Effects of Processing Time Distributions and Rescheduling Policies on Efficiency and Instability for Single Machine Settings

Chelsey Hvingelby

A Thesis

in

The Department

of

Mechanical, Industrial and Aerospace Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of

Master of Applied Science (Industrial Engineering) at

Concordia University

Montréal, Québec, Canada

June 2023

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: **Chelsey Hvingelby**

Entitled: **Effects of Processing Time Distributions and Rescheduling Policies on Efficiency and Instability for Single Machine Settings**

and submitted in partial fulfillment of the requirements for the degree of

Master of Applied Science (Industrial Engineering)

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the Final Examining Committee:

_____ Chair and Internal Examiner
Dr. Hossein Hashemi Doulabi

_____ Internal Examiner
Dr. Onur Kuzgunkaya

_____ Acting Supervisor
Dr. Masoumeh Kazemi-Zanjani

_____ Supervisor
Dr. Daria Terekhov

Approved by _____
Dr. Martin D. Pugh, Chair
Department of Mechanical, Industrial and Aerospace Engineering

_____ 2023

_____
Dr. Mourad Debbabi, Dean
Gina Cody School of Engineering and Computer Science

Abstract

Effects of Processing Time Distributions and Rescheduling Policies on Efficiency and Instability for Single Machine Settings

Chelsey Hvingelby

Operating room (OR) scheduling is becoming increasingly important and poses a challenging problem due to the presence of uncertainty. Motivated by OR scheduling, we study the single machine scheduling problem with uncertainty in both dynamic arrivals and stochastic processing times. We use rescheduling to handle uncertainty and study both efficiency and instability (a measure of schedule deviation). The chosen efficiency metric, final total weighted tardiness (FTWT), is the total weighted tardiness of the final schedule.

We develop a discrete-event simulation framework with embedded optimization to conduct two computational experiments. The first experiment analyzes the joint effect of different processing time distributions and rescheduling policies for the single machine scheduling problem with deterministic processing times and dynamic arrivals. Secondly, we analyze the joint effect of different processing time distributions and rescheduling policies for the single machine problem with stochastic processing times and dynamic arrivals.

For both experiments, we find that the variance of the processing time distributions has a greater impact than the shape of the distribution. Furthermore, the *eventJobCompletion* policy, which reschedules every time a job finishes processing, performs well across all distributions in both experiments. For the stochastic experiment, the *eventJobCompletion* policy also stood out as being the most consistent across all processing time distributions. Finally, in both experiments, we find that the average length of the rescheduling interval does not have an effect on total instability and FTWT.

Acknowledgments

First and foremost, I would like to thank my supervisor, Dr. Daria Terekhov, for her continued guidance and support throughout this entire process. Her dedication and mentorship have allowed this research, thesis, and my growth as an academic be what it is today. I also extend my extreme gratitude to Dr. Phil Troy for his guidance and expertise, specifically in relation to operating room scheduling and health care applications.

I would like to thank the examiners, Dr. Hossein Hashemi Doulabi, Dr. Onur Kuzgunkaya, and Dr. Masoumeh Kazemi-Zanjani, for their interest, questions, and valuable feedback. Thank you to my labmates in the Data-Driven Operations Research Lab for their discussions and for demonstrating successful academic careers.

Additionally, I would like to thank the Natural Sciences and Engineering Research Council of Canada, Concordia University, the Department of Mechanical, Industrial and Aerospace Engineering, and my supervisor for their financial support.

I am exceptionally fortunate to have such a caring, and very large, family who constantly encourage and support me through my every endeavor. To Mom and Scott, and Dad and Teresa, thank you for believing in me and celebrating my achievements. A very special thank you to my amazing grandparents for always cheering for me and showing an interest in my work and life. Zaida, I know you are immensely proud. Thank you to my friends who pull me away from my computer, make me laugh, and bring such a light to my life. Although my research focuses on scheduling under uncertainty, I do not enjoy facing uncertainty in my personal life. I would like to express my sincere gratitude to Mark for being my constant in the face of all the uncertainty the past few years have brought on. Lastly, thank you to my cat, George, for being George.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Uncertainty is unavoidable and can be a significant challenge in scheduling problems. This thesis is motivated by operating room (OR) scheduling, which has multiple sources of uncertainty (e.g., patient demand, uncertain surgery duration time, worsening patient conditions, machine or equipment breakdowns, and availability of upstream or downstream bed or personnel resources). Given the high level of uncertainty and the expense of ORs, OR scheduling has received a significant amount of attention in the literature. Recent advances in data collection using sensors and real-time monitoring of events (e.g., digital twins) have contributed to an increase in data-driven decision-making in scheduling, which emphasizes reacting to real-time events to guide scheduling decisions.

This thesis assumes a predictive-reactive approach of rescheduling is used to handle the uncertainty in dynamic arrivals and stochastic processing times. The central goal of this thesis is to determine the effect of various rescheduling policies and processing time distributional assumptions on efficiency (the performance of the system in the given objective function) and instability (the cost of schedule changes). To the best of our knowledge, within the rescheduling and OR scheduling literature, this thesis is the first to explore the joint effect of rescheduling policies and processing time distribution assumptions. Our findings, consistent with previous research, demonstrate that the rescheduling policies which utilize the most complete and accurate information perform best, regardless of the rescheduling frequency. Furthermore, the variance of the processing time distributions has more impact than the shape of the distribution itself.

1.1 Motivations

Our research is motivated by OR scheduling where both the efficiency and instability are important. In 2010, Cardoen et al. wrote that, "It is somehow contradictory to see that in a domain as practical as operating room planning and scheduling, so little research seems to be effectively applied" [1]. Barriers to simulation and machine learning model implementation in the health care industry include resistance to change from the users and organization [4], lack of interpretability [5], complexity [6], and difficulty to use [7]. While not unique to the health care industry, the barriers to implementation are heightened in relation to OR scheduling due to the sensitive and consequential nature of decisions and a high ethical standards. In this thesis we use rescheduling, which refers to the process of updating an existing schedule in response to realizations of uncertainty [8]. The main components of a rescheduling approach consist of the rescheduling strategy (whether schedules are developed in a predictive-reactive approach or a reactive approach such as a dispatching rule), rescheduling policy (when to perform the rescheduling), and rescheduling method (how the new schedule is generated) [8]. Employing a rescheduling approach to scheduling under uncertainty offers flexibility by allowing an organization to implement a rescheduling method which is best suited to the specific application. For example, a rescheduling method which is close to current practice such as a rule-based heuristic may reduce resistance to change and a rescheduling method which is highly interpretable is likely to increase the chance of implementation. Furthermore, this rescheduling method can be updated with organizational changes or adapted to a more complex model as organizational resistance reduces. Motivated by the resistance to change, lack of interpretability, and complexity barriers to model implementation, we choose to study the effect of different distributional assumptions in the context of rescheduling. Larsen and Pranzo list three advantages of dynamic rescheduling compared to other approaches such as stochastic programming: tractability (reasonable computational times depending on the application), ease of implementation (can use a pre-existing deterministic solver within a rescheduling framework), and the fact that uncertainty does not need to be embedded into the solver [9]. Although (depending on the chosen rescheduling strategy, policy, and method) rescheduling may perform worse than a different solution approach in terms of the performance metrics, we follow the guiding principle that "a small implemented

improvement is preferable than a big one on paper" [4]. When considering a rescheduling approach to OR scheduling problems, it is important to consider a measure of instability which may include inconveniences caused to doctors, the hospital, and or patients [10].

Specifically in the OR scheduling and planning literature, there exists a disparity among the statistical distributions used to model surgery durations or processing times. This field faces a particular difficulty in generating reproducible research due to high privacy requirements surrounding health care data. This thesis is motivated by the question of the impact of the choice of statistical distribution to model processing times, assuming equal variance, on efficiency and instability. In particular, the objective of this thesis is to determine the combined effect of rescheduling policies and processing time distributional assumptions on final total weighted tardiness (FTWT) and instability.

1.2 Thesis Overview

Chapter 2 provides a review of the relevant literature pertaining to this thesis. The chapter begins with a discussion of rescheduling and the metrics associated with it: efficiency and instability. The OR scheduling problem and its solution approaches are then described. This chapter includes five main sections: scheduling under uncertainty, rescheduling and relevant metrics, OR scheduling, positioning our work within the existing literature, and finally, a concluding section.

Chapter 3 details the problem setting of a single machine rescheduling problem with stochastic processing times and dynamic arrivals in a non-preemptive environment with the objective of minimizing total weighted tardiness. This chapter also presents the definitions of efficiency, specifically FTWT, and instability used throughout the thesis.

In Chapter 4, we outline the methodology used, including the discrete-event simulation (DES) framework, the various rescheduling policies we consider, and the complete rescheduling method used to solve the subproblems during each rescheduling action.

The experimental study is outlined in Chapter 5, including data acquisition using Synthea data and parameter setting. The simulation setup includes generating the initial states, and determining

the number of replications. Finally, the experimental outline and results are included for each experiment. Experiment 1 focuses on the deterministic case where the only source of uncertainty is in the dynamic arrival of jobs while experiment 2 focuses on the case where there is uncertainty in both the dynamic arrival of jobs and the processing times of jobs. Within the results section of each experiment, we perform the following analyses:

- We compare total instability and FTWT by scenario (combination of rescheduling policy and processing time distribution). In particular, we answer the following three questions:

  (1) Does the assumption of a specific statistical distribution for processing times have an effect on FTWT and total instability?

  (2) Can we determine a rescheduling policy which performs well under all distributions?

  (3) Can we identify a rescheduling policy whose performance is least affected by the different processing time distributions?

- We determine the effect of the average length of the rescheduling interval on total instability and FTWT.

- Given the time limit we impose on the complete rescheduling method, we analyze the percent of suboptimal solutions per scenario.

Finally, Chapter 6 concludes this thesis by summarizing the main contributions and proposing potential areas for future work.

1.3 Summary of Contributions

First, we develop a DES framework for dynamic single machine rescheduling which can be used to study the effects of input distributions and rescheduling policies on the instability and efficiency metrics. The proposed framework contributes to bridging the gap between scheduling and simulation literature by evaluating the effect of input distributional assumptions for scheduling problems.

Second, we analyze the joint effect of different processing time distributions and rescheduling policies for the single machine problem with deterministic processing times and dynamic arrivals.

Specifically, we use complete rescheduling and consider various periodic, event-driven and hybrid rescheduling policies.

Lastly, we analyze the joint effect of different processing time distributions and rescheduling policies for the single machine problem with stochastic processing times and dynamic arrivals. Again, we use complete rescheduling and consider various periodic, event-driven and hybrid rescheduling policies.

To compare total instability and FTWT by scenario, we answer the three questions posed in the previous section as follows:

(1) The variance of the distribution has a greater impact on FTWT and total instability than the shape of the distribution for both experiments.

(2) For both experiments, the *eventJobCompletion* policy performs well across all distributions.

(3) For the experiment with the deterministic processing times, no policy stood out as being least affected by the different processing time distributions. However, for the stochastic experiment, we identify the *eventJobCompletion* policy as being least affected by the different processing time distributions.

Furthermore, in both experiments, we find that the average length of the rescheduling interval does not have an effect on total instability and FTWT and that the *eventHighPriority* policy reaches the computational time limit for solving subproblems more frequently than the remaining policies.

Chapter 2

Literature Review

This thesis tackles the single machine scheduling problem with uncertainty in stochastic processing times and dynamic arrivals. The goal is to assess the effects of different processing time probability distributions and rescheduling policies on efficiency and instability. Relevant literature therefore includes scheduling under uncertainty, rescheduling as a particular approach to scheduling under uncertainty, and an analysis of efficiency and instability metrics when performing rescheduling. In all of the previously mentioned areas of literature, we focus on single machine scheduling. Since our work is motivated by operating room (OR) scheduling, we review existing OR scheduling literature with an emphasis on literature which employs rescheduling.

This chapter is divided into the following sections:

- Section 2.1 - Scheduling Under Uncertainty
- Section 2.2 - Rescheduling
    - Section 2.2.1 - Efficiency and Instability Definitions
- Section 2.3 - Operating Room Scheduling
- Section 2.4 - Positioning Our Work in Existing Literature
- Section 2.5 - Conclusions

2.1 Scheduling Under Uncertainty

Uncertainty can be related to resources or jobs [8, 11, 12]. Potential sources of uncertainty related to resources include machine breakdowns, resource availability (e.g. staffing uncertainty), and material availability (e.g. supply chain issues). Job-related uncertainty may include stochastic processing times, arrivals of urgent jobs, cancellations or late arrivals, and job priorities changing over time. Scheduling under uncertainty is a general term for scheduling where the problem includes one or more forms of uncertainty. We focus on dynamic scheduling, defined as scheduling where the set of jobs are not known apriori and jobs arrive over time [8, 13, 14, 15, 16]. Furthermore, we consider both the deterministic case (where the processing time of a job is known) and the case where the processing times are stochastic. Aytug et al. identify three key components of uncertainty: the cause of the uncertainty, the context of the uncertainty such as the state of the surrounding environment, and the result of the uncertainty [17].

Scheduling under uncertainty is often categorized as completely reactive scheduling, robust (proactive) scheduling, or predictive-reactive scheduling [11, 17]. Completely reactive scheduling does not involve scheduling ahead of time and usually employs heuristic approaches or dynamic dispatching rules [11, 17]. Robust scheduling attempts to create a predictive schedule to minimize the effects of uncertainty by using information about the uncertainties [11, 17]. Lastly, predictive-reactive scheduling consists of an initial schedule which is adapted in response to real time events in a process referred to as rescheduling [11, 17]. Bidot et al. introduce a classification of approaches for scheduling under uncertainty including proactive (robust), revision (predictive-reactive), progressive, and mixed [18]. Progressive consists of generating short-term schedules in a rolling time horizon and the mixed category can be defined as mostly revision with progressive techniques [18]. Online and offline solution methods are described in [19] where offline methods create schedules before realizations of uncertainty occur and online methods react to realizations of the uncertainty in real-time. Generating schedules offline, such as creating robust schedules, often incurs a significant computational cost while online methods may sacrifice performance in order to respond to realizations of uncertainty in a timely manner [19].

Montana lists four solution methodologies for solving dynamic scheduling problems and describes conditions under which each methodology performs well: predictive-reactive approaches are useful when disruptions to the original schedule are minor, continuous combinatorial optimization (a predictive-reactive approach in which the same scheduling algorithm is used for the original and subsequent schedules) is beneficial when the future is relatively predictable and the computational cost of combinatorial optimization is not prohibitive, completely reactive (dispatch rules) perform well when the level of uncertainty is high, and stochastic optimization is useful when there exists uncertainty which can be described statistically [13]. Our goal is to evaluate rescheduling policies and the effect of processing time probability distributions while keeping the method of constructing a schedule for a set of jobs fixed. Specifically, we use mixed-integer programming (MIP) to construct the schedule in each subproblem. Since the focus is on rescheduling, we do not review papers on stochastic optimization, robust optimization, or heuristic solution methodologies.

2.2 Rescheduling

Vieira et al. define rescheduling as "the processes of updating an existing production schedule in response to disruptions or other changes" [8]. *Rescheduling environments* can be static, meaning there is a finite set of jobs, or dynamic, in which case there is an infinite set of jobs [8]. *Rescheduling strategies* determine whether a schedule is generated or not, such as a dispatching rule or predictive-reactive strategy [8]. *Rescheduling policies* determine when to reschedule and include periodic, event-driven, and hybrid policies [8, 11]. Finally, *rescheduling methods* describe how schedules are created and consist of schedule generation methods, such as complete and partial rescheduling, and schedule repair methods, such as a right-shift heuristic [8, 12]. This thesis considers a dynamic rescheduling environment, a predictive-reactive rescheduling strategy, a complete rescheduling method, and explores various rescheduling policies.

Three main performance measures that are considered when performing rescheduling are schedule efficiency, schedule instability, and rescheduling costs [8]. Schedule efficiency measures how well a schedule performs in terms of the objective function being considered (e.g. total weighted tardiness, or makespan) [8] while instability measures the disruption caused by schedule changes

[15]. These two performance metrics often conflict as "rescheduling involve[s] trade-offs between quickly responding to information about the unexpected events versus system instability resulting from the response" [20]. An exploration of the various efficiency and instability metrics defined in literature is presented in Section 2.2.1.

Rescheduling may be utilized to react to realizations of uncertainty with the goal of maintaining feasibility or improving performance. Bahroun et al. propose a flexible decision support tool which constructs a schedule using a rolling horizon strategy, and implements automatic or manual corrections if the schedule becomes infeasible [21]. Hence, the framework proposed by [21] uses an event-driven rescheduling policy with the infeasibility of the schedule being the event that triggers rescheduling. However, they do not include functionality to reschedule to increase performance under new information, unless the schedule becomes infeasible.

Bidot et al. propose a general rescheduling framework which combines proactive, revision, and progressive techniques using a directed acyclic graph independent of the solver [18]. The nodes represent schedules while the edges represent transitions, either through a progression or revision technique, between schedules [18]. Larson and Pranzo bridge the gap between literature and practice by expanding on the work of [18] with the addition of a supervisor and solution read/write modules that would be required to integrate the framework in practice [9]. Pfeiffer et al. present a discrete-event simulation (DES) framework to compare rescheduling policies while considering both instability and efficiency metrics for two different problem settings: a single machine environment with uncertainty in arrivals and a job shop environment with uncertainty in job processing times [15]. However, the authors do not consider the joint effect of uncertainty in arrivals and processing times. Cowling and Johansson propose a framework to deal with incoming data regarding uncertainties, specifically in the duration of jobs, and decide on an appropriate action such as a schedule repair or complete rescheduling [12].

A dynamic job shop scheduling problem with the objective of minimizing total tardiness is solved in [22] using a periodic rescheduling approach. Recognizing that early idle times limit the flexibility in the schedule, the authors introduce a monotonically decreasing flexibility term in the objective function to penalize early idle times [22]. In their computational study, the authors show that the inclusion of the flexibility term decreased instability, even though minimization of instability

was not the objective. [22].

Based on the literature reviewed in this thesis, there is a consensus regarding the existence of a trade-off between schedule efficiency and instability [9, 12, 15, 18, 20, 23, 24]. For a single machine environment with uncertainty in dynamic arrivals and deterministic processing times, it has been shown that increasing the rescheduling interval (rescheduling less frequently) decreases instability but leads to a worse efficiency [15]. However, it has also been shown that minimal improvements to efficiency are made past a certain number of rescheduling actions [14, 17, 18, 23]. Early knowledge of uncertain events and ensuring the most recent information regarding the state of the system appear to be more important than the rescheduling frequency and lead to more effective rescheduling actions [9, 20].

2.2.1 Efficiency and Instability Definitions

Efficiency and instability are two performance metrics which can be used to weigh the benefit gained from performing rescheduling with the disruption caused by changing the schedule. Generally, performing a rescheduling action results in better efficiency and increased instability compared to a scenario where no rescheduling action is performed. The exact definition of efficiency and instability varies based on the problem formulation. Formal definitions of instability and efficiency used throughout this thesis can be found in Sections 3.3.3 and 3.3.4 respectively. The focus of this section is to examine and compare the various efficiency and instability definitions present in the literature. Although instability can sometimes be referred to as nervousness or stability (with the goal of maximizing) in the literature, we use the term *instability* throughout this literature review for consistency.

Since efficiency relates to the objective function, the goal can either be to minimize or maximize the objective function. However, in the rescheduling literature we explored, the majority of efficiency metrics focus on minimizing an objective function. Some of the efficiency metrics present in literature include maximum tardiness [25], mean tardiness [22], makespan [9, 24, 26], mean flow time [15], average completion time [12], and maximum lateness [23]. An example of an efficiency metric that should be maximized is the fraction of jobs completed on time [13]. In [14], a linear cost function of average flow time and set up frequency is used as the efficiency metric.

Cowling and Johansson define a general *utility* function which measures the benefit gained by performing rescheduling compared to a *do nothing strategy* which does not react to realizations of uncertainty [12]. For a single rescheduling event, the general utility function is computed as $U = v_{opt} - v_0$ where v_{opt} and v_0 represent the objective function value given by the schedule when rescheduling is employed and the schedule if the do nothing strategy is employed respectively [12]. In the case where the objective is minimization, the utility should be less than or equal to zero with the magnitude representing the benefit obtained from rescheduling. Since the utility metric is generic, any of the previously mentioned objective functions can be applied. However, the do nothing strategy is highly dependent on the problem formulation.

To compare the various instability metrics present in the literature, we define the following variables (which we attempt to keep consistent with the instability definition in Section 3.3.3). B_j and C_j represent the start and completion times of job j in the schedule prior to rescheduling while B'_j and C'_j represent the start and completion times of job j in the schedule created from rescheduling. Additional notation that is specific to an instability definition will be described throughout.

Some simple measures of instability that are applicable for both single and parallel machine environments are number of rescheduling actions [18, 23] and rescheduling frequency, or the average number of rescheduling actions performed per time unit [14]. However, these measures do not account for the magnitude of changes occurring per rescheduling action.

Cowling and Johansson define an instability metric which is a function of the start and completion times of jobs of the schedule prior to rescheduling and the schedule produced as a result of rescheduling [12]. The general formula is given by

$$I = \sum_{j=1}^{n} min\{\alpha(|B_j - B'_j| + |C_j - C'_j|), D_j\}$$

where n represents the total number of jobs, α is the cost per unit of time a job is displaced, and D_j represents the cost of outsourcing, or cancelling, job j [12]. The effects on instability come from both the changes to the processing times of jobs as well as re-sequencing as a result of rescheduling. While the formula provided by [12] is generic, it is used for a single machine processing problem setting with uncertain processing times and n fixed jobs (no dynamic arrivals).

Akkan considers a single operation insertion on a non-preemptive single machine and defines instability as the average deviation of start times of jobs given by $\hat{D} = \frac{1}{n-1}\sum_{j=1}^{n-1}|B_j - B'_j|$ where there are $n-1$ jobs in the schedule prior to rescheduling [25]. Based on their literature review, the author claims this is the most common measure of instability for single machine environments [25]. Similarly, Branke and Mattfeld define instability as the sum of the deviation of start times of jobs ($D = \sum_{j=1}^{n-1}|B_j - B'_j|$) [22]. Completion times do not need to be considered in the previous instability definitions since processing times are assumed to be deterministic in both cases.

For a single machine problem with a single general disruption (which the authors claim can be used to model a machine breakdown, rush order, or change in a jobs' processing time), two different measures of instability are discussed: a measure of starting time deviations, denoted D_0, and a measure of deviations in the sequence of jobs, denoted D_s [24]. The two metrics are defined as $D_0 = \sum_{j\in N'}|B'_j - B_j|$ and $D_s = \sum_{i\in N'}|B'_j - B^r_j|$ respectively where N' represents the set of previously scheduled jobs who have not yet completed processing at the time of rescheduling, and B^r_j is the start time of job j in the *right shift* schedule [24]. The right shift schedule uses the same job sequence as the previous schedule, but shifts any jobs to the right to prevent overlap and maintain feasibility of the single machine [24]. Although the summation in the instability definition given by D [22] is over all jobs, jobs whose processing has completed prior to rescheduling will contribute zero to the sum. Therefore, this metric is similar to D_0 defined in [24]. The authors present a use case for D_0 as a scheduling problem where limited resources are assigned to each job and rescheduling the resources incurs a cost while D_s could be useful for a scheduling problem where the re-arranging of jobs is physically or spatially difficult such as pallets scheduled for shipment [24].

Pfeiffer et al. address a limitation of the previously mentioned instability metrics by considering the impact of the time of rescheduling relative to the starting time of jobs through the use of a term called the *actuality penalty* [15]. By incorporating this additional term in the instability definition, the authors acknowledge that a disruption caused when rescheduling a job close to its starting time is likely more consequential than a disruption caused when rescheduling a job far in advance. The

new penalty function is given by

$$\bar{PN} = \frac{1}{|N'|} \sum_{j \in N'} [|B'_j - B_j| + \frac{k}{\sqrt{B_j - T}}]$$

where N' is the set of unprocessed jobs, k is the scaling factor for the actuality penalty, and T is the time the rescheduling action occurred [15]. Wu et al. measure instability as a sum of carrying costs (costs due to delaying jobs) and rushing costs (costs due to starting jobs earlier than previously scheduled) [26]. Similar to the actuality penalty in [15], the authors consider the impact of the time of rescheduling relative to the starting time of jobs for the rushing costs only through the inclusion of the following term in the instability definition $\sum_{j=1}^{n} \lambda e^{-\lambda(B'_j - T)} y_j$ where y_j is a binary variable which is equal to one if job j is rushed ($B'_j < B_j$) and λ is a tunable parameter [26].

Instability and efficiency can be considered simultaneously in the objective function [12, 14, 15, 24, 26], or as primary and secondary objectives [25]. Additionally, similar to our work, instability may be evaluated without being included as an objective [22, 23]. As will be seen in Section 3.3.3, our instability metric is closest to the measure of sequence deviation (D_s) given in [24].

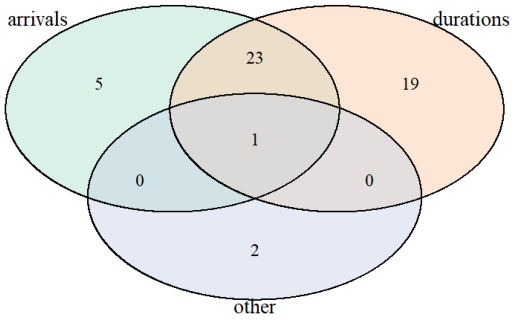
2.3 Operating Room Scheduling

Within a hospital, OR(s), surgeons, and anesthesiologists are among the most expensive resources [27] prompting a need to optimize OR scheduling. Many surgeries were cancelled in recent years due to the limited surgery availability resulting from the COVID-19 pandemic and the demand it created for hospital rooms and ventilators as well as the increased risks associated with undergoing a surgery in the height of the pandemic [28]. Consequently, there is a substantial backlog of elective surgeries and the increased waiting time for surgery will effect patients' lives either through adverse effects on prognosis, diminished of quality of life, and/or deterioration of mental health [28]. Although the importance of optimizing OR scheduling has been highlighted in recent years, OR scheduling is an extremely challenging problem due to its inherent uncertainty in surgery durations and possible emergency arrivals. Furthermore, the OR schedule is highly intertwined with schedules of other departments such as staff and upstream and downstream resources such as the

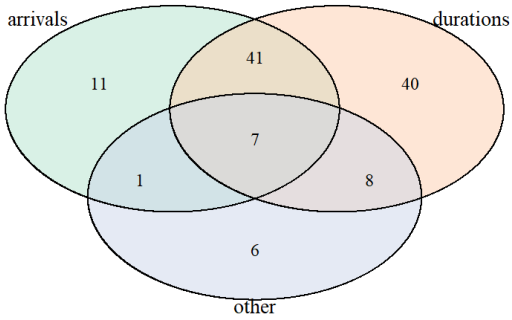
post-anesthesia care unit [27].

Within the topic of OR scheduling, there are many possible variations of the problem. Decisions related to OR scheduling are categorized into three levels: strategic (long term), tactical (medium term), and operational (short term or day of) [29]. Operational may be further divided into offline operational which usually involves elective patients only and online operational which monitors and schedules in real time [30]. Furthermore, there are three common methods of assigning OR time to specialty surgical groups: block scheduling, open scheduling and modified block scheduling [29]. In block scheduling, a set of predefined blocks are assigned to each surgery specialty whereas open scheduling assigns surgeries to ORs without first dividing the ORs into specialty blocks [29]. Modified block scheduling combines the open and block scheduling methods by assigning some blocks to specialties and leaving the remaining blocks open [29, 30]. OR scheduling can focus on inpatient or outpatient scheduling. Inpatients are defined as patients who remain in the hospital overnight following surgery while outpatients have a shorter hospital stay (approximately 4-6 hours) [30]. Our work focuses on the (online) operational decisions related to the sequencing and scheduling of inpatient surgeries. Similar to [31], our approach is relevant in open scheduling environments or for sequencing and scheduling surgeries within a pre-determined block.

We rely on survey papers [1, 2, 29, 30] to summarize some of the trends in OR scheduling literature over the past couple of decades. While our choice of OR scheduling literature reviews is not exhaustive, the set of chosen papers covers a broad timeline and allows for identification of trends over time. Cardoen et al. classify 115 manuscripts published between 2000 and 2010 by patient characteristics, performance measures evaluated, the level of the decision made, the research methodology, and the consideration of uncertainty [1]. Building on the work of [1], Samudra et al. utilize many of the same categorizations of the literature to review 216 technical OR scheduling papers published between 2004 and 2014 [2]. Based on the literature published between 2000 and 2014, non-elective patient types have received less attention than the elective patients [1, 2]. In both Venn diagrams in Figure 2.1, we observe that most of the papers which consider uncertainty in the arrivals also consider uncertainty in the durations [1, 2]. Uncertainty is considered in capacity-related problems more frequently than time assignment problems [2]. Our work lies in the intersection of the arrival and duration uncertainty and relates to the time assignment problem.



