Journal of Scheduling manuscript No.
(will be inserted by the editor)

Evaluating periodic rescheduling policies using a rolling horizon framework in an industrial-scale multipurpose plant

Zachariah Stevenson · Ricardo Fukasawa · Luis Ricardez-Sandoval

Received: date / Accepted: date

Abstract Periodic rescheduling is a commonly used method for scheduling short term operations. Through computational experiments that vary plant parameters, such as the load and the capacity of a facility, we investigate the effects these parameters have on plant performance under periodic rescheduling. The results show that choosing a suitable rescheduling policy depends highly on some key plant parameters. In particular, by modifying various parameters of the facility, the performance ranking of the various rescheduling policies may be reversed compared to the results obtained with nominal parameter values. This highlights the need to consider both facility characteristics and what the crucial objective of the facility is when selecting a rescheduling policy. This study considers a variant of the job shop problem, used to model the operation of an industrial-scale analytical services facility using different periodic rescheduling policies. A rolling horizon routine is used to schedule operations over the scheduling horizon. Performance is measured in terms of job throughput, job makespan, and proportion of jobs on time at the end of the scheduling horizon to obtain a more complete understanding of how performance varies between rescheduling policies.

Keywords Scheduling · Rescheduling · Fixed-Periodic · Frequency

1 Introduction

Scheduling is an integral part of any production operation. A schedule dictates what operations should be performed and when, in order to optimize a particular metric such as makespan, throughput, or profit. Proper scheduling can greatly

Zachariah Stevenson
University of Waterloo, Waterloo, Ontario N2L 3G1, Canada

Ricardo Fukasawa
University of Waterloo, Waterloo, Ontario N2L 3G1, Canada

Luis Ricardez-Sandoval
University of Waterloo, Waterloo, Ontario N2L 3G1, Canada E-mail: laricardezsandoval@uwaterloo.ca
ORCID: 0000-0001-9867-6778

increase the efficiency of a production plant and therefore is of great practical importance. Typically, works on scheduling have considered finding an optimal schedule in a static environment where all of the operating conditions are known with certainty throughout the scheduling horizon. While it is necessary to first understand scheduling in this context, in practice it may not always be realistic to assume that all of the information pertinent to generating an optimal schedule will be known in advance, and that no unexpected disruptions to the schedule will occur during operation.

In particular, an initially generated schedule can become infeasible or non-optimal because of uncertainties, such as a machine breaking down, the arrival of a rush order, or actual processing times differing from expected processing times (Baykasoğlu and Karaslan, 2017; Pfund and Fowler, 2017; Vieira et al., 2003; Hozak and Hill, 2009; Ouelhadj and Petrovic, 2008). These events motivate the need to reschedule operations in practice. The decisions involved with rescheduling can be divided into two issues of 'how-to' and 'when-to' reschedule (Sabuncuoglu and Kizilisik, 2003). The 'how-to' addresses how new schedules should be generated. Some examples include a full rescheduling of all operations or a partial rescheduling, where some operations that were previously scheduled remain fixed (Vieira et al., 2003; Sabuncuoglu and Kizilisik, 2003). The 'when-to' addresses when new schedules should be generated. Thorough reviews on process rescheduling are available elsewhere, for instance in Vieira et al. (2003).

Multiple works have been conducted over the last few decades investigating how rescheduling frequency affects schedule performance. There have been studies conducted in a variety of environments such as job shop scheduling (Baykasoğlu and Karaslan, 2017; Muhlemann et al., 1982; Church and Uzsoy, 1992; Shafaei and Brunn, 1999; Vieira et al., 2000a,b), chemical production scheduling (Gupta and Maravelias, 2016; Koller et al., 2018), material requirements planning (Yano and Carlson, 1987), and scheduling flexible manufacturing systems (Pfund and Fowler, 2017; Sabuncuoglu and Kizilisik, 2003; Kim and Kim, 1994; Sabuncuoglu and Karabuk, 1999). Many different types of disturbances have been considered as well, such as new job arrivals, random processing times, machine breakdowns, due date modifications, and rush order arrivals (Vieira et al., 2003; Ouelhadj and Petrovic, 2008).

Despite these efforts, a general agreement on how rescheduling frequency affects performance has not been conclusive in the literature. Some works suggest that both scheduling too frequently and not frequently enough result in decreased performance (Kim and Kim, 1994; Gupta and Maravelias, 2016). On the other hand, other studies suggest that one should reschedule as frequently as possible (Pfund and Fowler, 2017; Muhlemann et al., 1982; Shafaei and Brunn, 1999), although relative benefits may drop off after some critical point (Sabuncuoglu and Kizilisik, 2003; Church and Uzsoy, 1992; Sabuncuoglu and Karabuk, 1999). Hozak and Hill (2009) discussed these differences. They identified some modelling choices that may help explain why the conclusions have been mixed, such as how instability is modelled, the assumptions used for demands and productions, and not considering human factors. Irrespective of these differing conclusions, it is clear that the performance of different rescheduling frequencies will vary, and hence studying this tradeoff is of interest from a practical standpoint.

Nevertheless, most of the studies described above have focused on case studies that use a fixed set of parameters for their experiments and do not consider

how the choice of parameters affect the results, e.g. different plant capacities, or different plant loads. These results can lead to recommendations which may not be valid for similar facilities that are used for the same purpose but have different processing capacities or processing loads than the original facility studied. Additionally, they typically focus on a single metric for comparing performance even though a facility may consider many different objectives, some of which may be conflicting, e.g. maximizing job throughput and minimizing job makespan. Therefore, recommendations made based on optimizing a particular metric may not hold for another facility whose main objective is different.

To address these gaps in the literature, this study compares the performance of various rescheduling policies subject to different plant loads and plant capacities. The efficacy of each rescheduling policy is measured in terms of job throughput, proportion of jobs on time, and job makespan, and performance comparisons are made based on each of these metrics. These experiments fill in a gap in the literature and the results demonstrate that the performance of a rescheduling policy does depend on the plant characteristics, and therefore care should be taken when selecting a rescheduling policy. In this study, we consider the rescheduling of a multipurpose industrial-scale plant subject to new job arrivals. Various fixed period rescheduling policies are compared on the aforementioned facility through extensive, long-term computational experiments. Because of the length of the simulations, a rolling horizon implementation is considered in this work. Through an empirical analysis on the results of the computational case studies conducted, we determine some general recommendations on selecting a rescheduling frequency depending on the plant parameters and main objective of the plant.

The organization of this study is as follows: in section 2, the problem and corresponding model are formally defined. The rolling horizon approach used for the computational experiments is also described in this section. In section 3 we present the design of the experiments that were carried out. We then go on to present and discuss the results of the experiments, leading to some recommendations on choosing rescheduling frequencies based on the properties of the production environment. Section 4 presents concluding remarks and further research considerations.

