo Line2 'Skip predecessor checks without predecessors, go to next activity listed by j
  End If
  k = 0
  Do While Sheets("SGS").Range("F2").Offset(j, k) <> "" 'cycle through all predecessor columns
    i = 0 'will go through every row to find matching activity
    Do While Sheets("SGS").Range("F2").Offset(j, k) <> Sheets("SGS").Range("A2").Offset(i, 0) 'Find position of matching activity (F2,j,k)
      i = i + 1
    Loop
    'Activity number now matches predecessor number
    If Sheets("SGS").Range("D2").Offset(i, 0) = "" Then 'Check if predecessor ES exists
      i4 = 1 'indicator that we need to reloop
      GoTo Line2 'Predecessor's ES has not yet been determined, so skip to next activity
    End If
    If Sheets("SGS").Range("D2").Offset(i, 0) = 0 And Sheets("SGS").Range("D2").Offset(j, 0) = "" Then
      Sheets("SGS").Range("D2").Offset(j, 0) = Sheets("SGS").Range("D2").Offset(i, 0) + Sheets("SGS").Range("B2").Offset(i, 0) ‘Set ES as ES of predecessor plus duration of predecessor
      GoTo Line1f
    End If
    If Sheets("SGS").Range("D2").Offset(i, 0) + Sheets("SGS").Range("B2").Offset(i, 0) > Sheets("SGS").Range("D2").Offset(j, 0) Then 'only overwrite if predecessor’s ES + duration is larger than current ES
```

```
        i4 = 1 'indicator that we need to reloop, needed if even only one activity gets an updated ES
        Sheets("SGS").Range("D2").Offset(j, 0) = Sheets("SGS").Range("D2").Offset(i, 0) + Sheets("SGS").Range("B2").Offset(i, 0) 'ES + Duration of predecessor
      End If
Line1f:
      k = k + 1 'increment the predecessor column
   Loop
Line2:
   j = j + 1
Loop

   'Check if all activities are completed for ES looping*****
If i4 = 1 Then
   GoTo Line1
End If
'End ES Loop**************************************
```

Serial SGS Calculation (Visual Depiction of SGS Sheet):



Serial SGS Calculation (Visual Depiction of RES Sheet (a.k.a. resource field)):



	A	B	C	D	E	F	G	H	I	J	K	L
1	Activity	Duration	Priority	ES	LS	Predecessor						
2	11	0	3	93		8						
3		0	1	2	3	4	5	6	7	8	9	10
4	1	30	28	28	28	28	28	30	30	30	30	30
5	2	20	20	20	18	18	18	18	18	18	18	20
6	3	18	18	18	20	20	20	20	20	20	20	20
7	4	20	20	20	20	20	18	18	20	20	20	20
8	5	5	5	5	5	5	5	3	3	5	5	5
9	6	28	30	30	30	30	30	30	30	30	30	30
10	7	10	10	10	10	10	10	10	10	10	10	10
11	8	20	20	20	20	20	20	20	18	18	18	18
12	9	10	10	10	10	10	10	10	10	8	8	8
13	10	20	20	20	20	20	20	20	20	20	20	20
14	11	10	10	10	10	10	10	10	10	10	10	10
15	12	19	18	20	19	19	20	20	20	20	19	19
16	13	3	4	3	4	4	4	3	3	3	2	2
17	14	4	3	3	4	4	4	3	3	4	4	3
18	15	4	3	4	3	4	4	3	3	4	4	4
19	16	8	7	8	7	7	8	8	8	7	8	8
20	17	8	7	8	8	7	7	7	7	8	8	8
21	18	10	10	9	10	9	9	8	8	10	10	10
22	19	10	10	9	10	9	9	10	10	10	8	8
23	20	0	1	1	1	0	1	1	1	1	1	1
24	21	3	4	3	4	4	3	3	3	3	4	4

VBA Code for Serial SGS:

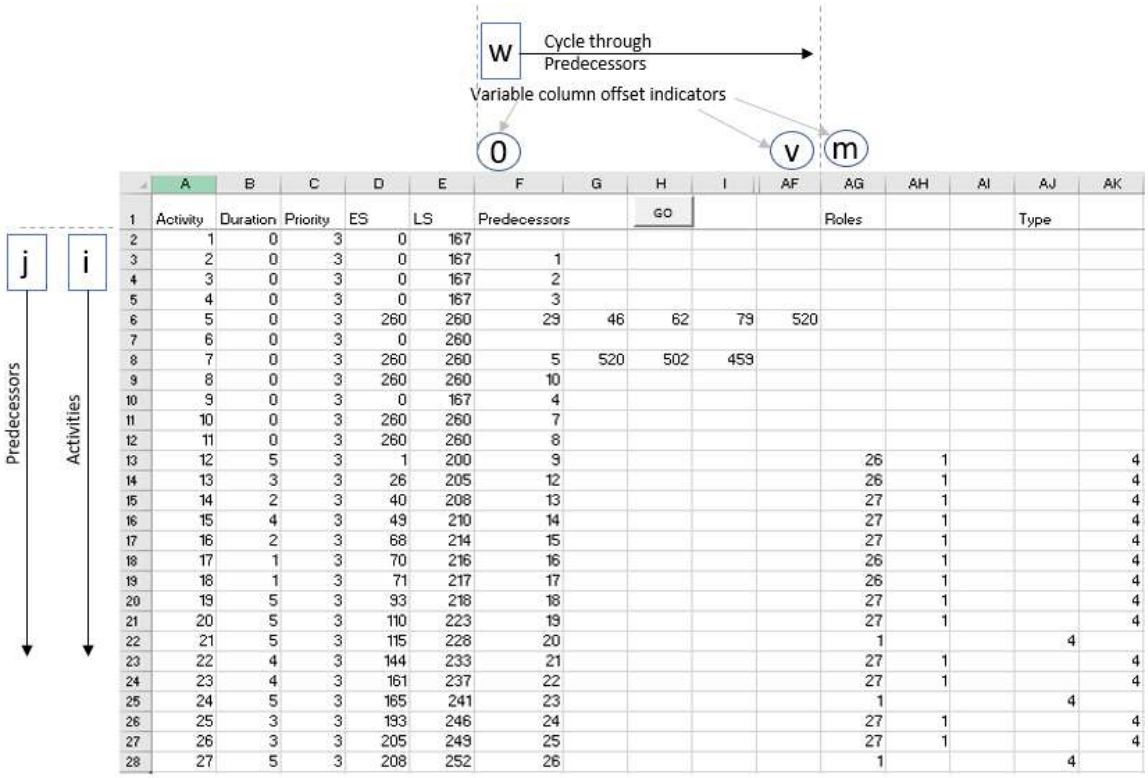
```
'Perform serial SGS, now that activities are sorted by ES and then priority. If check was successful (no lower priority predecessors), this was sorted by priority and then ES


