Performance Robust Project Scheduling Polices for Naval Ship Maintenance

by

John Couch

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

at

Dalhousie University
Halifax, Nova Scotia
September 2016

Dedication

For my family.

Many are the plans in the mind of a man, but it is the purpose of the Lord that will stand. (ESV)

Proverbs 19:21

Table of Contents

List of Tables

List of Figures

Abstract

The research presented in this thesis concentrates on robust project scheduling for resource and time-constrained naval maintenance projects. In the naval maintenance environment, and many other real world cases, project information is subject to considerable uncertainty and a deterministic baseline schedule quickly becomes unachievable. To limit the effect of these unexpected but inevitable schedule disruptions, scheduled resource buffers are used to absorb the changes and protect schedule quality.

A linear programming model is developed and used to evaluate the effectiveness of time and resource buffers in improving schedule stability, and ways these buffers can be implemented in maintenance schedules to provide the best overall schedule adherence. The model incorporates the effects of activity crashing decisions to represent time-quality trade-offs.

Experimental results show that periodic buffers provide better stability than no buffers and that buffers positioned around the longest activities with the most resource demands provide the best schedule adherence.

List of Abbreviations Used

AOP	Annual Operating Plan
AST	Actual Start Time
BST	Baseline Start Time
C_{max}	Total project makespan
CPM	Critical Path Method
EC	Engineering Change
ES	Early Start
EWP	Extended Work Period
FMFCB	Fleet Maintenance Facility Cape Breton
FMFCS	Fleet Maintenance Facility Cape Scott
LS	Late Start
MILP	Mixed-Integer Linear Program
MIP	Mixed-Integer Program
NC	Network Complexity
PERT	Program Evaluation and Review Technique
PSP	Project Scheduling Problem
PSPLIB	Project Scheduling Problem Library
RACP	Resource Availability Cost Problem
RCN	Royal Canadian Navy
RCPSP	Resource Constrained Project Scheduling Problem
RLP	Resource Levelling Problem
SWP	Short Work Period
TOC	Theory of Constraints

Glossary

Crashing: Reducing an activity's duration, usually at an increased resource cost.

Critical path: Sequence of tasks that cannot be delayed without affecting the expect project completion date.

Delay days: Difference in days between the expected baseline start date and the actual start date. Summed across all activities to give schedule quality or schedule robustness.

Dummy: Placeholder activity with zero resource demand and duration. Normally used to represent start and finish milestones.

Makespan: Duration of time required to finish all project activities.

Materiel: Military specific supplies or equipment.

Project managed work periods: Industry term used in fleet maintenance facilities to denote maintenance work periods that require full project teams due to their size and scope.

Robust: Able to withstand disruption.

Schedule quality: The closeness of the actual schedule in comparison to the initial baseline schedule.

Slack: Time an activity can be delayed without affecting the expected project completion date.

Throughput: Number of project activities completed at a given point in time.

Time windows: Time between a given start and finish time that an activity or project will be executed in.

Acknowledgements

Thank you Stephanie. Your strength and sacrifices allowed me to pursue this opportunity. Thank you Daniel, Hannah, Benjamin, Timothy and Simon for all of your patience. Thank you Dad and Mom for your prayers and for your faith in my efforts.

I am appreciative of the time and patience my thesis advisors Drs. Claver Diallo and Alireza Ghasemi allocated to my efforts. Completing this work would not have been possible without their encouragement. I would also like to thank Drs. Uday Venkatadri and Harvi Millar for their work on my examining committee.

Thank you to my Royal Canadian Navy colleagues who were always ready to answer questions and provide support. The Fleet Maintenance Facility team also helpfully explained their work processes and policies.

This thesis was completed with scholarship funding provided by the Industrial Engineering department and Faculty of Engineering at Dalhousie University, and a Gillespie Graduate scholarship.

Chapter 1: Introduction

The research presented in this thesis deals with project scheduling and control challenges inherent in naval maintenance work periods. In Canada, these naval maintenance projects are typically performed during four to eight week time windows. Each project consists of a portfolio of prioritized activities that can vary in duration from the full work period to less than a single day. The resources needed to complete each of these activities are drawn from a shared resource pool. Other ships involved in concurrent projects and maintenance requests use these shared resources as well. Developing good project schedules that perform well in a highly uncertain environment is critical to strong project performance and efficient resource utilization.

This thesis uses a mixed-integer program (MIP) model to apply a robust, or proactive, approach to schedule generation. The model develops schedules that are least sensitive to disruption due to unexpected changes. The performance-robust scheduling method developed in this MIP model represents a new academic contribution to the literature which, to date, is primarily concerned with either time or resource robustness in isolation. Other project scheduling characteristics incorporated into the model include variable resource availability, fixed project time window and multi-mode activity crashing to represent activity quality trade-offs.

To establish the research context, this introductory chapter provides a brief background on the project management industry in Section 1.1. Section 1.2 details the development of the project scheduling problem in academic research over time, Section 1.3 describes the naval maintenance project environment, Sections 1.4 and 1.5 outline the current approach developed to manage these work periods and how project performance is evaluated. Section 1.6 introduces a simplified problem scenario to demonstrate some common scheduling challenges and Section 1.7 ends the opening chapter with the thesis problem statement.

1.1 Project Management Overview

A project can be defined as a "one-time endeavour that consists of a set of activities, whose executions take time, require resources and incur costs or induce cash flows" (Schwindt & Zimmermann, 2015). This definition of a project can be applied quite liberally to most processes in modern business and industry. The key project components of time, resources and performance may be further complicated by additional factors such as due date penalties, precedence constraints and multiple execution modes. The task of effectively coordinating all of these characteristics from start to finish is known as project management.

The Project Management Institute, better known as PMI, is a global not-for-profit membership association of project management professionals. PMI lists the five primary project processes as initiating, planning, executing, monitoring and control, and closing (Project Management Institute, Inc., 2013). To oversee these process groups successfully, a project manager must "perform the project within time and cost estimates at the desired performance level in accordance with the client, while utilizing the required resources effectively and efficiently" (Schwindt & Zimmermann, 2015). Traditionally the responsibility of successfully managing a project is given to an individual project manager who directs the project throughout its lifecycle.

The research presented in this paper focuses on project management efforts occurring during the planning phase of the project management process, the point in the project where initial scheduling will ideally occur (Larson & Gray, 2013). The main output of the planning phase is a preliminary or baseline schedule consisting of an expected start and or finish time for each of the activities required for overall project completion. When the nature of the project requires it, the baseline project schedule may also include other information such as choosing from a set of potential job completion modes and making specific resource assignments.

An initial project schedule is designed to meet the intentions provided by management or an equivalent project sponsor. The objective to finish as soon as possible, thus minimizing the project makespan, is most commonly considered. It is also possible to schedule to achieve a variety of other management goals including lowest cost completion by a fixed deadline, balanced or levelled resource usage over a given time horizon, maximizing net present value (NPV), and reducing the lateness of each individual activity. A number of other potential scheduling objectives are covered in extensive literature surveys provided by Brucker, Drexl, Möhring, Neumann, and Pesch (1999), Hartmann and Briskorn (2010), and Węglarz, Józefowska, Mika, and Waligóra (2011).

An accurate baseline schedule is needed prior to the project start date to assign specific high value or scarce resources - such as heavy cranes, highly specialized technicians, generator test loads or a graving dock in the naval maintenance environment - to specific periods during the project. The impact of poorly allocating scarce or high value resources is significant to overall organizational performance. A slight delay affecting these crucial resources has the potential to disrupt the current project as well as the schedule and budget of other ongoing and subsequent projects.

For projects with large budgets and hundreds of tasks distributed across a wide range of resource types it is easily understood that a project manager is unlikely to find a schedule that distributes project resources in the most effective manner or achieves the best possible resource utilization unless specialized computer software is used. Despite the potential for significant scheduling inefficiencies, many project centred industries continue to generate schedules using inexact methods such as simple heuristic procedures or rules of thumb developed and refined over time by key employees responsible for scheduling.

Resistance to best possible scheduling practices in real world scheduling departments is often based on the premise that an optimal schedule is feasible only in a theoretical sense. The lack of slack or inefficiencies in an optimal schedule make it unable to adjust to any unexpected disruptions that are likely to occur. Project environments with high uncertainty experience frequent scheduling breakdowns when using optimal schedules developed from deterministic activity information assumed to be known in advance. Unexpected conflicts and delays related to unplanned scope updates quickly undermine organizational confidence in theoretically optimal schedules and causing project schedulers to prefer personal knowledge and experience to create workable baseline schedules that may or may not be efficient.

1.2 The Project Scheduling Problem

The value of utilizing an efficient project schedule is well recognized (Project Management Institute, Inc., 2013) and, as a result, project scheduling is well studied in academic literature, covering numerous project variations and industries.

The most well known attempts to apply mathematical modelling to project scheduling problems (PSP) are the critical path method (CPM) (Kelley Jr & Walker, 1959) and the program evaluation and review technique (PERT) (Malcolm, Roseboom, Clark, & Fazar, 1959). Both of these methods were designed to use known precedence relationships and duration information for each task to generate feasible project schedules that solved the PSP. The primary difference between these two early project scheduling methods is that PERT incorporates some known probability information regarding activity durations. The optimal solution provided by CPM is the schedule with the earliest possible project finish time that respects all given precedence constraints. This solution will also identify the activities that make up the *critical path*, a series of activities that will increase the overall project duration if any single one of them is delayed. Activities not on the critical path are considered to have slack or float time buffers that will absorb a delay equal to or less than the activity's float time without extending the expected duration.

CPM determines this solution and critical path information by iteratively evaluating duration times and precedence information about each activity within the project and determining the schedule that allows for the earliest possible project completion time. The mathematical design of this process is outlined in the following paragraphs.

All of the activities or jobs that make up a project are included in a group or set $\boldsymbol{J}$ which will often include two *dummy* milestone activities with zero duration to define the start and finish of the project. Activity duration times for each of the other activities within this set are generally assumed to be deterministic, or known with some relative certainty ahead of project execution. Typically, these activity durations are planned using preliminary estimates and initial surveys of the job scope or using historical data from similar activities previously completed. To model activity duration time, each activity j is given a duration or processing time denoted by parameter d_j.

Precedence information is inferred from the logical sequence of activities such as in manufacturing design where full scale production will not begin before its predecessor prototype testing concludes. Activity predecessors can be modelled as a series of sets where set P_j contains a list of i predecessors for activity j. An activity may have a number of predecessors tightly constraining where it fits in the project schedule or it may have no predecessors and have a large time window where it might be scheduled. Predecessor restrictions can also be viewed as technological constraints.

The project time horizon, the time window where the activities are to be scheduled, is created as a series of discrete time intervals that represent the most relevant unit of time such as hours or days. Activity duration d_j is given as a multiple of these time intervals. The start time for each activity j is represented by the integer variable x_j and then constrained by the provided predecessors P_j. The predecessor constraint is modeled by restricting the start time of an activity *j* to a time greater than or equal to the time when the preceding activity i occurs plus the duration time of activity i. For example, if activity B has a given duration of three days and is in the set of predecessors for activity C, the predecessor constraint will restrict the start time of C to occur after B starts plus the duration of B. If B was given a start time of day 2 ($x_B = 2$), C can begin no sooner than day 5 ($x_C \geq 5$). This predecessor constraint is modelled as

$$x_j \geq x_i + d_i \qquad \forall\ j\ \in\ \boldsymbol{J}, i\ \in\ P_j$$

After applying this constraint (shown as Eq. (1.2) in the model formulation below), the PSP can be solved for the given activity durations and predecessors to provide the earliest start time for each activity and the shortest overall project duration. The shortest PSP makespan objective, Eq. (1.1), will give the dummy finish milestone x_J the soonest completion time according to the following formulation:

$$minimize\ x_J \tag{1.1}$$

$$subject\ to$$

$$x_j \geq x_i + d_i \qquad \forall\ j\ \in\ \boldsymbol{J}, i\ \in\ P_j \tag{1.2}$$

Precedence and duration information are passed to a linear programming solver that uses this model to determine the optimal solution within the given constraint. When the best solution is found, the solver terminates and provides the earliest possible start time for all activities, giving the shortest overall project duration presuming all resource requirements can be met.

A limitation of the methods that solve the PSP in this manner, such as CPM and PERT, is that the schedules generated are only precedence and time feasible. The schedule may not be feasible once resource constraints are considered. This is an area of concern for large projects sharing a common pool of resources. Consequently, initial resource constrained project schedules created using CPM or PERT require additional analysis to consider the effects of resource constraints on the proposed schedule.

Projects are unique undertakings by definition and as such, there remains many ongoing opportunities to modify standard mathematical formulations of the project scheduling problem to match the numerous variations found in real world projects. Project properties, objectives and constraints vary by industry, type and application. These variations provide a large number of potential modelling approaches.

Theoretical research into project management and, more specifically, project scheduling is thoroughly studied in academic literature. Numerous exact and heuristic solutions are suggested to improve scheduling efficiency. The challenge in applying many of these models to the naval maintenance project environment and other complex project industries arises from the nondeterministic nature of project information found in these settings. It is a comparatively trivial problem to determine the best project schedule when all task and duration information is known in advance and is not affected by uncertainty. When project characteristics are expanded to include uncertain durations, variable resource availabilities and changing activities, the task of determining the best resource allocation becomes far more complex.

1.3 Naval Maintenance Environment

Canadian naval maintenance facilities are tasked with “providing reliable and solid engineering and maintenance direction, consultation and technical risk management to help ensure that [naval forces] can generate, employ and maintain an effective fleet” (Royal Canadian Navy, 2015). The Royal Canadian Naval fleet is currently in a period between retiring aging ships and strategizing how to replace them with the former happening sooner than expected (Gilmore, 2015) and the latter happening much later than would be ideal (Hansen, 2014). The pressure of retiring ships well before replacements are built places a premium on the resources invested in each of the existing naval vessels.

In addition to working on an older fleet, the effectiveness of ship maintenance and repair projects are further reduced by lower operations and maintenance budgets. These two factors have helped push maintenance demand up to the current capacity of existing repair facilities. To counter these effects and improve maintenance efficiencies, the Department of National Defence has outlined a key maintenance and materiel renewal goal to “[improve] the ‘demand’ associated with maintenance programs [by] rationalizing and better aligning maintenance requirements and schedules with operational requirements and priorities” (Department of National Defence, 2013). Improving maintenance schedules is expected to improve efficiency and increase the availability of the naval fleet.

Several other internal Canadian Forces (CF) and Royal Canadian Navy (RCN) documents further highlight the need to improve maintenance process to deliver better performance by better prioritizing and planning work projects to increase wrench time, the actual time spent by a technician completing maintenance work.

The RCN currently utilizes two naval dockyards to execute necessary maintenance during project managed work periods. On the west coast of Canada, 15 naval ships and submarines are repaired at Fleet Maintenance Facility Cape Breton (FMFCB) in Esquimalt, British Columbia and on Canada’s east coast, 18 naval ships and submarines are repaired in Halifax, Nova Scotia at Fleet Maintenance Facility Cape Scott (FMFCS) (Government of Canada, 2015). Each dockyard is responsible for the maintenance demand of the naval ships and submarines assigned to its region.

1.3.1 Uncertain Project Information

A majority of the academic literature reviewing project scheduling and management considers a strictly deterministic environment where the project portfolio, durations and resource costs are known in advance and not subject to uncertainty. Scheduling and executing a deterministic project schedule in an uncertain environment will result in frequent schedule breakdowns where an activity cannot start or finish as expected, resource conflicts due to rescheduled activities and a project portfolio that can no longer accommodate all of the initially planned activities.

To demonstrate the lack of flexibility in an unbuffered deterministic schedule a popularly cited scheduling example from Wiest and Levy is shown in Figure 1 (1977). This project has eight activities plus activity 1 the dummy start and activity 10 the dummy finish, both of which have zero duration and zero resource requirements. The top number next to each activity node is the activity duration and the bottom number is the number of resources required over this period. For simplicity, each of the activities require only one resource. This singular resource has ten units available for use during each period.

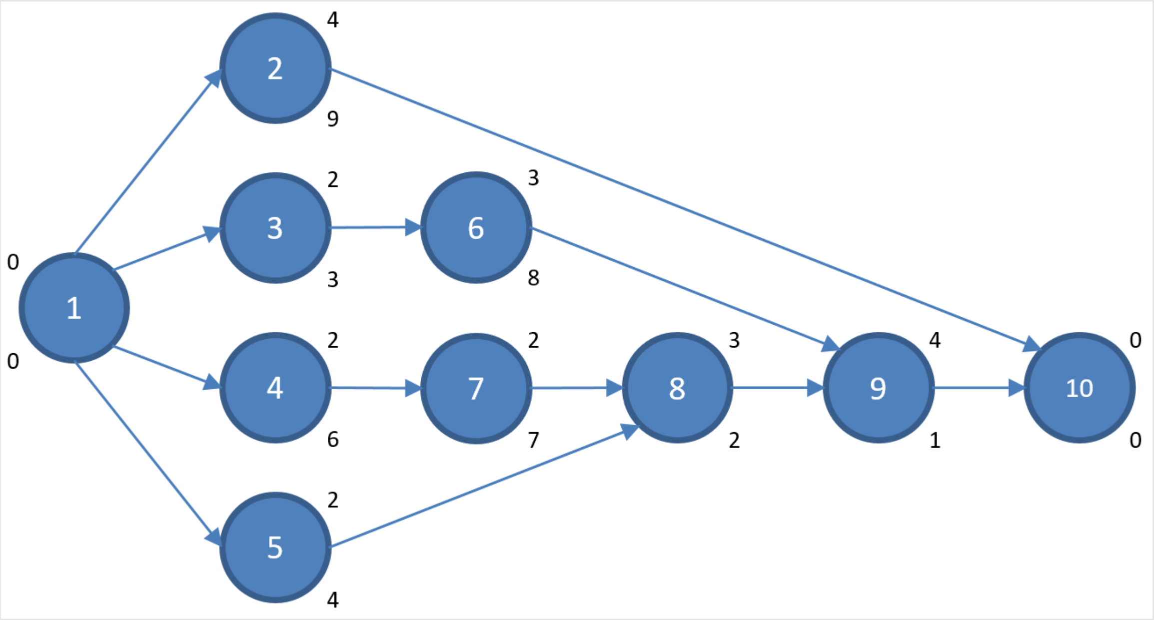

Figure 1: Activity on node project example

The precedence relationships shown in this network may be summarized by following each possible path, or chain, in sequence from 1 through 10. Following this set notation for each of the four potential paths gives the following relationships: <1,2,10>, <1,3,6,9,10>, <1,4,7,8,9,10> and <1,5,8,9,10>. The first path restricts activity 10 from starting until activities 1 and 2 are complete. The remaining three paths show activity 9 is also an immediate predecessor of activity 10.

The optimal schedule for this network for a deterministic scenario with an objective to minimize the project duration is given in Figure 2. Coincidently, in this schedule, all of the available resources are used in a perfectly level manner and all of the activities are on a critical path with no slack. This type of deterministic approach, where activity data is assumed known, fails to account for significant uncertainty present in the naval maintenance environment. Section 1.6 provides an amplifying example to demonstrate how an inflexible schedule does not perform well when both the portfolio of activities and the activity information are subject to change during project execution.

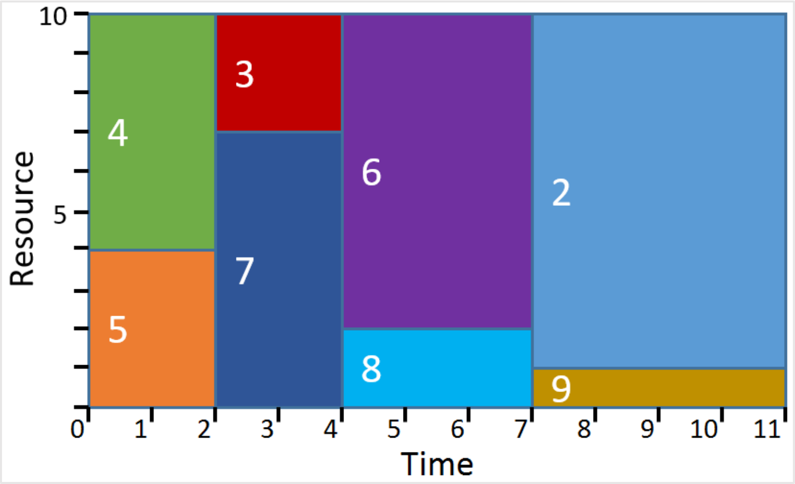

Figure 2: Optimal but inflexible resource profile

1.3.2 Imposed Time Window

Determining a project time window or deadline using a deterministic approach will typically involve some form of the critical path method (CPM) where activities are started as soon as possible according to activity sequencing, precedence or time lag constraints. The finish date of the last activity is then quoted as the desired project deadline. Finishing as soon as it is feasible to do so offers many benefits to both the project sponsor or customer and the service provider. The customer is able to access their asset sooner and begin to recognize a return on their investment in the project while the service provider is freed up to accept new work.

In a naval maintenance environment, the project time window is predetermined by operational timings that consider outstanding maintenance activities only on a general level. Finishing early does not provide a recognizable benefit for the ship or the repair facility in this scenario. Making the ship available sooner will not affect the date of its next operational commitment. In this case, the ship's command team would prefer to have additional backlogged maintenance tasks completed rather than an early work period completion. The service provider also receives limited benefits to finishing in the shortest time possible if the scope of work performed is unchanged since dockyard equipment and labour resources are fixed assets pooled amongst all ongoing projects.

Lean and efficient maintenance organizations are able to provide high resource utilization rates; however, they have limited capability to respond to unexpected and highly technical repair demand. Although it is not as common in most other project-centred industries, timely responsiveness to unplanned demand remains an essential attribute of a naval maintenance facility. The trade-off to providing a high level of service for unexpected demand is that other previously scheduled projects are delayed if available resources are already allocated to capacity. Short response times to unexpected maintenance demand also creates additional trade-off challenges between having resources available for quick response and scheduling high resource utilization. Quick maintenance response requires available resources while high utilization requires all resources to be scheduled as much as possible.

1.4 Project Managed Work Periods

Each naval dockyard completes the main volume of maintenance demand through *project managed work periods* that provide a better opportunity to improve efficiency and planning effectiveness than if maintenance was performed outside of dedicated maintenance time windows. These work periods include both Short Work Periods (SWP) typically of three to six weeks in duration and extended work periods (EWP) that may exceed six months.

