Scheduling process operations under uncertainty and integration with long term planning

by

Kavitha G. Menon

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Chemical Engineering

Waterloo, Ontario, Canada, 2022

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner:	Zukui Li Associate Professor, Dept. of Chemical & Materials Engineering University of Alberta
Supervisor(s):	Luis A Ricardez-Sandoval Associate Professor, Dept. of Chemical Engineering University of Waterloo Ricardo Fukasawa Associate Professor, Dept. of Combinatorics & Optimization University of Waterloo
Internal Member(s):	Hector Budman Professor, Dept. of Chemical Engineering University of Waterloo Nasser Mohieddin Abukhdeir Associate Professor, Dept. of Chemical Engineering University of Waterloo
Internal-External Member:	Fatma Gzara Professor, Dept. of Management Sciences University of Waterloo

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

This thesis centers upon the application of mathematical modelling, optimization theory and uncertainty analysis to the problem of scheduling batch operations for large scale industries. Over the years, decision making strategies such as scheduling, that deals with allocation of plant resources, has been widely adopted by industries to efficiently carry out their operations and achieve the desired targets. In this thesis, the focus is on scheduling in the context of *multijob, multitasking* batch plants. This class of scheduling problems are of practical importance, specially in the analytical services sector, where effective scheduling models could increase the efficiency in carrying out the plant operations and may lead to increased throughput, or reduced makespan, resulting in greater profits or customer satisfaction.

One key factor that needs to be accounted while developing scheduling models are uncertain parameters. Although scheduling has been a widely studied area, scheduling under uncertainty remains a challenging topic due to the complexities that arise when uncertain parameters are considered, particularly when these uncertainties are endogenous that are dependent on model decisions. Existing stochastic approaches that accounts for such decision dependent uncertainties often involves introduction of auxiliary binary variables to enforce non-anticipativity and results in large intractable models that requires decomposition/relaxation methods to handle the tractability issues. In order to address the above issue, in this thesis, studies were conducted on developing stochastic approaches that account for endogenous uncertainties without using auxiliary binary variables. A novel two-stage scenario based stochastic approach was developed for scheduling of batch operations under an endogenous uncertainty without using auxiliary binary variables or explicit non-anticipativity constraints. The proposed stochastic model are presented in this thesis along with the proof that shows careful formulation of the constraints enables implicit non-anticipativity enforcement in the proposed approach. In order to ensure that the model can capture the actual industrial setting more accurately, the two-stage approach was modified and a node-based multistage stochastic approach was developed that does not require auxiliary binary variables while also allowing multiple realizations for the uncertain parameter through out the scheduling time horizon. The proposed approach was validated using multiple case studies including an actual industrial case study and two case studies from the literature. The computational studies conducted using the case studies depicts significant benefits in terms of the value of stochastic solution (VSS). A comparison study was also conducted between the node-based multistage approach and a conventional binary variable approach from the literature. The results from the study shows upto 85% reduction in computational time when using the proposed node-based approach.

Along with scheduling of daily operations, a widely adapted policy by industries to increase the efficiency of plant operations is developing long term strategies such as operational planning that focuses on optimizing the long term objective. Operational planning and short term scheduling are interrelated activities. The decisions from such long term planning models can be used to guide the scheduling of daily operations. However, due to their disparate time scales and resulting complexities, the interaction between these decision making levels remains a challenging problem. Hence, one of the focus areas of this thesis includes developing an iterative integration framework for effective interaction of planning and scheduling models for a large scale *multijob* batch plant. The effective integration of a planning and scheduling model greatly depends on a planning model that easily interfaces with the scheduling model and provide it with the required input parameters. However, due to the longer time horizons, the planning models often consider various aggregation schemes and ignore detailed plant specifications such as the sequence effects of tasks which results in planning decisions that are not achievable by the scheduling model. Considering these limitations in the literature, a long term planning model was developed for a multijob batch plant that considers approximated sequence constraints and provides key planning decisions to the scheduling model. The study further proposes a calibration scheme to ensure that the estimated information used in the planning model are reasonable and an iterative framework involving rolling horizon method to solve the integrated planning and scheduling models. The proposed framework was validated using an actual industrial case study and the computational results show an average increase of 8.27% in terms of profit when the models are integrated via rolling horizon (RH) approach.

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my supervisors Prof. Luis A. Ricardez-Sandoval and Prof. Ricardo Fukasawa for their constant support, patience and guidance throughout my PhD studies. Luis and Ricardo, a thank you would not suffice how much I appreciate and value your guidance and support, even then, a heart felt thank you to both of you!

I would like to thank my examining committee for taking the time to review my thesis and providing valuable feedback.

I would like to thank NSERC, MITACS and our industrial partner from the analytical services sector for the financial support throughout my PhD studies.

I would also like to thank to our grad coordinator, Judy Caron for all the help and support she has provided over years.

I would like to thank my colleagues and friends at University of Waterloo for helping me through the studies and to get through some of the tougher times. In particular, Dr. Mahshad Valipour, Dr. Yue Yuan, Dr. Grigoriy Kimaev, Oscar Palma Flores, Donovan Chaffart, Dr. Mina Rafiei, Yael Izamal Valdez Navarro, Ilse Mariana Cerrillo Briones, Huabei You, Han Wang, Manuel Iglesias, Gabriel Patron, Yanshuo Peng, Sagar Patel, Yashdeep Jadeja, Mohab Ahmed, Milad, Madhuja Chakraborthy, Jannat, Anushka.

I would like to thank my friends Shanoja, Sunu, Kichu and Jintu for making my life easier in Canada.

I would like to thank my dearest friends Sushmita, Anjali, Haritha, Nandaja and Kedarnath for being my constant cheerleaders and confidantes for the longest of times.

Finally, I would like to thank my husband, my parents, my in-laws and my brother for being my pillars of strength and for all the love and support they always shower on me.
I thank my loving husband for being so understanding and supportive through out my PhD studies. Unni, thank you for being in my life. Though it may sound impossible to most people, we graced well through a 4 year long distance marriage, even when the pandemic kept us from meeting each other for over one and a half years. I'm lucky to have found you.
I thank my parents for their unconditional love and support, for believing in me more than I believe in myself & for being my forever confidence boosters.
I thank my brother for being my number one well wisher and for encouraging me to dream high.
I thank my in-laws for their immense love and for supporting me throughout every step I

take.

Without the love,support and encouragement of all these people, it would have been impossible to survive the pandemic years. Thank you all from the bottom of my heart!

Dedication

To my parents and my husband

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Scheduling is one of the decision making strategies that deals with decisions involving how and when to execute operations to optimize a chosen objective such as maximizing profits or minimizing costs, subject to operational constraints such as resource limitations or demands to be met. Over the years, scheduling based on mathematical optimization has been widely adopted by industries to efficiently carryout their operations and achieve the desired targets [5, 6, 7, 8]. Proper scheduling can greatly increase the efficiency of a production plant and therefore is of great practical importance. Due to its wide scope, developing optimization models for various scheduling applications has been a topic of interest in process systems engineering and operations research for multiple decades now. In this thesis, our focus is in scheduling in the context of a *multijob, multitasking* batch plant from the analytical services sector.

The analytical services sector is focused on carrying out analyses on samples that are ordered by clients for various purposes; for example, performing a nutritional analysis on a food item to create the nutritional facts panel before bringing the product to market, or performing air quality analyses to check for hazardous materials such as asbestos, or conducting analysis on samples to determine its properties and chemical composition for

mining industries. Processing plants in the analytical services sector may receive in the order of thousands of samples on a daily basis to be processed at their plant and as such require a suitable method of scheduling operations.

A *multijob* batch plant is similar to a *multiproduct* batch plant that refers to industrial plants capable of manufacturing multiple products simultaneously where each product may have a specific recipe (sequence of tasks) to follow; for e.g., chemical industries. What distinguishes a *multijob* plant is that they not only focus on fixed products and the recipes to follow may vary with the customer specifications; for example, analytical services industries (ASI), where each client order can be translated to a job with a set of samples that needs to be processed through a sequence of tasks chosen by the clients. Jobs arrive at the *multijob* plant and each job consists of a set of *samples* that needs to be processed through a sequence of tasks, referred to as *paths*. Moreover, such *multijob* batch plants often possess additional operational features such as *multitasking* where machines are able to process multiple samples from multiple jobs simultaneously. The goal is to generate a schedule for the plant, which dictates what samples to assign to which processes over the length of time that is to be scheduled such that an objective is optimized, while abiding by the operational constraints of the problem. For such industrial plants, an efficient scheduling model that can provide decisions to effectively utilize all the available resources and process large number of jobs in a timely manner could result in substantial increase in the plant efficiency. However, the studies that consider scheduling of such *multijob, multitasking* plants are quite limited in the literature [9, 2, 10, 11] and are worth exploring.

While developing scheduling models, it is highly unlikely that all the operational parameters are known *a priori* or remain constant through out the operational period. In order to obtain more realistic and practical solutions, it is important to account for these uncertainties in the scheduling model. Accounting for uncertainties in such process networks would increase the modelling challenges, but it will also result in more realistic solutions [12, 13].

One of the common sources of uncertainty in analytical services industries (ASI) is *task outcome*. For a process plant from an ASI sector, quality of analyses is of utmost importance. When a large process plant capable of processing thousands of samples on a

daily basis is considered, it is highly unlikely that all available resources function perfectly without any factors affecting its performance and provides a perfect outcome throughout the operation period. Many factors such as resource malfunction, fluctuations in process parameters, human intervention etc. could affect the performance of the tasks and may require some of the samples to repeat the whole/subset of its *path*. This possible fluctuation in the *task outcome* could be accounted in the scheduling model as an uncertain parameter. However, accounting for uncertainties such as task outcome comes with additional set of challenges as they fall under the category of endogenous uncertainties or decision dependent uncertainties. An outcome of a task will only be realized if and when the model decides to process that task. Accounting for such decision dependent uncertainties often result in complex models that are computationally intensive. For instance, accounting for such decision dependent uncertainties using the modelling approaches such as stochastic programming [14, 15] results in a disjunctive scheduling model which are further linearized using auxiliary binary variables [16]. Due to the introduction of binary variables and increased model size, it becomes difficult to solve the model directly; previous studies have thus focused on the development of model size reduction methods or employed relaxation or decomposition techniques to solve the otherwise intractable models [17, 18, 19, 20].

To the best of author's knowledge, there are no studies available in the literature that considers a stochastic programming approach which does not involve binary variables to model endogenous uncertainties. Therefore, one of the main focuses in this thesis is to address this gap by developing a novel stochastic approach for scheduling of batch operations for a *multijob, multitasking* plant while accounting for endogenous uncertainties such as *task outcome* without any auxiliary binary variables.

Along with scheduling of daily operations, a widely adapted policy by industries to increase the efficiency of plant operations is developing long term strategies that focuses on optimizing the long term objective, referred to as planning [21]. While scheduling deals with short term day to day operations, planning deals with longer time horizons spanning over weeks or months or even years. If time horizons that span over weeks/months are considered, in addition to the decisions that effect the daily operations, the model would allow making decisions that could be strategic for the industrial plant in the long run. In

order to make such long term decisions, in addition to the information that are key to the daily operations, other information needs to be considered. For e.g., any weekly/monthly demands or processing requirements that should be met or if the plant consists of processes that involve labour efforts, then the number of workers hired/available to work could also play a key role in the functioning of the plant and meeting the targets. Moreover, with such long term strategies, machine maintenance can also be strategically performed considering the processing demands. In order to obtain the most accurate decisions, the planning model should ideally account for all the information including the detailed specifications required for scheduling of daily operations and the additional high level information such as those mentioned above. However, when large scale industries are considered, scheduling daily operations itself may result in models with hundreds of thousands of variables and constraints. Hence, while accounting for longer time horizons, the common practice include developing a planning model that comprises the high level information and aggregated schemes to account for the plant specifications and a scheduling model that comprises the detailed plant specifications [1]. With these models, the goal is to obtain a long term plan that can be considered as the basis for the plant operations over the planning horizon considered. Once there is a plan, the next step would involve obtaining the daily plant operations via scheduling that attempts to follow the plan and help in achieving the planning targets over the longer horizon.

The major drawback when the planning and scheduling models are solved and studied separately is that the planning model, due to the aggregated schemes, tend to make inaccurate estimations of the capacity of the plant and result in sub optimal decisions and planning targets that cannot be achieved by the scheduling model [22]. In order to address this drawback, and ensure that the planning decisions are reasonable and achievable, adopting an efficient scheme to modify the planning decisions based on the scheduling model components is necessary; this is referred to as integration of planning and scheduling.

Several studies have been carried out in the literature to develop planning models for various applications. However, planning studies focusing on *multijob* batch plants from ASI sector where the job recipes are highly dependent on the customer orders are lacking from the literature. In addition, depending on the aggregation schemes employed,

most planning studies in the literature provide weekly/monthly decisions and studies on operational planning models that can provide daily processing targets or decisions to guide the scheduling operations are limited in the literature [23]. In this study, the aim is to address this gap and develop an operational planning model for large scale *multijob* batch plants that can provide daily operational decisions/targets, which can be further used to guide the scheduling model decisions. The study further extends to proposing an integration scheme that iteratively modifies the planning model in order to ensure that the planning decisions are reasonable and achievable.

1.2 Research Objectives

In order to address the challenges and shortcomings in the literature mentioned above, the current PhD study focuses on the following research objectives:

- Develop a novel two-stage stochastic programming approach to schedule batch operations under an endogenous uncertainty that does not involve using auxiliary binary variables.
- Investigate the challenges associated with expanding the two-stage stochastic approach to a multistage approach and develop a multistage stochastic approach that offers wider scope and flexibility in accounting for endogenous uncertainties.
- Develop a new long term planning model that considers sequencing effects and can provide key planning decisions including the daily processing targets and number of required workers to the scheduling model.
- Develop an iterative framework for integrating the planning and scheduling models for a multijob multitasking large scale industrial plant.

1.3 Expected Contributions

The work conducted in this thesis is expected to have the following contributions.

- Provide insights to the challenges in developing stochastic scheduling models that accounts for decision dependent (endogenous) uncertainties.

- Propose a novel two stage stochastic approach to account for an endogenous uncertainty for a *multijob* batch plant. The key contribution here is that the proposed approach does not involve introduction of auxiliary binary variables and constraints unlike the stochastic studies available in literature and therefore reduces the computational requirements.

- Propose a multistage approach to account for endogenous uncertainties that offers more flexibility and accuracy towards capturing uncertainty in actual industrial settings.

  A two-stage and multistage approach that does not require auxiliary binary variables and constraints would be computationally promising for large scale industrial problems and could be adapted to solve a class of scheduling problems under endogenous uncertainty commonly encountered in analytical or chemical industries.

- Present a novel operational planning model for large scale *multijob* batch plants that could provide a plan to be followed in order to achieve the long term objective. The key contribution of this study is that it considers large scale *multijob* batch plants and takes into account the job sequence effects to provide the key planning decisions including daily processing targets and required workers through out the planning horizon.

- Present iterative integration schemes to effectively integrate the long term planning and short term scheduling models to ensure that the planning model provides reasonable decisions to the scheduling model and to ensure that it consists of the capabilities to account for the variations in the job arrival.

With such a long term operational planning model and integration schemes, the operational efficiency of the industrial plants can be improved substantially.

1.4 Thesis Structure

This PhD thesis is organized as follows:

Chapter 2 provides the background information and literature review on the key topics of this thesis including the type of uncertainty, uncertainty modelling techniques, integration techniques for planning and scheduling models. The gaps in the literature and the motivation for this research is discussed in detail in this section.

Chapter 3 presents the novel two-stage stochastic programming approach developed for modelling endogenous uncertainty. The key novelties of the proposed approach are also discussed in this chapter. The findings and the contributions of this study has been published in *Annals of Operations Research*[24].

Chapter 4 presents the node-based multistage approach for modelling endogenous uncertainties. The novelties and advantages of the approach are presented in this chapter along with the results from multiple applications. This chapter is based on the manuscript *A novel multistage stochastic programming approach for short-term scheduling of batch processes under type II endogenous uncertainty* by Kavitha G.Menon, Luis A. Ricardez-Sandoval and Ricardo Fukasawa, which has been submitted to *INFORMS Journal on Optimization* and is currently under review.

Chapter 5 presents the developed operational planning model for large scale multi-job batch plants. This chapter also presents iterative integration schemes to effectively integrate the long term planning and short term scheduling models to ensure that the planning model provides reasonable and achievable decisions to the scheduling model.

Chapter 6 summarizes the research contributions of this thesis and also provides recommendations for potential future work in this area.

Chapter 2

Background & Literature review

Over the last few decades, a large number of studies have been carried out on developing efficient scheduling models and incorporating various aspects such as accounting for uncertainty and integrating with planning model for increasing the efficiency of large scale operations. Due to these studies and the resulting progress, most of the process industries today depend of such decision making strategies to effectively carry out their operations. In this chapter, the review of those studies that are relevant to the focus areas of this thesis are presented.

One of the fundamental aspects to be considered when developing scheduling models is the choice of the time representation. Hence, this chapter begins with a discussion on the time representation used in this study, followed by other relevant topics such as uncertainty modelling technique employed, the type of uncertainty accounted and the integration scheme adapted in the current study.

This chapter is structured as follows: Section 2.1 provides a brief overview on the time representation used in the scheduling model. Section 2.2 provides the background and review on the uncertainty modelling technique and the type of uncertainty considered in this thesis. Section 2.3 provides the background and literature review on the planning models for large scale operations and section 2.4 provides the background and literature review on the integration frameworks considered in the literature for enabling interaction

between the scheduling and planning models .

2.1 Time Representation

The time representation determines when operations may be scheduled, and can play a large role in determining the computational cost of solving the model and the final solution quality [25, 26]. Each point in time where an operation may be scheduled can be referred to as a timepoint.Time representation can be generally classified into two: continuous time representation and discrete time representation [27].

In continuous approach, the scheduling decisions occur at precise points in time during the scheduling horizon and the model itself determines where these points should be placed [28, 2]. Since the model is able to choose where the timepoints should be placed, the continuous representation may provide the best solutions [25]. However, the modeller must provide the model with a fixed number of timepoints to allocate as an input and the solution quality may vary with the number of timepoints. With fewer timepoints, the solution quality may decrease and with higher timepoints, the computational cost may drastically increase. Overall, selecting a suitable number of timepoints for the model can be challenging.

In the discrete representation models, the time horizon is discretized into a number of time points with the greatest common divisor of all processing times being the discretization interval (uniform discretization) [29]. Even though a discrete formulation will restrict any decision to be made only at pre-determined time points, the usage of very fine discretization interval can provide high quality solutions; however, this would increase the model size and can result in very high computational time. To overcome the above mentioned drawback of discrete representation, a non-uniform time discretization (NUD) was developed by the authors of [4]. Using this approach, events can occur at different times for different tasks, effectively allowing the model to make decisions for a particular task without adding unnecessary time points for the rest of the tasks. Figure 2.1 presents an illustration of uniform and non-uniform time discretization. While uniform discretization

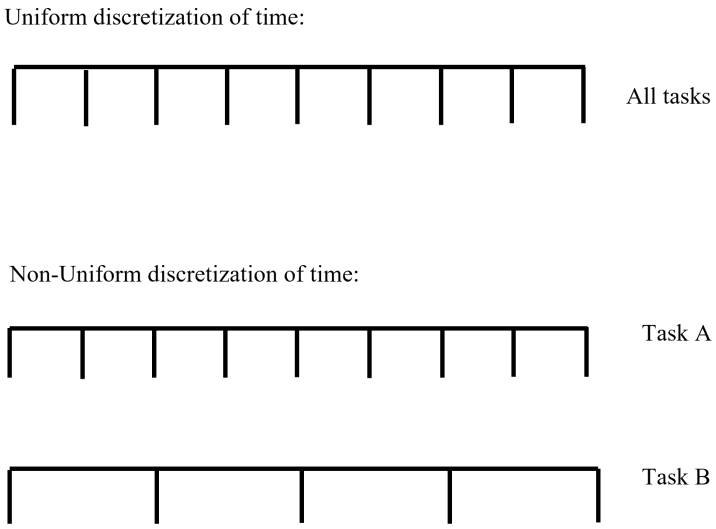


Figure 2.1: Illustration of uniform and non-uniform time discretization

assign same number of time points for all the tasks in the process network, non-uniform time discretization enables different discretization for different tasks.

The authors of [10] conducted a comparison study of the non-uniform discretization approach and the continuous approach for a multijob multitasking batch plant and observed that former representation obtained better trade off between the quality of solutions and the computational time compared to the latter one. Since the current study also considers a multijob multitasking batch plant that follows similar characteristics as that considered in [10], a non-uniform time discretization is adopted for the scheduling purposes as it was considered in [10].

2.2 Uncertainty

Uncertainty is often an inherent feature of many of the systems we aim to optimize. Some of the factors, both internal and external, such as market demands, equipment malfunction, task processing time, process parameters, product yield etc. may be subject to uncertainty. These parameters could often result in suboptimal or even infeasible solutions if they are

assumed constant or when modelled deterministically using their nominal values. Hence, accounting for parameter uncertainty is one of the important and challenging aspects of industrial process modelling [13, 30, 31].

In order to better explain the challenges associated with accounting for decision dependent uncertainties, in the following discussion, the modelling techniques widely used to account for the uncertain parameters are discussed first, followed by the discussion on the features of the uncertain parameter considered in this study.

2.2.1 Modelling Approaches

Many of the uncertain parameters that are commonly encountered in the process industries can be explicitly taken into account using the preventive approaches such as robust optimization techniques, and two-stage/ multistage stochastic programming approaches [32].

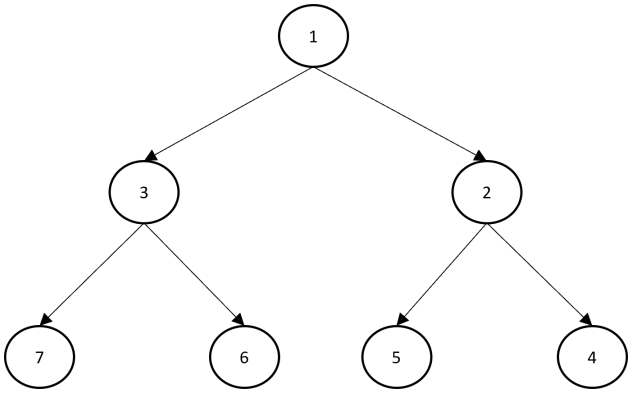
In a robust optimization (RO) approach, the worst case is optimized while guaranteeing feasibility for all possible realizations of the uncertainty defined by the uncertainty set [33, 34]. Multiple studies have developed robust optimization models to account for different uncertain parameters such as demands, processing time etc. in their model [35],[36]. The advantage of RO technique is that it guarantees solution feasibility for any value of uncertainty within the defined set whereas the downside of this approach is that it results in a conservative solution and does not allow any recourse action after the realization of uncertainty. To alleviate this set back of the RO approaches, the study in [37] introduced the concept of adjustable robust optimization (ARO), which include recourse in the form of affine decision rules that are functions of uncertain parameters. The studies in [38] and [39] applied ARO approach in production scheduling applications and reported higher benefits in comparison to the traditional RO approaches. However, the downside of ARO is that it does not allow adjusting those decision variables that are directly multiplied by the uncertain parameters, as this leads to bi-linearity in the system [40]. This feature of ARO affects the modelling efficiency and thereby limits the application of the approach.

