Production Scheduling Under Demand Uncertainty in the Presence of Feedback: Model Comparisons, Insights, and Paradoxes

Venkatachalam Avadiappan[1], Dhruv Gupta[1], Christos T. Maravelias*[2,3]

[1]Department of Chemical and Biological Engineering, University of Wisconsin-Madison, WI, USA

[2]Department of Chemical and Biological Engineering, Princeton University, NJ, USA

[3]Andlinger Center for Energy and Environment, Princeton University, NJ, USA

Abstract

We investigate the importance of accounting for uncertainty a priori in production scheduling in the presence of feedback. First, we examine different optimization models (deterministic, robust, and stochastic programming), used to generate the open-loop schedules and describe the modeling of uncertainty in each case. Second, we present a formal procedure for carrying out closed-loop simulations in order to study and compare the closed-loop performance across the models as attributes such as the demand uncertainty observation horizon, order size max-mean relative difference, and load on the process network are varied. Finally, we analyze the results of the simulations to draw insights on how the above attributes affect the closed-loop performance of the different models across networks and expound on the paradoxes observed.

Keywords: Online scheduling, Real-time optimization, Mixed-integer programming

*Corresponding author.

1 Introduction

Scheduling plays an important role in chemical production facilities, from batch production of fine chemicals to the continuous production of bulk chemicals (Harjunkoski et al., 2014). When a schedule is computed, disruptions or the arrival of new information can render the incumbent schedule sub-optimal or even infeasible, thereby necessitating online (re)scheduling (Subramanian et al. (2012); Maravelias (2021)). Rescheduling as an ongoing process in which decisions are made and revised in real-time during the execution of the schedule is termed as online scheduling.

In a chemical production environment, there can be uncertainties associated with the processing times of batches, batch yields, unit operating status, and demand for products. An important consideration when computing schedules is to account for these uncertainties in the scheduling model. To account for the uncertainties in a pre-emptive manner as the scheduling problem is solved offline, researchers have proposed various models based on robust optimization (Lin et al., 2004) and stochastic programming (Bonfill et al., 2005). However, whether the presence of feedback (through the online scheduling procedure) sufficient to account for the uncertainties, remains an open question. In the online scheduling procedure that employs a deterministic model, the uncertainty is not explicitly modeled and it relies on recourse based solely on feedback (Gupta and Maravelias, 2020). With a robust optimization model, the focus is on maintaining feasibility and typically, bounds on the uncertain parameters are used, resulting in optimization decisions in the open-loop solution, that are conservative irrespective of the actual uncertainty realization. While, using the stochastic programming model, a set of scenarios is determined based on the probability distributions of the uncertain parameters and a recourse strategy is computed through maximizing or minimizing an expectation function.

In this work, we investigate the significance of accounting for demand uncertainty a priori in the presence of feedback (i.e., using robust optimization, stochastic programming) compared to online scheduling based solely on feedback (i.e., using deterministic optimization).

The demand is expressed in the form of orders with the order sizes being drawn from a probability distribution. We carry out closed-loop simulations for different values of attributes such as the demand uncertainty observation horizon, order size max-mean relative difference, and the load on the process network. We explain why choosing the best model to mitigate demand uncertainty for any given process network is non-trivial and expound on the closed-loop performance of the different models, draw useful insights, and clarify observed paradoxes.

The paper is structured as follows. In Section 2, we provide the necessary background. In Section 3, we describe the simulations that are carried out to evaluate the closed-loop performance of the models as attributes are varied. In Section 4, we present the results of the closed-loop simulations. In Section 5, we draw insights, and in Section 6, we analyze the paradoxes observed in the closed-loop performance. Throughout the text, we use lower case italic letters for indices, uppercase boldface letters for sets, uppercase italic letters for variables, and greek letters for parameters.

2 Background

First, we present the general chemical production scheduling problem statement, problem representation, and model classification. Second, we describe what online scheduling is and review the literature. Third, we present the state-space resource task network (RTN) scheduling model used in this work. Finally, we review research that deals with scheduling under demand uncertainty.

2.1 Chemical production scheduling

2.1.1 Problem statement

The general chemical production scheduling problem can be stated as follows. Given:

(i) production facility data (e.g., unit capacities);

(ii) production recipes (e.g., processing times);

(iii) production costs (e.g., inventory holding costs and batch processing costs);

(iv) material availability (e.g., material delivery amounts and dates);

(v) resource availability (e.g., utility levels); and

(vi) production targets or orders with due-times;

scheduling seeks to determine:

(i) selection and sizing of batches to be processed;

(ii) assignment of these batches to units; and

(iii) sequencing and timing of these batches;

so as to meet the production targets at minimum cost, maximize profit by allowing excess sales, or optimize any other suitable objective. There can be problem features such as time-varying utility costs, sequence dependent changeovers, storage constraints, etc., which can be appropriately handled through existing scheduling models (Méndez et al. (2006); Harjunkoski et al. (2014)).

2.1.2 Problem representation

We represent the problem using a resource task network (RTN) (Pantelides, 1994), which primarily comprises of resources $r \in \mathbf{R}$ and tasks $i \in \mathbf{I}$. The set of resources is composed of continuous, $\mathbf{R}^C$ (includes material, energy supplies), and discrete, $\mathbf{R}^D$ (includes labor, equipment units) resources with: $\mathbf{R} = \mathbf{R}^C \cup \mathbf{R}^D$ and a task is an operation that consumes and/or generates resources during its execution. Given a scheduling problem, the RTN representation is constructed by identifying the relevant resources and tasks. An example RTN is shown in Figure 1, where tasks I1 and I2 are carried out in equipment U1. The tasks engage the equipment resource at the start of execution and disengage it at the end. Tasks I1 and I2 consume the material resource M0 and produce material resource M1 and M2, respectively. Here, $\mathbf{I} = \{\text{I1}, \text{I2}\}$, $\mathbf{R}^C = \{\text{M0}, \text{M1}, \text{M2}\}$, and $\mathbf{R}^D = \{\text{U1}\}$.

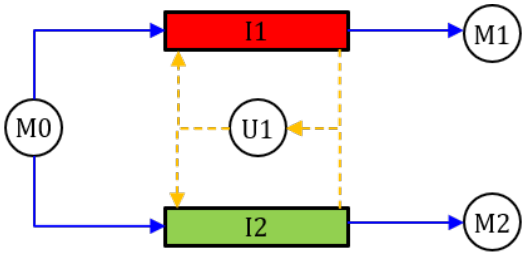


Figure 1: Example RTN representation. Blue arrows represent continuous resource interactions, while orange dashed arrows indicate discrete resource interactions. Tasks are denoted by rectangles, and resources by circles.

2.1.3 Model classification

Scheduling models can be classified on the basis of (i) optimization decisions, (ii) modeling elements, and (iii) modeling of time (Maravelias, 2012). With respect to the modeling of time, they can be classified, first, as sequence-based or time grid-based models and, second, as continuous or discrete time models. Discrete time grid-based models lend themselves to online scheduling approaches because the fixed model grid facilitates the synchronization with the re-optimization frequency. The discussion in this paper will be based on the most commonly used models, namely, models based on a single uniform grid, where the time horizon is discretized into periods of uniform length (denoted by δ) and tasks start and end at the time (grid) points. To ensure feasibility, time-related data are rounded, for example, processing times are adjusted as $\tau_i = \lceil \tau_i^A / \delta \rceil$, where τ_i^A is the actual processing time of task i. Although there could be discretization errors when using discrete time models, they have several advantages over continuous time models. For example, in discrete time models, accounting for inventory and utility costs as well as time varying prices and resource availabilities, does not introduce non-linearities nor does it require the introduction of new variables and constraints (Velez and Maravelias, 2014). Moreover, Sundaramoorthy and Maravelias (2011) showed that discrete time models are, in general, at least as effective as continuous time models and are better suited for large-scale instances with several additional processing features. For the purpose of all simulations in this work, we use $\delta = 1$ h. Although we use this specific discrete time model, the analysis presented is applicable to all discrete

time models and can be extended to continuous time models.

2.2 Online scheduling

Rescheduling as an ongoing process, wherein optimization problems are solved periodically in order to account for the feedback and new information is termed as online scheduling (Gupta et al. (2016); Gupta and Maravelias (2016)). Recently, Rawlings et al. (2019) proposed a novel approach to incorporate knowledge of the automation system in a chemical plant into the online scheduling problem. Building upon that, Avadiappan and Maravelias (2021) presented methods to incorporate real-time data in online scheduling through state estimation using a deterministic optimization model.

In each online scheduling iteration, an open-loop optimization problem is solved, to determine the current and future decisions. Every open-loop problem is associated with a prediction horizon denoted by H. When the current decisions are implemented, the prediction horizon is moved forward by a fixed number of periods denoted by Δ (i.e., the re-optimization time-step), and the next set of decisions are computed (i.e., the next open-loop problem is solved). The re-optimization time step should not be confused with δ, which denotes the time-grid spacing in discrete time models. The actual implemented schedule, determined based on the solution to a series of open-loop problems, is called the closed-loop schedule.

Researchers in the past have proposed various rescheduling strategies in response to disruptions or unexpected events (Vieira et al. (2003); Ave et al. (2019)). Méndez and Cerdá (2003) proposed a novel mixed integer linear programming (MILP) formulation for rescheduling, which can update schedules in response to unforeseen events. Janak et al. (2006) proposed partial rescheduling by identifying and fixing tasks not directly affected by an observed disturbance. Bonfill et al. (2008) demonstrated a proactive-reactive approach with a stochastic model used in a reactive framework for batch scheduling. Novas and Henning (2010) formulated a constraint programming model to compute rescheduling decisions, based on the occurrence of disruptive events. Kopanos and Pistikopoulos

(2014) proposed a rescheduling approach using the rolling horizon framework that relies on multiparametric programming techniques. Silvente et al. (2015) used a rolling horizon framework for the simultaneous optimization of energy supply and demand planning in microgrids. Touretzky et al. (2017) proposed a methodology to detect faults in the process that act as trigger for rescheduling. Pattison et al. (2017) proposed a feedback rescheduling mechanism consisting of periodic schedule updates that account for demand forecasts and event driven updates to account for process and market disturbances. Mathur et al. (2020) presented an online scheduling scheme, implemented in a rolling horizon fashion to account for uncertainties in electricity prices and water inflows in the operation of cascaded hydropower systems.