Do While Sheets("SGS").Range("A2").Offset(i, 0) <> ""
   Sheets("RES").Range("A2:ZZ2").Value = Sheets("SGS").Range("A2:ZZ2").Offset(i, 0).Value 'copy next activity above resource table
   f = Sheets("RES").Range("D2") 'maximum start time before considering resources
   n = f 'Current minimum feasible start time f
Line7a:
   k = p - 1
   For d = m To k 'Cycle through all roles
      If Sheets("RES").Range("F2").Offset(0, d) = "" Then 'Stop if role is blank
         GoTo Line7
      End If

      j = 0
      Do While Sheets("RES").Range("A4").Offset(j, 0) <> Sheets("RES").Range("F2").Offset(0, d) 'Find position j for resource row
         j = j + 1
      Loop

      e = m2 + (d - m) 'role number positioning
      n = f 'set start time unit to maximum minimum-feasible time unit found so far
      If Sheets("RES").Range("B4").Offset(j, n) < Sheets("RES").Range("F2").Offset(0, e) Then 'If it does not fit, cycle through duration increments
         Do While Sheets("RES").Range("B4").Offset(j, n) < Sheets("RES").Range("F2").Offset(0, e) 'Keep going until resource can support
            n = n + Sheets("RES").Range("B2") 'job can't start until ES + duration, so increment by another duration (this is more complicated but proven to be more efficient)
```

```
        Loop


        n = n - Sheets("RES").Range("B2") + 1 'We found INTERVAL where activity may
start, so start checking incrementally from last infeasible time unit + 1
      End If
Line8:
      ii = 0
      Do While ii = 0 'Incrementally check all time units from n
        w = n + Sheets("RES").Range("B2") - 1 'start time plus duration minus 1
        For v = n To w 'must check every time unit of the activity
          If Sheets("RES").Range("B4").Offset(j, v) < Sheets("RES").Range("F2").Offset(0,
e) Then 'If is does not fit, increment n and try again
            ii = 0
            n = n + 1
            GoTo Line8 'If unfeasibility exists, increment n and try again
          End If
        Next v
        ii = 1
      Loop 'Now n is equal to first feasible time unit


      If f < n Then
        f = n 'set f to maximum n
        GoTo Line7a
      End If
      j = 0
    Next d