2 Problem Definition and Methodology

2.1 The Model Formulation

We now discuss and define the model used for our experiments. The problem under consideration is a variant of the job shop problem. A facility receives a set of jobs to process, I, and has access to a set of processes, J. Each job $i \in I$ is made up of a discrete collection of samples, and a sequence of processes, P^i, called a path, that must be performed (in a particular order) on the samples. Using this notation, we specify the k'th process in the path of job i as P^i_k for $k = 1, \ldots, |P^i|$. Each process $j \in J$ of the plant has a set of identical resources (e.g. machines) that perform a single operation on samples. Each of these resources has a fixed capacity, $\kappa(j)$, measured in terms of samples and a fixed processing time, $\Pi(j)$. Furthermore, we let $\rho(j)$ represent the number of resources available for each process j. We assume that resources may not be interrupted once they have started processing, and that

each resource may process samples from many jobs simultaneously, as long as there is available capacity.

We consider scheduling operations for H minutes from time S until time $S+H$. We refer to the interval $[S, S+H]$ as the time horizon for the problem. Based on the results discussed in Lagzi et al. (2017), we choose to discretize time and use the flexible discrete-time formulation presented in that study. The flexible discrete-time formulation can schedule samples to run on processes only on a set of timepoints that are determined *a priori*. Each process j has a set of timepoints $\varepsilon(j) = \{\varepsilon(j,t) : t = 1,\ldots,|\varepsilon(j)|\}$, where $\varepsilon(j,t) \geq \varepsilon(j,t-1)\ \forall j = 2,\ldots,|\varepsilon(j)|$, $\varepsilon(j,1) = S$, and $\varepsilon(j,|\varepsilon(j)|) = S+H$.

To decide where to include timepoints for each process, we chose to follow the non-uniform discrete 60 (NUD60) time discretization used in Lagzi et al. (2017) as it was shown to have a good tradeoff between solving time and solution quality. In the NUD60 time discretization, each process j has an associated timestep $\Delta(j) = \min\{60, \Pi(j)\}$ and the timepoints for process j are $\varepsilon(j) = \{S, S+\Delta(j),\ldots,S+(\lfloor H/\Delta(j)\rfloor)\Delta(j), S+H\}$. This choice of timepoints allows processes which have short processing times to have fine granularity, and also allows us to be flexible enough with the scheduling of long processes by including a timepoint each hour.

Using the rolling horizon implementation, which will be described in section 2.2, requires the use of a few more parameters. To denote both the samples that arrive at the facility *and* samples that have not finished processing and are carried over from a previous horizon, we use $A(i,k,t)$ to be the number of samples for job i that arrive at process P_k^i at time $\varepsilon(P_k^i,t)$. $Z(j,t)$ denotes the number of resources of process j that have been pre-allocated at time $\varepsilon(j,t)$, e.g. a machine is scheduled to be taken down for maintenance, or a machine is still running a process that started in a previous horizon. This is needed for ensuring consistency between horizons in the rolling horizon approach.

For every job i, every process P_k^i in the path of job i, and timepoint t for process P_k^i we have an integer decision variable $x(i,k,t)$ which denotes the number of samples of job i that begin processing on process P_k^i at time $\varepsilon(P_k^i,t)$. Similarly, we let $w(i,k,t)$ be the number of samples of job i that can be processed on process P_k^i and instead wait at time $\varepsilon(P_k^i,t)$. For every process j and every timepoint $t = 1,\ldots,|\varepsilon(j)|$ we have an integer decision variable $y(j,t)$ which denotes the number of resources of process j that begin processing samples at time $\varepsilon(j,t)$.

Two additional sets are needed to complete the model. The first set is $\Theta_1(i,k,t) = \{t' = 1,\ldots,\left|\varepsilon(P_{k-1}^i)\right| : \varepsilon(P_k^i,t-1) < \varepsilon(P_{k-1}^i,t') + \Pi(P_{k-1}^i) \leq \varepsilon(P_k^i,t)\}$. $\Theta_1(i,k,t)$ is the set of timepoints t' such that if process P_{k-1}^i starts at t', then it will end in the interval $(\varepsilon(P_k^i,t-1), \varepsilon(P_k^i,t)]$. The second set is $\Theta_2(j,t) = \{t' = 1,\ldots,|\varepsilon(j)| : \varepsilon(j,t) < \varepsilon(j,t') + \Pi(j) \leq \varepsilon(j,t) + \Pi(j)\}$. $\Theta_2(j,t)$ is the set of timepoints t' for process j such that if process j started processing at t', then the process would still be running at time $\varepsilon(j,t)$.

We now show the complete model formulation, given by problem $(P1)$. The model takes the form of a *Flexible Discrete-Time Formulation*; for a more in depth discussion of the model, we refer the reader to Lagzi et al. (2017).

The constraints given by (1), and (2) above are flow constraints. They ensure that at every point in time for each process in the path of each job, every sample that is available at this point either waits at the current process or is processed. Constraints (3) ensure that at the first timepoint, no samples are processed, however note that this is without loss of generality as we may define a second time-

$$\max_{x,w,y} \sum_{i\in I}\sum_{k=1}^{|P^i|}\sum_{t=1}^{|\varepsilon(P^i_k)|} \nu(i,k,t)x(i,k,t) - \sum_{j\in J}\sum_{t=1}^{|\varepsilon(j)|} c(j,t)y(j,t) \tag{P1}$$

$$\text{s.t. } x(i,k,t) + w(i,k,t) - w(i,k,t-1) - \sum_{t'\in\Theta_1(i,k,t)} x(i,k-1,t') = A(i,k,t) \quad \forall i\in I, k=2,\dots,|P^i|, \; t=2,\dots,|\varepsilon(P^i_k)| \tag{1}$$

$$x(i,1,t) + w(i,1,t) - w(i,1,t-1) = A(i,1,t) \quad \forall i\in I, t=2,\dots,|\varepsilon(P^i_1)| \tag{2}$$

$$x(i,k,1) = 0 \quad \forall i\in I, k=1,\dots,|P^i| \tag{3}$$

$$w(i,k,1) = A(i,k,1) \quad \forall i\in I, k=1,\dots,|P^i| \tag{4}$$

$$\sum_{i\in I,k'=1,\dots,|P^i|:P^i_{k'}=j} x(i,k',t) \le \kappa(j)y(j,t) \quad \forall j\in J, t=1,\dots,|\varepsilon(j)| \tag{5}$$

$$\sum_{t'\in\Theta_2(j,t)} y(j,t') \le \rho(j) - Z(j,t) \quad \forall j\in J, t=1,\dots,|\varepsilon(j)| \tag{6}$$

$$x(i,k,t), w(i,k,t) \ge 0 \quad \forall i\in I, k=1,\dots,|P^i|, \; t=1,\dots,|\varepsilon(P^i_k)| \tag{7, 8}$$

$$y(j,t) \ge 0 \quad \forall j\in J, t=1,\dots,|\varepsilon(j)| \tag{9}$$

$$x, w, y \text{ integral}, \tag{10}$$

point with the same time as the first timepoint. Constraints (4) set the number of samples that are waiting at the beginning of the horizon based on any samples that arrive at the beginning of the horizon, and any samples that were waiting at the end of the previous horizon. Constraints (5) ensure that enough machines are allocated to meet the proposed schedule. Constraints (6) enforce that we do not allocate more machines than we have available. Constraints (7) - (10) enforce that our variables are non-negative, and integral.