Stochastic programming is another widely used approach to model uncertain parameters. When the uncertain parameters are assumed to be discrete with a known probability distribution, stochastic programming provides a solution that is feasible for all possible parameter realizations while optimizing the given objective function [14],[15]. One of the relevant advantages of this approach is the flexibility it offers in the decision-making process. In a stochastic programming approach, the modeler can fix a set of decisions (*here and now*) before the realization of the uncertain parameters, and upon the realization of uncertainty, a set of decisions (*wait and see*) can be made as a corrective/recourse action. In a two-stage stochastic approach, one set of *here and now (first-stage)* decisions are made before the realization of the uncertain parameters, and, upon the realization of uncertainty, *wait and see (second stage)* decisions are made as a corrective/recourse action.

In a multistage stochastic programming approach, uncertainty can be realized multiple times throughout the time horizon. Hence, in this approach, apart from the here and now decisions made at the beginning of the time horizon, recourse decisions can be made at different stages according to the sequence in which the uncertainty is revealed. These uncertainty realizations at each stage can be represented by a set of finite nodes. The complete sequential realization of the uncertain parameter in a multistage approach can be represented as a scenario tree of realizations. Figure 2.2 shows the scenario tree representation for a three-stage model, where the time horizon is divided into two time-periods (a time-period is when some uncertainty gets realized) and the uncertainty realizations for each time-period are represented as nodes. Root node represents the first stage decisions independent of any value of realization. The nodes 2 and 3 represent the possible uncertainty realizations in the first time-period, nodes 4 to 7 represent the possible realizations in the second time-period. Thus, if the time horizon is divided into two time-periods, it results in a three-stage decision-making process including the here and now (first stage) decisions and the second and third stage decisions made successively after the uncertainty realization in the corresponding time-period. Accordingly, for a scheduling horizon with M time-periods, the problem can be formulated as an $M + 1$ stage stochastic model.

One of the common stochastic linear programming (SLP) representation include formulating the problem for every possible scenario and adding constraints to ensure the

Figure 2.2: Scenario tree representation for a Three stage model



information structure associated with the decision process is honored [41]. The general representation of a scenario based two-stage SLP can be represented as follows:

$$Min \sum_{s \in S} (cx_s + g_s y_s) p_s \tag{2.1}$$

$$s.t. \qquad T_s x_s + W_s y_s \geq r_s \quad \forall s \in S \tag{2.2}$$

$$x_s - x = 0 \qquad \forall s \in S \tag{2.3}$$

$$x_s, y_s \geq 0 \quad \forall s \in S \tag{2.4}$$

where, x and x_s, y_s represents the first and second stage decisions respectively, T_s and W_s represents the coefficient matrices while r_s represents the right hand side vector. The second stage decision variables may vary with respect to the value of realization s but the first stage decisions that are independent of any realization cannot vary. Hence, additional constraints are defined that ensures that these first stage decisions remain the same for every scenario s through constraint (2.3). These constraints are known as non-anticipativity constraints (NACs). These NACs ensure that the solutions obtained are implementable, i.e., the actions that must be taken at any point in time depend only on information that is available at that time.

Several studies have been conducted in the literature that utilizes the multistage stochastic programming approach for modelling the uncertain parameters in scheduling models [42, 43, 44, 45]. The complexity associated with accounting for uncertainty parameters vary with respect to the type of uncertainty involved in the problem. When the uncertainties involved are decision dependent in nature, there are multiple challenges that needs to be addressed and this will be discussed in detail in the next section.

2.2.2 Types of Uncertainty

Uncertainties are often classified as exogenous or endogenous [46]. Exogenous uncertainties are those that are independent of model decisions and only depend on external factors, e.g. market demands that vary with respect to the customer requirements. On the other hand, endogenous uncertainties are those that are dependent on the model decisions; these can be further classified as type I and type II endogenous uncertainties [16]. Type I endogenous uncertainties are those where underlying probability distributions are dependent on the model decisions, e.g. in a competitive market, a decision to increase the production can have a negative impact on the product prices [47, 48, 49]. Type II endogenous uncertainties are those where underlying probability distributions do not vary, but the time of uncertainty realizations are dependent on the model decisions. For instance, task outcome, which will be realized only if and when the model decides to operate that task; or the quality and quantity of oil in an oil field is a random variable whose value does not depend on any decision we make, but the precise time when the uncertainty is realized is not known *a priori* and represents the time when we decide to analyze that oil field [16]. Endogenous uncertainties are often more complex in structure and requires computationally intensive models to obtain reasonable solutions. Most of the available literature on stochastic programming (SP) have focused on exogenous uncertainties. A review of studies conducted on such areas can be found in [12] and [50]. Endogenous uncertainties have been considered in relatively few stochastic programming publications and among them, only a handful of studies have considered type II endogenous uncertainties. In this thesis, the focus is on modelling scheduling problems with type II endogenous uncertainty.

2.2.3 Type II Endogenous Uncertainty

One of the key factors of a stochastic programming approach is to ensure non-anticipativity i.e., decisions made at any point in time should be based on the information available at that point and without anticipating any information from the future. Defining NACs becomes very challenging when the uncertainties are model dependent and the time at which the uncertainty is realized is not known *a priori*. For any exogenous uncertainty, where the time of uncertainty realization is known *a priori*, non-anticipativity can be easily implemented by defining constraint (2.3) for all model decisions prior to the time of uncertainty realization. However, when the time of uncertainty realization is not known, it becomes challenging and results in a complex model. Such uncertain parameters result in conditional non-anticipativity constraints with disjunctions [51, 52] that needs to be reformulated. This challenge is commonly resolved in the literature by introducing auxiliary binary variables. For detailed discussions on the scaling of computational cost for various disjunctive formulations, see [51].

The study in [53] were one of the early studies to account for type II endogenous uncertainties. Those authors presented an approach using auxiliary binary variables to solve problems where the time of realization of uncertainty depends only on binary decision variables; case studies involving small and medium size instances were presented in their work. The authors of [46] presented a multi-stage approach with auxiliary binary variables for a study on oil and gas field reserves with uncertainty in the reservoir properties (size). Those authors proposed a decomposition-based solution strategy to solve a large model as a sequence of two stage stochastic programming problems. A generalized approach for problems with both exogenous and type II endogenous uncertainties using auxiliary binary variables was proposed in [16]. The authors also explored theoretical properties of the system to reduce the model size due to use of auxiliary binary variables and non-anticipativity constraints. These theoretical properties focus mainly on two aspects -1) eliminating the redundant NAC's 2) scenario reduction techniques to obtain the reduced set of scenario pairs for which the NAC's are required, in order to reduce the model size. Due to the challenges because of the additional binary variables involved in reformulating the

disjunctions in the conditional NACs, it often results in computationally-intensive models that cannot be solved directly [54, 46, 55]. Major drawbacks of introducing auxiliary binary variables and constraints linking them to the model decisions is that it results in large model sizes, which further grows substantially as the number of possible realizations of uncertain parameter (scenarios) grows. As a result, many studies have focused on developing solution strategies and decomposition approaches to solve those reformulated disjunctive models.

Several studies have also modelled type II endogenous uncertainties using auxiliary binary variables and explicit NACs. In order to cope with the additional burden of binary variables, these studies explored a number of theoretical properties and scenario reduction techniques. The authors of [18] explored more theoretical properties for the model presented by [16] to address the issue of exponential increase in the NACs with increase in the uncertainty realization. The authors further present solution strategies as the reduced models are still large to be solved directly. The study in [17] considers models with both exogenous and endogenous uncertainties and derive theoretical properties for reducing the scenarios and NACs. They further present solution approaches including sequential scenario decomposition and lagrangean decomposition to solve the problem instances. The authors of [56] presented an optimization model for the operations planning of an offshore oil and gas field infrastructure. Those authors consider co-relations among the uncertain parameters to reduce the dimensionality of the model and present a lagrangean decomposition method to solve the large instances. Also, they further presented new decomposition algorithms to solve similar problems in [57]. The works, [58] and [55] presented disjunctive models for synthesis of process networks and planning of offshore oil field infrastructure, respectively. The authors present different decomposition strategies to solve the model instances. Studies, [19],[59],[20] and [60] focuses on multistage stochastic formulation for the planning of clinical trials in pharmaceutical R&D pipeline under endogenous uncertainties. Through these studies, authors present various theoretical properties and solution approaches to solve the model instances. Although there are such handful of studies available in the literature that address the issue of uncertainties with decision-dependent time of realization, due to the explicit NACs and the introduction of auxiliary binary variables, it often results in large intractable models.

To the best of author's knowledge, most of the current studies available in the literature introduce auxiliary binary variables to determine the time of uncertainty realization and use them to define explicit non-anticipativity constraints and subsequently focus on developing various solution strategies. The only exception that the author is aware of is the work of [61] where binary variables already exist that can be used for that purpose. Despite its modelling and computational complexities, addressing such uncertainties are inevitable for developing efficient scheduling models [50]. Hence, in this thesis one of the first focuses is on developing a novel two-stage stochastic approach for scheduling of batch operations while accounting for type II endogenous uncertainties without any auxiliary binary variables or explicit NACs.

As two-stage approach only allows the uncertain parameter to have one value of realization through out the time horizon, this may be a limiting assumption for many industrial applications. In the actual industrial settings, the uncertain parameter may have multiple realizations in the considered time horizon. Hence, the second focus in this thesis is to study the challenges of expanding the two-stage approach to a multistage approach that allows multiple realizations for the uncertain parameter [62]. Eventually, the goal is to develop a novel multistage stochastic model for scheduling of batch operations under type II uncertainty that does not require introduction of auxiliary binary variables.

2.3 Planning

Planning is a key decision making strategy widely used in process industries. Depending upon the decisions involved, planning could be categorized as strategic planning or operational planning [23]. Strategic planning determines the long term direction of the industry by considering the changing market and industry needs. With regards to process industries, strategic planning involves time horizon in the order of years and decisions including the building of new sites, adding or removing raw material suppliers and so on. Operational planning, on the other hand, determines the required production rates or raw materials for the industrial plant under consideration. These production rates are subsequently used in

the scheduling level to determine/schedule the daily operations of the plant. In this thesis, the focus is on operational planning.

The primary objective of an operational planning model is to provide realizable production targets, and therefore, such model needs to take into account not only the customer orders/demands, but also the production capacity of the plant. One approach is to apply the scheduling model over the entire planning horizon since the scheduling model rigorously takes into account the production capacity of the plant. However, model tractability issues make this approach less viable. Hence, various types of aggregation or relaxation schemes have been adopted when formulating the operational planning models.

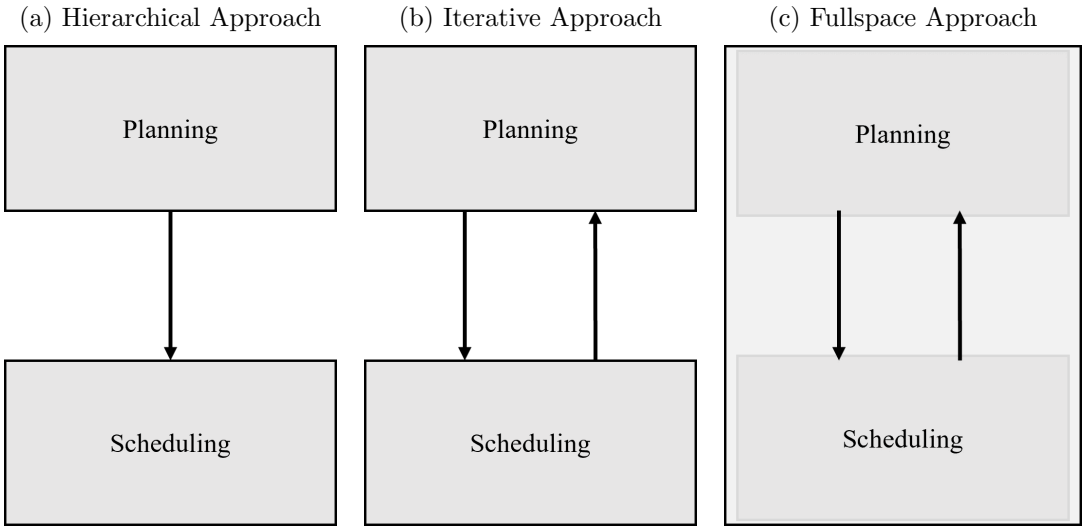
The aggregation schemes employed in the literature can be classified as time based aggregation schemes and unit based aggregation schemes. The authors of [63] proposed a planning model that utilizes aggregation schemes involving discretization of time horizon into time periods and aggregating the scheduling level constraints and variables into aggregate capacity, resource and variables. A similar time based aggregation scheme was adapted in studies [64],[65] and [66], where the authors discretized the planning horizon into commercial periods and then further discretized it into production periods and utilized the aggregate information. The authors of [67] also proposed a similar time based aggregation scheme where the time horizon was discretized into time periods and the decisions were only considered at the end of the time period. These time based aggregation schemes provide a tight upper bound on the production capacity of the plant but does not rigorously take the capacity into account. The study in [68] notes that a downside of the existing planning models is that it utilizes aggregate plant capacity to obtain aggregate production targets and fails to provide a daily production profile required by the scheduling model.

Another type of aggregation scheme proposed was the unit aggregation scheme which involved plant modelling by its set of bottleneck tasks [69, 68, 70]. However, in this approach the model disregards any sequencing effects, implying that these models assume that the bottleneck tasks can alone produce the final products without any upstream or downstream processing. Modelling the planning problems in such manner with only considering the bottleneck tasks would result in overestimating the true production capacity of the plant. The study in [69] noted that the most commonly used method in discrete

manufacturing industries is unit aggregation method. The authors of [70] proposed an operational planning with production disaggregation model that can supply daily production rates to the scheduling model. Those authors employ unit aggregation approach and estimate the capacity and processing time of an entire job recipe by adjusting the respective parameters of the bottleneck task involved in the job. These estimation methods maybe useful when the problem involves a multiproduct plant with fixed products or recipes, implying that the problem specification may involve multiple jobs that are designed to process the same set of task sequences. When the job recipes could be highly dependent on the customer order specifications, accounting for job sequence effects becomes more challenging and the existing estimates of unit aggregation methods become unreasonable. A long term planning model for such multijob batch plants that can provide the scheduling model with daily production profile is a topic that has not been addressed in the literature and is worth exploring (Note that as this thesis primarily focus on multijob batch plants from analytical services sector, where samples obtained from the clients are analysed/processed, the terms daily production profile and production targets may also be referred to as daily processing profile and processing targets respectively in the later sections of the thesis). Therefore, another focus of the thesis is to develop a long term planning model for a multijob batch plant that accounts for the job sequence effects and supply the scheduling model with daily processing targets.

Even with a planning model that considers the sequence effects and provides daily processing targets, it is likely that the obtained decisions may not accurately reflect the processing capacity of the plant, as planning models cannot rigorously account for the plant's processing capacity. Unless a scheme is adopted to refine the planning model constraints based on the key scheduling model components, the planning model may lead to suboptimal allocation of resources. Thus, integrating the planning and scheduling model has emerged as a key step to address the inaccuracies within the planning model by allowing interaction between the two models.

Figure 2.3: Schematic Representation of the solution techniques for integrating planning and scheduling models [1]



2.4 Integration of Planning and Scheduling

The challenge associated with the integration of the planning and scheduling lies in developing an efficient framework to enable the interaction between these models. Multiple studies are available in the literature where different authors propose different solution approaches to solve the planning and scheduling models, allowing various levels of interaction between the models [71, 72, 73]. The authors of [1] classifies these solution approaches into three - hierarchical approach, full space approach and iterative approach. The schematic representation of the approaches are presented in Figure 2.3.

The hierarchical approach provides a one way interaction where flow of information is from the planning model to the scheduling model. In this approach, the planning and scheduling models are integrated by providing the scheduling model with the processing targets obtained from the planning model with a goal of obtaining a scheduling solution [74, 75, 76]. Due to the one way interaction framework, this approach restricts the ability to modify the planning problem with respect to the actual scheduling model components.

Hence, the assumptions and approximations considered in the planning model could result in planning decisions that cannot be achieved by the scheduling model resulting in an infeasible scheduling problem. Next approach is the full space approach which involves solving the planning and scheduling model as a single problem where detailed scheduling sub problems are used for each planning period [77, 78, 79]. This approach often requires the problem to be decomposable into master and sub problems as it utilizes decomposition/relaxation methods [80, 81] to solve the otherwise computationally intractable problems.

The iterative approach provides a two way interaction between the planning and scheduling models [82]. The key decisions from planning model are provided to the scheduling model followed by a feed back from the scheduling model to the planning model in order to obtain better planning decisions. The authors of [83] proposed a bi-level decomposition algorithm for the simultaneous planning and scheduling of a multiproduct plant and the authors of [84] developed a bi-level model formulation using service level constraints to integrate the planning and scheduling models for multiproduct plants. The authors of studies, [68] and [70] proposed iterative framework for integrating an operational planning model and scheduling model for multiproduct batch plants using a rolling horizon (RH) approach. Rolling horizon algorithms were proposed by [85] based on the concept of separating the scheduling problem in a sequence of iterations, each of which models only part of the time horizon in detail, while the rest of the horizon is represented in an aggregate manner. Multiple studies in the literature employs similar rolling horizon schemes while developing long term planning/scheduling models or iterative frameworks for integrating the planning and scheduling models [86, 87, 88, 89]. As it requires fewer iterations to solve the problem, the rolling horizon method is generally efficient in the computational manner. However, the approach can only ensure the feasibility of the final solution and the quality of the solution depends on the estimates used in the planning problem. The authors of studies [69] and [87] propose methods to derive the feasible production capacity regions from the scheduling model and incorporate them to the planning problem which can be further utilized in the integration of multiproduct batch plants using RH approach. Although there are such iterative integration studies available in the literature, studies

focusing on large scale multijob batch plants where there are no fixed products and where the recipes are highly dependent on the job specifications are lacking from the literature.

Considering the above limitations, in this thesis, a long term operational planning model was developed for a multijob batch plant, followed by a two-step iterative integration framework involving the rolling horizon approach to enable effective interaction between the planning and scheduling models.

2.5 Chapter Summary

In this chapter, detailed discussions on key topics considered in this thesis were provided. Discussions on the time representation for scheduling operations and the type of uncertainties considered in this thesis were provided along with the uncertainty modelling techniques. The discussions point out that the studies available in the literature for modelling a type II endogenous uncertainty using a stochastic approach involves introduction of auxiliary binary variables and results in modelling and computational complexities. To the best of author's knowledge, there are no studies available in the literature that does not require auxiliary binary variables to model type II endogenous uncertainties using stochastic programming approach.

Further discussions on considering longer time horizons and developing an operational planning model indicates that there is a lack of such studies in the literature for multijob batch plants where there are no fixed products and where the job recipes are highly dependent on the customer specifications. The discussions are also provided on why integration of long term planning model with a detailed scheduling model is necessary and points out the lack of such studies conducted for large scale multijob batch plants. The gaps pointed above motivated the studies conducted in this thesis.

Chapter 3

Two Stage Stochastic Programming Approach

This chapter presents the proposed stochastic programming approach for scheduling of batch operations under type II endogenous uncertainties. The key novelty of the proposed approach is that it enforces non-anticipativity without using any auxiliary binary variables. To the best of author's knowledge, this is the first study that proposes a scenario based stochastic approach which does not require the utilization of auxiliary binary variables to model type II endogenous uncertainties. The proposed model is followed by the proof that demonstrates the implicit non-anticipativity enforcement of the proposed approach. The two-stage approach proposed in this study is considered as a positive step towards the direction of developing the multistage model (which can take into account multiple realizations of the uncertain parameter) with implicit non-anticipativity. The proposed framework is validated using an actual industrial case study from the analytical services sector.

This chapter is structured as follows: Section 3.1 provides the description of the deterministic problem followed by the deterministic scheduling model. Section 3.2 defines the uncertainty parameter and the modified process network. Section 3.3 provides the mathematical model and the subsequent proof that demonstrates implicit non-anticipativity

enforcement. Section 3.4 provides the computational experiments and the results and section 3.5 summarizes the findings and contributions of this chapter.

3.1 Deterministic Problem Description

Consider a process network with a set of tasks J and a set of jobs I. Each task $j \in J$ consists of a set of identical resources R_j that can perform the task and each resource has a capacity C_j. Every job $i \in I$ has to be processed through a sequence of q_i tasks known as a path, $P^i = \{P^i_1, P^i_2, \ldots\ldots P^i_{q_i}\}$. Each job $i \in I$ consists of A_i units that have to be processed sequentially through all the tasks in the path P^i. Note that the term units here is a generic term and could represent any materials, goods, samples and objects that need to be processed in a task. If a task j appears in the path of multiple jobs, a machine $b \in R_j$ can process units from different jobs simultaneously, given that the total number of units does not exceed the available capacity. This operational feature of simultaneously processing units from different jobs in a task is referred to as multitasking. The completion time required by a task j is represented by $\phi(j)$ and is considered a constant. Regarding the resource operations, the following assumptions are considered:

Assumption R1: There is no minimum working capacity for a resource, i.e. a resource can start processing any number of units between zero and C_j and that the transfer time between the tasks is negligible.

Assumption R2: The resources are non-preemptive; i.e., a resource cannot be interrupted while carrying out a task, this implies that once a resource starts processing a batch of units, the resource must operate until the completion time of the task without any interruption. Accordingly, units being processed in a resource cannot be removed while it is still in operation; similarly, new units cannot be added while the resource is in operation even if there is capacity available.

This can be considered as a variant of the job shop scheduling problem [90, 91, 92, 93, 94]. Similar to a job shop scheduling problem, in this problem there are a set of jobs (I) and a set of machines ($\cup_{j \in J} R_j$) and each job consist of a set of operations that needs to be

processed in order (path P^i). However,the current problem differs from the classical job shop problem in the following aspects:

- In the classical job shop scheduling, each machine can only process one unit (job) at a time, whereas, in the current problem, there are capacities and machines that can process multiple units from different jobs at the same time (multitasking), as long as capacities are respected.
- In classical job shop scheduling, each job consists of a single unit and in that sense, a job and its single unit are one and the same. That single unit needs to go through a sequence of tasks and it is finished whenever that single unit finishes all its tasks. In the current context, each job can be processed in multiple units and they don't need to go through the sequence of tasks at the same time. A job is only finished when all corresponding processing units finish all the required tasks.

Based on the above description of the process network, the goal is to generate a schedule for the plant, which dictates what units to assign to which processes over the length of time that is to be scheduled with an objective of maximizing the throughput. The solution should ensure that the units from each job $i \in I$ sequentially visits every task in its path P^i, such that each unit is processed at most by a single resource at a time and also the total sum of units processed by any resource at a time does not exceed its capacity C_j.