Line7:
```

```
'Activity fits at f, so update the resource table
k = p - 1
w = f + Sheets("RES").Range("B2") - 1
For d = m To k 'Update resource table for every role
e = m2 + (d - m) 'role number positioning
j = 0
Do While Sheets("RES").Range("A4").Offset(j, 0) <> Sheets("RES").Range("F2").Offset(0, d) 'Find position j for resource row
j = j + 1
Loop
For v = f To w 'update resource table for every time unit
Sheets("RES").Range("B4").Offset(j, v) = Sheets("RES").Range("B4").Offset(j, v) - Sheets("RES").Range("F2").Offset(0, e) 'Update correct value in table
Next v
Next d

'Copy new ES time to SGS page, a.k.a. start time of serial ES; note that this start time will not be the ES used in formulation
'This SGS only finds a reasonable first pass at a schedule, and finds a reasonably tight time horizon
Sheets("SGS").Range("D2").Offset(i, 0) = f

'Recalculate ES for next activity in list (i)
j = i + 1 'temporarily set this to next activity row in SGS page

If Sheets("SGS").Range("F2").Offset(j, 0) = "" And Sheets("SGS").Range("A2").Offset(j, 0) <> "" Then
Sheets("SGS").Range("D2").Offset(j, 0) = 0 'no predecessor means ES is still 0
GoTo Line4i 'Skip predecessor checks without predecessors, go to next activity
End If
```

```
k = 0

Do While k < m 'cycle through all predecessor columns, m is still first column for roles

i2 = 0 'will go through every row to find matching activity

Do While Sheets("SGS").Range("F2").Offset(j, k) <> Sheets("SGS").Range("A2").Offset(i2, 0) 'Find position of matching activity (F2,j,k)

i2 = i2 + 1

Loop

'Activity number now matches predecessor number

If Sheets("SGS").Range("D2").Offset(i2, 0) + Sheets("SGS").Range("B2").Offset(i2, 0) > Sheets("SGS").Range("D2").Offset(j, 0) Then 'only overwrite if this is the max function

Sheets("SGS").Range("D2").Offset(j, 0) = Sheets("SGS").Range("D2").Offset(i2, 0) + Sheets("SGS").Range("B2").Offset(i2, 0) 'ES + Duration of predecessor

End If

k = k + 1 'increment the predecessor column

Loop

j = 0

Line4i:

i = i + 1

Loop 'End of Serial SGS loop********Next, we lock-in the time horizon H
```

LS Calculation (Visual Depiction of SGS Sheet):



	A	B	C	D	E	F	G	H	I	AF	AG	AH	AI	AJ	AK
1	Activity	Duration	Priority	ES	LS	Predecessors		GO			Roles			Type	
2	1	0	3	0	167										
3	2	0	3	0	167	1									
4	3	0	3	0	167	2									
5	4	0	3	0	167	3									
6	5	0	3	260	260	29	46	62	79	520					
7	6	0	3	0	260										
8	7	0	3	260	260	5	520	502	459						
9	8	0	3	260	260	10									
10	9	0	3	0	167	4									
11	10	0	3	260	260	7									
12	11	0	3	260	260	8									
13	12	5	3	1	200	9					26	1			4
14	13	3	3	26	205	12					26	1			4
15	14	2	3	40	208	13					27	1			4
16	15	4	3	49	210	14					27	1			4
17	16	2	3	68	214	15					27	1			4
18	17	1	3	70	216	16					26	1			4
19	18	1	3	71	217	17					26	1			4
20	19	5	3	93	218	18					27	1			4
21	20	5	3	110	223	19					27	1			4
22	21	5	3	115	228	20					1			4	
23	22	4	3	144	233	21					27	1			4
24	23	4	3	161	237	22					27	1			4
25	24	5	3	165	241	23					1			4	
26	25	3	3	193	246	24					27	1			4
27	26	3	3	205	249	25					27	1			4
28	27	5	3	208	252	26					1			4	

VBA Code for LS Backward Pass:

```
'Calculate LS*************

Do While ii = 0 'ii only becomes 1 when all LS are determined

   Do While Sheets("SGS").Range("A2").Offset(i, 0) <> ""

      i2 = Sheets("Parameters").Range("L3") - Sheets("SGS").Range("B2").Offset(i, 0) 'reset i2, which is H minus activity duration

      If Sheets("SGS").Range("E2").Offset(i, 0) <> "" Then 'Abort search if LS is already determined

         GoTo Line9

      End If

      j = 0

      v = m - 1 'set to end of predecessor columns

      Do While Sheets("SGS").Range("A2").Offset(j, 0) <> ""