Researchers have also developed closed-loop approaches that integrate scheduling and control decisions (Burnak et al. (2019); Remigio and Swartz (2020); Andrés-Martínez and Ricardez-Sandoval (2022)). The interested reader is directed to reviews by Engell and Harjunkoski (2012), Baldea and Harjunkoski (2014), and Dias and Ierapetritou (2016). Zhuge and Ierapetritou (2012) proposed a closed-loop strategy to reschedule so as to mitigate disturbances at the control level. Chu and You (2014) solved a reduced integrated scheduling-control problem online, achieving computationally tractable schedules. McAllister et al. (2020) proposed potential choices for rescheduling penalties for economic model predictive control and closed-loop scheduling in response to practical rescheduling concerns. Kumar et al. (2019) presented a framework to compare the performance of deterministic and stochastic model predictive control using a battery management case study. While both the present work and the work of Kumar et al. (2019) are based on the same concept, scheduling involves discrete decisions which can lead to different relative performance across methods.

2.3 State-space resource task network model

In the context of production scheduling, a procedure to transform a MILP scheduling problem into state-space form was first proposed by Subramanian et al. (2012). Nie et al.

(2014) reformulated an extended RTN model into state-space form in order to facilitate reactive schedule design. The state-space form is advantageous as it offers a natural (re)formulation for models used in online scheduling. Although, we use the RTN based discrete time state-space model (adapted from Avadiappan and Maravelias (2021)), the ideas and conclusions presented in this work are general and not model specific. They are also applicable to state task network (STN) (Kondili et al., 1993) based discrete and continuous time models.

We define set $\mathbf{T}$ to represent the open-loop iterations and set $\mathbf{N}$, whose elements $n \in \mathbf{N}$ represent the time points and periods. A time period n is the time interval between points $n-1$ and n. A time grid starting from 0 and ending at $|\mathbf{N}^H|$ (consisting of $|\mathbf{N}^H|$ periods) is used in each open-loop iteration $t \in \mathbf{T}$. Here, subset $\mathbf{N}^H \subseteq \mathbf{N}$, denotes the time points in the prediction horizon of the open-loop problem. The progress of a batch of task $i \in \mathbf{I}$ is tracked through its progress status $k \in \mathbf{K}_i$, which increases from $k = 0$ at the start of a batch, to $k = \tau_i$ at its end, where τ_i is the processing time of task i expressed in time periods The main decision variables are $X_{in} \in \{0, 1\}$, which equal 1, when task i starts at time n, and $B_{in} \geq 0$, that denotes its batch-size. The lifted state variables $\bar{X}^k_{in}$ and $\bar{B}^k_{in}$ represent the status and size of the batches under execution at a given point in time, respectively. Effectively, the status k is represented using the binary state variables $\bar{X}^k_{in}$, wherein a batch of task i has status $k = \sum_{k'=0}^{\tau_i} k' \bar{X}^{k'}_{in}$. Here, uppercase italic letters with a bar represent the state variables.

Eqs. 1-4 represent the lifting equations for the evolution of the status of a batch. Eq. 1 assigns $k = 0$ (i.e., $\bar{X}^0_{in} = 1$) when the batch of task i starts at time point n and Eq. 2 represents the linear evolution of the status k with time point n. Eqs. 1- 2 together effectively represent the relation: $\bar{X}^k_{in} = X_{i(n-k)} \quad \forall i, n \in \mathbf{N}^H, k \in \{0, 1, ..\tau_i\}$. Similarly, for the batch size variables, Eqs. 3 and 4 are the state evolution equations and they together

effectively represent the relation: $\bar{B}^k_{in} = B_{i(n-k)} \quad \forall i, n \in \mathbf{N}^H, k \in \{0, 1, ..\tau_i\}$.

$$\bar{X}^0_{in} = X_{in} \quad \forall i, n \in \mathbf{N}^H \tag{1}$$

$$\bar{X}^k_{i(n+1)} = \bar{X}^{k-1}_{in} \quad \forall i, n \in \mathbf{N}^H, k \in \{1, 2, ..\tau_i\} \tag{2}$$

$$\bar{B}^0_{in} = B_{in} \quad \forall i, n \in \mathbf{N}^H \tag{3}$$

$$\bar{B}^k_{i(n+1)} = \bar{B}^{k-1}_{in} \quad \forall i, n \in \mathbf{N}^H, k \in \{1, 2, ..\tau_i\} \tag{4}$$

Eq. 5 enforces that the batch size of a task is between β^{LB}_i and β^{UB}_i.

$$\beta^{LB}_i X_{in} \leq B_{in} \leq \beta^{UB}_i X_{in} \quad \forall i, n \in \mathbf{N}^H \tag{5}$$

Eqs. 6 and 7 represent the resource balances, where R_{rn} is the level of resource r during period n and R^T_r is the terminal resource level. The consumption/production of resource r by task i at status k is denoted by parameters μ_{irk} and ν_{irk}. Note that negative values of the parameters indicate consumption while positive values denote production. The outgoing shipment of resource r at time n is given by V_{rn}.

$$R_{r(n+1)} = R_{rn} + \sum_{i \in \mathbf{I}} \sum_{k=0}^{\tau_i} \{\mu_{irk} \bar{X}^k_{in} + \nu_{irk} \bar{B}^k_{in}\} - V_{rn} \quad \forall r, n < |\mathbf{N}^H| \tag{6}$$

$$R^T_r = R_{rn} + \sum_{i \in \mathbf{I}} \sum_{k=0}^{\tau_i} \{\mu_{irk} \bar{X}^k_{in} + \nu_{irk} \bar{B}^k_{in}\} - V_{rn} \quad \forall r, n = |\mathbf{N}^H| \tag{7}$$

Eq. 8 ensures that the resource level is within the lower bound, λ^{LB}_r, and upper bound, λ^{UB}_r.

$$\lambda^{LB}_r \leq R_{rn} \leq \lambda^{UB}_r \quad \forall r, n \in \mathbf{N}^H \tag{8}$$

Eqs. 9 and 10 represent the backorder balance, where U_{rn} is the backlog level for product $r \in \mathbf{R}^P$ during period n and U^T_r is the terminal backlog level. The demand for

product r at time n is given by ξ_{rn}.

$$U_{r(n+1)} = U_{rn} - V_{rn} + \xi_{rn} \quad \forall r \in \mathbf{R}^P, n < |\mathbf{N}^H| \tag{9}$$

$$U_r^T = U_{rn} - V_{rn} + \xi_{rn} \quad \forall r \in \mathbf{R}^P, n = |\mathbf{N}^H| \tag{10}$$

Eqs. 11-14 represent the update equations, enforced at $n = 0$ in the current iteration t. Using these equations, we carryover the scheduling decisions, resource and backlog levels from the previous (open-loop) iteration $t-1$ to the current iteration t. The prescripts $t-1$ is used to denote variables from the previous iteration. Note that in each iteration we reset the time grid to begin from 0.

$$\bar{X}_{i,0}^k =_{(t-1)} \bar{X}_{i,0}^{k-1} \quad \forall i, k \in \{1, 2, ..\tau_i\} \tag{11}$$

$$\bar{B}_{i,0}^k =_{(t-1)} \bar{B}_{i,0}^{k-1} \quad \forall i, k \in \{1, 2, ..\tau_i\} \tag{12}$$

$$R_{r,0} =_{(t-1)} R_{r,1} \quad \forall r \tag{13}$$

$$U_{r,0} =_{(t-1)} U_{r,1} \quad \forall r \tag{14}$$

The objective function as given in Eq. 15, is to minimize the sum of backlog cost, inventory cost, and fixed cost of executing the tasks. Here, γ_r^U is the backlog cost for product resource $r \in \mathbf{R}^P$, γ_r^R is the inventory cost for material resource r, and γ_i^X is the fixed cost of executing a batch of task i.

$$\min \sum_{r \in \mathbf{R}^P} \gamma_r^U \Big(\sum_{n \in \mathbf{N}^H} U_{rn} + U_r^T \Big) + \sum_{r \in \mathbf{R}} \gamma_r^R \Big(\sum_{n \in \mathbf{N}^H} R_{rn} + R_r^T \Big) + \sum_{i \in \mathbf{I}} \sum_{n \in \mathbf{N}^H} \gamma_i^X X_{in} \tag{15}$$

Figure 2 shows the generation of a closed-loop schedule from a series of open-loop iterations for the example RTN shown in Figure 1. Batches of tasks I1 and I2 are scheduled over a prediction horizon consisting of five time periods. In iteration $t = 0$, the batches of the two tasks are slotted to occur with an idle time of one period in between them. However, in iteration $t = 1$, as the horizon is rolled forward, information about a rush order for one

batch of the product from task I2 is obtained. Therefore, the batch of I2 starts immediately after the end of the I1 batch and is followed by another I2 batch in order to satisfy the demand, as shown by the open-loop schedule. This decision is also reflected in the resulting closed-loop schedule given in the bottom row of Figure 2.

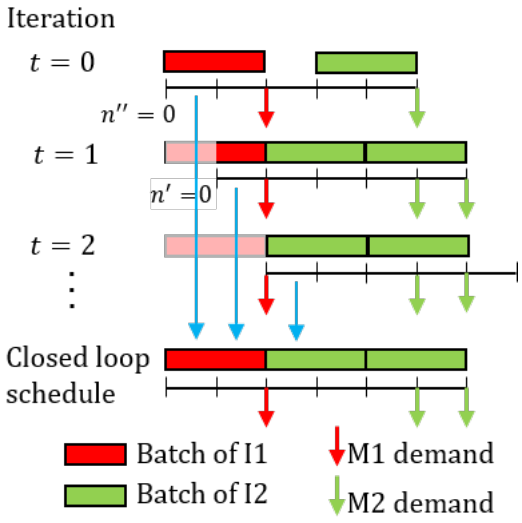


Figure 2: An example of closed-loop schedule generation. Only current decisions from each open-loop iteration are implemented (shown as blue arrows). Faded block in iterations $t = 1$ and $t = 2$ indicate past time. Red and green arrows denote the demand for products M1 and M2, respectively.

2.4 Scheduling under demand uncertainty

In chemical production scheduling, there can be uncertainties associated with demand or changes in product orders, processing time variability, equipment failures, recipe variations (Sand and Engell (2004); Li and Ierapetritou (2008); Engell (2009); Lappas and Gounaris (2016)). Researchers in the past have proposed various strategies to handle demand uncertainty. Petkov and Maranas (1997) addressed the multiperiod planning and scheduling of multiproduct plants under demand uncertainty using chance constraints. Balasubramanian and Grossmann (2004) presented an approximation strategy that consists of solving a series of two-stage stochastic programming models within a shrinking horizon framework to account for the uncertain demand, while overcoming the large computational times associated with the

solution of a multi-stage stochastic MILP model. Bonfill et al. (2004) proposed a stochastic optimization approach to manage risk in the short-term scheduling of multiproduct batch plants with uncertain demand. Cui and Engell (2010) presented a rescheduling approach based on the application of a two-stage stochastic programming model in a moving horizon framework, to handle uncertainties in demand, plant capacity, and yields. Vin and Ierapetritou (2001) addressed the problem of quantifying the schedule robustness under demand uncertainty. Lin et al. (2004) proposed a robust optimization method, wherein worst-case values of the uncertain parameters associated with processing times and market demands are used in the scheduling model.