(a) Venn diagram of the type of uncertainty considered based on the data from [1] (consisting of a total of 115 papers from 2000-2010).



(b) Venn diagram of the type of uncertainty considered based on the data from [2] (consisting of a total of 216 papers from 2004-2014).

Figure 2.1: Venn diagrams of the type of uncertainty considered in papers which consider uncertainty using data from [1, 2]. Note that totals may be inconsistent as some papers fall into multiple categories.

In a more recent literature review. Rahimi and Gandomi present a thorough scientometric study of OR scheduling papers published between 2010 and 2019 and group their results into the following three date ranges: $<$ 2005, 2005-2010, and 2010-2020 [29]. Contrary to [1, 2], Rahimi and Gandomi show that non-elective patients received more attention than elective patients between 2005-2010 [29]. Rahimi and Gandomi also demonstrate that, until recently (2010-2020), deterministic problem settings outnumbered problem settings with a consideration of uncertainty [29, Fig. 13]. Duration uncertainty is shown to be considered more frequently than uncertainty in arrivals [29, Fig 14]. However, one must be careful in interpreting trends over time from these figures ([29, Fig. 13, Fig. 14]) as the bar graphs do not take into consideration the total number of papers published within each date range.

Finally, Wang et al. provide a literature review of 178 papers spanning 2000-2022 with a focus on comparing inpatient and outpatient OR scheduling [30]. Only a third of papers consider uncertainty in demand and the majority of these papers fall into the inpatient setting [30, Table 3]. The authors note that mathematical programming is the most common methodology with stochastic programming being used to incorporate some of the uncertainty [30].

We categorize technical papers based on the sources of uncertainty considered similar to the Venn diagrams in Figure 2.1, exclude papers which do not consider any form of uncertainty, and focus on OR scheduling papers which employ rescheduling.

2.3.1 OR Scheduling with Uncertain Durations

Marques and Captivo use robust optimization to perform advanced scheduling of elective patients while considering uncertainty in surgery durations but do not sequence surgeries or assign start times [32]. Khaniyev et al. schedule a single OR for the next-day with stochastic durations but the set of (elective) surgeries are known apriori and pre-sequenced [33]. With the goal of minimizing a weighted sum of waiting time, idle time, and overtime costs, the authors employ stochastic optimization, explore various heuristics, and finally propose a combination of two heuristics which shows improved performance [33]. While [33] assumes pre-sequenced surgeries, Denton et al. sequence and schedule surgeries with uncertain durations using a two stage-stochastic program with recourse to minimize idle and overtime costs where the first stage focuses on the sequencing of surgeries [31]. The authors consider various heuristics to solve the sequencing problem and find that sequencing surgeries by smallest variance first, although not commonly implemented in practice, performs well [31]. Zhou et al. consider the next-day scheduling of a set of pre-determined (elective) surgeries including sequencing and assigning expected durations to minimize the expected time span and respect patient waiting time guidelines [34]. Day of elective surgery rescheduling with uncertain durations is handled in [35] using a right shift heuristic to reschedule when a surgery is running late. As a result of uncertainty in durations, Xiao et al. adapt a single OR daily schedule during execution using both exact methods as well as more computationally feasible heuristics [36].

2.3.2 OR Scheduling with Uncertain Arrivals

In agreement with the Venn diagrams in Figure 2.1, we observe relatively few papers which consider uncertainty in the form of dynamic arrivals but not surgery durations. OR rescheduling with dynamic arrivals of emergency patients is solved in [37] using a mixed-integer linear program (MILP) and, when there is a high elective surgery load, a genetic algorithm which is necessary due to the computational expense of the MILP.

2.3.3 OR Scheduling with Uncertainty in both Arrivals and Durations

OR scheduling with uncertainty in surgery durations and dynamic arrivals of elective patients is considered in [38]. The authors propose a method to reschedule cancelled patients on a rolling time horizon assuming a block scheduling strategy has previously been implemented and control instability by limiting the number of variations from the previous schedule [38]. Similar to our efficiency metric of minimizing total weighted tardiness, the authors choose a patient-centric efficiency metric of minimizing a combination of waiting time and tardiness [38]. While the work of [38] and this thesis are both considered operational scheduling, [38] focuses on advance scheduling and considers dynamic arrivals of previously cancelled patients whereas we focus on allocation scheduling and dynamic arrivals of emergency patients. Davarian and Behnamian extend the work of [38], most notably by including uncertainty in the arrival of emergency patients, and utilize robust optimization [39]. However, to the best of our knowledge, [39] does not limit the number of variations between successive schedules and therefore does not consider a measure of instability.

In a single OR setting, Stuart and Kozan schedule elective and non-elective outpatient surgeries on a daily basis and reschedule whenever a surgery is completed [40]. Van Essen et al. discuss the operational OR scheduling problem with the objective of minimizing deviation from various stakeholder preferences using an integer linear program (ILP) and periodic rescheduling to handle uncertainty [41]. Since shifting surgeries and inserting breaks into the schedule are the most common patterns observed by analyzing the ILP solutions, van Essen et al. generate a heuristic which can be solved faster than the exact method [41]. Their framework includes a human in the loop element (see also [4, 9, 21]) to present a manager or scheduler with a couple of possible schedules [41].

2.3.4 Rescheduling in OR Scheduling Problems

In this section, we focus on papers which use rescheduling to handle uncertainty present in OR scheduling and, when applicable, discuss their instability metrics. From the papers we reviewed, we observed two main categories of OR rescheduling: that which reschedules patients to a different day [10, 38, 39] and that which takes a more online approach of rescheduling on the day of surgery

[35, 40, 41, 42]. While both categories can be considered operational scheduling, the former is more aligned with advanced scheduling while the latter, which is similar to our work, is more closely aligned with allocation scheduling [2].

Three of the most common reasons for rescheduling are other (hospital related), change in a patients' program, and insufficient OR capacity as determined using machine learning [43]. Eshghali et al. embed machine learning into a three-phase schedule generation approach: phase one produces a weekly schedule which reserves emergency surgery capacity using results of a forecasting model, phase two sequences the elective surgeries to produce a daily schedule, and phase three reschedules on the arrival of an emergency patient where the patient's surgery duration is predicted using a random forest model [42]. When rescheduling, instability is limited through the constraint that surgeries must take place on the originally scheduled day, potentially resulting in overtime [42].

In [10], a tentative schedule is created and a single rescheduling action is performed a few days prior to the schedule being implemented. While not closely related to our specific use of rescheduling, this paper provides an interesting discussion of instability in which the authors define a general distance function to measure the difference between the tentative and final schedules and suggest that this function should be defined for each hospital individually to reflect that hospital's unique requirements and goals [10]. To minimize instability, cost-based objective functions used during the rescheduling process may include a term dedicated to the cost associated with postponing and preponing surgeries which can be different [37, 41] and potentially nonlinear in nature. When rescheduling using a MILP, a measure of instability can be minimized using a soft constraint or capped by introducing a hard constraint (e.g. [38, 39]).

Similar to our research, other papers choose not to minimize or control instability, but discuss the effect of various policies on measures of instability [35, 44]. Dellaert and Jeunet use data from a thoracic surgery center that treats both emergency and non-emergency patients and consider various combinations of the following approaches to perform tactical planning under uncertainty: overplanning (essentially creating a robust schedule by adding slack), rescheduling on a periodic basis, and allowing for flexibility in the types of surgeries being performed [44]. The authors define instability as a weighted sum of various instability-related costs such as the number of cancelled operations and the number of plan changes [44]. The trade-off between instability and average wait

time (the chosen efficiency metric) [44, Figure 1] is clearly defined. Day of surgery rescheduling with uncertainty in durations is explored in [35] where the rescheduling method is a right shift heuristic and three different event-driven rescheduling policies are explored. The authors consider instability in terms of both difference in start times, difference in completion times, number of reschedules, and number of surgeries rescheduled [35]. Rescheduling is only performed if the procedure has not yet finished and is running late by more than the allowable amount, known as the criterian amount, which is a tunable parameter and also the amount the heuristic right shifts by [35]. An indirect instability measure in the terms of overwhelming the staff with unnecessary schedule updates is also discussed [35].

2.3.5 Other Health Care Settings

Optimization in health care settings is becoming increasingly important given the aging population and subsequent high health care costs [27]. This vast area of research covers topics such as appointment scheduling, capacity planning, OR scheduling, and staff scheduling. While our motivating problem is OR scheduling, the literature on other health care applications can still provide insight into OR scheduling problems. For example, Corlu et al. use DES to model nurse dispatching for patient requests (such as medication distribution or bed turn over) while optimizing patient outcomes and reducing costs by assigning priorities to the patients where the priority increases as a function of time [45]. Increasing a patient's priority over time can be applied to OR scheduling as patient conditions may worsen over time. Modelling health care applications as machine scheduling problems can lead to interesting insights and the adaption or adoption of pre-existing solution methods. Hahn-Goldberg et al. model a chemotherapy clinic appointment scheduling problem as a flexible flowshop problem with the objective of minimizing makespan [46]. The authors make use of historical data to create a template schedule which is updating following realizations of demand uncertainty (dynamic arrivals and cancellations) [46]. In a similar setting, Kortbeek et al. consider appointment scheduling for outpatient treatment centers with both scheduled appointments and dynamic patient arrivals with the goal of identifying the best cyclic appointment schedule [47]. Based on a case study with two CT scanners, the scanners were only scheduled concurrently in the beginning of the day in the best clyclic schedule [47] likely due to the idea that early idle times lead to

decreased flexibility in the schedules [22]. Additionally, Kortbeek et al. [47] include measures of patient behaviour and satisfaction by allowing a parameter to specify how long patients are willing to wait before being seen. Hooshangi-Tabrizi et al. utilize two integer programs to first schedule chemotherapy appointments from a waiting list of patients and subsequently reschedule on a daily basis to cope with uncertainty in demand, patient treatment plans (including duration of treatment), patient cancellations, and staff resource availability [48]. Although duration of treatment is considered to be uncertain during the early planning phases, the duration is still known before the appointment begins [48]. The stochastic nature of surgery durations is a key differentiating factor between appointment and OR scheduling problems [30, 46, 47].

2.4 Positioning Our Work in Existing Literature

In their survey paper, Samudra et al. highlight a need for more reproducible research in the area of OR scheduling by making the data and models publicly available [2]. However, publicly releasing health care data is challenging due high privacy requirements. The authors also suggest that researchers who assume a particular probability distribution for a process should prove it to be the best fit distribution or show that the assumption has minimal effects on the conclusions [2]. In their literature review, Wang et al. prove that there is a lack of consensus regarding the appropriate distribution to model surgery durations of inpatient surgeries [30]. They show that the Lognormal distribution is most commonly employed while Empirical, Normal, Uniform, and Exponential distributions are assumed less regularly [30, Table 4]. Of the technical papers discussed in this literature review which made an assumption regarding the probability distribution of surgery durations, the most commonly assumed distribution is the Lognormal distribution [34, 38, 40, 49, 50], with some papers assuming an Exponential distribution [14, 33, 51], Normal distribution [33], Uniform distribution [49], Triangular distribution [15], Gamma distribution [45] and additive distributions [41]. With the exception of [52, 53], all OR scheduling papers we reviewed assume a probability distribution for surgery durations, which contradicts the need to evaluate the effect of this choice on the results.

Using a large dataset of surgery durations, May et al. compare Normal and Lognormal distributions and demonstrate that the Lognormal distribution fits the given data better [54]. Varmazyar et al. highlight the benefits of using a Continuous Phase-Type distribution compared to a Normal or Lognormal distribution to model surgery and Post-Anesthesia Care Unit durations [53]. Choi and Wilhlem compare three different probability distribution assumptions for the surgery durations: Lognormal, Gamma, and Normal [52]. However, the work of [52] is limited as they focus on finding the optimal rule to sequence up to three surgeries. Although our work is most closely related to that of [52], we extend the existing OR scheduling literature literature by comparing the effects of different probability distribution assumptions for surgery durations over a longer period of time (approximately one week).

To the best of our knowledge, this thesis answers a question which has not been answered in the existing literature: to what extent does the assumption of a single statistical distribution for processing times affect the efficiency, specifically final total weighted tardiness (FTWT), and instability when rescheduling is utilized to handle uncertainty in the processing times and dynamic arrivals. While there are general scheduling papers which study the effect of various rescheduling policies [9, 15, 18], none of the above study the joint effect of the assumptions of statistical distributions to model processing times and rescheduling policies. Hozak and Hill show that modelling choices affect the ideal frequency of rescheduling [20, Table 2], however they do not consider an analysis of different statistical distributions for processing times.

2.5 Conclusions

In this chapter, we provided a concise overview of various definitions and categorizations of scheduling under uncertainty, and reviewed the literature on rescheduling with a focus on single machine environments. A unified terminology was defined to compare various instability metrics found in the existing literature. We then described the OR scheduling problem, evaluated trends in OR scheduling research, and reviewed select OR scheduling literature with a focus on literature that included an aspect of rescheduling. Finally, we positioned our work in the existing scheduling and OR scheduling literature.

Chapter 3

Problem Setting

In this chapter, we describe the motivating problem, present the single machine scheduling problem of interest, and establish definitions and notation.

3.1 Motivating Problem

The general topic of operating room (OR) scheduling is described in this section. We then present a simplified OR scheduling problem which motivates this work, and finally map the simplified OR scheduling problem to the single machine scheduling problem of interest.

3.1.1 Operating Room Scheduling Problem

OR scheduling is concerned with the assignment of surgeries to ORs and times to the surgeries. In medium to large sized hospitals, there may be over a dozen ORs requiring scheduling [27]. One of the major challenges in OR scheduling arises due to the inherent uncertainty in the surgery durations. Dynamic arrivals of emergency surgeries may provide a second source of uncertainty. Furthermore, surgeries require resources such as the room itself, staff, and equipment. Pinedo provides an example of a set of resources as the OR, a specific surgeon, and a specific anesthesiologist [27]. In practice, the OR scheduling problem is extremely complex due to the intertwined schedules of various departments, the impact of the schedule on downstream units (such as the intensive care unit), and the simultaneous consideration of many different objectives [27].

The OR scheduling problem varies greatly depending on the specific hospital or application. This is demonstrated in the review paper by Samudra et al. [2], which categorizes 216 technical papers on OR scheduling published between 2000 and 2014. They classify the papers based on patient characteristics, performance metrics used, level of decisions being made, consideration of upstream or downstream resources, the types of uncertainty included, the research methodology used, and the inclusion of data [2].

3.1.2 Simplified Operating Room Scheduling Problem

We consider the scheduling of a single OR with stochastic surgery durations and dynamic arrivals of emergency surgeries as our motivating problem. Given the challenging nature of OR scheduling and our inclusion of uncertainty in both arrivals and durations, we make several simplifications, described below.

Dynamic Arrivals

We assume that dynamic arrivals correspond to the time at which a surgical procedure is requested for the patient and follow a stationary Poisson process.

Priorities

Similar to the Canadian Triage and Acuity Score (CTAS) used in emergency departments [55], we assume priorities are assigned to patients through an evaluation process called triage. Triaging surgeries has become increasingly important with the limited resources and backlog caused by the COVID-19 pandemic, leading many organizations, including the American College of Surgeons, to release guidelines for the triage of non-emergent surgeries [28, 56]. Similarly, a recent publication presents triage guidelines for endocrine surgery as a cancer treatment in light of the COVID-19 pandemic [57]. To simulate the results of triaging, we sample a patient's priority from a discrete Uniform distribution at the time of arrival and assume the priority is static, meaning it does not change over time.

Due Dates

Guidelines of acceptable wait times for specific types of surgeries may be provided by a governing organization [2]. We use the term due date to correspond to the time the surgery should be completed by to avoid possible health risks from a delayed surgery [30].

Uncertain Durations

Surgeries have uncertain durations as there are many complications that can arise during a surgical procedure. For the purpose of this thesis, we make the simplifying assumption that there are two points in time at which information about the surgery duration becomes available: when the surgery request first arrives (estimated duration is supplied) and when the surgery is completed (true duration becomes known). The concept of modelling surgery durations as an estimation followed by a true duration which becomes known at the time of completion of the surgery has also been used in similar OR rescheduling papers which utilize simulation [35, 41]. However, the *distributions* of the estimated and true surgery durations are assumed to be known and belong to a set of candidate distributions. We assume the processing times of all surgeries follow a single distribution.

Resources

In this thesis, we do not include resources other than the single OR. Therefore, we do not take into account how the OR schedule interacts with upstream or downstream units.

Setup Times and Patient Availability

We make the simplifying assumption that all surgeries can be scheduled immediately upon their arrival and do not consider setup times. Additionally, we assume that surgeries cannot be moved forward if an OR becomes available without a rescheduling action taking place. Although surgeries may get cancelled or postponed long term, due to changing patient conditions for example, this thesis does not allow for surgery cancellations.

3.1.3 Mapping the Operating Room Scheduling Problem to Single Machine Scheduling Problem

We describe how the simplified OR scheduling problem outlined in Section 3.1.2 can be mapped to a single machine scheduling problem. Surgeries are modeled as jobs and the OR as a single machine. Since only one patient can be operated on in a single OR at a time, the machine has single capacity. Additionally, given that surgeries cannot be stopped once they are started, preemptions are not allowed. The time at which a surgical procedure is requested for a particular patient is modelled as the dynamic arrival time of the corresponding job. Jobs have all the remaining attributes of surgeries, i.e., priorities, due dates, and stochastic processing times.

3.2 Problem Definition

Although this thesis is motivated by OR scheduling, we use terminology consistent with standard scheduling literature to keep our findings general. In this section we formalize the single machine scheduling problem motivated by the simplified OR scheduling problem.

We consider a single machine scheduling problem with stochastic processing times and dynamic arrivals in a non-preemptive environment. We assume a capacity of one on the single machine. Jobs arrive dynamically following a stationary Poisson process. Each job j has a priority, denoted as w_j, which is known at the time of arrival (r_j), and a due date, denoted as d_j. In this work, we assume that the job is released when it arrives and therefore choose to use the notation consistent with release times (r_j). Although each job has an estimated processing time, $\hat{p}_j$, the true processing time, p_j, does not become known until the job has finished processing. Using the notation of Graham et al. [58], we represent this problem as the stochastic version of $1|r_j|Z$ with dynamic arrivals where Z represents the choice of objective function.

To deal with the uncertainty in the arrivals and processing times, we reschedule at varying frequencies based on certain rescheduling policies. Since we generate an initial schedule and occasionally update it, this falls into the category of predictive-reactive approaches [8, 11, 17]. Using a computational framework, we evaluate the effects of different rescheduling policies and assumptions of a specific statistical distribution for the processing times on efficiency and total instability.

The literature on efficiency and instability related metrics is explored in Section 2.2.1 while the formal definitions of total instability and efficiency used in this thesis are found in Sections 3.3.3 and 3.3.4 respectively. To focus on the effects of the processing time distributions, we make assumptions regarding the distributions of the remaining job attributes.

Processing Times

We assume a job's estimated processing time, $\hat{p}_j$, is known at the time of arrival while the job's true processing time, p_j, does not become known to the scheduler until the completion time of the job. When determining the effect of the assumptions of a specific statistical distribution for the processing time on efficiency and total instability, we are particularly interested in the shape of the distributions.

We examine the following five distributions for the estimated processing times ($\hat{p}_j$):

- Uniform: processing times are evenly spread out within a possible range.
- Exponential: There is a long right tail in the processing time estimate.
- Left Truncated Normal: The distribution is bell-shaped. We left truncate the distribution at zero to avoid negative processing times.
- Lognormal: There is a long right tail in the processing time estimates meaning some durations are estimated to be longer than the majority of the durations.
- Bimodal: Some processing times are short (i.e. no complicating factors are expected) and others are longer (i.e. complicating factors are expected) but there are minimal processing times in the middle.

In their experiments, Cowling and Johansson considered $\hat{p}_j$ to be drawn from a Uniform distribution with p_j drawn from the same distribution at a different point in time [12]. Similarly, Bidot et al. assume $\hat{p}_j$ is drawn from a Normal distribution and p_j is drawn from a renormalized distribution where both distributions use min and max bounds to truncate the Normal distribution such that $\hat{p}_j$ can change to p_j in the middle of the processing (see Figure 9 in their paper) [18]. Lastly, Larson

and Pranzo consider p_j to be drawn from a Uniform distribution ranging from $\hat{p}_j$ to $\hat{p}_j + \hat{p}_j\epsilon$ where ϵ is used to control the level of uncertainty in the original estimate [9].

To allow for stochastic processing times, we consider two candidate distributions for p_j: the Normal distribution and the Uniform distribution. Both distributions are centered at $\hat{p}_j$ and have user-defined standard deviation. We again left-truncate the Normal distribution at zero to avoid infeasibilities caused by negative true processing times. With this formulation, $\hat{p}_j$ and p_j can be drawn from any distribution and the distributions do not need to be the same. Expanding on the ideas of Larson and Pranzo [9], we allow for two optional parameters ϵ^- and ϵ^+, which control the level of uncertainty in the estimated processing times. When supplied by the user, these allow for the computation of an upper and lower bound on the true processing time such that $p_j \in [\hat{p}_j - \hat{p}_j\epsilon^-, \hat{p}_j + \hat{p}_j\epsilon^+]$. These upper and lower bounds are used to renormalize the distribution of p_j.

Choice of Objective Function

Every time a rescheduling action is performed, we minimize total weighted tardiness as the performance metric of interest to the given problem setting. The priorities are used directly in the computation of weighted tardiness such that the highest priority jobs contribute to total weighted tardiness more than low priority jobs. The tardiness of job j, denoted T_j, is defined as the time that passes between a job's due date and the job's completion time ($T_j = max\{0, d_j - C_j\}$ where C_j is the completion time of job j).

Final Problem Formulation

Taking all the assumptions into consideration, we again use Graham et al.'s notation [58] to specify the problem definition in this thesis as the stochastic version of $1|r_j|\sum w_jT_j$ with dynamic arrivals. Pinedo expands on the notation by Graham et al. [58] by introducing notation for random variables [59]. Using this expanded notation, we further specify our problem as $1|R_j \sim exp(\lambda), X_j \sim \mathbf{P}|\sum w_jT_j$ where R_j represents the random variable for the inter-arrival time, X_j represents the random variable for the processing time, and $\mathbf{P}$ represents a processing time distribution set (estimated processing time distribution and true processing time distribution) from the candidate distributions outlined in this section.

3.3 Definitions

Here, we first define two terms that are key for describing our computational framework. We then define the two primary metrics of interest: total instability and efficiency.

3.3.1 Scenario

Throughout this thesis, we define a *scenario* as a specific set of assumptions regarding the underlying statistical distributions of the processing times and a chosen rescheduling policy. For example, in the stochastic case, we might have the scenario of drawing $\hat{p}_j$ from a Bimodal distribution (with parameters specified in Section 5.1.4), p_j from a Uniform distribution (centered at $\hat{p}_j$ and with standard deviation described in Section 5.1.4), and a periodic rescheduling policy with a period of 180 minutes.

3.3.2 Period of Interest

Due to the varying rescheduling policies, using a time-based simulation stopping criteria will result in different sets of completed jobs under different scenarios, which would prevent us from directly comparing results across scenarios.

We therefore utilize a state-based termination criteria which is dependent upon the completion of a specific set of jobs. We define a *period of interest*, denoted as $[0, \tau]$ where τ is a user-defined variable, in which all jobs that arrive within this interval are flagged as being jobs of interest. We denote this set of flagged jobs as $\mathbf{J}^*$ and terminate the simulation when all jobs in $\mathbf{J}^*$ are completed. The simulation end time, denoted as T, is therefore dynamically determined and can vary between scenarios. Figure 3.1 summarizes these concepts using a timeline. The relationship $0 < \tau \leq \mathrm{T}$ must hold regardless of the scenario.

3.3.3 Total Instability

When performing a rescheduling action, we allow rescheduling of jobs that have been previously assigned a start time but have not yet started. These rescheduling actions can occur at different frequencies and points in time depending on the rescheduling policy (see Section 4.2). As opposed

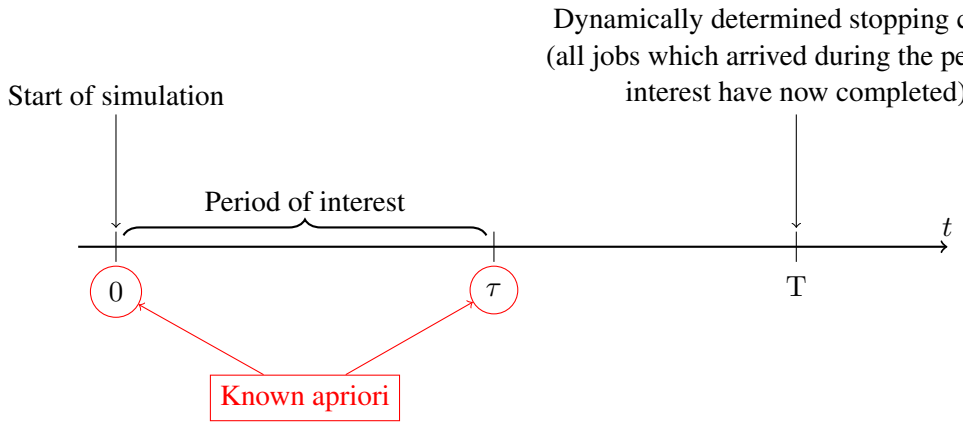


Figure 3.1: Timeline representation of *period of interest* and dynamically determined stopping criteria.

to scheduling problems that are deterministic and static, which only require a measure of efficiency to evaluate the quality of the schedule, scheduling problems with uncertainty require consideration of both efficiency and instability to measure the quality of the solution. We define *total instability* as a measure of schedule disruptions caused by the rescheduling actions. An example of computing this metric can be found in Appendix A.1 while Section 2.2.1 explores related metrics found in the literature.

We first define *instability* to measure the magnitude of a change in a schedule after a single rescheduling action for jobs in $\mathbf{J}^*$. As explained in Section 3.3.2, computing this metric over $\mathbf{J}^*$ allows us to make a more direct comparison between the results when employing different rescheduling policies.

Let $\mathbf{J}_t$ be the set of all jobs which have arrived at time t. We additionally define $\mathbf{J}_t^*$ to be the set of all jobs which have arrived by time t and which arrived during the period of interest ($\mathbf{J}_t^* \subseteq \mathbf{J}_t$). Consider a rescheduling action r that takes place at time t_r where S_r represents the previous schedule and S_r' represents the new schedule following a rescheduling action. Suppose that schedule S_r contains the set of jobs $\mathbf{J}_r$ with completion times $C_j \ \forall \ j \in \mathbf{J}_r$. Similarly, the schedule S_r' contains the set of jobs $\mathbf{J}_r'$ with completion times $C_j' \ \forall \ j \in \mathbf{J}_r'$. Since it is assumed that all jobs which have arrived by the time of rescheduling will be scheduled, $\mathbf{J}_r' = \mathbf{J}_{t_r}$. Also note that our problem definition does not allow for the cancellation of jobs and therefore $\mathbf{J}_r' = \mathbf{J}_r \cup \mathbf{J}_r^\eta$ where $\mathbf{J}_r^\eta$ are jobs that have arrived between the creation of S_r and the creation of S_r'. To simplify

this discussion further, let $\mathbf{J}_r^*$ represent the set jobs in schedule S_r which also arrived during the period of interest and let $\mathbf{J}_r'^*$ represent the set of jobs scheduled in S_r' which also arrived during the period of interest.

As the simulation progresses, we update the previously created schedule, S_r, with the most recent information. Under certain circumstances, a right shift heuristic is employed in order to maintain the feasibility of a schedule. Suppose we have a rescheduling action, r, taken at time t_r. In order for a right shift heuristic to be used between t_{r-1} and t_r, one of the following two cases must have occurred:

- Case 1: Suppose there exist two consecutive jobs i and j. Upon completion time of job i, the true processing time becomes known and $p_i > \hat{p}_i$. Furthermore, the scheduled idle time between the two jobs in S_r must be less than $p_i - \hat{p}_i$. Here the right shift heuristic is employed to ensure the schedule remains feasible.

- Case 2: Assume a rescheduling action is triggered at time t_r but there is a job i currently in progress with scheduled completion time $C_i < t_r$. Although the job has not finished and the true processing time is not known at this point in time, the relationship $p_i > \hat{p}_i$ is clear to the scheduler. The only information regarding the processing time that the scheduler gains access to at time t_r is that the processing time of job i has taken $t_r - B_i$ time units where B_i is the start time of job i. The amount the current processing time exceeds the estimated processing time is given by $(t_r - B_i) - \hat{p}_i$. Here the right shift heuristic is employed to ensure the most up to date information is used by updating $\hat{p}_i = t_r - B_i$. Further assume there exists a subsequent job j with $B_j < t_r$. Then, even though the true processing time p_i is not known, the right shift heuristic is used to maintain feasibility similar to the previous case.