         If Sheets("SGS").Range("F2").Offset(j, 0) = "" Then

            GoTo Line9a
```

```
        End If
        For w = 0 To v 'go through predecessor columns
          If Sheets("SGS").Range("F2").Offset(j, w) = "" Then
            GoTo Line9a
          End If
          If Sheets("SGS").Range("F2").Offset(j, w) = Sheets("SGS").Range("A2").Offset(i,
0) Then
            If Sheets("SGS").Range("E2").Offset(j, 0) = "" Then 'Abort if LS is not yet
determined for successor
              GoTo Line9
            End If
            If Sheets("SGS").Range("E2").Offset(j, 0) -
Sheets("SGS").Range("B2").Offset(i, 0) < i2 Then 'Since LS exists for successor, update i2
              i2 = Sheets("SGS").Range("E2").Offset(j, 0) -
Sheets("SGS").Range("B2").Offset(i, 0) 'set new lower LS
            End If
          End If
        Next w
Line9a:
        j = j + 1
      Loop
      Sheets("SGS").Range("E2").Offset(i, 0) = i2 'Set LS because either no successors
exist or all successors have been determined and i2 is min LS value
Line9:
      i = i + 1
    Loop 'Performed a full pass

    f = 0 'Check if all LS are completed after cycling through all activities***
    i = 0
    Do While Sheets("SGS").Range("A2").Offset(i, 0) <> "" 'loop through activities list
```