3.1.1 Process Network

The process network of tasks can be represented using a series of *directed graphs* consisting of vertices and directed edges [95]. As per definition, a *graph* $G = (V, E)$ consists of two sets V and E, where V represents the set of vertices and E represent the set of edges and each edge is an ordered pair (v, w) of vertices. When *directed graphs* are used to represent the process networks, the set of vertices V represent the set of tasks and the directed edges E represent the flow of material from one task to another. This representation can be extended for systems with multiple jobs by utilizing a series of *directed graphs* as

$G(i) = (V_i, E_i)\ \forall i \in I$, where the set of vertices V_i represents the set of tasks in the path of job i (P^i) and the directed edge set E_i represents the flow of units from one task to another. For every job $i \in I$ and for any $k = 1..q_{i-1}$, there exist a forward edge (P^i_k, P^i_{k+1}) representing the transfer of units to the subsequent tasks. As per the standard notation, the set of edges leaving a vertex $v \in V_i$ of graph $G(i)$ are denoted as $\delta^+_{(G(i))}(v)$,and the set of edges entering a vertex $v \in V_i$ of graph $G(i)$ are denoted as $\delta^-_{(G(i))}(v)$. Similarly, for any $k = 1..q_i$, the set of tasks to which units are transferred from a task P^i_k are denoted as $N^+_{(G(i))}(P^i_k) = \{P^i_h \in V_i : \exists(P^i_k, P^i_h) \in E_i\}$ and the set of tasks from which units are transferred to a task P^i_k are denoted as $N^-_{(G(i))}(P^i_k) = \{P^i_h \in V_i : \exists(P^i_h, P^i_k) \in E_i\}$. As the definition of P^i_k clearly indicates the k^{th} task of job i, in order to simplify the notations, hereafter, in this work the sets $N^+_{(G(i))}(P^i_k)$ and $N^-_{(G(i))}(P^i_k)$ will be defined as $N^+_G(P^i_k)$ and $N^-_G(P^i_k)$

3.1.2 Time Discretization

A key factor in developing the scheduling model includes the time discretization scheme employed in the model.

As mentioned in section 2.1, in this study, a non-uniform time discretization (NUD) scheme was adapted for the scheduling model. In a discrete formulation, scheduling decisions can only be taken at a discrete set of timepoints within the scheduling horizon. Specifically, for every task $j \in J$, $\Delta(j)$ is defined to be the interval between timepoints and define time point $\varepsilon(j,t) := (t-1)\Delta(j), \forall t = 1, \ldots, \lceil \frac{H}{\Delta(j)} \rceil$. Additional definitions include: $\varepsilon(j, \lceil \frac{H}{\Delta(j)} \rceil + 1) = H$, and the set of all timepoints $\varepsilon(j) := \{\varepsilon(j,t) : t = 1, \ldots, \lceil \frac{H}{\Delta(j)} \rceil + 1\}$.

As multiple tasks with different process completion times are considered in this study, to efficiently account for the flow of units within tasks, a function θ is introduced, which will help in identifying the timepoints of a preceding task that would lead to that task finishing between two timepoints of the following task. Consider a task P^i_k of job i and let k' be a task from which units are transferred to the task P^i_k, i.e. $k' \in N^-_G(P^i_k)$. Let $\varepsilon(P^i_k, t)$ be the time at which P^i_k starts processing. Then, the units available to be processed in task P^i_k includes the units from $P^i_{k'}$ that finished processing in the interval $(\varepsilon(P^i_k, t-1), \varepsilon(P^i_k, t)]$.

Figure 3.1: Example time discretization of a task j with $\Delta(j) = 20$ and a scheduling horizon of 480

In order to obtain all the timepoints associated with the units transferred from every task k', the function θ is defined as follows:

$$\theta(i,k,k',t) = \left\{ r \in \varepsilon'(P^i_{k'}) : \varepsilon(P^i_k, t-1) < \varepsilon(P^i_{k'}, r) + \phi(P^i_{k'}) \leq \varepsilon(P^i_k, t) \right\}.$$

where $\phi(P^i_{k'})$ represents the completion time (processing time) of task $P^i_{k'}$.

As an example, suppose that the discretization of task 1 is exactly as shown in Figure 3.1. Moreover, suppose that $\phi(1) = 20$ and units are transferred from task 1 to task 2, which has a time discretization with a time step, $\Delta(2) = 60$. Now, the total units available for processing in task 2 at time point $t = 6$ (i.e. $\varepsilon(2, 6) = 300$) is equal to the number of units that finished processing task 1 in time interval (240, 300]. This includes all the units that begun processing task 1 in (220, 280], and so $\theta(i, 2, 1, 6) = \{13, 14, 15\}$.

3.1.3 Deterministic Scheduling Model

The deterministic model for such large-scale multi-job multitasking facilities was first presented in [10].

Decision Variables

The key decision variables include:

- y_{ikt} - number of units processed in each task k of job i at timepoint t,
- x_{ikt} - number of units waiting to be processed in each task k of job i at timepoint t,

- z_{jt} - the number of resources to be operated at timepoint t for a task $j \in J$

The deterministic scheduling model (D1) is presented next.

Resource Constraint

Constraints (3.1) are the resource allocation constraints and it ensures that a resource cannot be interrupted once it starts processing a batch of units. The summation on the l.h.s takes into account the fact that once a resource (machine) starts processing a batch of units, then it cannot start another batch until the current one is finished. For example, consider a scheduling horizon with a time discretization of 1hr. Let the task 1 has a processing time of 2hrs and the available resources for task 1 be 4. Then, at $t = 2$, constraint (3.1) can be represented as $z_{11} + z_{12} \leq 4$.

$$\sum_{\varphi \in \varepsilon(j): \varepsilon(j,t) < \varepsilon(j,\varphi) + \phi(j) \leq \varepsilon(j,t) + \phi(j)} z_{j\varphi} <= |R_j| \quad \forall j \in J, \forall t \in 1..|\varepsilon(j)| \tag{3.1}$$

Capacity Constraint

Constraints (3.2) represent the capacity constraints, which enforces that the total number of units from all job $i \in I$:$j \in P^i$ that can be operated in a task j (multitasking) at time point t should not exceed the total capacity of all the resources available at that time point to perform the operation.

$$\sum_{i \in I} \sum_{k=1..q_i : P_k^i = j} y_{ikt} \leq z_{jt} c_j \quad \forall j \in J, \forall t \in 1..|\varepsilon(j)| \tag{3.2}$$

Initialization Constraint

Constraints (3.3) represent the initialization constraint that defines the flow balance at the first time point. The amount processed in any task P_k^i of a job i at first time point (y_{ik1}) is equal to the number of units available to be processed at the beginning of the time horizon (A_i). If the amount received exceeds the capacity of the task P_k^i, the difference is considered as the waiting units (i.e. units from job i that are waiting to be processed in task P_k^i, which is represented by x_{ik1}. Thus, constraints (3.3) distributes the available

units into amount that can be processed immediately and amount that is waiting to be processed. Note that it is assumed, there is infinite storage availability, i.e. no restriction is imposed on the number of waiting units; this assumption can be easily enforced with an extra constraint if required.

$$x_{ik1} = A_i - y_{ik1} \quad \forall i \in I, \forall k \in 1..q_i \tag{3.3}$$

Flow Balance Constraint

Constraints (3.4) represent the flow balance constraints for any task P_k^i of a job i at all time points except the first. Note that the term flow balance here is a generic term and with respect to the system under consideration, it can be referred to as mass balance or material balance or state balance constraints. This constraint defines that the total amount of units available for processing in task P_k^i of job i at time point t (including the units that have been waiting at the previous time point (x_{ikt-1}) and those that have finished processing previous tasks in the path ($y_{ik'r}$) is equal to the sum of units that can be processed at time point t (y_{ikt}) and the units that are waiting at time point t (x_{ikt}). The summation on the r.h.s over $r \in \theta(i, k, k', t)$ accounts for all the batches of task k' that has finished processing between the previous time point $(t-1)$ and the current time point (t). For example, let the processing time of task k be 2hrs and that of task k' be 1hr, then at $t = 2$ the constraint (3.4) can be represented as $x_{ik2} + y_{ik2} = x_{ik1} + \rho_{ik'k}(y_{ik'1} + y_{ik'2})$.

$$x_{ikt} + y_{ikt} = x_{ikt-1} + \sum_{k' \in 1..q_i : P^i_{k'} \in N_G^-(P_k^i)} \sum_{r \in \theta(ikk't)} y_{ik'r} \quad \forall i \in I, k \in 1..q_i, t \in 2..|\varepsilon(P_k^i)| \tag{3.4}$$

Bounds Constraint

Constraints (3.5) defines that the decisions are non-negative.

$$\begin{aligned} x_{ikt},\, y_{ikt} &\geq 0 \quad , \forall i \in I, k \in 1..q_i, t \in 1..|\varepsilon(P_k^i)| \\ z_{jt} &\geq 0 \qquad\quad , \forall j \in J, t \in 1..|\varepsilon(j)| \end{aligned} \tag{3.5}$$

Objective Function

The objective function of the scheduling model aims at maximizing the through put/processing

rate of the units by providing higher weights to the final tasks and minimum weights to the initial tasks. The weights are defined by the ratio $\frac{k}{q_i}$, where k varies from $1..q_i$. Thus, the first task in the path where $k = 1$ possess the minimum weight which further increases until the final task in the path where $k = 1..q_i$. Note that it is possible that the processing times of some tasks maybe higher than the scheduling horizon. As a result, the samples of a job may not complete processing all the tasks in its path within the considered scheduling horizon. To address this issue, weighted tasks are considered in the objective function instead of just the final task in order to ensure that the model allows processing of samples even if the considered scheduling horizon is less than the minimum completion time of the job.

$$Max \sum_{i \in I} \sum_{k \in 1..q_i} \sum_{t \in 1..|\varepsilon(P_k^i)|} \frac{k}{q_i} y_{ikt}$$

3.2 Uncertainty

When such process networks are considered, it is not expected that all the tasks perform in an ideal way (where all available resources function perfectly without any factors affecting its performance) and provides a perfect outcome throughout the operation period (time horizon). For a scheduling problem, the outcome of a task that performs ideally would result in 100% transfer of units to the succeeding task in a path/recipe. However, in actual practice, many factors such as resource malfunction, fluctuations in process parameters, human intervention etc. could affect the performance of the tasks, resulting in fewer units being transferred to the succeeding task. Based on the performance of the tasks (machines/resources), a certain fraction of the units may require additional processing requirements and are therefore sent back to one of the previous tasks as *recycle*. This recycled fraction of units A_i from job $i \in I$ has to repeat all or a subset of the previous tasks in its path. In order to account for the possibility of such unforeseen factors like machine errors, operating conditions, human intervention affecting the task performance and the obtained outcome, certain tasks can be considered as imperfect tasks, indicating deviation

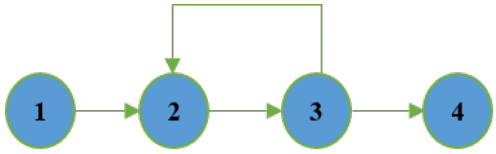


Figure 3.2: Illustration of a graph $G(i)$ of a job i with four tasks in its path

in performance of these tasks from the perfect/ideal case. To account for uncertainty in these tasks, the outcome of the task, i.e. the rate of units transferred from an imperfect task to the subsequent task in the path is considered as an uncertain parameter. As the time of the realization of uncertain parameter (outcome of the task) depends on the model decisions such as the time at which the units finish processing previous tasks in the path and time at which the imperfect task begins processing, it is considered as a type II endogenous uncertainty with decision dependent time of uncertainty realization.

3.2.1 Modified Process Network

Based on the above description of uncertainty, in order to account for the uncertain parameter, the process network is modified to include the following additional features. Every job $i \in I$ may contain an imperfect task in its path, represented by $P^i_{j'}$, where $j' \in 1..q_i$. From an imperfect task $P^i_{j'}$, a fraction of units (depending up on the realization of the task outcome) may have to be recycled back to one of the previous tasks in its path for reprocessing. Thus, for an imperfect task $P^i_{j'}$, apart from the forward edge, there exists, an additional backward edge that enters the task to which units are recycled, i.e. $(P^i_{j'}, P^i_m) \in E : j' > m$. Fig.3.2 presents an illustration of graph $G(i)$ of a job i with four tasks in its path and edges indicating the flow of units from one task to another. In the figure, vertices 1, 2, 3 and 4 represents the tasks in the path of a job, where units are processed and transferred from task 1 to task 4, represented by a forward edge. After processing the imperfect task 3, few units are recycled back to task 2 for reprocessing. This revisiting of task 2 is represented using a backward edge.

For any job $i \in I$ and any $k = 1..q_i$, the fraction of units transferred from a task is represented by ρ_{ikn}, $\forall n \in N^+_G(P^i_k)$. Note that allowing fractional values for ρ_{ikn}, may end

up with fractional number of units to be transferred within the tasks, even though the units may be integer in nature. Fractional units are allowed to simplify the modelling; this may be a limitation, but fractional units are allowed with the understanding that the model is supposed to provide a plan to be followed and adapted once the actual realization is observed. The sum of all the output rates from an imperfect task should be equal to unity, i.e. $\sum_{(n=1..|N_G^+(P_k^i)|)} \rho_{ikn} = 1$ (referring back to Fig.3.2, consider task 3, as units are transferred to task 2 and task 4 from task 3, this expression can be defined as $\rho_{132} + \rho_{134} = 1$). Regarding the process network and model features, the following assumptions are considered:

Assumption M1: a job can have at most one imperfect task and an imperfect task can have at most one backward edge leaving from it. While this assumption may be restrictive, it is accepted and valid for analytical services facilities or similar manufacturing applications where there usually include a quality checking task that determines if samples/units should be re-tested/recycled.

Assumption M2: the transfer rate of units, ρ_{ikn} of any forward edge should always be greater than zero, i.e. for an imperfect task the fraction of units transferred via the backward edge should always be less than 1.

Assumption M3: once the actual realization of the uncertain parameter, $\rho_{ikn} : k = j^{'}$ for an imperfect task $P_{j^{'}}^i$ is realized, every single time the task is executed for the remaining time horizon, the fraction of units transferred back via backward edge will be the same (i.e. ρ_{ikn}).

The author notes that M3 is a limiting assumption, since every time an imperfect task is executed, if there is a nonzero fraction of samples going back, it would imply that the task would never be successfully completed. However, the time horizons considered here are typically smaller than the time involved in completing the entire sequence of tasks. If the task completion times are high, a smaller time horizon would also imply that the imperfect task might be operated just once or twice. Thus, this assumption is not too restrictive, as the model can be solved again in the next time horizon with different parameter choices.

Based on the above description of the process network and the uncertainty considerations, the aim is to search for feasible solutions to this problem that consists of a schedule

(depending on the time horizon) for every realization of uncertainty (i.e. scenario), with an objective of maximizing the expected throughput. The schedule would provide a set of first-stage decisions irrespective of the realization of uncertainty. It includes decisions such as the batch sizes (number of units processed in each task), the number of units waiting to be processed in each task and the time at which a resource has to be operated. The schedule would also provide a set of second stage decisions that represent the recourse actions enacted upon the realization of uncertainty, i.e. based on the resolved outcome of the task; these decisions reflect the changes in the number of units transferred to subsequent tasks in the path.

3.3 Proposed Two-stage Model

Based on the above descriptions, the mathematical model for the optimization of batch operations under type II endogenous uncertainty via two-stage stochastic programming with implicit non-anticipativity is presented here.

The first stage decisions and second stage decisions of the model are described next, followed by the mathematical formulation.

As every job $i \in I$ may consist of an imperfect task in its path, once the imperfect task finishes processing the first batch of samples, uncertainty (the fraction of samples recycled) will be realized and the flow of samples for the remaining time horizon would vary with respect to the value of the realization (scenario). In this study, one possible realization of all the uncertain parameters is referred to as a scenario, i.e. if every job $i \in I$ contains an imperfect task $P^i_{j'}$ in its path, a scenario s can be represented as $(\rho^s_{1j'n}, \rho^s_{2j'n}, \dots \rho^s_{|I|j'n})$.

3.3.1 Model Decisions

The key decisions of this problem consists of the flow decisions that represents the number of samples to be processed from a job i in a task k of its path at a time t in a scenario s (w^s_{ikt}) and the number of samples waiting to be processed in a task k of job i at time t in a

scenario s (v^s_{ikt}). These decisions are referred to as the final implementable decisions. Similar to any other scenario-based stochastic model decisions [14, 96], one key requirement is to make sure that these decisions are non-anticipative in nature and remain the same for all scenarios until the time of uncertainty realization. As the uncertainty considered is endogenous and the time of uncertainty realization is unknown, this becomes very challenging. In the studies available in literature that involves type II endogenous uncertainties, in order to ensure the non-anticipativity of such final decisions, auxiliary binary variables are introduced [53, 46, 16]. In the present study, these final implementable decisions are defined as a combination of the decision variables x_{ikt}, y_{ikt} and x^s_{ikt}, y^s_{ikt}. Variables x_{ikt} and y_{ikt} represent the flow decisions (units processed and units waiting respectively) that can be made irrespective of any realization of the uncertainty, hence referred to as the first stage decisions. Variables x^s_{ikt} and y^s_{ikt} represent the scenario based flow decisions that has to be made as a recourse after the realization of uncertainty for every scenario s and are referred to as second stage recourse decisions. These first stage and the second stage recourse decisions are combined together to obtain the final implementable decisions at any time t for a scenario s.

Obtaining the first stage decisions

One of the main challenges in this work is to obtain a set of first stage decisions that are common to all the scenarios. As the time of uncertainty realization is unknown and decision dependent, it is impossible to predetermine the time until which the decisions of every scenario should be common; hence, in this study, the first stage decisions are defined for the entire horizon based on a fixed value of the uncertain parameter ($\rho_{ikn} : k = j^{'}$). The stage 1 decisions are defined in such a way with the understanding that these decisions have to be modified once the uncertainty is realized (which is achieved using the second stage recourse decisions). There are different possibilities for how this value of uncertain parameter can be fixed to obtain the first stage decisions. Few of those possibilities are presented here. First, by assuming the ideal case, i.e. the first stage decisions can be obtained by assuming that all the tasks including the imperfect task behaves ideally and there is a 100% transfer to the succeeding task from the imperfect task, implying that the

value of $\rho_{ij'n}$ for the succeeding task is equal to 1 and for the recycled task is equal to zero. In this case, the obtained plan based on first stage decisions would over allocate units to the tasks succeeding the imperfect task. As it is highly likely that the actual uncertainty realization (recycle rate) would be non-zero (resulting in lesser number of units being transferred to the successive task), some of the decisions taken in the first stage have to be undone in the second stage. This requires extending the bounds of the second stage variables (x^s_{ikt}, y^s_{ikt}) that reflects the variations in flow due to the actual realization to possess negative values as well.

Second, by assuming the worst case, i.e. the first stage decisions can be obtained by assuming that the recycling rate from an imperfect task $(\rho_{ikn} : k = j^{'})$ is always equal to its maximum value. For instance, if the possible uncertain realizations are $(\rho_{ij'n} = \{0.3, 0.5, 0.6\})$, then, in this case, the value of $\rho_{ij'n}$ for the recycled task is set equal to 0.6 and for the succeeding task is equal to 0.4. In this case, the obtained plan based on first stage decisions might over allocate units to the tasks preceding the imperfect task. Similar to the first case, in order to account for the actual realization, some of the decisions taken in the first stage have to be undone and therefore the bounds of the second stage variables that reflects the variations in flow due to the actual realization (x^s_{ikt}, y^s_{ikt}) has to be extended to encompass the negative values as well. Third, by assuming an expected (average) value, i.e. the first stage decisions can be obtained by assuming that the recycling rate from an imperfect task $(\rho_{ikn} :k = j^{'})$ is always equal to its average value. Similar to the first two cases, even in this case, the bounds of the second stage variables (x^s_{ikt}, y^s_{ikt}) have to be extended to possess negative values in order to undo some of the decisions taken in the first stage. Several tests were conducted using all these three cases (assuming the ideal case, worst case and average case) as the fixed value for the uncertain parameter to obtain the first stage decisions. However, these cases did not guarantee non-anticipativity enforcement.

The fourth way to choose the fixed value is based on what is certain to happen as per the known (guaranteed) aspects of the process operation. More specifically, for a job $i \in I$, all tasks except the imperfect task consists of only one forward edge, therefore it is certain that a complete transfer of units takes place from these tasks to its subsequent task.

Whereas, once the imperfect task is processed, a certain fraction of units is transferred via backward edge and the remaining units are transferred to the subsequent task, via forward edge. Even though the value of realization is unknown, as the possible realizations of the uncertain parameter are available, the maximum possible realization of the uncertain parameter is known and can be defined as U_B. Then, it is known for certain that from an imperfect task, at least a fraction of (1-U_B) units will be transferred to the subsequent task (via forward edge). Since it is not certain about the fraction of units transferred via the backward edge, the lower bound of the possible realizations, denoted by L_B is assumed for the backward edge for the first stage decisions. Thus, for every job $i \in I$, for the imperfect task, $P^i_{j'}$, ρ_{ikn}:$k = j'$ for forward edge and backward edge is defined as (1-U_B) and L_B respectively, whereas for all other tasks i.e. P^i_k:$k \neq j'$, ρ_{ikn} for forward edge is defined as 1. Here, by choosing the fixed value based on what is certain to happen, in the first stage, only the minimum number of units are allowed to transfer to the successive and recycled tasks from an imperfect task. Therefore, when the actual uncertainty realizes, the second stage variables (x^s_{ikt}, y^s_{ikt}) only have to account for the additional units transferred to the successive tasks and the additional units recycled back to the previous tasks; hence, in this case the bounds of the second stage variables can remain positive. The bounds of the second stage variables is key in enforcing the non-anticipativity implicitly. This will be discussed in detail following the model description and in the Theorem 1. Since this fourth choice of fixed value based on what is certain to happen is the only option for which implicit non-anticipativity can be guaranteed, this option is considered in this study and the assumption is stated as follows.

Assumption M4: the first-stage decisions are obtained by fixing the value of uncertain parameter, ρ_{ikn} as follows:

• For the imperfect task of every job $i \in I$, i.e. $P^i_{j'}$, $\rho_{ikn} : k = j'$ is defined as $1 - U_B$ for the forward edge and L_B for the backward edge, where U_B is the maximum possible realization of the uncertain parameter while L_B is the minimum possible realization of the uncertain parameter.

Note that as ρ_{ikn} (defined in the first stage constraints) do not represent the actual realization, but just what is known for certain to occur, the sum of the output rates for

the imperfect task in the first stage constraint may not add up to unity. However, ρ_{ikn} of the first stage constraint and the actual realization, $\rho^s_{i,k'k}$ of the second stage constraint together ensures that all the units processed in an imperfect task are either transferred to the successive task or the recycling task (task to which the units are sent back from an imperfect task for reprocessing) in the path of job i.

Another model decision includes the number of resources to be operated (z_{jt}) at time point t for a task $j \in J$ throughout the time horizon. Note that once the uncertainty is realized, the flow variables are adjusted based on the value of realization (x^s_{ikt}, y^s_{ikt}), but the number of resources (z_{jt}) remains a decision obtained irrespective of the uncertainty realization. This might be a restrictive assumption. However, z_{jt} is allowed to be a first stage decision as the current study does not account for personnel allocation. The operation of machines is often tied to personnel allocation and to ensure smooth functioning of operations, modifying the number of resources would require updating the allocation of personnel as well. At this point, personnel allocation is beyond the scope of current study. Therefore, the final model assumption is stated as follows.