Typically, each ship requires two to four SWPs in each fiscal year and one EWP every other year. The ship's operational schedule effectively determines the frequency and timing of these work periods although significant corrective maintenance demand can also affect the work period time windows if a major equipment failure occurred. A project managed work period involves a unique set of jobs and resource requirements that must be completed by a fixed deadline. This deadline is typically adjacent to operational commitments and any maintenance delays have the potential to interfere with scheduled deployments. The ship's operational requirements will generally warrant that all options be considered prior to extending the work period.

Both Fleet Maintenance Facility Cape Breton (FMFCB) and Cape Scott (FMFCS) follow a standardized project management process to improve efficiency and provide a high level of service quality during work periods. Relevant portions of this process are described in the following paragraphs (2013).

1.4.1 Initiation and Planning

The planning phase for a short work period (SWP) begins sixteen weeks prior (T minus 16 or T-16) to the work period start date (T). The first set of activities that are considered during the planning phase are ones requiring engineering changes (EC). ECs are modifications to a ship that involve a higher degree of planning, procurement and documentation than regular maintenance tasks. The larger scope associated with ECs causes their duration and resource requirements to have a higher degree of uncertainty when compared to commonly repeated maintenance work. At T-12 weeks, the project management team (PMT) begins meeting formally to discuss the operational goals of the approaching work period. EC planning is normally finished 12 weeks before project start and EC material procurement begins.

At T-8 weeks, the ship's engineering team will provide a prioritized list of jobs they would like to have completed during the work period. This list of requested work is added to the approved ECs and other major work the maintenance facility has already placed in queue of prospective maintenance tasks. Subject matter experts from the Fleet Technical Authority (FTA) and the naval engineering operations department (N-37) then categorize this list of jobs into one of the following priority levels: essential, high opportunity and normal opportunity. These priority levels are used during project portfolio creation and help project leaders deconflict resources when a higher priority activity arrives unexpectedly.

Any new priority work request submitted in week T-7 or later is marked as a *Late Request* and may require trading priorities with another job already in the draft work package to make room for it. The draft work package and schedule is also reviewed by work centre managers during this week.

Risk and mitigation planning occurs at the project buy-in meeting during week T-6. All essential and high opportunity work is planned by T-5 and the project then transitions into the scheduling phase. Due to the inherent uncertainty associated with the project environment, it is possible, and perhaps likely, that additional jobs will require planning throughout the project cycle.

1.4.1.1 Scheduling

The scheduling phase begins in week T-7 when potential start dates are considered based on material and labour resource availabilities for high priority jobs. Jobs continue to be added into this draft schedule on an ongoing basis once their job instructions, duration and resource requirements are planned. At T-3 weeks, the ship's command team will receive a copy of the draft project plan for review.

Near-term work requirements and the project schedule are covered as a part of the internal project kick-off meeting during week T-2. Work centre managers are given the project plan and told of any additional requirements that may affect the schedule, such as timings when access to the ship may be limited. Finally, a project kick-off meeting is held one week before the project start date. At this meeting, the finalized work package will be given to the ship's command team, a communications plan will be established and relevant shipboard activities planned concurrently with the work period will be discussed.

To prepare for the inherent levels of uncertainty related to activity information, the maintenance schedulers hold a resource capacity buffer until one or two weeks prior to activity start times before allocating them in the project schedule. This short lead time in resource allocation offers a limited degree of protection for the schedule, should delays occur. Unfortunately, it also has the undesirable effect of giving the impression that significant resource capacity will exist in three weeks' time. Resources expected to be available in the third week are allocated to queued backlogged activities in the following week and expedited high priority tasks routinely demand all remaining resource capacity.

These scheduling policies appear to provide some mid-term flexibility in project execution but result in a continual state of over-allocating resources beyond capacity in response to unpredicted project complications. Frequently scheduling resources up to or even above their maximum capacity level gives the appearance that resources will be used efficiently. The actual effect is reduced project scope and lower utilization when activities are subsequently removed from projects due to the lack of resource availability.

1.4.2 Control

Execution of the project involves the allocated resources completing the assigned project tasks. This phase will typically begin at the work period start date (T), but work can begin early in cases where the ship is available beforehand and resource capacity exists. Work centre resources have a general timeframe within the work period to complete these jobs but are given a moderate level of autonomy to complete the assigned tasks within the assigned time window according to their own schedule preference as long as overall management priorities between competing projects are observed. Project leaders for competing projects address any resource conflicts between the ongoing project managed work periods at weekly meetings throughout the project lifecycle. In the case where a planned prioritized job cannot be completed due to a resource shortage, management is consulted. If no feasible solution for completing the task during the work period can be reached, the ship's command team will be advised that a lower priority activity must be cancelled or deferred to the next scheduled work period to create the needed resource capacity.

1.4.3 Closing

At the end of the work period, a close out *Wash-up* meeting is held with the project team members to review project performance. The expected and actual labour hours are presented as are the expected and actual number of completed jobs categorized by priority level. Labour hours and job completion status are also broken down and displayed by week to display the project results as they progressed. Other work period information covered at this meeting includes growth work added to the project after the T-7 cut-off point and a list of all incomplete jobs broken down by priority level.

1.5 Project Evaluation

Project planners, schedulers and managers working in this naval maintenance environment face significant challenges in their attempts to manage the completion of all tasks assigned to their short work period. The project team requires a high level of flexibility to respond to the uncertainty prevalent in both the planning and execution phase of a short work period. Scope changes and additional demand for maintenance or repair activities will inevitably be requested after the project portfolio has already been determined. If no spare capacity exists to absorb these changes, the baseline schedule must be altered to accommodate them.

Evaluating Canadian naval maintenance projects requires the consideration of several unique characteristics that differentiate them from traditional projects found in other similar industries. The timeline for project managed work periods set aside for naval maintenance and repair are based around operational schedules known much farther in advance than expected maintenance demand. Under this procedure, the length of the work period may be adjusted between four to eight weeks depending on a general perception on how many major systems might require extensive maintenance work and how much time the ship is available for maintenance work. The complete portfolio of maintenance activities can only be created after this time window is determined and with only a preliminary knowledge of potential resource availabilities.

Prescribing the project time window prior to creating the project portfolio leads to challenges with resource allocation as well. The naval maintenance facility will have, in most cases, sufficient internal resources available to complete multiple maintenance projects at a given time, but the addition of variable workforce availability and sharing resources across multiple ongoing projects complicates efficient resource scheduling.

Uncertainty is another factor prevalent within naval maintenance projects. Activity planners are aware that activities may become much more complex and require more time or resources once the job begins. To help mitigate schedule disruptions that might result if an activity does grow in scope, a time buffer is often included as a part of the estimated duration. When this type of duration buffering occurs for each activity in the project portfolio, a significant overestimation of the overall project duration occurs. A chart highlighting the significance of scheduling uncertainty relating to activity duration estimations in a sample work period is shown in Figure 3. For the project shown in this figure, activities planned as a part of the initial portfolio are frequently overestimated and finish much sooner than the schedule would expect.

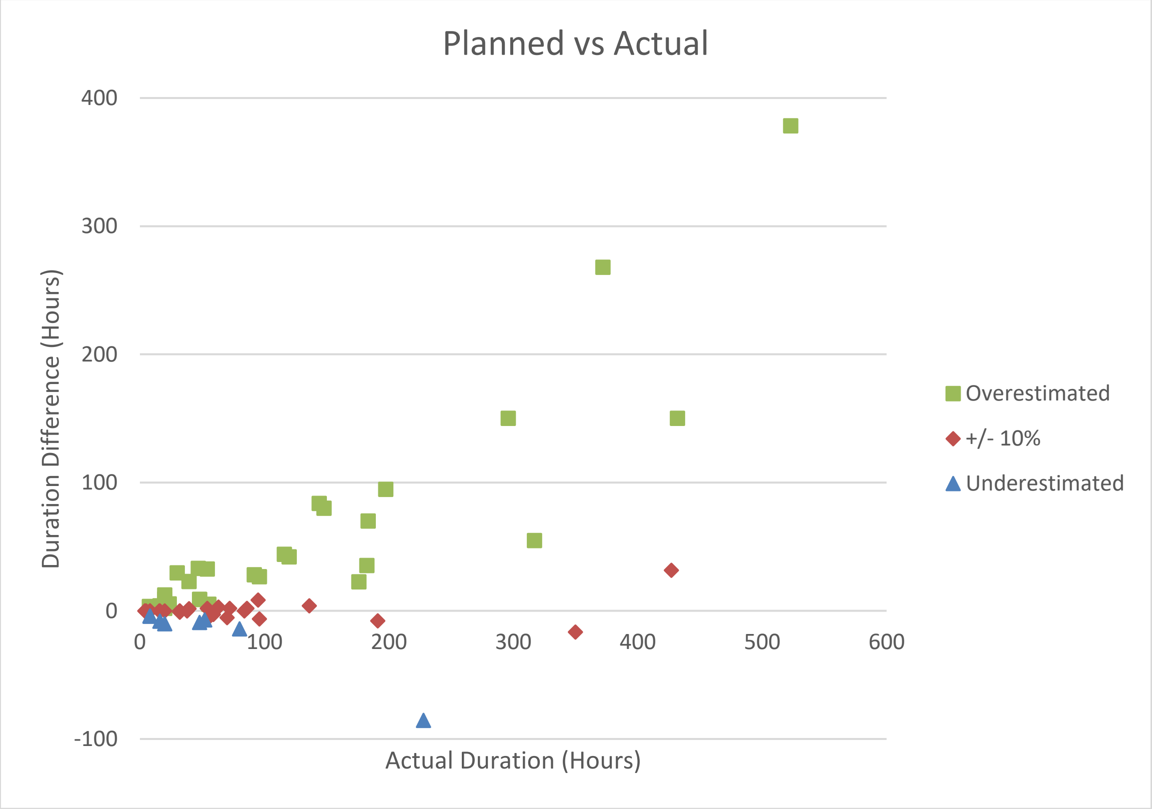

Figure 3: Planned vs actual activity durations in a sample work period

Overestimating activity duration during the planning phase obscures the buffer locations from the scheduling team, leaving them unaware which durations are buffered and which ones are not. When the schedule is built, the scheduling team does not know where the buffers are placed and it is less clear to the project leaders where potential slack may exist. The distance from the y-axis in Figure 3 represents the buffer size in hours that would be better inserted during the scheduling phase around activities with high uncertainty or to protect activity start times for high demand or critical activities.

A level of uncertainty also exists in factors that increase resource demand such as which activities are completed as a part of the final project work package. Exposing technologically complex electrical and mechanical systems on aging ship platforms to a harsh marine environment invariably leads to unexpected failures that may not be discovered until thorough inspections are performed during the work period. An analysis of historical work packages shows that unexpected work can increase the labour hours required during a work period from 10% to 50% of the initially planned hours. New maintenance requests and additional activities arising from preliminary surveys may add many new activities to the project portfolio. Table 1 provides a typical sample of work package variability where a portfolio is increased by 13 essential and high opportunity activities after the baseline schedule is already set.

Table 1: Growth in work package activities and hours

	Baseline (Confirmed Work)	**New & Discretionary**	**Additional Hours Accepted**
Essentials	23	4	88
ECs	9	4	0
High Opp	61	9	383
Normal Opp	39	54	32
Total	**132 activities**	**71 activities**	**503 hours**

Naval maintenance projects are further complicated by evolving geopolitical pressures that can cause even the very best schedules to become instantly irrelevant. These factors can cause situations where a low priority work period will unexpectedly be upgraded to a much higher priority in response to unplanned operational demands. A sudden shift in priorities requires a rapid reallocation of facility resources, often at the expense of efficiency.

Evaluating the effects of uncertainty on naval maintenance output has recently become more practical using a newly adopted resource management system. The availability of this new data in an assessable format makes it possible to form a clearer picture of overall maintenance facility performance and to identify areas for improvement. Aggregating maintenance data into high level organizational performance indicators and comparing them to industry benchmarks draws attention to three potentially problematic areas: project performance, resource utilization and schedule adherence.

1.5.1 Resource Utilization

Resource utilization measures the percentage of resources used compared to the actual resource capacity. If a schedule is perfectly executed, resource usage will equal the percentage of available resources scheduled. If a project changes and the initial resource allocation becomes infeasible, resource utilization may end up much lower than what was expected based on the initial resource allocation.

Resource utilization is important for renewable resources that are not consumed after use and are available for reassignment once their current activity is finished. A pool of hourly workers would be a renewable resource while their related overtime budget for a given project is a non-renewable resource. Consistently providing all available renewable resources to project activities would theoretically lead to high resource utilization, but this only occurs if the baseline schedule remains feasible throughout the project. Failing to assign renewable resources to activities, especially when backlogged maintenance demand exists, is viewed as an inefficient use of organizational resources. The project manager’s objective to manage a project to completion cannot be met without carefully scheduled resource usage.

The increased efficiency associated with high resource allocation results in a corresponding trade-off with resource flexibility. A policy of maximum resource allocation will create a schedule that is very sensitive to uncertainty and can lead to lower actual utilization numbers when project activities are delayed or require resources already assigned to other tasks. In this fully allocated scenario, no spare capacity is available to accommodate a change in project scope. In a tightly allocated scenario, new resource capacity requires accepting additional overtime or subcontracting costs, or cancelling or delaying conflicting tasks.

A monthly sample of resource group utilization rates displayed in Figure 4 confirms, even with the current strategy of high or full resource allocation, equivalent utilization levels are not realizable in project environments with high uncertainty.

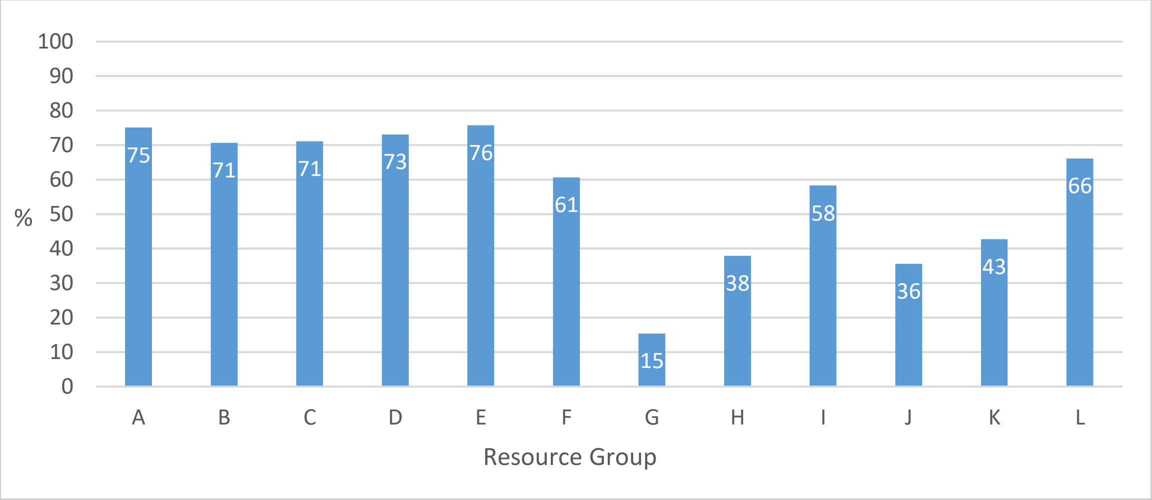

Figure 4: Monthly resource utilization percentages by resource group

A schedule that is expected to have good resource utilization based on scheduled allocations can quickly change to low resource usage if it cannot be followed due to unexpected changes or disruptions. For this reason, efficient resource utilization closely follows the schedule adherence metric. A work period that is unable to follow the original baseline schedule is likely to contribute to low overall resource utilization as well.

1.5.2 Schedule Adherence

Schedule adherence reflects how well the actual start and finish of each activity compares to its scheduled start and finish times. If the original baseline schedule is presumed to be the best possible way to complete the work period, the level of schedule adherence will indicate how close to optimal the project is executed. High schedule adherence, where the actual schedule matches the planned schedule quite closely, is likely to require less unexpected resources and is more likely to finish all activities on time than a project with low schedule adherence.

Activities on the project's critical path which fail to start or finish as initially planned will delay the project due date and create unplanned resource conflicts by postponing resource requirements into a period where the required capacity may not be available. In the case where multiple resource types are required for an individual task, it is also possible for a delayed activity or resource conflict to affect other activity paths due to a common resource requirement.

Schedule adherence is also referred to as *schedule robustness*, the term used in literature to quantify the accuracy of an initial baseline schedule compared to the schedule that is executed.

A schedule is considered robust if it has time or resource capacity held in reserve to account for unexpected events. Robustness can measure two scheduling characteristics, stability and quality. A schedule's stability is measured by *solution robustness*, the difference between planned and actual start times for each individual activity. *Quality robustness* measures the difference between the planned and actual objective values (Herroelen & Leus, 2004)

Evaluating solution robustness by starts per week for a sample naval maintenance work period reveals that only 48 of the 69 scheduled activities began during the week they were scheduled to start. Of the 21 delayed activities, only 2 were successfully started. The remaining 19 activities were cancelled or deferred to later work periods. These results are shown in Figure 5, which displays a weekly overview of the number of scheduled and actual activity starts in the first two columns, and the project leader's count of delayed or cancelled activities that do not start when expected in the last four columns. Weeks three through five specifically highlight resource conflicts as limitations on the potential to reschedule activities. With the resource pool already at its allocated capacity, it becomes increasingly difficult to reschedule activities in response to schedule breakdowns.

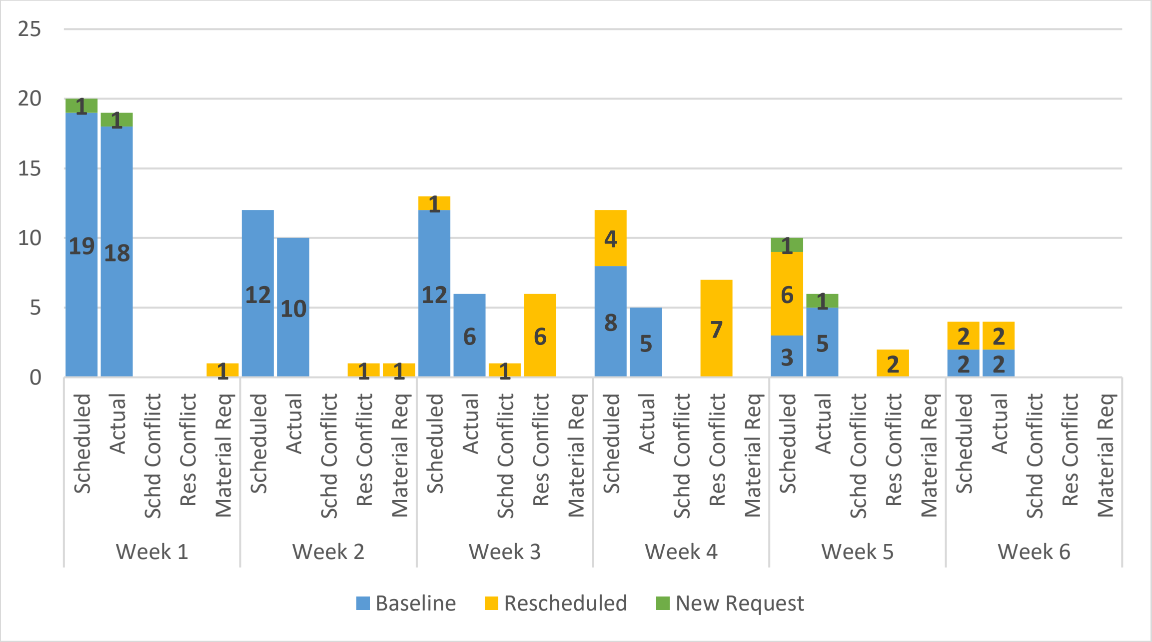

Figure 5: Weekly schedule starts with causes of delay

Cross-referencing this information with feedback from the labour resources initially assigned to the task leads to similar conclusions. Figure 6 provides a Pareto chart breakdown of known causes of incomplete activities during this sampled project. Better use of the comments field would provide a more thorough picture of these causes, but it is clear that unexpected conflicts do lead to reduced project performance. From the subset of assigned causes, schedule and resource conflicts account for 42% of the incomplete jobs.

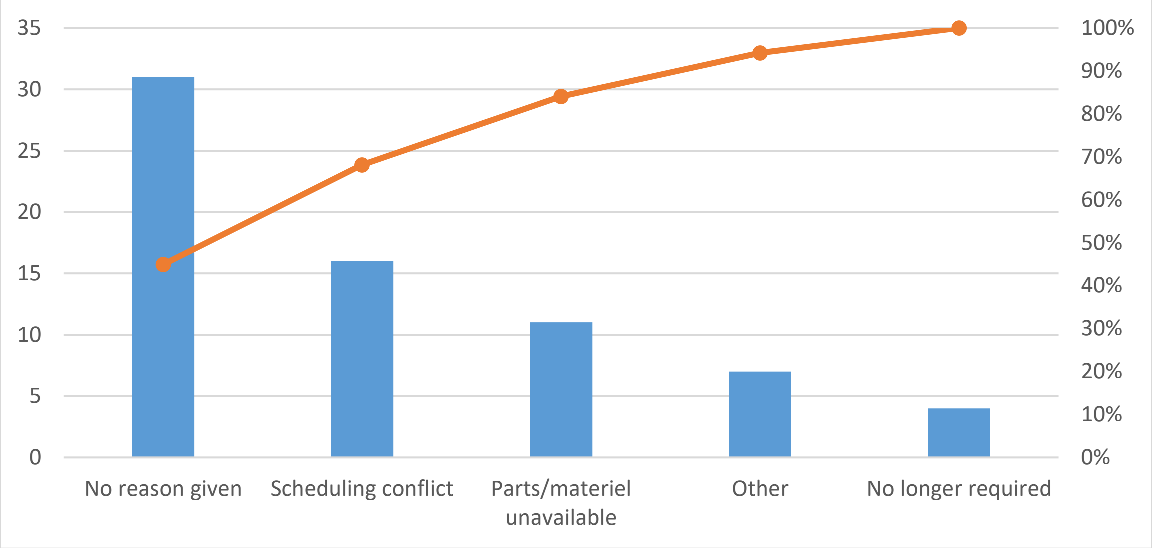

Figure 6: Causes of incomplete activities[1]

1.5.3 Project Performance

Responses during initial stakeholder interviews also suggested that lower than expected job completion rates led to lower overall project performance. A sample project close out summary shown in Table 2 outlines the disparity between confirmed, or planned to be completed, and actually completed activities. It is noteworthy that a number of high priority activities, those designated essential and high opportunity, fail to be completed during the project time window.