We then define *instability* for the rescheduling action r, taken at time t_r, as the sum of the absolute differences between the completion times of jobs in the previous schedule, S_r, and the completion times of jobs in the new schedule, S_r', for all jobs which arrived during the period of interest and were scheduled in S_r. The equation for *instability* is shown in Equation 1.

$$I(S_r, S_r') = \sum_{j \in \mathbf{J}_r^*} |C_j - C_j'| \tag{1}$$

In the literature, some metrics of instability use both start and completion times such as the work by Cowling and Johansson [12]. In our case, since S_r is updated as the simulation progresses, schedules S_r and S'_r use the same processing times for all common jobs. Therefore, considering one of completion and starting times is sufficient for the definition of instability given in Equation 1. Furthermore, the summation in Equation 1 is over the set $\mathbf{J}^*_r$ so that we only include jobs which arrived within the period of interest and which were scheduled in S_r. In other words, we do not include jobs in J^{η}_r as these jobs have not yet been scheduled and therefore should not contribute to the instability.

We assert that the right shift heuristic does not contribute to the instability metric defined in Equation 1. Instability is computed only when a rescheduling action takes place. Assume a right shift heuristic was used between rescheduling actions $r-1$ and r which takes place at time t_r. Since the right shift heuristic has been applied to the previous schedule, S_r, prior to t_r, no changes are necessary for the new schedule, S'_r to become feasible. Therefore, the only changes made to S_r to form S'_r occur as a result of re-sequencing the jobs to improve the objective function in light of new information or to insert newly arrived jobs $\mathbf{J}^{\eta}_r$ into the schedule.

Our instability metric, although proposed prior to reading the work of Wu et al., closely aligns with metric presented in [24]. Both metrics compare the new schedule to the right shifted previous schedule. The main difference between the two metrics is that the instability defined in Equation 1 uses completion times while the instability defined in [24] uses start times. Moreover, the instability metric in [24] is defined to accommodate uncertainty in the form of delays (e.g. machine breakdowns, additional processing times, and inserting jobs) whereas our definition, coupled with the discrete-event simulation (DES) framework which updates the previous schedule with the most current information, can also handle uncertainties such as a reduction in a job's processing time.

If $\mathbf{R}$ is the set of all rescheduling actions taken, then we define *total instability* to be the sum of the individual instability values across all rescheduling actions as shown in Equation 2.

$$I_{tot} = \sum_{r \in \mathbf{R}} I(S_r, S'_r) \tag{2}$$

3.3.4 Efficiency

Efficiency measures the performance of the system under a given rescheduling policy. Let $Z(S, \mathbf{J})$ refer to the objective function of interest which depends on schedule S and is to be computed over the set of jobs $\mathbf{J}$. Given a simulation run which ends at time T, efficiency is computed as shown in Equation 3 where S_{r_T} is the final schedule with the most updated job information available and $\mathbf{J}^*_\mathrm{T}$ is the set of jobs which arrived during the period of interest.

$$E = Z(S_{r_T}, \mathbf{J}^*_\mathrm{T}) \tag{3}$$

Efficiency is a general term which can incorporate any objective function of interest. As previously mentioned, we are interested in total weighted tardiness (TWT) which is computed by summing weighted tardiness across a set of jobs. We therefore define *final total weighted tardiness* (FTWT) as our specific efficiency metric which will be used throughout this thesis. In this case, a better efficiency corresponds to a lower FTWT metric. Given a simulation run which ends at time T, we sum the weighted tardiness for the jobs which arrived during the period of interest. The term *final* is used here to emphasize that efficiency, unlike instability which can be recorded at each rescheduling action, cannot be computed until the end of the simulation. Equation 4 demonstrates the computation of FTWT where the set $\mathbf{J}^*_T$ refers to the set of jobs which arrived during the period of interest and $T_j = max\{0, d_j - C_j\}$.

$$FTWT = E_{TWT} = \sum_{j \in \mathbf{J}^*_\mathrm{T}} w_j T_j \tag{4}$$

3.4 Conclusion

This chapter establishes the motivating problem of OR scheduling and its characteristics and maps a simplified OR scheduling problem to the single machine scheduling problem. We then define the problem of interest: minimizing total weighted tardiness in a non-preemptive single machine environment with dynamic arrivals and stochastic processing times. Finally, we define terms which are used throughout this dissertation as well as the two primary metrics of interest: total instability and efficiency.

Chapter 4

Methodology

Due to the dynamic and stochastic nature of the problem defined in Section 3.2, we create a computational framework using concepts from both scheduling and simulation literature to analyze the effects of the various rescheduling policies and processing time distributions on efficiency, specifically final total weighted tardiness (FTWT), and total instability. This chapter begins by describing the discrete-event simulation (DES) framework and the different rescheduling policies which are implemented. We then examine the optimization component used for scheduling.

4.1 Discrete-Event Simulation Framework

The logic behind a portion of the DES framework is presented in Figure 4.1. As part of the terminating DES framework, we use appropriate statistical techniques from simulation literature by generating initial states and performing the method of replication [60] as outlined in Sections 5.2.1 and 5.2.2 respectively. Furthermore, we employ common random numbers, generated for each job attribute separately, as a variance reduction technique [60, 61]. It should be noted that Figure 4.1 assumes that all the experimental setup stages (determining the pseudo warm-up period, generating common random numbers and initial states, and determining the number of replications) have been completed. Furthermore, the diagram shown in Figure 4.1 represents a single replication.

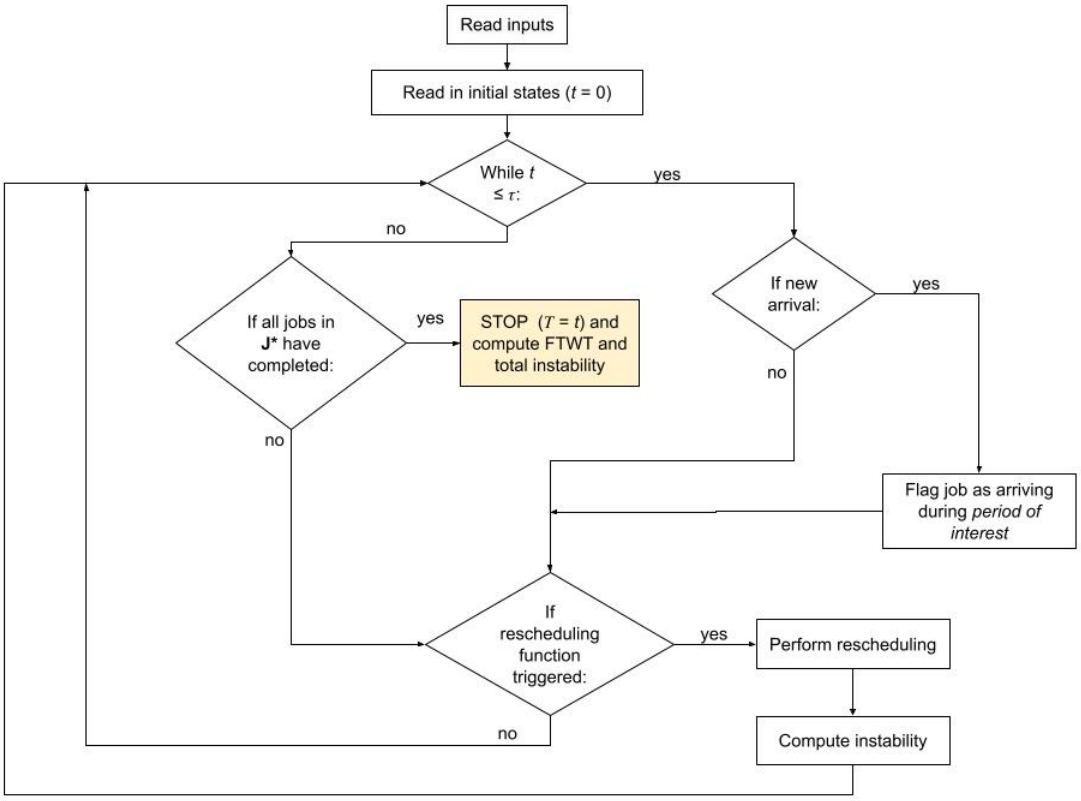


Figure 4.1: Flowchart demonstrating the logic behind the conditional statements of the DES framework.

The first step is to read in the inputs provided by the user which are listed and explained in detail in Appendix A.2. Based on these inputs, the correct initial states and common random numbers are located and read into the simulation framework.

If the period of interest, τ, has not yet been met, then any new arrivals are flagged as arriving during the period of interest. Conversely, if τ has been reached, then any newly arrived jobs are not flagged.

Based on the rescheduling policy, certain events or periodic time intervals will set off the rescheduling trigger, in which case a rescheduling action is performed. At each rescheduling action, we solve the subproblem described in Section 4.4. Updating the job records with the most recent information is necessary for solving the subproblems. Although this process is not shown in Figure 4.1, updates to the job information are completed after any event, such as an arrival or completion

of a job. Once the rescheduling action is finalized and a new schedule is obtained, the instability of that particular rescheduling action is computed.

If the period of interest has been reached and the set of all flagged jobs ($\mathbf{J}^*$) have been completed, the dynamically determined stopping criteria, denoted as T and shown in yellow in Figure 4.1, is met. Once the simulation is complete, we compute FTWT and total instability using the common job subset $\mathbf{J}^*$.

4.2 Rescheduling Policies

We evaluate three categories of rescheduling policies: periodic, event-driven, and hybrid [8]. The periodic category of rescheduling policies refer to those which are time-based while the event-driven rescheduling policies are based on the occurrence of certain events. Finally, the hybrid rescheduling policy is a combination of the periodic and event-driven policies. Below we outline the different rescheduling policies implemented from each of the three categories.

- Periodic:
    - A rescheduling action is triggered every Δ time units where Δ is the period.
- Event-Driven:
    - Job arrival: any new job that arrives to the system triggers a rescheduling action.
    - Completion: a rescheduling action is triggered at the time of completion of any job.
    - Started processing: a rescheduling action is triggered at the time of the start of any job's processing.
    - Priority arrival: the arrival of a job whose priority is within a particular priority set triggers the rescheduling action.
    - Number of arrivals: rescheduling is performed after every n arrivals.
    - Number of unscheduled jobs: a rescheduling action is triggered after the list of arrived but unscheduled jobs reaches a specified number, l [1].

[1] If strictly event-driven, the number of arrivals and number of unscheduled jobs policies with $n = l$ are identical. However, if used within a hybrid policy these two conditions for rescheduling will lead to different behaviour.

- Schedule completion: the completion of the current schedule triggers a rescheduling action.

- Hybrid:
  - Any of the event-driven policies used in conjunction with a periodic policy constitutes a hybrid policy. For example, we could employ a hybrid policy which reschedules on high priority arrivals and every 300 minutes ($\Delta = 300$).

Refer to Appendix A.2 for an explanation of the inputs required for each rescheduling policy.

4.3 Defining Subproblems

Whenever a rescheduling action is triggered, we use a rescheduling method to update the existing schedule in response to realizations of uncertainty [8]. We refer to this updating of the existing schedule as solving a subproblem (see module 'Perform rescheduling' in Figure 4.1). Vieira et al. outline two main categories of rescheduling methods: schedule generation, consisting of nominal schedules and robust schedules, and schedule repair consisting of right-shift rescheduling, partial rescheduling, and complete regeneration [8]. In the DES framework, the 'Perform rescheduling' module is independent and could use any rescheduling method desired. Based on the rescheduling method used, the goal of a rescheduling action may differ such as rescheduling for feasibility versus optimality.

As the simulation progresses, the job data is updated to ensure the subproblems include accurate data. Therefore, the subproblems consist of the set of jobs which have arrived and all their attributes as well as the state of the simulation (e.g. which jobs have started or completed and the current simulation time).

In this dissertation, we assume each subproblem is a deterministic scheduling problem where the most up to date information regarding the processing times is used to schedule the jobs. The rescheduling method used to solve the subproblems is described in the following section.

4.4 Method for Solving Subproblems

In this section, we describe the method of complete rescheduling used to solve the subproblems at every rescheduling action. For the purpose of this thesis, we implement complete rescheduling using a mixed integer programming (MIP) model as it allows us to find an optimal solution for the majority of the subproblems, thus reducing the variability between scenarios due to suboptimal subproblem solutions. The complete rescheduling method, under the category of schedule repair methods, reschedules all jobs not already started on the machine, even if they are not directly affected by a realization of uncertainty [8]. For example, suppose we employ the job arrival rescheduling policy. Even though the pre-existing jobs are not directly affected by the arrival of the new job, all pre-existing jobs that have not yet started will be considered for rescheduling.

The subproblems are modelled using a basic MIP sequence-based formulation model which has been adapted to our specific problem. Most notably, constraint 6 ensures started jobs cannot change their scheduled start times while constraint 7 guarantees jobs that have not yet started cannot start earlier than the current time. A major disadvantage of using MIP is the computational effort required [8]. Thus, we enforce a user-specified time limit to the subproblems to ensure computational feasibility. In the case where a suboptimal solution is returned due to the time limit, we flag this subproblem for further analysis and record the optimality gap.

When solving the subproblems, we assume deterministic processing times. The value of the deterministic processing time assumed for a job j depends on the state of the job j at time t. Therefore, we define the *appropriate processing time* of job j, denoted as $\pi_j(t)$, to represent the processing time of job j which is used when solving the subproblems at time t. Let $\mathbf{J}_t^s$ be the set of jobs whose processing has started at time t and $\mathbf{J}_t^c$ be the set of jobs whose processing has completed at time t. Furthermore, let B_j be the previously scheduled start time of job j and C_j be the previously scheduled completion time of job j.

There are three possibilities for the *appropriate processing time* of job $\pi_j(t)$. If j has completed by time t, the true processing time is used. However, if j is currently in progress at time t and was meant to be completed based on the previous schedule ($C_j < t$), then the estimated processing time plus the overtime already occurred is utilized. Lastly, if the job has not yet started or has started but

does not satisfy the aforementioned condition of $C_j < t$, then the estimated processing time is used. Equation 5 shows the formula for determining $\pi_j(t)$.

$$\pi_j(t) = \begin{cases} p_j, & \text{if } j \in \mathbf{J}_t^c \\ t - B_j, & \text{if } j \in \mathbf{J}_t^s \setminus \mathbf{J}_t^c \wedge (C_j < t) \\ \hat{p}_j, & \text{otherwise} \end{cases} \tag{5}$$

Using the notation from Pinedo [62], the subproblems described in this section are given by $1|r_j|\sum w_j T_j$.

4.4.1 Variable Definitions

For this section, assume that we are performing a rescheduling action at time t. The former schedule is denoted as S while the new schedule created from the rescheduling action will be denoted as S'. The following input sets are maintained by updating the job information throughout the simulation.

Input Sets

$\mathbf{J}_t$: The set of jobs available to be scheduled at time t (all jobs arrived by time t).

$\mathbf{J}_t^*$: The set of jobs which have arrived by time t and which have been flagged as arriving during the period of interest (arrived in the interval $[0, \tau]$).

$\mathbf{J}_t^s$: The set of jobs whose processing has already started at time t ($\mathbf{J}_t^s \subseteq \mathbf{J}_t$).

$\mathbf{J}_t^c$: The set of jobs which have already been completed at time t ($\mathbf{J}_t^c \subseteq \mathbf{J}_t^s \subseteq \mathbf{J}_t$).

Input Parameters

t: The time at which the rescheduling action is taking place.

$\pi_j(t)$: The appropriate processing time of job j at time t (see Equation 5).

r_j: The arrival time of job j.

d_j**:** The due date of job j.

w_j**:** The importance weight of job j.

M**:** The dynamically determined "Big M" (see Section 4.4.3).

C_j**:** The completion time of job j in S (updated as the simulation progresses).

B_j**:** The starting time of job j in S (updated as the simulation progresses).

T_j**:** The tardiness of job j in S (also updated as the simulation progresses).

Sequencing Decision Variable

$$x_{ij} = \begin{cases} 1, & \text{if job } j \text{ is performed (not necessarily immediately) after job } i \\ 0, & \text{otherwise} \end{cases}$$

Other Decision Variables

C'_j = The completion time of job j in S'.

T'_j = The tardiness of job j in S'.

B'_j = The start time of job j in S'.

4.4.2 Model

This section establishes the objective function and constraints in the MIP model used to solve the subproblems.

Objective Function:

Minimize total weighted tardiness of *all* jobs which have arrived by time t.

$$min \sum_{j \in \mathbf{J}_t} w_j T'_j$$

Subject To (Constraints):

(1) Constraint to ensure either job i is scheduled after job j, or job j is scheduled after job i.

$$x_{ij} + x_{ji} = 1 \quad \forall\, i, j \in \mathbf{J}_t : i < j$$

(2) Constraint to ensure no overlap occurs on the single machine.

$$C'_j \geq C'_i + \pi_j(t) - M(1 - x_{ij}) \quad \forall\, i, j \in \mathbf{J}_t : i \neq j$$

(3) Define x as a binary variable.

$$x_{ij} \in \{0, 1\} \quad \forall\, i, j \in \mathbf{J}_t : i \neq j$$

(4) Constraint to ensure non-negative completion times and enforce arrival times [2].

$$C'_j \geq \pi_j(t) + r_j \quad \forall\, j \in \mathbf{J}_t$$

(5) Constraints to define tardiness.

$$T'_j \geq C'_j - d_j \quad \forall\, j \in \mathbf{J}_t$$

$$T'_j \geq 0 \quad \forall\, j \in \mathbf{J}_t$$

[2] Although only jobs which have already arrived will be available (and known) to the scheduler, we still include the release times of jobs in the model for generality.

(6) Constraint to define start times for jobs which have already started (including, by extension, completed jobs) to ensure that the job's start time is not altered in the new schedule.

$$C'_j = B_j + \pi_j(t) \quad \forall\, j \in \mathbf{J}_t^s$$

(7) This constraint ensures that jobs which have not yet started cannot start earlier than the current simulation time. Note that $\pi_j(t) = \hat{p}_j \;\; \forall \;\; j \notin \mathbf{J}_t^s$ (Equation 5).

$$C'_j \geq t + \pi_j(t) \quad \forall\, j \notin \mathbf{J}_t^s$$

4.4.3 Big M Details

Since the simulation framework iteratively calls the MIP model with subproblems of varying size, the choice of the penalty constant, M, is of particular importance. M must be large enough to guarantee correctness [63] of the no overlap constraint (constraint 2) without eliminating any feasible solutions. However, if M is chosen too large, the search space may be unnecessarily large resulting in longer run times.

In their work studying MIP formulations for job shop scheduling, Ku and Beck [63] set the value of M equal to the sum of all processing times. However, setting M equal to the sum of all processing times does not apply to this problem as the dynamic arrival of jobs may result in forced idle times.

The maximum possible idle time for the single machine at some arbitrary point in time, t, would occur if no jobs were processed over the time interval $[0, t]$. Equation 6 extends the idea presented by Ku and Beck [63] by representing M as a function of time to account for the maximum possible idle time.

$$M(t) = \sum_{j \in \mathbf{J}_t} \pi_j(t) + t \tag{6}$$

M can be further reduced by considering the set of jobs which have completed at time t, given by $\mathbf{J}_t^c$. Taken together, the last two terms in Equation 7 represent a tighter upper bound on the total idle time. We calculate the value of M within the rescheduling framework according to Equation 7

and pass the result as a parameter to the MIP model.

$$M(t) = \sum_{j \in \mathbf{J}_t} \pi_j(t) + t - \sum_{j \in \mathbf{J}_t^c} \pi_j(t) \tag{7}$$

4.4.4 Subproblem Size

For the MIP model presented in Section 4.4.2, if a job has started or completed, its start time is fixed (see Section 4.4.2 constraint 6). Therefore, the computational challenge arises from the number of jobs whose start time is allowed to change. The number of jobs whose sequencing is allowed to change for a rescheduling action at time t is given by $|\mathbf{J}_t \setminus \mathbf{J}_t^s|$. The total number of sequencing decision variables to be determined is then given by $|\mathbf{J}_t \setminus \mathbf{J}_t^s|^2 - |\mathbf{J}_t \setminus \mathbf{J}_t^s|$. The decision variables of C_j', B_j', and T_j' are fixed for all $j \in \mathbf{J}_t^s$. As such, the number of these decision variables that are not fixed is given by $3|\mathbf{J}_t \setminus \mathbf{J}_t^s|$ and the total number of decision variables which are not fixed is given by Equation 8.

$$|\mathbf{J}_t \setminus \mathbf{J}_t^s|^2 + 2|\mathbf{J}_t \setminus \mathbf{J}_t^s| \tag{8}$$

The number of decision variables is directly related to the number of jobs which have not yet started processing, which can also be referred to as a queue. In general, queues occur when non-homogeneous jobs must be processed using limited resources [64]. In this dissertation, we consider a queue to be made up of both jobs waiting to be scheduled and jobs which have been scheduled but not yet started processing. Therefore, the queue length has a substantial impact on the computational complexity of the MIP model. Potential methods to overcome this computational hurdle include using heuristics instead of exact methods, or relaxing the MIP model to find suboptimal solutions.

4.5 Conclusion

In this chapter, we propose a DES framework used to study the problem of interest in the dynamic setting. We outline various rescheduling policies in the periodic, event-driven, and hybrid

categories. We then define subproblems which are solved using the rescheduling method of complete rescheduling. We describe the MIP model used to perform the complete rescheduling and its details. Finally, we discuss the effect the subproblem size has on the computational complexity of the MIP model.

Chapter 5

Experimental Study

The two main goals of this dissertation are to computationally evaluate the effect that rescheduling policies and assumptions made regarding the distributions of job processing times have on efficiency, specifically final total weighted tardiness (FTWT), and total instability. To compare the use of different statistical distributions for processing times, we choose distributions whose first two moments are similar. Given that there are many such distributions, we narrow the scope of our experimental study. Although the first two moments could be assumed without relying on data, we obtain these summary statistics from the Synthea data (see Section 5.1.2) to ensure our findings are more realistic with respect to our motivating problem of OR scheduling. Furthermore, using the Synthea data allows us to compare various distributions with the distribution whose shape best reflects the true data.

This chapter describes the setup and design of the experiments as well as the experimental results. In Section 5.1 we discuss implementation details, outline the collection of surgery duration data from the Synthea data set, fit a distribution to the surgical durations obtained from the Synthea data set, and finally outline the parameters of job attributes used in the experiments. The experimental setup stages are described in Section 5.2 including generating the initial states and determining the number of replications. Finally, the outline and results of experiment 1 are presented in Section 5.3 while the outline and results of experiment 2 are presented in Section 5.4.

5.1 Experimental Preliminaries

5.1.1 Implementation Details

The same implementation is used for both experiments outlined in this chapter. The discrete-event simulation (DES) framework is written in Python version 3.7.9 using SimPy version 4.0.1 [65] for the simulation components. Furthermore, we use IBM's Decision Optimization CPLEX Modeling for Python, DOcplex version 2.23.222 [66], to solve the mixed-integer programming (MIP) model. SciPy version 1.7.3 [67] is used for random number generation.

5.1.2 Data Acquisition

To relate our findings back to the motivational problem, we use Synthea, an open sourced software package provided by the MITRE Corporation, to simulate a synthetic patient population [3][1]. Synthea generates extensive patient-oriented data with tables related to all aspects of a patient's health such as *patients*, *encounters*, *conditions*, *devices*, *allergies*, *medications*, *observations*, *procedures*, *providers*, *payers*, etc. Using Synthea, we generate the records of one million patients and store the tables related to surgeries (*providers*, *procedures*, *patients*, *organization*, and *encounters*) in a relational database which is then queried using SQL.

Based on the 144 unique procedure descriptions in the database, we identify 28 of those to be surgically related. Due to the dynamically arriving jobs in our problem setting, we further limit our search to surgeries in the emergency class (contributing to approximately 11% of the procedure descriptions identified as surgically related). Each unique procedure description contains fields for both an encounter and a procedure duration. To distinguish the duration type most appropriate for surgery as opposed to the entire hospital stay, we create boxplots of these two duration types for each of the unique procedure descriptions in the emergency class. The distinction between the duration types is typically easy to determine, but not consistent throughout the different procedure descriptions.

We choose a single procedure description from the seven different procedure descriptions relating to emergency surgery for further analysis, while considering the following requirements:

[1] Synthea is available at https://github.com/synthetichealth/synthea

Summary Statistic	Value
mean	140.27
median	139.00
min	45.00
max	239.00
range	194.00
standard deviation	55.80
variance	3113.98
skew	0.05
kurtosis	-1.19

Table 5.1: Summary statistics for the laparoscopic removal of gall bladder durations (in minutes) using data generated from Synthea [3].

- The chosen procedure description should be identifiable as a surgical procedure.
- The procedure description should have a sufficient number of occurrences to be able to fit a distribution (preferably $n > 100$).
- There should be a range of different duration values.
- The durations should be validated (either through literature or using expert opinion).
- The distinction between the procedure and encounter durations should be explainable.

The procedure description *Laparoscopic Removal of Gall Bladder* satisfies all of the above criteria with $n = 413$. The encounter duration (median duration of 17 hours) represents a stay in the hospital while the procedure duration (median 139 minutes) most likely resembles the surgery duration [68]. Using the summary statistics of this procedure (shown in Table 5.1), we are able to estimate the parameters of various distributions in Section 5.1.4.

5.1.3 Fitting Initial Distribution to Synthea Duration Data

To generate a set of initial states that is common across all experiments (see Section 5.2.1), we require the assumption of a single distribution for the surgery durations. Therefore, we use statistical analysis to arrive at the best fit distribution given the emergency laparoscopic gall bladder removal surgery duration data. Our best fit analysis was implemented using the Python package fitter version 1.5.1 [69] with default settings and a timeout of 60 seconds. This package returns the

Distribution	RSS
uniform	1.31E-04
gennorm	1.31E-04
johnsonsb	1.50E-04
halfgennorm	1.80E-04
bradford	1.87E-04

Table 5.2: Python **fitter** results for best fit distribution.

best fit distribution from the available 80 SciPy distributions [69]. The best fit distribution is the Uniform distribution with a minimum of 45 and a maximum of 239 based on the residual sum of squares (RSS) results shown in Table 5.2 (note that the distribution parameters have been excluded from this table).

5.1.4 Choosing Parameters of Job Attributes

Given the unavailability of real data, and the limitations of the synthetic data used, this section outlines the parameters chosen for various job attributes. As demonstrated in Appendix A.2, any inputs to the DES framework can be easily adapted to reflect a given problem. Unless otherwise specified, all measures of time are given in minutes.

Job Arrivals

To model the dynamic arrival of jobs, a stationary Poisson process is used. We therefore assume that the number of arrivals follows a Poisson distribution and that the inter-arrival times follow an Exponential distribution with rate parameter λ. The arrival time of a particular job j is denoted as r_j.

For the purpose of this discussion, we assume deterministic processing times. Then, from a queueing theory perspective, the problem presented in this work is represented by an M/D/1 queue since jobs are arriving from a Poisson process, processing times are deterministic, and there is a single server. The definition of utilization (ρ), as it pertains to a single server queueing problem, represents the proportion of time the server is busy and can be computed as shown in Equation 9 where $\bar{x}$ represents the mean processing time [64]. When $\rho \geq 1$, the number of jobs in the queue tends toward infinity [64]. Therefore, to have a stable system (here we use the term "stable" in

the queueing theory sense) where the number of jobs in the queue does not tend toward infinity, a utilization of $\rho < 1$ is required. This is particularly important for the purpose of this thesis when considering the computational difficulty of solving the subproblems (see Section 4.4.4). As shown in Equation 10, the chosen rate parameter in our experiments, $\lambda = \frac{1}{180}$, ensures $\rho < 1$.

$$\rho = \lambda \bar{x} \tag{9}$$

$$\rho = \frac{1}{180}(140.27) = 0.78 < 1 \tag{10}$$

Job Priorities

We define three priority classes by setting $num\_priority\_levels = 3$. Jobs are considered to have a high, medium, or low priority, corresponding to $w_j = 3$, $w_j = 2$, and $w_j = 1$ respectively. Job priorities are drawn from a discrete Uniform distribution.

Job Due Dates

The due date of a job represents the time (in minutes) at which the job should be completed and is computed using a function which depends on both r_j and w_j as shown in Equation 11 where δ_{w_j} is specified for each possible value of w_j.