```
        If Sheets("SGS").Range("E2").Offset(i, 0) = "" Then
            f = 1
        End If
        i = i + 1
    Loop
    If f = 0 Then
        ii = 1
    End If 'Finish Checking completeness***
    i = 0
Loop
```

8.2 Appendix B – Experiment 3 Summary Data

Note: In all experiments, 5 in-progress activities are delayed by 3 days, at numerical day 22 of the project

Pri-R-DD Solution Time, for 100 activities

Iteration=	0	1	2	3	4	5
Trial 1	10	19	12	17	14	11
Trial 2	12	29	16	9	5	
Trial 3	8	14	12	1		
Trial 4	27	33	30	22	1	
Trial 5	14	24	6			
Trial 6	12	23	12	8	6	
Trial 7	7	3	6			
Trial 8	12	20	23	18	6	
Trial 9	18	24	19	6		
Trial 10	31	37	57	27	5	

Pri-R-DD Deviation Days, for 100 activities

Iteration=	0	1	2	3	4	5
Trial 1	68	38	12	13	6	3
Trial 2	85	30	6	3	0	
Trial 3	34	20	11	0		
Trial 4	47	29	18	12	0	
Trial 5	60	30	0			
Trial 6	65	38	14	5	0	
Trial 7	15	3	0			
Trial 8	45	12	6	4	0	
Trial 9	42	9	6	0		
Trial 10	117	73	44	36	0	

total deviations

0	1	2	3	4	5
14	13	4	1	1	1
29	10	2	1	0	
19	11	7	0		
15	9	8	4	0	
24	10	0			
22	18	5	2	0	
5	1	0			
17	4	2	2	0	
5	3	2	0		
39	35	19	13	0	

Late activity solution	makespan delay
yes	3 0
no	3 3 0
no	3 0
no	3 0
no	3 3 0
no	3 0
no	0
yes	0
no	0
no	3 3 0

Pri-R-DD Solution Time, for 120 activities

Iteration=	0	1	2	3	4	5
Trial 1	20	20	10			
Trial 2	36	107	54	8		
Trial 3	13	12				
Trial 4	43	111	98	47	35	9
Trial 5	30	38	17	9		
Trial 6	8	21	18	9	6	
Trial 7	36	86	71	45	8	
Trial 8	10	17	12	9	8	
Trial 9	14	30	20	12	9	
Trial 10	23	51	30	9		

Pri-R-DD Deviation Days, for 120 activities

Iteration=	0	1	2	3	4	5
Trial 1	30	3	0			
Trial 2	83	57	18	0		
Trial 3	35	0				
Trial 4	97	57	30	9	3	0
Trial 5	45	19	3	0		
Trial 6	18	10	8	3	0	
Trial 7	89	59	38	23	0	
Trial 8	20	14	8	3	0	
Trial 9	42	24	12	6	0	
Trial 10	51	18	6	0		

total deviations

0	1	2	3	4	5
14	1	0			
23	31	6	0		
12	0				
26	19	10	3	1	0
20	12	1	0		
7	4	3	1	0	
32	22	15	11	0	
13	11	3	1	0	
16	10	4	2	0	
17	6	2	0		

	makespan delay
no	0
no	0
no	0
yes	3 3 0
no	0
no	0
no	0
no	0
no	0
no	0

Pri-R-DD Solution Time, for 140 activities

Iteration=	0	1	2	3	4	5
Trial 1	50	120	62	38	14	
Trial 2	42	92	61	15		
Trial 3	53	131	100	82	61	14
Trial 4	34	58	43	33	14	13
Trial 5	56	157	60	43	12	
Trial 6	28	55	30	20	10	
Trial 7	48	101	26	13		
Trial 8	34	105	49	36	47	15
Trial 9	52	99	71	57	13	
Trial 10	121	149	17			

Pri-R-DD Deviation Days, for 140 activities

Iteration=	0	1	2	3	4	5
Trial 1	45	12	6	3	0	
Trial 2	45	19	4	0		
Trial 3	117	76	50	32	16	0
Trial 4	57	24	12	9	3	0
Trial 5	124	66	33	3	0	
Trial 6	60	41	28	6	0	
Trial 7	32	22	9	0		
Trial 8	42	17	9	3	2	2
Trial 9	46	27	12	3	0	
Trial 10	58	21	0			

total deviations

0	1	2	3	4	5
20	4	2	1	0	
10	7	2	0		
41	28	32	24	16	0
33	8	4	3	1	0
42	22	11	1	0	
24	17	14	2	0	
10	9	3	0		
18	9	3	1	1	1
20	15	4	1	0	
22	7	0			

	makespan delay
no	0
no	0
no	3 0
no	0
no	3 3 3 0
no	0
yes	0
yes	0
no	0
no	0

Pri-R-DD Solution Time, for 160 activities

Iteration=	0	1	2	3	4	5
Trial 1	92	86	27	23	22	
Trial 2	63	166	20	14		
Trial 3	90	195	113	47	16	
Trial 4	194	46	16			
Trial 5	135	194	154	89	16	
Trial 6	214	447	136	112	23	
Trial 7	211	320	139	34	17	
Trial 8	126	162	175	69	15	