3 Simulations

First, we describe the attributes that capture the uncertainty in demand. Second, we outline the procedure to compute the load on a process network which plays a pivotal role in closed-loop performance. Third, we describe how the demand uncertainty is accounted for in the models we consider. Finally, we discuss the generation of instances and the closed-loop evaluation procedure.

3.1 Uncertainty in demand

We analyze the closed-loop performance of different scheduling models under demand uncertainty, wherein demand is modeled in the form of orders with the order sizes being drawn from a symmetric triangular probability distribution. The uncertainty in demand is represented by two attributes: (i) observation horizon (η), and (ii) order size max-mean relative difference (ϵ) (Gupta and Maravelias, 2019). The observation horizon is the length of the horizon within which demand is deterministically known. In other words, order sizes are deterministically known within the observation horizon, from $n = 0$ to $n = \eta$, while the sizes of orders due at time points between η and H (prediction horizon length, $H = |\mathbf{N}^H|$) are estimated based on the optimization model used (refer to Section 3.3). When the prediction horizon is rolled forward, for the next online iteration, the order sizes are updated as per

the new observations (refer to Figure 3(**A**)). Note that when $\eta \geq H$, there is no demand uncertainty, since the size of all the orders within the prediction horizon is known beforehand. The order sizes for a product can vary from one time point to another. The order size max-mean relative difference (ϵ) captures the difference between the maximum and mean order size relative to the mean order size, as shown in Figure 3(**B**). Without loss of generality, we model the order sizes to follow a symmetrical triangular distribution, i.e., the maximum and minimum possible order sizes are equidistant from the mean, $c = \frac{a+b}{2}$ in Figure 3(**B**).

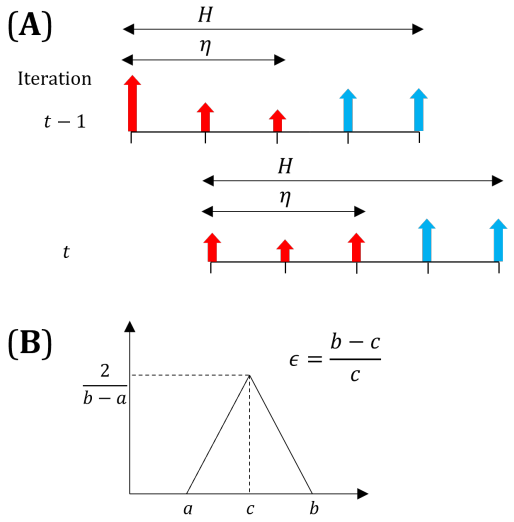


Figure 3: Attributes of demand uncertainty. (**A**) Two online scheduling iterations showing how the observation horizon of length η moves along as the prediction horizon is rolled forward. Block arrows denote orders (length proportional to order size). Deterministically known order sizes are shown in red color and the estimated sizes are shown in blue color. (**B**) Triangular probability distribution function from which demands are sampled and order size max-mean relative difference is calculated.

3.2 Load on network

The load on a process network (Λ) is an attribute that is calculated based on the production capacity of the network (Ψ) and the demand profile. In general, it is not obvious how to compute the production capacity of a network when multiple batches of different tasks are to be executed depending on the product-mix (i.e., ratio of different products to be produced). Gupta and Maravelias (2019) introduced a systematic procedure to determine

the production capacity of a network, by solving a modified profit maximization problem with periodicity constraints, that results in schedules with peak capacity utilization. Using this procedure, we compute Ψ as the sum of the production quantities of all products per unit time. The load on network is then given by $\Lambda = \xi/\Psi$, where ξ is the sum of orders over all products per unit time.

3.3 Optimization models

3.3.1 Deterministic

In deterministic optimization, the demand uncertainty is not explicitly modeled. The scheduling decisions are computed using the state-space RTN model given in Section 2.3, considering that the order sizes are deterministically known for time points within the observation horizon η, while the mean of the triangular probability distribution from which demands are sampled, is used for order sizes at time points beyond η. The model mitigates the effect of demand uncertainty by treating them as disturbances and solely relies on feedback provided by the online scheduling procedure. Note that feedback in the context of demand uncertainty, is based on information about the future orders. The main advantage of deterministic optimization is that the online computations are typically inexpensive, for small to medium scale scheduling instances.

3.3.2 Robust

In the robust optimization approach that we employ in this work, an order size near the maximum order size is assumed for the orders at time points beyond η. In reality, the worst-case demand (i.e., maximum order size) rarely occurs and scheduling based on the worst-case value of the uncertain parameters leads to conservative solutions (Moradi and MirHassani, 2016). In this work, we choose the 95^{th} percentile value from the triangular distribution as the near maximum order size. Note that the robust model formulation is same as the deterministic formulation given in Section 2.3, except for the value of the demand parameter ξ_{rn}. Hence, the robust optimization model retains the advantage of having similar

computational tractability as the deterministic model.

3.3.3 Stochastic programming

The stochastic programming formulations can accommodate decision making at different stages according to the sequence in which uncertainty is revealed. The evolution of the paths due to the uncertainty in parameters is represented using a scenario tree (Birge and Louveaux, 2011). In multi-stage stochastic programming, the uncertainty is modeled by a scenario tree with multiple stages. At each stage, the decisions are based on the realized values of the parameters until that point in the tree, while the values of the uncertain parameters are only known probabilistically. Multi-stage problems inherently have a complex nested structure, hence, they are approximated to two-stage problems. In a two-stage problem, the decision variables are divided into the first and second stage variables. The first stage variables represent "here and now" decisions that are made before uncertainty is revealed and thus, have to be identical for all scenarios. The second stage variables represent "wait and see" decisions, which are recourse variables that are independent for each scenario.

In this work, we initially consider three different stochastic programming formulations based on: (i) multi-stage; (ii) two-stage; and (iii) reduced two-stage approach (see illustration, using an example, in Figure 4). For the multi-stage approach, we see, in Figure 4(**A**), that the scenarios branch out at $n = 6, 8$ where the demand is uncertain. While, using the two-stage approach in Figure 4(**B**), all scenarios branch out at $n = 6$. In the reduced two-stage approach in Figure 4(**C**), fewer scenarios branch out at $n = 6$ since a mean order size (based on probability distribution) is assumed at $n = 8$ to limit the total number of scenarios and keep the problem tractable. In other words, the more distant future is represented by only one deterministic scenario with the expected values of the uncertain parameters, whereas the immediate future is modeled by a tree of scenarios that represent different order sizes (Cui and Engell, 2010).

We find that the closed-loop performance of the different stochastic programming formulations are similar (refer to Supplementary Information), so we choose the reduced two-

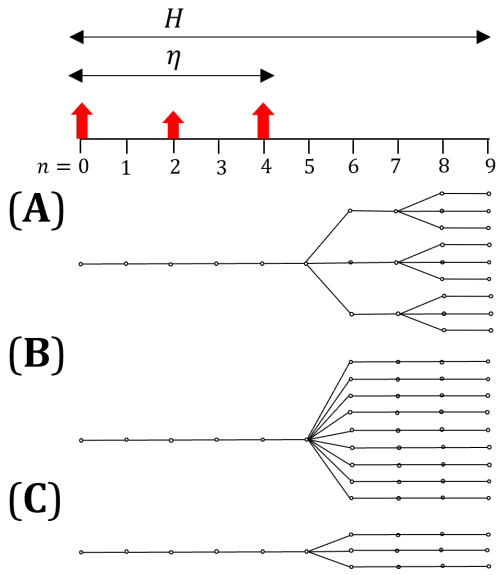


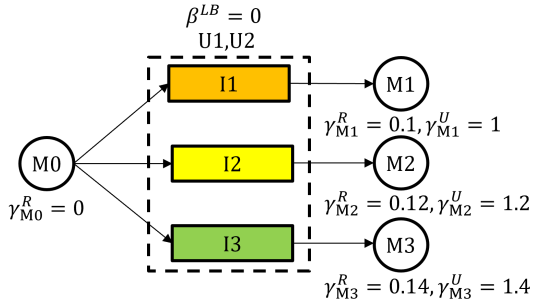
Figure 4: Illustration of scenario trees for stochastic programming (**A**) multi-stage, (**B**) two-stage, and (**C**) reduced two-stage approaches. Deterministically known order sizes are shown in red color and orders are due every alternate time point.

stage approach to keep the problem tractable. In this approach, a set of finite scenarios $s \in \mathbf{S}$ and the associated probabilities are determined from the continuous probability distribution using a moment matching technique (Høyland and Wallace, 2001). Here, an optimization problem is solved to determine an approximating distribution composed of ten scenarios such that the first three moments of the approximating distribution match (as well as possible) the moments of the given continuous probability distribution. The model formulation for the reduced two-stage stochastic programming approach is given in Appendix A. Note that the online solution of the reduced two-stage stochastic programming model for medium to large-scale instances requires large computational times since each closed-loop simulation requires the solution of tens of open-loop optimizations. Moreover, to generate results in this work, we solve millions of models that leads to scalability issues, thereby making it highly challenging to use large-scale networks.

3.4 Instances

We investigate the closed-loop performance of the optimization models as a function of load (Λ), order size max-mean relative difference (ϵ), and observation horizon (η), by

varying one of these attributes at a time, while keeping the others fixed. For a given process network (say, network NT1 shown in Figure 5), we use a long prediction horizon (e.g., $H = 24$ periods, for the networks used in this work) for the open-loop problem and choose a value for the time between orders Ω, and the re-optimization time-step Δ. Note that the prediction horizon length is chosen such that using a horizon longer than the chosen value would not change results of the closed-loop simulations. Re-optimization time step Δ is chosen based on the criteria outlined in Gupta and Maravelias (2019). Moreover, re-optimizing more frequently than the greatest common factor of time-related data does not yield any additional benefit towards closed-loop performance. We consider observation horizon, $\eta = 6, 10, 14$ periods such that, on average, it corresponds to different number of orders that are deterministically known during each open-loop iteration. Moreover, we consider load, $\Lambda = 0.25, 0.5, 0.75, 1$, and order size max-mean relative difference, $\epsilon = 0.375, 0.75$, and simulation horizon of 48 periods.