---

[1] Two activities have comments stating they are incomplete due to a scheduling conflict and due to material unavailability. They are counted twice to account for both causes.

Table 2: Scheduled and actually completed activities

	Baseline (Confirmed Work)		**New & Discretionary Work**		**Incomplete Activities**
	Activities	**Activities Completed**	**Activities**	**Activities Completed**	
Essentials	23	21 (91%)	4	2 (50%)	4
ECs	9	4 (44%)	4	3 (75%)	6
High Opp	61	41(67%)	9	8 (89%)	21
Normal Opp	39	25 (64%)	54	53 (98%)	15
Total	**132**	**91 (69%)**	**71**	**66 (93%)**	**46**

Each of the planned but not completed tasks represents a potential decrease in operational capability for the ship when it leaves the maintenance period and results in increased risk. When the level of risk is deemed acceptable, trade-offs must be made to accept reduced project performance due to incomplete activities. Examples of trade-off decisions include accepting decreased redundancies in parallel or backup systems, executing a lower quality or temporary repair, and operating without any functionality beyond what is required for the current mission. Each of these examples limit command flexibility to change operational assignments without first performing additional maintenance and repair.

Resource utilization, schedule adherence and project performance outcomes have clear connections to one another. Good resource utilization rates require carefully planned schedules that can reliably be met. Developing management policies that allow work period schedules to be more robust, and therefore less sensitive to uncertainty, would reduce the frequency of activities being dropped from the project schedule. Improved activity completion rates would then provide ships with greater certainty regarding which capabilities they will have when the work period is finished.

1.6 Simplified Problem Scenario

As discussed, efficiently managing naval maintenance projects is a challenging task due to the inherent complexity and uncertainty involved. Even in the event that a very efficient schedule can be determined, unexpected delays and resource conflicts are likely to severely disrupt such a schedule and eliminate any expected efficiency improvements.

For a scenario subject to a high level of uncertainty, the makespan and resource allocation schedule given previously in Figure 2 will become infeasible as soon as either of these parameters increase. If any activity takes longer than expected or requires additional resources, the project finish date will be postponed. There is also no opportunity to add unplanned activities to the project if a new task is requested.

Uncertainty during project execution is common in many industries where baseline schedules can be disrupted for a variety of reasons: differences in how long activities are expected to take, what resources are available, changes to which activities are in the project portfolio and equipment or materiel delays (Herroelen & Leus, 2004).

Supposing the network from Figure 1 in Section 1.3.1 is executed in an uncertain environment and the project is altered by the following four small, but unexpected, disruptions that are not known until the activity is expected to begin:

a) activity 3 finishes one time period sooner
b) activity 5 requires one additional resource
c) activity 6 finishes one time period sooner
d) activity 7 requires one additional time period

The impact of these four disruptions on the initial inflexible schedule from Figure 2 is highlighted in Figure 7 on the following page.

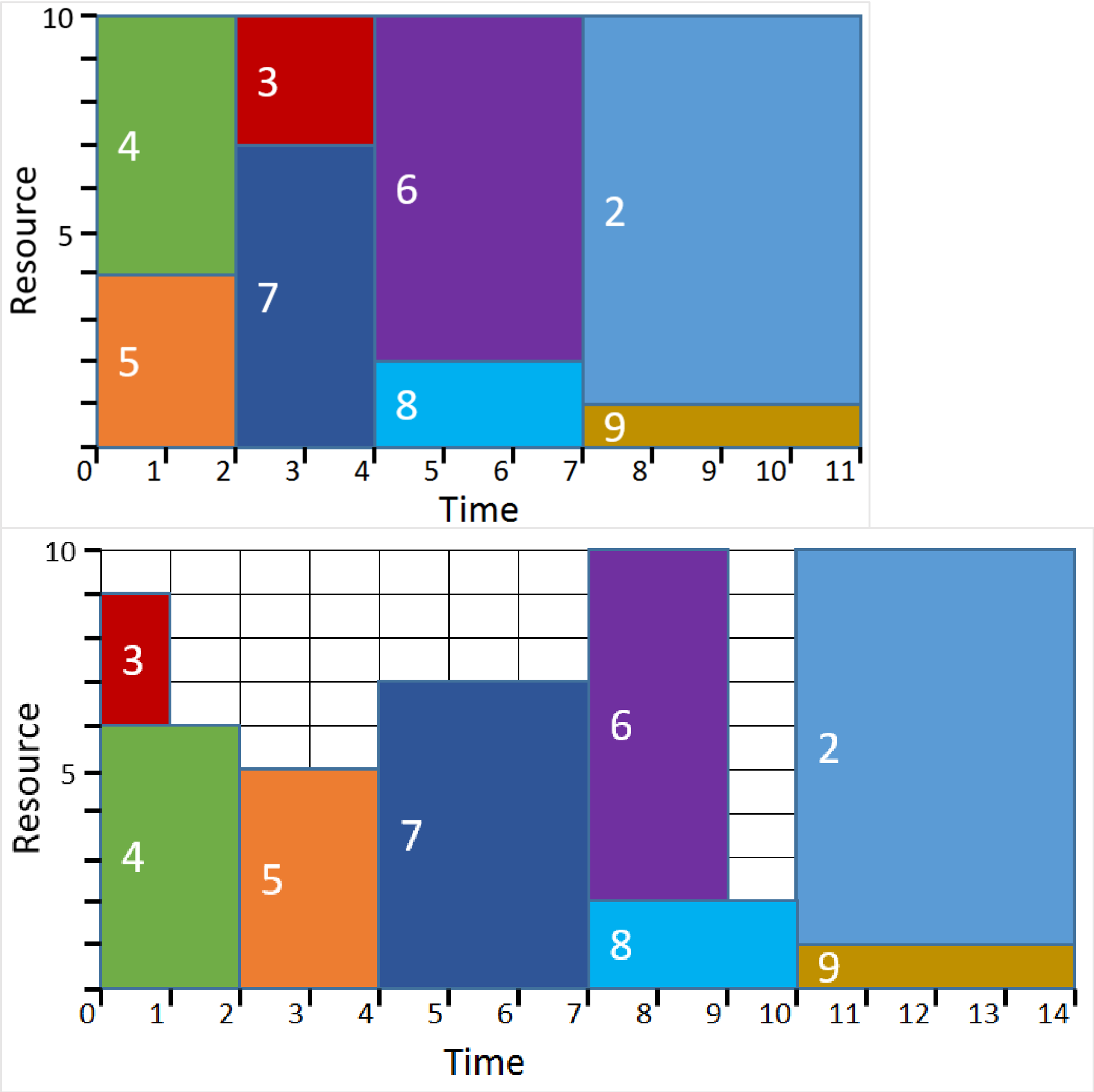

Figure 7: Effect of uncertainty on the inflexible schedule given in Figure 2

The first immediately observable effect of these activity disruptions is an increased project makespan. Rather than finishing at period 11, the finish date is now delayed by almost thirty percent to period 14. With strict deadlines in place, this type of delay would be unacceptable to the customer. A straightforward solution would be to buffer the inflexible schedule from Figure 2 with three periods and quote a project delivery time at time 14 as shown in Figure 8. This simplistic approach would protect the project deadline but does not protect starting times for individual activities.

Measuring solution robustness by the sum of individual start time variation between the schedules from Figure 7 and Figure 8 gives an overall deviation of 18 time units with only activity four starting when originally planned. It would be possible to delay the start of activity three to its baseline start time to improve these metrics but left-shifting activities to their earliest possible start time is a more realistic approach using a makespan minimization objective.

Limiting start time variance is important in a naval maintenance environment where shared resources are scheduled up to and beyond expected capacity, and opportunities to reschedule activities are limited. In the more likely case where inflexible activity time windows further restrict rescheduling options, schedule adherence becomes even more important to overall performance. Well-placed time buffers protect the baseline schedule from unexpected events and reduce the impact of disruptions on activity start times.

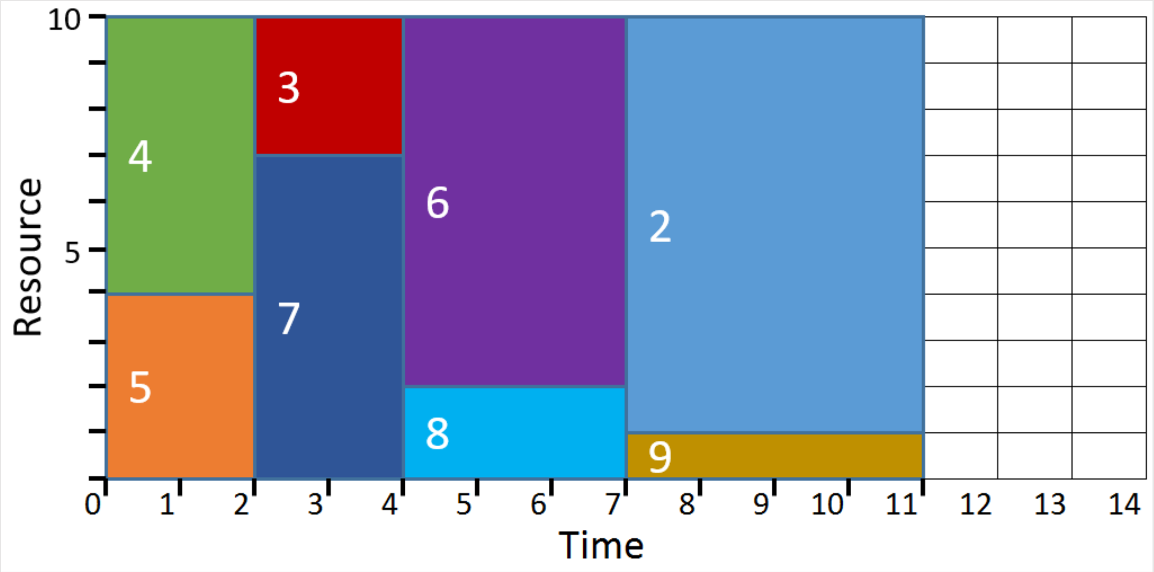

Figure 8: Simplistic makespan buffering approach

Another limitation of a simple deadline buffering approach is the variation in available resource capacity. With no capacity available in the first eleven periods, it is not possible to add any activities to the early part of the schedule without disrupting the planned activities. The 100% resource capacity in the final three periods will also be difficult to utilize. Small per period resource availabilities can be given low priority backlogged work to improve utilization numbers. This backlogged work is unlikely to require a large percentage resource capacity in a single period making large buffers more wasteful if the schedule does occur as planned.

Difficult activity trade-off decisions are also more common with tightly scheduled activities like those shown in Figure 8. In a multi-project environment, unexpected changes between competing projects may reduce the expected resource availability levels for a given project. If the resource profile of the example project is reduced to nine, all of the activities are subject to either a delay or cancellation. Simply by chance, the more dispersed schedule given in Figure 7 has resource buffers protecting the first four activities and would only need to consider half of the activities for delay or cancellation.

Based on these observations, we suggest that strategically located time and resource buffers will outperform a simplified makespan buffering approach and will improve overall project performance in high uncertainty project environments. Time and resource buffers can be used to protect both the activity start times and the project deadline, manage resource level variations and limit undesirable activity trade-off scenarios.

1.7 Problem Statement

The research presented in this thesis concentrates on developing robust schedules that perform well in uncertain project environments. Managers are expected to allocate a limited number of required resources within a fixed time window using these schedules. Typical characteristics of project managed work periods are mathematically modeled and optimal project schedules are generated using linear programming (LP). The effects of project uncertainty on activity scheduling are mitigated by concurrently scheduling resource buffers during the activity scheduling process. Various buffering approaches are considered to evaluate potential scheduling policies and determine which strategy performs best to improve the project's activity completion rate. The suggested scheduling policies are tested using test sets with characteristics similar to real world data to validate their expected effectiveness.

Incorporating robustness or flexibility into a project schedule runs counter to the lean idea of being as efficient as possible. Proponents of robust scheduling approaches would argue, in environments with high uncertainty, a highly lean or efficient schedule is only optimal in its ability to be executed in the real world. This thesis presents scenarios where it is possible to, at least to some extent, offset the cost of adding time and resource robustness to the project schedule through improved scheduling accuracy, realized resource utilization and increased activity performance.

To address this problem, the remainder of this thesis is organized as follows: approaches to project scheduling challenges are discussed in Chapter 2, which also covers a contextual review of the literature. Chapter 3 details the methodology observed during the research process, Chapter 4 presents a discussion of experimental results and Chapter 5 closes with some concluding remarks.

Chapter 2: Literature Review

Academic research into project scheduling primarily revolves around the resource constrained project scheduling problem, or RCPSP, which is clearly the most prevalent project scheduling problem in literature. Comprehensive surveys of this problem such as (Węglarz, Józefowska, Mika, & Waligóra, 2011) and (Brucker, Drexl, Möhring, Neumann, & Pesch, 1999) include over 200 related references. Within the RCPSP modelling framework, several generalizations and expansions have also been explored. The following review describes a popular RCPSP mathematical formulation and discusses some relevant variants in Section 2.1. Two time constrained models, the resource availability cost problem (RACP) and the resource leveling problem (RLP) are reviewed in Section 2.2 to provide a perspective on two closely related versions of the RCPSP. Project scheduling under uncertainty is discussed in Section 2.3, including reviews of proactive, stochastic and reactive scheduling approaches. Finally, an overview of project management research in similar maintenance environments is covered in Section 2.4.

2.1 Resource Constrained Project Scheduling

Some of the original mathematical models designed to expand the project scheduling problem from initial CPM and PERT approaches to models that incorporate resource constraints include Wiest's linear programming formulation (Wiest J. D., 1963) and heuristic approaches (Wiest J. D., 1967), and Pritsker *et al.*'s 0-1 programming formulation (Pritsker, Waiters, & Wolfe, 1969). These modelling improvements increased the applicability of the project scheduling problem to include resource constrained environments. The expanded model would come to be known to as the resource constrained project scheduling problem (RCPSP), now considered the base project scheduling model from which the vast majority of project scheduling research is developed.

Pritsker *et al.*'s 0-1 formulation of the RCPSP uses binary, or 0-1 variables, to indicate whether a specific activity *j* finishes at time *t*.

$$x_{jt} = \begin{cases} 1, if\ activity\ j\ finishes\ at\ time\ t \\ 0, otherwise \end{cases}$$

Using a binary variable for each job across all possible periods substantially increases the number of possible solutions the LP solver must consider during the optimization process. To reduce this processing demand, the number of required binary variables can be lowered using durations and predecessor information known about each activity at the start of the project. The maximum length of the project horizon D is determined simply by summing the duration d of each individual activity j for all $\boldsymbol{J}$ activities as shown in (2.1). In this case, the set notation for all project activities can be written as

$\boldsymbol{J} = \{1, \ldots, J+1\}$.

$$D = \sum_{j=1}^{J} d_j \tag{2.1}$$

It is possible to perform a recursive forward and backwards pass over the project network to determine the earliest and latest possible start times for each activity to limit the range of potential variables in the model. The earliest start (ES) time for activity j is the maximum earliest finish time of each of the i activities in activity j's predecessor set P_j. The latest start (LS) time is found working backwards from D, the upper limit of the project horizon found from Eq. (2.1), and subtracting the minimum latest start of all the successor activities that must come after it ($i \in S_j$) plus its own duration d_j. Pseudocode for these procedures with a project start time of 0 is described as follows:

```
Early Start (ES) Algorithm
    $ES_1 := 0;\ EF_1 := 0;$
    $\boldsymbol{for}\ j := 2\ to\ J+1\ \boldsymbol{do}$
    $\boldsymbol{begin}$
        $ES_j := max\{EF_i \mid i \in P_j\};$
        $EF_j := ES_j + d_j;$
    $\boldsymbol{end};$
```

```
Late Start (LS) Algorithm
    $LF_{J+1} := D;\ LS_{J+1} := D;$
    $\boldsymbol{for}\ j := J\ \boldsymbol{downto}\ 0\ \boldsymbol{do}$
    $\boldsymbol{begin}$
        $LF_j := min\{LS_i \mid i \in S_j\};$
        $LS_j := LF_j - d_j;$
    $\boldsymbol{end};$
```

For project networks with a high number of precedence constraints, or network complexity (NC) (Kolisch, Sprecher, & Drexl, 1995), determining the ES and LS for each activity can greatly reduce the number of binary variables required. Projects with a lower network complexity will only see a small improvement in variable reduction since many of the project's activities are not constrained by predecessors and may be scheduled throughout the majority of the project horizon.

A mathematical RCPSP model based on the formulation developed by Pritsker, Waiters, and Wolfe (1969) and expanded by Christofides, Alvarez-Valdez, and Tamarit (1987) is detailed in the following section.

RCPSP Model

Indices:
$0 \ldots J+1$ for activities
$1 \ldots K$ for resource types
$0 \ldots D$ for time horizon

Sets:
P_j immediate predecessors for activity *j*
$\boldsymbol{K}$ resources
$\boldsymbol{J}$ activities

Parameters:
D integer, periods in the planning horizon
J integer, number of activities
d_j integer, duration of activity *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*

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

$$minimize \sum_{t=ES_j}^{LS_j} t \cdot x_{J+1,t} \tag{2.2}$$

$$subject\ to$$

Activity completion

$$\sum_{t=ES_j}^{LS_j} x_{jt} = 1 \qquad \forall\, j \in \boldsymbol{J} \tag{2.3}$$

Predecessors

$$\sum_{b=t}^{LS_j} x_{jb} + \sum_{b=ES_i}^{min\{LS_i, t+d_j-1\}} x_{ib} \leq 1 \qquad \forall j \in \boldsymbol{J}, i \in P_j \tag{2.4}$$

Resource availability

$$\sum_{j=1}^{J} \sum_{b=\max\{t-d_j+1,\, ES_j\}}^{\min\{LS_j, t\}} r_{jk} \cdot x_{jb} \leq R_k \qquad \forall\, k \in \boldsymbol{K}, t \in D \tag{2.5}$$

$$x_{jt} \in \{0,1\} \qquad \forall\, j \in \boldsymbol{J}, t \in ES_j \ldots LS_j \tag{2.6}$$

The objective function, shown as Eq. (2.2), minimizes the start time of activity $J + 1$, a dummy activity with zero duration created to mark the finish milestone of all the jobs in set $\boldsymbol{J}$. Therefore, the start time of activity $J + 1$ also represents the overall project completion time. The predecessor set for activity $J + 1$ includes all jobs not already listed as a predecessor in any other set to complete all paths through the project network.

This model objective is constrained by a job completion constraint, Eq. (2.3), which ensures each activity within the project is scheduled exactly once. This also prevents the solution from omitting any of the activities to provide a shorter completion time.

The precedence constraint in Eq. (2.4) restricts all jobs from starting before any of their predecessors are completed. This constraint is given in the disaggregated discrete time (DDT) format first proposed by Christofides *et al.* (1987) rather than Pritsker *et al.*'s discrete time (DT) constraint from Eq. (2.7).

$$\sum_{t=ES_j}^{LS_j} t \cdot x_{jt} \geq \sum_{t=ES_i}^{LS_i} (t + d_i) \cdot x_{it} \qquad \forall\, j \in \boldsymbol{J}, i \in P_j \qquad 2.7$$

The DT precedence constraint is similar to the one used in the PSP (1.2) and is easier to understand; however, the DDT precedence constraint is shown to usually provide an exact solution in less processing time than the DT constraint due to its stronger LP relaxation (Koné, Artigues, Lopez, & Mongeau, 2011; Möhring, Schulz, Stork, & Uetz, 2003; Sankaran, Bricker, & Juang, 1999). The applicability of the DDT LP model for the type of problem sets evaluated in this thesis is discussed further in Section 3.3.1.

The third constraint, Eq. (2.5), represents the most useful contribution of the RCPSP, the ability to account for resource limitations. The total allocated resources across all $\boldsymbol{J}$ activities must be less than R_k, the total type k resource availability, and must hold across each time t and for each k type of resource.

The final constraint, Eq. (2.6), is a straightforward binary restriction on the key decision variables that determine at what time t activity j will start during the project time horizon.

The traditional objective of the RCPSP shown in Eq. (2.2) is similar to the PSP objective from Eq. (1.1) in that they are both set to minimize the starting time of the final project. An interesting characteristic of this commonly used objective function is that while it provides the shortest schedule or makespan as its solution, it may not schedule loosely constrained activities as soon as it is possible to do so. Scheduling slack activities after their early start time reduces opportunities to accommodate scope changes affecting these activities. Nudtasomboon and Randhawa (1997) is one of the few references to recognize this affect in RCPSP literature when they consider its impact on throughput.

To avoid starting non-critical activities later than necessary when attempting to determine the optimal project makespan, the following steps should be performed:

1. Solve the project schedule using objective Eq. (2.2)
2. Constrain the project makespan to this solution
3. Replace $x_{J+1,t}$ in the objective function Eq. (2.2) with x_{jt} as shown in Eq. (2.8)

$$\sum_{j=1}^{J} \sum_{t=ES_j}^{LS_j} t \cdot x_{jt} \tag{2.8}$$

4. Resolve the model minimizing start times for all activities in $\boldsymbol{J}$

Another approach to account for loosely scheduled slack activities is to use the bi-objective given in Eq. (2.9) below. This modified objective simply adds a weighted version of Eq. (2.8) to Eq. (2.2), creating a secondary objective that minimizes start times for all activities. This secondary objective should have a reduced weighting factor, $\alpha < 1$, to ensure that throughput (minimizing the sum of all start times) is not prioritized over makespan (finishing as early as possible). Using the objective in Eq. (2.8) alone gives a solution that prioritizes all shorter duration activities at the start of the schedule and delays longer activities until the end without actually minimizing project completion time.

$$\sum_{t=ES_j}^{LS_j} t \cdot x_{J+1,t} + \alpha \sum_{j=2}^{J} \sum_{t=ES_j}^{LS_j} t \cdot x_{jt} \tag{2.9}$$

Modifying the objective using either approach ensures all jobs are started as early as possible, gives the shortest makespan with the earliest start time for each activity and removes any potential start time delays for jobs not tightly constrained by predecessors or resources.