Trial 9	71	146	119	23		
Trial 10	61	152	69	23		

Pri-R-DD Deviation Days, for 160 activities

Iteration=	0	1	2	3	4	5
Trial 1	30	7	4	1	0	
Trial 2	117	26	4	0		
Trial 3	85	38	21	6	0	
Trial 4	70	6	0			
Trial 5	115	38	10	7	0	
Trial 6	85	43	6	1	0	
Trial 7	131	50	31	13	0	
Trial 8	148	79	50	23	0	
Trial 9	27	12	3	0		
Trial 10	15	6	3	0		

total deviations

0	1	2	3	4	5
4	3	2	1	0	
48	13	2	0		
24	11	6	2	0	
22	2	0			
50	22	6	5	0	
49	20	6	1	0	
59	18	11	5	0	
57	34	24	15	0	
9	4	1	0		
5	2	1	0		

	makespan delay
no	0
no	0
no	0
no	0
no	3 0
no	1 0
no	0
no	1 1 1 1 0
no	0
no	0

Pri-R-DD Solution Time, for 180 activities

Iteration=	0	1	2	3	4	5
Trial 1	525	188	181	110	21	
Trial 2	298	434	350	115	28	
Trial 3	246	437	142	38		
Trial 4	341	717	556	257	142	120
Trial 5	429	388	31			
Trial 6	96	203	164	151	108	22
Trial 7	159	133	87	21		
Trial 8	248	206	177	115	20	
Trial 9	87	229	182	162	129	21
Trial 10	352	618	415	209	105	24

Pri-R-DD Deviation Days, for 180 activities

Iteration=	0	1	2	3	4	5
Trial 1	62	13	6	4	0	
Trial 2	113	45	21	3	0	
Trial 3	65	23	6	0		
Trial 4	130	73	28	9	8	3
Trial 5	48	23	0			
Trial 6	45	25	14	6	3	0
Trial 7	90	24	9	0		
Trial 8	94	39	27	6	0	
Trial 9	35	23	11	6	3	0
Trial 10	125	100	58	36	12	0

total deviations

0	1	2	3	4	5
20	6	4	2	0	
48	19	11	1	0	
36	8	2	0		
42	24	10	3	3	1
12	19	0			
20	10	5	2	1	0
35	10	3	0		
29	8	4	2	0	
15	11	7	2	1	0
40	29	19	12	4	0

	makespan delay
no	0
no	0
no	0
yes	0
no	0
no	0
no	0
no	0
no	0
no	0

Pri-R-DD Solution Time, for 200 activities

Iteration=	0	1	2	3	4	5
Trial 1	156	249	49	37		
Trial 2	1493	1720	1575	283	36	
Trial 3	1855	766	512	413	39	
Trial 4	455	428	370	219	125	28
Trial 5	437	310	327	49		
Trial 6	585	839	422	240	230	31
Trial 7	1300	1391	705	376	47	
Trial 8	526	997	527	301	248	36
Trial 9	408	876	693	430	113	22
Trial 10	367	1072	625	369	31	

Pri-R-DD Deviation Days, for 200 activities

Iteration=	0	1	2	3	4	5
Trial 1	18	6	3	0		
Trial 2	230	89	31	6	0	
Trial 3	81	38	20	8	0	
Trial 4	109	62	36	15	3	0
Trial 5	28	8	3	0		
Trial 6	58	36	23	13	3	0
Trial 7	98	34	12	6	0	
Trial 8	78	52	29	15	6	0
Trial 9	153	59	37	21	3	0
Trial 10	108	61	38	14	0	

total deviations

0	1	2	3	4	5
6	2	1	0		
78	21	11	2	0	
25	13	7	3	0	
18	10	7	5	1	0
11	3	1	0		
31	22	9	5	1	0
27	11	4	2	0	
43	19	8	3	2	0
56	18	12	7	1	0
28	26	15	4	0	

	makespan delay
no	0
no	3 0
no	0
no	0
no	0
no	0
no	0
no	0
yes	0
no	0

Below are some calculation results from the source data above, in a format more suitable for graphical analysis:

Activities	Solution Time			Deviation-Days Reduction			Deviation-Days Reduction %			Deviations Reduction			Deviations Reduction %		
	Rnd 1	Rnd 2	Rnd 3	Rnd 1	Rnd 2	Rnd 3	Rnd 1	Rnd 2	Rnd 3	Rnd 1	Rnd 2	Rnd 3	Rnd 1	Rnd 2	Rnd 3
100	10	19	12	30	26	-1	0.44	0.68	-0.08	1	9	3	0.07	0.69	0.75
100	12	29	16	55	24	3	0.65	0.80	0.50	19	8	1	0.66	0.80	0.50
100	8	14	12	14	9	11	0.41	0.45	1.00	8	4	7	0.42	0.36	1.00
100	27	33	30	18	11	6	0.38	0.38	0.33	6	1	4	0.40	0.11	0.50
100	14	24	6	30	30	0	0.50	1.00		14	10		0.58	1.00	
100	12	23	12	27	24	9	0.42	0.63	0.64	4	13	3	0.18	0.72	0.60
100	7	3	6	12	3	0	0.80	1.00		4	1		0.80	1.00	
100	12	20	23	33	6	2	0.73	0.50	0.33	13	2	0	0.76	0.50	0.00