Case	β^{UB}	τ_{I1}	τ_{I2}	τ_{I3}
A	15	3	6	6
B	10	2	4	4
C	5	1	2	2

Figure 5: Network NT1 with 3 tasks, 4 material resources, and 2 units, with 3 different cases (parameter values) denoted by **A**, **B**, and **C**.

To start with, we determine the production capacity Ψ of a process network for a given demand ratio of the products. In this work, for every network, we assume a relative demand ratio of 1 across all the end products, but other ratios can be trivially handled using the same approach. We then select load Λ at different levels and for each one of them, we find

the demand to be met per unit time for each of the products. Based on time between orders Ω, we calculate the mean order size of the symmetric triangular distribution from which demands are sampled (every Ω periods). Using order size max-mean relative difference ϵ and the mean order size, the symmetric triangular probability distribution at each Λ, can be identified. A demand sample consists of order quantities for all products drawn from their respective distributions, at specified time points over the simulation horizon. For a given network, we draw 50 such samples leading to the generation of 50 instances. This number of demand samples was chosen so that the relative closed-loop performance of the different optimization models remains practically the same if another set of 50 samples were used. Finding the closed-loop solution for a given instance, using the tuple (model, Λ, ϵ, η), wherein model refers to the choice of deterministic, robust, stochastic programming model, is referred to as a run. Each run requires solution to several open-loop problems to find the closed-loop schedule. Note that we use the same set of 50 demand samples for a given Λ across different models, ϵ, and η.

3.5 Closed-loop evaluation

Through the online scheduling procedure using optimization models (described in Section 3.3) and given Λ, ϵ, η, a closed-loop schedule is obtained for the simulation horizon of 48 periods. To avoid any initial schedule ramp-up from affecting our conclusions, the closed-loop cost is evaluated only from periods 10 to 48. For each instance, we obtain multiple closed-loop solutions over the space of the attributes Λ, ϵ, η and evaluate the corresponding closed-loop costs. We then average over the 50 instances (i.e., demand samples), to find the mean closed-loop cost, which is used to compare the closed-loop performance across the models. Note that the closed-loop framework can also be used with other objective functions such as profit maximization, which can be reduced to a per period (stage) metric (e.g., \$/hr). To obtain a closed-loop schedule, the requisite number of open-loop optimizations are solved using a termination criterion of 1% relative optimality gap unless specified otherwise (though most instances were indeed solved to optimality), using default solver options in CPLEX

12.10.0 via GAMS 32.2.0, using the Della cluster at Princeton University. MATLAB R2018b was used to import and analyze data from GAMS gdx files, that contained the scheduling decisions and closed-loop costs.

4 Results

We carry out closed-loop simulations on network NT1 shown in Figure 5 by varying Λ, ϵ, and η for the different models with $H = 24$ periods, $\Omega = 10$, and $\Delta = 1$. In Figure 6, we show the results of the simulations for the parameter values corresponding to case **A** in network NT1. Note that cases **A**, **B**, and **C** differ from each other only in the processing times of tasks and the maximum unit capacity (refer to Figure 5). In the panels, each data point denotes the mean closed-loop cost over 50 demand samples and is scaled by the least (mean) closed-loop cost within that panel. Note that in Figure 6, the schedule costs are different across different loads, hence, a cost of 1, in the panel corresponding to $\Lambda = 0.25$ is not the same as cost of 1 in the panel corresponding to $\Lambda = 0.5$. We carry out a two-way analysis of variance (ANOVA) on the results to discern statistical significance (Wonnacott and Wonnacott, 1984). We perform ANOVA on the closed-loop cost over 50 demand samples to ascertain whether there is a statistically significant difference between the closed-loop performance of the different optimization models. We find the p_{ANOVA} values (shown in top-left of each panel in Figure 6), that denote the probability of realization of observed data if the null hypothesis was true. We define our null hypothesis to be that the attributes under consideration (i.e., ϵ and choice of model) have no effect on closed-loop costs. If p_{ANOVA} values are less than 0.05 (i.e., 5%), it indicates a statistically significant effect of the attributes, while in the faded panels in Figure 6, there is no statistically significant effect of the choice of model on closed-loop costs since the corresponding p_{ANOVA} values are greater than 0.05. The best choice of model for a given set of attributes is the one that incurs the least mean closed-loop cost.

In Figure 7, we show the results of the simulations with $\eta = 6$, for network NT1

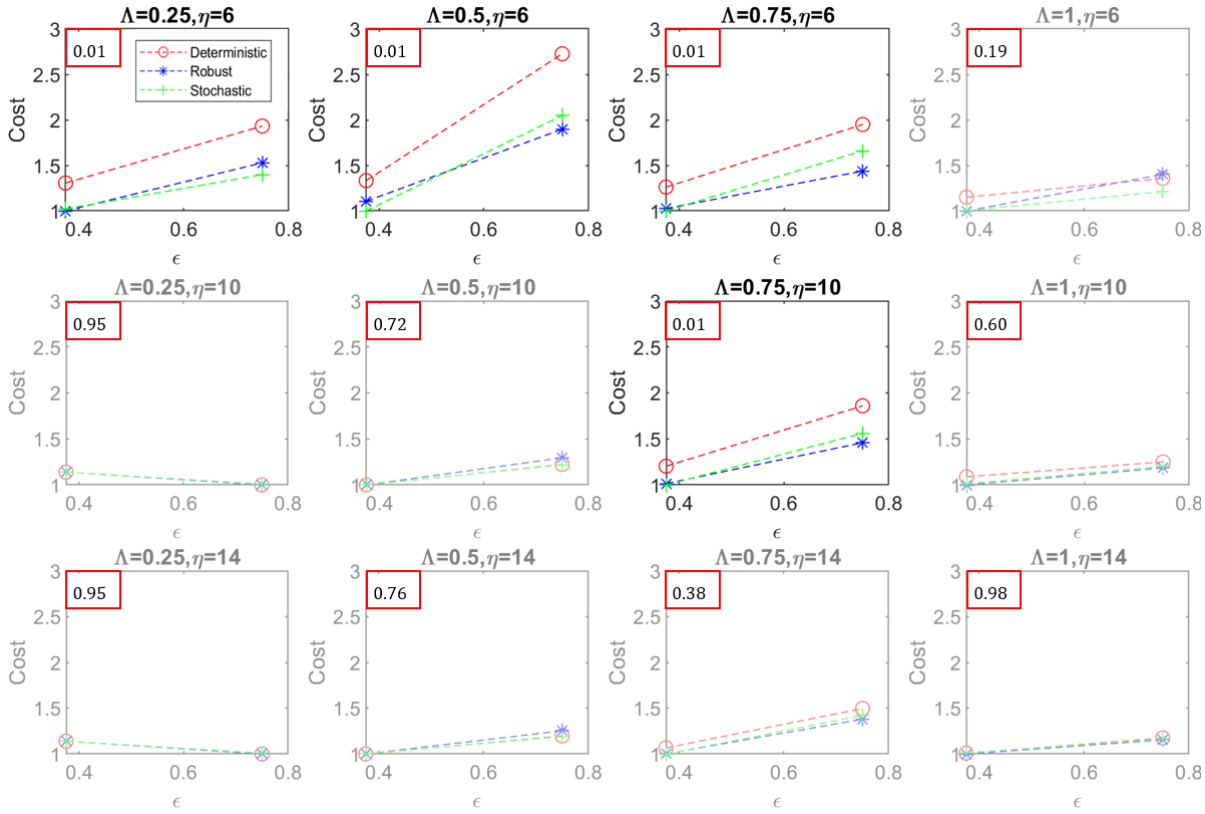


Figure 6: Effect of Λ, ϵ, and η on closed-loop cost using deterministic, robust, and stochastic programming models for network NT1 case **A**. The p_{ANOVA} value is given at the top-left of each panel and faded panels indicate that there is no statistically significant effect of the choice of model on the closed-loop cost at those values of Λ and η.

cases **B** and **C** (refer to Supplementary Information for results of $\eta = 10, 14$, for NT1 cases **B** and **C**). From **A** to **C**, the processing times and maximum unit capacity are decreased proportionally such that the production capacity of the networks remains the same. Qualitatively, we claim that the "speed" of the network NT1 increases from **A** to **C**, since the network is able to respond quickly as batches of smaller sizes with shorter processing times can be scheduled to meet the demand. When we compare the closed-loop performance of the deterministic and robust models, we observe that, in general, as the speed of the network increases, the deterministic models start performing relatively better than the robust models (refer to Figure 7**C**). This is due to the larger inventory cost incurred in the robust model, since multiple batches are required to meet the demand for a given load in network NT1 case **C** as compared to case **A**. However, at lower speeds and a shorter observation horizon η (refer to the first row in Figure 6), the deterministic model performs poorly due to the large backlog cost incurred, as batches start before the demand is deterministically known since the processing times are large. The stochastic programming model does not follow any general trend as the speed of the network is increased. Therefore, even for a relatively simple network NT1, we observe that changing the speed of the network (from **A** to **C**) changes the relative closed-loop performance of the models (compare the individual panels in Figure 6 and 7) for a given set of attributes Λ, ϵ, and η. Moreover, we find that changing the time between orders (Ω), the ratio of unit backlog to inventory cost (γ_r^U/γ_r^R) also changes the relative closed-loop performance of the models (shown in Supplementary Information).

We carry out simulations on a more complex network NT2 shown in Figure 8 and obtain the results shown in Figure 9. Due to the increased computational burden encountered during the solution of the open-loop optimizations, we set the termination criterion to 5% relative optimality gap. We also carry out simulations on two other networks and show the results in Supplementary Information. We observe trends and obtain useful insights that are applicable across most networks for specific ranges of attributes, as explained in the following section.