Given that the Synthea data does not contain information regarding the due dates of surgical procedures, our chosen values of δ_{w_j} are motivated by the meta-analysis finding that performing laparoscopic cholecystectomy within the 72 hour window is optimal [70]. We choose values for δ_{w_j} using the concept of early versus delayed surgery wait times [70, Table 1] but scale by a factor of $\frac{1}{8}$ to facilitate the need for rescheduling. Consequently, we assume $\delta_3 = 180$, $\delta_2 = 540$, and $\delta_1 = 900$, and compute the due dates according to Equation 12.

$$d_j(r_j, w_j) = r_j + \delta_{w_j} \tag{11}$$

$$d_j(r_j, w_j) = \begin{cases} r_j + 180, & \text{if } w_j = 3 \\ r_j + 540, & \text{if } w_j = 2 \\ r_j + 900, & \text{if } w_j = 1 \end{cases} \tag{12}$$

Estimated Job Processing Durations

This section outlines the parameters used for the estimated processing time distributions given in Section 3.2 (Uniform, Exponential, Left Truncated Normal, and Bimodal). In the book *Probability distributions: with truncated, log and bivariate extensions*, Thomopoulos discusses two popular methods in estimating distribution parameters from sample data: maximum-likelihood-estimate (MLE) which uses the sample data to mathematically estimate the parameters, and the method-of-moments (MoM) which estimates the parameters using algebraic manipulation and summary statistics [71]. For the chosen Synthea data, we estimate the distribution parameters using MoM with sample mean $\bar{x} = 140.27$ and sample standard deviation $s = 55.80$ (see Section A.3). The resulting parameters are shown in Table 5.3 and are entered into the DES framework as explained in Appendix A.2.

Distribution	Parameter	Value
Uniform	a	43.62
	b	236.92
Exponential	σ	71291.58E-07
Left Truncated Normal	γ	0.00
	μ	140.27
	σ^2	55.80
Lognormal	μ	4.87
	σ	0.15
Bimodal	ρ	0.60
	μ_1	100.45
	μ_2	200.00
	σ_1	18.28
	σ_2	36.57

Table 5.3: Parameters used for the estimated processing time distributions.

To validate the fitted distributions, we generate one million estimated processing times from each distribution and report the sample means and standard deviations, rounded to two decimal

distribution	mean	variance
true data	140.27	55.80
Uniform	140.24	55.78
Exponential	140.20	140.10
Left Truncated Normal	141.18	54.68
Lognormal	140.19	55.74
Bimodal	140.34	55.81

Table 5.4: Sample mean and variance of the chosen distributions for $\hat{p}_j$ (n = 1000000).

places, in Table 5.4 (see Figure A.1 for the probability density functions of the distributions). Table 5.4 shows that the Uniform distribution with the chosen parameters most closely resembles the true data while the Exponential distribution with the chosen parameters does not fit the data well due to the characteristic that the mean and standard deviation of the Exponential distribution are equal.

True Job Processing Durations

We explore two candidate distributions for the true processing time of a job (p_j) used in the stochastic experiment: the Normal distribution and the Uniform distribution. We assume the candidate distributions are centered at $\hat{p_j}$ and have a standard deviation equal to the standard deviation found using the Synthea data ($s = 55.80$ Table 5.1). As explained in Section 3.2, we further allow for the optional parameters of ϵ^- and ϵ^+ to control the level of uncertainty in the processing times by defining a lower bound $l_j = \hat{p}_j - \hat{p}_j\epsilon^-$ and an upper bound $u_j = \hat{p}_j + \hat{p}_j\epsilon^+$ for p_j. The distributions for p_j are then renormalized to fit within the provided lower and upper bounds. For the bounded distributions in experiment 2, we assume $\epsilon^- = 0.1$ and $\epsilon^+ = 0.2$. Using these parameters reduces the standard deviation of the candidate distributions and, since $\epsilon^- \neq \epsilon^+$, also has an effect on the mean.

Since the distribution of p_j depends on $\hat{p}_j$, Appendix A.3 demonstrates an example of determining the distribution parameters for p_j assuming that $\hat{p}_j = \bar{x} = 140.27$ (Table 5.1). Given that the parameters of the true processing time distributions are computed internally within the DES framework based on the value of $\hat{p}_j$, the only parameters required as input to the simulation framework are ϵ^-, ϵ^+, and s (see Appendix A.2).

To visualize the true processing time distributions, we generate 1 million values of p_j from each

Distribution	mean	variance
Normal	141.20	54.61
Uniform	140.28	55.80
Normal [.1, .2]	146.94	12.03
Uniform [.1, .2]	147.29	12.15

Table 5.5: Example sample mean and variance of the chosen distributions for p_j assuming $\hat{p}_j = 140.27$ (n = 1000000).

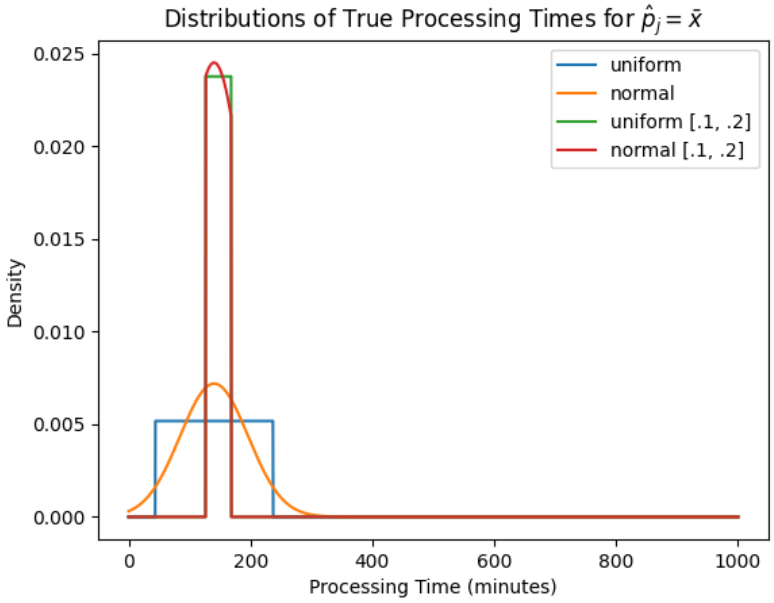


Figure 5.1: Example PDF of true processing time distributions for $\hat{p}_j = 140.27$.

distribution based on the assumption $\hat{p}_j = 140.27$ and report on the sample means and standard deviations, rounded to two decimal places, in Table 5.5. We define *Normal [.1, .2]* and *Uniform [.1, .2]* to be the bounded Normal and Uniform distributions with $\epsilon^- = 0.1$ and $\epsilon^+ = 0.2$. The probability density function (PDF) for the specific value of $\hat{p}_j = 140.27$, shown in Figure 5.1, demonstrates the reduction in variability when using the bounded distributions.

5.2 Simulation Setup

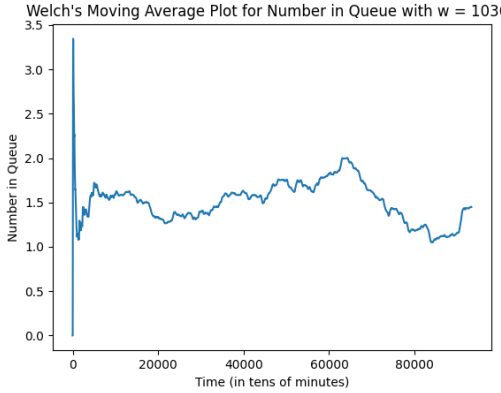
This section outlines the steps required to set up the DES framework including generating initial states, and determining the number of replications.

5.2.1 Initial States

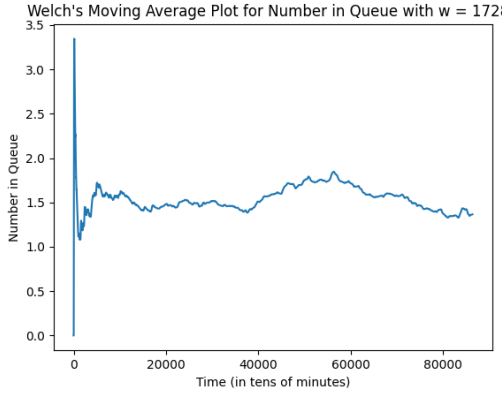
We define an initial state as the list of jobs waiting in the queue and all their attributes. Traditionally, an initial state should also include the job currently in progress and its remaining processing time (if any). However, since our problem setting is a single machine with no preemptions, any job which is currently in progress on the machine must complete without interruption. Therefore, we arrive at our definition by assuming the initial state starts at the time when the job currently on the machine finishes its processing, similar to the initial state simplification presented by Law's bank example [60, Sec: 9.4.3]. Given the simulation length, the effect of the initial state approximation should be minimal [60, Sec: 9.4.3].

To generate initial states, we employ a heuristic in which a *pseudo warm-up period* is determined [60]. Classically, the concept of warm-up period is used for statistical analysis of steady state parameters [60]; in this dissertation, we instead use this concept for generating initial states. We inspect the behaviour of the queue length over time to determine the pseudo warm-up period such that the initialization bias has been removed. To do so, we run the simulation with deterministic processing times drawn from a Uniform distribution with a minimum of 45 and a maximum of 239 according to the best fit distribution on the data using MLE (refer to Section 5.1.3). During simulation runs used for determining the pseudo warm-up period, we use the Earliest Due Date (EDD) first dynamic dispatching rule. The inputs required for determining the pseudo warm-up period are shown in Table A.2. To find a simulation run length sufficient to determine the pseudo warm-up period, we test four different choices of simulation length (ranging from 31 days to 24 months). Due to time constraints, we limit our analysis of determining the pseudo warm-up period to two replications.

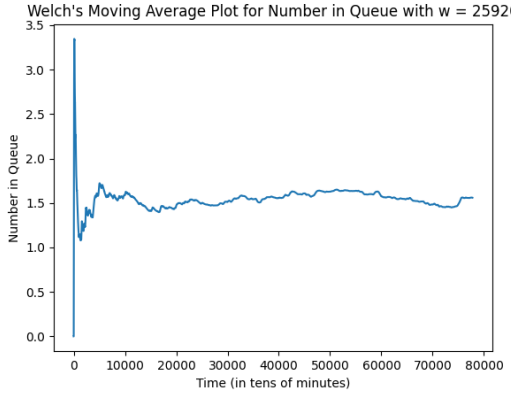
Plots of the queue length over time may be difficult to inspect due to the high-frequency oscillations [60] resulting from the possibly large number of state changes over the long simulation run length. We therefore perform smoothing using Welch's method (following the steps summarized in [60, 72]). Queue length observations are recorded whenever there is a change to the length of the queue, violating the assumption of equally spaced observations required for Welch's method [73]. Motivated by [73], we bin the data into equal time intervals of ten minutes after which we



(a) $w = \frac{m}{10}$



(b) $w = \frac{m}{6}$



(c) $w = \frac{m}{4}$

Figure 5.2: Welch moving average plots for the number of jobs in the queue using $n = 2$ and a simulation run length of 1036800 minutes ($m = 103680$) with various window sizes.

compute the time persistent average number in the queue within each bin. This new data satisfies the assumption of equally spaced observations and can be used for Welch's method. Additionally, Welch's method requires a number of replications and a window size parameter for the moving averages to be specified. The window size, denoted w, is a positive integer with $w \leq \frac{m}{4}$ where m is equal to the number of observations [60, 72]. We evaluate various window sizes with $w \in \{\frac{m}{4}, \frac{m}{6}, \frac{m}{10}\}$. Although we perform Welch's graphical procedure on all four of the simulation run lengths, we include Welch plots for a simulation run length of 24 months only, shown in Figure 5.2, since this is the simulation run length determined to be sufficiently long to infer the pseudo warm-up period.

As can be seen in Figure 5.2, increasing w results in a smoother plot. Note that the x-axis is in

tens of minutes as a result of binning the data. Based on the plots in Figure 5.2, we assume 100000 minutes (69.44 days) is sufficient for removing the initialization bias. To be conservative, we choose a pseudo warm-up period of 129600 minutes (90 days).

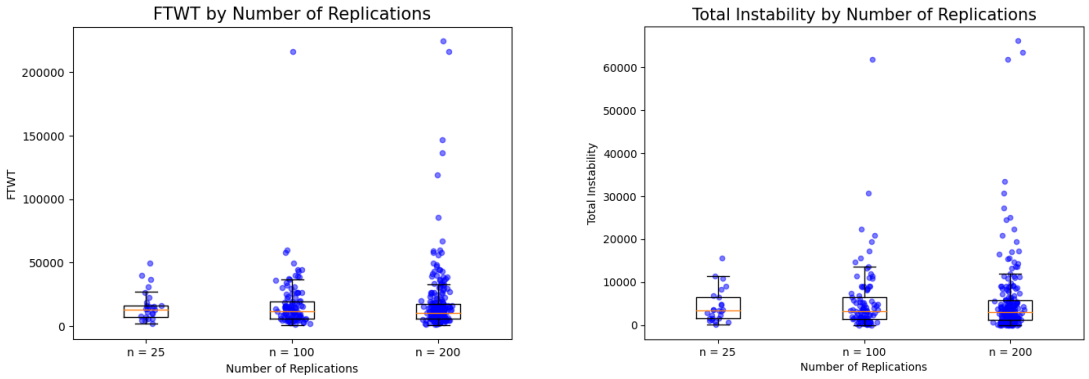
We run the simulation with all the same assumptions as those used to determine the pseudo warm-up period with the stopping criteria set as the end of the pseudo warm-up period (see Table A.3 for the DES framework inputs). The jobs remaining in the queue at the end of the pseudo warm-up period form the initial state. There is an initial state for every replication but the same initial states are used across various scenarios. Therefore, we repeat the above process for the required number of replications.

5.2.2 Number of Replications

Due to the large number of scenarios being considered in the experiments (see Sections 5.3.1, 5.4.1), it is not feasible to evaluate all scenarios when determining the number of replications. Therefore, we choose a single scenario with high variability, namely the deterministic case with processing times drawn from an Exponential distribution and the *eventJobCompletion* policy, and use it to determine the number of replications. We compute FTWT and total instability for this scenario using different number of replications. By inspecting the plots in Figure 5.3, we conclude that $n = 100$ is a sufficient number of replications while still ensuring computational feasibility for the purpose of this thesis.

5.3 Experiment 1: Deterministic Case

In this section, we investigate the effects of the assumption of a particular statistical distribution for the processing times and rescheduling policies on total instability and FTWT for the deterministic problem. The overarching goal is to be able to identify a specific policy, or a set of appropriate policies, which could be recommended depending on the processing time distributions.



(a) (b)

Figure 5.3: Boxplots of (a) FTWT and (b) total instability by number of replications for the scenario with deterministic processing times drawn from an Exponential distribution and the *eventJobCompletion* rescheduling policy.

5.3.1 Experiment Outline

Experiment 1 focuses on the deterministic case ($p_j = \hat{p}_j \; \forall j \in J$). We investigate the effects of the estimated processing time distribution on the instability and efficiency results under different rescheduling policies. In this experiment, we use the estimated processing time distribution to generate the processing times which are deterministic and known at the time of arrival.

The distributions of job arrivals and priorities are defined in Section 5.1.4 and the construction of due dates is defined via Equation 12. The initial states are generated according to Section 5.2.1 and we use 100 replications for each scenario as described in Section 5.2.2. The period of interest is $\tau = 10080$ minutes, corresponding to one week.

Independent Variables

The following distributions, with parameters summarized in Table 5.3, are explored in this experiment: Uniform, Exponential, Left Truncated Normal, Lognormal, and Bimodal.

Nine different rescheduling policies are evaluated in this experiment. For the periodic rescheduling policies, we relate Δ to the average inter-arrival time given by $\frac{1}{\lambda}$. The different rescheduling policies are given below:

- *periodic140*: periodic policy with $\Delta = 140$ ($\Delta < \frac{1}{\lambda}$)

- *periodic180*: periodic policy with $\Delta = 180$ ($\Delta = \frac{1}{\lambda}$)
- *periodic220*: periodic policy with $\Delta = 220$ ($\Delta > \frac{1}{\lambda}$)
- *eventArrival*: event job arrival policy
- *eventHighPriority*: event priority arrival policy with priority set $\{3\}$
- *eventJobCompletion*: event job completion policy
- *eventScheduleCompletion*: event schedule completion policy
- *hybrid220HighPriority*: combination of event driven rescheduling on high priority arrivals (event priority arrival policy with priority set $\{3\}$) and periodic rescheduling with $\Delta = 220$
- *hybrid220ScheduleCompletion*: combination of event driven on schedule completion and periodic rescheduling with $\Delta = 220$

Let $\mathbf{D}$ refer to the set of distributions and $\mathbf{R}$ refer to the set of rescheduling policies to be considered. Then this experiment consists of $|\mathbf{R}| \cdot |\mathbf{D}|$ scenarios (combinations of distributions and policies) to be run. Using the above sets of distributions and rescheduling policies, this experiment requires 45 scenarios.

Dependent Variables

The dependent variables are the simulation results for each of the different scenarios. In this thesis, we are primarily interested in FTWT and total instability.

5.3.2 Experiment Results

In this section, we explore the results of experiment 1, the deterministic case. We compare FTWT and total instability by scenario and analyze the effect of the average length of the rescheduling interval on the two performance metrics. Finally, we report on the percent of subproblems which reached the time limit resulting in suboptimal solutions. A discussion of the variability in the two performance metrics for scenario subsets can be found in Section A.5.1.

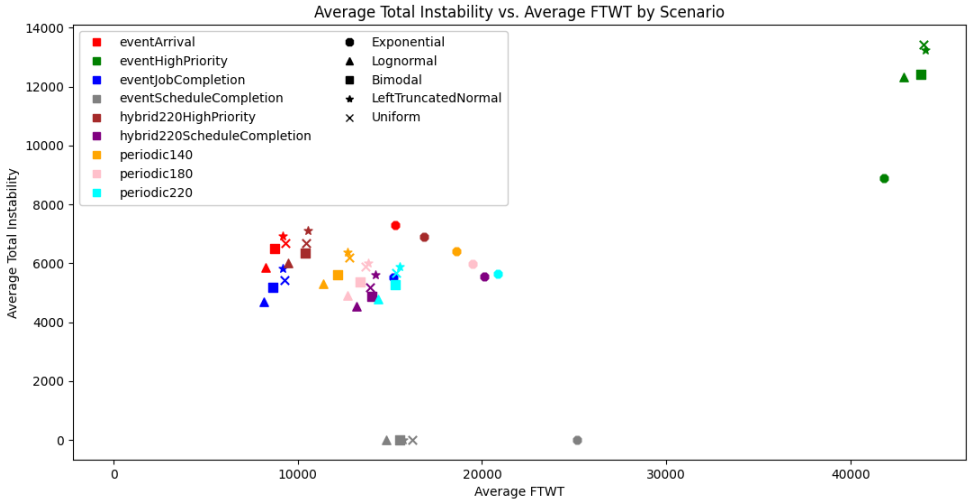


Figure 5.4: Average total instability (minutes) vs. average FTWT (weighted minutes) for the deterministic case by scenario, where a scenario is defined by a rescheduling policy (colours) and a processing time distribution assumption (shapes).

Comparing Total Instability and FTWT by Scenario

Total instability and FTWT are computed as described in Sections 3.3.3 and 3.3.4 respectively. The average of these metrics, taken over the 100 replications, are plotted in Figure 5.4 for each scenario, where scenarios are represented using shapes to refer to the processing time distribution assumptions and colours to refer to the rescheduling policies.

To analyze the effect of the processing time distributions and rescheduling policies on FTWT and total instability, we construct confidence intervals and perform hypothesis testing. Since we want to compare the various scenarios, our goal falls into the "All Pairwise Comparisons" category where we construct multiple confidence intervals (or perform the related hypothesis test) simultaneously [60]. In this case, it is important to consider the multiple-comparisons problem [60] where performing multiple t-tests (or constructing multiple confidence intervals) without accounting for the multiple comparisons (specifically the joint Type I error rate [74]) increases the probability of a false positive, or a Type I error [75]. One method to account for this issue relies on the Bonferroni inequality [60, Sec. 9.7] to establish an individual confidence level of $1 - \frac{\alpha}{c}$ such that the overall confidence level is at least $1 - \alpha$ where c is the number of intervals being constructed and α is the significance level. Specifically in the case of all pairwise comparisons, $c = \frac{k(k-1)}{2}$ [60, Sec. 10.3.2].

For experiment 1, we have a total of 45 scenarios being considered with two different performance metrics ($k = 90$). Then $c = \frac{90(90-1)}{2} = 4005$ and, assuming $\alpha = 0.05$, an individual confidence level of $1 - \frac{\alpha}{c} = 1 - \frac{0.05}{4005} = 0.99999$ would be required, resulting in exceedingly large confidence intervals. Adjusting the confidence levels based on the Bonferroni inequality can be conservative, causing an increase in false negatives, or Type II errors [75]. Law recommends using the Bonferroni adjustment when the number of confidence intervals being constructed is around ten or less [60, Sec. 9.7]. In cases where c is very large, the recommendation is to construct the regular confidence intervals, but to be cautious in reporting findings as there will likely be confidence intervals which do not contain their true means [60, Sec. 9.7]. Since the goal of this thesis is identifying general patterns and behaviours, we follow the approximate approach suggested by Law [60] by constructing regular 95% paired-t confidence intervals and performing the related paired t-tests for difference in means. However, there is a high likelihood that there will exist confidence intervals which do not contain their true means. If a decision maker wanted to further analyze a small subset of the scenarios or decide the best scenario among a small subset, then they should proceed with the Bonferroni adjustment.

Throughout our analysis, we assume both metrics are equally important. In practice, more importance may be placed on one of the performance metrics over the other. The units of measurement for total instability are in minutes. However, FTWT is a weighted metric and therefore is affected by both the weight and tardiness of a job.

We present the following results capturing the performance of the various rescheduling policies:

(1) As observed in Figure 5.4, the *eventHighPriority* policy performs poorly as it results in both a high average FTWT and a high average total instability for all distributions. Recall that *eventHighPriority* refers to the case where rescheduling occurs every time a high priority job (a job with priority $w_j = 3$) arrives.

FTWT: for each distribution, we observe a statistically significant difference ($p < 0.001$) between *eventHighPriority* and each of the remaining policies with respect to FTWT.

Total instability: the Lognormal, Bimodal, Left Truncated Normal, and Uniform distributions show statistically significant differences ($p < 0.001$) between the *eventHighPriority* policy

and every other policy with respect to total instability. The Exponential distribution shows statistically significant differences ($p < 0.05$) between the *eventHighPriority* policy and every other policy with respect to total instability.

For example, assuming the Exponential distribution, the *periodic220* policy outperforms *eventHighPriority* at the 95% confidence level, with 95% paired-t confidence intervals (*eventHighPriority* - *periodic220*) for FTWT and total instability being [16990.67, 24897.81] and [1733.45, 4745.28] respectively.

(2) As expected, *eventScheduleCompletion* has zero total instability for all distributions. Therefore, we can use this policy as a benchmark to determine if a different rescheduling policy is worth considering: a policy with a higher average FTWT than *eventScheduleCompletion* would not be advantageous. By inspecting Figure 5.4, we identify two policies with potentially higher average FTWT than *eventScheduleCompletion* across a majority of distributions: *periodic220* and *eventHighPriority*. The results of the paired t-tests and paired-t confidence intervals for the difference of means in FTWT of *eventScheduleCompletion* minus *eventHighPriority* and *periodic220* are shown in Table 5.6. We see that *eventHighPriority* performs worse in FTWT than *eventScheduleCompletion* across all distributions at the 95% confidence level and the difference between the two policies is statistically significant ($p < 0.001$). With the exception of the Exponential distribution, *periodic220* does not produce statistically significant differences in FTWT when compared to *eventScheduleCompletion*. Therefore, even if *periodic220* performs marginally better than *eventScheduleCompletion* for a given distribution, it is likely not worth considering due to the total instability cost incurred.

(3) From Figure 5.4, we observe that *eventArrival* outperforms *eventHighPriority* in both average FTWT and average total instability across all distributions. The results of the paired t-test and confidence interval (*eventArrival* minus *eventHighPriority*) for each distribution are shown in Table 5.7. Given that *eventArrival* reschedules on every new arrival while *eventHighPriority* reschedules on high priority job arrivals, *eventHighPriority* performs a subset of the rescheduling actions that *eventArrival* performs. Since *eventArrival* takes advantage of information regarding the uncertainty (dynamic arrivals) more frequently by rescheduling on every

Scenario			
Rescheduling Policy	Distribution	CI	p-value
eventHighPriority	Exponential	[12030.46, 21216.25]	1.54E-10***
	Lognormal	[23295.00, 32958.62]	6.30E-20***
	Bimodal	[23605.29, 32990.78]	8.18E-21***
	Left Truncated Normal	[23625.13, 33005.38]	7.64E-21***
	Uniform	[22873.92, 32624.26]	2.24E-19***
periodic220	Exponential	[-6976.01, -1665.76]	1.77E-3**
	Lognormal	[-3925.20, 704.77]	0.73
	Bimodal	[-2340.44, 1863.62]	0.82
	Left Truncated Normal	[-2195.64, 1738.33]	0.82
	Uniform	[-3286.99, 1559.92]	0.48

Table 5.6: Deterministic experiment 95% paired-t confidence intervals and p-value (* : $p < 0.05$, ** : $p < 0.01$, *** : $p < 0.001$) for paired two-tailed t-test for FTWT of *eventScheduleCompletion* minus *eventHighPriority* and *periodic220*.

new arrival, we expect it to perform better than *eventHighPriority* in terms of average FTWT. However, Table 5.7 demonstrates that *eventArrival* also outperforms the *eventHighPriority* in average total instability for each distribution.

FTWT: the results show a statistically significant difference between the FTWT of the two rescheduling policies ($p < 0.001$) for each distribution, while the confidence interval shows that *eventArrival* outperforms *eventHighPriority* at the 95% confidence level.

Total instability: the 95% confidence intervals for total instability also demonstrate that *eventArrival* results in a lower total instability than *eventHighPriority*. The t-test shows a statistically significant difference between the total instability results of the two rescheduling policies with $p < 0.05$ for the Exponential distribution and $p < 0.001$ for the remaining distributions.

Question 1: Does the assumption of a specific statistical distribution for processing times have an effect on FTWT and total instability?

(1) As seen in Figure 5.4, the Exponential distribution differs from the remaining distributions with a higher average FTWT and slightly higher average total instability across nearly all rescheduling policies (Table 5.8). This could be a result of the higher variability in the Exponential distribution (see Table 5.4).

	FTWT		Total Instability	
Distribution	CI	p-value	CI	p-value
Exponential	[-30418.62, -22620.83]	4.98E-24***	[-3060.80, -106.63]	0.04*
Lognormal	[-39158.41, -30160.71]	1.19E-27***	[-8413.83, -4530.91]	2.24E-09***
Bimodal	[-39395.68, -30803.12]	1.86E-29***	[-7620.14, -4215.68]	5.92E-10***
Left Truncated Normal	[-39261.65, -30481.75]	1.38E-28***	[-8397.33, -4228.70]	3.56E-08***
Uniform	[-39179.68, -30130.17]	1.78E-27***	[-8601.37, -4886.24]	1.39E-10***

Table 5.7: Deterministic experiment 95% paired-t confidence intervals and p-values (* : $p < 0.05$, ** : $p < 0.01$, *** : $p < 0.001$) for paired t-test of *eventArrival* minus *eventHighPriority* for each distribution and metric.

FTWT: we observe statistically significant differences between the Exponential distribution and each of the remaining distributions for all rescheduling policies $(p < 0.001)$ except *eventHighPriority*.

Total instability: we find that there are no statistically significant differences between the Exponential distribution and the remaining distributions except for the scenarios including the *eventHighPriority* rescheduling policy $(p < 0.001)$.

The Exponential distribution is commonly used to represent processing times in stochastic scheduling problems [52], in particular due to the tractable solutions available [62, Table F.1]. In our literature review, we observe the use of the exponential distribution for surgery durations [33, 51, 76], emergency department service times [77], and job processing times in other non-healthcare settings [78, 79, 80, 81, 82]. The frequent appearance of exponentially distributed processing times in literature highlights the importance of our result as incorrectly assuming an Exponential distribution can lead to erroneous conclusions (ones that are statistically significantly different from what they should be if the best fit distribution were different).

(2) As shown in Figure 5.4, the Uniform and Left Truncated distributions have similar results across all rescheduling policies. This observation is formalized in Table 5.9 which shows, for each rescheduling policy, there are no statistically significant differences between the two distributions with respect to FTWT or total instability. We remark that these two distributions have similar variances (Table 5.4).