The objective function presented in ($P1$) is designed to maximize the number of samples that begin processing during the horizon. This was selected to incentivize the processing of samples even if it was not possible to finish processing them during the time horizon H to ensure that progress is made on finishing jobs. $\nu(i,k,t)$ in ($P1$) is used to represent the per sample value of processing the k'th process in job i's path at time $\varepsilon(P^i_k,t)$ and $c(j,t)$ is the per machine cost of starting process j at time $\varepsilon(j,t)$.

We will now discuss the choice of objective function used for our experiments. We compare the performance of the various policies based on job throughput, average job makespan, and proportion of jobs on time, over the scheduling horizon. These metrics are discussed in more detail in section 3.3. However, we do not optimize these metrics directly because of the inherent difficulties associated both with optimizing multiple and conflicting objectives, and the length of some job paths being longer than the lengths of the subhorizons used. For instance, suppose job throughput was our main objective and hence the objective function only provided incentive to processing the last process in each job's path. If we consider a job whose path length is longer than the length of the subhorizon, then the model would have no incentive for processing the job through earlier processes in its path and potentially no progress would be made on the job. Instead of optimizing these metrics directly, we used the objective function above and chose the values for the

objective function parameters to be the following:

$$\nu(i,k,t) = \left(1 + \frac{|\varepsilon(P_k^i)|-t}{|\varepsilon(P_k^i)|}\right)\left(\frac{k}{\sum_{m=1}^{|P^i|} m}\right), \quad \forall i \in I, k = 1,\ldots,|P^i|, t = 1,\ldots,|\varepsilon(P_k^i)|$$
$$c(j,t) = 0.001,\ \forall j \in J, t = 1,\ldots,|\varepsilon(j)|$$

These values were chosen to put a larger weight on processing samples that are further along in their path, and also to put a higher priority on processing samples earlier in the horizon. The rationale behind these decisions was that, by prioritizing samples that were closer to being finished we would push currently open jobs toward completion before starting new jobs. Furthermore, if a schedule could be shifted in time we would prefer it be executed as early as possible so that resources may be left unused to accommodate for possible job arrivals in the future. This was accomplished by assigning more weight to processing samples earlier in the horizon. We also assign a cost for using resources so that optimal schedules will allocate process resources if and only if the schedule assigns samples to be run on those resources. These decisions were done in an attempt to obtain attractive solutions (measured in terms of job throughput, job makespan, and proportion of jobs on time), in the rolling horizon framework. We note that these choices for the objective function parameters do not assign a cost to schedule disruptions between horizons. These costs are more difficult to quantify and are beyond the scope of this work.

2.2 Rolling Horizon Routine

A rolling horizon routine was used with the present model to simulate the operation of the facility over long periods of time (e.g. months). Decomposing the scheduling horizon into several smaller, contiguous time horizons is necessary as scheduling over the entire horizon at once would be computationally intractable. Furthermore, when scheduling operations in practice for a facility whose future arrivals are not known in advance, and over an indeterminate amount of time, the operator must implicitly use a rolling horizon strategy to schedule operations. The choice of how often to reschedule operations and how long the horizons should be will impact when new job arrivals are considered by the model, the computational cost of scheduling, and the quality of the actual implemented schedule. For these reasons, it is important to gain insight on the tradeoffs of using different policies so that an operator can make an informed decision when choosing how to reschedule operations in practice.

The routine can be thought of as partitioning the entire horizon that needs to be scheduled into smaller sub-horizons and then solving each of these sub-horizons sequentially. Let Γ denote the time horizon that needs to be scheduled, and denote the i'th sub-horizon by Γ_i for some partitioning of Γ into n sub-horizons, see figure 1. We start by setting $i = 1$ and the state of the facility as having an initial set of jobs arriving at time S and all machines empty. We then solve the model discussed in section 2.1 over sub-horizon Γ_i to generate a schedule $Sched_i$. Denote the set of jobs that arrive at the facility during sub-horizon Γ_i as A_i. The state of the facility is updated using $Sched_i$ and A_i to reflect the current jobs and the current machine usage at the beginning of sub-horizon Γ_{i+1}. After updating the state of

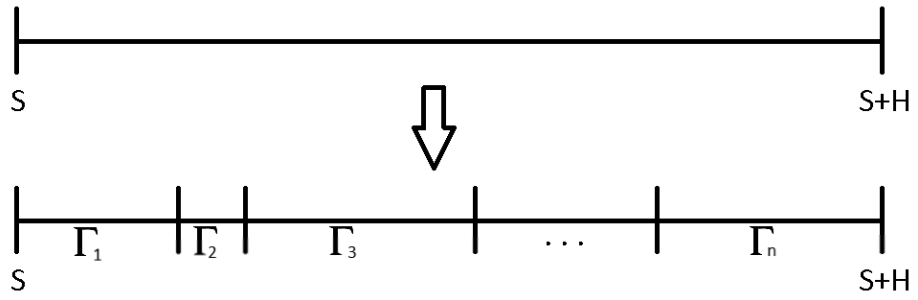

Fig. 1 Above is the scheduling horizon before partitioning. Below is a partition of scheduling sub-horizons that we may actually solve.

the facility, we then check if there are other sub-horizons to schedule over. If there are, then we increment i and repeat the process. If we have scheduled over all of the sub-horizons, then we stop and concatenating $(Sched_1, \ldots, Sched_n)$ gives us a feasible schedule for the entire horizon Γ.

After generating a schedule $Sched_i$ and considering arrivals A_i for sub-horizon Γ_i, several operations must be performed to update the state of the facility for the following sub-horizon. These operations are done to maintain consistency of the facility between horizons and we describe them next. Jobs which arrive during the sub-horizon Γ_i have their arrival times pushed back to the start of sub-horizon Γ_{i+1} since we may not consider these jobs during the current horizon but want the model to be able to schedule them at the beginning of the next sub-horizon. Samples which arrived at the facility prior to sub-horizon Γ_i are carried over to the following sub-horizon. If a sample was in progress of being processed at the end of Γ_i, then its arrival time is set to whenever the current process ends, and the process it arrives at is set to the next process in its path. If a sample was not being processed at the end of Γ_i, then its arrival time is set to the beginning of Γ_{i+1}. Any machines that will continue running beyond the end of Γ_i are set to continue running during Γ_{i+1} using the $Z(j,t)$ variables, until they finish running.

This procedure allows us to generate a schedule which spans Γ by solving smaller sub-problems when Γ is prohibitively large to schedule all at once. However, it is worth noting that even without considering new arrivals, this procedure is a heuristic with respect to scheduling over the entire horizon. Each sub-schedule $Sched_i$ may be optimal with respect to its corresponding sub-horizon, but the concatenated schedule $(Sched_1, \ldots, Sched_n)$ will not necessarily be optimal with respect to Γ due to the myopic nature of scheduling over each sub-horizon individually.

3 Computational Experiments

As discussed in section 2, the plant used for our experiments is based on a multipurpose industrial-scale analytical services facility. Due to confidentiality agree-

Table 1 Descriptions of rescheduling policies considered.