Although the RCPSP makespan minimization problem presented in Section 0 is “the standard problem in project scheduling literature” (Hartmann & Briskorn, 2010), it remains challenging to solve optimally for very large projects within a reasonable amount of computation time.

The RCPSP belongs to the class of strongly NP-hard problems (Blazewicz, Lenstra, & Kan, 1983) where the time and number of steps required to solve the optimal solution increases with the size of the problem. For the RCPSP this means that projects with longer time horizons and a large list of activities require an exponentially increasing number of variables that must be solved to find an optimal solution. This increase in variables corresponds to an exponentially increasing requirement for computing power to find exact solutions for these large scheduling problems. Very large problems are usually better solved using heuristic methods that can more efficiently approximate a satisfactory solution rather than attempting to consider all possible solutions.

Despite these mathematical limitations, disaggregated discrete time RCPSP formulations can determine the exact optimal solution, or shortest possible schedule, for moderately sized real world project scheduling problems within a reasonable length of computing time. As commercial solvers improve and computational power increases, it will become increasingly practical to use this type of model formulation for problems with larger time horizons and a large number of activities.

2.1.1 RCPSP Variants and Extensions

Following the inclusion of resources into the project scheduling problem and the resulting difficulty in solving it, awareness of the RCPSP grew quickly and the problem soon became a high interest research area. Academic research in the RCPSP now covers a wide range of approaches such as improving solution times using heuristic or exact approaches, making a variation of the problem computationally practical or introducing new problem variants to expand the applicability of the RCPSP and allow it to more closely model real-world project conditions (Coelho & Vanhoucke, 2011; Schutt, Feydy, Stuckey, & Wallace, 2013). Brucker *et al.* suggest a classification scheme to describe the different RCPSP variations by resource characteristics, activity information, and the problem's objective (Brucker, Drexl, Möhring, Neumann, & Pesch, 1999). Under the suggested scheme, the RCPSP would be classified as $PS \mid prec \mid C_{max}$, referring to a project scheduling (PS) resource environment with precedence constraints ($prec$) and an objective of minimizing makespan (C_{max}).

An adaptation of Schwindt and Zimmermann's table of popular variants of the resource constrained project scheduling problem is listed in Table 3 (2015). For more in-depth information on the scope of project scheduling problem covered in the literature, resources available to the reader include Demeulemeester and Herroelen's research handbook (2002) and Schwindt and Zimmermann's comprehensive handbook (2015). Hartmann and Briskorn (2010) and Węglarz, Józefowska, Mika, and Waligóra (2011) provide recent surveys of many RCPSP extensions.

Table 3: Extensions of the standard project scheduling problem

Attribute	**Characteristics**
Type of constraints	Time-constrained problem
	Resource-constrained problem
Type of precedence relations	Ordinary
	Generalized (min/max time lags)
Type of resources (workers, machines, money, spare parts, raw materials, etc.)	Renewable (workers)
	Non-renewable (overtime)
	Cumulative (reservoir, cash flow)
	Continuous (energy, money)
	Variable requests and availability
Type of activity splitting	Non-pre-emptive
	Integer pre-emption
	Continuous pre-emption
Number of execution modes	Single-mode
	Multi-mode (project crashing)
Number of objectives	Single-criterion (makespan)
	Multi-criteria (cost, quality, resource usage)
Type of objective function	Regular (improved objective reduces makespan)
	Non-regular (improved objective may increase makespan)
Level of information	Deterministic
	Stochastic
	Unknown level of uncertainty

The standard RCPSP model alone remains limited in its applicability to real world project scheduling in industries with fixed deadlines and limited incentive to finish early, and in projects with a large degree of inherent uncertainty. The following section reviews time-constrained project scheduling models that are less commonly studied in the literature but often have practical application.

2.2 Time Constrained Project Scheduling

Project-focused industries are typically concerned with resource usage since project performance and execution time is primarily constrained by the availability of resources. Due to the nature of the naval maintenance project environment considered in this research, time constraints imposed by the assigned project time window more impactful than the resource constraints. Two of the more popular time constrained project scheduling models are reviewed below.

2.2.1 Resource Availability Cost Problem

The resource availability cost problem (RACP) is a variation of the RCPSP with a non-regular objective function. In contrast to the RCPSP's standard, objective of reducing the project duration, the RACP objective function is considered non-regular because it reduces per-period resource usage at the expense of increasing the project makespan. Specifically, the RACP is concerned with using the lowest number of each type of resource in any given time period across the predetermined project horizon, or minimizing the maximum number of resources required in each period. The number of required resources effectively represents the availability cost or resource investment necessary to complete the project within the given time period. In contrast to the time-based objective used by the RCPSP where the shortest feasible schedule is found according to given resource restrictions, the RACP optimizes for a resource-based objective that solves for lowest cost resource allocation schedule within a specified deadline.

The resource availability cost problem, also referred to as the resource investment problem (RIP), was initially studied and presented by Möhring (1984). Demeulemeester provided a clever minimum bounding algorithm (MBA) (Demeulemeester E. , 1995) which iteratively solves feasible solutions at efficient points to find the optimal availability cost solution. Demeulemeester's efficient points strategy determines lower bounds for all resource demands by iteratively relaxing resource constraints for each resource. More recently, Rodrigues and Yamashita (2010) update this exact algorithm by organizing Demeulemeester resource relaxation with a branching strategy that cuts dominated solutions while generating good bounds. Heuristic approaches have also been considered in recent research including an artificial immune system (AIS) algorithm (Van Peteghem & Vanhoucke, 2013), genetic algorithms (Ranjbar, Kianfar, & Shadrokh, 2008; Shadrokh & Kianfar, 2007), particle swarm optimization

(PSO) (Qi, Guo, Lei, & Zhang, 2014) and an invasive week optimization (IWO) algorithm (Van Peteghem & Vanhoucke, 2015).

The RACP can be formulated mathematically using a similar set of indices, sets, parameters and variables as was presented for the RCPSP in Section 0. The two changes shown below are a modified objective from makespan minimization to resource cost minimization and the maximum resource availability R_k parameter from Eq. (2.5) now represents an additional set of variables that determine the minimum resource level required to complete the project by the given deadline.

RACP Model

Indices:
$0 \ldots J+1$ for activities
$1 \ldots K$ for resource types
$0 \ldots D$ for time horizon

Sets:
P_j immediate predecessors for activity *j*
$\boldsymbol{K}$ resources
$\boldsymbol{J}$ activities

Parameters:
c_k integer, cost per unit of resource k
D integer, periods in the planning horizon
J integer, number of activities
K integer, number of resource types
d_j integer, duration of activity *j*
r_{jk} integer, activity j demand for resource type k
ES_j integer, earliest start time of activity *j*
LS_j integer, latest start time of activity *j*

Variables:
R_k, integer, maximum per period demand for resource type k
x_{jt}, binary, 1 if activity j starts at time t, 0 otherwise

$$minimize \sum_{k=1}^{K} c_k R_k \tag{2.10}$$

$$subject\ to$$

$$\sum_{t=ES_j}^{LS_j} x_{jt} = 1 \qquad \forall\, j \in \boldsymbol{J} \tag{2.11}$$

$$\sum_{b=t}^{LS_j} x_{jb} + \sum_{b=ES_i}^{min\{LS_i, t+d_j-1,\}} x_{ib} \leq 1 \qquad \forall j \in \boldsymbol{J}, i \in P_j \tag{2.12}$$

$$\sum_{j=1}^{J} \sum_{b=\max\{t-d_j+1,\, ES_j\}}^{\min\{LS_j, t\}} r_{jk} \cdot x_{jb} \leq R_k \qquad \forall\, k \in \boldsymbol{K}, t \in D \tag{2.13}$$

$$x_{jt} \in \{0,1\} \qquad \forall\, j \in \boldsymbol{J}, t \in ES_j \dots LS_j \tag{2.14}$$

$$R_k \geq 0 \qquad \forall\, k \in \boldsymbol{K} \tag{2.15}$$

The RACP objective function in Eq. (2.10) is shifted from minimizing the total project makespan to minimizing the maximum per-period resource R_k and consumption cost c_k for each individual resource k in the set of resources $\boldsymbol{K}$. Equation (2.15) is also added to define R_k as a non-negative variable that determines the largest amount of resource k required in any given period.

The objective function from Eq. (2.10) is further constrained by the activity completion constraint Eq. (2.11), precedence constraint Eq. (2.12), resource constraint Eq. (2.13) and binary decision variable constraint Eq. (2.14) which each remain identical to those previously listed for the RCPSP model formulation as Equations (2.3), (2.4), (2.5) and (2.6) in Section 0.

2.2.1.1 RACP Variants and Extensions

The RACP is useful in project scenarios where committed resources are considered tied to the project until its completion. In this situation, choosing a schedule with the lowest number of each resource type will result in the lowest resource cost. The RACP is particularly relevant to project teams that are exclusively assigned to one project for its full duration before being released to other activities. Projects that do not release committed resources prior to the prescribed deadline may also be modeled as RACP instances. In these scenarios, efficient resource utilization is likely considered more valuable than project execution time. In most project environments, resources usually become available for use on other projects or no longer bill against the project when they are not directly working on a project activity. Resultantly this problem is infrequently encountered in industry and only a relatively small amount of research expanding on the RACP has been published to date.

A similar model can evaluate time/resource trade-offs which also minimize the maximum total resource usage throughout the project duration. Trade-offs can be explored by adding additional execution modes to the standard RACP. This model expansion is called the multi-mode resource availability cost problem (MRACP). The MRACP models a project completion scenario where the deadline is essentially a flexible due date with cost penalties for finishing late and bonus payments for finishing early. Solving the MRACP for multiple deadlines will determine exact time/resource cost curves (Yamashita & Morabito, 2009) for small project instances. Larger projects with much longer computation times are evaluated heuristically in (Colak & Azizoglu, 2013) and (Van Peteghem & Vanhoucke, 2015).

Robust optimization can be considered for the RACP in a model called the resource availability cost problem with scenarios (RACPS) (Yamashita, Armentano, & Laguna, 2007). In the RACPS model, solution robustness is introduced by solving the RACP for a set of equally likely scenarios and choosing the solution that works best across all of these possible scenarios. Section 2.3.1 covers proactive or robust project scheduling in further detail.

The RACP objective is, to some extent, more representative of the type of project situation found in naval maintenance work periods that are more tightly constrained by time rather than resources; however, minimizing availability cost does not meaningfully describe naval maintenance objectives where projects release resources to a shared pool once the assigned activity is completed.

2.2.2 Resource Levelling Problem

The resource levelling problem (RLP) is another type of time-constrained project scheduling problem where activities and their corresponding resource demands are scheduled to meet a predetermined deadline. The traditional RLP objective is to provide a project schedule capable of completing all project activities prior to the due date using a levelled or smoothed resource profile where the resource demand in each period is as close as possible to the average demand throughout the project duration. This objective is most often modelled in literature by scheduling a project's resource usage with minimal variance from one period to the next.

The concept of resource levelling originates in literature not long after academic research on the project scheduling problem. Burgess and Killebrew first suggested a generalized heuristic to minimize the sum of squared resource usage in each period (1962). Minimizing the sum of squared resource usage in each period exponentially penalizes high resource levels and leads to solutions with generally constant resource demand. Improved heuristics were later given by (Bandelloni, Tucci, & Rinaldi, 1994) and (Neumann & Zimmermann, 1999).

Exact solutions for the resource levelling problem include algorithms requiring implicit enumeration developed by Ahuja (1976) and Younis and Saad (1996), an integer programming approach from Easa (1989), a dynamic programming solution (Bandelloni, Tucci, & Rinaldi, 1994) and Neumann and Zimmermann's successive scheduling branch-and-bound procedure (2000).

Conceptual descriptions of the RCPSP and RACP were modeled previously using binary time-indexed variables. An example of the resource leveling problem formulation using this same time-indexed format is given as follows:

2.2.2.1 RLP Formulation

Indices:
$0 \dots J+1$ for activities
$1 \dots K$ for resource types
$0 \dots D$ for time horizon

Sets:
P_j immediate predecessors for activity j
$\boldsymbol{K}$ resources
$\boldsymbol{J}$ activities

Parameters:
c_k integer, cost per unit of resource k
D integer, periods in the planning horizon
J integer, number of activities
d_j integer, duration of activity j
r_{jk} integer, activity j demand for resource type k
ES_j integer, earliest start time of activity j
LS_j integer, latest start time of activity j

Variables:
R_{kt}, integer, demand of resource type k in period t
x_{jt}, binary, 1 if activity j starts at time t, 0 otherwise

$$minimize \sum_{k=1}^{K} c_k \sum_{t=0}^{D-1} {R_{kt}}^2 \qquad 2.16$$

$$subject\ to$$

$$\sum_{t=ES_j}^{LS_j} x_{jt} = 1 \qquad \forall\, j \in \boldsymbol{J} \qquad 2.17$$

$$\sum_{t=ES_j}^{LS_j} t \cdot x_{jt} \geq \sum_{t=ES_i}^{LS_i} (t + d_i) \cdot x_{it} \qquad \forall\, j \in \boldsymbol{J}, i \in P_j \qquad 2.18$$

$$R_{kt} \geq \sum_{j=1}^{J} r_{jk} \sum_{b=\max\{t-d_j+1,\, ES_j\}}^{\min\{LS_j, t\}} x_{jb} \qquad \forall\, k \in \boldsymbol{K}, t \in D \qquad 2.19$$

$$x_{jt} \in \{0,1\} \qquad \forall\, j \in \boldsymbol{J}, t \in ES_j \ldots LS_j \qquad 2.20$$

$$R_{kt} \geq 0 \qquad \forall\, k \in \boldsymbol{K} \qquad 2.21$$

Like the RCPSP and RACP, the resource leveling problem also makes use of the common job completion constraint in Eq. (2.17), predecessor restrictions Eq. (2.18) and binary start time decision variable Eq. (2.20). The resource availability constraint in Eq. (2.19) and resource level decision variables in Eq. (2.21) are now indexed by time period t to allow for a variable resource profile and enumeration of project resource consumption at each time index (Schwindt & Zimmermann, 2015).

The objective function given in Eq. (2.16) minimizes the weighted sum of squared resource usage, which is now no longer a nonlinear objective. The parameter c_k represents the cost or weight associated with of using each unit of resource k. Using linear programming techniques for this resource levelling objective and most of the other resource levelling objectives mentioned in the literature requires a linearization of the objective function before solving. A description of exact linearization techniques for this resource levelling objective is covered by Rieck, Zimmermann, and Gather (2012).

Several other resource levelling objectives are proposed in literature to represent many other management priorities. A few of the frequently used minimization objectives are shown below.

$$\sum_{k=1}^{K} c_k \sum_{t=0}^{D-1} |R_{kt} - A_{kt}| \qquad 2.22$$

$$\sum_{k=1}^{K} c_k \sum_{t=0}^{D-1} |R_{kt} - R_{kt+1}| \qquad 2.23$$

The first objective listed in Eq. (2.22) minimizes the absolute deviation from a given resource level which may vary over the project time horizon or be set to a fixed value. Equation (2.23) minimizes the difference in resource usage from one adjacent period to the next.

The absolute values in these two equations are nonlinear and require linearization before they can be used by a linear programming (LP) solver. The linearization of Eq. (2.22) is shown in Equations (2.24), (2.25) and (2.26). Equation (2.23) can be linearized in the same manner.

$$minimize$$

$$\sum_{k=1}^{K} c_k \sum_{t=0}^{D-1} ({u_{kt}}^{+} + {u_{kt}}^{-}) \qquad 2.24$$

$$subject\ to$$

$$R_{kt} - A_{kt} = {u_{kt}}^{+} - {u_{kt}}^{-} \qquad 2.25$$

$${u_{kt}}^{+}, {u_{kt}}^{-} \geq 0 \qquad 2.26$$

2.2.2.2 RLP Variants and Extensions

Research into the resource levelling problem also includes additional extensions to the model described in the previous section. A closely related model is the total overload cost problem (TOCP) (Easa, 1989; Rieck, Zimmermann, & Gather, 2012), which uses a similar formulation but slightly modifies the objective to Eq. (2.27). The TOCP will only penalize increases in utilization above a fixed amount which could be used to model overtime premiums or other similar costs related to exceeding a given resource utilization level.

$$\sum_{k=1}^{K} c_k \sum_{t=0}^{D-1} max(0, R_{kt} - A_{kt}\ ) \qquad 2.27$$

The total adjustment cost problem (TACP) is another closely related problem that solves a similar objective (Kreter & Zimmermann, 2014). The TACP evaluates the cumulative costs of increasing and decreasing the utilization of project resources in adjacent periods. This problem type is commonly used to model hiring and layoff costs in industries where changing the number of resources used from period to period results in increased project costs.

Naval maintenance projects do not involve recognizable costs from increasing or decreasing resource usage from one period to the next and are generally resource feasible. The RLP does present similar features in terms of evaluating overtime usage and solving for a schedule with a given deadline that can be applied to naval maintenance project objectives.

From strictly a deterministic standpoint, the previously discussed problems do not represent naval maintenance projects due to their inability to account for sources of uncertainty. An exact solution to the RCPSP with a makespan minimization objective will be a very tight solution with minimal remaining resource availability and as little as possible slack between activities. This type of solution initially appears to be very efficient, but similar to the example discussed in Section 1.6; even minor unexpected events have the potential to make this schedule completely infeasible. In naval maintenance projects, and almost all other project environments, effectively managing uncertainty is a critical determinant of project success. Discussion on expanding project scheduling models to account for uncertainty is continued in in the following section.

2.3 Project Scheduling Under Uncertainty

The majority of published work on the previously discussed scheduling problems focuses exclusively on a project scenario where all data is considered deterministic and available during the scheduling phase. Projects are unique undertakings, which makes it likely that many projects have considerable uncertainty regarding activity duration, resource requirements and resource availabilities. The uncertainty inherent in the naval maintenance environment would also place naval maintenance projects in this situation. As shown in Figure 6 previously, schedule disruptions frequently occur in naval maintenance projects due to a variety of factors such as weather, equipment breakdown, unforeseen resource conflicts, scope growth, lack of materials and conflicts with other activities.

Accounting for peripheral, uncontrollable events in project scheduling is broadly categorized as one of three approaches: proactive scheduling, stochastic scheduling or reactive scheduling.

Proactive scheduling is designed to be as robust as possible prior to project execution, stochastic scheduling uses known probability distributions to plan scheduling policies in response to uncertainty as the project unfolds, and reactive scheduling which repairs the baseline schedule following unplanned project variability.

2.3.1 Proactive Scheduling

Proactive scheduling, also referred to as robust scheduling, has received an increasing level of interest over the last several years and is particularly relevant in situations where it is not possible to obtain or collect probability information for project data. The primary catalyst for growing interest in robust scheduling procedures was Goldratt's proposed critical chain and buffer management method (CC/BM) based on the theory of constraints (TOC) (1997). Herroelen and Leus (2005) provide a detailed review Goldratt's CC/BM approach using time buffers to protect an initial baseline schedule from disruption due to project uncertainty. In their review, they address multiple shortcomings of this method including vague tiebreaking rules and overly conservative schedule buffering.

In Lambrechts, Demeulemeester, & Herroelen (2008), eight proactive strategies are proposed for generating robust baseline schedules including scheduling according to cumulative instability weight (weighting activities according to their perceived necessity to begin when scheduled), and inserting time and resource buffers. They also propose a reactive policy of reoptimizing the schedule following unplanned disruptions or rescheduling activities in their original sequence, if reoptimization is not practical.

Proactive scheduling has also been applied to uncertain project activity costs in the discrete time/cost trade-off problem (Hazır, Erel, & Günalay, 2011). The authors evaluate schedule robustness using three cost based measures; expected realized cost, worst-case cost and cost of the reference scenario. Li and Wu further examine the time-cost trade-off problem using a multiobjective robust optimization model that solves for robust Pareto solutions under varying risk levels (2014). Robust schedules with uncertain activity durations was also recently studied in (Artigues, Leus, & Nobibon, 2013) which considers the minmax absolute-regret robustness measure for the RCPSP.

Research into general robust optimization may also be used to model uncertainty in project information. Bertsimas and Sim develop a robust optimization model that permits only a subset of potential disruptions; a project schedule is affected by γ out of n possible deviations (for $\gamma \leq n$). A comprehensive description of robust optimization developments in mathematical programming is also available to the reader for further explanation of this area (Ben-Tal, Ghaoui, & Nemirovski, 2009).

2.3.2 Stochastic Scheduling

Managing schedule uncertainty using stochastic programming, a popular response considered in the literature, assumes the probabilities for uncertainties in project information is known. This approach is only possible if accurate historical data is sufficient to develop meaningful probability distributions. Recent work on project scheduling with stochastic activity durations includes a sequence model formulation where activity order and start times are determined (Lamas & Demeulemeester, 2014) and proactive policies to protect schedule stability (Deblaere, Demeulemeester, & Herroelen, 2011).

Research into project uncertainty also includes uncertain resource availabilities that can be used to model equipment breakdowns, employee absenteeism, or unexpected conflicts with other projects. Lambrechts, Demeulemeester and Herroelen introduce this variant in (2008). Lamas and Demeulemeester (2014) use chance constrained programming to account for stochastic activity durations and introduce a new robustness measure to quantify the joint probability that each activity actually starts at the time it was initially scheduled to.

Stochastic task insertion (STI) is studied in (Archer, 2008) which evaluates a series of buffer insertion strategies in response to stochastic events and provides recommendations for how project managers can develop predictive schedules when stochastic distributions of unexpected events are known.

The opportunity to apply stochastic scheduling procedures is quite limited in the naval maintenance environment where the level of uncertainty in the known project information exceeds quantifiable probabilistic durations. Uncertainty can also exist in resource requirements and types, and even in what activities will be completed as part of the project portfolio.