Rescheduling policy	Lognormal		Bimodal		Left Truncated Normal		Uniform	
	FTWT	TI	FTWT	TI	FTWT	TI	FTWT	TI
eventArrival	1.28E-07***	0.04*	8.17E-07***	0.23	6.49E-07***	0.67	3.15E-06***	0.45
eventHighPriority	0.48	2.60E-05***	0.23	1.62E-05***	0.19	4.43E-06***	0.18	5.90E-06***
eventJobCompletion	1.25E-07***	0.19	9.05E-07***	0.56	1.03E-06***	0.70	3.74E-06***	0.93
eventScheduleCompletion	3.33E-06***	–	1.24E-05***	–	8.21E-05***	–	2.18E-04***	–
hybrid220HighPriority	1.38E-07***	0.18	4.03E-06***	0.39	1.62E-06***	0.82	3.24E-06***	0.79
hybrid220ScheduleCompletion	1.08E-06***	0.13	1.67E-05***	0.30	7.61E-06***	0.93	5.05E-06***	0.62
periodic140	1.41E-07***	0.12	2.09E-06***	0.23	3.50E-06***	0.97	1.96E-05***	0.80
periodic180	1.48E-06***	0.13	1.18E-05***	0.35	2.25E-05***	0.96	1.63E-05***	0.92
periodic220	4.59E-06***	0.19	7.37E-05***	0.55	8.46E-05***	0.75	8.57E-05***	0.95

Table 5.8: Deterministic experiment p-values (* : $p < 0.05$, ** : $p < 0.01$, *** : $p < 0.001$) for difference in means between the Exponential distribution and the remaining distributions (shown in columns) for each rescheduling policy and both FTWT and total instability (TI). We use '–' to denote the scenario where we are unable to compute a p-value due to the zero variance in total instability when the *eventJobCompletion* policy is used.

	p-value	
Rescheduling Policy	FTWT	Total Instability
eventArrival	0.88	0.75
eventHighPriority	0.93	0.79
eventJobCompletion	0.93	0.60
eventScheduleCompletion	0.63	–
hybrid220HighPriority	0.90	0.61
hybrid220ScheduleCompletion	0.74	0.56
periodic140	0.90	0.82
periodic180	0.85	0.87
periodic220	0.86	0.78

Table 5.9: Deterministic experiment p-values (* : $p < 0.05$, ** : $p < 0.01$, *** : $p < 0.001$) for the paired t-test for difference in means between the Uniform and Left Truncated distribution results across all rescheduling policies. We use '–' to denote the scenario where we are unable to compute a p-value due to the zero variance in total instability when the *eventJobCompletion* policy is used.

(3) In most rescheduling policies, with the exception of *eventHighPriority* and *eventScheduleCompletion*, we observe a consistent pattern with respect to the distributions. This can be seen in Figure 5.4 by examining each colour (rescheduling policy) separately and identifying the relationship of all the shapes (distributions) to each other. This pattern suggests that the assumptions regarding the underlying statistical distributions have an effect on the results. Given the large number of scenarios, we focus on the *periodic180* policy to conduct statistical analysis of this observation. Table 5.10 displays the p-values between every possible pair of distributions when the *periodic180* policy is used.

Excluding comparisons involving the Exponential distribution, the only statistically significant result is between the Lognormal and Uniform distributions with respect to total instability ($p < 0.05$) when the *periodic180* policy is used (Table 5.10). Furthermore, the Lognormal distribution would underestimate the true total instability at the 95% confidence level (confidence interval [-1899.38, -45.51] for the difference in means using the Lognormal minus the Uniform). Recall that the best fit distribution found using MLE on the data was the Uniform distribution. Possible reasons for incorrectly assuming the Lognormal distribution include unavailability of data, basing the decision on commonly used distributions in literature, or as a result of the underlying distribution changing over time. The underestimation of the total instability resulting from assuming the Lognormal distribution rather than the Uniform

distribution is likely undesirable and could have negative consequences if a downstream or upstream decision was made based on the simulation results. This proves that there exists a situation where the shape of the distribution has statistically significant effects.

Based on the above results, we conclude that the level of variability in the processing time distribution has a larger effect on the FTWT results than the shape of the distribution. Future work should perform analysis on the remaining rescheduling policies similar to the analysis conducted for the *periodic180* policy in result (3) above.

Question 2: Can we determine a rescheduling policy which performs well under all distributions?

- Assuming the decision maker cares about each metric equally, Figure 5.4 suggests that *eventJobCompletion* is the best performing policy for all distributions evaluated. This observation is particularly interesting since *eventJobCompletion* is a fairly simple policy.

  Focusing on the Lognormal distribution, in Table 5.11 we show the p-values and confidence intervals for the difference in means between *eventJobCompletion* and every other policy for both performance metrics. A confidence interval where both bounds are negative would mean that *eventJobCompletion* outperforms the other policy at the 95% confidence level.

  FTWT: with the exception of *eventArrival*, there is a statistically significant difference between the FTWT of *eventJobCompletion* and of every other policy ($p < 0.001$) assuming a Lognormal distribution. Furthermore, *eventJobCompletion* outperforms every other policy with respect to FTWT at the 95% confidence level.

  Total instability: In terms of total instability, the only policy which performs better than *eventJobCompletion* at the 95% confidence level under the assumption of a Lognormal distribution is *eventScheduleCompletion* (which has zero total instability). With the exception of the *hybrid220ScheduleCompletion* and *periodic220* policies, the remainder of the policies show statistically significant differences in total instability compared with *eventJobCompletion* (see Table 5.11 for p-values).

	Lognormal		Bimodal		Left Truncated Normal		Uniform	
	FTWT	TI	FTWT	TI	FTWT	TI	FTWT	TI
Exponential	1.48E-06***	0.13	1.18E-05***	0.35	2.25E-05***	0.96	1.63E-05***	0.92
Lognormal			0.27	0.25	0.29	0.21	0.17	0.04*
Bimodal					0.69	0.45	0.69	0.32
Left Truncated Normal							0.85	0.87

Table 5.10: Deterministic experiment p-values (* : $p < 0.05$, ** : $p < 0.01$, *** : $p < 0.001$) for the paired t-test for difference in means for both FTWT and total instability (TI) between every possible pair of distributions assuming the *periodic180* policy is employed.

	FTWT		Total Instability	
Rescheduling Policy	p-Value	CI	p-Value	CI
eventArrival	0.08	[-185.91, 9.10]	6.42E-20***	[-1358.47, -960.04]
eventHighPriority	9.15E-28***	[-39234.14, -30261.79]	9.14E-12***	[-9581.28, -5681.97]
eventScheduleCompletion	1.94E-07***	[-8956.37, -4285.94)]	1.46E-11***	[3473.42, 5897.69]
hybrid220HighPriority	2.48E-12***	[-1607.90, -970.94]	8.34E-15***	[-1595.15, -1028.29]
hybrid220ScheduleCompletion	1.16E-33***	[-5549.51, -4472.37]	0.32	[-126.00, 386.01]
periodic140	1.31E-35***	[-3570.99, -2913.18]	1.35E-07***	[-838.91, -406.2]
periodic180	4.33E-38***	[-4954.51, -4099.43]	0.04*	[-457.25, -12.63]
periodic220	5.62E-40***	[-6765.28, -5653.21]	0.36	[-359.06, 130.46]

Table 5.11: Deterministic experiment 95% paired-t confidence intervals and p-values (* : $p < 0.05$, ** : $p < 0.01$, *** : $p < 0.001$) for paired t-test of *eventJobCompletion* minus the secondary rescheduling policy assuming the processing times follow a Lognormal distribution.

Question 3: Can we identify a rescheduling policy whose performance is least affected by the different processing time distributions?

Identifying a rescheduling policy which is least affected by the assumption of a specific statistical distribution could be useful for the case where the true underlying distribution is not known or the distribution evolves over time. Specifically discussing health care applications, Pinedo advises caution in using probability distributions fit to data as the distributions may change over time due to improvements in operating techniques or evolving trends in patient characteristics [27].

- In terms of average total instability, the policy which is least affected by the assumption of a specific statistical distribution is *eventScheduleCompletion* since it has zero average total instability across all distributions. However, it is difficult to visually identify a policy which is least affected by the assumption of a specific statistical distribution when the performance measures are considered equally important and further numerical analysis would be required.

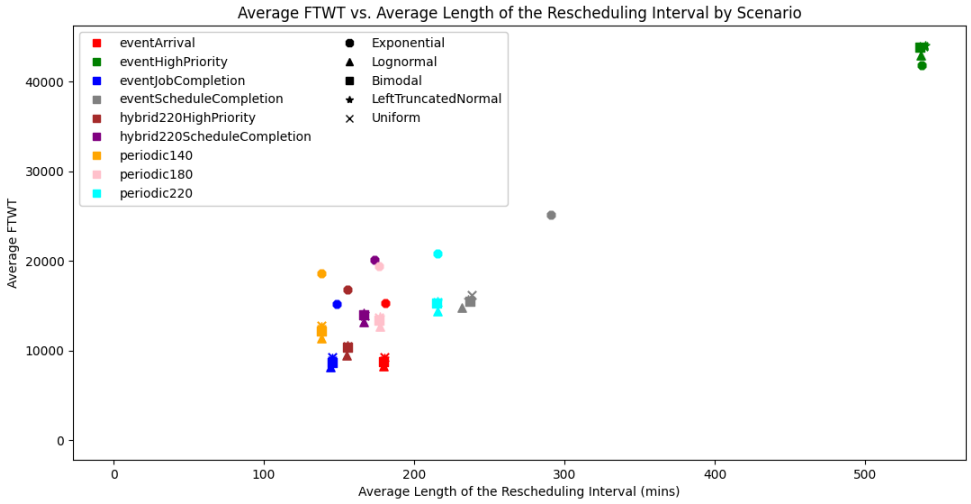
Effect of the Average Length of the Rescheduling Interval on Total Instability and FTWT

In this section, we wish to determine if there is a relationship between the average length of the rescheduling interval and either of the performance metrics. Intuitively, one may be tempted to analyze the number of rescheduling actions directly, however this does not take into account the difference in simulation run lengths resulting from the dynamically determined stopping criteria, T. Instead, we focus on the average length of the rescheduling interval computed as $\frac{\mathrm{T}}{NRA}$ where NRA is the number of rescheduling actions taken. Throughout this section, we also refer to the

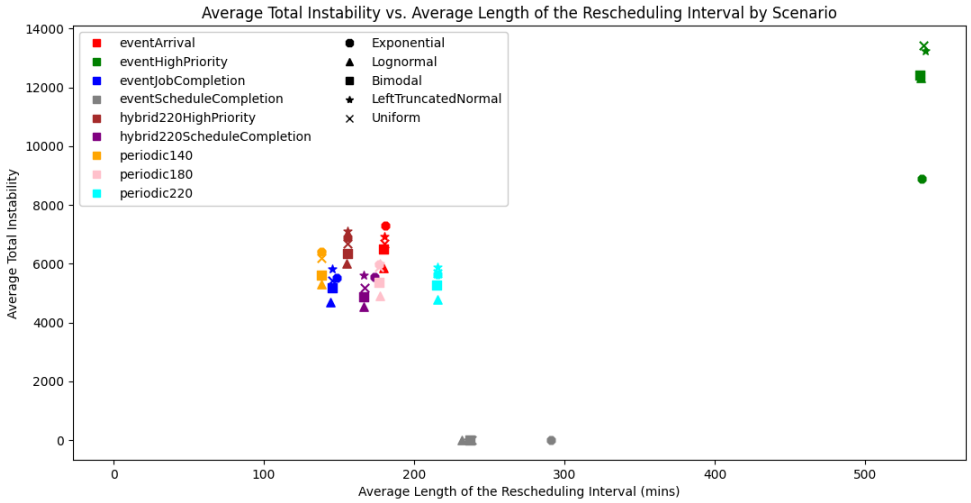
rescheduling frequency which is the inverse of the average length of the rescheduling interval [8, 14]. Figure 5.5 shows average FTWT and average total instability plotted against the average length of the rescheduling interval.

For the three periodic policies, Figure 5.5 shows that more frequent rescheduling leads to a lower average FTWT and a higher total instability. We do not observe any relationship for the remaining policies evaluated in Figure 5.5. One may expect that more frequent rescheduling leads to better efficiency, in our case lower FTWT. The *periodic140* policy has the most frequent rescheduling with an average length of the rescheduling interval of approximately 140 minutes as seen in Figure 5.5(a). We can compare this to the best performing policy, *eventJobCompletion*, whose average length of the rescheduling interval is slightly larger. Comparing these two policies, we observe that *periodic140* has a statistically significant ($p < 0.001$) difference in the FTWT results and the 95% paired-t confidence interval shows that *eventJobCompletion* outperforms *periodic140* across all distributions (Table 5.12). Since preemptions are not allowed, rescheduling at the completion time of each job takes into account the maximum information possible. We conclude that more frequent rescheduling does not necessarily result in lower FTWT and FTWT is affected by the quality of the information obtained at the time of rescheduling rather than the frequency of the rescheduling actions themselves.

As demonstrated in Figure 5.5(a), the FTWT of the *eventArrival* and *eventJobCompletion* policies are notably similar in the deterministic experiment. In a particular scenario where a job is completed and there are no pending jobs in the queue, *eventJobCompletion* behaves similar to *eventArrival* by rescheduling at the time of the next arrival. Under the assumption that preemptions are not allowed, these two policies both reschedule at times that allow them to make complete use of the realizations of uncertainty (job arrivals). Since the rescheduling method is independent of the policies, the MIP model may introduce slack into a schedule if none of the jobs have tardiness. Based on our observations, it appears that as we get closer to the point in the schedule where the slack has been introduced, a subsequent rescheduling algorithm may remove or reduce the slack. This is beneficial in the case where the job which immediately follows the scheduled slack is currently in progress when a new job arrives that will result in tardiness if not started immediately. If the slack had been reduced or removed prior to the arrival of the new job, the tardiness of the new job will



(a) Average FTWT vs. average length of the rescheduling interval by scenario.



(b) Average total instability vs. average length of the rescheduling interval by scenario.

Figure 5.5: Scatter plots of (a) average FTWT (weighted minutes) and (b) average total instability (minutes) vs. average length of the rescheduling interval for the deterministic experiment by scenario, where a scenario comprises a rescheduling policy (colours) and a processing time distribution assumption (shapes).

Distribution	CI	p-value
Exponential	[-3879.73, -2975.48]	3.61E-27***
Lognormal	[-3570.99, -2913.18]	1.31E-35***
Bimodal	[-3869.19, -3191.15]	1.67E-37***
Left Truncated Normal	[-3872.02, -3111.05]	3.35E-33***
Uniform	[-9432.28, -4495.13]	8.92E-33***

Table 5.12: Deterministic experiment 95% paired-t confidence intervals and p-values (* : $p < 0.05$, ** : $p < 0.01$, *** : $p < 0.001$) for paired t-test of *eventJobCompletion* minus *periodic140* for each distribution.

be less than if the slack was not reduced. In our specific problem setting, *eventArrival* reschedules less frequently than *eventJobCompletion* since the average inter-arrival time is 180 minutes and the average processing time of a job is 140.27 minutes. Since *eventArrival* reschedules less frequently, there is less opportunity to remove potential slack from a previous schedule. For a given simulation replication, it is possible that both *eventArrival* and *eventJobCompletion* result in the same FTWT, that *eventArrival* performs better with respect to FTWT, or that *eventJobCompletion* performs better with respect to FTWT. However, we observe that more often than not, *eventJobCompletion* outperforms *eventArrival*.

Percent of Suboptimal Solutions

All of the rescheduling policies evaluated in this thesis utilize the complete rescheduling method which requires solving multiple MIP models sequentially. The size of each MIP model depends on the number of jobs which have not yet started processing, and can vary substantially from policy to policy or subproblem to subproblem (see Section 4.4.4). Due to the computational challenges of solving large MIP models, we enforce a time limit of 300 seconds per subproblem for all policies considered. Table 5.13 summarizes these results by taking the average percent of rescheduling actions which exceeded the time limit aggregated by rescheduling policy.

As can be observed in Table 5.13, the average percent of rescheduling actions which exceed the time limit for the *eventHighPriority* policy is approximately ten times the value of any of the other policies. The *eventHighPriority* policy likely reaches the time limit more often because the average length of the rescheduling interval is much larger than any of the other policies (see Figure 5.5) thus allowing the queue of unscheduled jobs to grow between rescheduling actions. Since the number

Rescheduling Policy	Average % Exceeding Time Limit
eventArrival	7.71E-01
eventHighPriority	5.82
eventJobCompletion	6.80E-01
eventScheduleCompletion	1.57E-01
hybrid220HighPriority	5.75E-01
hybrid220ScheduleCompletion	4.00E-01
periodic140	5.65E-01
periodic180	5.41E-01
periodic220	5.58E-01

Table 5.13: The percent of rescheduling actions exceeding the time limit averaged across all distributions for each rescheduling policy for the deterministic experiment.

of decision variables in the MIP model is directly related to the number of jobs waiting to start processing (both scheduled and unscheduled), there are likely more sequencing decision variables in the MIP models for this policy than remaining policies.

5.4 Experiment 2: Stochastic Case

In this section, we investigate the effects of the processing time statistical distribution and rescheduling policies on total instability and FTWT for the case when the processing time realizations become known only once a job is completed. We call this experiment the "stochastic case" since, at the time when rescheduling decisions are made, the true processing times of uncompleted jobs are not known. The goals of this experiment are the same as experiment 1.

5.4.1 Experiment Outline

Experiment 2 models the stochastic case where an estimated processing time, $\hat{p}_j$, is provided, but the true processing time, p_j, does not become known until job j has completed processing. In this experiment, we investigate the effects of different true processing time distributions on the instability and efficiency results under various rescheduling policies. To reduce the dimensionality in this experiment, we assume a fixed distribution for $\hat{p}_j$ as the Uniform distribution over the interval [43.62, 236.92] from Section 5.1.4.

As in experiment 1, the distributions of job arrivals and priorities are defined in Section 5.1.4

and the construction of the due dates follows Equation 12. Initial states are generated according to the procedures outlined in Section 5.2.1 and, based on the analysis in Section 5.2.2, we use 100 replications for each scenario. We use the same period of interest as experiment 1 with $\tau = 10080$ minutes, corresponding to one week.

Independent Variables

The following true processing time distributions are used in this experiment (see Section 5.1.4):

- Deterministic ($p_j = \hat{p}_j \; \forall j \in J$)
- Uniform
- Normal
- Bounded Uniform: Uniform distribution with level of uncertainty parameters $\epsilon^- = 0.1$ and $\epsilon^+ = 0.2$
- Bounded Normal: Normal distribution with level of uncertainty parameters $\epsilon^- = 0.1$ and $\epsilon^+ = 0.2$

To further reduce the computational requirement of this experiment, we have chosen to evaluate a subset of the nine different rescheduling policies implemented in experiment 1. In particular, we choose the following rescheduling policies:

- *periodic180*: periodic policy with $\Delta = 180$
- *eventArrival*: event job arrival policy
- *eventHighPriority*: event priority arrival policy with priority set $\{3\}$
- *eventJobCompletion*: event job completion policy
- *eventScheduleCompletion*: event schedule completion policy
- *hybrid220HighPriority*: combination of event driven on high priority arrivals (event priority arrival policy with priority set $\{3\}$) and periodic rescheduling with $\Delta = 220$

Using the formula $|\mathbf{R}| \cdot |\mathbf{D}|$, we conclude that this experiment requires 30 scenarios.

Dependent Variables

The dependent variables are FTWT and total instability.

5.4.2 Experiment Results

In this section, we explore the results of experiment 2, the stochastic experiment. We compare FTWT and total instability by scenario, analyze the effect of the average length of the rescheduling interval, and report on the percent of subproblems which reached the time limit resulting in suboptimal solutions. Refer to Section A.5.2 for a discussion of the variability in the two performance metrics for scenario subsets.

Comparing Total Instability and FTWT by Scenario

Figure 5.6 plots the average of total instability and FTWT over 100 replications where shapes represent the processing time distribution assumptions and colours represent the rescheduling policies. Also following experiment 1, for every possible pair of scenarios we construct regular 95% paired-t confidence intervals and perform the related paired t-tests for difference in means. See Section 5.3.1 for a description of why it is not feasible to use the Bonferroni equation to adjust for the multiple comparisons for this experiment (with $c = \frac{60(60-1)}{2} = 1770$ and a confidence level of $1-\frac{\alpha}{c} = 1-\frac{0.05}{1770} = 0.99998$). We emphasize that all conclusions made in the analysis of experiment 2 assume that the decision maker cares about both performance metrics equally. Recall the units of measurement for total instability and FTWT are minutes and weighted minutes respectively.
We observe the following results:

(1) In Figure 5.6, we observe that, similarly to experiment 1, *eventHighPriority* (rescheduling every time a high priority job arrives) performs poorly with high average FTWT and total instability. For each distribution, there is a statistically significant difference between *eventHighPriority* and each of the remaining policies ($p < 0.001$) for both performance metrics. Moreover, for each distribution, the 95% paired-t confidence interval shows that the *eventHighPriority* policy performs worse than every other policy in both FTWT and total instability. For example, assuming a Normal distribution for the true processing times, the 95%

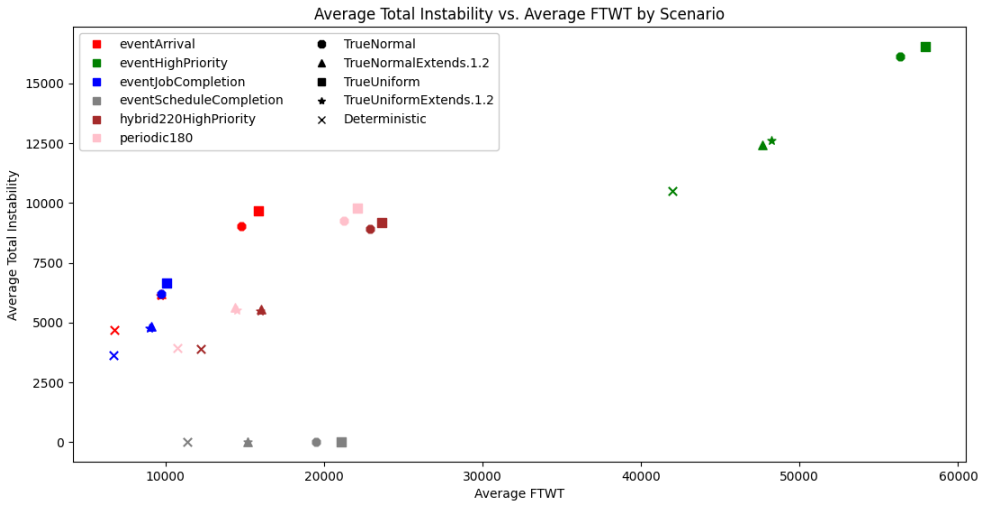


Figure 5.6: Average total instability (minutes) vs. average FTWT (weighted minutes) for the stochastic case by scenario, where a scenario comprises a rescheduling policy (colours) and a processing time distribution assumption (shapes).

paired-t confidence intervals constructed for *eventHighPriority* minus *eventJobCompletion* for FTWT and total instability are [39743.04, 53491.1] and [7669.17, 12162.36] respectively.

(2) As in experiment 1, *eventScheduleCompletion* results in zero total instability for each scenario. We again use this policy to determine if another policy is worth considering. Paired-t confidence intervals and paired t-tests for the difference of means in FTWT between *eventScheduleCompletion* and each of *eventHighPriority*, *periodic180*, and *hybrid220HighPriority* are shown in Table 5.14. We see that *eventHighPriority* performs worse in FTWT than *eventScheduleCompletion* at the 95% confidence level and the difference between the two policies is statistically significant ($p < 0.001$) across all distributions. The *hybrid220HighPriority* policy performs worse than *eventScheduleCompletion* with statistically significant differences in means ($p < 0.001$) for the Normal and Uniform distributions only. Furthermore, *periodic180* does not show statistically significant differences in FTWT compared to *eventScheduleCompletion* for any of the distributions. Even inn cases where a statistically significant difference is not found, the policy is likely not advantageous due to the increase in total instability compared to the zero instability of the *eventScheduleCompletion* policy.

Scenario			
Rescheduling Policy	Distribution	CI	p-value
eventHighPriority	Deterministic	[-36746.34, -24380.74]	3.66E-16***
	Normal	[-44238.22, -29417.81]	2.81E-16***
	Uniform	[-44211.06, -29506.54]	1.83E-16***
	Normal[0.1, 0.2]	[-39173.52, -25792.39]	8.79E-16***
	Uniform[0.1, 0.2]	[-39808.89, -26181.65]	9.95E-16***
hybrid220HighPriority	Deterministic	[-2065.73, 362.78]	0.17
	Normal	[-5374.45, -1393.78]	1.11E-03**
	Uniform	[-4237.65, -964.93]	2.24E-03**
	Normal[0.1, 0.2]	[-2409.63, 749.64]	0.30
	Uniform[0.1, 0.2]	[-2333.07, 742.04]	0.31
periodic180	Deterministic	[-560.85, 1771.5]	0.31
	Normal	[-3919.5, 399.03]	0.11
	Uniform	[-2731.97, 633.16]	0.22
	Normal[0.1, 0.2]	[-699.61, 2318.11]	0.29
	Uniform[0.1, 0.2]	[-693.39, 2125.43]	0.32

Table 5.14: Stochastic experiment 95% paired-t confidence intervals and p-values (* : $p < 0.05$, ** : $p < 0.01$, *** : $p < 0.001$) for paired two-tailed t-test for FTWT of *eventScheduleCompletion* minus *eventHighPriority*, *hybrid220HighPriority*, and *periodic180*.

Question 1: Does the assumption of a specific true processing time distribution have an effect on FTWT and total instability?

(1) Figure 5.6 suggests that the deterministic case results in the lowest average FTWT and average total instability for all rescheduling policies. For each rescheduling policy, we perform paired t-tests for difference in means between the deterministic case and each of the remaining distributions for both FTWT and total instability. Our results show that there is a statistically significant difference ($p < 0.001$) between the deterministic case and each of the other distributions for both metrics (with the exception of the total instability of *eventScheduleCompletion* which is zero for all distributions). We conclude that there is a statistically significant difference in FTWT and total instability between the deterministic and stochastic cases.

(2) Based on Figure 5.6, it appears that the two bounded distributions (Normal[0.1, 0.2] and Uniform[0.1, 0.2]) yield similar results across all rescheduling policies. In Table 5.15, we record the p-values for the paired t-test between the bounded Normal and the bounded Uniform distributions for both FTWT and total instability. The only statistically significant difference that

	p-value	
Rescheduling Policy	FTWT	Total Instability
eventArrival	0.54	0.27
eventHighPriority	0.02 *	0.12
eventJobCompletion	0.48	0.51
eventScheduleCompletion	0.92	–
hybrid220HighPriority	0.98	0.60
periodic180	0.46	0.09

Table 5.15: P-values (* : $p < 0.05$, ** : $p < 0.01$, *** : $p < 0.001$) for the paired t-test for difference in means between the bounded Normal and bounded Uniform distributions across all rescheduling policies for the stochastic experiment. We use '–' to denote the scenario where we are unable to compute a p-value due to the zero variance in total instability when the *eventJobCompletion* policy is used.

	p-value	
Rescheduling Policy	FTWT	Total Instability
eventArrival	0.30	0.28
eventHighPriority	0.13	0.40
eventJobCompletion	0.66	0.32
eventScheduleCompletion	0.16	–
hybrid220HighPriority	0.44	0.60
periodic180	0.44	0.36

Table 5.16: P-values (* : $p < 0.05$, ** : $p < 0.01$, *** : $p < 0.001$) for the paired t-test for difference in means between the Normal and Uniform distributions across all rescheduling policies for the stochastic experiment. We use '–' to denote the scenario where we are unable to compute a p-value due to the zero variance in total instability when the *eventJobCompletion* policy is used.

can be observed occurs for the FTWT of the *eventHighPriority* policy ($p < 0.05$).

We perform the same analysis between the Normal and Uniform distributions as shown in Table 5.16. Although Figure 5.6 suggests that the Normal distribution results in slightly worse outcomes than the Uniform distribution, the difference is not statistically significant, as shown in Table 5.16.

Taken together, the results of Tables 5.15 and 5.16 demonstrate that, across nearly all policies, there are no statistically significant differences in either performance metrics when distributions with similar means and variances are used for the true processing times.

(3) To simplify the analysis of performing a pairwise comparison of all distributions, we focus on the *periodic180* rescheduling policy as in experiment 1. Table 5.17 shows the p-values

resulting from the paired t-tests between every possible pair of distributions assuming the *periodic180* policy is used. As previously observed, the deterministic distribution has statistically significant differences from each of the remaining distributions in both FTWT and total instability ($p < 0.001$). The Normal and Uniform distributions differ statistically from the bounded Normal and bounded Uniform distributions in both FTWT and total instability ($p < 0.001$). However, there is no statistically significant difference between the Normal and Uniform distribution for either performance metric. Similarly, the two bounded distributions do not show statistically significant differences in either of the two metrics. We observe similar results for the remaining five policies. [2]

The previous three results suggest that the level of variability in the true processing time distributions, which is highest for the Uniform and Normal distributions and zero for the deterministic case (see Table 5.5 for an example), has a larger impact on the FTWT and total instability results than the shape of the distribution.

Question 2: Can we determine a rescheduling policy which performs well under all distributions?

- As shown in Figure 5.6, *eventJobCompletion* is the best performing policy for any distribution. This is a compelling observation as *eventJobCompletion* was also the best performing policy in experiment 1.