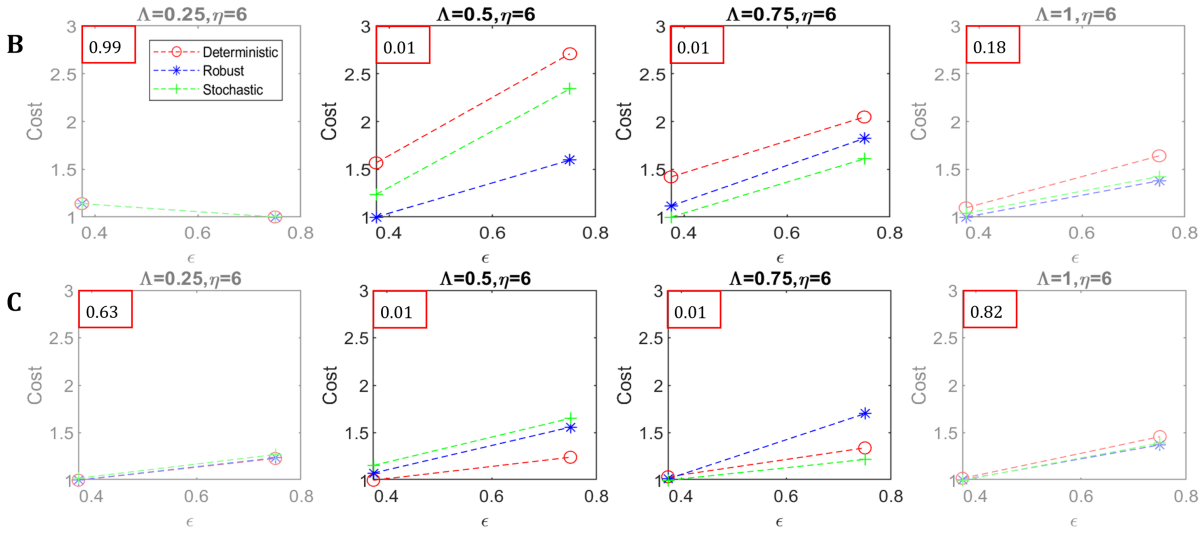


Figure 7: Effect of Λ, ϵ on closed-loop cost using deterministic, robust, and stochastic programming models with $\eta = 6$ for network NT1 cases **B** and **C**.

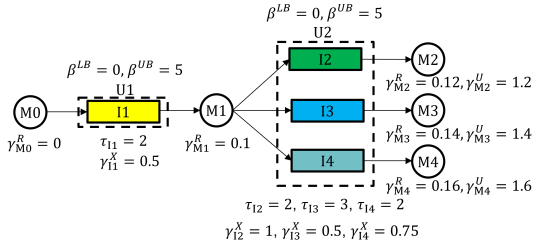


Figure 8: Network NT2 with 4 tasks, 5 material resources, and 2 units.

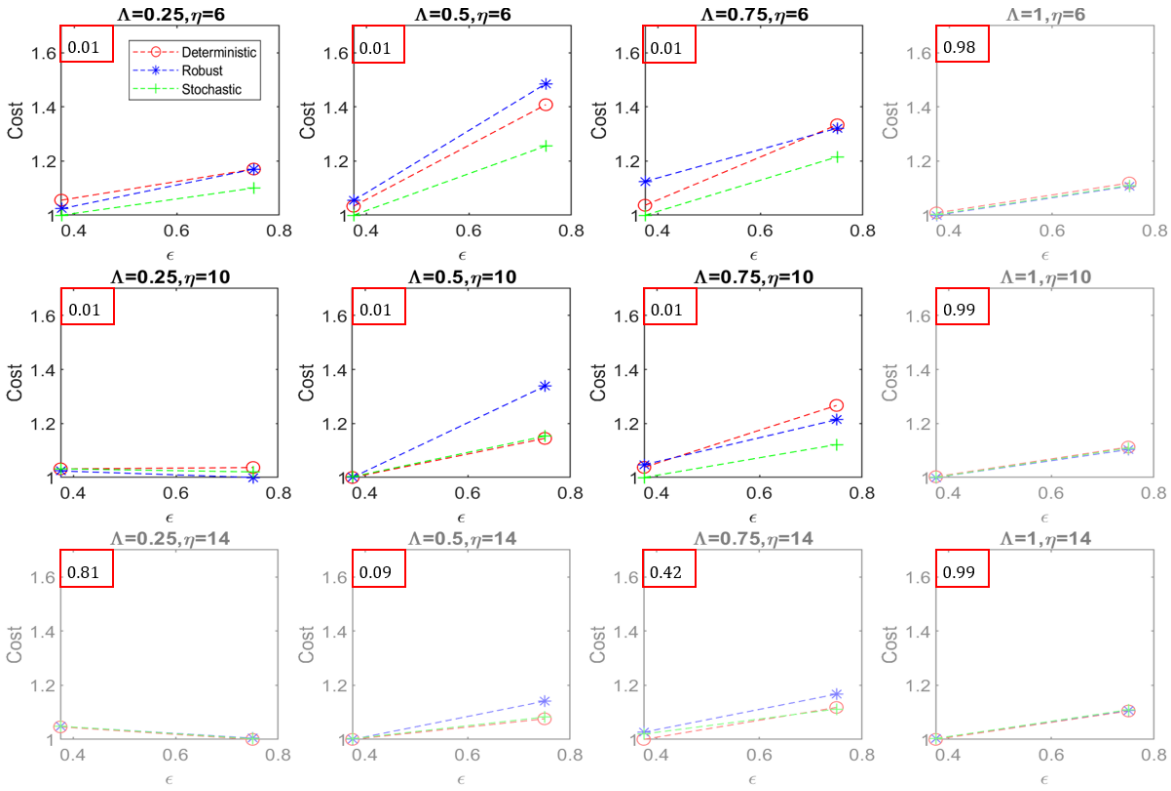


Figure 9: Effect of Λ, ϵ, and η on closed-loop cost using deterministic, robust, and stochastic programming models for network NT2.

5 Discussion

In this section, we discuss the closed-loop performance of models as Λ, ϵ, and η are varied. We also explain the importance of accounting for uncertainty a priori through the stochastic programming model compared to scheduling based solely on feedback using a deterministic model.

5.1 Model performance

On analyzing the closed-loop performance of the deterministic, robust, and stochastic programming models across multiple networks (refer to Figures 6,7, and 9), we present trends that are observed for specific ranges of attributes and draw useful insights.

First, we observe that at very low loads (say, $\Lambda \leq 0.25$), given order size max-mean relative difference ϵ, and a long enough observation horizon η (depends on the network),

closed-loop performances of the deterministic, robust, and stochastic programming models are similar as seen by the coincident nature of the plots of closed-loop cost versus ϵ (refer to first column in Figures 6,7, and 9). Moreover, based on two-way ANOVA tests, we find that there is no statistically significant effect of the choice of model on the closed-loop cost. This implies that the deterministic model can perform as well as a computationally expensive stochastic programming model for these set of attributes (refer to Supplementary Information for model statistics). This is because at very low loads and a long enough observation horizon, batches start in the schedule to meet a given demand only after it is deterministically known and they are aligned to finish coinciding with the due-times to reduce the inventory costs. Though the demand is uncertain at time points beyond η, the initial decisions (i.e., batch start times and sizes) in the open-loop schedules using different models are similar, resulting in similar closed-loop performance.

Second, we see that at low or intermediate loads (say, $\Lambda \leq 0.75$), given ϵ and a short observation horizon η, the best choice of model solely depends on the network characteristics. For example, at $\Lambda = 0.5$ and $\eta = 6$ in network NT1 case **A** (shown in Figure 6), the robust and stochastic programming model perform much better than the deterministic model. In all models, the batches start before the demand is deterministically known (since batch processing times are large and η is short). In the deterministic model, batches of sizes corresponding to the mean order size start, resulting in heavy backlog costs when there are larger than mean orders. However, at $\Lambda = 0.5$ and $\eta = 6$ in network NT1 case **C** (shown in Figure 7), we find that the deterministic model performs much better since it results in lower backlog and inventory costs compared to the other models. Here, surprisingly, the deterministic model also outperforms the stochastic programming model (explained in detail in Section 6.2). Typically, at intermediate loads and short observation horizons, across most networks, the stochastic programming model gives better closed-loop performance and would be a safer choice (refer to third column in Figures 6,7, and 9), since it incurs lower inventory cost than the robust model and lower backlog cost than the deterministic model.

Third, for high loads (say, $\Lambda \sim 1$), irrespective of ϵ and η, the closed-loop performances of the models are similar based on the two-way ANOVA test (refer to fourth column in Figures 6,7, and 9). This is because a high load requires the network to be operated close to its production capacity, and there is not enough slack in the "packed" schedule to fit in additional batches, thus, large order sizes will lead to a backlog, regardless of the observation horizon. Consequently, even the robust models that assume a near maximum order size at due-times beyond η, do not perform better than the deterministic model. Moreover, note that the notion of very low, intermediate, and high loads are qualitative and may not exactly correspond to the ranges of Λ specified in this section.

Fourth, we find that the mean closed-loop cost for all models increases as ϵ increases, at loads greater than a threshold (say, $\Lambda > 0.25$ for network NT1 cases **A**, **B**, **C**, and NT2). A larger ϵ implies many orders larger than the mean as well as many orders smaller than the mean (since we use a symmetric triangular distribution). The unit backlog cost is 10 times as expensive as the unit inventory cost, thus, the backlog cost incurred when the order sizes are large predominantly influences the closed-loop cost. However, at loads lesser than the threshold and a long enough observation horizon (refer to Figure 6 at $\Lambda = 0.25$ and $\eta = 10$), the mean closed-loop cost for all models decreases as ϵ increases. This is because at low loads, the backlog costs are negligible and the inventory cost decreases as the bigger batches are aligned to finish coinciding with the due-times of larger than mean orders. Hence, the inventory cost for other smaller batches is reduced, when ϵ is larger.

Fifth, for a long enough observation horizon η, across all Λ and ϵ, the closed-loop performances of the models are similar based on the two-way ANOVA test. In Figure 10, we see that for network NT1 case **A** at $\eta = 14$, the mean closed-loop costs for the models seem to coincide in each panel, for different Λ. This is because the demand is deterministically known at many time points in the prediction horizon (given $H = 24$), and when $\eta \geq H$, the open-loop problem becomes deterministic as there is no uncertainty in demand. Therefore, the closed-loop performances of the models are similar for a long enough observation horizon.

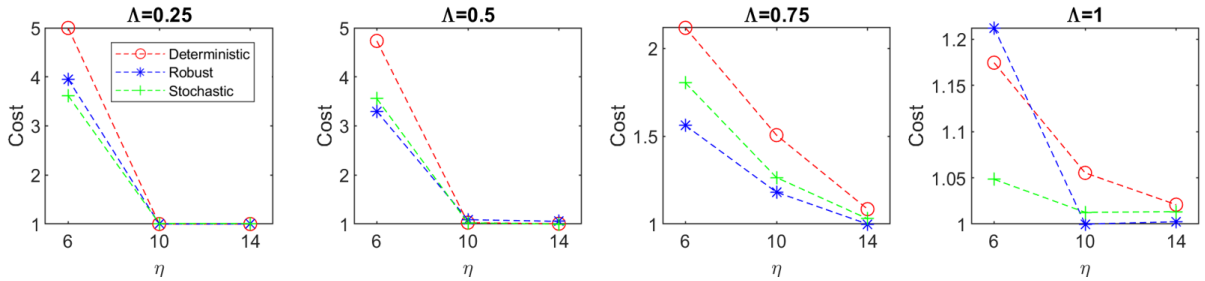


Figure 10: Effect of η on closed-loop cost for $\epsilon = 0.75$ across different Λ using deterministic, robust, and stochastic programming models for network NT1 case **A**.