Assumption M5: the number of resources is allowed to be a first stage decision and are not modified with the realization of uncertainty.

3.3.2 Stochastic Model

The mathematical formulation is presented next, followed by the detailed description of the model components.

Objective Function

$$\max \sum_{s \in S} \sum_{i \in I} \sum_{k \in 1..q_i} \sum_{t \in 1..|\varepsilon(P^i_k)|} \frac{k}{q_i} w^s_{ikt} \psi_s$$

First Stage Constraints

Constraints: (3.1) − (3.3), (3.5),

$$x_{ikt} + y_{ikt} = x_{ikt-1} + \sum_{k' \in 1..q_i : P^i_{k'} \in N^-_G(P^i_k)} \sum_{r \in \theta(i,k,k',t)} \rho_{ik'k} y_{ik'r} \quad , \forall i \in I, k \in 1..q_i, t \in 2..|\varepsilon(P^i_k)| \tag{3.6}$$

Second Stage Constraints

$$\sum_{i \in I} \sum_{k \in 1..q_i : P^i_k = j} y_{ikt} + y^s_{ikt} \leq z_{jt} C_j \quad , \forall s \in S, j \in J, t \in 1..|\varepsilon(j)| \tag{3.7}$$

$$(x^s_{ikt} + x_{ikt}) + (y^s_{ikt} + y_{ikt}) = (x^s_{ikt-1} + x_{ikt-1}) + \sum_{k' \in 1..q_i : P^i_{k'} \in N^-_G(P^i_k)} \sum_{r \in \theta(i,k,k',t)} \rho^s_{ik'k} (y^s_{ik'r} + y_{ik'r})$$
$$, \forall s \in S, i \in I, k \in 1..q_i, t \in 2..|\varepsilon(P^i_k)| \tag{3.8}$$

$$y^s_{ik1} = x^s_{ik1} = 0 \quad , \forall s \in S, i \in I, k \in 1..q_i \tag{3.9}$$

$$y^s_{ikt},\ x^s_{ikt} \geq 0 \quad , \forall s \in S, i \in I, k \in 1..q_i, t \in 2..|\varepsilon(P^i_k)| \tag{3.10}$$

$$w^s_{ikt} = y^s_{ikt} + y_{ikt} \quad , \forall s \in S, i \in I, k \in 1..q_i, t \in 1..|\varepsilon(P^i_k)| \tag{3.11}$$

$$v^s_{ikt} = x^s_{ikt} + x_{ikt} \quad , \forall s \in S, i \in I, k \in 1..q_i, t \in 1..|\varepsilon(P^i_k)| \tag{3.12}$$

The objective function is defined in terms of the final implementable decisions (w^s_{ikt}) that accounts for both the first (y_{ikt}) and second stage (y^s_{ikt}) variables involved in the

model, and the probabilities associated with each scenario. The objective function is to maximize the expected production/processing rate of the units by providing higher weights to the final tasks and minimum weights to the initial tasks. The weights are defined by the ratio $\frac{k}{q_i}$, where k varies from $1..q_i$. Thus, the first task in the path where $k = 1$ possess the minimum weight which further increases until the final task in the path where $k = 1..q_i$. ψ_s represents the probability of each scenario s. Note that the primary focus of this study is to demonstrate the implicit non-anticipativity enforcement of the proposed approach without any auxiliary binary variables or explicit NACs. It will be demonstrated later in this section that the key components in enforcing non-anticipativity do not involve the objective function. Therefore, the proposed framework for implicit non-anticipativity can be applied irrespective of the objective function considered (minimization of make span /maximization of profit etc.). A detailed evaluation of the objective function was considered beyond the scope of this study.

Constraints (3.1)-(3.6) represent the first-stage formulations. Constraints (3.1) - (3.3) and (3.5) are same as that of from the deterministic model. Constraint (3.6) represents the flow balance constraints for any task P_k^i of a job i at all time points except the first. This constraint defines that the total amount of units available for processing in task P_k^i of job i at time point t (including the units that have been waiting at the previous time point (x_{ikt-1}) and those that have finished processing previous tasks in the path ($y_{ik'r}$)) is equal to the sum of units that can be processed at time point t (y_{ikt}) and the units that are waiting at time point t (x_{ikt}). Note that the uncertain parameter is fixed for the first stage flow constraint and $\rho_{ik'k}$ represents the fixed value. As discussed previously, the fixed value is based on what is certain to happen (based on the guaranteed aspects of the process operation) and irrespective of the actual realization. Thus, constraint (3.6) together with constraints (3.1), (3.2), (3.3) and (3.5) provide the first stage decisions irrespective of the actual realization of uncertain parameters.

Constraints (3.7) to (3.12) represent the second-stage constraints. Constraints (3.7) represents the capacity restrictions for the total number of units that can be processed at any time t in a task j. To account for the actual realization of uncertainty and the variations in flow of units based on the actual realization, a second-stage flow balance constraint

is introduced. Constraint (3.8) represents the second-stage flow balance constraint with the first and second stage variables (y_{ikt}, y^s_{ikt}) coupled for every scenario s and the actual realization of the uncertain parameter $\rho^s_{ik'k}$. Note that for any task other than the imperfect task, i.e. $P^i_k{:}k \neq j^{'}$, $\rho^s_{ik'k} = \rho_{ik'k}$. Constraint (3.9) represents the initialization of the second-stage variables. Second stage variables here are those that depicts the variations in flow of units based on the actual realization of uncertainty. As this study considers an endogenous uncertainty that depends on the model decisions, i.e. in order for the uncertainty to realize, a decision has to be made before, implying that the earliest possible time for the uncertainty to realize would be the second time point. Hence, the second-stage variables are initialized to zero as there would not be any uncertainty realizations at the first time point.

Constraint (3.10) defines that the second-stage variables for all the tasks should be non-negative. As discussed earlier in this section, the bounds of the second stage variables depends on the fixed value chosen for the uncertain parameters in the first stage constraint. Since it is assumed that a minimum fraction of units, $(1 - U_B)$ are transferred forward from an imperfect task and a minimum of (L_B) units being recycled back, any realization (actual) of uncertain parameter would result in addition of units to the successive tasks of an imperfect task and to the tasks to which units may be recycled back. While the first stage decision y_{ikt} represents the number of units being processed in a task P^i_k of job i at time point t, a positive second stage decision y^s_{ikt} implies that based on the actual realization of uncertainty more units from job i are to be processed in task P^i_k at time point t. For any task prior to the imperfect task, a positive second stage variable implies that the value of uncertainty realization is greater than the minimum value and a batch of units have been recycled back from the imperfect task for reprocessing. For any task subsequent to the imperfect task, a positive second stage variable implies that the value of uncertainty realization is less than the maximum possible realization (U_B) and higher number of units can be transferred to the subsequent task from the imperfect task.

Furthermore, constraints (3.11) and (3.12) provides the final implementable decisions (w^s_{ikt} and v^s_{ikt}) by considering both first stage decisions and the second stage recourse decisions for every task P^i_k at every time point t, for every scenario s.

Thus, the model provides a set of final implementable decisions (w^s_{ikt} and v^s_{ikt}) for every scenario s, which is defined as a combination of the stage 1 (x_{ikt}, y_{ikt}) decisions that provides the minimum number of units to be transferred to the successive tasks irrespective of the uncertainty realization, and the stage 2 recourse decisions (x^s_{ikt}, y^s_{ikt}) that provides the additional number of units to be transferred to the successive tasks based on the actual uncertainty realization. Because of the way the stage 2 recourse decisions and constraints are defined, the model ensures that these decisions (x^s_{ikt}, y^s_{ikt}) attain a non zero value only after the uncertainty realization (see Theorem 1). Therefore, the final implementable decisions would become equivalent to:

- $w^s_{ikt} = y_{ikt}$ & $v^s_{ikt} = x_{ikt}$ (until the time of uncertainty realization)
- $w^s_{ikt} = y_{ikt} + y^s_{ikt}$ & $v^s_{ikt} = x_{ikt} + x^s_{ikt}$ (after the uncertainty is realized)

Thus, the definition of the stage 1 and stage 2 recourse decisions and the corresponding constraints ensures that all the w^s_{ikt} and v^s_{ikt} decisions until the time of uncertainty realization remain the same for every scenario s and thereby enforces non-anticipativity implicitly without any auxiliary binary variables.
The proof for the implicit non-anticipativity enforcement is presented next.

Theorem 1. *Assume the second-stage decisions $x^s_{ikt}, y^s_{ikt}, w^s_{ikt}$ satisfy constraints (3.8) - (3.12) and the first-stage decisions x_{ikt}, y_{ikt} satisfy constraints (3.2) - (3.6) under the assumption that the first-stage decisions are obtained by fixing the value of uncertain parameter, ρ_{ikn} as follows:*

- *For the imperfect task of every job $i \in I$, i.e. $P^i_{j'}$, $\rho_{ikn} : k = j'$ is defined as $1 - U_B$ for the forward edge and L_B for the backward edge, where U_B and L_B are the maximum and minimum possible realizations of the uncertain parameter, respectively.*
- *For all the remaining tasks $P^i_k : k \neq j'$ of job $i \in I$, ρ_{ikn} for the forward edge is defined as 1.*

Then, the final implementable decisions before the time of uncertainty realization (t'_i) is same as that of the first-stage decisions, i.e.

$$v^s_{ikt} = x_{ikt} \qquad \forall t \in 1..|\varepsilon(P^i_k)| : t < t'_i, i \in I, k \in 1..q_i, s \in S$$

$$w^s_{ikt} = y_{ikt} \qquad \forall t \in 1..|\varepsilon(P^i_k)| : t < t^{'}_i, i \in I, k \in 1..q_i, s \in S$$

where $t^{'}_i \in 1..|\varepsilon(P^i_{k'})|$ *is the time at which* $y_{ik'r}$ *attains a positive value for the first time, given* $P^i_{k'} = P^i_{j'}$

Proof. Rearranging the first-stage flow balance constraint (3.6) yields:

$$x_{ikt} - x_{ikt-1} + y_{ikt} - \sum_{k' \in 1..q_i : P^i_{k'} \in N^-_G(P^i_k)} \sum_{r \in \theta(i,k,k',t)} \rho_{ik'k} y_{ik'r} = 0 \tag{3.13}$$

Rearranging the second-stage flow balance constraint (3.8) yields:

$$\begin{aligned} x_{ikt} - x_{ikt-1} + y_{ikt} - \sum_{k' \in 1..q_i : P^i_{k'} \in N^-_G(P^i_k)} \sum_{r \in \theta(i,k,k',t)} \rho^s_{i,k'k} y_{ik'r} + \\ x^s_{ikt} - x^s_{ikt-1} + y^s_{ikt} - \sum_{k' \in 1..q_i : P^i_{k'} \in N^-_G(P^i_k)} \sum_{r \in \theta(i,k,k',t)} \rho^s_{i,k'k} y^s_{ik'r} = 0 \end{aligned} \tag{3.14}$$

Let $P^i_{j'}$ be the imperfect task in the path of a job $i \in I$ and $t^{'}_i$ be the first time point $\forall i \in I$ at which $y_{ik't}$ attains a non-zero value. Thus, $\forall t < t^{'}_i, y_{ik't} = 0$. Accordingly, in constraints (3.13) and (3.14), $\forall t < t^{'}_i$:

$$\begin{aligned} \sum_{k' \in 1..q_i : P^i_{k'} \in N^-_G(P^i_k) \& P^i_{k'} = P^i_{j'}} \sum_{r \in \theta(i,k,k',t)} \rho_{ik'k} y_{ik'r} = 0 \\ \sum_{k' \in 1..q_i : P^i_{k'} \in N^-_G(P^i_k) \& P^i_{k'} = P^i_{j'}} \sum_{r \in \theta(i,k,k',t)} \rho^s_{i,k'k} y_{ik'r} = 0 \end{aligned} \tag{3.15}$$

(Note that depending upon the imperfect task, the value of $t^{'}_i$ may vary for each job $i \in I$ that considers at imperfect task. However, same proof can be applied to every job $i \in I$ and the corresponding $t^{'}_i$).

The theorem will be demonstrated by induction on t. Consider the flow balance equations

$\forall t \in 1..t_i' - 1$.

Base case:
$t = 1$: Constraint (3.9) implies that:

$$y^s_{ik1} = x^s_{ik1} = 0 \quad \forall i \in I, k \in 1..q_i, s \in S \tag{3.16}$$

Induction step: Assuming the theorem holds true for time $t-1$, consider time t (Note: $t \in 1..t_i' - 1$):
$t = t$: Substituting (3.15) in equation (3.13) yields:

$$x_{ikt} - x_{ikt-1} + y_{ikt} + \sum_{k' \in 1..q_i : P^i_{k'} \in N^-_G(P^i_k) \& P^i_{k'} \neq P^i_{j'}} \sum_{r \in \theta(i,k,k',t)} \rho_{i,k'k} y_{ik'r} = 0 \tag{3.17}$$

Now, substituting equations (3.15) and (3.17) in equation (3.14) yields:

$$x^s_{ikt} - x^s_{ikt-1} + y^s_{ikt} - \sum_{k' \in 1..q_i : P^i_{k'} \in N^-_G(P^i_k)} \sum_{r \in \theta(i,k,k',t)} \rho^s_{i,k'k} y^s_{ik'r} = 0 \tag{3.18}$$

As the theorem holds for $t-1$:

$$x^s_{ikt-1} = 0 \quad \& \quad y^s_{ik'r} = 0 \quad (r \in 1 \ldots t-1) \tag{3.19}$$

Substituting (3.19) in equation (3.18) yields:

$$x^s_{ikt} + y^s_{ikt} = 0 \tag{3.20}$$

Applying the bounds constraint (3.10) in the above equation yields:

$$x^s_{ikt} = y^s_{ikt} = 0 \tag{3.21}$$

Equation (3.21) shows that all second stage decisions for any task P^i_k of job i and time

$t < t_i^{'}$ are equal to zero in every scenario s. Substituting equation (3.21) in constraints (3.11) and (3.12) gives:

$$v_{ikt}^{s} = x_{ikt} \tag{3.22}$$

$$w_{ikt}^{s} = y_{ikt} \tag{3.23}$$

Equation (3.22) and (3.23) shows that, for any task P_k^i of job i and $t < t_i^{'}$, the final implementable decisions for every scenario $s \in S$ is the same as that of the first stage decisions and is independent of any realization of uncertain parameter, which proves that non-anticipativity is guaranteed by the present framework. □

Even though the model consists of other constraints, non-anticipativity can be guaranteed using constraints (3.2) – (3.12), as stated in the theorem. Thus, this approach can be applied to other problems with a similar set of constraints and can be adapted under similar situations as long as the following conditions hold:

• The uncertain parameter can be accounted for in the model using flow balance constraints.

• The value of the uncertain parameter in the first stage constraint is fixed as described in the above section.

3.4 Computational Experiments

In this section, the stochastic model presented in section 3.3.2 is applied to an industrial case study and discuss the obtained results. The proposed two-stage programming framework has been used to model the scheduling of batch operations for the industrial plant from the analytical services sector.

The industrial plant consists of over 180 tasks and each task has a number of identical machines to perform the task. Thousands of samples, hundreds of tasks and several identical machines to perform each task, results in a very large model. The capacities and processing times of the individual processes vary greatly. The largest capacity among all processes is over 1,300 times the size of the smallest capacity, similarly the processing times

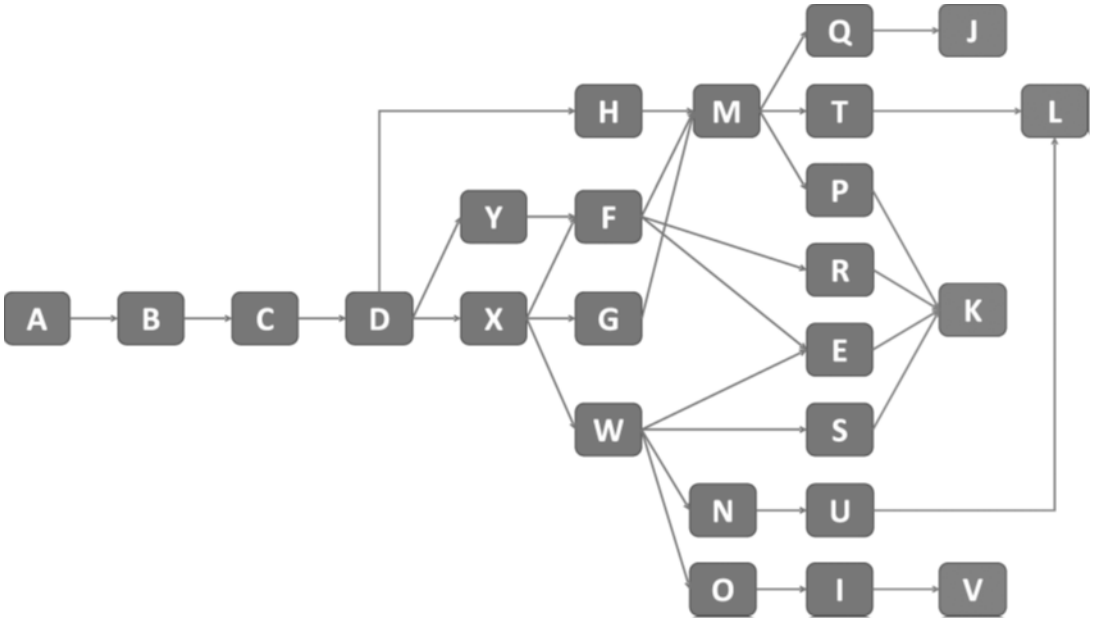


Figure 3.3: A simplified representation of the process network of the plant [2]

of the processes vary from a few minutes to several days. Due to the confidentiality agreement, the actual plant data cannot be presented here; however, in the Appendix A, the normalized data to indicate the main characteristics of the plant under study is provided. Figure 3.3 is only included to convey to the readers the structure of the process networks, where A through Y represent the tasks. The actual plant consists of hundreds of tasks resulting in a complex network of processes. Note that the data from the same industrial plant will be utilized in the computational experiments for the later chapters of this thesis as well.

In a recent study, [10] compared non-uniform time discretization (NUD) and continuous time formulations for scheduling of batch processes. Based on the observations, those authors reported that a scheme of NUD60 has better tradeoff between computational time and solution quality. In a NUD60 discretization scheme, the maximum allowed time-step (the time elapsed between two consecutive time points, which is usually set equal to the completion time of the task) for any task is 60 time-units; i.e. if the task completion time is less than 60, time step is same as the completion time of that task and is equal to 60

otherwise. Based on those results, the scheduling horizon in this work was discretized using a scheme of NUD60.

If there exists an imperfect task in the path of a job i, every time when an imperfect task finishes processing a batch of units, a fraction of those units are recycled back to a previous task in the path, depending on the realization of the uncertain parameter. Six possible discrete recycling rates are considered for an imperfect task, i.e. $\rho^s_{i,k'k} \in \{0, 0.1, 0.2, 0.3, 0.4, 0.5\}$. In this case, as the maximum possible realization of uncertainty is 0.5, in the first stage constraint, the value of the fraction of units transferred from an imperfect task $(1 - U_B)$ was fixed as 0.5 to the subsequent task (via forward edge) and 0 to the recycling task (via backward edge).

In this computational study, one scenario represents a possible recycling rate for the imperfect task of each of the jobs considered. For instance, if the number of jobs considered are 5, then a scenario can be represented as (0, 0.3, 0.1, 0.4, 0.2), which denotes that the recycling rate for the imperfect task in job 1 is 0%, while the recycling rate for the imperfect task in 5^{th} job is 20%. Note that the time of uncertainty realization for each job may be different, i.e. the time at which the imperfect task of job 1 finish processing the first batch of samples may differ from the time at which the imperfect task of job 2 finish processing the first batch of samples.

As mentioned in section 3.2.1, one of the key assumptions considered in this study is that there can be at most one imperfect task in the path of a job. To illustrate the significance of the assumption, a counter example is presented where the non-anticipativity was violated in the two-stage approach when multiple imperfect tasks were considered. Consider the path of a job as: [1,2,4,5,175,166,103,56,187]. Task 103 and Task 187 were considered as the imperfect tasks. Two scenarios were considered as follows: scenario1 – (0.1, 0.2) and scenario2 – (0.1, 0.4) (Note that a scenario (0.1,0.2) here represents the recycle rate of imperfect tasks, i.e., Task 103 and Task 187 respectively). The model was solved for this instance and in this case, the second stage decisions after the uncertainty realization of task 103 until the uncertainty realization of task 187 should be the same in order to ensure non-anticipativity. However, the second stage model decisions for these two scenarios were different, thus violating non-anticipativity. Therefore, with the current approach, this

assumption is a key factor to enforce non-anticipativity.
All the computational experiments were performed on an i7-3.40GHz Windows machine with 16GB RAM using IBM ILOG CPLEX Optimizer 12.7.1.

3.4.1 Results

The model has been solved for different instances. Instances of 5, 15 and 25 jobs were considered, where each job considers one imperfect task in its path and for every job, the time of uncertainty realization can be different depending upon the number of tasks preceding the imperfect task in the path of a job and the processing times of these tasks. The most frequent jobs at the facility were chosen for these instances, i.e. an instance of 5 jobs include the most frequent 5 jobs received at the facility. All the 5 jobs consists of at least 8 tasks in its path. This can be considered as one of the smallest instances as it does not encompass the entire process network. To involve more processes and to increase the process network to an actual industrial scale, instances of higher number of jobs (15 and 25) were also considered. Similar to the first instance, the most frequent 15 jobs and 25 jobs were chosen for these instances. There are 57 and 69 tasks present in the 15 and 25 job instances, respectively. The paths of these job instances are provided as supplementary information and the characteristics of the tasks involved (normalized data) are presented in Appendix A. Some of the jobs consist of tasks with relatively high processing time, due to which it is likely that the uncertainty might not even realize if smaller scheduling horizons were considered. Therefore, to ensure that the model takes in to account the actual realizations and provide the necessary recourse decisions, scheduling horizons of 16hrs and 24hrs are considered in this study. Note that increasing the time horizon also extensively increases the model size; however, as the proposed approach does not require auxiliary binary variables, relatively larger (industrial) instances can be solved using the proposed framework.

The benefits in using a two-stage stochastic approach was quantified by calculating what is referred here as a modified value of stochastic solution (VSS) for every instance. As noted in [14], the definition of classical VSS is the difference between the expected values

obtained from implementing the solution yielded by the proposed two-stage stochastic formulation and the solution obtained by a deterministic formulation that substitutes the uncertain parameters with their mean/expected values. Hence, the mean value of the uncertain parameters are used to estimate the VSS. While the classical VSS uses the objective function of the problem, evaluation is carried out for the proposed approach by calculating the expected values using a modified objective function and hence the difference between the expected values is referred as modified VSS. The detailed explanation of the modified approach and justification of using a different objective function to evaluate is given in the next paragraph.

Note that one of the important goals of a two-stage model is to identify a first-stage solution that is pertinent to all possible observations of the uncertain parameters. Therefore, in this study, the first stage decisions obtained from the deterministic and two-stage model are evaluated and the obtained results are used for calculating the modified value of stochastic solution (VSS). The evaluation procedure can be summarized as follows:

1. Solve the deterministic model and obtain the first-stage decisions.

2. The first-stage decisions are provided as inputs (pre-determined decisions) to the modified two-stage model (described below) and solve the model independently for every scenario.