  Again, due to the large number of scenarios, we focus on a single distribution, specifically the Normal distribution, to perform statistical analysis. In Table 5.18, we present the confidence intervals and p-values for the difference in means between the *eventJobCompletion* policy and every other policy for both FTWT and total instability. The *eventJobCompletion* policy differs significantly in both FTWT and total instability from each of the remaining rescheduling policies ($p < 0.001$) when the Normal distribution is used for the true processing durations. Furthermore, the confidence intervals indicate that *eventJobCompletion* outperforms every

[2] The only exception being that *eventHighPriority* shows a statistically significant difference ($p < 0.05$) between the bounded normal and bounded uniform distributions. It is also worth noting that, since the *eventScheduleCompletion* policy results in zero total instability for all distributions, p-values for the total instability cannot be constructed.

	Normal		Uniform		Normal[0.1, 0.2]		Uniform[0.1, 0.2]	
	FTWT	TI	FTWT	TI	FTWT	TI	FTWT	TI
Deterministic	8.63E-12***	2.91E-08***	9.73E-12***	1.10E-08***	2.43E-16***	8.47E-12***	1.61E-17***	9.04E-11***
Normal			0.44	0.36	2.09E-08***	5.00E-06***	2.79E-08***	2.43E-06***
Uniform					6.01E-09***	5.00E-07***	9.89E-09***	3.11E-07***
Normal[0.1, 0.2]							0.46	0.09

Table 5.17: P-values (* : $p < 0.05$, ** : $p < 0.01$, *** : $p < 0.001$) for the paired t-test for difference in means for both FTWT and total instability (TI) between every possible pair of distributions assuming *periodic180* for the stochastic experiment.

	FTWT		Total Instability	
Rescheduling Policy	p-Value	CI	p-Value	CI
eventArrival	4.05E-10***	[-6513.94, -3629.73]	1.34E-09***	[-3668.41, -1996.91]
eventHighPriority	6.02E-24***	[-53491.10, -39743.04]	6.93E-14***	[-12162.36, -7669.17]
eventScheduleCompletion	5.79E-11***	[-12418.94, -7159.17]	9.17E-14***	[4792.92, 7624.52]
hybrid220HighPriority	2.87E-22***	[-15239.1, -11107.24]	2.22E-07***	[-3652.2, -1739.91]
periodic180	1.36E-17***	[-13738.35, -9360.24]	4.08E-06***	[-4274.07, -1811.47]

Table 5.18: Stochastic experiment 95% paired-t confidence intervals and p-values (* : $p < 0.05$, ** : $p < 0.01$, *** : $p < 0.001$) for paired t-test of *eventJobCompletion* minus *eventArrival*, *eventHighPriority*, *eventScheduleCompletion*, *hybrid220HighPriority*, and *periodic180* assuming Normal distribution for the true processing times.

other policy in FTWT and total instability with the one exception being the total instability *eventScheduleCompletion*, which has zero total instability by definition.

In both experiment 1 and experiment 2, there is a statistically significant difference ($p < 0.001$) between the *eventArrival* and *eventJobCompletion* policies for each distribution with respect to total instability. As discussed in Section 5.3.2, there are no statistically significant differences between the two policies across almost all distributions with respect to FTWT (with the exception being the exponential distribution) in experiment 1. However, in experiment 2, there are statistically significant differences ($p < 0.001$) between the FTWT of the two policies for every distribution except the deterministic case. We conclude that adding uncertainty in the processing times caused these two policies to differ from one another with respect to FTWT. Intuitively, in experiment 1 where the only source of uncertainty is the dynamic arrival of jobs, both *eventArrival* and *eventJobCompletion* make maximum use of the uncertainty information. However, in experiment 2, uncertainty arises from both the stochastic processing times and dynamic arrival of jobs. In this case, *eventJobCompletion* reacts to realizations of the uncertainty in processing times more effectively than *eventArrival*, resulting in a better FTWT.

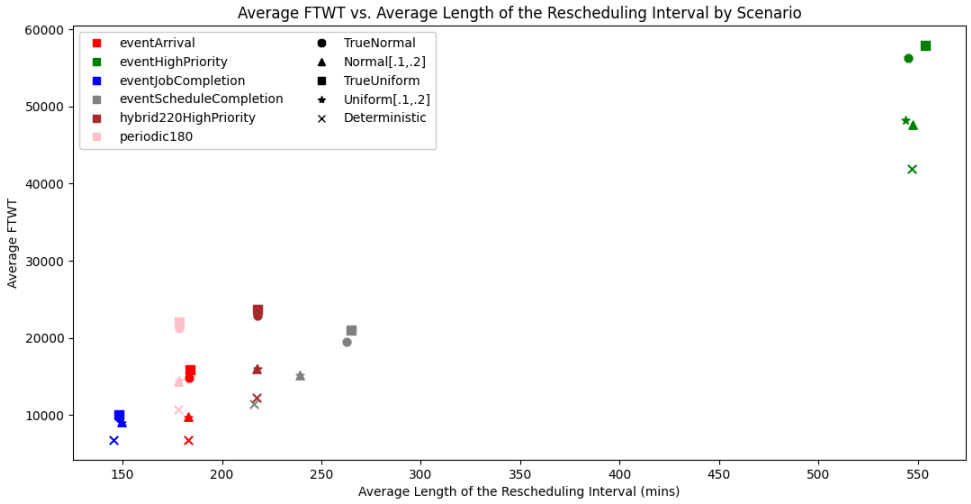
Question 3: Can we identify a rescheduling policy whose performance is least affected by the different processing time distributions?

- We identify *eventJobCompletion* to be the policy which is least affected by the assumption of a specific statistical distribution based on the proximity of the blue points in Figure 5.6. We find this to be an interesting conclusion given that *eventJobCompletion* is the policy which performs best in both the deterministic case and the various stochastic true processing time distributions.

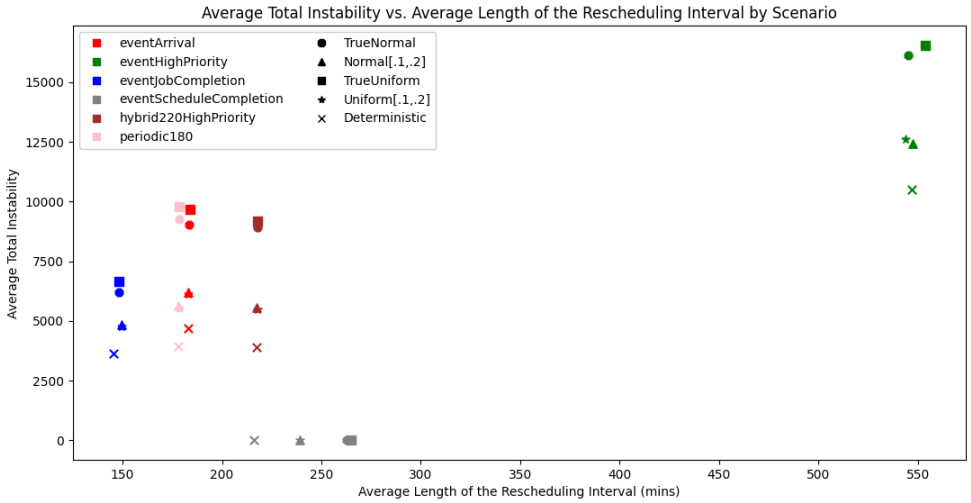
Effect of the Average Length of the Rescheduling Interval on Total Instability and FTWT

Here we determine if there is a relationship between the average length of the rescheduling interval and either FTWT or total instability. The importance of using the average length of the rescheduling interval as opposed to the average number of rescheduling actions is described in experiment 1. Figure 5.7 shows the average FTWT and average total instability plotted against the average length of the rescheduling interval.

Figure 5.7(a) indicates that there may be a slight positive relationship between the average length of the rescheduling interval and the average FTWT, although this does not hold in all cases. Again, intuitively, one may expect more frequent rescheduling to result in a lower average FTWT. However, as shown in Figure 5.7(a), *hybrid220HighPriority* reschedules more frequently than *eventScheduleCompletion* for each distribution except deterministic and *hybrid220HighPriority* results in a larger average FTWT for all distributions. The difference in means between *hybrid220HighPriority* and *eventScheduleCompletion* is statistically significant ($p < 0.05$) for the Normal and Uniform distributions only. This example shows that increasing the rescheduling frequency does not necessarily lead to a decrease in FTWT. In our experiments using *eventScheduleCompletion*, the average length of the queue (number of jobs which have not yet started processing) ranges from 1.28 in the deterministic case to 1.92 under the assumption of the Uniform distribution for the true processing times in the stochastic case. The relatively small average number of uncompleted jobs in any given schedule likely contributes to the positive performance of *eventScheduleCompletion* and we emphasize that different arrival and or average processing times could change the performance of this policy.



(a) Average FTWT vs. average length of the rescheduling interval by scenario.



(b) Average total instability vs. average length of the rescheduling interval by scenario.

Figure 5.7: Scatter plots of (a) average FTWT (weighted minutes) and (b) average total instability (minutes) vs. average length of the rescheduling interval for the stochastic experiment by scenario, where a scenario comprises a rescheduling policy (colours) and a processing time distribution assumption (shapes).

Rescheduling Policy	Average % Exceeding Time Limit
periodic180	0.90
hybrid220HighPriority	1.00
eventScheduleCompletion	0.07
eventJobCompletion	0.28
eventHighPriority	7.11
eventArrival	1.00

Table 5.19: The percent of rescheduling actions exceeding the time limit averaged across all distributions for each rescheduling policy for the stochastic experiment (including the deterministic case).

These findings coincide with the conclusion made in experiment 1 that FTWT is affected by the quality of the information obtained at time of rescheduling. Figure 5.7(b) does not demonstrate a relationship between the average length of the rescheduling interval and the average total instability.

Percent of Suboptimal Solutions

Due to the computational complexity of the MIP required to be solved during a rescheduling action (see Section 4.4.4), we enforce a time limit of 300 seconds for each MIP model. Table A.5 shows the number of rescheduling actions which reached the time limit and the average optimality gap for each scenario. In Table 5.19, we observe that the average percent of rescheduling actions which exceed the time limit for the *eventHighPriority* policy is substantially higher than the remaining policies. This is in accordance with the results of experiment 1 which also demonstrate that the *eventHighPriority* policy reaches the time limit more often. Nearly all rescheduling policies reach the MIP time limits less frequently in the deterministic case as compared to the stochastic cases, but do not necessarily have a smaller average optimality gap (see Table A.5).

5.5 Conclusion

We began this chapter with experimental preliminaries including synthetic data collection, fitting a distribution to the data, and estimating job attribute parameters. In order to conduct rigorous simulation experiments, we outlined the setup steps which include generating the initial states and determining the number of replications.

From the results of experiment 1, we observe that *eventHighPriority* performs poorly while *eventJobCompletion* performs the best across all distributions. The exponential distribution has the largest effect on the FTWT results, likely due to the high variability of this distribution.

In experiment 2, we again observe that *eventHighPriority* performs poorly while *eventJobCompletion* performs the best across all distributions. The variance of the distributions used to generate the stochastic processing times in this experiment appears to have a larger effect on the results than the shape of the distributions. In addition, in both experiments, we see that FTWT is affected by the quality of the information obtained at the time of rescheduling rather than the frequency of rescheduling actions.

The two main goals of this dissertation are to evaluate the effect that the rescheduling policies and assumption of a specific distribution for the processing times have on FTWT and total instability. With respect to the rescheduling policies, both the deterministic and the stochastic experiments show that the quality and completeness of information at the time of performing a rescheduling action, rather than the frequency of rescheduling, contributes to a lower average FTWT, or a better efficiency. Furthermore, both the deterministic and stochastic experiments show that instability is not directly correlated with the frequency of rescheduling actions. A decision maker should choose a rescheduling policy which makes use of the most complete information based on the sources of uncertainty present in their problem. Amongst the policies analyzed in this thesis, *eventJobCompletion* has the most complete information at the time of rescheduling in both the deterministic and stochastic experiments. With respect to the processing time statistical distribution assumptions, we note that the variance of the distributions analyzed has a greater effect on the results than the shape of the distribution in both the deterministic and stochastic cases.

Chapter 6

Conclusions and Future Work

This final chapter concludes the work presented in this thesis and provides suggestions for future work.

6.1 Conclusions

Operating rooms (ORs) are a costly component of a hospital and difficult to schedule due to the uncertainties involved. Motivated by OR scheduling, we solve a simplified OR scheduling problem posed as a single machine scheduling problem with dynamic arrivals and stochastic processing times. Although motivated by OR scheduling, we believe that the insights gained from this research can be applied to other service industry or job scheduling applications.

Taking a predictive-reactive approach, we employ rescheduling to react to realizations of uncertainty. In particular, we conduct two experiments: the deterministic experiment has uncertainty in the dynamic arrivals of jobs while the stochastic experiment contains uncertainty in both the dynamic arrivals of jobs and the processing times of jobs. The main goal of this thesis is to determine the effect of various rescheduling policies and processing time distributional assumptions on efficiency, specifically final total weighted tardiness (FTWT), and total instability. In particular, we ask the following three questions:

(1) Does the assumption of a specific statistical distribution for processing times have an effect on FTWT and total instability?

(2) Can we determine a rescheduling policy which performs well under all distributions?

(3) Can we identify a rescheduling policy whose performance is least affected by the different processing time distributions?

Furthermore, we determine the effect of the average length of the rescheduling interval on total instability and FTWT and, given the time limit imposed on the complete rescheduling mixed-integer programming (MIP) model, we analyze the percent of suboptimal solutions per scenario.

To answer the aforementioned questions, we developed a discrete-event simulation (DES) framework, varied the rescheduling policies and processing time distributional assumptions, and performed statistical analysis on the results. Our results indicate that, for both experiments, the variance of the processing time distribution has a greater impact than the shape of the distribution, and that the *eventJobCompletion* policy performs well across all distributions. For the stochastic experiment, we were able to identify *eventJobCompletion* as the policy whose performance is least affected by the different processing time distributions. In both experiments, the average length of the rescheduling interval does not appear to have an effect on total instability and FTWT. Finally, the *eventHighPriority* policy reached the computational time limit more often than the remaining policies in both experiments.

6.2 Future Work

To further advance the research presented in this thesis, we propose relaxing some of the assumptions made for the simplified OR scheduling problem presented in Section 3.1.2 and studied throughout this thesis. This will be of particular importance if this methodology is to be replicated and implemented for a specific hospital. For example, future work could consider incorporating the breakdowns of medical equipment, the availability of upstream or downstream resources, and the challenge of scheduling personnel resources from different departments. Furthermore, future work may focus on including sequence-dependent setup times in the model (e.g., cleaning and prepping the OR between related versus unrelated surgeries), and allowing for the cancellation or long-term postponement of surgeries (e.g., due to patient availability or worsening patient conditions).

One of the motivations for modelling the estimated and true processing times separately is the possibility to incorporate machine learning, where the estimated processing time would be the prediction and the true processing time reflects the error in the prediction. Future work can embed machine learning methods into the DES framework. Furthermore, future work may consider additional points in time at which more information regarding the processing time of the job becomes known, rather than the two points of time included in this thesis: the estimated processing time known at a job's arrival and the true processing time known at a job's completion. In the stochastic experiment, to reduce the number of scenarios, we fix a distribution for the estimated processing times while allowing the true processing time distributions to vary. Future work may vary both the estimated and true processing time distributions simultaneously in the stochastic experiment.

In addition, further work may perform a similar analysis under different arrival rates to determine if the results are consistent under different demands. As higher arrival rates result in a larger queue, this may require a heuristic rather than solving a MIP model at each subproblem. Future work should also explore and discuss the trade-off between the quality of the subproblem solutions and the computational costs of producing the schedules (see [13]).

In this thesis, we study instability without minimizing it directly. A potential area for future work includes considering instability when solving the sub problems, either directly in the objective function or through anticipatory scheduling [22]. Extending this research to different environments, such as parallel machine scheduling, would broaden the findings of this research.

Appendix A

A.1 Example of Computing Total Instability and Final Total Weighted Tardiness

This example demonstrates how total instability and final total weighted tardiness (FTWT) are computed for the single machine scheduling problem defined in Chapter 3. In this example, we consider $\tau = 3$ and choose to perform periodic rescheduling with $\Delta = 4$. At each rescheduling action, complete rescheduling is performed (using IBM's Decision Optimization CPLEX Modeling for Python, DOcplex version 2.23.222 [66]).

We consider three different priority classes with $w_j = 3$ being the highest priority class. Furthermore, due dates are assigned following equation 13. All of the data used in this example is found in Table A.1.

$$d_j(r_j, w_j) = \begin{cases} r_j + 2, & \text{if job } w_j = 3 \\ r_j + 5, & \text{if job } w_j = 2 \\ r_j + 9, & \text{if job } w_j = 1 \end{cases} \tag{13}$$

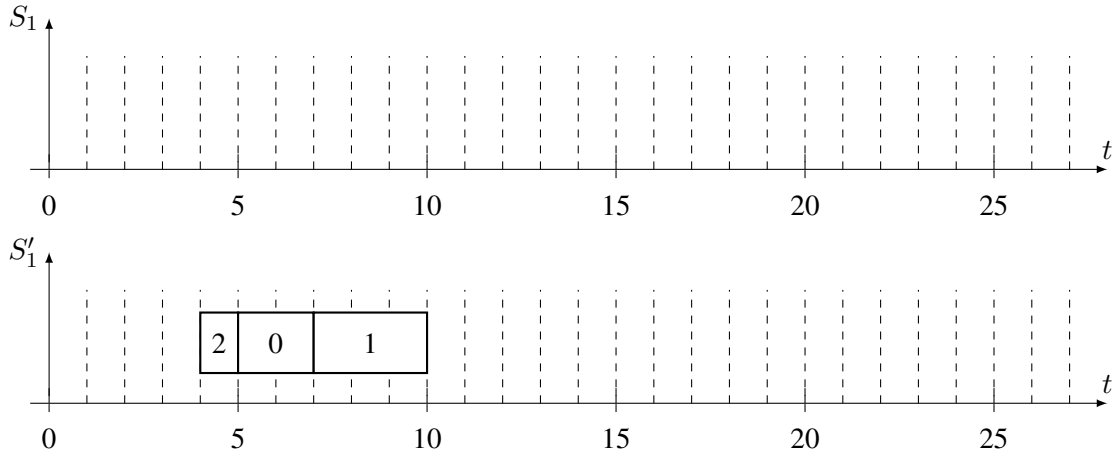
Example:

- $\mathbf{t = 0}$: Job 0 arrives.
- $\mathbf{t = 1}$: Job 1 arrives.
- $\mathbf{t = 2}$: Job 2 arrives.

job	w_j	r_j	d_j	$\hat{p}_j$	p_j
0	2	0	5	2	3
1	1	1	10	3	4
2	3	2	4	1	3
3	3	5	7	3	3
4	2	6	11	2	2
5	1	8	17	5	5
6	1	11	20	2	6
7	3	14	16	3	4
8	2	20	25	2	2

Table A.1: Job data for example of computing metrics.

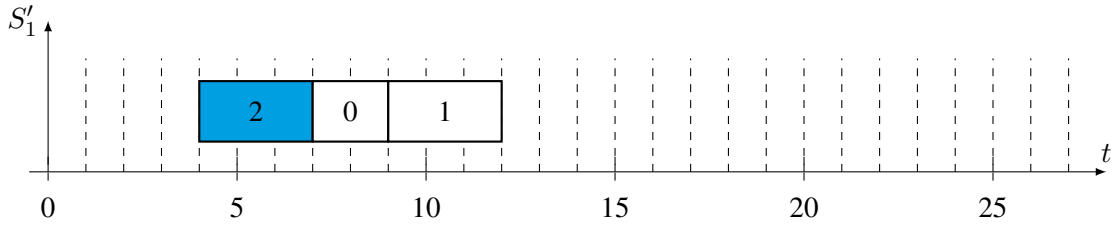
- $\mathbf{t = 4}$: Periodic rescheduling $(r = 1)$.



$\mathbf{J_1^*} = \{\}$

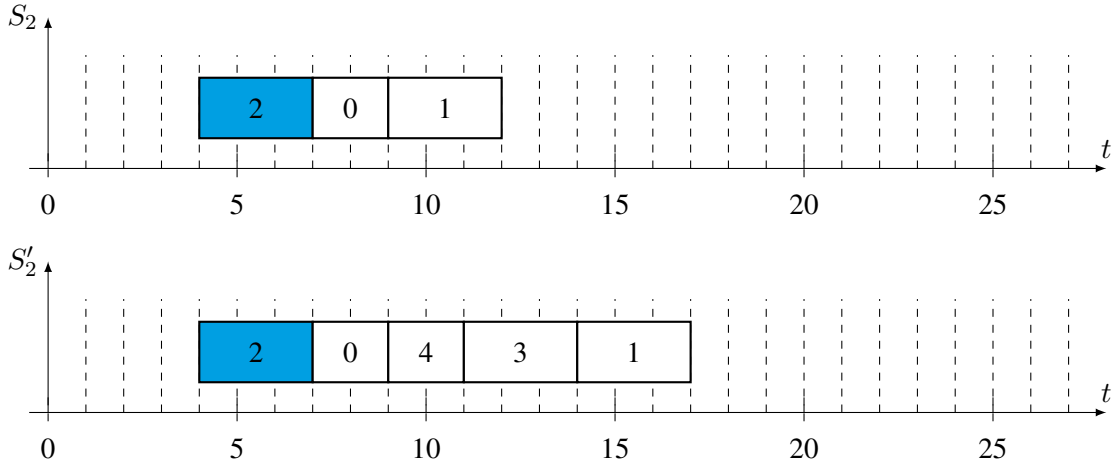
$I(S_1, S'_1) = \sum_{j \in \mathbf{J_1^*}} |C_j - C'_j| = 0$

- $\mathbf{t = 4}$: Job 2 starts processing.
- $\mathbf{t = 5}$: Job 3 arrives.
- $\mathbf{t = 6}$: Job 4 arrives.
- $\mathbf{t = 7}$: Job 2 finishes processing. Since $p_2 > \hat{p}_2$, the schedule is updated using the right shift heuristic for feasibility.



- $\mathbf{t = 7}$: Job 0 starts processing.

- $\mathbf{t = 8}$: Periodic rescheduling ($r = 2$).

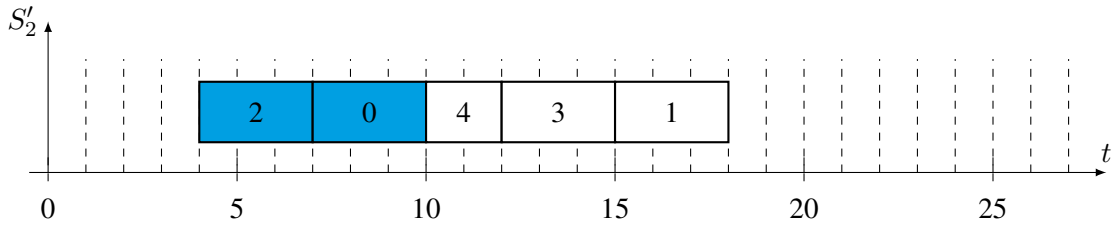


$\mathbf{J_2^*} = \{0, 1, 2\}$

$I(S_2, S'_2) = \sum_{j \in \mathbf{J_2^*}} |C_j - C'_j| = |C_0 - C'_0| + |C_1 - C'_1| + |C_2 - C'_2| = 0 + 0 + 5 = 5$

- $\mathbf{t = 9}$: Job 5 arrives.

- $\mathbf{t = 10}$: Job 0 finishes processing. Since $p_0 > \hat{p}_0$, the schedule is updated using the right shift heuristic for feasibility.

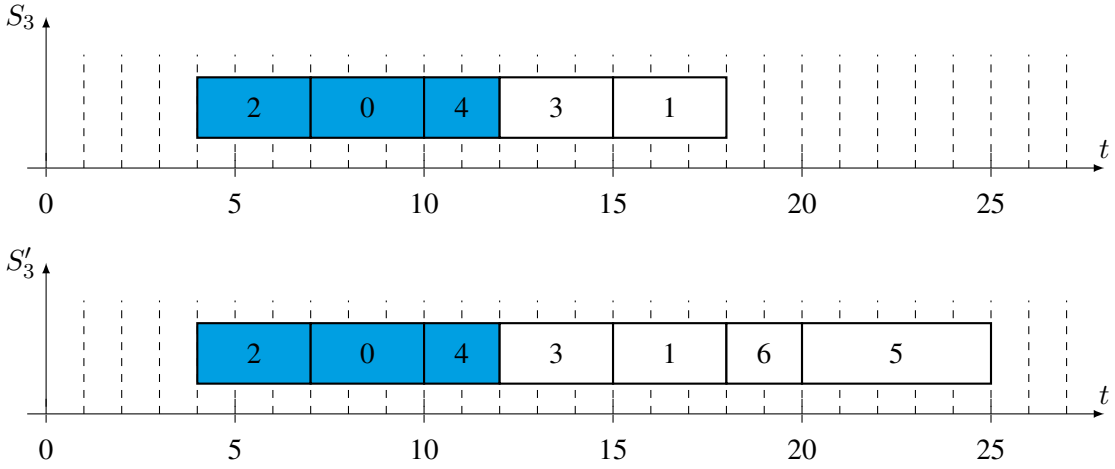


- $\mathbf{t = 10}$: Job 4 starts processing.

- $\mathbf{t = 11}$: Job 6 arrives.

- $\mathbf{t = 12}$: Job 4 finishes processing.

- $\mathbf{t = 12}$: Periodic rescheduling $(r = 3)$.



$\mathbf{J_3^*} = \{0, 1, 2\}$

$I(S_3, S'_3) = \sum_{j \in \mathbf{J_3^*}} |C_j - C'_j| = |C_0 - C'_0| + |C_1 - C'_1| + |C_2 - C'_2| = 0 + 0 + 0 = 0$

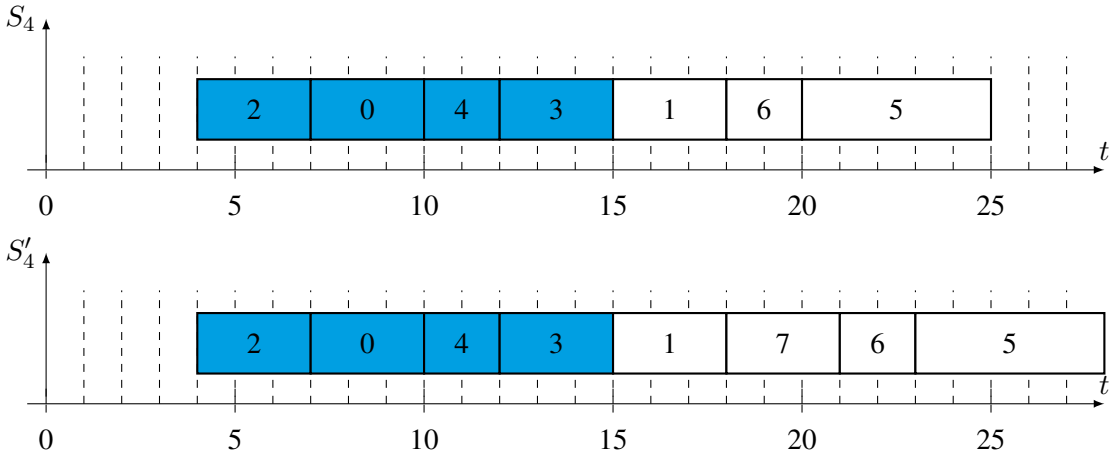
- $\mathbf{t = 12}$: Job 3 starts processing.

- $\mathbf{t = 14}$: Job 7 arrives.

- **t = 14**: Job 3 finishes processing.

- **t = 14**: Job 1 starts processing.

- **t = 16**: periodic rescheduling $(r = 4)$.



$\mathbf{J_4^*} = \{0, 1, 2\}$

$I(S_4, S'_4) = \sum_{j \in \mathbf{J_4^*}} |C_j - C'_j| = |C_0 - C'_0| + |C_1 - C'_1| + |C_2 - C'_2| = 0 + 0 + 0 = 0$

- $\mathbf{t = 18}$: Job 1 finishes processing. All jobs in $\mathbf{J}^*$ have now completed processing and the simulation terminates at $\mathrm{T} = 18$.

Total instability is computed as shown in Equation 14. Finally, the computation of FTWT is shown in Equation 15 where $\mathbf{J}^*_{\mathrm{T}} = \{0, 1, 2\}$.

$$\begin{aligned} I_{tot} = \sum_{r \in \mathbf{R}} I(S_r, S'_r) &= I(S_1, S'_1) + I(S_2, S'_2) + I(S_3, S'_3) + I(S_4, S'_4) \\ &= 0 + 0 + 5 + 0 = 5 \end{aligned} \tag{14}$$

$$\begin{aligned} FTWT = \sum_{j \in \mathbf{J}^*_{\mathrm{T}}} w_j T_j &= w_0 T_0 + w_1 T_1 + w_2 T_2 \\ &= 2(5) + 1(8) + 3(3) = 10 + 8 + 9 = 27 \end{aligned} \tag{15}$$

A.2 Inputs to the Simulation Framework

Below is a list describing the input parameters to the discrete-event simulation (DES) framework and their acceptable inputs. Unless otherwise specified, all time units are given in minutes.