2.3.3 Reactive Scheduling Methods

Reactive scheduling policies are often used in conjunction with a proactive procedure that generates the robust baseline schedule that is then modified by the reactive policy after a schedule breakdown. Reactive scheduling approaches are most applicable in project environments where significant disruptions are likely to occur and with high frequency (Szelke & Kerr, 1994). The complexity of schedule repair actions in the literature vary from simple schedule repair action such as right shifting all activities effected by the disruption (Smith, 1995) to other methods that completely reschedule all of the remaining incomplete activities according to a selected priority rule (Rajendran & Holthaus, 1999).

Other reactive approaches include Van de Vonder *et al.*'s use of heuristics to repair project schedules following an unexpected breakdown. The schedule repair heuristic seeks to minimize the deviation between the original schedule and the actual schedule (Van de Vonder, Ballestin, Demeulemeester, & Herroelen, 2007). This model is further expanded to include mode-switching costs in the rescheduling objective in addition to minimizing deviations in activity starting time (Deblaere, Demeulemeester, & Herroelen, 2011). A large list and description of potential reactive scheduling policies is covered by Möhring *et al.* (1984).

The extreme response to scheduling in a highly indeterminate environment is to eliminate baseline scheduling and instead rely on a strictly reactive policy to assign resources on a short-term, possibly daily, basis. This type of reaction would negate the gains from optimally allocating resources at an organizational level for the project activities that do occur as expected. Baseline schedules are also important for quoting accurate delivery dates, ordering project materials, observing management priorities, determining meaningful objectives, establishing time windows for independent work, and measuring and controlling project progress. Benefits of scheduling according to a stable baseline are discussed further in (Aytug, McKay, Mohan, & Uzsoy, 2005; Van de Vonder, Ballestin, Demeulemeester, & Herroelen, 2007).

Artigues and Roubellat expand reactive scheduling from uncertain activity durations and resource availabilities to include unexpected activity insertion using a polynomial activity insertion algorithm (2000). Their proposed insertion algorithm attempts add an unexpected activity to the schedule while minimizing the maximum lateness of the project.

A recovery problem is introduced in (Zhu, Bard, & Yu, 2005) which manages schedule disruptions by solving for the lowest cost response required to get the project back in line with the initial baseline plan. Response actions include rescheduling, alternative activity execution modes, supplementing current resources and activity cancellation.

The reactive scheduling policies in place at the naval maintenance facility studied for this research generally involve trade-off decisions where additional resources may be made available by trading off a lower priority activity on the project in question or cancelling an activity from another concurrent project.

2.4 Project Management in Marine Dockyards

A number of other publications recognize the potential cost improvements from optimizing marine maintenance project scheduling in both commercial and naval environments. One of the more encompassing approaches to naval maintenance planning is a decision support system (DSS) used to provide the Royal Netherlands Navy with workload-based capacity planning to allocate work through an adaptive search RCPSP model (de Boer, Schutten, & Zijm, 1997). The presented DSS utilizes deterministic planning approaches that do not plan or account for any project uncertainty. An update on de Boer *et al.*'s method is included in (Hans, Herroelen, Leus, & Wullinka, 2007) which mentions the value of including uncertainty in the model. Hans *et al.* also introduce a case study of a private ship repair yard that was inefficiently using a single-project approach while attempting to complete several ship repair projects at the same time using a resource pool shared between each of the projects. The authors suggest applying variability reduction techniques prior to considering reactive policy rules or robust prescheduling.

A production planning DSS is also proposed by Pinha and Ahluwalia for short term planning in the commercial ship repair industry (2014). Their model uses event driven simulation to analyze uncertainty in resource usage and future resource demand but propose only reactive scheduling responses to these potential disruptions.

Boyle, Little, Manning and van der Krogt describe the successful implementation of their constraint-based mathematical model developed to schedule naval maintenance for the Irish Naval Service (2011). Boyle *et al.* highlight the model's response to uncertainty; however, this response is purely reactive and only considers complete rescheduling following schedule breakdown. The project data is also considered deterministic even though uncertainty is expected.

Certa, Galante, Lupo, and Passannanti solve for a multi-objective Pareto frontier applied to naval repair decisions (2011). The mathematical model presented minimizes the global maintenance cost and the system downtime required to complete the assigned work. The Pareto frontier is used to describe the trade-off between the benefits of naval maintenance vs the cost of reduced availability while the maintenance is being completed. All parameters evaluated by this algorithm are assumed known and not subject to any variability.

None of the reviewed references present a proactive scheduling model designed to account for uncertainty in activity duration, resource demand and portfolio. Developing robust scheduling policies to protect against each of these three areas of uncertainty will increase schedule stability and allow for improved project performance.

Chapter 3: Research Methodology

This chapter details the research procedure followed during the development of this thesis. This procedure includes development of the problem scope, initial LP modelling, validation using test sets, and using model outputs to generate improved scheduling policies.

The initial premise of this thesis originated from discussions with the east coast RCN Engineering Operations department regarding the challenges of meeting the long term weekly production targets set by the Annual Operating Plan (AOP). These weekly production targets are developed to project a year of upcoming maintenance based on expected demand and historical trends. The AOP maintenance forecast is subject to significant uncertainty and its target maintenance levels are frequently updated on a short term basis to match practical outputs for upcoming weeks.

Understanding the challenges of AOP development involved job shadowing and interviewing relevant personnel in management, operations and production departments. This job shadowing process began in Business Operations and Planning with the Operations Analyst directly involved in generating the historical and forecast data used to create the AOP. We identified opportunities to reduce uncertainty using essential preventive maintenance schedules to create baseline demand levels. Improving historical indicators using average weekly maintenance hours in proportion to the assigned length of the work period was also considered. Finally, we requested access to the operational schedule to help predict which vessels would require operational engineering changes or would be expected to require additional maintenance hours based on workload. Incorporating each of these factors will contextualize historical maintenance levels in a more meaningful manner and help identify useful trend information.

Shadowing and interviewing employees involved in the project management process included meetings with planners, schedulers, supply officers, project leaders and portfolio managers. During this time, existing short term project scheduling practices were observed and much of the project performance data referred to in Section 1.5 was collected.

An analysis of this project performance data and information provided during interviews with project management stakeholders revealed that uncertainty in short term maintenance scheduling was also a challenge to effective project management. The project measures presented in Section 1.5 indicate that short term disruptions have a negative effect on overall performance and a lack of schedule robustness lowers overall facility outputs.

Multiple project scheduling models and formulation techniques were considered to evaluate potential schedule robustness improvements. Further details of these approaches are covered in Chapter 2. The model formulation used during primary experimentation is based on Christofides *et al.*'s discrete disaggregated time (DDT) mixed integer linear program (MILP) (1987) for the multi-mode resource constrained project scheduling problem (MRCPSP). A modified objective and additional constraints are added to account for a larger range of constraining factors.

3.1 Expanded Model Characteristics

The project characteristics captured by the model presented in this thesis are designed to represent real world project management conditions found in a naval maintenance and repair facility. In addition to the typical constraints found in the standard RCPSP, such as precedence relations, several other project attributes are modelled. The following paragraphs outline the components included in this model to account for many of the specific conditions found in naval maintenance projects.

3.1.1 Fixed Time Window with Due Date

Within the naval maintenance environment, maintenance projects are assigned to time blocks in which the maximum amount of essential and high opportunity maintenance work is completed by a shared pool of resources. A due date is modelled to represent the fixed end of the project managed work period, after which the ship will no longer be available for maintenance work.

3.1.2 Variable Resource Availability

Project resources are drawn from a pool of facility resources that includes materials, equipment and labour. Resources are scheduled for time units consistent with the project time horizon, which would typically be in days, but hours or weeks would also be a reasonable option. Resource availability over the project horizon is not fixed. Variations may occur due to employee vacations, equipment unavailability and resource allocations to other projects. Accordingly, resource availability for renewable resources is given on a daily basis to model fluctuations in the resource profile.

3.1.3 Multiple Resource Types

The material, equipment and labour can be represented by different renewable resource types that can be reused each day up to their capacity level. Project activities may require only one resource type or several resource types. Unallocated renewable resources cannot be carried forward to a future period meaning that all unused renewable resource capacity is essentially lost.

3.1.4 Multiple Execution Modes with Quality Considerations

For some project activities, it may be possible to complete the job requirements using a secondary method. In naval maintenance routines there are certain types of work where a temporary repair is implemented and the more comprehensive procedure is postponed until the next available work period. The decision to accept a decreased level of performance to save time results in a corresponding decrease in scope, which could negatively affect job quality, equipment redundancy or operational capability. A temporary repair would not be expected to last as long as a full repair and would have lower durability. Another type of temporary maintenance action could involve accepting repairs on only a primary component while delaying service for the secondary component. In this case, there would be no backup option if the first component failed. Activity crashing decisions, where an activity duration may be decreased but only with a resulting higher resource requirement, via overtime or subcontracting, can also be represented using multiple execution modes.

3.1.5 Model Objective Statement

Several possible scheduling objectives have been previously discussed, any of which may be used when creating project schedules designed to meet management guidelines. While makespan minimization is most typically chosen, it is also helpful to consider other resource-based objectives such as time/cost trade-offs when deciding whether to extend the project due date to reduce costs or to consume more resources in an effort to finish early. The project managed work periods studied for this research do not fall under usual project guidelines where a set of activities are used to create a project time window. Instead, a project time horizon is made available in advance and then activities are selected and scheduled to fit within this assigned work period.

The experimentation process, explained further in 3.4, used multiple objectives to evaluate scheduling strategies. The first step involved using the makespan minimization objective to determine the shortest makespan for the fixed list of 30 project activities. The time-varying resource disruptions in the RCPSP/t instances were then examined to determine which size buffer would be applicable. Of the 30 instances considered, 29 had less than ten percent of the periods throughout the project horizon affected by disruptions. To provide a more robust scheduling window in response to this expected level of disruption, the minimized maximum makespan was increased by 10 percent. This buffered project makespan was then evaluated using a periodic buffering objective and a weighted buffering objective. Each buffering strategy was tested using random resource disturbances to evaluate its potential scheduling robustness.

3.1.6 Model Assumptions

Several conditions are assumed to follow the traditional RCPSP MILP format. In the presented model, vetting the activities included in the initial project portfolio is not under consideration. It is assumed that the pool of activities included in the project will allow for a feasible schedule within the available resources and project time window. Project input data regarding precedence constraints, activity durations and resource requirements are assumed known in advance. Resource availability forecasts are known but subject to random periods of reduced availability due to parallel projects, vacations, etc. Project activities cannot be pre-empted or temporarily paused, once they are scheduled and started, they must continue until completion or be cancelled.

3.2 Modelling Robustness

Two approaches were initially considered to improve project scheduling performance. The first approach to improving schedule robustness was to use known variations in resource availability to arrange activities around expected bottlenecks in the project timeline. The second was to level activity priorities to improve trade-off options when deconflicting schedules. Both of these approaches are described below.

3.2.1 Resource and Time Buffering

Resource availability is not constant across the full project timeline due to a number of factors already mentioned. Variations in resource availability are common in naval maintenance work periods as well as in other multi-project environments where pooled resources have already been assigned to concurrent projects with different start times and different priority levels.

Variations in the resource availability profile can create time windows where capacity is relatively scarce. The RR_{kb} parameter represents the total resource k capacity available at time instance b. To limit the range of possible start time variables that must be considered, the total resource demand is calculated for all activities over all feasible time periods and is referred to as resource availability Ra_{jkt}, subscripted for activity j, resource k and time t. The Ra_{jkt} parameter is calculated by summing per period resource availability starting at the activity's early start time up until the activity is expected to finish as shown in Eq. (3.1). This calculation is determined for each activity and resource over all feasible periods up to the activity finish time.

$$Ra_{jkt} = \sum_{b=t}^{\min\{D-1,t+d_j-1\}} RR_{kb} \qquad \forall\, j \in \boldsymbol{J}, k \in \boldsymbol{K}, t \in ES_j..LS_j \qquad 3.1$$

A sample output of this parameter is shown in Table 4. These values are calculated for activity 2 from the example referred to previously in Sections 1.3 and 1.6.

Table 4: Sample values for the resource availability parameter

$Ra_{2,1,0}$	**38**	$Ra_{2,1,6}$	**39**	$Ra_{2,1,12}$	33
$Ra_{2,1,1}$	**37**	$Ra_{2,1,7}$	**38**	$Ra_{2,1,13}$	34
$Ra_{2,1,2}$	**36**	$Ra_{2,1,8}$	**36**	$Ra_{2,1,14}$	35
$Ra_{2,1,3}$	**36**	$Ra_{2,1,9}$	34	$Ra_{2,1,15}$	**36**
$Ra_{2,1,4}$	**37**	$Ra_{2,1,10}$	32	$Ra_{2,1,16}$	**36**
$Ra_{2,1,5}$	**38**	$Ra_{2,1,11}$	32	$Ra_{2,1,17+}$	≤27

The resource availability parameter also helps narrow the solution space by identifying infeasible start times using known activity information. The sample activity Ra_{jkt} values listed in Table 4 are calculated using activity 2, which has an expected resource demand of 9 units and an expected duration of 4 periods. From this information, all Ra_{jkt} values below 36 ($duration * resource\ demand$) can be eliminated as feasible start times. Only the start times shown in bold in Table 4 still need to be evaluated. Higher Ra_{jkt} values indicate start times that will provide greater resource flexibility during project execution, which helps increase the possibility of the activity starting as scheduled even if project disruptions occur.

The feasible window defined by the resource availability profile for activity 2 is displayed visually in Figure 9. The greyed out time blocks represent infeasible periods where no combination of duration and resource demand will meet the expected activity requirements.

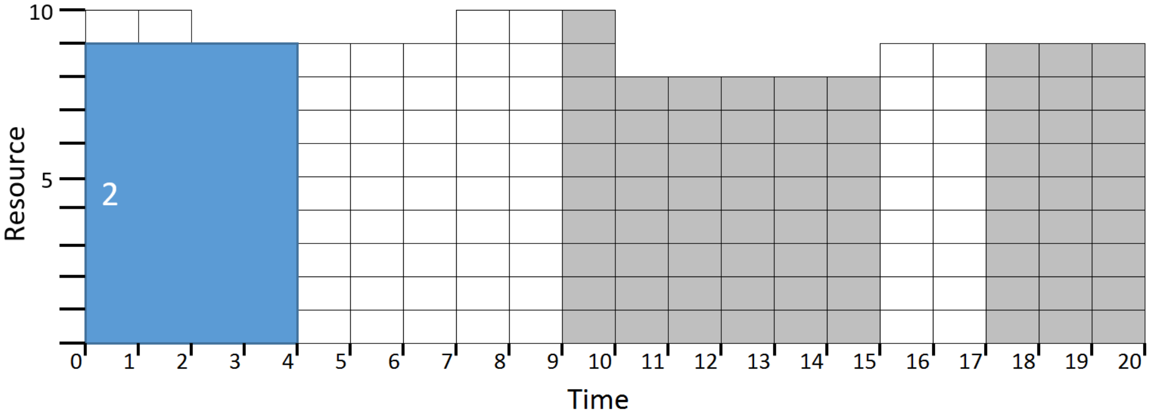

Figure 9: Resource Availability Profile for Activity 2

The primary benefit of the variable resource profile information is that it can be used to identify activity start times that best take advantage of large resource availabilities and limit the number of activities that are scheduled when resource availability is low. Avoiding known resource bottlenecks adds additional robustness to the project schedule by preserving resource capacity for potential schedule disruptions.

Per period resource availability can be incorporated into the RCPSP model objective using Eq. (3.2), which now includes activity resource request levels r_{jk} (resource demand of activity j for resource k) that were not a part of the initial Ra_{jkt} calculations. Maximizing this objective for all N activities and K resources provides a solution where longer activities with large resource demands are scheduled during windows of high resource availability and outside of potential resource bottlenecks whenever it is possible to do so.

$$Maximize \sum_{k=1}^{K} \sum_{j=1}^{N} \sum_{t=ES_j}^{LS_j} x_{jk} r_{jk} Ra_{jkt} \qquad \forall\, j \in \boldsymbol{J} \qquad 3.2$$

3.2.2 Performance Buffering by Levelling Activity Priorities

It is also possible to limit the impact of unexpected changes in maintenance demand on high priority activities already by reducing the number of essential tasks that are executed concurrently. When two high priority tasks are scheduled in parallel, a schedule disruption can cause one of these essential activities to be delayed or dropped from the project, if no other feasible time window exists. By levelling out essential tasks whenever possible, lower priority activities scheduled in parallel can be used as an additional resource buffer to ensure high priority activities can be completed. Priority levelling is also more straightforward for project leaders to determine which task to delay or cancel to free up additional resources that are needed elsewhere. Figure 10 shows an example of a schedule with resource and time buffers but not levelled by priority. Figure 11 gives a priority levelled schedule that includes activity priority information in the scheduling process. It should be noted that precedence constraints were not considered in this simplified example.

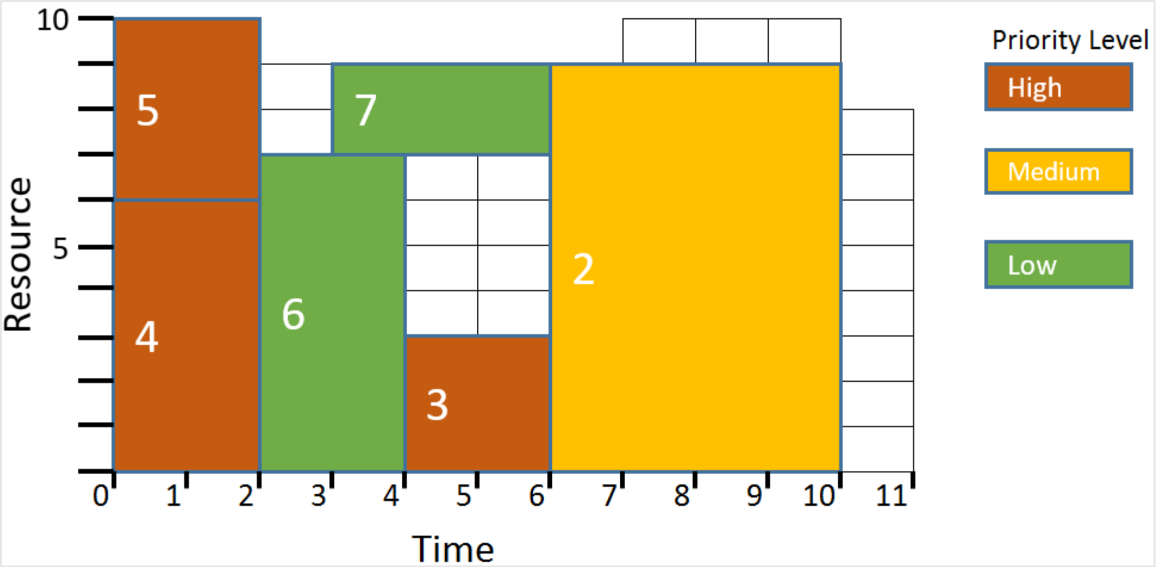

Figure 10: Buffered schedule unlevelled by activity priority

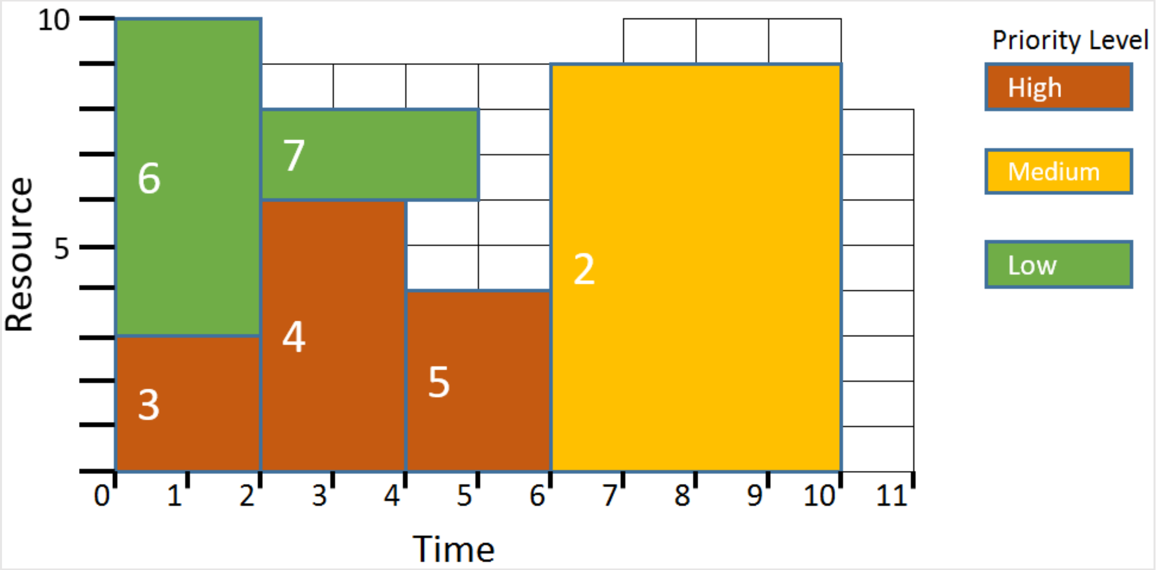

Figure 11: Buffered schedule also levelled by activity priority

These two figures show that using the schedule from Figure 10 will result in the delay or cancelation of a high priority activity should available resources decrease by any amount in either of the first two periods. Conversely, it would take a resource shortfall of more than 3 units to disrupt a high priority activity using the schedule from Figure 11.

3.3 Linear Programming Feasibility

Many recent research efforts into RCPSP heuristics cite the significant computing resources required to determine exact or optimal RCPSP scheduling solutions. Initial mathematical formulations created to solve the RCPSP in the early 1960s conceded this position as well. Early MILP RCPSP research concluded that exact analytic techniques were impractical due to the “enormous number of possible schedules” and the improbability of completely enumerating all of these potential schedules “pending further progress in linear programming techniques” (Wiest J. D., 1967).

Substantial improvements in computing technology and optimization software over the last half century have reduced solution times for many previously unsolvable RCPSP instances to under a few seconds using linear programming. These improvements in technology have led to new LP approaches to solve the project scheduling problem under resource constraints such as Bianco and Caramia (2013), Christofides, Alvarez-Valdez, and Tamarit (1987), Koné, Artigues, Lopez, and Mongeau (2011), and Kyriakidis, Kopanos, & Georgiadis (2012). Technological improvements have also increased the viability of using linear programming to generate optimal project schedules in many real world instances. Powerful software programs such as IBM ILOG CPLEX (2015) and Gurobi Optimizer (2015) are now used to solve very large real world problems with millions of constraints and variables (Gurobi Optimization, 2015).