   For evaluation purposes only, the two-stage model was modified in order to account for the decisions from the deterministic model. Note that the deterministic model is solved for the average value of the uncertain parameters, whereas the actual realization could be less than or greater than the average value. Therefore, to evaluate the first stage decisions, the two-stage model is modified such that:
   • First stage decisions from the deterministic model are provided as inputs;
   • Positive and non-positive values for the second stage variables are allowed;
   Note that deterministic decisions were based on the average value and the actual realization could be greater than or less than the average value. Therefore, for the evaluation of the decisions obtained from the deterministic model, the negative values are allowed as well (E.g. Lets consider that the deterministic decisions were obtained

based on the average value of 25% recycle and actual realization was 10%, then, the recourse action here requires to correct the previous decision of 25% and in order to accomplish that the second stage variables should possess a non-positive value).

• It includes a penalty term in the objective function to penalize the negative second stage variables;

Since negative variables are allowed, a penalty term is included in the objective function to penalize the negative variables in order to restrict the variables to attain a negative value only when necessary to ensure feasibility.

3. Estimate the average value of the throughput (total number of units that finished processing of all tasks in its path) for all the simulations.

   As there are the penalty terms in the objective function of the modified two-stage model, the focus is on the value of throughput instead of the value of objective function so that the comparison between the two-stage and the deterministic model remains reasonable and irrespective of the weights used for the penalty term.

The same evaluation procedure can be used to evaluate the first stage decisions obtained from the original two-stage model. However, note that if the same set of scenarios were used to evaluate the decisions, the throughput obtained from evaluating the two-stage model will be same as that obtained from the original model. Calculating the modified VSS would depict the benefits (in terms of throughput) in using the stochastic approach rather than the deterministic approach. For convenience, hereafter, in this work, the modified VSS is referred as VSS.

The deterministic model was solved by setting the recycling rate of units equal to the average value of all possible rates, i.e. in this case 25%. The evaluation results of both deterministic and two-stage model were used for calculating the VSS and the results obtained for randomly chosen 50 scenarios of the corresponding instances are reported in the Table 3.1.

As shown in Table 3.1, the VSS for all the instances shows significant benefits in using the proposed two-stage stochastic model in comparison to the deterministic model with

Table 3.1: Comparison results obtained from different instances

Scenario	Instance	Through put		VSS
		Deterministic	Two Stage Model	
50	5Jobs 16Hrs	62.65	68.14	8.05%
	5Jobs 24Hrs	140.55	157.84	10.95%
	15Jobs 16Hrs	152.37	162.88	6.45%
	15Jobs 24Hrs	254.53	296.62	14.18%
	25Jobs 16Hrs	182.5	193.98	5.91%
	25Jobs 24Hrs	249.32	296.6	15.9%

average value for the uncertain parameter. This table also shows that, as the time horizon increases, the value of VSS increases. This shows that as the time horizon increases, there is more scope for improvement and therefore results in higher benefits in using the proposed two-stage stochastic model.

One of the key features of the framework is that it allows multiple jobs, where each job can have a distinct time of uncertainty realization. To illustrate this feature of the framework, the time of uncertainty realization for each of the five jobs of the smallest instance (5Jobs 16Hrs) considered in this case study are presented in the Table 3.2. As the time of uncertainty realization depends on the number of tasks preceding the imperfect task in the path of a job and their task processing times, for some jobs, it takes longer time for the uncertainty to realize. For shorter time horizons, this may result in the imperfect task being operated maybe once or twice. As the time horizon increases, the chances of the imperfect task being operated for multiple times also increases. This could be one contributing factor for observing improved VSS when longer time horizons were considered.

The computational times required to obtain the optimal solution for various instances are provided in Table 3.3. As the number of jobs, scenarios and the time horizon increases, the model size increases and results in higher computational times, as shown in Table 3.3.

Although the model could solve for instances with 25 jobs and 50 scenarios for a time horizon of 24hrs, a further increase in the number of jobs or scenarios could be challenging to solve. For instance, the current model (with the specified computational machine settings)

Table 3.2: Time of uncertainty realization for different jobs in the instance 5Jobs 16Hrs

Job	Time of uncertainty realization (minutes)
Job 1	480
Job 2	600
Job 3	900
Job 4	780
Job 5	660

Table 3.3: Computational times of different instances

Instance	Computational Time (sec)		
	5 scenarios	25 scenarios	50 scenarios
5Jobs 16Hrs	24.99	73.89	143.5
5Jobs 24Hrs	50.16	136.39	255.84
15Jobs 16Hrs	119.9	435.28	917.34
15Jobs 24Hrs	208.82	920.36	1997.9
25Jobs 16Hrs	120.49	510.5	1125.6
25Jobs 24Hrs	222.79	976.02	2563.93

was unable to provide a solution for an instance of 40 jobs with 50 scenarios for a time horizon of 24hrs. Introduction of decomposition strategies or relaxation strategies may be required to solve such larger instances. Using such strategies may also further reduce the computational time for the current instances. As the main objective of this study was to address the challenge of enforcing implicit non-anticipativity for a type II endogenous uncertainty, introducing decomposition strategies are considered outside the scope of this work.

To further elucidate the potential benefits of the proposed approach, an estimate of the number of auxiliary binary variables that would have been necessary to enforce non-anticipativity, if the non-anticipativity would had been enforced via introduction of auxiliary binary variables is provided next. The information available in the literature to estimate the number of auxiliary binary variables is used for the estimation. In order to ensure non-anticipativity, the following logic condition has to be enforced [19]:

$$\{t < t^{s,s^{'}}\} \implies \{C_{t,s} = C_{t,s^{'}}\} \quad \forall s, s^{'}, t$$

where, $t^{s,s^{'}}$ is the time period/time point at which two scenarios s and $s^{'}$ become distinguishable and $\boldsymbol{C_{t,s}}$ represents the vector of optimization decisions at time period t in a scenario s.The conversion of the above logic condition to MIP constraints can be achieved by introducing $\mathrm{O}(|S|^2|T|)$ auxiliary binary variables (before applying any scenario reduction properties or strategies) and $\mathrm{O}(|S|^2|T|M)$ constraints [19, 60], where $|S|$ represents the total number of scenarios and $|T|$ represents the total number of time periods or time points and M represents the cardinality of vector $\boldsymbol{C_{t,s}}$. Let us simplify the above expression even further to include any other possibilities and consider that at least $\mathrm{O}(|S||T|)$ auxiliary binary variables and $\mathrm{O}(|S||T|M)$ constraints will be required to enforce non-anticipativity via this approach. Note that the above logic is with respect to a single job, i.e. if there are multiple jobs and each job has a different time of uncertainty realization, then the time at which two scenarios become distinguishable would be different for different jobs. To account for such job specific time of uncertainty realizations, the number of auxiliary binary variables based on the simplified case would be $\mathrm{O}(|S||T||I|)$, where I represents

Table 3.4: Estimated number of auxiliary binary variables

Instance	Number of auxiliary binary variables
5Jobs 16Hrs, 50 scenarios	4,250
5Jobs 24Hrs, 50 scenarios	6,250

the number of jobs. Based on the simplified representation, if the approach involving the binary variables were to be applied for this current case study, the number of required auxiliary binary variables for various instances are provided in Table 3.4. From the table, it can be observed that for one of the small instances considered in this case study (5Jobs, 16hr horizon and 50 scenarios), an estimated 4,250 auxiliary binary variables would have been necessary to add to the model to enforce non-anticipativity, if non-anticipativity was enforced via auxiliary binary variables. In addition, enforcing non-anticipativity using auxiliary binary variables would also require explicit definition of non-anticipativity constraints $O(|S||T|M|I|)$ utilizing each of the introduced binary variable, which can often result in a large computationally expensive model, even for small/medium scale problems. For instance, considering the same instance (5Jobs 16Hrs, 50 scenarios), the value of M would be 32, resulting in an additional 136,000 constraints that would need to be added to the two-stage model to enforce non-anticipativity. As the proposed approach involves implicit non-anticipativity, it eliminates the requirement of these 4,250 additional auxiliary binary variables or 136,000 explicit non-anticipativity constraints.

3.5 Chapter Summary

This chapter presented a novel two-stage stochastic programming approach for scheduling of batch processes with endogenous uncertainty where the time of uncertainty realization is model dependent. The proposed approach accounts for non-anticipativity implicitly without introducing any auxiliary binary variables or explicit non-anticipativity constraints. While developing decomposition/relaxation strategies and scenario reduction techniques constitute most of the literature studies on type II endogenous uncertainty, in this study,

using the proposed approach, it is shown that the model can solve large scale case studies in reasonable times without any decomposition or relaxation strategies. However, using such strategies would further improve the performance of the model and allow solving such larger instances at even smaller computational times. The proposed approach can be adapted to other problems with similar characteristics and those that satisfies the set of constraints (constraints (3.2) – (3.12)) that are key to enforcing non-anticipativity. The computational results from the case study depicts the applicability of the proposed approach in industrial scale optimization problems with type II endogenous uncertainties. Computational studies exhibited a VSS (value of stochastic solution) of 9% on average, confirming the benefits in using the proposed stochastic approach to model uncertainty.

Chapter 4

Node-based Multistage Stochastic Programming Approach

In the previous chapter, a novel two-stage stochastic programming approach for scheduling of batch operations under type II endogenous uncertainty was presented. Even though the proposed two stage approach has its novelties and applications, one limitation of the proposed model is that it assumes that the uncertainty realization remains constant throughout the time horizon and only allows two sets of decisions. This may not often be the case in an actual industrial setting. Governed by various operating factors, uncertainties in industrial processes (e.g., recycle rate, product yield) are unlikely to have only one realization throughout the operating time horizon. The two-stage approach is thus unable to capture the entire feature of the system and restricts the model capability in making efficient decisions. Moreover, the proposed two-stage approach was validated and proved under the assumption that for every job there can be at most one task that is uncertain. Although it is a restrictive assumption, it was required to guarantee implicit non-anticipativity. In this chapter, the previous two-stage approach is improved to propose a node-based multistage model accounting for type II endogenous uncertainty with implicit non-anticipativity, while addressing the two major limitations of the previously proposed two-stage approach (first,multiple realizations for the uncertain parameter; second, possibility of multiple tasks

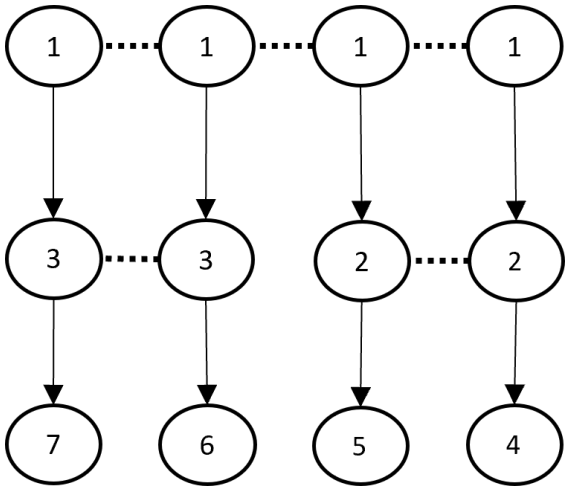
that are uncertain in the same sequence (job)).

This chapter is structured as follows: chapter begins with a discussion on the challenges in simply extending the scenario based two-stage approach into a multistage approach in section 4.1. Section 4.2 provides the discussions on the novel features and considerations in the process network. Section 4.3 presents the proposed node-based stochastic model and section 4.4 presents an alternative process network representation. Finally, section 4.5 presents the results and discussion from the computational experiments and section 4.6 summarizes the contributions from this chapter.

4.1 Scenario-based formulation & challenges

In the previous study, a scenario-based two-stage approach was used to model the type II endogenous uncertainty [24]. In that approach, a path from the root node to the leaf node is considered a scenario and it can be implemented by generating copied variables for each scenario. This can be represented by the alternative representation of the scenario tree. Alternative representation of the scenario tree in Figure 2.2 is shown in Figure 4.1, where each scenario is presented as a distinct path and the dotted lines represent the copied variables [30]. When using such approaches involving copied variables, we need to ensure that these copied variables have the same values for all scenarios until these scenarios become distinguishable [96, 41]. This is usually accomplished by introducing NACs that force these copied variables to be same until the scenarios are distinguishable. However, this implementation becomes complex when type II uncertainties are involved as the time of uncertainty realization is unknown. In the two-stage model, this challenge was addressed by ensuring that the second stage variables for every scenario were set to a value of zero until the time of uncertainty realization and these variables attained non-zero values only to represent the required actions after the realization of uncertainty. However, when considering a multistage approach, this becomes even more challenging as the uncertainty is realized sequentially in every time-period. In case of the scenario-based approach, for the first time-period (i.e., second stage), the non-anticipativity can be ensured if all the second

Figure 4.1: Alternative Scenario representation for a three-stage model (Scenario-based approach)



stage decisions prior to the first uncertainty realization are zero, but for the later time-periods this cannot be enforced as the uncertainty has already been realized in the previous time-period and the corresponding stage decisions possess a non-zero value to account for the uncertainty realized. To ensure that these non-zero stage-wise decisions taken in each time-period is non-anticipative of the future realizations, additional conditions are required at every stage for a scenario-based approach. When the time of uncertainty realization is decision dependent, this may require introduction of auxiliary variables and explicit NACs. Hence, the previously proposed scenario based two-stage stochastic approach with implicit non-anticipativity enforcement cannot be readily extended to a multistage approach as ensuring the multistage variables possess a zero value until the realization of uncertainty at every stage becomes challenging. To address the above challenge, the current study consider a node-based multistage formulation.

4.2 Problem Description

The plant features and the nature of uncertainty remains the same as that of described in sections 3.1 and 3.2. Hence, only the additional information required for the multistage approach are included here.

As mentioned in section 4.1, in this study, the uncertainty is modelled based on a node-based approach, where there is a scenario tree [97] and each node of the scenario tree corresponds to a possible realization of the uncertain parameter. Even though the plant features and the nature of uncertainty remains the same, the assumptions for the process networks are different for the multistage approach. It is no longer assumed that there can be only one imperfect task per job. The current study also allow the process network to be more general and flexible by not imposing any restrictions on the number of forward/backward edges that can leave/enter any task.

The aim is to search for feasible solutions to this problem that consists of a schedule (depending on the time horizon) that accounts for the uncertainty realizations at every stage, with an objective of maximizing the expected throughput. The schedule would provide a set of first-stage decisions irrespective of the realization of uncertainty. It includes decisions such as the batch sizes (number of units processed in each task), the number of units waiting to be processed in each task and the time at which a resource has to be operated. The schedule would also provide a set of multistage decisions that represent the recourse actions enacted at every stage upon the realization of uncertainty, i.e. based on the resolved outcome of the task; these decisions reflect the changes in the number of units transferred to subsequent tasks in the path.

To account for the uncertainty in the multistage framework, the scheduling horizon $[0, H]$ (where H represents the length of the horizon) is partitioned into time periods (time intervals). $M := \{1, \ldots, |M|\}$ represent the set of indices of such time periods $[l_1, u_1], (l_2, u_2], \ldots, (l_{|M|}, u_{|M|}]$ where $l_1 = 0$, $u_{|M|} = H$, $l_{i+1} = u_i, \forall i = 1, \ldots, |M| - 1$. For simplicity, we will abuse notation and refer to $m \in M$ to either denote the index of a time period, or its actual corresponding time interval, where the meaning should be clear based on the context.

The main idea behind splitting the time horizon into $|M|$ time periods is as follows. In the two-stage model [24], it was assumed that once the uncertain parameters ρ^n_{iks} get realized within the scheduling horizon, all other realizations of such uncertain parameters (i.e. recycle rates or yields) will be the same throughout the scheduling horizon. This assumption allowed ensuring that non-anticipativity was satisfied in the two-stage stochastic programming approach. However, this may be a limiting and unrealistic assumption in several applications. Therefore, in the multistage approach, the scheduling horizon is partitioned into time periods where uncertainty parameters in time period m will correspond to nodes in the scenario tree of depth m, as illustrated in Figure 4.2. The new assumption is:

• once the uncertainty is realized in a time-period, it remains constant throughout that time-period, i.e., if an imperfect task completes processing multiple times in the same time-period, it will have the same value of uncertainty realization.

Note that this may still be a limiting assumption; however, this starts becoming a much more reasonable assumption if the expectation is that time periods are short enough that uncertainty would get realized at most once in a time period.

Thus, the scheduling problem with $|M|$ time periods will be modelled as a $(|M|+1)$ stage MILP problem, where the first stage represents the decisions obtained irrespective of the realization of uncertainty and the recourse decisions in time period m represent the $(m+1)$-th stage decisions.

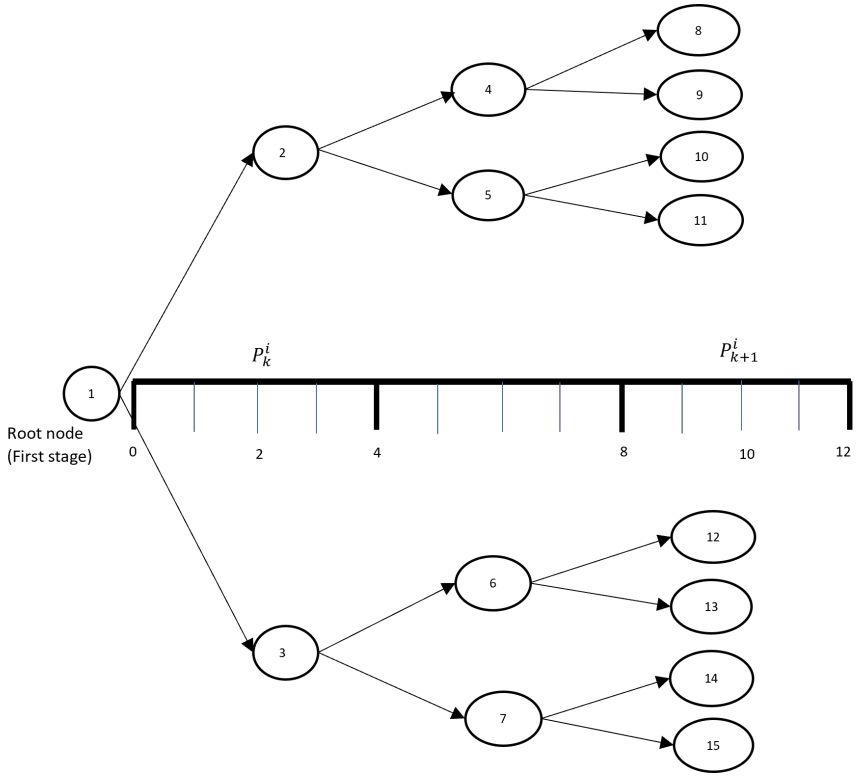
In addition to the time discretization discussed in section 3.1.2, the following sets are defined for the multistage approach.

• $\varepsilon(j)_m$ represent the timepoints for task j that are in time-period $m \in M$.

• For a set of timepoints $\varepsilon(j)$ or $\varepsilon(j)_m$, $\varepsilon'(j)$ or $\varepsilon'(j)_m$ represent the set of indices corresponding to those timepoints.

For illustrative purposes, figure 4.3 shows an example of a task j with $\Delta(j) = 20$ and a scheduling horizon with $H = 480$, divided into 4 time periods. In this example, $\varepsilon(j) = \{0, 20, 40, \ldots, 440, 460, 480\}$, $\varepsilon(j)_1 = \{0, 20, 40, 60, 80, 100, 120\}$ and $\varepsilon(j)_2 = \{140, 160, 180, 200, 220, 240\}$. Also, $\varepsilon'(j) = \{1, 2, 3, \ldots, 24, 25\}$, $\varepsilon'(j)_1 = \{1, 2, 3, 4, 5, 6, 7\}$

Figure 4.2: Illustration of how time periods relate to uncertainty stages.



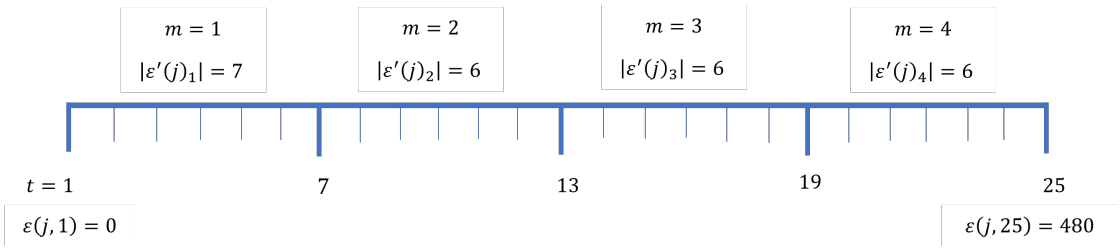
and $\varepsilon'(j)_2 = \{8, 9, 10, 11, 12, 13\}$.

In order to ensure better understanding, few additional notations are defined here that will be needed later on. Let $\alpha(n)$ be the path from root node to the current node, e.g., for Figure 4.2, $\alpha(4) = (1, 2, 4)$, $\alpha(8) = (1, 2, 4, 8)$. Given a node n:

- $p(n)$ - denotes the node in $\alpha(n)$ that precedes n, that is, node n's parent node.
  For example, in Figure 4.2, for $n = 12$, $p(n) = 6$.

- $n(r)$ - denotes the node in $\alpha(n)$ that corresponds to time period m' where m' is the time period containing r. If r is larger than any of the time periods in $\alpha(n)$, then $n(r) = n$.
  For example, in Figure 4.2, for $n = 12$ and $r = 5$, then $n(r) = 6$

Figure 4.3: Example time discretization of a task j with $\Delta(j) = 20$ and a scheduling horizon of 480, divided into 4 time periods



4.3 Proposed multi-stage model

As discussed in Section 4.2, a node-based approach is considered to model uncertainty and each time-period will correspond to a certain stage in the multi-stage stochastic program. With such an approach, all we need to ensure here is that, in every time-period, the decisions taken for nodes with the same preceding node has to be the same until the time of uncertainty realization in that time-period. As this study deals with a type II endogenous uncertainties where the time of uncertainty realization is not known *a priori*, enforcing these non-anticipativity constraints remains a challenging problem. In order to address this challenge, the multistage variables are defined and the formulation is designed in such a way that the model would implicitly ensure non-anticipativity at every stage. The key features of the proposed node-based approach include how the flow decision variables are defined and the subsequent design of flow balance constraints. The key idea here is that the final action that can be implemented at any time is divided into a combination of three decisions: 1) a decision that can be made irrespective of any realization of uncertainty throughout the time horizon, 2) a decision that can be made in the current time-period based on the past realization of uncertainty but irrespective of any current or future realizations 3) a decision that can be made in the current time-period after the realization of the uncertain parameter. These decisions are represented in the model by the first stage and multistage decision variables. In the following discussion, a high level overview of what is the idea behind how the stages relate to each other is presented first. A detailed explanation of all

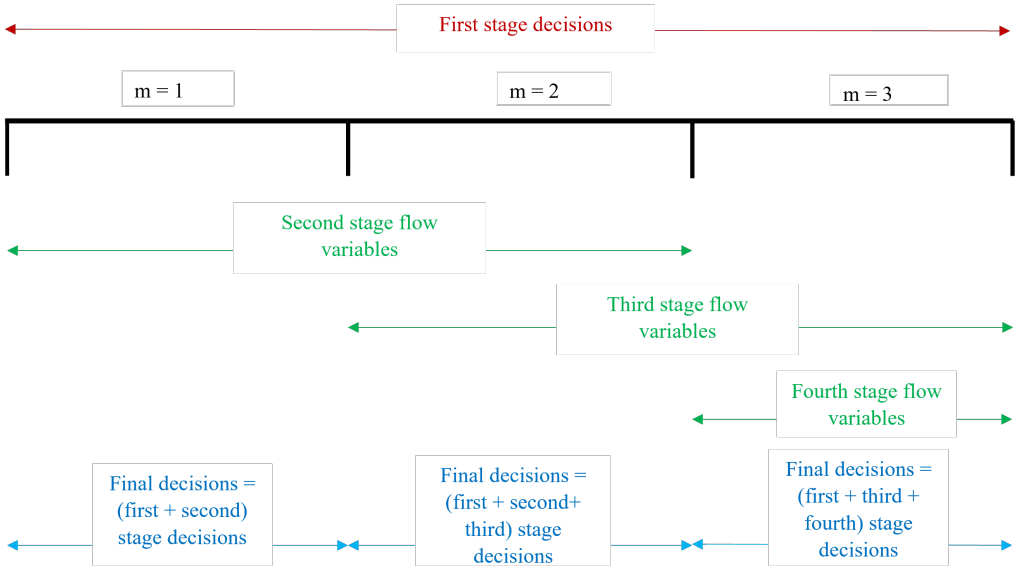
model variables and constraints will be given later.