- **num_priority_levels** $\in \mathbb{N}$: the number of priority levels.
- **machine_capacity** $\in \mathbb{N}$: the capacity of the machine (current code only supports $machine\_capacity = 1$).
- **stats_rounding** $\in \mathbb{W}$: the number of decimal places that the reported statistics should be rounded to.
- **replications** $\in \mathbb{N}$: the number of replications for the specific scenario.
- **store_all_events** $\in \{0, 1\}$: binary variable indicating whether or not the event log should be stored.
- **parent_folder_name**: text variable for the parent folder name.

- **folder_name**: text variable for the folder name (child of parent folder given in parent_folder_name).
- **version_num** $\in \mathbb{R}^+$: represents the version number of the scenario.
- **overwrite_files** $\in \{0, 1\}$: binary variable to indicate whether files can be overwritten. If files cannot be overwritten and already exist, an error will occur and the user must change the folder names and or version numbers before proceeding.
- **arrival_distribution** $\in \{$exponential$\}$: name of the distribution that will be used for arrivals. Additional distributions can easily be added to the code in the future.
- **processing_distribution** $\in \{$lognormal, uniform, left_truncated_normal, exponential, bimodal$\}$: name of the distribution that will be used for processing times. For future work, additional distributions can easily be added to the code.
- **arrival_params**: comma separated list of the parameters that are required for the arrival distribution. All parameters are described in detail in Section 5.1.4.
    - if arrival_distribution == exponential: $[\frac{1}{\lambda}]$ where λ is the rate parameter.
- **processing_params**: comma separated list of the parameters that are required for the processing distribution. All parameters are described in detail in Sections 5.1.4 and A.3.1.
    - if processing_distribution == lognormal: $[\mu, \sigma]$.
    - if processing_distribution == uniform: $[a, b]$.
    - if processing_distribution == left_truncated_normal: $[\gamma, \mu, \sigma^2]$.
    - if processing_distribution == exponential: $[\sigma]$.
    - if processing_distribution == bimodal: $[\mu_1, \sigma_1, \mu_2, \sigma_2, \rho]$.
- **priority_distribution** $\in \{$discrete_uniform$\}$: name of the distribution to be used for the priorities. Additional distributions can easily be implemented if desired in future work.
- **priority_params**: comma separated list of the parameters that are required for the priority distribution.

- if priority_distribution == discrete_uniform: $[1, num\_priority\_levels]$.

- **due_dates**: Comma separated list of the parameters required for construction of due dates: $[\delta_1, \delta_2, \delta_3]$ (see Equation 11).

- **rescheduling_type** $\in$ {periodic, event, hybrid}: indicates the type of rescheduling desired.

- **rescheduling_period** $\in \mathbb{N}$: the rescheduling period, Δ, must be non-empty if rescheduling_type $\in$ {periodic, hybrid}.

- **rescheduling_event_list**: the rescheduling event list takes a comma separated list (if rescheduling_type $\in$ {event, hybrid}, the list length must be greater than or equal to one) where possible list entries are:

  - arrival: reschedule when a new job arrives.
  - completion: reschedule when a job finishes processing.
  - departure: reschedule when a job departs.
  - started_processing: reschedule when a job starts processing.
  - priority_arrival: reschedule when a job of a certain priority arrives.
  - num_arrivals: reschedule when a certain number of jobs has arrived (i.e. every x arrivals).
  - num_unscheduled: reschedule when the number of arrived, unscheduled jobs reaches a certain number.
  - schedule_completion: reschedule when the last job of the current schedule has been completed.

- **priority_arrival_list** with elements $\in \{1, ..., num\_priority\_arrivals\}$: comma separated list of the priorities whose arrivals should indicate rescheduling. This field must be non-empty when priority_arrival $\in$ rescheduling_event_list.

- **num_arrivals_parameter** $\in \mathbb{N}$: the value, n, for which num_arrivals rescheduling policy operates on. This field must be non-empty if num_arrivals $\in$ rescheduling_event_list.

- **num_unscheduled_parameter** $\in \mathbb{N}$: the value, l, for which num_unscheduled policy operates on. This field must be non-emtpy if num_unscheduled $\in$ rescheduling_event_list.
- **mdl_time_limit** $\in \mathbb{N}$: the time limit, in seconds, for the optimization model. If left empty, no time limit on the complete optimization will be enforced.
- **stochastic** $\in \{0, 1\}$: a binary variable indicating whether the simulation contains processing times that are stochastic (1) or deterministic (0).
- **extend_negative** $\in [0, 1]$: this variable, denoted ϵ^-, is used to determine the lower bound for true processing time distribution ($\hat{p}_j - \hat{p}_j\epsilon^-$).
- **extend_positive** $\in [0, 1]$: this variable, denoted ϵ^+, is used to determine the upper bound for the true processing time distribution ($\hat{p}_j + \hat{p}_j\epsilon^+$).
- **true_distribution** $\in \{uniform, normal, deterministic\}$: name of the distribution that will be used for the true processing times (deterministic implies $\hat{p_j} = p_j$).
- **true_params**: comma separated list of the parameters that are required for the true_distribution. All parameters are described in detail in Section A.3.2.
    - if true_distribution == uniform: $[\sigma]$
    - if true_distribution == normal: $[\sigma]$
    - if true_distribution == deterministic: []
- **t1** $\in \mathbb{W}$: the starting time of the simulation (normally set to 0).
- **t2** $\in \mathbb{W}$: equivalent to τ, the stopping time of the period of interest where $t2 > t1$.
- **num_jobs_random_number_generation** $\in \mathbb{W}$: determines the number of entries for a job's attribute that should be generated per replication if a common random number file needs to be created. Due to the small storage requirements, it is best to overestimate this parameter. Note that if this parameter is not large enough, the simulation will run out of jobs and the user will encounter an error.

- **scheduling_strategy** $\in \{\text{complete}\}$: indicates the method which should be implemented for solving the subproblems. Currently only complete rescheduling is implemented for the purpose of this thesis. However, this parameter could be adapted in future work to instead use a heuristic or solve the problem suboptimally for computational efficiency.

- **generate_pseudo_warm_up** $\in \{0, 1\}$: binary variable to indicate whether the goal of the scenario is to determine the pseudo warm-up period (1) or not (0).

- **pseudo_warm_up_sim_length** $\in \mathbb{N}$: the simulation run length required to determine the pseudo warm-up period. This value should be much larger than the expected pseudo warm-up period. If generate_pseudo_warm_up == 1, then this field must be non-empty.

- **generate_initial_states** $\in \{0, 1\}$: binary variable to indicate whether the goal of the scenario is to generate the initial states (1) or not (0).

- **pseudo_warm_up_period** $\in \mathbb{N}$: the previously determined pseudo_warm_up period which acts as the stopping criteria when generating the initial states. If generate_initial_states == 1, then this field must be non-empty.

- **number_initial_states** $\in \mathbb{N}$: the number of initial states that should be generated. This field must be non-empty if generate_initial_states == 1.

A.3 Distribution Parameter Details

This chapter provides details regarding the parameter estimation of the various distributions for the estimated processing time durations and true processing time durations respectively. Parameter estimation is accomplished using the method-of-moments approach [71] with sample mean $\bar{x} = 140.27$ and sample standard deviation $s = 55.80$.

A.3.1 Estimated Processing Time Distributions

This section focuses on the parameter estimation of the estimated processing time distributions. The distributions used in this section are Uniform, Exponential, Left Truncated Normal, Lognormal,

and Bimodal. The probability density function (PDF) of the distributions using the parameters estimated in this section is shown in Figure A.1.

Uniform Distribution

For the continuous Uniform distribution, we aim to estimate parameters a and b using the equations $\mu = (b+a)/2$ and $\sigma = (b-a)/\sqrt{12}$. The parameter estimates, denoted as $\hat{a}$ and $\hat{b}$, are then computed as shown in Equation 16 and Equation 17 respectively [71].

$$\hat{a} = \bar{x} - \frac{s\sqrt{12}}{2} = 140.268715 - \frac{55.803025\sqrt{12}}{2} = 43.61504 \tag{16}$$

$$\hat{b} = \bar{x} + \frac{s\sqrt{12}}{2} = 140.268715 + \frac{55.803025\sqrt{12}}{2} = 236.922390 \tag{17}$$

With respect to the continuous Uniform distribution, we have the unique advantage of being able to compare the parameter estimates resulting from the MoM with the parameter estimates arising from the MLE approach taken in Section 5.1.3 ($\hat{a}_{MLE} = 45$ and $\hat{b}_{MLE} = 239$). However, to be consistent with all distributions that will be evaluated in the experiments, we will use the parameters estimated from the MoM.

Exponential Distribution

The exponential distribution assumes $\mu = \sigma$. Although this assumption does not fit our data well, we still choose to evaluate the exponential distribution due to its unique shape. The single parameter, θ, is estimated using the MoM where $\mu = \sigma = \frac{1}{\theta}$ [71] as shown in Equation 18.

$$\hat{\theta} = \frac{1}{\bar{x}} = \frac{1}{140.268715} = 0.007129 \tag{18}$$

Left Truncated Normal Distribution

The left truncated Uniform distribution was chosen where the truncation is required to ensure positive processing times. The truncation threshold, denoted as γ, is set to zero. Then the remaining parameters μ and σ^2 can be estimated according to Equations 19 and 20 [71].

$$\hat{\mu} = \bar{x} = 140.268715 \tag{19}$$

$$\hat{\sigma^2} = s^2 = 55.803025^2 = 3113.977599 \tag{20}$$

Lognormal Distribution

The lognormal distribution for a variable x is related to the normal distribution through the normally distributed variable y such that $y = ln(x)$ and $x = e^y$ [71]. Using the same notation as [71], we define the distributions of $x \sim LN(\mu_y, \sigma_y^2)$ and $y \sim N(\mu_y, \sigma_y^2)$. Since we are assuming our data comes from a lognormal distribution, we have $\mu_x = \bar{x} = 140.268715$ and $\sigma_x^2 = s^2 = 55.803025^2$. We must then use the MoM to provide parameter estimations for μ_y and σ_y using Equations 21 and 22 respectively [71].

$$\hat{\mu_y} = \ln\left[\frac{\bar{x}^2}{\sqrt{\bar{x}^2 + s^2}}\right] = \ln\left[\frac{140.268715^2}{\sqrt{140.268715^2 + 55.803025^2}}\right] = 4.870097 \tag{21}$$

$$\hat{\sigma_y^2} = \ln\left[1 + \frac{s^2}{\bar{x}^2}\right] = \ln\left[1 + \frac{55.803025^2}{140.268715^2}\right] = 0.146926 \tag{22}$$

Bimodal Distribution

We can define a Bimodal distribution as a mixture of two normal distributions $f(x) = \rho g_1(x) + (1 - \rho)g_2(x)$ where ρ is the mixing parameter, $g_1(x) \sim N(\mu_1, \sigma_1^2)$ and $g_2(x) \sim N(\mu_2, \sigma_2^2)$. Due to the number of parameters requiring estimation, we make the following additional assumptions:

- We assume a mixing parameter of $\rho = 0.6$.
- We assume the mean for the second component distribution is $\mu_2 = 200$.
- A relationship between the variances of the two component distributions is defined as $\sigma_2^2 = 4\sigma_1^2$.

By implementing these assumptions, we reduce the number of parameters that need to be estimated (μ_1, σ_1, σ_2), and add an additional equation to make this possible.

We define $Y \sim f(x)$, $X_1 \sim g_1(x)$, and $X_2 \sim g_2(x)$ and let D represent the random variable for the component distributions. We use the law of total expectation to determine the expected value of the distribution Y as shown in Equation 23 [83].

$$\begin{aligned} E[Y] &= E[E[Y|D]] = \sum_{d=1}^{2} E[Y|X_d]P(X_d) \\ &= \rho\mu_1 + (1-\rho)\mu_2 \end{aligned} \tag{23}$$

Similarly, we use the law of total variance to determine the variance of the distribution as shown in Equation 24 [83].

$$\begin{aligned} Var(Y) &= E[Var(Y|D)] + Var(E[Y|D]) \\ &= \sum_{d=1}^{2} Var(Y|X_d)P(X_d) + \sum_{d=1}^{2} Var(E[Y|X_d]) \\ &= \rho\sigma_1^2 + (1-\rho)\sigma_2^2 + \rho(\mu_1 - \mu)^2 + (1-\rho)(\mu_2 - \mu)^2 \end{aligned} \tag{24}$$

By rearranging Equations 23 and 24, we are able to estimate μ_1 as shown in Equation 25. Substituting $\hat{\sigma_2}^2 = 4\hat{\sigma_1}^2$ and $\hat{\mu_1}$ into Equation 24 allows us to rearrange and solve for $\hat{\sigma_1}^2$ as shown in Equation 26. Finally, we obtain the estimate $\hat{\sigma_2}$ using Equation 27.

$$\hat{\mu_1} = \frac{\bar{x} - (1-\rho)\cdot\mu_2}{\rho} = \frac{140.268715 - (0.4)(200)}{0.6} = 100.447858 \tag{25}$$

$$\begin{aligned} \hat{\sigma_1} &= \sqrt{\frac{s^2 - \rho\cdot(\hat{\mu_1} - \bar{x})^2 - (1-\rho)\cdot(\mu_2 - \bar{x})^2}{4 - 3\cdot\rho}} \\ &= \sqrt{\frac{55.803025^2 - 0.6\cdot(100.447858 - 140.268715)^2 - 0.4\cdot(200 - 140.268715)^2}{4 - 3(0.6)}} \\ &= 18.283770 \end{aligned} \tag{26}$$

$$\hat{\sigma_2} = 2\hat{\sigma_1} = 2(18.283770) = 36.567540 \tag{27}$$

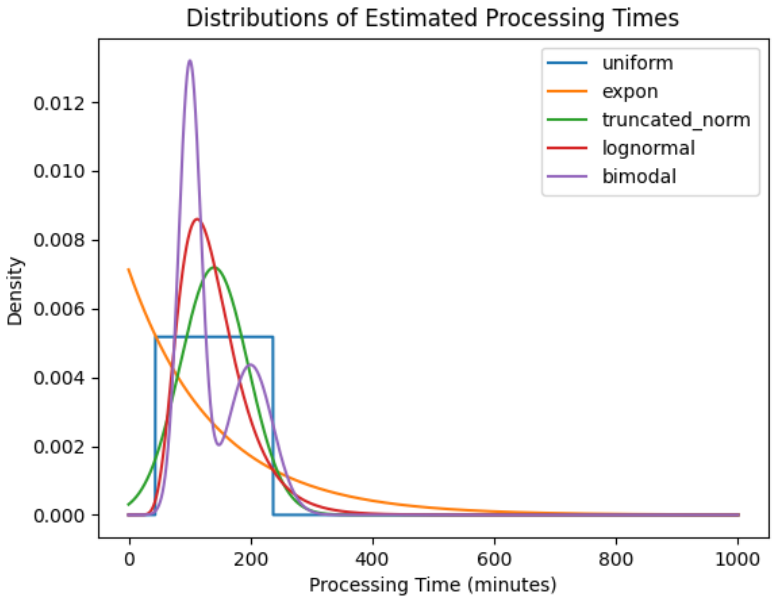


Figure A.1: PDF of estimated processing time distributions.

A.3.2 True Processing Time Distributions

This section focuses on the parameter estimation for the true processing time distributions which include Left Truncated Normal, Uniform, Bounded Normal, and Bounded Uniform. Since the distribution of p_j depends on $\hat{p}_j$, we demonstrate an example of determining the distribution parameters for p_j assuming that $\hat{p}_j = \bar{x} = 140.27$ (Table 5.1). Hence, for this example, $l_j = 126.24$ and $u_j = 168.32$ given $\epsilon^+ = 0.2$ and $\epsilon^- = 0.1$.

Left Truncated Normal Distribution

Given the assumption of $\hat{p}_j = \bar{x}$ used for this example, the parameters for the normal distribution are identical to those for the left truncated normal distribution in Section A.3.1. The distribution is again left truncated ($\gamma = 0$) to avoid negative processing times. The estimated parameters are then $\hat{\mu} = 140.268715$ and $\hat{\sigma} = 3113.977599$.

Uniform Distribution

Again, given the assumption of $\hat{p}_j = \bar{x}$, the parameters for the Uniform distribution are identical to those shown for the Uniform distribution in Section A.3.1 with $\hat{a} = 43.61504$ and $\hat{b} = 236.922390$.

Bounded Normal Distribution

We renormalize the normal distribution between $[l_j, u_j]$. The remaining parameters $\hat{\mu} = \hat{p}_j = 140.268715$ and $\hat{\sigma}^2 = s^2 = 3113.977599$ remain the same.

Bounded Uniform Distribution

After defining l_j and u_j, the parameters for the renormalized Uniform distribution ($\hat{a}'$ and $\hat{b}'$) are simply given by l_j and u_j respectively.

A.4 Tables of DES Framework Inputs

The inputs to the DES framework for determining the pseudo warm-up period and generating the initial states are presented in Tables A.2 and A.3 respectively. Any fields not included in Tables A.2 and A.3 are left blank in the spreadsheet which is read into the DES framework.

A.5 Analysis of the Variability in FTWT and Total Instability Results

Sections A.5.1 and A.5.2 analyze the variability in the FTWT and total instability results for experiment 1 and experiment 2 respectively.

A.5.1 Experiment 1 Analysis of Variability

Given the large number of scenarios considered (45), we investigate the variability in FTWT and total instability for a subset of the scenarios that performed well based on the previous section: *eventJobCompletion*, *eventArrival*, and *periodic180* (performs average and the length of the

Input	Value
num_priority_levels	3
machine_capacity	1
stats_rounding	6
replications	2
store_all_events	1
parent_folder_name	ExperimentalSetup
folder_name	DeterminingWarmUpPeriod
version_num	{1.1, 1.2, 1.3, 1.4}
overwrite_files	1
arrival_distribution	exponential
processing_distribution	uniform
arrival_params	180
processing_params	45, 239
priority_distribution	discrete_uniform
priority_params	1, 3
due_dates	900, 540, 180
num_jobs_random_number_generation	100000
generate_pseudo_warm_up	1
pseudo_warm_up_sim_length	{89280, 803520, 44640, 1036800}
generate_initial_states	0

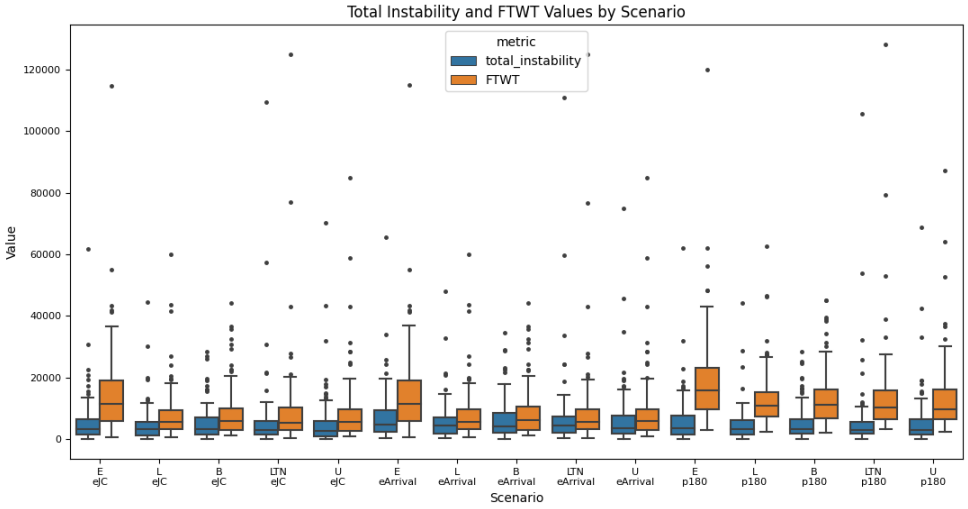
Table A.2: Inputs for generating pseudo warm-up period.

rescheduling interval is in the middle of the periodic policies considered). Variability in the performance metrics, measured using the interquartile range (IQR), is visually represented with the boxplots shown in Figure A.2(a) for the chosen scenario subset. For both metrics, the IQR is largest for the exponential distribution as shown in Figure A.2(a).

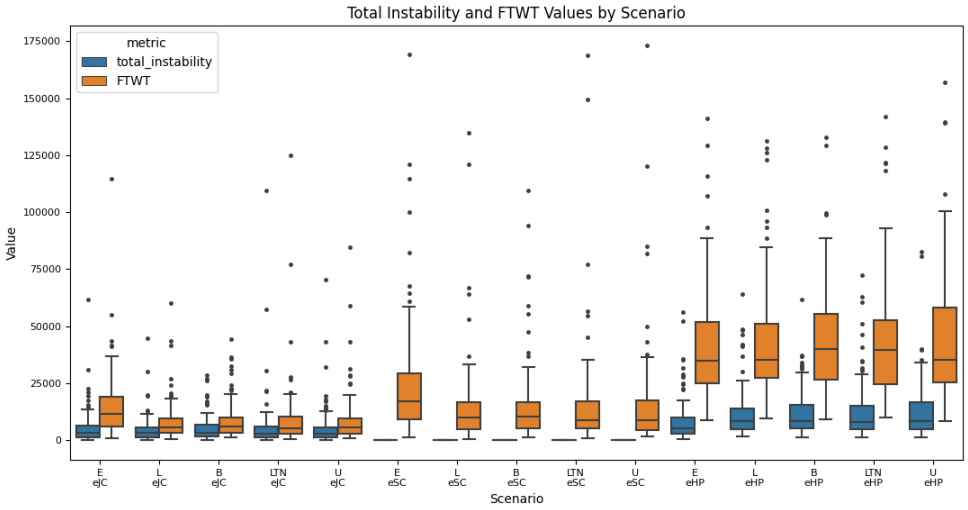
The variability of FTWT and total instability is further explored based on a subset of 'extreme' rescheduling policies: *eventJobCompletion* (performs the best across all distributions), *eventScheduleCompletion* (extreme in its zero total instability), and *eventHighPriority* (performs poorly across all distributions). As demonstrated in Figure A.2(b), *eventHighPriority* performs poorly and has a large IQR in both FTWT and total instability across all distributions. Based on Figures A.2(a) and A.2(b), we see that the Exponential distribution leads to the largest IQR in FTWT and total instability for all rescheduling policies evaluated, except *eventHighPriority*.

Input	Value
num_priority_levels	3
machine_capacity	1
stats_rounding	6
replications	1
store_all_events	1
parent_folder_name	ExperimentalSetup
folder_name	GeneratingInitialStates
version_num	1
overwrite_files	1
arrival_distribution	exponential
processing_distribution	uniform
arrival_params	180
processing_params	45, 239
priority_distribution	discrete_uniform
priority_params	1, 3
due_dates	900, 540, 180
num_jobs_random_number_generation	100000
generate_pseudo_warm_up	0
generate_initial_states	1
pseudo_warm_up_period	129600
num_initial_states	100

Table A.3: Inputs for generating initial states.



(a) Boxplot of total instability and FTWT by scenario for well-performing policies.



(b) Boxplot of total instability and FTWT by scenario for extreme policies.

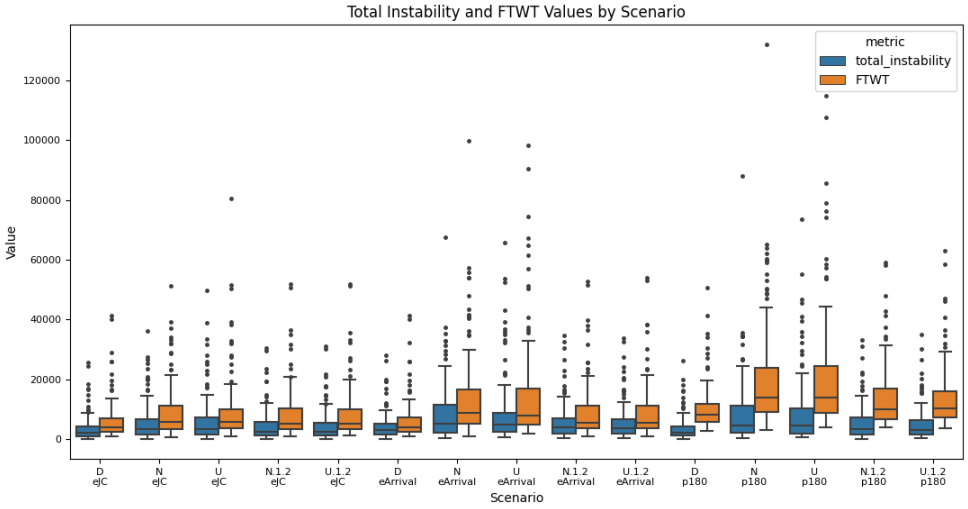
Figure A.2: Boxplots of total instability (minutes) and FTWT (weighted minutes) by scenario for chosen scenario subsets of the stochastic experiment with $n = 100$. The x-axis represents the scenario, where the first line refers to the processing time distribution with Exponential (*E*), lognormal (*L*), Bimodal (*B*), Left Truncated Normal (*LTN*), and Uniform (*U*). The second line of the x-axis label refers to the rescheduling polices of *eventJobCompletion* (*eJC*), *eventArrival* (*eArrival*), *periodic180* (*p180*), *eventScheduleCompletion* (*eSC*), and *eventHighPriority* (*eHP*).

A.5.2 Experiment 2 Analysis of Variability

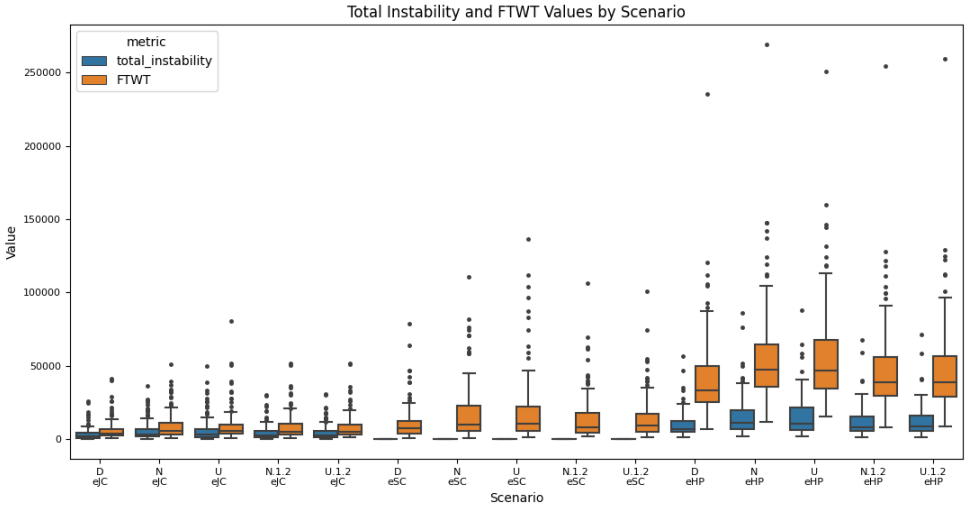
We investigate the variability in total instability and FTWT for a subset of the scenarios which performed well to moderately well: *eventJobCompletion*, *eventArrival*, and *periodic180*. The variability in the results, measured using the interquartile range (IQR), is visually represented with boxplots as shown in Figure A.2(a) for the scenario subset. Across nearly all rescheduling policies considered, the Normal and Uniform distributions have larger IQR in both metrics than the other distributions [1]. Both of these unbounded distributions have larger variance in their processing times, compared to the bounded distributions or deterministic case (see Table 5.5 for an example).

Figure A.3(b) depicts boxplots for the same subset of 'extreme' policies as in experiment 1. The figure demonstrates a large IQR in both performance metrics for *eventHighPriority* (which also performs poorly in both performance metrics).

[1] The only exception is the IQR of the Uniform and bounded Uniform distributions for the *eventJobCompletion* policy which are similar.



(a) Boxplot of total instability and FTWT by scenario for a scenario subset of well performing policies.



(b) Boxplot of total instability and FTWT by scenario for a scenario subset of extreme policies.

Figure A.3: Boxplots of total instability (minutes) and FTWT (weighted minutes) by scenario for chosen scenario subsets of the stochastic experiment with $n = 100$. The x-axis represents the scenario, where the first line refers to the processing time distributions of deterministic (*D*), Normal (*N*), Uniform (*U*), Normal[0.1, 0.2] (*N.1.2*), and Uniform[0.1, 0.2] (*U.1.2*). The second line of the x-axis label refers to the rescheduling polices of *eventJobCompletion* (*eJC*), *eventArrival* (*eArrival*), *periodic180* (*p180*), *eventScheduleCompletion* (*eSC*), and *eventHighPriority* (*eHP*).