Even with these advances, a set of large problems is still too computationally challenging to solve with linear programming; however, frequent hardware and software improvements continue to increase the size and complexity of RCPSPs that can be solved using this method. The number of constraints and variables needed to model the real world test data considered in this thesis remain well within the existing hardware and software limitations for exact MILP solutions.

3.3.1 Linear Programming Formulations

Multiple linear programming models have been proposed to solve the RCPSP or one of its variants. Koné *et al.* (2011) evaluate the performance of several different LP formulations including discrete time (DT), disaggregated discrete time (DDT), flow-based continuous time (FCT), start/end event (SEE) and On-Off event (OOE). Their study concluded that while there is no dominant LP formulation to find exact solutions with a commercial solver, the superior linear relaxation of the DDT formulation makes it the best option for solving problems with relatively short time horizons (2011). The time windows for the problem instances considered in this thesis are well represented by discrete time blocks and would be considered relatively small, therefore the DDT formulation represents the best currently known option to solve these problems. The small number of predecessor restrictions in typical work periods allows for project networks with low complexity that also makes this problem type well suited for the DDT formulation.

3.3.2 Existing Software Limitations

Project managers, project leaders and schedulers within the fleet maintenance facility generally rely on commercial project management software in conjunction with activity information stored in spreadsheets to plan and devise schedules for upcoming project managed work periods. These software packages may also be used to provide data to support project planning and resource allocation decisions. Project management software planning methods are proprietary and limited information is provided about the quality of the calculated schedule.

Project professionals using commercial software have limited ability to verify whether the provided schedule is an exact schedule or is unnecessarily longer than the optimal duration. Several studies discuss the low quality of software provided solutions for varying resource based objectives including Baumann and Trautmann (2015), Kastor and Sirakoulis (2009) and Kolisch (1999). Each of these studies finds significant gaps between optimal solutions and those provided by commercial project management software.

The results presented by Kolisch evaluated the resource allocation capabilities of earlier versions of popular project management software using a large group of test problems. Table 5 lists the mean percentage difference between the optimal solution and the solution found by the software package for three of the more popular trialed software packages. These results suggest that adapting schedules developed with project management software will lead to longer than necessary project durations when compared to exact solutions.

Table 5: Mean and standard deviation of the percentage difference between optimal and project management software solutions

Project Management Software Package	μ	σ
Microsoft Project	5.35	6.53
Primavera Project Planner	4.39	6.04
Project Manager Workbench	6.69	8.60

Kastor and Sirakoulis test the quality of resource leveling capabilities in project management software programs using information from two real world construction projects (2009). With only two test instances, their sample size is too small to draw significant conclusions but their results show how proprietary software priority heuristics can give a wide range of solutions depending on which software program or priority rule the user selects. The schedule durations provided by the software package and the corresponding priority rule are presented in Table 6.

Table 6: Schedule duration using project management software and real world project data

Priority Rule	Duration	
	1st Instance	2nd Instance
Microsoft Project Standard	744	314
Primevara 6 Options		
-Latest Start Time (LST)	709	308
-Positional Weighted Method (PWM)	744	319
-Late Finish Time (LFT)	744	319
-Enhanced Positional Weighted Method (EPWM)	823	308
-Minimum Slack (MSLK)	823	327
-Shortest Processing Time (SPT)	893	336

Baumann and Trautmann completed the most extensive project management software testing in the literature in their evaluation if resource allocation capabilities for popular software packages using the project scheduling problem library (PSPLIB), a large set of test problems, each containing 30, 60 or 120 activities. The mean makespan deviation percentages compared with the optimal or best known durations are listed in Table 7 as published by Baumann and Trautmann (2015). This study includes up to date versions of the most popular project manamgnent software tools currently in use by project managers. The study results show that scheduling using standard project management software will give later finish dates than the finish date of an optimal schedule.

Table 7: Mean and variance of relative makespan deviation

Project Management Software Package	Mean	Variance
Microsoft Project 2010	9.54	111.33
Primavera P6[2]	5.69	48.84
Microsoft project 2013	16.40	299.27

The basic heuristics used by these software packages to handle resource leveling, allocation and utilization are not competitive in comparison with schedules found using exact solutions. The deviation between optimal and provided schedule durations increases further as the number of activities and resource scarcity increase. These poor results suggest commercial software packages are limited in their ability to generate efficient resource constrained project. Exact solutions found using an MILP model would not have these efficiency issues or the unnecessary schedule buffers inserted by proprietary heuristics.

3.4 Experimentation

In response to the scheduling variability generally observed throughout the naval maintenance program, the primary experimental objective is to develop scheduling policies capable of absorbing unexpected change with as little disruption as possible. As described previously, scarce resources are often costly to reallocate and may cause cascading scheduling conflicts to other concurrent activities. Robust schedules avoid these rescheduling costs and reduce the burden of deconflicting large interrelated projects.

Demeulemeester and Herroelen (2002) were the first authors to present schedule robustness experimentation based on Goldratt's critical chain and buffer management (CC/BM) strategy (1997). Their experiments evaluated the effectiveness of CC/BM as a robust schedule mechanism using a test set of 110 problems and measuring the effects of randomly disturbing activity durations. This thesis uses a similar experimentation procedure to evaluate the robust scheduling policies.

More recently, Hartmann discussed the RCPSP extension of resource capacities and requests varying with time (2015). To analyze this problem further, Hartmann modified the PSPLIB test

---

[2] Results assume the user is able to select the best priority rule for the corresponding activity information

set based on the ProGen instances developed by Kolisch and Sprecher (1996) by creating variable resource availabilities from period to period. Table 8 lists the number of resource availability disruptions (the number of periods with a decrease in available resources) occurring in a subset of these RCPSP/t instances. To allow for reproducible results and maintain a level of continuity with current academic approaches to this problem, we continue to use the time varying RCPSP instances (RCPSP/t) created by Hartmann in our experimentation. Hartmann's first four RCPSP/t instance sets failed to include a meaningful number of resource disruptions and were excluded from our experimentation procedure.

Table 8: Resource availability disruptions in the tested RCPSP/t instances

PSPLIB Instance		Number of resource disruptions	Percent of makespan affected (%)
5	1	3	5.66
	2	5	4.88
	3	9	8.57
	4	4	4.26
	5	7	7.89
6	1	2	3.39
	2	2	3.39
	3	1	2.08
	4	2	4.76
	5	21	20.90
7	1	2	3.64
	2	5	9.52
	3	4	9.52
	4	4	7.84
	5	2	4.55
8	1	4	9.09
	2	3	5.88
	3	5	9.43
	4	4	8.33
	5	2	3.45
9	1	7	6.86
	2	5	4.50
	3	3	4.41
	4	6	6.12
	5	5	5.95
10	1	3	5.88
	2	4	7.14
	3	2	3.23
	4	5	8.62
	5	1	2.44

For each experiment iteration, a PSPLIB test instance was formatted into readable input data using Microsoft Excel VBA macros and transferred to GUSEK v0.2 (GLPK Under Scite Extended Kit) (2008). GUSEK was used to build and compile each instance that was then passed to Gurobi Interactive Shell 6.5.0 for optimization under an academic license (2015).

3.4.1 Experimentation Procedure

The availability of project data with sufficient detail to accurately model short work periods for the naval maintenance periods under study was quite limited. The most significant challenge to precise data collection was ambiguity over when activities were finished on the ship in comparison to when the work centres had time to input their completion into the data management system. Under the current work conditions, there is understandably less priority placed on accurately recording project data than on ensuring activities are completed.

With limited access to detailed project management records, Kolisch's standard PSPLIB instances are used as reasonable representations of what would be expected in real world projects (1996). Each of the tested instances are solved to minimize the makespan ($\min C_{max}$) objective described in Section 0 and each activity is scheduled as early as possible within this shortest possible completion time. The schedule buffering strategies are evaluated using a consistent buffer size of $\min C_{max} + 10\%$. Each of the buffered or robust schedules is then compared to the corresponding RCPSP/t instance to determine its effectiveness in absorbing unexpected disruptions. To simulate the effects of uncertainty, the RCPSP/t schedule cannot reschedule tasks sooner than they were planned by the initial baseline schedule. This constraint also represents real world challenges related to material just in time lead times and the difficulty of reallocating scare resources.

The three buffering strategies considered in these experiments are simple buffering, periodic buffering and weighted buffering. The following paragraphs provide details on each of these approaches.

3.4.1.1 Simple buffering

The first buffering strategy is to hedge against potential delays using the straightforward approach of adding on a $100 * \delta\%$ time buffer to the end of the $\min C_{max}$ baseline. The project instance is then solved using the RCPSP/t time-varying resource profile generated by Hartmann. An immediate drawback of this approach is only the activities scheduled at the end of the project can be completed without delaying their start times. On the positive side, the completion date of the tightly planned RCPSP/t $\min C_{max}$ schedule is more likely to be earlier than other buffered schedules. The window for disruption is also shorter using this approach. If the first project disruptions occur during the buffered period at the end of the project, none of the project activities will be affected since they will have already finished.

Applying this simple buffering approach to the example problem cited earlier with give the schedule shown in Figure 12. For this network the following simplified precedence constraints are now enforced: <2, 4, 6, 8> and <3, 5, 7, 9>.

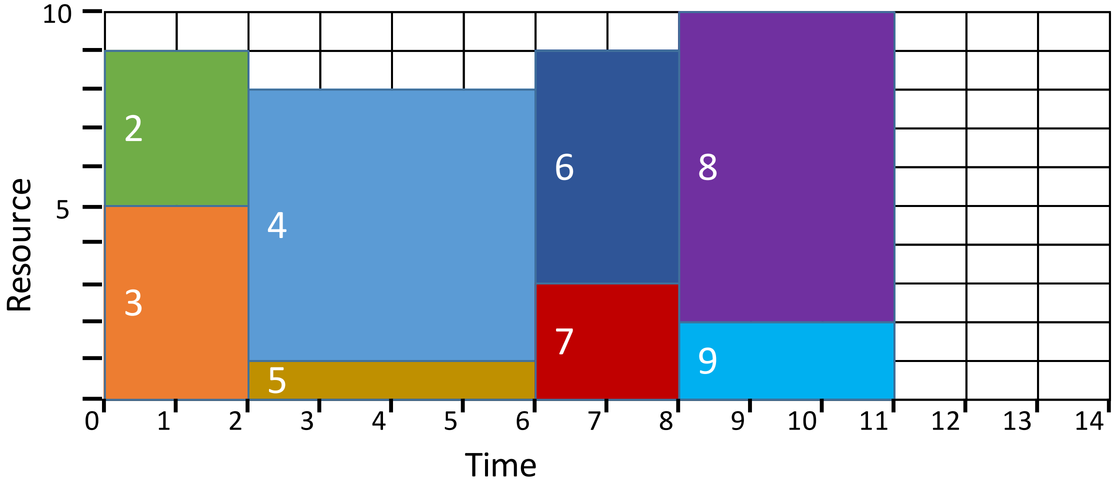

Figure 12: Simple buffering approach for an example project network

3.4.1.2 Periodic Buffering

The second buffering approach is to move the $100 * \delta\%$ buffer from the end of the project makespan MS to distributed points in time throughout the project schedule. The points tested for these experiments are at ¼, ½ and ¾ of the shortest makespan plus $100 * \delta\%$ but may be placed at any position throughout the makespan by modifying the elements of set PB according to the scheduler's preference. These quarterly buffers were implemented into the schedule using the objective shown in Eq. (3.3). This objective effectively reserves resource capacity during quarterly periods. Equation (3.4) constrains the schedule within the buffered timeline. Adding Equations (3.5), (3.6) and (3.7) as activity completion, precedence and resource availability constraints described in the RCPSP model from Section 0 completes the periodic buffering model.

Periodic Buffering Model

Indices:
$0 \dots J+1$ for activities
$1 \dots K$ for resource types
$0 \dots D$ for time horizon

Sets:
P_j immediate predecessors for activity j
$\boldsymbol{K}$ resources
$\boldsymbol{J}$ activities
PB buffer positions

Parameters:
D integer, periods in the planning horizon
J integer, number of activities
d_j integer, duration of activity 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
MS integer, optimal makespan if no buffers are used
δ percentage buffer added to the optimal makespan

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

$$minimize \sum_{j=1}^{J}\sum_{k=1}^{K}\sum_{t=\max\{round((1+\delta)*MS)-d_j+1,ES_j\}}^{\min\{LS_j,round((1+\delta)*MS)*PB\}} r_{jk}x_{jt} \quad \forall\, PB \in \left(\frac{1}{4},\frac{1}{2},\frac{3}{4}\right) \qquad 3.3$$

$subject\ to$

$$\sum_{t=ES_J}^{LS_J} t * x_{Jt} \leq (1+\delta) * MS \qquad 3.4$$

$$\sum_{t=ES_j}^{LS_j} x_{jt} = 1 \qquad \forall\, j \in \boldsymbol{J} \qquad 3.5$$

$$\sum_{b=t}^{LS_j} x_{jb} + \sum_{b=ES_i}^{\min\{LS_i,t+d_j-1\}} x_{ib} \leq 1 \qquad \forall j \in \boldsymbol{J}, i \in P_j \qquad 3.6$$

$$\sum_{j=1}^{J}\sum_{b=\max\{t-d_j+1,\, ES_j\}}^{\min\{LS_j,t\}} r_{jk} \cdot x_{jb} \leq R_k \qquad \forall\, k \in \boldsymbol{K}, t \in D \qquad 3.7$$

$$x_{jt} \in \{0,1\} \qquad \forall\, j \in \boldsymbol{J}, t \in ES_j \ldots LS_j \qquad 3.8$$

Applying this periodic buffering approach model to the same example tested using simple buffering will generate the project schedule shown visually in Figure 13.

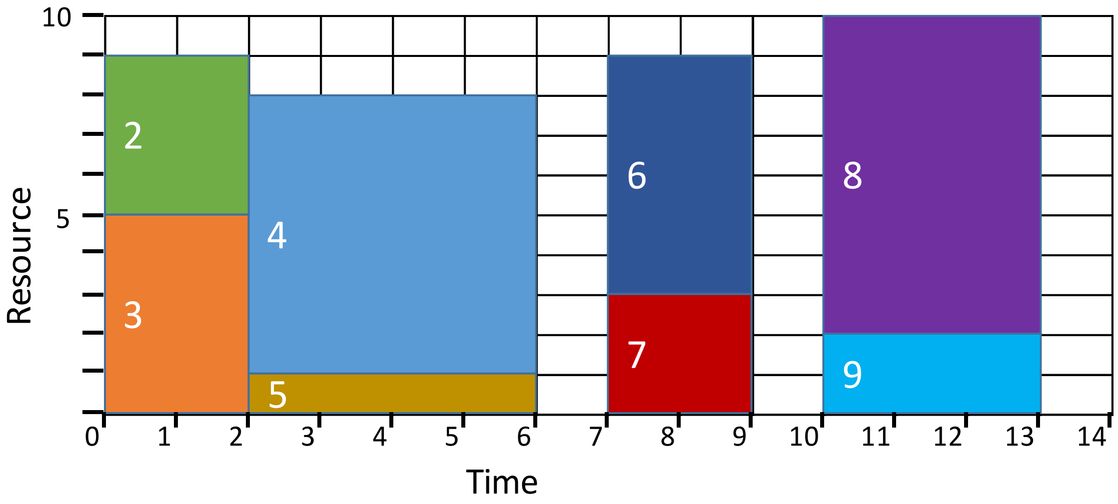

Figure 13: Periodic buffering approach for an example project network

3.4.1.3 Weighted Buffering

The final buffering policy tested uses activity duration and demand to place buffers after high capacity, long duration activities which are more disruptive to the schedule when moved than shorter, lower capacity activities. This policy was implemented into the RCPSP model using a positive integer variable z_{ij}, defined by Eq. (3.16), for each activity and its corresponding predecessors. This variable is used in Equations (3.10) and (3.13) to determine the length of the buffer placed between activity i and each of its j predecessors from the set P_j.

This policy also requires a weighting parameter B_j to determine which activities will require the most resources. This parameter is calculated using Eq. (3.9), which combines the expected duration and total resource demand of each activity by multiplying the duration of the activity d_j by its total resource demand r_{jk} across all resources. This procedure is similar to the one described in Section 3.2.1.

$$B_j = d_j \sum_{k=1}^{K} r_{jk} \qquad \forall\, j \in \{1..J\} \qquad 3.9$$

This weighting parameter could also be further modified to account for additional factors such as activity priority or cost to reschedule if that information is available to the decision maker.

Weighted buffers are inserted into the schedule using the objective from Eq. (3.10), modifying the standard RCPSP precedence constraint as shown in Eq. (3.13) and maintaining the makespan restriction from Eq. (3.4), now labelled Eq. (3.11) below. Adding these three equations to the activity completion constraint and the resource availability constraint described in previous models produces a solution that buffers the schedule following lengthy activities with high resource demand.

Weighted Buffering Model

Indices:
$0 \dots J+1$ for activities
$1 \dots K$ for resource types
$0 \dots D$ for time horizon

Sets:
P_j immediate predecessors for activity j
$\boldsymbol{K}$ resources
$\boldsymbol{J}$ activities

Parameters:
D integer, periods in the planning horizon
J integer, number of activities
d_j integer, duration of activity 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
MS integer, optimal makespan if no buffers are used
δ percentage buffer added to the optimal makespan
B_j integer, activity weighting parameter

Variables:
x_{jt}, binary, 1 if activity j starts at time t, 0 otherwise
z_{ij}, positive integer

$$maximize \sum_{j=1}^{J} z_{ij} B_j \qquad \forall\, i \in P_j \qquad 3.10$$

$$\sum_{t=ES_J}^{LS_J*(1+\delta)} t * x_{Jt} \le (1+\delta) * MS \qquad 3.11$$

$$\sum_{t=ES_j}^{LS_j} x_{jt} = 1 \qquad \forall\, j \in \boldsymbol{J} \qquad 3.12$$

$$\sum_{t=ES_j}^{LS_j} t \cdot x_{jt} \ge z_{ij} + \sum_{t=ES_i}^{LS_i} (t + d_i) \cdot x_{it} \qquad \forall\, j \in \boldsymbol{J}, i \in P_j \qquad 3.13$$

$$\sum_{j=1}^{J} \sum_{b=\max\{t-d_j+1,\, ES_j\}}^{\min\{LS_j, t\}} r_{jk} \cdot x_{jb} \le R_k \qquad \forall\, k \in \boldsymbol{K}, t \in D \qquad 3.14$$

$$x_{jt} \in \{0,1\} \qquad \forall\, j \in \boldsymbol{J}, t \in ES_j \dots LS_j \qquad 3.15$$

$$z_{ij} \in \boldsymbol{Z}^{+} \qquad \forall\, j \in \boldsymbol{J}, i \in P_j \qquad 3.16$$

Depending on the activity information and limits set on z_{ij}, these buffers may be spread across several activities or following a single, very large activity along all of the project's critical paths. Figure 14 shows a graphical representation of the sample project schedule using the same project network as simple and periodic buffering.

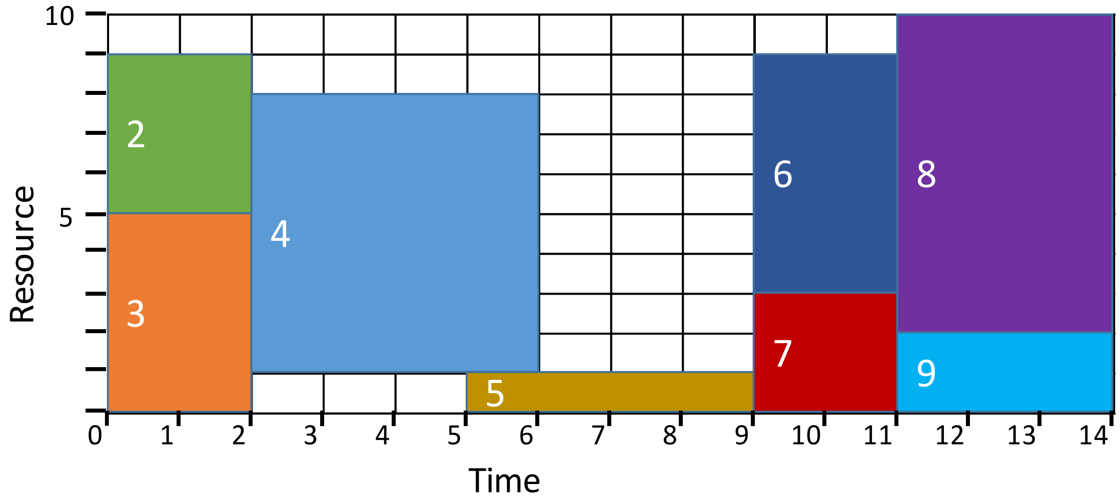

Figure 14: Weighted buffering approach for an example project network

Using this weighted buffering procedure to protect baseline schedule quality has not previously been discussed in the literature as far as the author is aware.

3.4.1.4 Reduced quality modes

In addition to improving schedule robustness, we also considered the option of using multiple activity execution modes to provide a degree of performance robustness. Rather than having to remove an activity from the project portfolio or delay the project completion time to free up resources in response to an unexpected disruption, we can consider an alternate, lower quality way to complete the activity. The advantage of this secondary execution mode is that it has a shorter duration which represents a real world equivalent of a temporary or partial repair that is deemed sufficient in the short term such that it can delay the required more substantial repair to a time period beyond the immediate operational horizon.

In the naval maintenance environment, this type of short-term repair would require a thorough risk assessment from the Fleet Technical Authority (FTA) and an approved waiver prior to being accepted. To account for the reduced quality of this repair mode and the increased risk assumed when choosing it, a random increase (50-150%) in resource demand is incurred each time a lower quality execution mode is chosen. A random representative increase in demand is considered reasonable in this case since practically assessing the true cost of a reduced quality repair for naval maintenance work is not well quantified.

The current library of multimode PSPLIB problems (multimode resource constrained project scheduling problem, MRCPSP) does not correspond with the RCPSP and RCPSP/t instances used in our experimentation and are not used to evaluate the multimode component. Instead, we add randomly execution modes to the previously used RCPSP/t instances to create a MRCPSP/t subset. The weighted buffering scenario can then be re-evaluated using two different sets of these MRCPSP/t instances to determine the magnitude of any schedule or performance improvements.