ID	Description
4P	Schedule the current day's operations at the beginning of every day and revise the remainder of the day's schedule 3 times during the day at equal intervals
2P	Schedule the current day's operations at the beginning of every day, then revise the second half of the schedule halfway through the day
S	Schedule the current day's operations at the beginning of every day
3D	Schedule the next three days of plant operation at the beginning of every third day
5D	Schedule the next five days of plant operation at the beginning of every fifth day

ments, we cannot disclose detailed data. The facility is rather large with nearly 200 distinct processes, each of which may have multiple identical machines. During a thirty day timespan, the facility received jobs comprising of over 150 unique paths, using approximately 100 unique processes. Over this timespan, they received several hundred jobs comprising of more than 20,000 samples. This large volume of jobs leads to large formulations when using long horizon lengths in the rolling horizon routine described in section 2.2. The capacities and processing times of the individual processes vary greatly. The largest capacity amongst all processes is over 1,300 times the size of the smallest capacity, similarly the processing times of the processes vary from a few minutes to several days. In the supplementary information (Online Resource 1), we present normalized values for capacity, processing time, and number of resources for each process. These qualities differentiate the plant studied from the simpler and smaller facilities found in other rescheduling studies (Shafaei and Brunn, 1999; Vieira et al., 2000b; Gupta and Maravelias, 2016). Although our experiments use a plant based on the facility of our industrial partner, the model defined in section 2.1 is a general flow shop formulation that may be used for other industrial applications that require a multipurpose plant. Accordingly, the discussions and conclusions presented in section 3.4 may be extrapolated to other applications using multipurpose plants.

3.1 Rescheduling Policies

For the experiments presented in this work, we decided to consider only fixed periodic rescheduling policies. This decision was made because it is the simplest policy that may be widely implemented in practice and we do not anticipate the use of more complex scheduling policies to be highly beneficial in our context. Since the disturbances we consider are not highly disruptive to the plant operation, i.e. new jobs arriving will not make the current schedule infeasible, we do not expect to gain much by triggering immediate reschedules which dissuades us from using an *event-driven* or *hybrid* policy. The decision to allow all operations to be rescheduled (the 'how-to' reschedule) was made to allow the facility to completely change schedules if needed to obtain the best performance possible. This is an optimistic way to reschedule as we assume that we are able to pivot from our current schedule to a new schedule without significantly affecting the plant's current operations. Table 1 gives a summary of the rescheduling policies considered in this work.

As shown in Table 1, policies "4P", and "2P", which can be thought of as "4 Parts", and "2 Parts", both schedule operations more than once per day at equal intervals. For example, if we consider scheduling operations for twenty four hours per day with the "4P" policy, then we will first generate a schedule for the next twenty four hours at the beginning of the day. We will follow this schedule for the first six hours, and then a new schedule will be generated for the following eighteen hours of the day. We will then generate a schedule for the remaining twelve hours after twelve hours and similarly for the last six hours after eighteen hours. These rescheduling policies emulate the operation of a facility that is concerned with the short term, day to day operations and wishes to reschedule frequently based on new job arrivals. The reason we do these reschedulings is because any new jobs that arrive during the first six hours may not be scheduled at the beginning of the day as they have not yet arrived at the facility, however by rescheduling after six hours, we may generate a new schedule for the remainder of the day that considers these new arrivals. The "2P" policy uses a similar strategy, but only schedules operations twice, once at the start of the day and again halfway through the day.

The "S" policy, referred to as "Single", generates a single schedule for the current day at the beginning of the day and will follow it, ignoring new job arrivals until a new schedule is generated at the beginning of the following day. This policy simulates a facility that is concerned with day to day operations but will not disrupt or modify the current schedule. The "3D", and "5D" policies which can be thought of as "3 Days", and "5 Days", schedule operations for the next three and five days, respectively. During this time, these policies behave like the "S" policy, ignoring new job arrivals until the next time operations are rescheduled. For this reason, in the worst case scenario when scheduling with the "5D" policy, the model may not be aware of a job which arrives at the facility until five days after it initially arrived. These policies simulate a facility that is more concerned with schedule stability, rather than reacting quickly based on the newest job arrivals.

These period lengths were chosen for testing so that we may test a wide range of policies and also policies which may be typically used in practice. For instance, the "5D" policy which corresponds to scheduling on a weekly basis, and the "S" policy which corresponds to scheduling once per day are both natural candidates for rescheduling. The "4P" policy was chosen to obtain information about how rescheduling very frequently would perform and the "2P" and "3D" policies were chosen to test policies which fell between the "S" policy and the other two extremes.

3.2 Design of Experiments

This section provides the details on how each experiment was performed. We let the initial state of the facility be empty with no machines running ($Z(j,t) = 0$, $\forall j, t$) and an initial influx of jobs all arriving at the beginning of the horizon (time S). We carry out the rolling horizon routine for 60 days, assuming that new jobs arrive uniformly at random throughout each day. We decided to carry out the experiments for 60 days so that the plant was able to reach a stable operation and then continue running to obtain results when measuring proportion of jobs on time with one month lead times.

Unless otherwise specified, the following parameters were used for the experiments presented in section 3.4. To assign job paths, actual job arrivals to the facility were recorded over the span of a high production month using actual historical data from the plant. Job paths were sampled according to the observed frequencies of each path during this timespan. In total, we observed 152 different paths during this time, with the path length ranging from 2 to 12 processes, an average path length of approximately 7.5 processes, and a median path length of 8 processes. The raw processing time for the paths sampled from varied from 13 minutes to 13,993 minutes, with an average of 2,926 minutes and a median of 2,226 minutes. There were some similarities in the observed paths as jobs which arrive may be roughly categorized into one of several categories depending on the analysis being done. These similarities appear as some reoccurring, short sub-sequences in the paths. The number of samples in each job was selected uniformly at random between ten and fifty, which was determined based on the observed historical plant data during this month.

We allowed the facility to be fully operational for the first eight hours each day. During this period, the plant operates with complete staffing and all available resources may be used to process samples. During the following sixteen hours each day we did not allow new operations to begin, but previously started operations were allowed to continue processing. The way this was handled with the "2P" policy was to first schedule the eight hour shift, and then revise the schedule for hours five through eight after hour four. Similarly, we revise the remainder of the schedule after hours two, four, and six for the "4P" policy. This was done to simulate actual plant operation. Workers may begin new operations while they are on site during the first eight hours each day, during the following sixteen hours only a few workers are present to supervise the operation of previously started processes.

After fixing the design parameters for each experiment, ten random instances with different job arrivals and starting jobs were generated and scheduled over to obtain an accurate representation of the results.

3.3 Performance Metrics

To compare the performance of the various policies considered in this study, we use job completion, average job makespan, and proportion of jobs on time. We begin by presenting the formulation of these metrics, before discussing them below.