Sixth, we note that for intermediate loads (say, $0.25 < \Lambda \leq 0.75$), as η increases, the deterministic and stochastic programming models lead to a larger reduction in closed-loop cost in comparison to the robust model. This can be clearly seen by the slope of the different lines in the panels corresponding to $\Lambda = 0.5, 0.75$ in Figure 10. The slope of the line representing the robust model (indicated in blue color) is less compared to the other lines as η increases from 6 to 10. As the observation horizon increases, the backlog costs reduce considerably in the deterministic and stochastic programming models since batches start after demand is deterministically known. However, in the robust model, batches always start early irrespective of η since a near maximum order size is assumed for the uncertain demand parameters, resulting in low backlog and high inventory costs. Thus, the decrease in closed-loop cost as η increases is mainly due to the decrease in inventory cost. In general, at intermediate loads, as η increases, the decrease in backlog costs in the deterministic and stochastic programming model is much larger compared to the decrease in inventory cost in the robust model.

5.2 Importance of accounting for uncertainty a priori

The stochastic programming model accounts for demand uncertainty a priori by modeling the uncertain demand by a set of scenarios and, in addition, uses feedback to revise the decisions. In other words, it is both a proactive as well as a reactive approach, while the deterministic model is a purely reactive approach that relies solely on feedback

to mitigate uncertainty in demand. To compare the closed-loop performance between these models and understand the importance of accounting for uncertainty a priori, we carry out simulations for network NT1 case **A**, by varying the re-optimization time-step Δ. The results are shown in Figure 11, wherein, we observe that at intermediate and high loads (say, $\Lambda > 0.25$), for a short observation horizon, the stochastic programming model at a higher Δ (i.e., model is re-optimized less frequently) outperforms the deterministic model with a lower Δ (i.e., model is re-optimized more frequently). For example, at $\Lambda = 0.75$, stochastic programming model with $\Delta = 6$ outperforms the deterministic model with $\Delta = 1$.

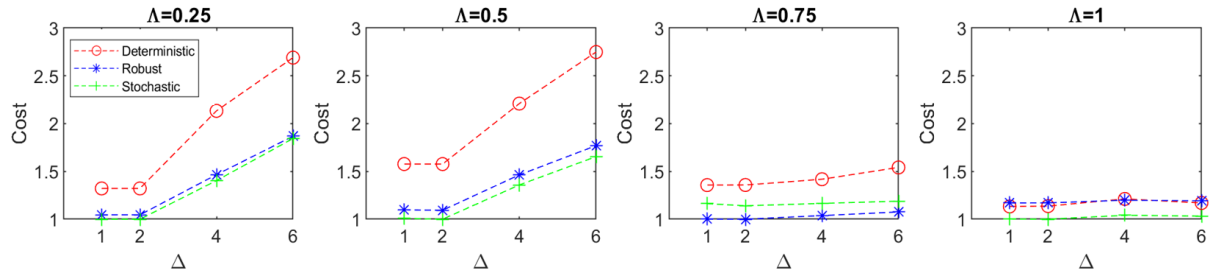


Figure 11: Effect of Δ on closed-loop cost for $\epsilon = 0.75$ and $\eta = 6$ across different Λ using deterministic, robust, and stochastic programming models for network NT1 case **A**.

In the deterministic model at intermediate and high loads, a smaller batch may start resulting in lost production, that cannot be overcome by more frequent re-optimization. While in the stochastic programming model, the batch sizes are larger, thereby resulting in better closed-loop performance, though the model may not be re-optimized as frequently. Consequently, the overall computational time for the (online) stochastic programming approach could be reduced through infrequent re-optimization. Furthermore, as Δ increases, a closed-loop schedule tends to approximate an open-loop schedule (due to infrequent feedback), and naturally, the stochastic programming model starts to perform much better than the deterministic model.

In closing, we note that another approach to generate high quality robust schedules is through the addition of terminal constraints (Risbeck et al., 2019). Recently, McAllister et al. (2021) proposed an inherently robust closed-loop algorithm using terminal constraints

to track a reference trajectory. However, when there is large variability in demand, identifying such periodic reference trajectories (of high quality) is non-trivial. In that respect, the methods presented in this paper provide guidelines that are readily applicable to all problems though the proposed methods do not provide the theoretical guarantees derived in McAllister et al. (2021).

6 Paradoxes

In this section, we present certain paradoxes that are observed based on the results of the simulations across different networks and explain the reasoning behind them.

6.1 Closed-loop cost increases as η increases

In general, the closed-loop cost for all models is expected to decrease as the observation horizon η increases, for all demand samples across Λ and ϵ. This is because in an online scheme, the future demands become deterministically known at an earlier point of time, leading to processing of batches of appropriate sizes. However, we observe that at high loads (say, $\Lambda \sim 1$), for certain demand samples, the closed-loop cost increases as η increases. For example, in Figure 12, we show the closed-loop schedules for a specific demand sample using the deterministic model as η is varied. We note that in Figure 12(**B**) when $\eta = 10$, the closed-loop cost is higher than that at $\eta = 6$ because of the increase in backlog cost. While computing decisions at iteration $t = 0$ with $\eta = 6$, a mean order size is assumed at due-time 10, whereas when $\eta = 10$, the actual demand at time 10 (which is lesser than mean order size for this sample) is deterministically known, hence, a smaller batch of task I3 is executed in unit U2 at time 6 in Figure 12(**B**). This results in a lost production opportunity, leading to backlogs as the (higher than mean) demand at time 20 cannot be met. These trends are also observed for demand samples using the robust and stochastic programming models at high loads. Note that though these trends are observed for 10 of the 50 demand samples at $\Lambda = 1$ and $\epsilon = 0.375$ in network NT1 case **A**, the mean closed-loop cost over 50 samples does not necessarily increase as η increases.

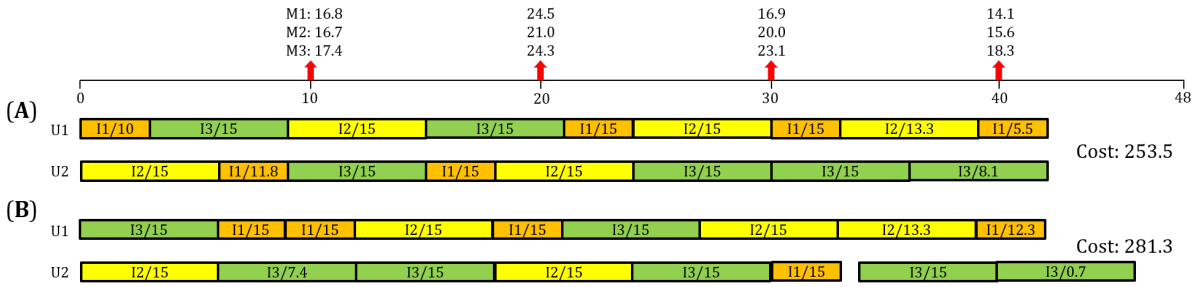


Figure 12: Gantt chart showing closed-loop schedules with $\Lambda = 1$ and $\epsilon = 0.375$ using deterministic model for network NT1 case **A**. Batches are color coded and sizes are indicated within, demand for products is shown on top of the red arrows at due-times, and closed-loop cost is given towards the right of schedule. The observation horizons are (**A**) $\eta = 6$, and (**B**) $\eta = 10$.

6.2 Deterministic model outperforms stochastic programming model

In most of the panels in Figures 6,7, and 9, we see that the stochastic programming model performs atleast as well as the deterministic model. This is because the uncertainty in demand is modeled by a set of scenarios compared to using a mean order size and the scenarios effectively capture the underlying probability distribution. However, in Figure 13(**A**), we see that the deterministic model outperforms the stochastic programming model based on the two-way ANOVA test, at $\Lambda = 0.5$ and $\eta = 6$ for network NT1 case **C**. Moreover, all the 50 demand samples follow this trend. In Figure 13(**B**) and (**C**), we show a part of the closed-loop schedule for one of the samples using the deterministic and stochastic programming model, respectively. The processing times of tasks in network NT1 case **C** are smaller, hence, the batches can start after the demand is deterministically known. In this example, in the deterministic model, batches start at time 4 when the actual demand at time 10 is known, since $\eta = 6$. While in the stochastic programming model, smaller batches start at time 3, resulting in low production amounts, and leading to backlogs since the demand at time 10 is not satisfied.

Furthermore, we know that as ϵ increases, the mean order size used in the deterministic model is still the same while a higher maximum order size is used in the robust model and a new set of scenarios associated with the triangular distribution is used in the stochastic

Figure 13: (**A**) Closed-loop performance of the models for network NT1 case **C** at $\Lambda = 0.5$ and $\eta = 6$. Gantt chart showing the first 20 periods of simulation horizon for the closed-loop schedules using (**B**) deterministic model, and (**C**) stochastic programming model. Batches are color coded and sizes are indicated within, demand for products is shown on top of the red arrows at due-times.

programming model. However, this does not imply that the deterministic model performs relatively worse as ϵ increases. Here, in Figure 13(**A**), as ϵ increases, we see that the deterministic model performs relatively better because the increase in backlog cost for the deterministic model is lesser than the increase in inventory costs for the other models for network NT1 case **C**.

7 Conclusions

While demand uncertainty has been studied extensively in the context of the open-loop problem, the impact of accounting for it in the presence of feedback, which has been shown to be so powerful, has not been considered. Accordingly, the goal of this work was to investigate the importance of accounting for demand uncertainty a priori in production scheduling in the presence of feedback. The contributions of the work are three-fold: (i) we introduced new concepts necessary to understand how feedback in scheduling impacts closed-loop performance; (ii) we developed guidelines in terms of characteristics and attributes that impact the selection of the most appropriate model; and (iii) we drew new insights regarding the behavior of closed-loop scheduling solutions.

In terms of concepts, we first presented different optimization models (deterministic, robust, and stochastic programming) used to generate the open-loop schedules and described

how the demand uncertainty is modeled in each case. Second, we introduced two attributes that can be used to describe demand uncertainty in the context of online scheduling: the order size max-mean relative difference (ϵ), the impact of which, as a metric of demand variation, is understood, alas not widely studied; and the observation horizon (η), whose importance was, to our knowledge for the first time, recognized and studied. Third, we introduced the concept of "speed of a network" which is a metric of how fast a production system can react which, in that respect, fills a gap in the analogies that have been drawn between scheduling and dynamic optimization. Fourth, we identified the key attributes and instance characteristics (e.g., network load) that impact the selection of the most appropriate model.