The multimode component of our experimentation used VBA scripting to assign a secondary mode to 10 or 20% of the activities chosen at random from each of the RCPSP/t instances tested previously. Each secondary mode was given a non-zero duration that is reduced from the activity's original duration by a factor of its overall resource usage percentage. An activity with a duration of 8 and a resource demand of 20 of the 40 available resources will have a reduced duration of 4 $(8 - 8 * 20/40 = 4)$ if the secondary mode is selected. The trade-off for this reduced duration is a random increase in resource cost between 50 and 150% of the original demand. Adding a second mode to activities with an original duration of 1 time unit does not improve the solution since this mode cannot offer a reduced duration trade-off for the increased resource demand.

MRCPSP/t Instance Creation Process

1. Randomly select 6 of 30 activities and give each once a second mode
2. For each of the six activities:
    a. reduce the duration of this secondary mode to max{original duration less its overall resource usage factor, 1}
    b. increase the resource demand to a random value between 50 and 150% of the initial demand

3.4.2 Scenario Evaluation

Each of the buffering strategies was evaluated for scheduling quality and schedule lateness using the baseline buffered schedule and the resulting time-varying resource schedule. Schedule quality, the number of days each activity is delayed, was measured using Eq. (3.17) which calculates the sum of the difference between actual individual activity start times (AST) and baseline activity start times (BST). A smaller schedule quality value indicates that the project could be executed similarly to the baseline plan despite unexpected changes to the initial project information. Total schedule lateness was measured using the difference between the actual and planned project completion date as shown in Eq. (3.18).

$$Schedule\ Quality = \sum_{j=1}^{J+1} \left(AST_j - BST_j\right) \qquad 3.17$$

$$Total\ Schedule\ Lateness = AST_{J+1} - BST_{J+1} \qquad 3.18$$

Chapter 4: Results and Discussion

Each of the three buffering approaches are evaluated using the schedule quality, activities delayed and schedule lateness metrics when subjected to time-varying resource availability. The schedule quality for all three approaches is displayed graphically in Figure 15, which orders the test instances from the least amount of days delayed to the largest number of days delayed when using the simple buffering approach. Each of the Hartmann RCPSP/t instances is subject to a varying amount of disruption occurring at random intervals throughout the project schedule (2015). Individual results for each instance in the three test sets used for this research can be found in Appendix A.

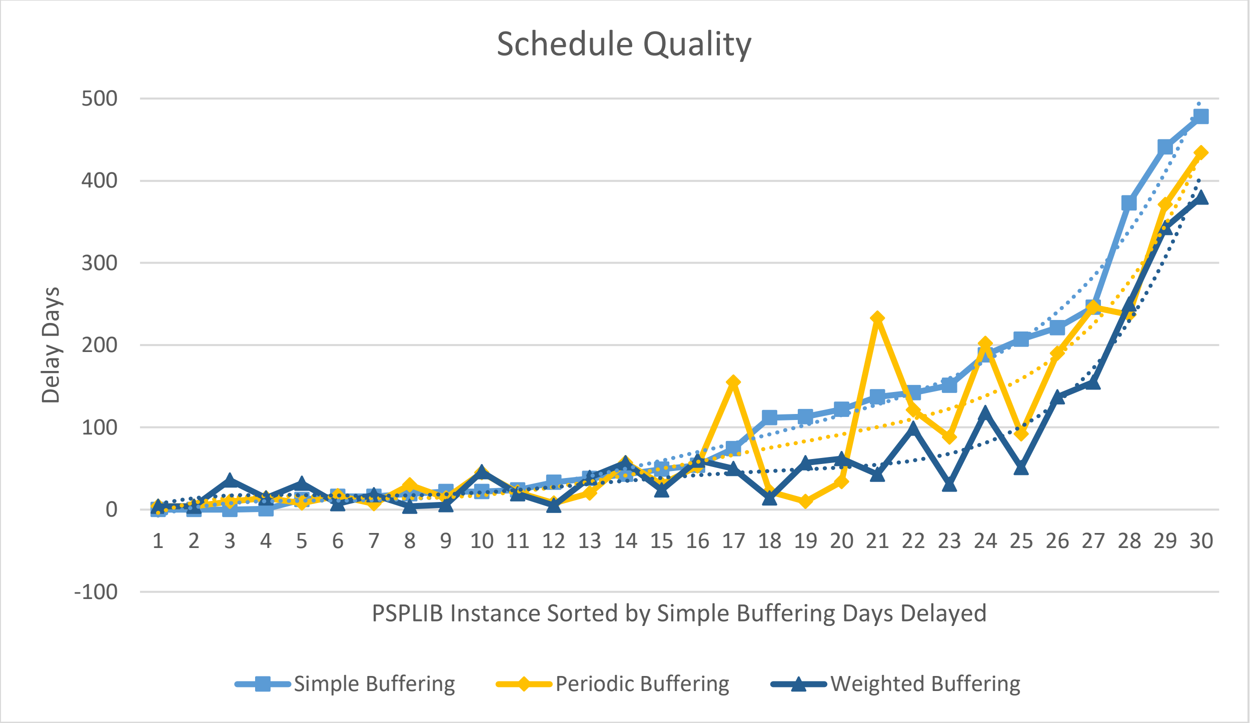

Figure 15: A comparison of schedule quality using simple, periodic and weighted buffering

All three approaches result in a low number of delay days for the first 15 project instances shown in Figure 15. Each of these 15 instances had a low number of disruptions and all three approaches are able to finish within the $min\ Cmax + 10\%$ buffer without extensions or activity crashing decisions. This result is intuitive since deterministic schedules require no buffering and perform well with known project information. This is verified using a regression analysis between the number of resource disruptions and the resulting number of days delayed. The coefficient of determination for this linear regression gives an R^2 value of 0.57, which suggests there is a positive relationship between the frequency of project disruptions and the number of days each project activity is delayed. In terms of scheduling quality, timelines with more disruptions tend to perform much worse using simple buffering techniques when compared to the other buffering options.

Table 9 shows the average number of delay days, delayed jobs and lateness for all activities when using each of the buffering approaches. Simple buffering delays 12.3 activities per project (41.0%) for an average total of 111.8 delay days. When using periodic buffering, the average number of delay days is 92.60 spread across an average of 10.53 activities (35.1%). Buffering a project schedule using the weighted method gives 72.17 delay days and 9.833 delayed jobs (32.8%) on average. From these results, the periodic buffering approach tends to perform at a level between simple and weighted buffering with comparable lateness values to weighted buffering.

Table 9: Average metrics for each buffering approach

Buffering Approach	Delay Days	Delayed Activities	Total Schedule Lateness
Simple	111.8	12.30	5.50[3]
Periodic	92.60	10.53	8.70
Weighted	72.17	9.833	7.97

The information from Table 9 is also visually summarized in Figure 16 on the following page.

---

[3] This result counts projects that finish early as finishing exactly on time due to the assumption that finishing early offers no benefit to the ship or maintenance facility when operational time windows are predetermined months in advance.

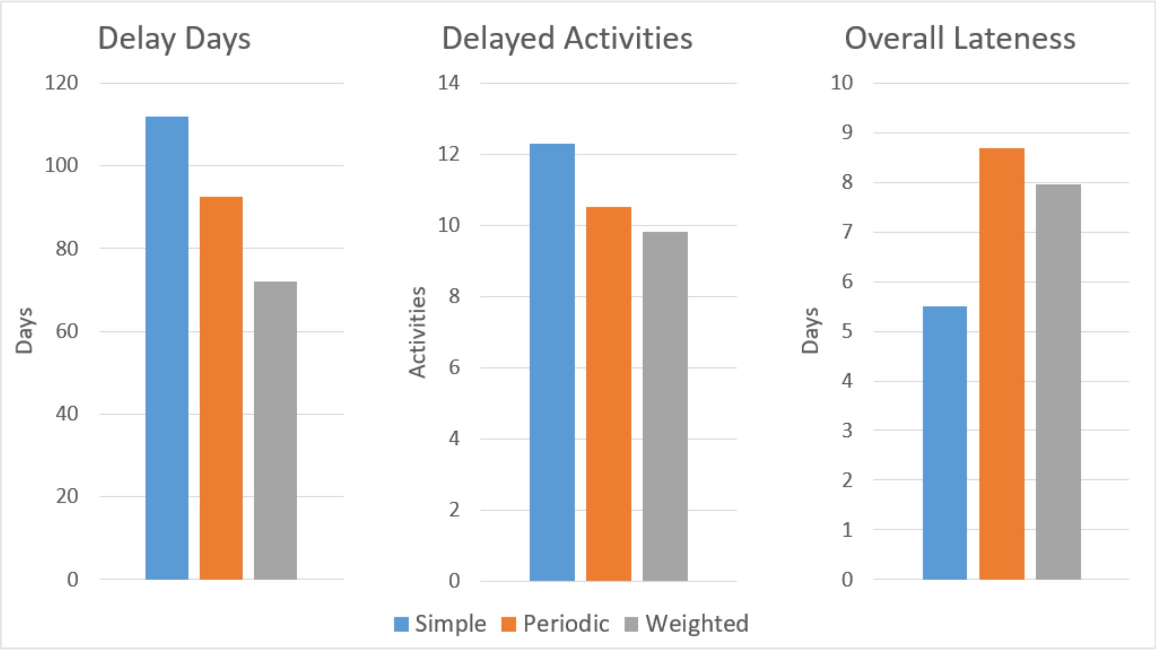

Figure 16: A comparison of simple, periodic and weighted buffering using delay days, the number of delayed activities and overall project lateness

Weighted schedule buffering is the only buffering approach considered here which attempts to use some of the known project characteristics (specifically activity duration and resource demand) to determine where buffers are placed. As a result, the scheduling quality achieved using this method is improved for projects with a high frequency of disruption compared to both simple and periodic buffering approaches. The first 15 data points shown in Figure 15 highlight the similarity between all three approaches when applied to projects experiencing a lower level of variation of 50 or fewer delay days. When the frequency of disruption increases above 50 delay days, weighted buffering consistently performs better than simple buffering and has fewer total delay days than periodic buffering in 11 out of 15 (73.3%) of these instances.

A paired t-test comparing the delay days resulting from each of the three buffering methods rejects the hypothesis that all three approaches are equal using a 5% level of significance. An alternate hypothesis of periodic buffering delay days < simple buffering delay days results in a test statistic of 0.026 and for weighted buffering delay days < simple buffering the test statistic is 0.0001. The test set confirming fewer delay days when using weighted over periodic buffering is 0.014.

When considering the number of activities delayed after disruption, simple buffering delays a greater number of jobs than both periodic and weighted buffering with t-test statistics of 0.036

and 0.0097 respectively. The difference between periodic and weighted buffering delayed jobs cannot be confirmed at a statistically significant level.

The schedule lateness values suggest that simple buffering remains the best approach for finishing the project as close as possible to the desired project finish date. This result should be expected since no buffers go unused until the very end of the project when the effects of the delays are least disruptive. Simply buffered projects also have a more compressed time window compared to periodic and weighted buffering. This means simple buffering has fewer periods that have the potential to be disrupted.

The weighted buffering strategy was also tested in conjunction with multiple activity execution modes to determine their impact on scheduling quality. The initial testing approach was to randomly select 10% of the project's 30 activities and give them a crashed or reduced quality execution mode with a shorter duration but higher resource cost. After testing the multimode option on the first 20 instances and noticing almost no effects, we doubled the number of activities with dual execution modes in the MRCPSP/t instances from 10 to 20%. This increased the chance that one of the disrupted activities would have a contingency execution mode. It was important to control the percentage of activities that can be executed in more than one mode to model the real life condition that only a limited number of project activities can be completed in more than one way. The final ten instances were tested with an increased number of multimode activities and the improvements were significant.

In terms of improved schedule lateness, a secondary execution mode for 20% of the project activities shortened project lateness by an average of 3.0 days (55.6%) in comparison to weighted buffering alone. Scheduling quality was also improved by an average of 13.7 days (26.3%) for this subset of instances. An instance by instance comparison of the schedule quality generated by these two models is shown in Figure 17.

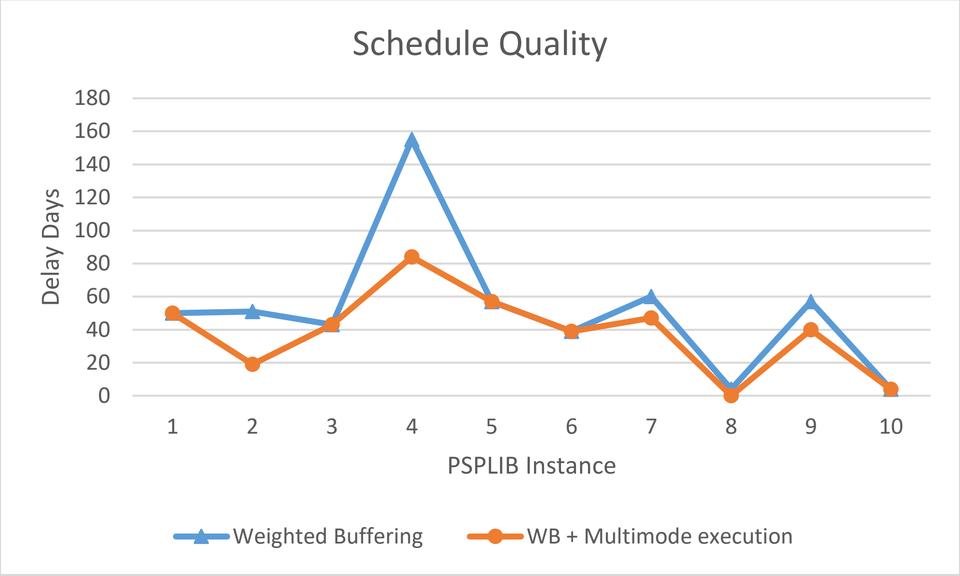

Figure 17: Schedule quality using weighted buffering and weighted buffering with multiple activity execution modes

Evaluating each of these schedule buffering policies highlights the importance of taking account of potential disruptions during the scheduling process. This is especially true for project environments where high rescheduling costs are incurred when the schedule breaks down. Strategically locating time and resource buffers provide significant improvements in terms of schedule quality, and when combined with multiple execution modes, can improve project lateness.

In project industries where significant cost penalties are associated with schedule modifications, it may be more important to prioritize schedule quality over schedule lateness. A penalty for completing the project late may be less than the cost required to recover a severely disrupted schedule and still finish on time. An assumption made in this research is that there are project conditions where significantly delaying individual activities beyond their baseline start date can be costlier in certain situations than delaying the overall project completion date to better adhere to the initial schedule. In this case, the project manager should consider schedule robust approaches such as the suggested buffering methods and secondary execution modes.

Chapter 5: Conclusion

Maintaining acceptable schedule quality without compromising project lateness is a challenging trade-off for project managers working in uncertain environments. The work presented in this thesis discusses the sources and levels of uncertainty present in the naval maintenance environment. A summary of the project management process within a Canadian naval dockyard is also given. A review of the literature related to project scheduling is also included to give an overview potential solution approaches.

This thesis analyzed three buffer placement approaches to improve schedule quality for naval maintenance projects; simple buffering, periodic buffering and weighted buffering. The first approach involved developing a mathematical model to represent the existing scheduling policy of simplistic makespan buffering employed at a naval maintenance facility. The second schedule buffering approach considered periodic buffer placement to place reserve capacity at regular intervals throughout the makespan. The final buffering approach introduced a novel weighted buffering approach that allocated buffers after the activities with the largest cumulative resource demand.

All three buffering models were designed to represent project situations with predetermined due dates and multiple resource types. These models were then expanded to solve for varying resource demand. Each model was used to buffer a common series of computer generated project schedules from the Project Scheduling Library. These models were then solved a second time using time-varying resource disruptions from Hartmann's RCPSP/t problem subset (2013). Comparing the project schedules before and after disruption showed the effectiveness of each approach on schedule quality, the number of disrupted activities and schedule lateness.

Additional activity execution modes were also added to the weighted buffering approach to test their effect on schedule stability. A secondary execution mode can be used to represent a lower quality, temporary repair or an activity crashing decision where additional resources are used to reduce activity duration.

A resource availability parameter is introduced to narrow the solution space and to identify key points in the schedule where buffers should be placed using the weighted approach. Preserving schedule quality using activity priority levelling was also discussed.

Experimental results confirm that simple makespan buffering is the best approach to protect against overall project lateness but performs poorly in terms of schedule quality. Using periodic buffering in project schedules provides improved schedule quality but also increases project lateness. The weighted buffering approach gives the best schedule quality results but has comparable increases in project lateness with periodic buffering.

Results from this thesis can be used to substantiate consideration for strategic buffer positioning to preserve schedule quality in project environments, such as the naval maintenance industry, where project scope is subject to substantial uncertainty. If significant costs are required to recover schedule disruptions, the option of trading off project lateness for improved schedule robustness should be considered.

Further efforts to measure the cost benefit of implementing strategic buffering options into naval maintenance schedules would be worthwhile to consider in addition to the cost trade-off decision that is made between maintaining the project due date and efficient resource utilization. Future research into the relationship between buffer size and the expected level of uncertainty would strengthen the results presented here. Determining a quantifiable trade-off value between reduced quality execution modes and project lateness would also be an interesting topic to study.

Bibliography

Ahuja, H. N. (1976). *Construction performance control by networks.* New York; Toronto: Wiley.

Archer, S. (2008). *Stochastic resource constrained project scheduling with stochastic task insertion problems.* (Doctoral dissertation). Ann Arbor: ProQuest.

Artigues, C., & Roubellat, F. (2000). A polynomial activity insertion algorithm in a multi-resource schedule with cumulative constraints and multiple modes. *European Journal of Operational Research, 127*(2), 297-316. doi:10.1016/S0377-2217(99)00496-8

Artigues, C., Leus, R., & Nobibon, F. T. (2013). Robust optimization for resource-constrained project scheduling with uncertain activity durations. *Flexible Services and Manufacturing Journal, 25*(1-2), 175-205. doi:10.1007/s10696-012-9147-2

Aytug, H. L., McKay, K., Mohan, S., & Uzsoy, R. (2005). Executing production schedules in the face of uncertainties: A review and some future directions. *European Journal of Operational Research, 161*(1), 86-110.

Bandelloni, M., Tucci, M., & Rinaldi, R. (1994). Optimal resource leveling using non-serial dyanamic programming. *European Journal of Operational Research, 78*(2), 162-177.

Baumann, P., & Trautmann, N. (2015). Resource-constrained project scheduling with project management information systems. In *Project Management and Scheduling Vol. 2* (pp. 1385-1400). Springer International Publishing.

Ben-Tal, A., Ghaoui, L. E., & Nemirovski, A. (2009). *Robust optimization.* Princeton: Princeton University Press.

Bertsimas, D., & Sim, M. (2003). Robust discrete optimization and network flows. *Mathematical programming, 98*(1), 49-71. doi:10.1007/s10107-003-0396-4

Bettoni, L. M. (2008). *GUSEK (GLPK Under Scite Extended Kit)*. Retrieved from http://gusek.sourceforge.net

Bianco, L., & Caramia, M. (2013). A new formulation for the project scheduling problem under limited resources. *Flexible Services and Manufacturing Journal, 25*(1-2), 6-24.

Blazewicz, J., Lenstra, J. K., & Kan, A. R. (1983). Scheduling subject to resource constraints: classification and complexity. *Discrete Applied Mathematics, 5*(1), 11-24.

Boyle, G., Little, J., Manning, J., & van der Krogt, R. (2011). A constraint-based approach to ship maintenance for the irish navy. *Proceedings of the Irish Transport Research Network Conference (ITRN-11).* Cork.

Brucker, P., Drexl, A., Möhring, R., Neumann, K., & Pesch, E. (1999). Resource-constrained project scheduling: Notation, classification, models, and methods. *European Journal of Operational Research, 112*(1), 3-41.

Burgess, A. R., & Killebrew, J. B. (1962). Variation in activity level on a cyclical arrow diagram. *Journal of Industrial Engineering, 13*(2), 76-83.

Certa, A., Galante, G., Lupo, T., & Passannanti, G. (2011). Determination of Pareto frontier in multi-objective maintenance optimization. *Reliability Engineering and System Safety, 96*, 861-867. doi:10.1016/j.ress.2010.12.019

Christofides, N., Alvarez-Valdez, R., & Tamarit, J. M. (1987). Project scheduling with resource constraints: A branch and bound approach. *Nicos Christofides, 29*(3), 262-273.

Coelho, J., & Vanhoucke, M. (2011). Multi-mode resource-constrained project scheduling using RCPSP and SAT solvers. *European Journal of Operational Research, 213*(1), 73-82.

Colak, E., & Azizoglu, M. (2013). A resource investment problem with time/resource trade-offs. *Journal of the Operational Research Society, 65*(5), 777-790.

de Boer, R., Schutten, J. M., & Zijm, W. (1997). A decision support system for ship maintenance capacity planning. *CIRP Annals-Manufacturing Technology, 46*(1), 391-396.

Deblaere, F., Demeulemeester, E., & Herroelen, W. (2011). Proactive policies for the stochastic resource-constrained project scheduling problem. *European Journal of Operational Research, 214*(2), 308-316. doi:10.1016/j.ejor.2011.04.019

Deblaere, F., Demeulemeester, E., & Herroelen, W. (2011). Reactive scheduling in the multi-mode RCPSP. *Computers & Operations Research, 38*(1), 63-74. doi:10.1016/j.cor.2010.01.001

Demeulemeester, E. (1995). Minimizing resource availability costs in time-limited project networks. *Management Science, 41*(10), 1590-1598.

Demeulemeester, E. L., & Herroelen, W. S. (2002). *Project scheduling: a reasearch handbook.* Boston: Kluwer Academic Publishers.

Department of National Defence. (2013). *Defence Renewal Charter.* National Defence.

Easa, S. M. (1989). Resource leveling in construction by optimization. *Journal of Construction Engineering and managment, 115*(2), 302-316.

Fleet Maintenance Facility Cape Scott. (2013, March 18). Project Planning and Implementation. *Naval Engineering and Maintenance Instruction*.