Let $arr(i)$ be the arrival time of job i, $fin(i)$ be the time that the last sample of job i finishes processing, $jobs(t) = \{i : arr(i) \leq t\}$ be the set of jobs that arrive before or at time t, and $comp(i,t) = 1$ if job i has finished processing before or at time t, and 0 otherwise. Then job completion at time t, average job makespan at time t, and proportion of jobs on time at time t with lead times of d minutes were

defined as follows:

$$completion(t) = \frac{\sum_{i \in jobs(t)} comp(i,t)}{|jobs(t)|} \tag{1}$$

$$avg\_makespan(t) = \frac{\sum_{i \in jobs(t):comp(i,t)=1} fin(i) - arr(i)}{|\{i \in jobs(t) : comp(i,t) = 1\}|} \tag{2}$$

$$on\_time(d,t) = \frac{|\{i \in jobs(t) : comp(i,t) = 1, fin(i) - arr(i) \leq d\}|}{|jobs(t)|} \tag{3}$$

For comparing policies using these metrics, we use the value measured at the end of the 60 day horizon. Since we do not require that all jobs finish processing during the corresponding scheduling horizon, for measuring average job makespan we consider only those jobs that have finished processing at the end of the time horizon. Job completion at time t was defined as the proportion of jobs that finished processing all samples out of all the jobs that have arrived at the facility between the beginning of the first horizon and t. Makespan was taken to be the difference in minutes between when the final sample finished processing for a given job and when that job initially arrived at the plant, and was only considered for jobs that have finished processing. Proportion of jobs on time with respect to some lead time d was defined as the number of jobs that finished within the lead time d out of all jobs that had arrived before time t.

3.4 Results

The present study was performed on a 48 core machine running at 2.3GHz, with access to 256GB of RAM. The implementation was done using the Julia programming language (version 0.6.2), the CPLEX.jl (version 0.3.1) and JuMP.jl (Dunning et al., 2017) (version 0.18.0) packages, and CPLEX (version 12.6.0.0) for the solver. All of the CPLEX parameters were set to their default values, except for a limitation to use only 2 CPU cores, setting the relative MIP gap to be 0.5%, and imposing time limits on the instances. Time limits were set to be 15 minutes for each day being scheduled in the horizon, e.g. if the instance was using a "3D" policy as described above, then the time limit was set to be 45 minutes. This policy was set by the industrial partner based on their scheduling requirements. The time limits were met by some instances that solved three or five days consecutively, but for all of the problems that reached the time limit the relative optimality gap was at most 5% and was typically less than 1%.

Performance profiles are one of the main representations used below for showing the results of the experiments. In our context, we compare the performance of the different rescheduling policies over a large test set and use performance profiles to show the performance of each policy relative to the best performing policy with respect to some metric over each of the tests in the test set. Examples will be given in the following section as to how to read these figures. For more detailed information about performance profiles we refer the reader to Dolan and More (2002).

3.4.1 Results with Moderate Facility Load

The experiments described next were considered to obtain results that were representative of a facility that received enough new jobs to be running constantly, but not enough that the queue of waiting jobs grows indefinitely. We considered the parameters of the problem to be the following: the facility starts with 5,000 samples (approximately 170 jobs) arriving at the beginning of the first horizon, jobs comprising 500 samples arrive uniformly at random throughout each day, and the facility is operated for 8 hours each day as described in section 3.2. These numbers of samples were determined experimentally. The names of the plots in the figures below have the following format: (number of samples added per day) - (rescheduling policy), for instance "500 - 4P" corresponds to the performance of rescheduling four times per day for instances where 500 samples are added each day over the 60 day horizon.

Figure 2 shows a performance profile comparing average job completion percentage over the 60 day horizon between the different rescheduling policies. On the x-axis we have the performance of each policy measured as a factor of the performance of the best performing policy on each test. On the y-axis we have the proportion of tests that a policy achieves within a given factor of the optimal performing policy for each test. This Figure shows that the "4P" policy clearly dominates the other policies. The point $(1, 1)$ in the curve of the "4P" policy in the figure indicates that it was the best performer in all of the tests. Similarly, the point $(1.02, 0.5)$ in the curve of the "3D" policy indicates that in 50% of the instances, the "3D" policy performed within 2% of the best performing policy for that instance. We also observe that rescheduling more frequently continues to improve job completion performance for each of the policies tested. Furthermore, in the worst case, rescheduling only once every five days is approximately 5.6% worse than rescheduling four times per day. This is indicated by the point $(\approx 1.056, 1)$ in the curve of the "5D" policy and by noting that for all smaller x values, the curve lies below 1. Therefore, depending on how much importance is placed on schedule stability, it may be worth it to delay the rescheduling of operations to only once every few days. However, if maximum performance is desired, rescheduling more frequently is better.

Table 2 summarizes the results obtained when considering the other metrics. The performance profiles for the other metrics are similar to Figure 2, so instead of presenting each performance profile, we present the average performance factor (APF) values as a more informative measure of relative differences. The APF of a policy is defined to be the mean performance (measured as a factor of the best performing policy) a policy achieves over the ten random instances. The APF was used to compare the relative performance of each policy and estimate the differences in APF values between pairs of policies when decreasing rescheduling frequency from most to least frequent, i.e. when moving from the "4P" policy toward the "5D" policy.

From Table 2 we draw the same conclusion as above: rescheduling more frequently improves performance. The degree of performance gain is dependent on the chosen metric, and can vary from negligible improvements to an order of magnitude difference between the "4P" and "5D" policies in some cases. Furthermore, we observe that the relative benefit obtained by moving from less frequent rescheduling policies to more frequent rescheduling policies diminishes as one begins to

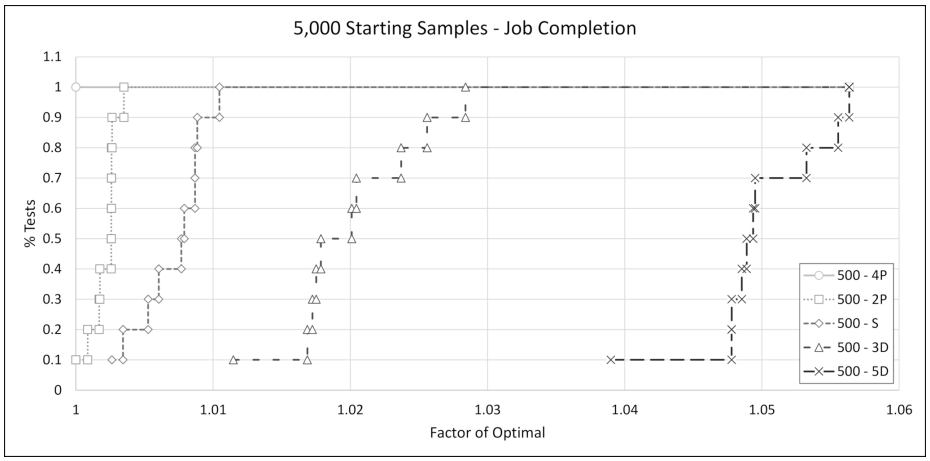

Fig. 2 Performance profile comparing job completion amongst rescheduling policies for instances starting with 5,000 samples and 500 samples arriving each day.

reschedule frequently. This is demonstrated by the generally increasing APF differences as we move from the "4P" policy toward the "5D" policy. We note that the performance differences between policies also decrease in general with respect to proportion of jobs on time as lead times are increased. Therefore, if longer job turn around times are acceptable, then the smaller performance loss may be a worthwhile tradeoff for the added schedule stability achieved by rescheduling only once every few days. For more detailed results, we include the performance profiles which generate Table 2 as supplementary information (Online Resource 1).