100	18	24	19	33	3	6	0.79	0.33	1.00	2	1	2	0.40	0.33	1.00
100	31	37	57	44	29	8	0.38	0.40	0.18	4	16	6	0.10	0.46	0.32
120	20	20	10	27	3	0	0.90	1.00		13	1		0.93	1.00	
120	36	107	54	26	39	18	0.31	0.68	1.00	-8	25	6	-0.35	0.81	1.00
120	13	12		35	0	0	1.00			12			1.00		
120	43	111	98	40	27	21	0.41	0.47	0.70	7	9	7	0.27	0.47	0.70
120	30	38	17	26	16	3	0.58	0.84	1.00	8	11	1	0.40	0.92	1.00
120	8	21	18	8	2	5	0.44	0.20	0.63	3	1	2	0.43	0.25	0.67
120	36	86	71	30	21	15	0.34	0.36	0.39	10	7	4	0.31	0.32	0.27
120	10	17	12	6	6	5	0.30	0.43	0.63	2	8	2	0.15	0.73	0.67
120	14	30	20	18	12	6	0.43	0.50	0.50	6	6	2	0.38	0.60	0.50
120	23	51	30	33	12	6	0.65	0.67	1.00	11	4	2	0.65	0.67	1.00
140	50	120	62	33	6	3	0.73	0.50	0.50	16	2	1	0.80	0.50	0.50
140	42	92	61	26	15	4	0.58	0.79	1.00	3	5	2	0.30	0.71	1.00
140	53	131	100	41	26	18	0.35	0.34	0.36	13	-4	8	0.32	-0.14	0.25
140	34	58	43	33	12	3	0.58	0.50	0.25	25	4	1	0.76	0.50	0.25
140	56	157	60	58	33	30	0.47	0.50	0.91	20	11	10	0.48	0.50	0.91
140	28	55	30	19	13	22	0.32	0.32	0.79	7	3	12	0.29	0.18	0.86
140	48	101	26	10	13	9	0.31	0.59	1.00	1	6	3	0.10	0.67	1.00
140	34	105	49	25	8	6	0.60	0.47	0.67	9	6	2	0.50	0.67	0.67
140	52	99	71	19	15	9	0.41	0.56	0.75	5	11	3	0.25	0.73	0.75
140	121	149	17	37	21	0	0.64	1.00		15	7		0.68	1.00	
160	92	86	27	23	3	3	0.77	0.43	0.75	1	1	1	0.25	0.33	0.50
160	63	166	20	91	22	4	0.78	0.85	1.00	35	11	2	0.73	0.85	1.00
160	90	195	113	47	17	15	0.55	0.45	0.71	13	5	4	0.54	0.45	0.67
160	194	46	16	64	6	0	0.91	1.00		20	2		0.91	1.00	
160	135	194	154	77	28	3	0.67	0.74	0.30	28	16	1	0.56	0.73	0.17
160	214	447	136	42	37	5	0.49	0.86	0.83	29	14	5	0.59	0.70	0.83
160	211	320	139	81	19	18	0.62	0.38	0.58	41	7	6	0.69	0.39	0.55
160	126	162	175	69	29	27	0.47	0.37	0.54	23	10	9	0.40	0.29	0.38
160	71	146	119	15	9	3	0.56	0.75	1.00	5	3	1	0.56	0.75	1.00
160	61	152	69	9	3	3	0.60	0.50	1.00	3	1	1	0.60	0.50	1.00
180	525	188	181	49	7	2	0.79	0.54	0.33	14	2	2	0.70	0.33	0.50
180	298	434	350	68	24	18	0.60	0.53	0.86	29	8	10	0.60	0.42	0.91
180	246	437	142	42	17	6	0.65	0.74	1.00	28	6	2	0.78	0.75	1.00
180	341	717	556	57	45	19	0.44	0.62	0.68	18	14	7	0.43	0.58	0.70
180	429	388	31	25	23	0	0.52	1.00		-7	19		-0.58	1.00	
180	96	203	164	20	11	8	0.44	0.44	0.57	10	5	3	0.50	0.50	0.60
180	159	133	87	66	15	9	0.73	0.63	1.00	25	7	3	0.71	0.70	1.00
180	248	206	177	55	12	21	0.59	0.31	0.78	21	4	2	0.72	0.50	0.50
180	87	229	182	12	12	5	0.34	0.52	0.45	4	4	5	0.27	0.36	0.71
180	352	618	415	25	42	22	0.20	0.42	0.38	11	10	7	0.28	0.34	0.37
200	156	249	49	12	3	3	0.67	0.50	1.00	4	1	1	0.67	0.50	1.00
200	1493	1720	1575	141	58	25	0.61	0.65	0.81	57	10	9	0.73	0.48	0.82
200	1855	766	512	43	18	12	0.53	0.47	0.60	12	6	4	0.48	0.46	0.57
200	455	428	370	47	26	21	0.43	0.42	0.58	8	3	2	0.44	0.30	0.29
200	437	310	327	20	5	3	0.71	0.63	1.00	8	2	1	0.73	0.67	1.00
200	585	839	422	22	13	10	0.38	0.36	0.43	9	13	4	0.29	0.59	0.44
200	1300	1391	705	64	22	6	0.65	0.65	0.50	16	7	2	0.59	0.64	0.50
200	526	997	527	26	23	14	0.33	0.44	0.48	24	11	5	0.56	0.58	0.63
200	408	876	693	94	22	16	0.61	0.37	0.43	38	6	5	0.68	0.33	0.42
200	367	1072	625	47	23	24	0.44	0.38	0.63	2	11	11	0.07	0.42	0.73

8.3 Appendix C – Gusek Code for Priority-Duration Formulation