Gilmore, S. (2015, August 4). *The sinking of the Canadian Navy.* Retrieved from http://www.macleans.ca/news/canada/the-sinking-of-the-canadian-navy/

Goldratt, E. (1997). *Critical Chain.* Great Barrington: The North River Press.

Government of Canada. (2015, September 1). *Welcome to the Royal Canadian Navy*. Retrieved from Royal Canadian Navy: http://www.navy-marine.forces.gc.ca/en/index.page

Gurobi Optimization. (2015, September 13). *Gurobi Optimizer*. Retrieved from Gurobi Optimization: http://www.gurobi.com/products/gurobi-optimizer

Gurobi Optimization. (2015, 11 8). *Industries Using Gurobi*. Retrieved from Gurobi Optimization: http://www.gurobi.com/products/industries/industry-overview

Hans, E., Herroelen, W., Leus, R., & Wullinka, G. (2007). Ahierarchical approach to multi-project planning under uncertainty. *The International Journal of Managment Science, 35*, 363-577. doi:10.1016/j.omega.2005.10.004

Hansen, K. (2014, October 2). *Ship cuts send RCN naval ranking tumbling according to their own system*. (Canadian Naval Review) Retrieved from http://www.navalreview.ca/2014/10/ship-cuts-send-rcn-naval-ranking-tumbling-according-to-their-own-system/

Hartmann, S. (2012). *Project scheduling under limited resources: models, methods, and applications.* Springer Science & Business Media.

Hartmann, S. (2013). Project scheduling with resource capacities and requests varying with time: a case study. *Flexible Services and Manufacturing Journal, 25*(1), 74–93. doi:10.1007/s10696-012-9141-8

Hartmann, S. (2015). Time-Varying Resource Requirements and Capacities. In C. Z. Schwindt, *Handbook on Project Management and Scheduling Vol. 1* (pp. 163-176). Springer International Publishing.

Hartmann, S., & Briskorn, D. (2010). A survey of variants and extensions of the resource-constrained project scheduling problem. *European Journal of Operational Research, 207*(1), 1-14.

Hazır, Ö., Erel, E., & Günalay, Y. (2011). Robust optimization models for the discrete time/cost trade-off problem. *International Journal of Production Economics, 130*(1), 87-95. doi:10.1016/j.ijpe.2010.11.018

Herroelen, W., & Leus, R. (2002). Project scheduling under uncertainty: Survey and research potentials. *European journal of operational research, 165*(2), 189-306. doi:10.1016/j.ejor.2004.04.002

Herroelen, W., & Leus, R. (2004). Robust and reactive project scheduling: a review and classification of procedures. *International Journal of Production Research, 42*(8), 1599-1620. doi:10.1080/00207540310001638055

Herroelen, W., & Leus, R. (2005). Idenification of popular misconcepions about project scheduling and time buffering under resource constraints. *Journal of the Operational Research Society, 56*(1), 102-109. doi:10.1057/palgrave.jors.2601813

IBM. (2015, 11 8). *CPLEX Optimizer*. Retrieved from IBM Software: http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/

Józefowska, J., & Węglarz, J. (2006). *Perspectives in modern project scheduling.* New York: Springer.

Kastor, A., & Sirakoulis, K. (2009). The effectiveness of resource levelling tools for resource constraint project scheduling problem. *International Journal of Project Management, 27*(5), 493-500. doi:10.1016/j.ijproman.2008.08.006

Kelley Jr, J. E., & Walker, M. R. (1959). Critical-path planning and scheduling. *Papers presented at the December 1-3, 1959, eastern joint IRE-AIEE-ACM computer conference*, 160-173.

Kolisch, R. (1999). Resource allocation capabilities of commercial project management software packages. *Interfaces, 29*(4), 19-31. doi:10.1287/inte.29.4.19

Kolisch, R., & Sprecher, A. (1996). PSPLIB - A project scheduling problem library. *European Journal of Operational Research, 96*, 205-216.

Kolisch, R., Sprecher, A., & Drexl, A. (1995). Characterization and Generation of a General Class of Resource-Constrained Project Scheduling Problems. *Managment Science, 41*(10), 1693-1703.

Koné, O., Artigues, C., Lopez, P., & Mongeau, M. (2011). Event-based MILP models for resource-constrained project scheduling problems. *Computers & Operations Research, 38*(1), 3-13.

Kreter, S. R., & Zimmermann, J. (2014). The total adjustment cost problem: Applications, models, and solution algorithms. *Journal of Scheduling, 17*(2), 145-160.

Kyriakidis, T. S., Kopanos, G. M., & Georgiadis, M. C. (2012). MILP formulations for single-and multi-mode resource-constrained project scheduling problems. *Computers & Chemical Engineering, 36*, 369-385.

Lamas, P., & Demeulemeester, E. (2014). A purely proactive scheduling procedure for the resource-constrained project scheduling problem with stochastic activity durations. *Journal of Scheduling*, 1-20. doi:10.1007/s10951-015-0423-3

Lambrechts, O., Demeulemeester, E., & Herroelen, W. (2008). Proactive and reactive strategies for resource-constrained project scheduling with uncertain resource availabilities. *Journal of scheduling, 11*(2), 121-136.

Larson, E. W., & Gray, C. F. (2013). *Project Managment: The Managerial Process.* New York: McGraw-Hill Irwin.

Li, M., & Wu, G. (2014). Robust optimization for time-cost tradeoff problem in construction projects. *Abstract and Applied Analysis, 2014*(1), 1-7. doi:10.1155/2014/926913

Malcolm, D. G., Roseboom, J. H., Clark, C. E., & Fazar, W. (1959). Application of a technique for research and development program evaluation. *Operations Research, 7*(5), 646-669.

Mastor, A. A. (1970). An experimental investigation and comparative evaluation of production line balancing techniques. *Management Science, 16*(11), 728-746.

Möhring, R. H. (1984). Minimizing costs of resource requirements in project networks subject to a fixed completion time. *Operations Research, 32*(1), 89-120.

Möhring, R. H., Radermacher, F. J., & Weiss, G. (1984). Stochastic scheduling problems I—General strategies. *Zeitschrift für Operations Research, 28*(7), 193-260. doi:10.1007/BF01919323

Möhring, R. H., Schulz, A. S., Stork, F., & Uetz, M. (2003). Solving Project Scheduling Problems by Minimum Cut Computations. *Managment Science, 49*(3), 330-350. doi:10.1287/mnsc.49.3.330.12737

Neumann, K., & Zimmermann, J. (1999). Resource levelling for projects with schedule-dependent time windows. *European Journal of Operational Research, 117*(3), 591-605.

Neumann, K., & Zimmermann, J. (2000). Procedures for resource leveling and net present value problems in project scheduling with general temporal and resource constraints. *European Journal of Operational Research, 127*(2), 425-443.

Nudtasomboon, N., & Randhawa, S. U. (1997). Resource-constrained project scheduling with renewable and non-renewable resources and time-resource tradeoffs. *Computers & Industrial Engineering, 32*(1), 227-242.

Optimization, G. (2015). *Gurobi Optimizer Reference Manual*. Retrieved from http://www.gurobi.com

Pinha, D., & Ahluwalia, R. (2014). Decision Support System for Production Planning in the Ship Repair Industry. *Industrial and Systems Engineering Review, 2*(1), 25-61.

Pritsker, A. A., Waiters, L. J., & Wolfe, P. M. (1969). Multiproject scheduling with limited resources: A zero-one programming approach. *Management science, 16*(1), 93-108.

Project Management Institute, Inc. (2013). *A guide to the project management body of knowledge (PMBOK® Guide).* Newtown Square: PMI.

Qi, J., Guo, B., Lei, H., & Zhang, T. (2014). Solving resource availability cost problem in project scheduling by pseudo particle swarm optimization. *Systems Engineering and Electronics, 25*(1), 69-76.

Rajendran, C., & Holthaus, O. (1999). A comparative study of dispatching rules in dynamic flowshops and jobshops. *European journal of operational research, 116*(1), 156-170. doi:10.1016/S0377-2217(98)00023-X

Ranjbar, M., Kianfar, F., & Shadrokh, S. (2008). Solving the resource availability cost problem in project scheduling by path relinking and genetic algorithm. *Applied Mathematics and Computation, 196*(2), 879-888.

Rieck, J., Zimmermann, J., & Gather, T. (2012). Mixed-integer linear programming for resource leveling problems. *European Journal of Operational Research, 22*(1), 27-37.

Rodrigues, S. B., & Yamashita, D. S. (2010). An exact algorithm for minimizing resource availability costs in project scheduling. *European Journal of Operational Research, 206*(3), 562-568.

Royal Canadian Navy. (2015, September 1). *MARLANT Units*. Retrieved from Royal Canadian Navy: http://www.navy-marine.forces.gc.ca/en/about/structure-marlant-units.page

Sankaran, J. K., Bricker, D. L., & Juang, S.-H. (1999). A strong fractional cutting-plane algorithm for resource-constrained project scheduling. *International Journal of Industrial Engineering*, 99-111.

Schutt, A., Feydy, T., Stuckey, P. J., & Wallace, M. G. (2013). Solving RCPSP/max by lazy clause generation. *Journal of Scheduling, 16*(3), 273-289.

Schwindt, C., & Zimmermann, J. (2015). *Handbook on Project Management and Scheduling Vol. 1.* Springer International Publishing.

Shadrokh, S., & Kianfar, F. (2007). A genetic algorithm for resource investment project scheduling problem, tardiness permitted with penalty. *European Journal of Operational Research, 181*(1), 86-101.

Smith, S. F. (1995). Reactive scheduling systems. In D. E. Brown, & W. T. Scherer, *Intelligent Scheduling Systems* (pp. 155-192). Springer US.

Szelke, E., & Kerr, R. (1994). Knowledge-based reactive scheduling. *Production Planning & Control, 5*(2), 124.

Szmerekovsky, J. G., & Venkateshan, P. (2012). An integer programming formulation for the project scheduling problem with irregular time–cost tradeoffs. *Computers & Operations Research, 39*(7), 1402-1410.

Van de Vonder, S., Ballestin, F., Demeulemeester, E., & Herroelen, W. (2007). Heuristic procedures for reactive project scheduling. *Computers & Industrial Engineering, 52*(1), 11-28. doi:10.1016/j.cie.2006.10.002

Van de Vonder, S., Demeulemeester, E., & Herroelen, W. (2005). Heuristic procedures for generating stable project baseline schedules. *Proceedings of 3rd Euro conference for young OR researchers and practitioners (ORP3)*, (pp. 11-19). Valencia, Spain.

Van Peteghem, V., & Vanhoucke, M. (2013). An artificial immune system algorithm for the resource availability cost problem. *Flexible Services and Manufacturing Journal, 25*(1-2), 122-144.

Van Peteghem, V., & Vanhoucke, M. (2015). Heuristic Methods for the Resource Availability Cost Problem. In *Handbook on Project Management and Scheduling Vol. 1* (pp. 339-359). Springer International Publishing.

Węglarz, J., Józefowska, J., Mika, M., & Waligóra, G. (2011). Project scheduling with finite or infinite number of activity processing modes - A survey. *European Journal of Operational Research, 208*(3), 177-205.

Wiest, J. D. (1963). The Scheduling of Large Projects with Limited Resources. Carnegie Institute of Technology.

Wiest, J. D. (1967). A heuristic model for scheduling large projects with limited resources. *Managment Science, 13*(6), B-359-B-377.

Wiest, J. D., & Levy, F. (1977). *A Managment Guide to PERT/CPM: with GERT/PDM/CPM and Other Networks.* Englewood Cliffs, NJ: Prentice-Hall.

Yamashita, D. S., & Morabito, R. (2009). A note on time/cost tradeoff curve generation for project scheduling with multi-mode resource availability costs. *International Journal of Operational Research, 5*(4), 429-444.

Yamashita, D. S., Armentano, V. A., & Laguna, M. (2007). Robust optimization models for project scheduling with resource availability cost. *Journal of Scheduling, 10*(1), 67-76.

Younis, M. A., & Saad, B. (1996). Optimal resource leveling of multi-resource projects. *Computers & industrial engineering, 31*(1), 1-4. doi:10.1016/0360-8352(96)00116-7

Zhu, G., Bard, J., & Yu, G. (2005). Disruption management for resource-constrained project scheduling. *Journal of the Operational Research Society, 56*(3), 365-381. doi:10.1057/palgrave.jors.2601860

Appendix A: Experimental Buffering Results by Instance

Table 10: Experimental results using a simple buffering approach

PSPLIB Instance		RCPSP Finish	RCPSP/t Finish	Schedule Quality	Delayed jobs	Lateness	Buffered Finish	Buffered Lateness[4]
5	1	53	61	151	27	8	58	3
	2	82	111	142	7	29	90	21
	3	76	105	441	31	29	83	22
	4	63	85	188	16	22	69	16
	5	76	100	373	26	24	83	17
6	1	59	63	122	25	4	64	-1
	2	51	51	0	0	0	56	-5
	3	48	48	0	0	0	52	-4
	4	42	50	112	18	8	46	4
	5	67	102	478	21	35	73	29
7	1	55	55	1	1	0	60	-5
	2	42	44	22	9	2	46	-2
	3	42	61	221	21	19	46	15
	4	44	48	16	5	4	48	0
	5	44	50	22	3	6	48	2
8	1	44	44	16	8	0	48	-4
	2	51	51	33	8	0	56	-5
	3	53	64	49	5	11	58	6
	4	48	48	24	5	0	52	-4
	5	58	58	12	2	0	63	-5
9	1	83	88	74	13	5	91	-3
	2	92	97	207	22	5	101	-4
	3	68	82	137	15	14	74	8
	4	71	98	246	19	27	78	20
	5	70	78	43	9	8	77	1
10	1	42	46	38	8	4	46	0
	2	56	62	54	10	6	61	1
	3	62	62	0	0	0	68	-6
	4	58	63	113	22	5	63	0
	5	41	42	19	13	1	45	-3

---

[4] Projects with negative lateness values finished prior to the buffered due date

Table 11: Experimental results using periodic buffering at the 1st, 2nd and 3rd quarter points

PSPLIB Instance		RCPSP Finish	RCPSP/t Finish	Schedule Quality	Delayed jobs	Lateness
5	1	58	63	88	21	5
	2	90	111	121	9	21
	3	83	105	371	29	22
	4	69	94	202	12	25
	5	83	100	237	21	17
6	1	64	68	34	13	4
	2	56	59	10	4	3
	3	52	54	4	2	2
	4	46	51	22	10	5
	5	73	102	434	22	29
7	1	60	67	14	2	7
	2	46	59	45	5	13
	3	46	60	190	16	14
	4	48	51	17	9	3
	5	48	50	14	6	2
8	1	48	48	7	2	0
	2	56	56	8	3	0
	3	58	64	31	5	6
	4	52	58	21	6	6
	5	63	63	8	1	0
9	1	91	102	155	15	11
	2	101	111	92	18	10
	3	74	87	233	20	13
	4	78	98	246	21	20
	5	77	84	57	9	7
10	1	46	51	20	5	5
	2	61	66	53	15	5
	3	68	70	4	2	2
	4	63	64	10	4	1
	5	45	48	30	9	3

Table 12: Experimental results using weighted buffering according to duration and demand levels

PSPLIB Instance		RCPSP Finish	RCPSP/t Finish	Schedule Quality	Delayed jobs	Lateness
5	1	58	61	31	10	3
	2	90	111	99	8	21
	3	83	105	343	21	22
	4	69	85	118	17	16
	5	83	100	250	21	17
6	1	64	68	62	12	4
	2	56	63	36	6	7
	3	52	54	4	2	2
	4	46	48	14	10	2
	5	73	102	380	22	29
7	1	60	67	14	2	7
	2	46	59	46	4	13
	3	46	61	137	13	15
	4	48	50	7	4	2
	5	48	50	6	3	2
8	1	48	51	18	7	3
	2	56	56	5	3	0
	3	58	64	24	5	6
	4	52	58	19	4	6
	5	63	71	32	4	8
9	1	91	95	50	10	4
	2	101	103	51	18	2
	3	74	77	43	16	3
	4	78	98	155	16	20
	5	77	84	57	12	7
10	1	46	50	39	9	4
	2	61	66	60	16	5
	3	68	70	4	2	2
	4	63	70	57	15	7
	5	45	45	4	3	0

Table 13: Results of three buffering approaches by overall delayed days, number of delayed jobs and project lateness

PSPLIB Instance		Simple Buffering			Periodic Buffering			Weighted Buffering		
		Schedule Quality	Delayed jobs	Lateness	Schedule Quality	Delayed jobs	Lateness	Schedule Quality	Delayed jobs	Lateness
5	1	151	27	3	88	21	5	31	10	3
	2	142	7	21	121	9	21	99	8	21
	3	441	31	22	371	29	22	343	21	22
	4	188	16	16	202	12	25	118	17	16
	5	373	26	17	237	21	17	250	21	17
6	1	122	25	-1	34	13	4	62	12	4
	2	0	0	-5	10	4	3	36	6	7
	3	0	0	-4	4	2	2	4	2	2
	4	112	18	4	22	10	5	14	10	2
	5	478	21	29	434	22	29	380	22	29
7	1	1	1	-5	14	2	7	14	2	7
	2	22	9	-2	45	5	13	46	4	13
	3	221	21	15	190	16	14	137	13	15
	4	16	5	0	17	9	3	7	4	2
	5	22	3	2	14	6	2	6	3	2
8	1	16	8	-4	7	2	0	18	7	3
	2	33	8	-5	8	3	0	5	3	0
	3	49	5	6	31	5	6	24	5	6
	4	24	5	-4	21	6	6	19	4	6
	5	12	2	-5	8	1	0	32	4	8
9	1	74	13	-3	155	15	11	50	10	4
	2	207	22	-4	92	18	10	51	18	2
	3	137	15	8	233	20	13	43	16	3
	4	246	19	20	246	21	20	155	16	20
	5	43	9	1	57	9	7	57	12	7
10	1	38	8	0	20	5	5	39	9	4
	2	54	10	1	53	15	5	60	16	5
	3	0	0	-6	4	2	2	4	2	2
	4	113	22	0	10	4	1	57	15	7
	5	19	13	-3	30	9	3	4	3	0

Appendix B: Sample MRCPSP GMPL Code

GMPL Coding adapted from Hartmann (2012).

```
param D integer; # Planning horizon (number of periods)
param J integer; # Number of activities
param M {0 .. J+1} integer; # Number of modes for job j
param d {j in 0 .. J+1, 1 .. M[j]} integer; # processing time for job j

set P {0 .. J+1} within {0 .. J}; # Immediate predecessors of job j
set KR; # Renewable resources
set KN; # Nonrenewable resources

param r {j in 0 .. J+1, 1 .. M[j], KR union KN} integer; # job j demand in mode m for resource k
param RR {KR} integer; # Renewable resource availability
param RN {KN} integer; # Nonrenewable resource availability
param EF {0 .. J+1} integer; # Earliest finish time for job j
param LF {0 .. J+1} integer; # Latest finish time for job j

var x {j in 0 .. J+1, 1 .. M[j], EF[j] .. LF[j]} binary; # indicates when job j finishes in mode m at time
t

minimize Makespan:
sum {t in EF[J+1] .. LF[J+1]} t * x[J+1,1,t];

subject to JobModeCompletion {j in 0 .. J+1}:
sum {m in 1 .. M[j]} sum {t in EF[j] .. LF[j]} x[j,m,t] = 1;

subject to PrecedenceRelations {j in 1 .. J+1, h in P[j]}:
sum {m in 1 .. M[h]} sum {t in EF[h] .. LF[h]} t * x[h,m,t] <=
sum {m in 1 .. M[j]} sum {t in EF[j] .. LF[j]} (t-p[j,m]) * x[j,m,t];

subject to RenewableResources {k in KR, t in 1 .. D}:
sum {j in 1 .. J} sum {m in 1 .. M[j]} r[j,m,k] * sum {q in max(t,EF[j]) .. min(t+p[j,m]-1, LF[j]) }
x[j,m,q]
<= RR[k];

subject to NonrenewableResources {k in KN}:
sum {j in 1 .. J} sum {m in 1 .. M[j]} r[j,m,k] * sum {t in EF[j] .. LF[j]} x[j ,m,t] <= RN[k];

solve;

end;
```

Appendix C: Sample RACP GMPL Code

```
param D integer; #Project Deadline
param K integer; #Total Resource Types
param n integer; #Total Activities
param ES {0..n+1}:=0; #Earliest Start Time
param LS {0..n+1}:=D; #Latest Finish Time
param c {1..K}; #Cost of Resource Level
param M {0 .. n+1}; # Number of modes for job n
param p {i in 0..n+1, 0..M[i]}; #Processing Time
param W := 0; #RCPSP weighting factor

set P {0 .. n+1} within {0 .. n}; #Predecessors
set KR; #Renewable resources
set KN; # Nonrenewable resources

param RR {KR union KN} integer; # Resource availability
param r {i in 0..n+1, 0 .. M[i], KR union KN} integer; #Activity Demand for Resource

var R{k in 1..K}; # Total resource used
var x{i in 0..n+1, 1 .. M[i], t in ES[i]..LS[i]} binary;

maximize RAC:
sum{k in 1..K-1} (RR[k]-R[k])*c[k]-R[K]
+W*sum{i in 1..n+1, m in 1 .. M[i], t in ES[i]..LS[i]} x[i,m,t]*t;

subject to job_completion{i in 1..n+1}:
sum{m in 1 .. M[i]}sum{t in ES[i]..LS[i]} x[i,m,t] = 1;

subject to predecessors{i in 0..n+1, j in P[i]}:
sum{m in 1 .. M[i]}sum{t in ES[i]..LS[i]} t*x[i,m,t] >=
sum{m in 1 .. M[j]}sum{t in ES[j]..LS[j]} (t+p[j,m])*x[j,m,t];

subject to resource_capacity{k in 1..K, t in 0..D}:
sum{i in 1..n+1, m in 1 .. M[i], b in max(t-p[i,m]+1, ES[i])..min(LS[i],t)} r[i,m,k]*x[i,m,b] <= R[k];

subject to NonrenewableResources {k in KN}:
sum{i in 1..n, m in 1..M[i], t in ES[i]..LS[i]} r[i,m,k]*p[i,m]*x[i,m,t] <= RR[k];

solve;

end;
```