The key decision variables of the node-based approach are as follows (note that for simplicity and to focus on better understanding of the proposed approach, the variables are represented with just the time and node indices in the discussion below; complete representation of variables are presented in the later sections):

- **First stage flow variables** (x_t, y_t, z_t)**:** these variables represent those decisions that are made before the realization of uncertainty and that remain feasible irrespective of any uncertainty realization. They are defined for the complete time-horizon.
- **Multistage decision variables:** The multistage decision variables consists of the following:
  - Flow variables (x_t^n, y_t^n): these variables represent the necessary recourse actions at a timepoint t based on the realization of uncertainty at node n. These variables are defined for two consecutive time periods, more precisely, if node n is at time period m, then these variables will be defined for time period m and $m+1$ (if it exists). The idea is that these variables provide the required recourse actions based on the uncertainty realization in the current time-period and continue providing recourse actions until the uncertainty is realized in the subsequent time-period.
  - Final implementable decisions (W_t^n, V_t^n): these decisions represent the final action that needs to be taken at time point t and node n. These final implementable decisions include the first stage decision and the multistage decisions active in the current time-period. For a node $n \in N_m$, the decision variables that are active in time-period m include the first stage decisions that are made irrespective of any realization throughout the time horizon, the decision variables from node n ($m+1$-th stage decisions) and the decision variables from node n_{m-1} (m-th stage decisions).

Figure 4.4 shows a diagram of which variables are defined in each time period.

Figure 4.4: Illustration of how multistage variables are defined in the node-based approach



4.3.1 Detailed decision variables

A complete definition of all variables that are involved in the multistage model is given below.

The first-stage variables are as follows:

- z_{jt}: number of resources to be operated for a task $j \in J$ at timepoint $t \in \varepsilon'(j)$,
- y_{ikt}: number of units from a job $i \in I$ to be processed in the task P_k^i at time $t \in \varepsilon'(P_k^i)$, for $k = 1, \ldots, q_i$,
- x_{ikt}: number of units from a job $i \in I$ waiting to be processed in the task P_k^i at time $t \in \varepsilon'(P_k^i)$, for $k = 1, \ldots, q_i$.

The multi-stage flow variables for a node $n \in N_m$ are as follows:

- y_{ikt}^n: number of units from a job $i \in I$ to be processed in the task P_k^i at time $t \in \varepsilon'(P_k^i)_m \cup \varepsilon'(P_k^i)_{m+1}$, for $k = 1, \ldots, q_i$, based on uncertainty realization of node n,

- x^n_{ikt}: number of units from a job $i \in I$ waiting to be processed in the task P^i_k at time $t \in \varepsilon'(P^i_k)_m \cup \varepsilon'(P^i_k)_{m+1}$, for $k = 1, \ldots, q_i$, based on uncertainty realization of node n.

(note that, in the above notation, if $m = |M|$, $(P^i_k)_{m+1} = \emptyset$).

The multi-stage final implementable decisions for a node $n \in N_m$ are as follows:

- W^n_{ikt}: number of units from a job $i \in I$ to be processed in the task P^i_k at time $t \in \varepsilon'(P^i_k)_m$, for $k = 1, \ldots, q_i$, based on uncertainty realization of node n,
- V^n_{ikt}: number of units from a job $i \in I$ waiting to be processed in the task P^i_k at time $t \in \varepsilon'(P^i_k)_m$, for $k = 1, \ldots, q_i$, based on uncertainty realization of node n.

These final implemental decisions will be a combination of first stage decisions and the recourse decisions (multi-stage flow variables).

4.3.2 Stochastic Model

The proposed multistage stochastic programming model is presented in this section.

The first constraints that are defined to obtain the decisions irrespective of any value of uncertainty realization are defined as same as that of the two-stage model. The constraints included are (3.1) - (3.3) and (3.6). The detailed discussion on these constraints can be found in sections 3.3.1 and 3.3.2.

Before presenting the constraints involving the multistage variables, note that, to simplify the exposition, the constraints will be presented in their most generic form, with the understanding that any terms which are not defined in them, for certain values of the parameters will become zero. For instance, if a term like $y^{p(n)}_{ikt}$ appears, for some $n \in N_1$, this term would be equal to zero. Likewise, $x^{n(t-1)}_{ikt-1}$ for $t = 1$ would be equal to zero.

Constraints (4.1) represent the initialization constraint for the multistage decision variables. The multistage variables of the first time-period are initialized to zero as there would

not be any uncertainty realizations at the first time point as this study focuses on type II endogenous uncertainty, where the realization depends on an already made model decision.

$$y^n_{ik1} = x^n_{ik1} = 0 \quad , \forall n \in N_1, i \in I, k \in 1..q_i \tag{4.1}$$

Constraints (4.2) - (4.3) provide the final implementable decisions (W^n_{ikt} and V^n_{ikt}) by considering the first stage flow decisions and the multistage recourse decisions for every task P^i_k at every time point t and node n.

$$W^n_{ikt} = y^n_{ikt} + y_{ikt} + y^{p(n)}_{ikt} \quad , \forall m \in M, n \in N_m, i \in I, k \in 1, \ldots, q_i, t \in \varepsilon'(P^i_k)_m \tag{4.2}$$

$$V^n_{ikt} = x^n_{ikt} + x_{ikt} + x^{p(n)}_{ikt} \quad , \forall m \in M, n \in N_m, i \in I, k \in 1, \ldots, q_i, t \in \varepsilon'(P^i_k)_m \tag{4.3}$$

Constraints (4.4) represent the total flow balance at any time-point throughout the time horizon. It can be seen as a version of the constraint (3.6) for the W^n_{ikt} and V^n_{ikt} variables, but now considering the actual realization of uncertainty.

$$\begin{aligned}(x^{n(t-1)}_{ikt-1} + x^{p(n(t-1))}_{ikt-1} + x_{ikt-1}) + \sum_{k' \in N^-_G(P^i_k)} \sum_{r \in \theta(i,k,k',t)} \rho^n_{ik'k}(y^{n(r)}_{ik'r} + y^{p(n(r))}_{ik'r} + y_{ik'r}) = \\ (x^n_{ikt} + x^{p(n)}_{ikt} + x_{ikt}) + (y^n_{ikt} + y^{p(n)}_{ikt} + y_{ikt}) \\ , \forall m \in M, n \in N_m, i \in I, k \in 1, \ldots, q_i, t \in \varepsilon'(P^i_k)_m\end{aligned} \tag{4.4}$$

Before presenting the next set of constraints, the motivation behind them are discussed here. The idea behind the first stage flow-conservation constraints (3.6) was to make a plan throughout the whole time horizon based on what was certain to happen, in such a way that any first stage decisions could be implemented. Now, when a certain node $n \in N_m$ is considered, at any point in time during time period m, there are already a lot more uncertainty that has been realized before that time period, namely the uncertainty of all nodes preceding n in $\alpha(n)$. Therefore, the idea is that, at this point, it is possible to

make a plan throughout time period m based on any realizations of uncertainty that have happened before node n. However, note that any such plan cannot involve variables x^n, y^n since these are variables defined for the current node n for which it is not known when uncertainty will be realized yet. The idea then is to use the variables $x^{p(n)}, y^{p(n)}$ to make such a plan for period m, since these are variables that were defined for uncertainties that have been realized already. Making this plan for time period m was precisely the purpose of defining variables for two consecutive time periods as was shown in Figure 4.4 and was discussed at the beginning of this section. Similar to what was done in constraints (3.6), the plan will take into account only transfer rates that are sure will happen. Thus, the term $\beta^n_{ik'k}$ is defined as zero for an imperfect task and is equal to the actual rate $\rho^n_{ik'k}$ for the perfect tasks (since there is no uncertainty associated with those). Formally, $\beta^n_{ik'k}$ is defined as:

$$\beta^n_{ik'k} = \begin{cases} 0, & \text{if } P^i_{k'} \in \Im_i \\ \rho^n_{ik'k} & \text{otherwise} \end{cases} \quad ,\forall m \in 1,\dots,M, n \in N_m, i \in I, k \in 1,\dots,q_i, t \in \varepsilon'(P^i_k)_m$$

The flow balance equations are defined considering the above definition; this can be seen as versions of (4.4) substituting ρ by β and removing all terms that use x^n or y^n variables, and also the first stage variables (since these were already considered in (3.6)). This will therefore give a plan that can be followed based on just the uncertainty that is already known at time period m. Formally this leads to the following constraints.

$$\begin{aligned} x^{p(n)}_{ikt} + y^{p(n)}_{ikt} = x^{p(n(t-1))}_{ikt-1} + x^{n(t-1)}_{ikt-1} + \\ \sum_{k' \in N^-_G(P^i_k)} \sum_{r \in \theta(i,k,k',t)} \beta^n_{ik'k}(y^{n(r)}_{ik'r} + y^{p(n(r))}_{ik'r}), \\ \forall m \in 2..M, n \in N_m, i \in I, k \in 1,\dots,q_i, t = \left(\varepsilon'(P^i_k)_m\right)_1 \end{aligned} \tag{4.5}$$

$$x_{ikt}^{p(n)} + y_{ikt}^{p(n)} = x_{ikt-1}^{p(n(t-1))} + \sum_{k' \in N_G^-(P_k^i)} \sum_{r \in \theta(i,k,k',t)} \beta_{ik'k}^n y_{ik'r}^{p(n(r))} + \sum_{k' \in N_G^-(P_k^i)} \sum_{r \in \theta(i,k,k',t) \setminus \varepsilon'(P_k^i)_m} \beta_{ik'k}^n y_{ik'r}^{n(r)}, \quad (4.6)$$
$$\forall m \in 2..M, n \in N_m, i \in I, k \in 1, \ldots, q_i, t \in \varepsilon'(P_k^i)_m \setminus \left(\varepsilon'(P_k^i)_m\right)_1$$

Note that (4.5) is just defined for the first timepoint of period m and (4.6) is defined for all other timepoints in period m. The reason for this difference is because different variables may be or not defined as x^n and y^n variables depending on t.

The next set of constraints are capacity constraints, which enforce that the total number of units from all job $i \in I : j \in P^i$ that can be operated in a task j (multitasking) at time point t should not exceed the total capacity of all the resources available at that time point to perform the operation.

$$\sum_{i \in I} \sum_{k \in 1,\ldots,q_i : P_k^i = j} W_{ikt}^n \leq z_{jt} C_j \quad , \forall m \in M, n \in N_m, j \in J, t \in \varepsilon'(j) \quad (4.7)$$

Finally, the objective function (4.8) is defined in terms of the final implementable decisions (W_{ikt}^n) that accounts for both the first (y_{ikt}) and second stage (y_{ikt}^n) variables involved in the model, and the probabilities ψ_n associated with each node. Note that the ultimate goal is to maximize throughput (number of units that finished their complete path of tasks), but due to a short-term scheduling horizon, many units may not actually be able to fully finish their path. Thus, the following objective function, which has been studied and validated previously [10] to add incentive for units to go as close as possible to finishing completely by providing higher weights to the final tasks and smaller weights to the initial tasks.

$$\max \sum_{m \in M} \sum_{n \in N_m} \sum_{i \in I} \sum_{k=1}^{q_i} \sum_{t \in \varepsilon'(P_k^i)_m} \left(\frac{k}{q_i} W_{ikt}^n\right) \psi_n \quad (4.8)$$

The final model is therefore:

$$\begin{aligned} \max \quad & \sum_{m\in M}\sum_{n\in N_m}\sum_{i\in I}\sum_{k=1}^{q_i}\sum_{t\in\varepsilon'(P_k^i)_m}(\tfrac{k}{q_i}W_{ikt}^n)\psi_n \\ \text{s.t.} \quad & (3.1)-(3.3),(3.6),(4.1)-(4.7) \\ & x,y,W,V\geq 0 \\ & z\geq 0 \text{ and integer} \end{aligned} \tag{M}$$

The key development in model (M) is to be able to choose the right set of constraints so that there is no need to enforce nonanticipativity explicitly by using additional binary variables. The careful model design is, therefore, a key contribution, and the formal statement of the theorem that guarantees that it achieves its purpose is below.

Theorem 2. *Assume* $(x,y,W,V)\geq 0$ *satisfies to constraints* (3.1) - (3.3),(3.6), (4.1)-(4.7). *Define*

$$t'_{ikm} := \min\left\{ t\in\varepsilon'(P_k^i)_m : \begin{array}{c} y_{ik'r}^{n(r)} \vee y_{ik'r}^{p(n(r))} \vee y_{ik'r} > 0, \\ \textit{for some} \quad k'\in N_G^-(P_k^i)\cap\Im_i, r\in\theta(i,k,k',t) \end{array} \right\} \tag{4.9}$$

that is, t'_{ikm} *is the first timepoint of task* k *that comes after an uncertain task in* $N_G^-(P_k^i)$ *has finished during period* m. *Define*

$$\tau_{im} := \min\left\{\varepsilon(t'_{ikm}) : k=1,\ldots,q_i\right\}, \tag{4.10}$$

that is, the first time when any uncertain task in P^i *has finished during period* m.

Then $\forall n,n'\in N_m$ *such that* $p(n)=p(n')$*:*

$$W_{ikt}^n = W_{ikt}^{n'} \quad ,\forall m\in M, n\in N_m, i\in I, k\in 1..q_i, t\in\varepsilon'(P_k^i)_m : \varepsilon(P_k^i,t)<\tau_{im} \tag{4.11}$$

$$V_{ikt}^n = V_{ikt}^{n'} \quad ,\forall m\in M, n\in N_m, i\in I, k\in 1..q_i, t\in\varepsilon'(P_k^i)_m : \varepsilon(P_k^i,t)<\tau_{im} \tag{4.12}$$

Note that, in the statement above, τ_{im} represents the time at which uncertainty is realized for job i in time period m. Also, if $\tau_{im}=\infty$ (for instance if all y variables are

zero throughout time period m), this means that nonanticipativity holds throughout time period m.

Proof. Recall the flow balance constraints.
The first-stage flow balance constraints (3.6), will be rewritten as:

$$x_{ikt} + y_{ikt} - x_{ikt-1} - \sum_{k' \in N_G^-(P_k^i)} \sum_{r \in \theta(i,k,k',t)} \rho_{ik'k} y_{ik'r} = 0 \quad , \forall i \in I, k \in 1, \ldots, q_i, t \in \varepsilon'(P_k^i) \setminus \{1\} \tag{4.13}$$

The multistage flow balance constraints (4.4), (4.5) and (4.6) will be rewritten as:

$$\begin{aligned} x_{ikt} + y_{ikt} - x_{ikt-1} - \sum_{k' \in N_G^-(P_k^i)} \sum_{r \in \theta(i,k,k',t)} \rho^n_{ik'k} y_{ik'r} + \\ x^{p(n)}_{ikt} + y^{p(n)}_{ikt} - x^{p(n(t-1))}_{ikt-1} - x^{n(t-1)}_{ikt-1} - \sum_{k' \in N_G^-(P_k^i)} \sum_{r \in \theta(i,k,k',t)} \rho^n_{ik'k} y^{p(n(r))}_{ik'r} + \\ x^n_{ikt} + y^n_{ikt} - \sum_{k' \in N_G^-(P_k^i)} \sum_{r \in \theta(i,k,k',t)} \rho^n_{ik'k} y^{n(r)}_{ik'r} = 0 \\ , \forall m \in M, n \in N_m, i \in I, k \in 1, \ldots, q_i, t \in \varepsilon'(P_k^i)_m \end{aligned} \tag{4.14}$$

$$\begin{aligned} x^{p(n)}_{ikt} + y^{p(n)}_{ikt} - x^{p(n(t-1))}_{ikt-1} - x^{n(t-1)}_{ikt-1} - \\ \sum_{k' \in N_G^-(P_k^i)} \sum_{r \in \theta(i,k,k',t)} \beta^n_{ik'k} (y^{n(r)}_{ik'r} + y^{p(n(r))}_{ik'r}) = 0, \\ \forall m \in 2..M, n \in N_m, i \in I, k \in 1, \ldots, q_i, t = \left(\varepsilon'(P_k^i)_m\right)_1 \end{aligned} \tag{4.15}$$

$$\begin{aligned} x^{p(n)}_{ikt} + y^{p(n)}_{ikt} - x^{p(n(t-1))}_{ikt-1} - \\ \sum_{k' \in N_G^-(P_k^i)} \sum_{r \in \theta(i,k,k',t)} \beta^n_{ik'k} y^{p(n(r))}_{ik'r} - \sum_{k' \in N_G^-(P_k^i)} \sum_{r \in \theta(i,k,k',t) \setminus \varepsilon'(P_k^i)_m} \beta^n_{ik'k} y^{n(r)}_{ik'r} = 0, \\ \forall m \in 2..M, n \in N_m, i \in I, k \in 1, \ldots, q_i, t \in \varepsilon'(P_k^i)_m \setminus \left(\varepsilon'(P_k^i)_m\right)_1 \end{aligned} \tag{4.16}$$

The following claim will be proved for a fixed m, by induction on t:

Claim: $x^n_{ikt} = y^n_{ikt} = 0$ **for all** $t \in \varepsilon'(P^i_k)_m : \varepsilon(P^i_k, t) < \tau_{im}$

Note that if $\varepsilon(P^i_k, t) \geq \tau_{im}, \forall t \in \varepsilon'(P^i_k)_m$, there is nothing to be proved. Hence, only $t \in \varepsilon'(P^i_k)_m : \varepsilon(P^i_k, t) < \tau_{im}$ is considered here.

BASE CASE: $t = (\varepsilon'(P^i_k)_m)_1$

If $m = 1$, then t will be the very first timepoint in $\varepsilon(P^i_k)$, that is, $t = 1$. Therefore (4.1) implies $x^n_{ikt} = y^n_{ikt} = 0$, so the claim is true.

If $m > 1$, then replacing (4.15) and (4.13) in (4.14) yields:

$$\sum_{k' \in N^-_G(P^i_k)} \sum_{r \in \theta(i,k,k',t)} (\rho_{ik'k} - \rho^n_{ik'k})\, y_{ik'r} + \sum_{k' \in N^-_G(P^i_k)} \sum_{r \in \theta(i,k,k',t)} (\beta^n_{ik'k} - \rho^n_{ik'k})\, y^{p(n(r))}_{ik'r} + \sum_{k' \in N^-_G(P^i_k)} \sum_{r \in \theta(i,k,k',t)} (\beta^n_{ik'k} - \rho^n_{ik'k})\, y^{n(r)}_{ik'r} + x^n_{ikt} + y^n_{ikt} = 0 \quad (4.17)$$
$$, \forall n \in N_m, i \in I, k \in 1, \ldots, q_i,$$

Now note that, if $k' \in N^-_G(P^i_k) \setminus \Im_i$, then $\beta^n_{ik'k} = \rho^n_{ik'k} = \rho_{ik'k}$. Moreover, if $k' \in N^-_G(P^i_k) \cap \Im_i$, then $y_{ik'r} = y^{p(n(r))}_{ik'r} = y^{n(r)}_{ik'r} = 0$, for all $r \in \theta(i,k,k',t)$ by definition of τ_{im}.

Therefore all the summation terms in (4.17) will be equal to zero, so those equations can be rewritten as:

$$x^n_{ikt} + y^n_{ikt} = 0 \quad , \forall n \in N_m, i \in I, k \in 1, \ldots, q_i, \quad (4.18)$$

Now (4.18) and the nonnegativity constraints imply $x^n_{ikt} = y^n_{ikt} = 0$.

INDUCTION STEP: $t > (\varepsilon'(P^i_k)_m)_1$, $\varepsilon(P^i_k, t) < \tau_{im}$

For $m = 1$, (4.14) can be simplified as:

$$x_{ikt} + y_{ikt} - x_{ikt-1} - x^{n(t-1)}_{ikt-1} - \sum_{k' \in N_G^-(P_k^i)} \sum_{r \in \theta(i,k,k',t)} \rho^n_{ik'k} y_{ik'r} + x^n_{ikt} + y^n_{ikt} - \sum_{k' \in N_G^-(P_k^i)} \sum_{r \in \theta(i,k,k',t)} \rho^n_{ik'k} y^{n(r)}_{ik'r} = 0 \quad (4.19)$$
$$, n \in N_1, i \in I, k \in 1, \ldots, q_i$$

Replacing (4.13) in (4.19) yields:

$$-x^{n(t-1)}_{ikt-1} - \sum_{k' \in N_G^-(P_k^i)} \sum_{r \in \theta(i,k,k',t)} \left(\rho^n_{ik'k} - \rho_{ik'k}\right) y_{ik'r} + x^n_{ikt} + y^n_{ikt} - \sum_{k' \in N_G^-(P_k^i)} \sum_{r \in \theta(i,k,k',t)} \rho^n_{ik'k} y^{n(r)}_{ik'r} = 0 \quad (4.20)$$
$$, n \in N_1, i \in I, k \in 1, \ldots, q_i,$$

But then, in this case, $n(t-1) = n$, and $n(r) = n$ for all $r \in \theta(i,k,k',t)$. Thus, by induction $x^{n(t-1)}_{ikt-1} = y^{n(r)}_{ik'r} = 0$. But also, if $k' \in N_G^-(P_k^i) \setminus \Im_i$, then $\rho^n_{ik'k} = \rho_{ik'k}$. Moreover, if $k' \in N_G^-(P_k^i) \cap \Im_i$, then $y_{ik'r} = 0$, for all $r \in \theta(i,k,k',t)$ by definition of τ_{im}. Thus, (4.20) can be simplified to

$$x^n_{ikt} + y^n_{ikt} = 0 \quad , n \in N_1, i \in I, k \in 1, \ldots, q_i, \quad (4.21)$$

which together with nonnegativity implies the result.