In terms of guidelines, we derived the following: (i) At very low and high loads or when the observation horizon is long enough, the closed-loop performances of the models are similar, thus, a deterministic model which is computationally inexpensive, could be the preferred choice. (ii) At intermediate loads or when the observation horizon is small, the best choice of model depends on the network characteristics. However, for most networks, the stochastic programming model would be a safer choice in terms of closed-loop solution quality. (iii) At intermediate or high loads, and when the observation horizon is short, we showcased the importance of accounting for uncertainty a priori compared to feedback, since the stochastic programming model that is re-optimized infrequently performs much better than the deterministic model that is re-optimized frequently. Note that the closed-loop framework employed in this work can also be used to handle endogenous uncertainty. However, drawing useful insights based on how effectively the different optimization models account for endogenous uncertainty might be non-trivial.

It is important to clarify that we did not derive any theoretical results regarding the relative performance of the three models. Though we note that the available theoretical results for MPC provide guarantees with respect to a nominal solution and not performance ranking across competing approaches. Nevertheless, the derived guidelines appear to be

broadly applicable because they remained consistent across the different networks we studied (see Supplementary Information); and, most importantly, because we were able to explain, intuitively, why they hold true. Finally, using a set of what, on the surface, appear to be paradoxes, such as the increase in closed-loop cost as the observation horizon increases, we showed that predicting the best model to be used for a given network and set of attributes is non-trivial. This is important because it goes against the conventional wisdom according to which accounting for uncertainty always leads to better results.

To the best of our knowledge, the work presented herein is the first of its kind in understanding the importance of accounting for demand uncertainty a priori compared to closed-loop batch scheduling based solely on feedback (i.e., using deterministic model).

Acknowledgment

The authors acknowledge financial support from the National Science Foundation under grant CBET-2026980. The authors acknowledge the simulations reported in this paper was substantially performed using the Princeton Research Computing resources at Princeton University which is consortium of groups led by the Princeton Institute for Computational Science and Engineering (PICSciE) and Office of Information Technology's Research Computing.

Appendix A. Stochastic programming model

The variables and constraints (Eqs. 16-29) of the reduced two-stage stochastic programming model are similar to those of the state-space RTN model in Section 2.3 but are replicated across the set of scenarios $s \in \mathbf{S}$. Here, the subset $\mathbf{S}_n$ represents the scenarios at time point n. For time points in the first stage (i.e., $n \leq \sigma$, where σ is the duration of first stage), $\mathbf{S}_n$ is composed of only one scenario, i.e., $s \in \mathbf{S}_n = \{1\}$, and for $n > \sigma$, $\mathbf{S}_n$ is composed of the ten scenarios that branch out. Note that in Eqs. 26-29, variables corresponding to the only scenario (i.e., $s = 1$) from the previous iteration $t - 1$, is used to assign initial values to

the variables at iteration t.

$$\bar{X}^0_{ins} = X_{ins} \quad \forall i, n \in \mathbf{N}^H, s \in \mathbf{S}_n \tag{16}$$

$$\bar{X}^k_{i(n+1)s} = \bar{X}^{k-1}_{ins} \quad \forall i, n \in \mathbf{N}^H, k \in \{1, 2, ..\tau_i\}, s \in \mathbf{S}_n \tag{17}$$

$$\bar{B}^0_{ins} = B_{ins} \quad \forall i, n \in \mathbf{N}^H, s \in \mathbf{S}_n \tag{18}$$

$$\bar{B}^k_{i(n+1)s} = \bar{B}^{k-1}_{ins} \quad \forall i, n \in \mathbf{N}^H, k \in \{1, 2, ..\tau_i\}, s \in \mathbf{S}_n \tag{19}$$

$$\beta^{LB}_i X_{ins} \leq B_{ins} \leq \beta^{UB}_i X_{ins} \quad \forall i, n \in \mathbf{N}^H, s \in \mathbf{S}_n \tag{20}$$

$$R_{r(n+1)s} = R_{rns} + \sum_{i \in \mathbf{I}} \sum_{k=0}^{\tau_i} \{\mu_{irk} \bar{X}^k_{ins} + \nu_{irk} \bar{B}^k_{ins}\} - V_{rns} \quad \forall r, n < |\mathbf{N}^H|, s \in \mathbf{S}_n \tag{21}$$

$$R^T_{rs} = R_{rns} + \sum_{i \in \mathbf{I}} \sum_{k=0}^{\tau_i} \{\mu_{irk} \bar{X}^k_{ins} + \nu_{irk} \bar{B}^k_{ins}\} - V_{rns} \quad \forall r, n = |\mathbf{N}^H|, s \in \mathbf{S}_n \tag{22}$$

$$\lambda^{LB}_r \leq R_{rns} \leq \lambda^{UB}_r \quad \forall r, n \in \mathbf{N}^H, s \in \mathbf{S}_n \tag{23}$$

$$U_{r(n+1)s} = U_{rns} - V_{rns} + \xi_{rns} \quad \forall r \in \mathbf{R}^P, n < |\mathbf{N}^H|, s \in \mathbf{S}_n \tag{24}$$

$$U^T_{rs} = U_{rns} - V_{rns} + \xi_{rns} \quad \forall r \in \mathbf{R}^P, n = |\mathbf{N}^H|, s \in \mathbf{S}_n \tag{25}$$

$$\bar{X}^k_{i,0,1} =_{(t-1)} \bar{X}^{k-1}_{i,0,1} \quad \forall i, k \in \{1, 2, ..\tau_i\} \tag{26}$$

$$\bar{B}^k_{i,0,1} =_{(t-1)} \bar{B}^{k-1}_{i,0,1} \quad \forall i, k \in \{1, 2, ..\tau_i\} \tag{27}$$

$$R_{r,0,1} =_{(t-1)} R_{r,1,1} \quad \forall r \tag{28}$$

$$U_{r,0,1} =_{(t-1)} U_{r,1,1} \quad \forall r \tag{29}$$

The non-anticipativity constraints (Eqs. 30-34) enforce that the first stage variables are equal across different scenarios since these decisions have to be made before the demand uncertainty is realized. Note that in Eqs. 30 and 31, the first stage decisions X_{ins}, B_{ins} indirectly affect the second stage lifted state variables $\bar{X}^k_{ins}, \bar{B}^k_{ins}$, hence, the equations are

written for time points $n' = n + k$ with $n \leq \sigma$.

$$\bar{X}^k_{in's} = \bar{X}^k_{i,n',1} \quad \forall i, k \in \{0, 1, ..\tau_i\}, n \leq \sigma, n' = n + k, s \neq 1 \tag{30}$$

$$\bar{B}^k_{in's} = \bar{B}^k_{i,n',1} \quad \forall i, k \in \{0, 1, ..\tau_i\}, n \leq \sigma, n' = n + k, s \neq 1 \tag{31}$$

$$R_{rns} = R_{r,n,1} \quad \forall r, n \leq \sigma, s \neq 1 \tag{32}$$

$$U_{rns} = U_{r,n,1} \quad \forall r, n \leq \sigma, s \neq 1 \tag{33}$$

$$V_{rns} = V_{r,n,1} \quad \forall r, n \leq \sigma, s \neq 1 \tag{34}$$

The objective function is given in Eq. 35, wherein ρ_s is the probability of scenario s.

$$\min \sum_{r \in \mathbf{R}^P} \sum_{s \in \mathbf{S}} \gamma^U_r \rho_s (\sum_{n \in \mathbf{N}^H} U_{rns} + U^T_{rs}) + \sum_{r \in \mathbf{R}} \sum_{s \in \mathbf{S}} \gamma^R_r \rho_s (\sum_{n \in \mathbf{N}^H} R_{rns} + R^T_{rs}) + \sum_{i \in \mathbf{I}} \sum_{s \in \mathbf{S}} \sum_{n \in \mathbf{N}^H} \gamma^X_i \rho_s \bar{X}^0_{ins} \tag{35}$$

Nomenclature

Indices/sets	
$i \in \mathbf{I}$	Tasks
$k \in \mathbf{K}$	Progress status
$n \in \mathbf{N}$	Time points or periods
$r \in \mathbf{R}$	Resources
$s \in \mathbf{S}$	Scenarios
$t \in \mathbf{T}$	Open-loop iterations
Subsets	
$\mathbf{N}^H$	Time points in open-loop horizon
$\mathbf{R}^P$	Product resources
$\mathbf{S}_n$	Scenarios at time point n
Parameters	

$\beta_i^{LB}/\beta_i^{UB}$	Lower/upper bound on batch size of task i
γ_r^R	Inventory cost for material resource r
γ_r^U	Backlog cost for product resource r
γ_i^X	Fixed cost of executing batch of task i
Δ	Re-optimization time-step
δ	Discretization of time grid
ϵ	Order size max-mean difference relative to mean
η	Observation horizon – time ahead for which the order sizes are deterministically known
Λ	Load
$\lambda_r^{LB}/\lambda_r^{UB}$	Lower/upper bound on level of resource r
μ_{irk}	Resource-task interactions of resource r with task i at status k independent of batch size
ν_{irk}	Resource-task interactions of resource r with task i at status k that depend on batch size
ξ_{rn}	Demand amount for product resource r at time point n
ρ_s	Probability of scenario s
τ_i	Processing time of task i in time periods
Ω	Time between orders
Binary variables	
X_{in}	Task start variable, equals 1, when task i starts at time point n
$\bar{X}_{in}^k$	Lifted task start variable, equals 1, when task i has status k at time n
Continuous variables	
B_{in}	Batch size of task i starting at time n
$\bar{B}_{in}^k$	Lifted batch size variables
R_{rn}	Level of resource r during period n

U_{rn}	Backlog level for product resource r during period n
V_{rn}	Shipment quantity of product resource r at time n

References

Andrés-Martínez, O. and Ricardez-Sandoval, L. A. (2022). A nested online scheduling and nonlinear model predictive control framework for multi-product continuous systems. *AIChE Journal*, 68(5):e17665.

Avadiappan, V. and Maravelias, C. T. (2021). State estimation in online batch production scheduling: concepts, definitions, algorithms and optimization models. *Computers & Chemical Engineering*, 146:107209.

Ave, G. D., Alici, M., Harjunkoski, I., and Engell, S. (2019). An explicit online resource-task network scheduling formulation to avoid scheduling nervousness. In Kiss, A. A., Zondervan, E., Lakerveld, R., and Özkan, L., editors, *29th European Symposium on Computer Aided Process Engineering*, volume 46 of *Computer Aided Chemical Engineering*, pages 61 – 66. Elsevier.