A.6 Analysis of the Percent of Subotimal Solutions

In this section, we report the full analysis regarding the subproblems which result in suboptimal solutions due to exceeding the user-defined time limit of 300 seconds. The results are aggregated by scenario in Tables A.4 and A.5 for experiment 1 and experiment 2 respectively. Each table records the total number of rescheduling actions taken over the 100 replications, the total number of subproblems which exceeded the time limit (over all 100 replications), the percent of subproblems which exceeded the time limit, and the average relative optimality gap of the subproblems which exceeded the time limit.

Scenario					
Rescheduling Policy	Distribution	Total NRA	Total Exceeded Time Limit	Percent Exceed Time Limit	Average Optimality Gap
eventArrival	Exponential	6366	3	4.71E-02	0.09
eventArrival	Lognormal	6137	20	3.26E-01	0.13
eventArrival	Bimodal	6070	25	4.12E-01	0.11
eventArrival	LeftTruncatedNormal	6277	106	1.69E+00	0.09
eventArrival	Uniform	6151	85	1.38E+00	0.10
eventHighPriority	Exponential	2387	23	9.64E-01	0.07
eventHighPriority	Lognormal	2371	157	6.62E+00	0.10
eventHighPriority	Bimodal	2380	171	7.18E+00	0.09
eventHighPriority	LeftTruncatedNormal	2367	176	7.44E+00	0.10
eventHighPriority	Uniform	2417	167	6.91E+00	0.09
eventJobCompletion	Exponential	7636	3	3.93E-02	0.09
eventJobCompletion	Lognormal	7524	23	3.06E-01	0.13
eventJobCompletion	Bimodal	7383	28	3.79E-01	0.09
eventJobCompletion	LeftTruncatedNormal	7609	112	1.47E+00	0.07
eventJobCompletion	Uniform	7490	90	1.20E+00	0.09
eventScheduleCompletion	Exponential	4364	8	1.83E-01	0.08
eventScheduleCompletion	Lognormal	4918	6	1.22E-01	0.06
eventScheduleCompletion	Bimodal	4732	5	1.06E-01	0.08
eventScheduleCompletion	LeftTruncatedNormal	4821	8	1.66E-01	0.10
eventScheduleCompletion	Uniform	4817	10	2.08E-01	0.08
hybrid220HighPriority	Exponential	7324	2	2.73E-02	0.13
hybrid220HighPriority	Lognormal	7046	20	2.84E-01	0.15
hybrid220HighPriority	Bimodal	6975	23	3.30E-01	0.10
hybrid220HighPriority	LeftTruncatedNormal	7174	91	1.27E+00	0.08
hybrid220HighPriority	Uniform	7057	68	9.64E-01	0.10
hybrid220ScheduleCompletion	Exponential	6509	2	3.07E-02	0.13
hybrid220ScheduleCompletion	Lognormal	6576	14	2.13E-01	0.13
hybrid220ScheduleCompletion	Bimodal	6532	20	3.06E-01	0.09
hybrid220ScheduleCompletion	LeftTruncatedNormal	6670	56	8.40E-01	0.08
hybrid220ScheduleCompletion	Uniform	6570	40	6.09E-01	0.08
periodic140	Exponential	8179	1	1.22E-02	0.03
periodic140	Lognormal	7864	20	2.54E-01	0.11
periodic140	Bimodal	7800	23	2.95E-01	0.08
periodic140	LeftTruncatedNormal	8004	102	1.27E+00	0.07
periodic140	Uniform	7887	78	9.89E-01	0.10
periodic180	Exponential	6406	2	3.12E-02	0.06
periodic180	Lognormal	6176	14	2.27E-01	0.11
periodic180	Bimodal	6092	18	2.95E-01	0.09
periodic180	LeftTruncatedNormal	6275	77	1.23E+00	0.09
periodic180	Uniform	6169	57	9.24E-01	0.10
periodic220	Exponential	5270	2	3.80E-02	0.12
periodic220	Lognormal	5090	9	1.77E-01	0.12
periodic220	Bimodal	5061	21	4.15E-01	0.09
periodic220	LeftTruncatedNormal	5170	63	1.22E+00	0.09
periodic220	Uniform	5096	48	9.42E-01	0.11

Table A.4: Data on the suboptimal solutions arising from the time limit imposed on the MIP models used to solve the subproblems for the deterministic experiment by scenario.

Scenario					
Rescheduling Policy	Distribution	Total NRA	Total Exceeded Time Limit	Percent Exceed Time Limit	Average Optimality Gap
periodic180	TrueNormal	6582	102	1.55	0.07
periodic180	Normal[.1,.2]	6079	25	0.41	0.06
periodic180	TrueUniform	6691	136	2.03	0.06
periodic180	Uniform[.1,.2]	6077	26	0.43	0.07
periodic180	Deterministic	5954	5	0.08	0.06
hybrid220HighPriority	TrueNormal	5407	80	1.48	0.07
hybrid220HighPriority	Normal[.1,.2]	5050	38	0.75	0.07
hybrid220HighPriority	TrueUniform	5502	109	1.98	0.05
hybrid220HighPriority	Uniform[.1,.2]	5035	37	0.73	0.07
hybrid220HighPriority	Deterministic	4899	4	0.08	0.10
eventScheduleCompletion	TrueNormal	4435	6	0.14	0.07
eventScheduleCompletion	Normal[.1,.2]	4679	2	0.04	0.05
eventScheduleCompletion	TrueUniform	4443	6	0.14	0.11
eventScheduleCompletion	Uniform[.1,.2]	4676	0	0.00	
eventScheduleCompletion	Deterministic	5081	2	0.04	0.08
eventJobCompletion	TrueNormal	7323	22	0.30	0.07
eventJobCompletion	Normal[.1,.2]	7188	27	0.38	0.08
eventJobCompletion	TrueUniform	7326	23	0.31	0.10
eventJobCompletion	Uniform[.1,.2]	7185	26	0.36	0.07
eventJobCompletion	Deterministic	7280	11	0.15	0.07
eventHighPriority	TrueNormal	2546	218	8.56	0.08
eventHighPriority	Normal[.1,.2]	2411	144	5.97	0.09
eventHighPriority	TrueUniform	2584	256	9.91	0.08
eventHighPriority	Uniform[.1,.2]	2419	156	6.45	0.09
eventHighPriority	Deterministic	2341	110	4.70	0.09
eventArrival	TrueNormal	6310	109	1.73	0.08
eventArrival	Normal[.1,.2]	5970	33	0.55	0.09
eventArrival	TrueUniform	6510	126	1.94	0.06
eventArrival	Uniform[.1,.2]	5975	35	0.59	0.09
eventArrival	Deterministic	5860	10	0.17	0.07

Table A.5: Data on the suboptimal solutions arising from the time limit imposed on the MIP models used to solve the subproblems for the stochastic experiment by scenario.

Bibliography

[1] B. Cardoen, E. Demeulemeester, and J. Beliën, "Operating room planning and scheduling: A literature review," *Eur. J. Oper. Res.*, vol. 201, no. 3, pp. 921–932, 2010, doi: 10.1016/j.ejor.2009.04.011.

[2] M. Samudra, C. Van Riet, E. Demeulemeester, B. Cardoen, N. Vansteenkiste, and F. E. Rademakers, "Scheduling operating rooms: achievements, challenges and pitfalls," *J. Scheduling*, vol. 19, no. 5, pp. 493–525, 2016, doi: 10.1007/s10951-016-0489-6.

[3] J. Walonoski *et al.*, "Synthea: An approach, method, and software mechanism for generating synthetic patients and the synthetic electronic health care record," *J. Amer. Med. Inform. Assoc.*, vol. 25, no. 3, pp. 230–238, 2018, doi: 10.1093/jamia/ocx079.

[4] C. Dosi, M. Iori, A. Kramer, and M. Vignoli, "Facing implementation barriers to healthcare simulation studies," in *Health Care Syst. Eng.*, 2020, pp. 117–129, doi: 10.1007/978-3-030-39694-7_10.

[5] G. Stiglic, P. Kocbek, N. Fijacko, M. Zitnik, K. Verbert, and L. Cilar, "Interpretability of machine learning-based prediction models in healthcare," *Wiley Interdisciplinary Reviews: Data Mining Knowl. Discovery*, vol. 10, no. 5, 2020, doi: 10.1002/widm.1379.

[6] K. Morrison, "Artificial intelligence and the NHS: a qualitative exploration of the factors influencing adoption." *Future Healthcare J.*, vol. 8, no. 3, pp. e648–e654, 2021, doi: 10.7861/fhj.2020-0258.

[7] S. C. Brailsford *et al.*, "Overcoming the barriers: a qualitative study of simulation adoption in the NHS," *J. Oper. Res. Soc.*, vol. 64, no. 2, pp. 157–168, 2013, doi: 10.1057/jors.2011.130.

[8] G. E. Vieira, J. W. Herrmann, and E. Lin, "Rescheduling manufacturing systems: A framework of strategies, policies, and methods," *J. Scheduling*, vol. 6, no. 1, pp. 39–62, 2003, doi: 10.1023/A:1022235519958.

[9] R. Larsen and M. Pranzo, "A framework for dynamic rescheduling problems," *Int. J. Prod. Res.*, vol. 57, no. 1, pp. 16–33, 2019, doi: 10.1080/00207543.2018.1456700.

[10] F. Ballestin, A. Perez, and S. Quintanilla, "Scheduling and rescheduling elective patients in operating rooms to minimise the percentage of tardy patients," *J. Scheduling*, vol. 22, no. 1, pp. 107–118, 2019, doi: 10.1007/s10951-018-0570-4.

[11] D. Ouelhadj and S. Petrovic, "A survey of dynamic scheduling in manufacturing systems," *J. Scheduling*, vol. 12, no. 4, pp. 417–431, 2009, doi: 10.1007/s10951-008-0090-8.

[12] P. Cowling and M. Johansson, "Using real time information for effective dynamic scheduling," *Eur. J. Oper. Res.*, vol. 139, no. 2, pp. 230–244, 2002, doi: 10.1016/S0377-2217(01)00355-1.

[13] D. Montana, "A comparison of combinatorial optimization and dispatch rules for online scheduling," in *Proc. 2nd Multidisciplinary Conf. Scheduling: Theory Applications*, 2005, pp. 353–362. [Online]. Available: http://www.davidmontana.net/papers/mista05.pdf

[14] G. E. Vieira, J. W. Herrmann, and E. Lin, "Analytical models to predict the performance of a single-machine system under periodic and event-driven rescheduling strategies," *Int. J. Prod. Res.*, vol. 38, no. 8, pp. 1899–1915, 2000, doi: 10.1080/002075400188654.

[15] A. Pfeiffer, B. Kádár, and L. Monostori, "Stability-oriented evaluation of rescheduling strategies, by using simulation," *Comput. Industry*, vol. 58, no. 7, pp. 630–643, 2007, doi: 10.1016/j.compind.2007.05.009.

[16] M. Thomas and H. Szczerbicka, "Evaluating online scheduling techniques in uncertain environments," in *Proc. 3rd Multidisciplinary Int. Scheduling Conf. (MISTA 2007)*, 2007. [Online]. Available: https://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=4464A28EAF841032BE3684A04E8B9BFE?doi=10.1.1.87.454&rep=rep1&type=pdf

[17] H. Aytug, M. A. Lawley, K. McKay, S. Mohan, and R. Uzsoy, "Executing production schedules in the face of uncertainties: A review and some future directions," *Eur. J. Oper. Res.*, vol. 161, no. 1, pp. 86–110, 2005, doi: 10.1016/j.ejor.2003.08.027.

[18] J. Bidot, T. Vidal, P. Laborie, and J. C. Beck, "A theoretic and practical framework for scheduling in a stochastic environment," *J. Scheduling*, vol. 12, no. 3, pp. 315–344, 2009, doi: 10.1007/s10951-008-0080-x.

[19] A. D. Filippo, M. Lombardi, and M. Milano, "The blind men and the elephant: Integrated offline/online optimization under uncertainty," in *Proc. Twenty-Ninth Int. Joint Conf. Artif. Intell. (IJCAI-20)*, Jul. 2020, pp. 4840–4846, doi: 10.24963/ijcai.2020/674.

[20] K. Hozak and J. Hill, "Issues and opportunities regarding replanning and rescheduling frequencies," *Int. J. Prod. Res.*, vol. 47, no. 18, pp. 4955–4970, 2009, doi: 10.1080/00207540802047106.

[21] Z. Bahroun, A. Shamayleh, R. As'ad, and R. Zakaria, "Flexible decision support tool for dynamic single machine scheduling problems," *J. Ind. Prod. Eng.*, vol. 38, no. 3, pp. 213–238, 2021, doi: 10.1080/21681015.2021.1883135.

[22] J. Branke and D. C. Mattfeld, "Anticipation and flexibility in dynamic scheduling," *Int. J. Prod. Res.*, vol. 43, no. 15, pp. 3103–3129, 2005, doi: 10.1080/00207540500077140.

[23] L. K. Church and R. Uzsoy, "Analysis of periodic and event-driven rescheduling policies in dynamic shops," *Int. J. Comput. Integr. Manuf.*, vol. 5, no. 3, pp. 153–163, 1992, doi: 10.1080/09511929208944524.

[24] S. D. Wu, R. H. Storer, and P.-C. Chang, "One-machine rescheduling heuristics with efficiency and stability as criteria," *Comput. Operations Res.*, vol. 20, no. 1, pp. 1–14, 1993, doi: 10.1016/0305-0548(93)90091-V.

[25] C. Akkan, "Improving schedule stability in single-machine rescheduling for new operation insertion," *Comput. Operations Res.*, vol. 64, pp. 198–209, 2015, doi: 10.1016/j.cor.2015.05.015.

[26] S. D. Wu, R. H. Storer, and P.-C. Chang, "A rescheduling procedure for manufacturing systems under random disruptions," in *New Directions Operations Res. Manuf.*, 1992, pp. 292–306, doi: 10.1007/978-3-642-77537-6_18.

[27] M. Pinedo, *Planning and scheduling in manufacturing and services*, 2nd ed. New York, NY, USA: Springer, 2009.

[28] K. Søreide *et al.*, "Immediate and long-term impact of the covid-19 pandemic on delivery of surgical services." *Brit. J. Surgery*, vol. 107, no. 10, pp. 1250–1261, 2020, doi: 10.1002/bjs.11670.

[29] I. Rahimi and A. H. Gandomi, "A comprehensive review and analysis of operating room and surgery scheduling," *Arch. Comput. Methods Eng.*, vol. 28, no. 3, pp. 1667–1688, 2020, doi: 10.1007/s11831-020-09432-2.

[30] L. Wang, E. Demeulemeester, N. Vansteenkiste, and F. E. Rademakers, "Operating room planning and scheduling for outpatients and inpatients: A review and future research," *Operations Res. Health Care*, vol. 31, 2021, doi: 10.1016/j.orhc.2021.100323.

[31] B. Denton, J. Viapiano, and A. Vogl, "Optimization of surgery sequencing and scheduling decisions under uncertainty," *Health Care Manage. Sci.*, vol. 10, no. 1, pp. 13–24, 2007, doi: 10.1007/s10729-006-9005-4.

[32] I. Marques and M. E. Captivo, "Different stakeholders' perspectives for a surgical case assignment problem: Deterministic and robust approaches," *Eur. J. Oper. Res.*, vol. 261, no. 1, pp. 260–278, 2017, doi: 10.1016/j.ejor.2017.01.036.

[33] T. Khaniyev, E. Kayış, and R. Güllü, "Next-day operating room scheduling with uncertain surgery durations: Exact analysis and heuristics," *Eur. J. Oper. Res.*, vol. 286, no. 1, pp. 49–62, 2020, doi: 10.1016/j.ejor.2020.03.002.

[34] Y. Zhou, M. Parlar, V. Verter, and S. Fraser, "Surgical scheduling with constrained patient waiting times," *Prod. Operations Manage.*, vol. 30, no. 9, pp. 3253–3271, 2021, doi: 10.1111/poms.13427.

[35] R. W. Allen, K. M. Taaffe, and G. Ritchie, "Surgery rescheduling using discrete event simulation: A case study," in *Proc. Winter Simul. Conf. (WSC)*, 2014, pp. 1365–1376, doi: 10.1109/WSC.2014.7019991.

[36] G. Xiao, W. van Jaarsveld, M. Dong, and J. van de Klundert, "Models, algorithms and performance analysis for adaptive operating room scheduling," *Int. J. Prod. Res.*, vol. 56, no. 4, pp. 1389–1413, 2018, doi: 10.1080/00207543.2017.1328140.

[37] E. Erdem, X. Qu, and J. Shi, "Rescheduling of elective patients upon the arrival of emergency patients," *Decis. Support Syst.*, vol. 54, no. 1, pp. 551–563, 2012, doi: 10.1016/j.dss.2012.08.002.

[38] B. Addis, G. Carello, A. Grosso, and E. Tanfani, "Operating room scheduling and rescheduling: a rolling horizon approach," *Flexible Services Manuf. J.*, vol. 28, no. 1-2, pp. 206–232, 2016, doi: 10.1007/s10696-015-9213-7.

[39] F. Davarian and J. Behnamian, "Robust finite-horizon scheduling/rescheduling of operating rooms with elective and emergency surgeries under resource constraints," *J. Scheduling*, vol. 25, no. 6, pp. 625–641, 2022, doi: 10.1007/s10951-022-00741-x.

[40] K. Stuart and E. Kozan, "Reactive scheduling model for the operating theatre," *Flexible Services Manuf. J.*, vol. 24, no. 4, pp. 400–421, 2012, doi: 10.1007/s10696-011-9111-6.

[41] J. T. van Essen, J. L. Hurink, W. Hartholt, and B. J. van den Akker, "Decision support system for the operating room rescheduling problem," *Health Care Manage. Sci.*, vol. 15, no. 4, pp. 355–372, 2012, doi: 10.1007/s10729-012-9202-2.

[42] M. Eshghali, D. Kannan, N. Salmanzadeh-Meydani, and A. M. Esmaieeli Sikaroudi, "Machine learning based integrated scheduling and rescheduling for elective and emergency patients in the operating theatre," *Ann. Operations Res.*, pp. 1–24, 2023, doi: 10.1007/s10479-023-05168-x.

[43] F. A. Van Der Schoot and R. M. De Carvalho, "Reducing rescheduling events in operating rooms by applying pattern recognition techniques," in *8th Int. Conf. Internet Things: Syst., Manage., Secur. (IOTSMS)*, pp. 1–8, doi: 10.1109/IOTSMS53705.2021.9705008.

[44] N. Dellaert and J. Jeunet, "Hospital admission planning to optimize major resources utilization under uncertainty," in *3rd World Conf. Prod. Operations Manage., POM TOKYO: "Manuf. Fundam.: Necessity Sufficiency"*, 2008, Accessed: Feb. 1, 2021. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01557188

[45] C. G. Corlu, J. Maleyeff, J. Wang, K. Yip, and J. Farris, "Real-time nurse dispatching using dynamic priority decision framework," in *Winter Simul. Conf. (WSC)*, 2020, pp. 782–793, doi: 10.1109/WSC48552.2020.9384076.

[46] S. Hahn-Goldberg, M. W. Carter, J. C. Beck, M. Trudeau, P. Sousa, and K. Beattie, "Dynamic optimization of chemotherapy outpatient scheduling with uncertainty," *Health Care Manage. Sci.*, vol. 17, no. 4, pp. 379–392, 2014, doi: 10.1007/s10729-014-9268-0.

[47] N. Kortbeek *et al.*, "Designing cyclic appointment schedules for outpatient clinics with scheduled and unscheduled patient arrivals," *Perform. Eval.*, vol. 80, pp. 5–26, 2014, doi: 10.1016/j.peva.2014.06.003.

[48] P. Hooshangi-Tabrizi, I. Contreras, N. Bhuiyan, and G. Batist, "Improving patient-care services at an oncology clinic using a flexible and adaptive scheduling procedure," *Expert Syst. Appl.*, vol. 150, 2020, doi: 10.1016/j.eswa.2020.113267.

[49] S. A. Erdogan and B. Denton, "Dynamic appointment scheduling of a stochastic server with uncertain demand," *INFORMS J. Comput.*, vol. 25, no. 1, pp. 116–132, 2013, doi: 10.1287/ijoc.1110.0482.

[50] H. Liu, T. Zhang, S. Luo, and D. Xu, "Operating room scheduling and surgeon assignment problem under surgery durations uncertainty," *Technol. Health Care*, vol. 26, no. 2, pp. 297–304, 2018, doi: 10.3233/THC-170825.

[51] A. Baumgart, A. Zoeller, C. Denz, H. Bender, A. Heinzl, and E. Badreddin, "Using computer simulation in operating room management: Impacts on process engineering and performance," in *40th Annu. Hawaii Int. Conf. Syst. Sci. (HICSS'07)*, Jan. 2007, pp. 2113–2122, doi: 10.1109/HICSS.2007.593.

[52] S. Choi and W. E. Wilhelm, "An analysis of sequencing surgeries with durations that follow the lognormal, gamma, or normal distribution," *IIE Trans. Healthcare Syst. Eng.*, vol. 2, no. 2, pp. 156–171, 2012, doi: 10.1080/19488300.2012.684272.

[53] M. Varmazyar, R. Akhavan-Tabatabaei, N. Salmasi, and M. Modarres, "Operating room scheduling problem under uncertainty: Application of continuous phase-type distributions," *IISE Trans.*, vol. 52, no. 2, pp. 216–235, 2020, doi: 10.1080/24725854.2019.1628372.

[54] J. H. May, D. P. Strum, and L. G. Vargas, "Fitting the lognormal distribution to surgical procedure times*," *Decis. Sci.*, vol. 31, no. 1, pp. 129–148, 2000, doi: 10.1111/j.1540-5915.2000.tb00927.x.

[55] M. J Murray, "The canadian triage and acuity scale: A canadian perspective on emergency department triage," *Emergency Med.*, vol. 15, no. 1, pp. 6–10, 2003, doi: 10.1046/j.1442-2026.2003.00400.x.

[56] American College of Surgeons, "Covid-19: Guidance for triage of non-emergent surgical procedures," facs.org, accessed Nov. 11, 2022. [Online]. Available: https://www.facs.org/for-medical-professionals/covid-19/clinical-guidance/triage/

[57] Y. Jozaghi *et al.*, "Endocrine surgery in the coronavirus disease 2019 pandemic: Surgical triage guidelines," *Head & Neck*, vol. 42, no. 6, pp. 1325–1328, 2020, doi: 10.1002/hed.26169.

[58] R. L. Graham *et al.*, "Optimization and approximation in deterministic sequencing and scheduling: a survey," in *Ann. Discrete Math.* Elsevier, 1979, vol. 5, pp. 287–326, doi: 10.1016/S0167-5060(08)70356-X.

[59] M. Pinedo, "Stochastic scheduling with release dates and due dates," *Operations Res.*, vol. 31, no. 3, pp. 559–572, 1983, doi: 10.1287/opre.31.3.559.

[60] A. M. W. Law, *Simulation Modeling and Analysis*, 5th ed. New York, NY, USA: McGraw-Hill New York, 2015, Accessed: Nov. 26, 2022. [Online]. Available: https://www.amazon.ca/Simulation-Mcgraw-hill-Industrial-Engineering-Management-ebook/dp/B00VF61N4A/ref=tmm_kin_swatch_0?_encoding=UTF8&qid=&sr=

[61] R. D. Wright and T. E. J. Ramsay, "On the effectiveness of common random numbers," *Manage. Sci*, vol. 25, no. 7, pp. 649–656, 1979, doi: 10.1287/mnsc.25.7.649.

[62] M. Pinedo, *Scheduling: theory, algorithms, and systems*, 5th ed. New York, NY, USA: Springer, 2016.

[63] W.-Y. Ku and J. C. Beck, "Mixed integer programming models for job shop scheduling: A computational analysis," *Comput. Operations Res.*, vol. 73, pp. 165–173, 2016, doi: 10.1016/j.cor.2016.04.006.

[64] M. Haviv, *Queues: a course in queueing theory*. New York, NY, USA: Springer, 2013, doi: 10.1007/978-1-4614-6765-6.

[65] O. Lünsdorf and S. Scherfke, *SimPy: Discrete event simulation for Python 4.0.1*, Accessed: Mar. 1, 2021. [Online]. Available: https://simpy.readthedocs.io/en/4.0.1/

[66] IBM, *IBM® Decision Optimization CPLEX® Modeling for Python (DOcplex) V2.25*, Accessed: Jan. 15, 2021. [Online]. Available: https://ibmdecisionoptimization.github.io/docplex-doc/

[67] P. Virtanen *et al.*, "Scipy 1.0: fundamental algorithms for scientific computing in python," *Nature Methods*, vol. 17, no. 3, pp. 261–272, 2020, doi: 10.1038/s41592-019-0686-2.

[68] L. F. A. de Carvalho, K. Fierens, and M. Kint, "Mini-laparoscopic versus conventional laparoscopic cholecystectomy: a randomized controlled trial." *J. Laparoendoscopic Adv. Techn.*, vol. 23, no. 2, pp. 109–16, 2013, doi: 10.1089/lap.2012.0349.

[69] fitter, *FITTER documentation (1.5.1)*, Accessed: Apr. 15, 2022. [Online]. Available: https://fitter.readthedocs.io/en/latest/index.html

[70] A. M. Cao, G. D. Eslick, and M. R. Cox, "Early laparoscopic cholecystectomy is superior to delayed acute cholecystitis: a meta-analysis of case-control studies." *Surgical endoscopy*, vol. 30, no. 3, pp. 1172–82, 2016, doi: 10.1007/s00464-015-4325-4.

[71] N. T. Thomopoulos, *Probability distributions: with truncated, log and bivariate extensions*. Cham, Switzerland: Springer, 2018, doi: 10.1007/978-3-319-76042-1.

[72] P. S. Mahajan and G. Ingalls, "Evaluation of methods used to detect warm-up period in steady state simulation," in *Proc. 2004 Winter Simul. Conf. (WSC)*, 2004, pp. 671–680, doi: 10.1109/WSC.2004.1371374.

[73] M. Rossetti, *Simulation Modeling and Arena*, 3rd and Open Text ed. Retrieved from https://rossetti.github.io/RossettiArenaBook/ licensed under the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License., 2021.

[74] M. K. Smith, "Common misteakes in using statistics: Spotting and avoiding them – multiple inference," web.ma.utexas.edu, 2013, Accessed: Jan. 17, 2023. [Online]. Available: https://web.ma.utexas.edu/users/mks/statmistakes/multipleinference.html

[75] M. Goldman, "Stat C141: Statistics for bioinformatics," UC Berkeley, 2008, Accessed: Jan. 17, 2023. [Online]. Available: https://www.stat.berkeley.edu/~mgoldman/Section0402.pdf

[76] W. S. Lovejoy and Y. Li, "Hospital operating room capacity expansion," *Manage. Sci*, vol. 48, no. 11, pp. 1369–1387, 2002, doi: 10.1287/mnsc.48.11.1369.266.

[77] O. EL-Rifai, T. Garaix, V. Augusto, and X. Xie, "A stochastic optimization model for shift scheduling in emergency departments," *Health Care Manage. Sci.*, vol. 18, no. 3, pp. 289–302, 2015, doi: 10.1007/s10729-014-9300-4.

[78] N. Megow, M. Uetz, and T. Vredeveld, "Models and algorithms for stochastic online scheduling," *Math. Operations Res.*, vol. 31, no. 3, pp. 513–525, 2006, doi: 10.1287/moor.1060.0201.

[79] F. Rezaei, A. A. Najafi, R. Ramezanian, and E. Demeulemeester, "Simulation-based priority rules for the stochastic resource-constrained net present value and risk problem," *Comput. Ind. Eng.*, vol. 160, 2021, doi: 10.1016/j.cie.2021.107607.

[80] T. Boudoukh, M. Penn, and G. Weiss, "Scheduling jobshops with some identical or similar jobs," *J. Scheduling*, vol. 4, no. 4, pp. 177–199, 2001, doi: 10.1002/jos.72.

[81] D. Terekhov, T. T. Tran, and J. C. Beck, "Investigating two-machine dynamic flow shops based on queueing and scheduling," in *Proc Twentieth Int. Conf. Automated Planning Scheduling (ICAPS 2010)*, Accessed: Sep. 29, 2020. [Online]. Available: https://tidel.mie.utoronto.ca/pubs/flowShopQueueingScheduling_ICAPS10_workshop.pdf

[82] D. Terekhov, T. Tran, D. Down, and J. C. Beck, "Long-run stability in dynamic scheduling," in *Proc. Twenty-Second Int. Conf. Automated Planning Scheduling (ICAPS 2012)*, no. 1, 2012, Accessed: Sep. 29, 2020. [Online]. Available: https://ojs.aaai.org/index.php/ICAPS/article/view/13524/13373

[83] J. K. Blitzstein and J. Hwang, *Introduction to probability*. New York, NY, USA: Chapman and Hall/CRC, 2014, doi: https://doi.org/10.1201/b17221.