Finally, for $m > 1$,

Replacing (4.16) and (4.13) in (4.14) yields:

$$\begin{aligned}\sum_{k'\in N_G^-(P_k^i)}\sum_{r\in\theta(i,k,k',t)}(\rho_{ik'k}-\rho^n_{ik'k})\,y_{ik'r}+\sum_{k'\in N_G^-(P_k^i)}\sum_{r\in\theta(i,k,k',t)}(\beta^n_{ik'k}-\rho^n_{ik'k})\,y^{p(n(r))}_{ik'r}\\ -x^{n(t-1)}_{ikt-1}+\sum_{k'\in N_G^-(P_k^i)}\sum_{r\in\theta(i,k,k',t)\setminus\varepsilon'(P_k^i)_m}(\beta^n_{ik'k}-\rho^n_{ik'k})\,y^{n(r)}_{ik'r}+\\ x^n_{ikt}+y^n_{ikt}-\sum_{k'\in N_G^-(P_k^i)}\sum_{r\in\theta(i,k,k',t)\cap\varepsilon'(P_k^i)_m}\rho^n_{ik'k}y^{n(r)}_{ik'r}=0\\ ,\forall n\in N_m, i\in I, k\in 1,\dots,q_i,\end{aligned}\tag{4.22}$$

Now note that, if $k'\in N_G^-(P_k^i)\setminus\Im_i$, then $\beta^n_{ik'k}=\rho^n_{ik'k}=\rho_{ik'k}$. Moreover, if $k'\in N_G^-(P_k^i)\cap\Im_i$, then $y_{ik'r}=y^{n(r)}_{ik'r}=y^{p(n(r))}_{ik'r}=0$, for all $r\in\theta(i,k,k',t)$ by definition of τ_{im}. Therefore (4.22) can be rewritten as:

$$-x^{n(t-1)}_{ikt-1}+x^n_{ikt}+y^n_{ikt}-\sum_{k'\in N_G^-(P_k^i)}\sum_{r\in\theta(i,k,k',t)\cap\varepsilon'(P_k^i)_m}\rho^n_{ik'k}y^{n(r)}_{ik'r}=0\quad ,\forall n\in N_m, i\in I, k\in 1,\dots,q_i,\tag{4.23}$$

Once more, in this case, $n(t-1)=n$ and $n(r)=n$ for all $r\in\theta(i,k,k',t)\cap\varepsilon'(P_k^i)_m$. Therefore by induction $y^{n(r)}_{ik'r}=x^{n(t-1)}_{ikt-1}=0$. Thus

$$x^n_{ikt}+y^n_{ikt}=0\quad ,\forall n\in N_m, i\in I, k\in 1,\dots,q_i,\tag{4.24}$$

and so nonnegativity once more implies the result of the claim. □

Now, to finish the proof of Theorem 2, note that using the above claim, (4.2) and (4.3) can be rewritten as:

$$W^n_{ikt}=y_{ikt}+y^{p(n)}_{ikt}\quad ,\forall m\in M, n\in N_m, i\in I, k\in 1,\dots,q_i, t\in\varepsilon'(P_k^i)_m:\varepsilon(P_k^i,t)<\tau_{im}\tag{4.25}$$

$$V^n_{ikt} = x_{ikt} + x^{p(n)}_{ikt} \quad ,\forall m \in M, n \in N_m, i \in I, k \in 1, \ldots, q_i, t \in \varepsilon'(P^i_k)_m : \varepsilon(P^i_k, t) < \tau_{im} \tag{4.26}$$

Thus, if two nodes n, n' have $p(n) = p(n')$, their W, V variables will be equal up until uncertainty gets realized in time period m. □

4.4 State Task Network Representation

In this thesis, to conveniently define the incoming and outgoing streams of a task and the assumptions considered, the standard graphs are used to represent the process network. However, the process network can be denoted using state task network (STN) representation as well. The key difference of graph representation from STN representation is that it does not involve an explicit representation for the units (materials) inventory.

The proposed model can be adapted to an STN representation by performing the following steps:

• Introduce the concept of states and replacing the first stage and multistage flow decisions, i.e., replace x_{ikt}(the number of units waiting to be processed in the task P^i_k) by the state variable $S_{is't}$ (where s' represents the state) and y_{ikt} (the number of units to be processed in the task P^i_k) by the batch variable B_{ikt}. Similar replacements need to be made for the multistage decisions as well.

• As shown in the figure, in an STN representation, if there are b tasks, there exist $(b+1)$ states. Therefore, introduce an extra constraint for the final state.

For better understanding, Figure 4.6 presents the corresponding STN representation of the example job given in Figure 4.5. This figure considers an STN representation of job i with six tasks in its path (recipe). There are seven states (represented by circles) including the final product state. To illustrate the transformation of the proposed approach to an STN representation, a case study presented in section 4.5.2 was implemented using the STN representation.

Figure 4.5: Illustration of a graph $G(i)$ of a job i with six tasks in its path

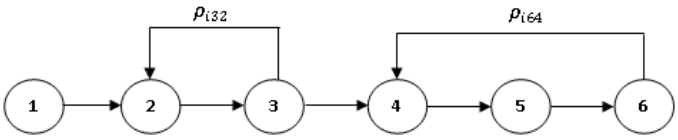
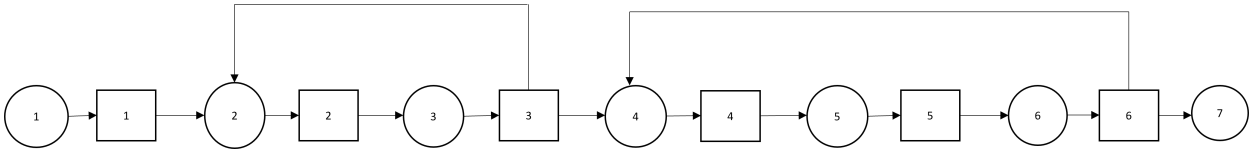


Figure 4.6: Corresponding STN illustration of a job i with six tasks in its path (recipe)



4.5 Computational Experiments

In this section the results from the computational experiments conducted using the proposed multistage stochastic framework are presented. The framework was tested using the industrial case study from ASI sector. Though this model has been developed focusing on the multijob plant from the analytical services sector, it can be easily adapted for other applications. In order to show the adaptive capabilities of the proposed approach, the multistage model was also tested using two other case studies chosen from the literature. These two case studies were chosen as they would allow us to represent those production systems where type II uncertainties such as variations in recycle rate or product yield may occur frequently. Following the discussion on the case studies and the corresponding results, the results from the comparison study conducted between the proposed approach and an approach involving auxiliary binary variables to define the NACs are presented. The comparison study is conducted using the large scale industrial case study. All the computational experiments were performed on an i7-3.40GHz Windows machine with 16GB RAM using IBM ILOG CPLEX Optimizer 12.7.1.

Before presenting the case studies, a detailed description is provided first on how the evaluation of benefits in using the multistage stochastic programming has been carried out.

4.5.1 Evaluation of multistage stochastic programming

The benefits of using a stochastic programming model can be quantified by the Value of the Stochastic Solution (VSS). The VSS is the difference between the expected values obtained from implementing the solution yielded by the stochastic formulation and the solution yielded by a deterministic formulation that substitutes the uncertain parameters with their nominal values [14]. The expected value is calculated by solving the problem, implementing the here and now decisions, and evaluating the decisions using a large number of uncertainty realizations. To be more specific, for each case study, the multistage stochastic model was formulated, and the expected value is calculated by carrying out the following steps:

(i) Solve the stochastic problem for a randomly chosen set of nodes.

(ii) Obtain the multistage decisions.

(iii) Fix the obtained decisions of each time-period before the uncertainty realization and evaluate them over a large number of events, n_e. An event here refers to one set of sequential nodes throughout the time horizon with one unique uncertainty realization in each time-period. The model is then solved individually for each event. Note that, though the multistage decisions are obtained by solving the stochastic problem for a chosen set of discrete realizations (i.e., step i), for the evaluation of these decisions, the events are chosen randomly from all the possible realizations, in other words, the evaluation is done using out-of-sample events.

(iv) Obtain the mean throughput value of all the simulations (considering the above example, mean value of all the n_e simulations).

The above evaluation procedure is based on the random set of nodes that were chosen initially in step (i). To attain a more reasonable analysis, the above procedure is repeated for multiple sets of nodes, n_s and the average throughput of all the random sets is considered as the expected value of the stochastic formulation, which is further used in the calculation of the VSS. Similarly, the expected value of the deterministic formulation can be obtained using the same procedure with the following differences:

(a) Solve the deterministic formulation by setting the uncertain parameters to their nominal values.

(b) Allow positive and non-positive values for the multistage decision variables.
Note that the deterministic model is solved using the nominal values and the actual realization could either be greater or lesser than the nominal value. Therefore, in order to ensure the feasibility of the system, the model is allowed to undo some of the decisions made by allowing non-positive values for the multistage decision variables. For example, assume that the deterministic decisions were obtained based on the average value of 25% recycle but the actual realization was 10%. Thus, the recourse action requires to correct the previous decision of 25%, to accommodate this condition, the multistage variables may possess a non-positive value.

(c) Modify the objective function by including an additional penalty term for penalizing the non-positive decisions.
As non-positive variables are allowed, a penalty term is included in the objective function to penalize the non-positive variables in order to restrict the variables to attain a non-positive value only when necessary to ensure feasibility. The objective remains the same as that given in section 4.3.2, i.e., to maximize the expected throughput with an additional term with negative weights for the non-positive decisions. This is referred to as the modified objective function. In order to assure the consistency in the estimation of the expected value for both stochastic and deterministic formulations, the evaluation of the decisions (i.e., step iii of the above procedure) is carried out using the modified objective function. As there is a penalty term in the modified objective function, the focus is on the value of throughput instead of the value of objective function for calculating the VSS so that a fair comparison is made, irrespective of the weights used for the penalty term. Note that this modified VSS [24] is different from the classical VSS, since the latter uses the objective function of the problem and the events used to solve the stochastic program (in-sample), and the former uses a modified objective function and out-of-sample events. Nonetheless, to simplify the language it is referred to as VSS from this point forward.

The evaluation procedure described above is used in estimating the expected values and thereby quantifying the benefits of using the proposed multistage stochastic approach by calculating the VSS for all the three case studies. As the initial set of nodes in step (i) are chosen randomly, for a reasonable analysis, rather than establishing the results and observations based on one chosen set of nodes, for all three case studies, n_s random sets are chosen (e.g.,n_s=500) and for each set the whole evaluation procedure (with randomly chosen 1500 events, i.e., $n_e = 1500$) is repeated.

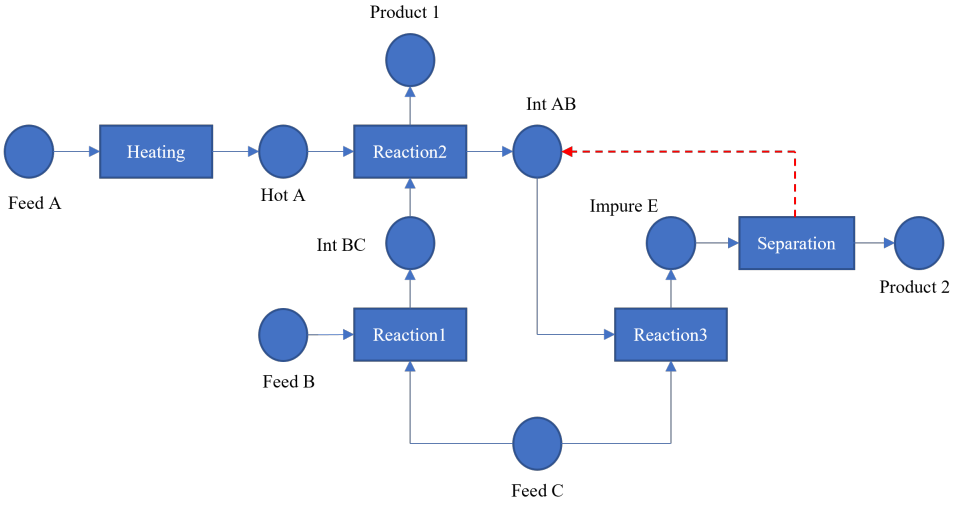
4.5.2 Results for case study I

The authors of the study,[3] presented a case study involving the production of two products 1 and 2 from three different feed stocks A, B and C that follows a specific recipe. The production process network is presented in Figure 4.7. This process represents a single job process network, as shown in Figure 4.7. Those authors presented the process network as a deterministic case study. In order to demonstrate the applicability of the proposed approach, an uncertainty is assumed in the recycle rate of intermediate AB (Int AB) that are recycled back from the separation task (the recycle stream is highlighted in red in the Figure 4.7). Six possible discrete realizations were considered for the recycle rate from task separation - {0,0.05,0.1,0.15,0.2,0.25}.

To incorporate the features of the present case study, the proposed framework was modified to allow a single resource to perform multiple tasks and thereby allowing different task completion times with respect to the task being processed by the resource. However, these modifications in the constraints does not affect the non-anicipativity enforcement and therefore can be easily adapted to accommodate the required modifications.

Following the modifications involved in transforming the proposed framework into a State Task Network (STN) form mentioned in section 4.4, the materials (feeds, intermediates and products), represented by circles in Kondili's STN representation are explicitly considered in the network as states. The first stage flow decisions here include $S_{1s't}$, the state variable (the number of materials in each state s' at time t) and B_{1kt}, the batch variable (the number of materials processed in task k at time t). Similarly, the second

Figure 4.7: The production process network 1 from [3], considering uncertainty in the recycle stream represented by dashed lines from the imperfect task separation



stage flow decisions include $B^n_{1kt}, S^n_{1s't}$ for every node n. In an STN representation, the flow balances are considered based on states, therefore, an additional constraint was included for the final states (product 1 and product 2).

The stochastic model was thus formulated, and the model was solved for a scheduling horizon of 12 hrs. The process parameters used in the model such as equipment capacity, storage capacity, processing time and initial inventory were obtained from [3]. As the processing times of the tasks involved range from 1-2hrs, a uniform discretization (UD) of time was used for this case study, with timestep ($\Delta(j)$) equal to 1hr.

The time horizon was divided into multiple time periods and the problem was solved for two, three and four stages with 3, 9 and 27 nodes respectively in the first, second and third time periods. For this small case study with single job and 5 tasks, the model was solved in less than 2 seconds. For each case, the VSS was calculated using the procedure explained above. The results for this case study are presented in Table 4.1. The mean value of the evaluation of the 500 random sets are considered to calculate the VSS. The frequency distribution of the throughput value obtained after the evaluation of the 500

Table 4.1: VSS for the production process network 1

	Average Throughput	VSS(%)
Deterministic	248	
Two Stage	262	5.6
Three Stage	272	9.6
Four Stage	276	11.29

randomly chosen sets is presented in Figure 4.8. The mean and standard deviation of the frequency distribution are 262 and 12.9 for the two stage, 272 and 13 for the three stage and 272 and 13.18 for the four-stage model respectively. The coefficient of variance which indicates the extent of variability of the samples with respect to the mean of the population is less than 5%. Note that higher the coefficient of variation, greater the level of dispersion around the mean. As the dispersion rate is less than 5%, the mean value of the frequency distribution i.e., the mean throughput value from all the 500 instances is used to compute the VSS. Thus, the throughput presented in Table 4.1 represents the average throughput from the 500 instances.

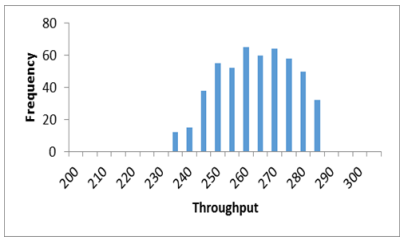
As shown in Table 4.1, there is a significant improvement in the VSS as the number of stages are increased from two to four. There is an 11.29% increase in the throughput when compared to the deterministic formulation. These results show that there are significant advantages in using the proposed multistage approach in comparison to the deterministic approach or the two-stage approach, which only improved throughput by 5.6%.

4.5.3 Results for Case Study II

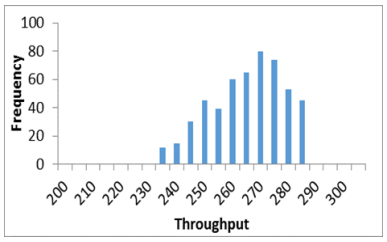
The authors of the study, [4] presented a fermentation process network where the operation takes place in three phases, a fermentation process followed by purification and packaging. The process network is shown in Figure 4.9. The first phase (fermentation) includes tasks T11, T12; the second phase (purification) includes tasks T21, T22, T23 whereas the third phase (packaging) includes tasks T31, T32, T33 and T34. The network provides the production recipe for the products P1 to P5. This case study was adapted to demonstrate the applicability of the proposed multistage approach, i.e., uncertainty was assumed in the

Figure 4.8: Frequency distribution of the mean throughput value from 500 instances for process network 1 and different number of stages

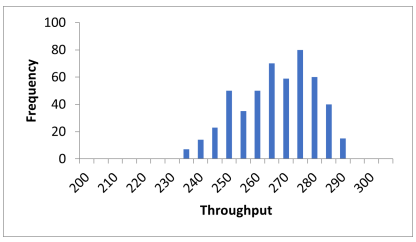
(a) Two stage



(b) Three stage



(c) Four stage

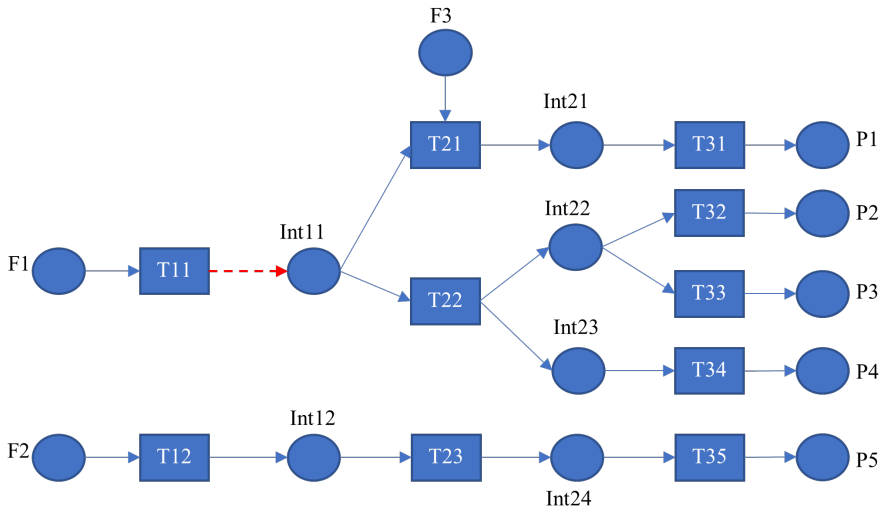


product yield of task T11 (highlighted in red in Figure 4.9). Five discrete possibilities were considered for the product yield (uncertainty parameter) - {0.6,0.7,0.8,0.9,0.1}. This implies that the maximum fraction of materials that can be lost due to incomplete processing in the imperfect task (task T11) is 0.4.

Those authors solved the case study for different instances considering varying processing times. The processing times considered in this study for the three phases of tasks shown in the Figure 4.9 are 2, 1 and 0.5hrs respectively for the 1st, 2nd and 3rd phases of tasks in the process network. A uniform discretization (UD) of time was used for this case study, with timestep ($\Delta(j)$) equal to 0.5hr. All the process parameters such as storage capacity, initial inventory, prices, and demands required to solve the model were adopted from [4]. The model was solved for a time horizon of 12hrs.

The time horizon was divided into multiple time periods and the problem was solved for two, three and four stages with 3, 9 and 27 nodes respectively in the first, second and third time periods. The computational time for this case study involving a single job and

Figure 4.9: Production process network 2; [4], where T11 is considered as the imperfect task (represented with a dashed red line)



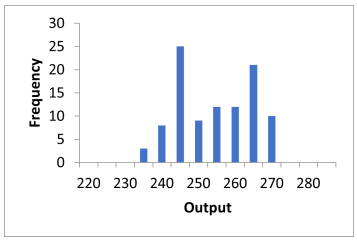
10 tasks, including 1 imperfect task ranged from 0.5- 3 seconds. To calculate the VSS, 100 sets of nodes were randomly chosen and for each instance, the multistage decisions were evaluated using 1,500 events. The frequency distribution of the mean throughput value obtained after the evaluation of the 100 chosen sets are presented in Figure 4.10. The mean and standard deviation of the distribution are 251.15 and 9.87 for the two stage, 260.23 and 11.22 for the three stage and 268.85 and 11.89 for the four-stage model, respectively. As the coefficient of variance less than 5% for each, the mean throughput value of the 100 instances was used to calculate the VSS and are presented in the Table 4.2. The results from this case study also depicts a significant increase in the VSS, which further increases up to 9.59% with increase in the number of stages from two to four.

4.5.4 Results from Large Scale Industrial Case study

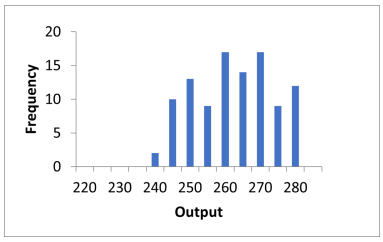
The performance of our node-based multistage framework was also tested using the actual industrial-scale study from the analytical services sector. The description of the industrial

Figure 4.10: Frequency distribution of the mean throughput value from 100 instances for process network 2 and different number of stages

(a) Two stage



(b) Three stage



(c) Four stage

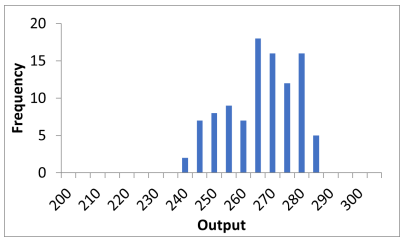


Table 4.2: VSS for the production process network 2

	Average Throughput	VSS(%)
Deterministic	243	
Two Stage	251.15	3.24
Three Stage	260.23	6.61
Four Stage	268.85	9.59

plant and specifications is provided in section 3.4.

In this study, as the processing times of the tasks involved ranges from a few minutes to several hours, a non-uniform discretization scheme (approach NUD60 in [10]) was used, where the maximum allowed time-step (the time elapsed between two consecutive timepoints, which is usually set equal to the completion time of the task) for any task is 60 time-units. This implies that if the task completion time is less than 60, time step is same as the completion time of that task and is equal to 60 otherwise. If there exist an imperfect task in the path of a job i, every time when an imperfect task finishes processing a batch of units, a fraction of those units is recycled back to a previous task in the path, depending on the realization of the uncertain parameter. Six possible discrete recycling rates are considered for an imperfect task, i.e., $\rho^n_{ik'k} \in \{0, 0.1, 0.2, 0.3, 0.4, 0.5\}$. The model has been solved for different instances. Cases of 5 and 10 jobs were considered, where each job considers 10-12 tasks in its path including multiple imperfect tasks. Each job has at least one imperfect task in its path. There are 24 and 39 unique tasks present in the

5 Job and 10 Job instances, respectively. With at least one imperfect task in every job, there is a total of 9 imperfect tasks (4 jobs with 2 imperfect task and one job with one imperfect task in its path) and 16 imperfect tasks (6 jobs with 2 imperfect tasks and 4 jobs with 1 imperfect task in its path) in the cases with 5 jobs and 10 jobs, respectively. For every job, the time of first uncertainty realization can be different depending upon the number of tasks preceding the imperfect task in the path of a job and the processing times of these tasks. The most frequent jobs at the facility were chosen for these instances, i.e., an instance of 5 jobs include the most frequent 5 jobs received at the facility. Some of the jobs consists of tasks with relatively high processing time, due to which it is likely that the uncertainty might not even realize if smaller scheduling horizons were considered. Therefore, to ensure that the model takes in to account the actual realizations and provide the necessary recourse decisions, a scheduling horizon of 24hrs was considered in this case study. Note that larger time horizons extensively increase the model size; however, as the proposed approach does not require auxiliary binary variables, our node-based framework is able to solve relatively larger (industrial) instances. In order to provide an insight to the size of the model we are dealing with, we note that the number of real variables and constraints in a two stage model for an instance of 10 jobs and a scheduling horizon of 24hrs is 117163 and 191425, respectively. Note that the size of the model increases with increase in number of stages.