Table 3 shows the average number of variables and constraints that were used when solving each model for the industrial-scale facility. To clarify, these numbers correspond to the average size of each model that was solved for each of the policies tested. Note that each policy may solve a different number of models to generate a schedule over the same scheduling horizon. For instance, the "4P" policy generates a schedule four times per day, and hence solves 240 models over a 60 day horizon, whereas the "5D" policy will solve 12 models over the same 60 day horizon. We also include average problem solving times. As expected, the average time required to solve each model increases as the lengths of the horizons scheduled in each model increases. However, we also note that there is a tradeoff between solving smaller models and the total number of models that are solved over the 60 day horizon by observing the total solving times for the different policies. We observe that scheduling once per day using the "S" policy requires the least amount of time, scheduling multiple times per day increases the total solving time moderately, and scheduling over long horizons for the "3D" and "5D" policies increases the total solving time considerably. This drastic increase in cost when scheduling over multiple days can be attributed to the much larger problems that are being solved. The problems are larger with respect to the number of timepoints considered because of the added length of the horizon. Furthermore, by stockpiling new job arrivals for several days, accounting for all of them during the next horizon increases the number of jobs considered, which further increases the problem size. Depending on the application and how quickly results are needed,

Table 2 Average performance factor (APF) values of each metric for instances with 5,000 starting samples and 500 arriving samples per day.

ID	Job Completion APF (Difference)
4P	1.0 (-)
2P	1.0021 (0.0021)
S	1.007 (0.0049)
3D	1.0199 (0.0129)
5D	1.0496 (0.0297)
ID	Job Makespan APF (Difference)
4P	1.0 (-)
2P	1.0705 (0.0705)
S	1.2422 (0.1717)
3D	1.66 (0.4178)
5D	2.0822 (0.4222)
ID	Proportion of Jobs On Time APF, 1 Day Lead Times (Difference)
4P	1.0 (-)
2P	1.2256 (0.2256)
S	2.5536 (1.328)
3D	8.1484 (5.5948)
5D	11.059 (2.9106)
ID	Proportion of Jobs On Time APF, 1 Week Lead Times (Difference)
4P	1.0 (-)
2P	1.0031 (0.0031)
S	1.0113 (0.0082)
3D	1.0307 (0.0194)
5D	1.1075 (0.0768)
ID	Proportion of Jobs On Time APF, 1 Month Lead Times (Difference)
4P	1.0002 (-)
2P	1.0021 (0.0019)
S	1.007 (0.0049)
3D	1.0204 (0.0134)
5D	1.05 (0.0296)

the differences between these problem solving times may be a consideration when selecting a rescheduling policy.

Overall, the results obtained by using nominal values for load parameters agree with the most common conclusion in the literature that more frequent rescheduling is better and that the relative benefit may diminish as rescheduling frequency increases. We noted that the degree of this benefit varies depending on the performance measure chosen for comparison.

3.4.2 Effects of Different Plant Loads

To observe the effect of plant load on the results, we generated new instances that varied both the starting number of samples and the number of arriving samples per day from their nominal values. In particular, we kept the number of starting samples constant but varied the number of arriving samples per day in some experiments. In other experiments, we varied the number of starting samples but

Table 3 Problem size statistics for instances with a moderate job load.

	4P	2P	S	3D	5D
Average Number of Variables Per Model	56,212	59,297	65,251	234,488	522,046
Average Number of Constraints Per Model	44,525	46,092	49,123	165,283	340,737
Average Solving Time Per Model (s)	0.64	0.96	1.82	16.8	86.5
Total Solving Time for 60 Day Horizon (s)	153	115	109	336	1,038

kept the number of samples arriving per day constant. For each of these pairs of number of starting samples and number of arriving samples per day, we generated and solved ten random instances with 60 day horizons, which were used to produce the results shown in Table 4 and Table 5.

Table 4 shows the APF values for each policy, both in terms of job completion and average makespan, when the number of starting samples was varied, and the number of samples added per day was fixed to 500. We tested with 2,500, 5,000, and 10,000 starting samples as shown in Table 4 (column 2). By observing the job completion and average makespan APF differences for the three different starting loads, we note that, as the number of samples available at the beginning of the experiment increases, the performance difference between policies decreases. These results also tend to suggest that more frequent rescheduling improves performance.

Table 5 shows the APF values for each policy when the number of starting samples is fixed to 5,000 and the number of samples arriving daily is varied. Similar to above, we tested with 250, 500, and 1,000 samples arriving per day and calculated the APF differences between policies as rescheduling frequency is decreased, for these different loads. Contrary to the observations for Table 4, Table 5 shows that increasing the number of daily sample arrivals actually increases the performance differences observed between policies. However, we again note that more frequent rescheduling appears to perform better.

Note that by increasing the amount of samples arriving at the beginning of the horizon, the plant begins the experiment with a larger queue of jobs. As a consequence of this, all policies have a *backlog* of jobs to schedule and hence there is less advantage to receiving new job arrivals more promptly, since there are already many jobs to schedule. It is sensible then that the performance differences between policies decreases as the number of starting samples increases. Similarly, when more samples are received at the facility on a daily basis, by rescheduling frequently the model is able to consider these many new samples earlier than policies which will not consider these samples until at least the next day. Therefore, it is reasonable that performance differences increase as the number of daily sample arrivals increases. We also note that these results suggest that plant capacity may play a role, as we observe that the differences between policies is variable with respect to the load on the facility and the presence of a backlog of jobs.

With regards to how the proportion of jobs on time reacts to different loads on the facility, we largely observe the same trends as when moderate loads were used above. That is, as the number of starting samples was increased, the performance differences between policies decreased. However, as the number of daily job arrivals was increased, the performance differences between policies increased as well.

Table 4 APF differences comparison for instances starting with various numbers of samples and 500 new samples per day.

ID	Starting Samples	Job Completion APF	Job Completion APF Differences	Average Makespan APF	Average Makespan APF Differences
4P	2,500	1.0	-	1.0	-
2P	2,500	1.002	0.002	1.0658	0.0658
S	2,500	1.0083	0.0063	1.2333	0.1675
3D	2,500	1.0213	0.013	1.6709	0.4376
5D	2,500	1.0537	0.0324	2.1863	0.5154
4P	5,000	1.0	-	1.0	-
2P	5,000	1.0021	0.0021	1.0705	0.0705
S	5,000	1.007	0.0049	1.2422	0.1717
3D	5,000	1.0199	0.0129	1.66	0.4178
5D	5,000	1.0496	0.0297	2.0822	0.4222
4P	10,000	1.0	-	1.0	-
2P	10,000	1.0019	0.0019	1.0559	0.0559
S	10,000	1.0068	0.0049	1.1707	0.1148
3D	10,000	1.0179	0.0111	1.415	0.2443
5D	10,000	1.0418	0.0239	1.7077	0.2927

Table 5 APF differences comparison for instances starting with 5,000 samples and various daily arrival loads.