Balasubramanian, J. and Grossmann, I. E. (2004). Approximation to multistage stochastic optimization in multiperiod batch plant scheduling under demand uncertainty. *Industrial & Engineering Chemistry Research*, 43(14):3695–3713.

Baldea, M. and Harjunkoski, I. (2014). Integrated production scheduling and process control: A systematic review. *Computers & Chemical Engineering*, 71:377–390.

Birge, J. R. and Louveaux, F. (2011). *Introduction to Stochastic Programming*. Springer Publishing Company, Incorporated, 2nd edition.

Bonfill, A., Bagajewicz, M., Espuña, A., and Puigjaner, L. (2004). Risk management in the scheduling of batch plants under uncertain market demand. *Industrial & Engineering Chemistry Research*, 43(3):741–750.

Bonfill, A., Espuña, A., and Puigjaner, L. (2005). Addressing robustness in scheduling batch processes with uncertain operation times. *Industrial & Engineering Chemistry Research*, 44(5):1524–1534.

Bonfill, A., Espuña, A., and Puigjaner, L. (2008). Proactive approach to address the uncertainty in short-term scheduling. *Computers & Chemical Engineering*, 32(8):1689–1706.

Burnak, B., Diangelakis, N. A., Katz, J., and Pistikopoulos, E. N. (2019). Integrated process design, scheduling, and control using multiparametric programming. *Computers & Chemical Engineering*, 125:164–184.

Chu, Y. and You, F. (2014). Moving horizon approach of integrating scheduling and control for sequential batch processes. *AIChE Journal*, 60(5):1654–1671.

Cui, J. and Engell, S. (2010). Medium-term planning of a multiproduct batch plant under evolving multi-period multi-uncertainty by means of a moving horizon strategy. *Computers & Chemical Engineering*, 34(5):598 – 619. Selected Paper of Symposium ESCAPE 19, June 14-17, 2009, Krakow, Poland.

Dias, L. S. and Ierapetritou, M. G. (2016). Integration of scheduling and control under uncertainties: Review and challenges. *Chemical Engineering Research and Design*, 116:98–113. Process Systems Engineering - A Celebration in Professor Roger Sargent's 90th Year.

Engell, S. (2009). Uncertainty, decomposition and feedback in batch production scheduling. In Jeżowski, J. and Thullie, J., editors, *19th European Symposium on Computer Aided Process Engineering*, volume 26 of *Computer Aided Chemical Engineering*, pages 43–62. Elsevier.

Engell, S. and Harjunkoski, I. (2012). Optimal operation: Scheduling, advanced control and their integration. *Computers & Chemical Engineering*, 47:121–133. FOCAPO 2012.

Gupta, D. and Maravelias, C. T. (2016). On deterministic online scheduling: Major considerations, paradoxes and remedies. *Computers & Chemical Engineering*, 94(2):312–330.

Gupta, D. and Maravelias, C. T. (2019). On the design of online production scheduling algorithms. *Computers & Chemical Engineering*, 129:106517.

Gupta, D. and Maravelias, C. T. (2020). Framework for studying online production scheduling under endogenous uncertainty. *Computers & Chemical Engineering*, 135:106670.

Gupta, D., Maravelias, C. T., and Wassick, J. M. (2016). From rescheduling to online scheduling. *Chemical Engineering Research and Design*, 116:83 – 97. Process Systems Engineering - A Celebration in Professor Roger Sargent's 90th Year.

Harjunkoski, I., Maravelias, C. T., Bongers, P., Castro, P. M., Engell, S., Grossmann, I. E., Hooker, J., Méndez, C., Sand, G., and Wassick, J. (2014). Scope for industrial applications of production scheduling models and solution methods. *Computers & Chemical Engineering*, 62:161 – 193.

Høyland, K. and Wallace, S. (2001). Generating scenario trees for multistage decision problems. *Management Science*, 47:295–307.

Janak, S. L., Floudas, C. A., Kallrath, J., and Vormbrock, N. (2006). Production scheduling of a large-scale industrial batch plant. ii. reactive scheduling. *Industrial & Engineering Chemistry Research*, 45(25):8253–8269.

Kondili, E., Pantelides, C., and Sargent, R. (1993). A general algorithm for short-term scheduling of batch operations—i. milp formulation. *Computers & Chemical Engineering*, 17(2):211 – 227. An International Journal of Computer Applications in Chemical Enginnering.

Kopanos, G. M. and Pistikopoulos, E. N. (2014). Reactive scheduling by a multiparametric programming rolling horizon framework: A case of a network of combined heat and power units. *Industrial & Engineering Chemistry Research*, 53(11):4366–4386.

Kumar, R., Jalving, J., Wenzel, M. J., Ellis, M. J., ElBsat, M. N., Drees, K. H., and Zavala, V. M. (2019). Benchmarking stochastic and deterministic mpc: A case study in stationary battery systems. *AIChE Journal*, 65(7):e16551.

Lappas, N. H. and Gounaris, C. E. (2016). Multi-stage adjustable robust optimization for process scheduling under uncertainty. *AIChE Journal*, 62(5):1646–1667.

Li, Z. and Ierapetritou, M. (2008). Process scheduling under uncertainty: Review and challenges. *Computers & Chemical Engineering*, 32(4):715 – 727. Festschrift devoted to Rex Reklaitis on his 65th Birthday.

Lin, X., Janak, S. L., and Floudas, C. A. (2004). A new robust optimization approach for scheduling under uncertainty:: I. bounded uncertainty. *Computers & Chemical Engineering*, 28(6):1069–1085. FOCAPO 2003 Special issue.

Maravelias, C. (2021). *Chemical Production Scheduling: Mixed-Integer Programming Models and Methods*. Cambridge University Press.

Maravelias, C. T. (2012). General framework and modeling approach classification for chemical production scheduling. *AIChE Journal*, 58(6):1812–1828.

Mathur, P., Swartz, C. L., Zyngier, D., and Welt, F. (2020). Uncertainty management via online scheduling for optimal short-term operation of cascaded hydropower systems. *Computers & Chemical Engineering*, 134:106677.

McAllister, R. D., Rawlings, J. B., and Maravelias, C. T. (2020). Rescheduling penalties for economic model predictive control and closed-loop scheduling. *Industrial & Engineering Chemistry Research*, 59(6):2214–2228.

McAllister, R. D., Rawlings, J. B., and Maravelias, C. T. (2021). The inherent robustness of closed-loop scheduling. Computers and Chemical Engineering. George Stephanopoulos issue, under review.

Moradi, S. and MirHassani, S. (2016). Robust scheduling for multi-product pipelines under demand uncertainty. *The International Journal of Advanced Manufacturing Technology*, 87(9):2541–2549.

Méndez, C. A. and Cerdá, J. (2003). Dynamic scheduling in multiproduct batch plants. *Computers & Chemical Engineering*, 27(8):1247 – 1259. 2nd Pan American Workshop in Process Systems Engineering.

Méndez, C. A., Cerdá, J., Grossmann, I. E., Harjunkoski, I., and Fahl, M. (2006). State-of-the-art review of optimization methods for short-term scheduling of batch processes. *Computers & Chemical Engineering*, 30(6):913–946.

Nie, Y., Biegler, L. T., Wassick, J. M., and Villa, C. M. (2014). Extended discrete-time resource task network formulation for the reactive scheduling of a mixed batch/continuous process. *Industrial & Engineering Chemistry Research*, 53(44):17112–17123.

Novas, J. M. and Henning, G. P. (2010). Reactive scheduling framework based on domain knowledge and constraint programming. *Computers & Chemical Engineering*, 34(12):2129 – 2148. 10th International Symposium on Process Systems Engineering, Salvador, Bahia, Brasil, 16-20 August 2009.

Pantelides, C. (1994). Unified frameworks for optimal process planning and scheduling. Proceedings on the second conference on foundations of computer aided operations, pages 253–274.

Pattison, R. C., Touretzky, C. R., Harjunkoski, I., and Baldea, M. (2017). Moving horizon closed-loop production scheduling using dynamic process models. *AIChE Journal*, 63(2):639–651.

Petkov, S. B. and Maranas, C. D. (1997). Multiperiod planning and scheduling of multiproduct batch plants under demand uncertainty. *Industrial & Engineering Chemistry Research*, 36(11):4864–4881.

Rawlings, B. C., Avadiappan, V., Lafortune, S., Maravelias, C. T., and Wassick, J. M. (2019). Incorporating automation logic in online chemical production scheduling. *Computers & Chemical Engineering*, 128:201 – 215.

Remigio, J. E. and Swartz, C. L. (2020). Production scheduling in dynamic real-time optimization with closed-loop prediction. *Journal of Process Control*, 89:95–107.

Risbeck, M. J., Maravelias, C. T., and Rawlings, J. B. (2019). Unification of closed-loop scheduling and control: State-space formulations, terminal constraints, and nominal theoretical properties. *Computers & Chemical Engineering*, 129:106496.

Sand, G. and Engell, S. (2004). Modeling and solving real-time scheduling problems by stochastic integer programming. *Computers & Chemical Engineering*, 28:1087–1103.

Silvente, J., Kopanos, G. M., Pistikopoulos, E. N., and Espuña, A. (2015). A rolling horizon optimization framework for the simultaneous energy supply and demand planning in microgrids. *Applied Energy*, 155:485–501.

Subramanian, K., Maravelias, C. T., and Rawlings, J. B. (2012). A state-space model for chemical production scheduling. *Computers & Chemical Engineering*, 47(20):97–110.

Sundaramoorthy, A. and Maravelias, C. (2011). Computational study of network-based mixed-integer programming approaches for chemical production scheduling. *Industrial & Engineering Chemistry Research*, 50.

Touretzky, C. R., Harjunkoski, I., and Baldea, M. (2017). Dynamic models and fault diagnosis-based triggers for closed-loop scheduling. *AIChE Journal*, 63(6):1959–1973.

Velez, S. and Maravelias, C. T. (2014). Advances in mixed-integer programming methods for chemical production scheduling. *Annual Review of Chemical and Biomolecular Engineering*, 5(1):97–121. PMID: 24910915.

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

Vin, J. P. and Ierapetritou, M. G. (2001). Robust short-term scheduling of multiproduct batch plants under demand uncertainty. *Industrial & Engineering Chemistry Research*, 40(21):4543–4554.

Wonnacott, T. and Wonnacott, R. (1984). *Introductory Statistics for Business and Economics*. Wiley.

Zhuge, J. and Ierapetritou, M. G. (2012). Integration of scheduling and control with closed loop implementation. *Industrial & Engineering Chemistry Research*, 51(25):8550–8565.