```
param H; #Planning Horizon

param R; #Resource Type

param n; #No. of activities

param q; #number of duration modes, related to overtime

param u; #Number of reductions

set A,default{1..n}; # set of activities

set P within A cross A; #predecessors

set RR,default {1..R}; #No. of resources

set M,default {1..q}; #Overtime modes

param D{i in A, m in M}; #Durations

param RD{i in A, k in RR}; #Resources demand for activity i

param AV{k in RR}; #Resource Availability

param ES{i in A}; #0; #Early Start time

param LS{i in A};#H-D[i]; #Late Start time

param PR{i in A}; #priority

param S{i in A}; #last schedule start times

var x{i in A, t in ES[i]..LS[i],m in M}, binary;

maximize objective: sum{m in 1..q, i in 1..n, t in ES[i]..LS[i]} x[i,t,m]*(1-(0.001*t))*(100/(PR[i]^5))*((0.001+D[i,1])^1.1);

s.t. one{i in A} : sum{t in ES[i]..LS[i],m in M} x[i,t,m] = 1;

s.t. two{(i,j) in P} : sum{t in ES[j]..LS[j],m in M} t*x[j,t,m] >= sum{t in ES[i]..LS[i],m in M} x[i,t,m]*(t + D[i,m]);

s.t. three{k in RR,t in 0..H} : sum{i in A,m in M,s in max(t-D[i,m]+1, ES[i])..min(LS[i],t)} RD[i,k]* x[i,s,m] <= AV[k];

s.t. four: sum{i in A,t in ES[i]..LS[i]}(x[i,t,2]+(2*x[i,t,3])+(3*x[i,t,4]))<=u; #No more than u reductions

solve;

display objective,x,PR;
```

```
#this file requires a .dat file with all the right data, in the right format. See next Appendix.

end;
```

8.4 Appendix D – Data File Output Format for Gusek for a 110-Activity Example

This can be copy-pasted from MS Excel into a text file with the same name as the Gusek .mod file:

```
data;

param H:=	241	;

param R:=	20	;

param n:=	110	;

param q:=	4	;

param u:=	1	;

set P:=

(	1	,	2	),

(	1	,	3	),

…

(	109	,	110	) ;


param: ES LS PR:=

1	0	241	1

2	0	241	1

…

110	0	241	1 ;


param D: 1 2 3 4:=

1	1	1	1	1

2	6	5	4	3

…

110	10	9	8	7 ;


param AV:=

1	5


	175
```

2 5

…

20 5 ;

param RD: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 :=

1 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2

…

110 0 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 ;

end;

8.5 Appendix E – Gusek Code for Discrete-Time Priority-Duration RCPSP Model for WO Scheduling with Activity Adjacency

```
param H; #Planning Horizon

param K; #Resource Type

param n; #Number of activities

param w; #Number of WOs

set A,default{1..n}; # set of activities

set W,default{1..w}; #set of WOs

set P within A cross A; #predecessors among activities from different WOs

set RR,default {1..K}; #No. of resources

set AW within A cross W; #work package assignment for activities

param D{i in A}; #Activity durations

param RD{i in A, k in RR}; #Resources demand for activity i

param AV{k in RR}; #Resource Availability

param ES{i in A}; #Early Start time, for operations/activities

param LS{i in A}; #Late Start time, for operations/activities

param ESw{z in W}; #Early Start time, for WOs

param LSw{z in W}; #Late Start time, for WOs

param PR{i in A}; #priority

param f{i in A}; #relative operation/activity time distance from WO start time

var x{i in A, t in ES[i]..LS[i]}, binary; #Activity/operation start times

var y{z in W, t in ESw[z]..LSw[z]}, binary; #WO/parent unit start times

maximize objective: sum{i in 1..n,t in ES[i]..LS[i]} x[i,t]*(1-(0.001*t))*(100/(PR[i]^5))*((0.001+D[i])^1.1); #objective function only cares about activity start times


s.t. one{i in A} : sum{t in ES[i]..LS[i]} x[i,t] = 1;

s.t. two{z in W} : sum{t in ESw[z]..LSw[z]} y[z,t] = 1;

s.t. three{(i,j) in P} : sum{t in ES[j]..LS[j]} t*x[j,t] >= sum{t in ES[i]..LS[i]} x[i,t]*(t + D[i]);
```

```
s.t. four{k in RR,t in 0..H} : sum{i in A,s in max(t-D[i]+1, ES[i])..min(LS[i],t)} RD[i,k]* x[i,s]
<= AV[k];

s.t. five{(i,z) in AW} : sum{t in ESw[z]..LSw[z]} y[z,t]*t = sum{t in ES[i]..LS[i]} x[i,t]*(t - f[i]);

solve;

display objective,x,y,PR;

#this file requires a .dat file with all the right data, in the right format.

end;
```

8.6 Appendix F – Data File Output Format for Gusek for a 110-Activity Example

This can be copy-pasted from MS Excel into a text file with the same name as the Gusek .mod file:

```
data;

param H:=	374	;

param K:=	53	;

param n:=	225	;

param w:=	12	;

set P:=

(	5	,	18	),

(	19	,	35	),

…

(	188	,	225	) ;


set AW:=

(	1	,	1	),

(	2	,	1	),

…

(	225	,	12	);


param: ES LS D PR f:=

1	0	32	6	1	0

2	0	32	8	1	0

…

225	21	93	2	1	21	;


	param: ESw LSw:=


	179
```

1 0 32

2 0 62

…

12 0 72 ;

param AV:=

1 20

2 20

…

53 5 ;

param RD: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 :=

1 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2

…

225 0 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 ;

end;