ID	Samples Added Per Day	Job Completion APF	Job Completion APF Differences	Average Makespan APF	Average Makespan APF Differences
4P	250	1.0001	-	1.0	-
2P	250	1.0024	0.0023	1.0657	0.0657
S	250	1.0083	0.0059	1.2145	0.1488
3D	250	1.0221	0.0138	1.5785	0.364
5D	250	1.0477	0.0256	1.9481	0.3696
4P	500	1.0	-	1.0	-
2P	500	1.0021	0.0021	1.0705	0.0705
S	500	1.007	0.0049	1.2422	0.1717
3D	500	1.0199	0.0129	1.66	0.4178
5D	500	1.0496	0.0297	2.0822	0.4222
4P	1,000	1.0002	-	1.0	-
2P	1,000	1.0029	0.0027	1.0658	0.0658
S	1,000	1.0099	0.007	1.2333	0.1675
3D	1,000	1.0263	0.0164	1.6709	0.4376
5D	1,000	1.0556	0.0293	2.1863	0.5154

These observations agree with the results discussed above concerning makespan and job completion, and follow from the same discussion. We omit the inclusion of these Figures here for brevity, but include them in the supplementary information (Online Resource 1).

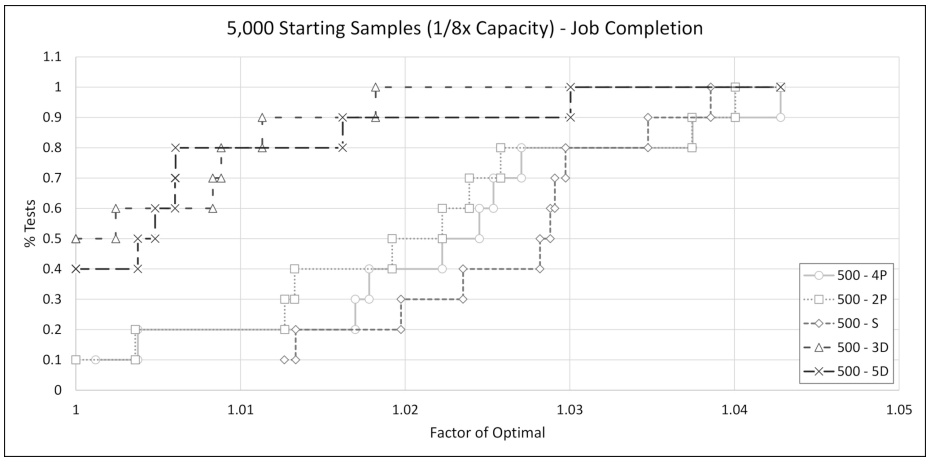

Fig. 3 Performance profile comparing job completion amongst rescheduling policies for instances with one eighth the original capacity.

3.4.3 Effects of Varying Plant Capacity

Based on the observations obtained from the previous scenario, we investigate the role plant capacity plays on the rescheduling frequency. In the following experiments, the capacity of all the processes in the plant was tested at 8x, 4x, 2x, $\frac{1}{2}$x, $\frac{1}{4}$x, and $\frac{1}{8}$x their original capacity. These values were chosen to obtain data for a wide range of overall capacities with the $\frac{1}{8}$x representing a facility that is severely starved for resources and the 8x representing a facility which has more than enough resources to process the given demand. We performed ten random instances for each plant capacity, and instances were run with moderate job loads. When the capacity was between one half the original capacity and eight times the original capacity, the conclusions with respect to job throughput remained the same as when nominal parameter values were used as in section 3.4.1 and so we include these results as supplementary information (Online Resource 1). When the plant capacity was lowered to one eighth of the original capacity, we observed that longer rescheduling policies outperformed frequent rescheduling, as shown in Figure 3. With greatly reduced plant capacity, there is more contention for resources and the longer horizons used by the "3D" and "5D" policies allows for better utilization of processes as discussed previously. This follows the same trend observed when increasing the load on the facility, although the results are more pronounced with the drastic reduction in capacity across all processes. Therefore, if there is high resource contention in the facility, rescheduling less often may actually improve job completion performance.

The makespan performance results remained largely the same as to those presented with moderate facility load in section 3.4.1. However, when the capacity was restricted to one quarter of the original capacity, the performance differences of the rescheduling policies decreases greatly. This is shown by the APF differences between the different policies compared to when the plant has full capacity, as shown in Table 6.

Table 6 Average makespan APF differences comparison between one quarter capacity and full capacity instances.

ID	Average Makespan APF with Quarter Capacity	Average Makespan APF Differences with Quarter Capacity	Average Makespan APF with Full Capacity	Average Makespan APF Differences with Full Capacity
4P	1.0258	-	1.0	-
2P	1.0082	-0.0176	1.0705	0.0705
S	1.0309	0.0227	1.2422	0.1717
3D	1.0866	0.0557	1.66	0.4178
5D	1.1974	0.1108	2.0822	0.4222

Fig. 4 Proportion of jobs on time for one month lead times and instances with one eighth the original capacity.

When measuring the proportion of jobs on time we observe that, for lead times of at least one week, and capacity one quarter or less than the original capacity, the policies with longer horizons perform best. Figure 4 shows the proportion of jobs on time for each of the different policies with one eighth capacity and one month lead times. The results with one week lead times are similar to those with one month lead times, however in these cases the "5D" policy falls short of the other policies. This observation is likely caused by the increased latency between job arrival time and the first time a job may be scheduled, that the "5D" policy is subject to. The performance profiles for these other cases are included as supplementary information (Online Resource 1). The results when capacity was increased beyond the original capacity did not differ substantially from those obtained using moderate facility load, presented in section 3.4.1.

These experiments demonstrate that, for plants which are starved for resources, the longer horizons used when scheduling less frequently can allow for better resource utilization. This better utilization can then be translated into better job completion performance and more jobs considered on time if long lead times are acceptable. These results differ from the general consensus in the literature that more frequent rescheduling is beneficial and show that plant and problem specific

parameters, particularly with respect to facility load, may play an important role when choosing a suitable rescheduling policy.

3.4.4 Other Factors Considered

Beyond the results that were shown above, we also performed additional experiments simulating a plant that was run continuously for 24 hours each day as opposed to the 8 hours per day mode used for the experiments described above. For these tests we used the "4P", "2P", and "S" policies described in Table 1 but instead of using an 8 hour day with a 16 hour gap between horizons we used a 24 hour day with no gap. We also used a "2D" policy that is similar to the "3D", and "5D" policies which schedules two days at a time. This was done because solving three days and five days using a 24 hour horizon was computationally taxing and so we lessened the number of days down to two.

Additionally, when creating random jobs while generating test instances, the set of job paths that was sampled from was varied in some experiments. We tested sampling from paths that could be requested at the facility but did not necessarily arrive during the month of observation, and also paths that were completely random and may not be realistic to arrive at the facility.

In both of these cases, the conclusions drawn from these experiments did not change from those discussed when considering the experiments with nominal parameter values. The results were very similar for these experiments, with the most notable difference being that the performance differences between policies was less than when the plant was run with actual paths that may arrive at the facility. A possible explanation is that the increased diversity in paths spreads the process demand over more processes, and hence leads to less resource contention. The results of these experiments suggest that our previous results do not rely on our choice of plant operating mode nor choice of job paths to sample from.

4 Conclusions

This work presents a comparison between several different periodic rescheduling policies varying from generating a schedule four times per day to generating a schedule only once every five days. Experiments were conducted simulating a real analytical services facility to compare the performance between the policies using a rolling horizon routine and the non-uniform discrete-time model presented in Lagzi et al. (2017).

Based on the results obtained through the computational experiments, this work shows that choosing a suitable rescheduling policy can depend greatly on the environment of the facility that is being modelled. By varying the capacity of the processes in the experiments we observed that in some instances when there is great resource contention, less frequent rescheduling policies may outperform more frequent rescheduling policies both in job completion and proportion of jobs on time. In particular, in some experiments, we were able to observe nearly a complete reversal of the results obtained when nominal parameter values were used.

When nominal parameter values were used, we observed that more frequent rescheduling can have a significant positive impact on improving the proportion of

jobs on time and makespan of a production plant. Additionally, the job completion also benefited from the more frequent rescheduling.

In environments where capacity is not much of a concern or where very short lead times are required, more frequent rescheduling policies seem to perform best as well. However, if long lead times are acceptable or capacity is the main limiting constraint, then scheduling less frequently with longer horizons can improve performance.

Future work will consider implementing further sources of uncertainty such as job retesting if a process malfunctions or the possibility of machine breakdowns. With these augmentations, a more robust scheduling policy may be desirable to reschedule immediately when a more serious disruptive event occurs and to forgo rescheduling when it is not necessary. Additionally, more research into quantifying the costs of rescheduling such as schedule disruption and latency introduced by deviating from the originally intended schedule could be beneficial. In this study we assumed that these costs were negligible but a study focusing on quantifying and measuring these costs could help us to better understand the other side effects of rescheduling. Furthermore, it would be worth investigating a dynamic policy whose rescheduling period is not fixed and instead depends on the status of the facility. It seems intuitive that a dynamic policy may be able to surpass the performance of fixed policies; however, it is not clear how much this may improve performance or what other tradeoffs and considerations may be inherent with such a policy.

Acknowledgements The financial support provided by the Natural Sciences and Engineering Research Council of Canada (NSERC), Ontario Centers for Excellence (OCE), and the industrial partner in the analytical services sector is gratefully acknowledged.

References

A. Baykasoğlu and F. S. Karaslan. Solving comprehensive dynamic job shop scheduling problem by using a grasp-based approach. *International Journal of Production Research*, 55(11):3308–3325, 2017. doi: 10.1080/00207543.2017.1306134.

L. K. Church and R. Uzsoy. Analysis of periodic and event-driven rescheduling policies in dynamic shops. *Int. J. Computer Integrated Manufacturing*, 5(3):153–163, 1992. doi: 10.1080/09511929208944524.

Elizabeth D. Dolan and Jorge. J More. Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91(2):201–213, 2002. doi: 10.1007/s101070100263.

Iain Dunning, Joey Huchette, and Miles Lubin. Jump: A modeling language for mathematical optimization. *SIAM Review*, 59(2):295–320, 2017. doi: 10.1137/15M1020575.

D. Gupta and C. Maravelias. On deterministic online scheduling: Major considerations, paradoxes and remedies. *Computers and Chemical Engineering*, 94(2):312–330, 2016. doi: 10.1016/j.compchemeng.2016.08.006.

K. Hozak and J. A. Hill. Issues and opportunities regarding replanning and rescheduling frequencies. *International Journal of Production Research*, 47(18):4955–4970, 2009. doi: 10.1080/00207540802047106.

M. H. Kim and Y.-D. Kim. Simulation-based real-time scheduling in a flexible manufacturing system. *Journal of Manufacturing Systems*, 13(2):85–93, 1994. doi: 10.1016/0278-6125(94)90024-8.

R. Koller, L. Ricardez-Sandoval, and L. Biegler. Stochasic back-off algorithm for simultaneous design, control, and scheduling of multiproduct systems under uncertainty. *AIChE J*, 64(7):2379–2389, 2018. doi: 10.1002/aic.16092.

S. Lagzi, D. Yeon Lee, R. Fukasawa, and L. Ricardez-Sandoval. A computational study of continuous and discrete time formulations for a class of short-term scheduling problems for multipurpose plants. *Ind. Eng. Chem. Res.*, 56(31): 8940–8953, 2017. doi: 10.1021/acs.iecr.7b01718.

A. P. Muhlemann, A. G. Lockett, and C.-K. Farn. Job shop scheduling heuristics and frequency of scheduling. *International Journal of Production Research*, 29 (2):227–241, 1982. doi: 10.1080/00207548208947763.

D. Ouelhadj and S. Petrovic. A survey of dynamic scheduling in manufacturing systems. *J Sched*, 12:417–431, 2008. doi: DOI10.1007/s10951-008-0090-8.

M. E. Pfund and J. W. Fowler. Extending the boundaries between scheduling and dispatching: hedging and rescheduling techniques. *International Journal of Production Research*, 55(11):3294–3307, 2017. doi: 10.1080/00207543.2017.1306133.

I. Sabuncuoglu and S. Karabuk. Rescheduling frequency in an fms with uncertain processing times and unreliable machines. *Journal of Manufacturing Systems*, 18(4):268–283, 1999. doi: 10.1016/S0278-6125(00)86630-3.

I. Sabuncuoglu and O. B. Kizilisik. Reactive scheduling in a dynamic and stochastic fms environment. *International Journal of Production Research*, 41(17):4211–4231, 2003. doi: 10.1080/0020754031000149202.

R. Shafaei and P. Brunn. Workshop scheduling using practical (inaccurate) data part 1: The performance of heuristic scheduling rules in a dynamic job shop environment using a rolling time horizon approach. *International Journal of Production Research*, 37(17):3913–3925, 1999. doi: 10.1080/002075499189682.

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. *International Journal of Production Research*, 38(8):1899–1915, 2000a. doi: 10.1080/002075400188654.

G. E. Vieira, J. W. Herrmann, and E. Lin. Predicting the performance of rescheduling strategies for parallel machine systems. *Journal of Manufacturing Systems*, 19(4):256–266, 2000b. doi: 10.1016/S0278-6125(01)80005-4.

G. E. Vieira, J. W. Herrmann, and E. Lin. Rescheduling manufacturing systems: A framework of strategies, policies, and methods. *Journal of Scheduling*, 6(1): 39–62, 2003. doi: 10.1023/A:1022235519958.

C. A. Yano and R. C. Carlson. Interaction between frequency of rescheduling and the role of safety stock in material requirements for planning systems. *International Journal of Production Research, Vol. 25, No. 2, 221-232*, 25(2):221–232, 1987. doi: 10.1080/00207548708919835.