The results for this case study are presented in Table 4.3. The throughput presented in the Table 4.3 is the average throughput of the 500 instances. Similar to the previous case studies, this problem was also solved up to four stages. It can be noted that, for this large-scale case study as well, there is a similar trend in terms of the VSS when the number of stages is increased from two to four. It can also be noted that as the number of jobs increases, the benefit in using the proposed multistage approach also increases. The computational time for solving the multistage model (two-stage to four-stage) for the first case of 5 jobs and 9 imperfect tasks ranged from 10-40 seconds, and for the second case of 10 jobs with 16 imperfect tasks, the computational time ranged from 50-160 seconds.

Table 4.3: VSS for the large-scale industrial case study

Instance	Average Throughput	VSS(%)
5Jobs24Hrs-Deterministic	985.2	
5Jobs24Hrs-Two Stage	1021.84	3.58
5Jobs24Hrs-Three Stage	1052.34	6.38
5Jobs24Hrs-Four Stage	1083.68	9.08
10Jobs24Hrs-Deterministic	1106.6	
10Jobs24Hrs-Two Stage	1154.77	4.17
10Jobs24Hrs-Three Stage	1196.91	7.54
10Jobs24Hrs-Four Stage	1231.72	10.15

4.5.5 Comparison Study – Proposed approach against binary variable approach

To better quantify the computational advantage of avoiding the use of binary variables, a comparson study has been conducted between the proposed approach and an approach that uses binary variables to enforce non-anticipativity. Note that these types of approaches that use binary variables have been used in several other works in the literature [16, 58, 19, 17]. The industrial case study from section 4.5.4 was chosen to perform the comparison study as it allows analyzing the model performances with respect to the variation in number of stages and also variation in number of jobs, in addition to being a large scale model. For convenience, in the following discussion, the model involving binary variables is referred to as binary variable approach. The key modifications required for non-anticipativity enforcement using the binary variable approach are included in Appendix B. The same set of instances of 10 jobs from Table 4.3 were solved using the node-based approach and the binary variable approach. In order to obtain a better analysis, the smallest instances with a single job were also solved using both the approaches. All the instances were solved to optimality using both the approaches. While both the approaches returned the same objective value for smaller single job instances, the objective value varied up to 6% for larger instances of 10 jobs. This can be attributed to the value of Big-M constant chosen in the constraints (B.2) in the binary variable approach which sets a lower bound for the

Table 4.4: Comparison results - Node-based approach Vs Binary variable approach

Instance	Computational Time(Sec)		% Reduction
	Binary Variable Approach	Node Based Approach	
1Job Two Stage	0.228	0.152	33.33%
1Job Three Stage	0.358	0.225	37.15%
1Job Four Stage	0.703	0.404	42.53%
1Job Five Stage	1.289	0.667	48.25%
10Jobs Two Stage	87.43	54.8	37.32%
10Jobs Three Stage	202.05	99.65	50.68%
10Jobs Four Stage	806.07	156.235	80.61%
10Jobs Five Stage	1752.16	255.02	85.44%

batch variables. A preliminary sensitivity analysis on the values of Big-M indicate that the change in M values does not cause significant changes on the computational costs. Since our key focus is to analyze the computational prospects in avoiding binary variables, a detailed study on choosing the Big-M value is beyond the scope of this work. The values used for M in the computational studies are presented in the Appendix B. The computational time required to solve the instances by both the approaches are provided in the Table 4.4. It can be noted that for the smallest instance of 1 job and three stages, the reduction in computational time while using the node-based approach is more than 33%. The computational gain in using the node-based approach increases with the number of stages and number of jobs. For the instance of 10 jobs and five stages, the reduction in computational time increases to more than 85%. Finally, note that the solution method for both approaches was just to solve the corresponding formulations using CPLEX, and more advanced solution methods have been studied in the literature that could be applied to speed up the solution times. However, it can also be noted that since both formulations were solved without the use of more advanced solution methods, the comparison that is made is still a fair one. Detailed discussions on the model performance and how it scales with increase in number of stages are provided in Appendix B.

4.6 Chapter Summary

In this chapter, a novel multistage stochastic programming approach was presented to model scheduling problems with type II endogenous uncertainty. The proposed model follows a node-based solution approach and allows the necessary flexibility to the model in capturing the time-dependent variability in the system behavior. The proposed multistage approach allows sequential realizations of the uncertain parameter throughout the time horizon and enforces implicit non-anticipativity without introducing auxiliary binary variables or explicit NACs. In addition, the current approach allows the possibility of multiple tasks of a job to be uncertain. In this study, the definition of multistage variables and the design of flow balance constraints are the key components in enforcing implicit non-anticipativity.

The proposed approach depicts significant benefits in terms of VSS. The model was validated using three different case studies including an actual large-scale industrial plant and two case studies adapted from the literature. Each case study exhibited significant increments in the VSS as the number of stages was increased. A comparison study was also conducted between the node-based approach and the conventional binary variable approach (from the literature). The results from the study shows up to 85% reduction in computational time while using the node-based approach in comparison to the binary variable approach.

Chapter 5

Planning and Scheduling of batch operations

As mentioned in chapter 1, the different aspects that can be incorporated with scheduling to enhance the efficiency of an industrial plant include uncertainty modelling and considering long term strategies. In chapters 3 and 4, the approaches to account for uncertainties in a scheduling model were discussed and two stage and multistage formulations for scheduling batch operations were presented. In this chapter, the focus is on the long term strategies that are adapted by the industries in order to plan ahead and achieve the long term objectives. Multiple studies in the literature developed such strategies for various multiproduct batch plants, where there are fixed products and recipes [98, 78, 70]. However, studies considering multijob batch plants such as industrial plants from ASI sector, where there are no fixed products/recipes and where the job specifications including recipe depends on the customer specifications are lacking from the literature.

The main objective of this study is to develop a long term planning model and an integration framework to solve the planning and scheduling models for a large scale multijob batch plant. An operational planning model that can consider the jobs arrived in the plant and provide the daily processing profile (which indicates how to carry out the daily plant operations in order to achieve the long term objective), could help enhance

the efficiency and economic prospects of the industrial plant. However, there are a few major challenges associated with this. First, when longer time horizons are considered, a planning model with detailed plant and job specifications could become intractable. In order to address this challenge, the current study considers various approximations while accounting for resource (machine) utilization and job sequence (recipe) effects. The second major challenge is that, as this study considers multijob batch plants where the job recipes are dependent on the customer specifications, it is challenging to ensure that the estimates or approximations used in the planning model are reasonable. To address this challenge, a calibration scheme is employed in the current study that helps to ensure that the approximations used in the planning model and the resulting planning decisions are reasonable. The third major challenge here is that when longer time horizons are considered for the multijob batch plants, the job arrival distributions for the future weeks/months are not known a priori and cannot be easily predicted. This challenge is addressed by using an iterative integration approach involving the rolling horizon method. Thus, the key parts of the proposed integration framework include: 1) the calibration scheme for ensuring that the estimates used in the planning model are reasonable, 2) iterative integration scheme involving the rolling horizon method for the integration of the planning and scheduling models. The proposed framework is validated using an actual industrial case study from the analytical services sector.

This chapter is structured as follows: section 5.1 presents the detailed problem statement and section 5.2 presents the model formulation for the planning and scheduling problems. Section 5.3 provides the integration framework for the effective interaction between the planning and scheduling models. Finally, section 5.4 provides the results and discussion from the computational experiments followed by the chapter summary in section 5.5.

5.1 Description of relevant problems

As discussed previously in section 2.3, while incorporating long term strategies for industrial operations, most often such problems are solved as an aggregated planning model and

a detailed scheduling model that are further integrated to obtain meaningful and feasible solutions. In order to provide better understanding of the problems under consideration, in this section, the overall problem is described first, followed by the detailed description on why and how the planning and scheduling problems are defined as two problems that are later integrated to obtain the final solutions.

Similar to the discussion in section 3.1, the multijob multitasking batch plant consists of a set of J tasks and receives a set of I jobs (customer orders). Each job $i \in I$ has a specific number of units (samples or materials), A_i that needs to be processed through a specific sequence of tasks, P^i (referred as *paths*) and each task $j \in J$ has multiple machines R_j with a capacity C_j to perform the task. The processing time and associated labour time for a task j is Φ_j and L_j respectively. In addition to the plant specifications in section 3.1, the economic aspects are also considered here. The jobs arrived generate a revenue of $\mathfrak{R}_i$ when their processing is completed within the due date, which is a week from the day of its arrival. All units (samples) from the jobs that do not complete processing within the due date are considered as backlogged samples. These backlogged samples generate a discounted revenue $\mathfrak{R}_j^{'}$ that are less than the one that would have received when completed within the due date. The utility cost associated per task $j \in J$ is U_j and LC represents the labour cost per hour.

The main goal is to obtain all the key decisions required to carryout the plant operations for a longer time horizon, such as the number of units (samples) to be processed from a job $i \in I$ in a task $j \in J$ at a time t; the number of machines (resources) of task $j \in J$ to be operated at a time t; and the number of workers required to operate these tasks, in order to maximize the long term profit.

However, there are a few challenges associated with the above goal. First, when a large multijob batch plant is considered for longer time horizons, an optimization model that consists of all the detailed plant and job information could become intractable. For instance, for the multijob industrial plant from the ASI sector (section 3.4), scheduling 300 jobs (where each job has a set of samples to be processed through a sequence of tasks) for an 8hr horizon results in a scheduling model (section 3.1.3) with 100,876 variables and 54,851 constraints. Accordingly, if we were to consider an optimization model spanning

over months, for example: 60 days, then it would consist of at least 6,052,560 variables and 3,291,060 constraints. Note that the model in section 3.1.3 does not account for the economic aspects and the labour information. With these additional details, the number of variables and constraints for a long term model would be even higher and thus results in an intractable model. Second, the future demands or the job arrival rates are unknown. The specifications (path and units) and the number of jobs that may arrive in the future weeks or months are not known *a priori* and cannot be easily predicted. Due to these challenges, solving a single optimization problem for obtaining the long term decisions for the batch plant becomes less viable. Therefore, in order to solve the problem under consideration, the problem is split into two: an operational planning problem and a detailed scheduling problem. By breaking the problem into two, in addition to addressing the model tractability issue, it will also allow flexibility in the model formulation allowing different assumptions and objectives for both models. Though being influenced by the planning problem, the scheduling problems can make decisions autonomously to optimize their model objectives, which can be different from the economic objective of the planning problem. As noted previously, it is difficult to predict the future demand or jobs arrivals. Hence, these features would also allow the scheduling model to accommodate the variations in job arrivals or rush order arrivals in the plant. These are all key components when it comes to a multijob batch plant where jobs (customer orders) and its specifications may vary with respect to the customer requirements. The operational planning problem and the scheduling problem are described next.

The operational planning problem overlooks certain details and consists of aggregated plant and job information in order to provide the key decisions involved in attaining the objective of maximizing the long term profit. In order to obtain these decisions, the planning problem considers the following. Consider a planning horizon, P, discretized into n_w weeks. Each week $w \in n_w$ is further discretized into d_w days as shown in the figure 5.1. n_d represents the total number of days, i.e., $n_d = \sum_w d_w$. The planning model considers a set of I_w jobs every week $w \in n_w$. It is assumed that all jobs I_w for a week $w \in n_w$ are available on the first day of the week. Each job $i \in I_w$ consists of A_i units that has to be processed sequentially through a set of of q_i tasks in its path P^i. In order to account

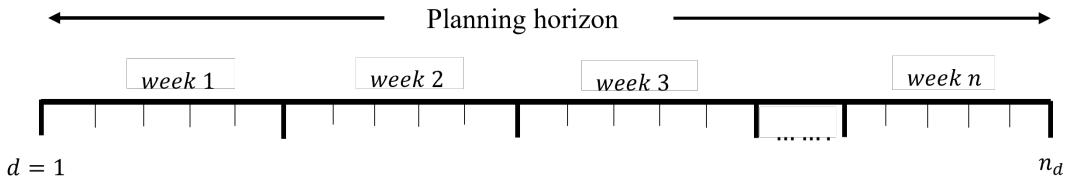


Figure 5.1: Discretization of planning horizon

for the plant capacity information, the maximum number of times a task can be operated in a day $d \in 1..n_d$ is estimated by considering the task processing time Φ_j and the daily horizon S, i.e., $N_j = (S/\Phi_j)$. Every machine (resource) for a task $j \in J$ can be operated at most N_j times per day (explained in detail in section 5.2.1). Unlike other multiproduct batch plants often considered in the planning and scheduling literature [68, 70, 22, 87], the multijob batch plants considered in this study do not consist of fixed products (or recipes) and the processing demand for any task depends on the jobs (customer orders) arrived. Hence, it is important to account for job sequence effects in the planning problem. As the finest discretization of the planning horizon constitutes the discretization into days, the detailed sequence effects involving the processing time of the tasks in the job path (P^i) cannot be accounted for in the planning problem. However, an approximation of the sequence effects can be considered in the problem. That is, the total number of units processed in a task k in the path of job i, i.e., $P^i_k \in P^i$ is at most the total number of units processed in the previous task in the path of job i, P^i_{k-1}.

When supplied with the job arrival information for the entire horizon, the process information (aggregated capacity information and the labour time information) and the economic information for regular samples/units (units that complete processing within the due date), backlogged samples/units, labour cost and utility cost, the operational planning model determines the key decisions such as the number of regular units processed in a task j on a day d of week w (B^w_{jd}); the number of backlogged units processed in a task j on a day d (BB^w_{jd}); the number of workers required to achieve these targets (NW), where each worker's shift length is SL hrs per day; and the number of times a task j can be operated on a day d of week w (Z^w_{jd}). The total number of regular units and backlogged units processed in a task j on a day d constitutes the processing target for task j for a

day d, and is referred as the daily processing target for a task j, $\mathbf{B_{jd}}$. These decisions, $\mathbf{B_{jd}}$ and NW obtained from the planning model indicate what the daily processing rate (processing target) of a task should be and what are the number of required workers in the batch plant, in order to maximize the long term profit. These decisions could act as a basis for scheduling daily plant operations and can be supplied to the scheduling model to guide the scheduling decisions.

The scheduling model for the multijob batch plant is similar to the model discussed in section 3.1.3 with an objective of maximizing the throughput. It accounts for the detailed plant specifications (R_j, C_j, Φ_j), job arrivals (I, A_i), complete job sequences (P^i) and follows a finer time discretization for the daily scheduling horizon S (refer to section 3.1.2 for time discretization details). In addition, the detailed scheduling model accounts for the decisions supplied by the planning model (processing targets $\mathbf{B_{jd}}$ for each task $j \in J$ on a day $d \in 1..n_d$ and the number of workers (NW)). While accounting for the detailed job and plant specifications and the decisions from the planning model, the scheduling model attempts to achieve the processing targets and provide daily schedules for the batch plant. The modifications required for the scheduling model in order to account for the planning decisions are discussed in detail in section 5.2.2.

Even though there are benefits in breaking the problem into two and supplying decisions from the long term planning model to the detailed scheduling model, it may not be most effective unless the planning decisions are accurate. Due to the approximations considered in the planning model for the capacity utilization and the job sequences, the planning decisions might not be accurate and could even become infeasible. In addition, the planning model determines the decisions for a given job distribution for the entire horizon and does not account for the variations in the job arrival information. This leads to a third challenge in addition to the two challenges discussed in the beginning of this section. In order to address this challenge, the current study considers the following: 1) a calibration scheme that aims at modifying the task capacity bounds in the planning model to reflect more accurately the actual plant capacity, 2) an iterative integration scheme involving the rolling horizon approach that allows accounting for the variations in the job arrivals in the planning model.

In order to ensure proper understanding, the planning and scheduling models are presented first in the next section 5.2, followed by the detailed discussion of the calibration scheme and the iterative integration method in section 5.3.

5.2 Model Formulation

5.2.1 Operational Planning problem

Before presenting the detailed mathematical model, the decision variables in the planning model are highlighted below.
B^w_{jd} - the batch of units processed in task j on day d of week w
S^w_{jd} - the amount of units waiting to be processed in task j on day d of week w
BC^w_{ikd} - Number of units processed in task k of job i on day d of week w
BB^w_{jd} - Number of backlog units processed in a task j on day d of week w
BL^w_{jd} - Number of backlog units waiting to be processed in task j on day d of week w
Z^w_{jd} - Number of times a task j is operated in day d of week w
LT^w_{jd} - Labour time required for task j in day d of week w
NW - Required number of workers

The detailed planning model is presented next.

Capacity Constraints

The planning model uses an aggregated capacity information for determining the planning decisions. Constraints (5.1) ensure that the number of times a task j can be operated in day d should always be less than the maximum number of times that task can be operated in that period. N_j provides the maximum number of times a resource of task j can be operated in any day. Constraints (5.2) ensure that the total number of units that can be processed in a day d (including both the new samples B^w_{jd} and the backlogged samples BB^w_{jd}) does not exceed the available capacity for the task.

$$Z^w_{jd} \leq R_j N_j \quad \forall j \in J, \forall w \in 1..n_w, \forall d \in 1..d_w \tag{5.1}$$

$$B^w_{jd} + BB^w_{jd} \leq C_j Z^w_{jd} \quad \forall j \in J, \forall w \in 1..n_w, \forall d \in 1..d_w \tag{5.2}$$

Job Path Approximation Constraints

In order to provide the model with an aggregated information on the sequence of tasks through which the units from a job should be processed, the job path approximation constraints are defined. Constraints (5.3) specify that the number of units to be processed in the first task in a path of job i (BC^w_{i1d}) should not exceed the total units in that job (A_i). Constraints (5.4) specify that the number of units processed in a task k of job i should not exceed the number of units processed in the preceding task of the job. Constraints (5.5) define that the number of units processed in a task j in day d is equal to the total number of units from all job i with task j in its path that were processed in day d.

$$\sum_{d \in 1..d_w} BC^w_{i1d} \leq A_i \quad \forall i \in I_w, \forall w \in 1..n_w \tag{5.3}$$

$$\sum_{d \in 1..d_w} BC^w_{ikd} \leq \sum_{d \in 1..d_w} BC^w_{ik-1d} \quad \forall i \in I_w, \forall k \in 2..q_i, \forall w \in 1..n_w \tag{5.4}$$

$$B^w_{jd} = \sum_{i \in I_w} \sum_{k=1..q_i : P^i_k = j} BC^w_{ikd} \quad \forall j \in J, \forall w \in 1..n_w, \forall d \in 1..d_w \tag{5.5}$$

Batch Constraints

Constraints (5.6) - (5.10) refer to the general flow balance across the planning horizon. Constraints (5.6) distribute the total number of units available to be processed in a task j in week w as the number of units processed in task j in day 1 of week w (B^w_{j1}) and the sum of units waiting to be processed in task j in day 1 of week w (S^w_{j1}). Constraints (5.7) represent the flow balance constraint for all days of the week except the first. Constraints (5.8) define the backlog units (BL^1_{jd}) and the number of backlog units processed (BB^1_{jd}) in the first week. As there are no backlogs in the first week, these variables are equated to zero in the constraints (5.8). Constraints (5.9) provide the flow balance for the backlog units for the first day of a week w. They define the total backlogged units for a task j in

first day of a week w (BL^w_{j1}) as a combination of the backlogged units from the previous day, i.e. the last day d_w of the previous week $w-1$ ($BL^{w-1}_{jd_{w-1}}$) and the unprocessed units among the sample arrivals of the previous day ($S^{w-1}_{jd_{w-1}}$) and the number of backlogged units processed in the day d (BB^w_{jd}). Constraints (5.10) provide the flow balance for backlogged units across the days.

$$B^w_{j1} + S^w_{j1} = \sum_{i \in I_w} \sum_{k=1..q_i : P^i_k = j} A_i \quad \forall j \in J, \forall w \in 1..n_w \tag{5.6}$$

$$S^w_{jd} = S^w_{jd-1} - B^w_{jd} \quad \forall j \in J, \forall w \in 1..n_w, \forall d \in 2..d_w \tag{5.7}$$

$$BL^1_{jd} = BB^1_{jd} = 0 \quad \forall j \in J, \forall d \in 1..d_1 \tag{5.8}$$

$$BL^w_{j1} = BL^{w-1}_{jd_{w-1}} + S^{w-1}_{jd_{w-1}} - BB^w_{j1} \quad \forall j \in J, \forall w \in 2..n_w \tag{5.9}$$

$$BL^w_{jd} = BL^w_{jd-1} - BB^w_{jd} \quad \forall j \in J, , \forall w \in 2..n_w, \forall d \in 2..d_w \tag{5.10}$$

Labor Time Constraint

Constraints (5.11) provide the total labour time required for a task j in day d (LT^w_{jd}). Constraints (5.12) provide the number of workers required throughout the planning horizon based on the processing requirements.

$$LT^w_{jd} = Z^w_{jd} L_j \quad \forall j \in J, \forall w \in 1..n_w, \forall d \in 1..d_w \tag{5.11}$$

$$NW \geq \frac{\sum_{j \in J} LT^w_{jd}}{60 * SL} \quad \forall w \in 1..n_w, \forall d \in 1..d_w \tag{5.12}$$

Objective Function

The objective function of the planning model aims at maximizing the long term profit. The timely completion of jobs are the key to generating the maximum revenue. Hence the long term objective of the planning problem is defined as follows:

$$Max \sum_{w\in 1..n_w} \sum_{d\in 1..d_w} \sum_{i\in I_w} BC^w_{iq_id}\mathfrak{R}_i + \sum_{w\in 1..n_w} \sum_{d\in 1..d_w} \sum_{j\in J} BB^w_{jd}\mathfrak{R}'_j - \sum_{w\in 1..n_w} \sum_{d\in 1..d_w} \sum_{j\in J} Z^w_{jd}U_j - LC(NW * SL * n_w * d_w)$$

The units that have completed processing the final task (q_i) in the path of a job i within a week of its arrival generate the maximum revenue ($BC^w_{iq_id}\mathfrak{R}_i$) while the units that are processed after the weekly due date generates a discounted revenue ($BB^w_{jd}\mathfrak{R}'_j$). The objective function also takes into account the utility cost ($Z^w_{jd}U_j$) and the labor cost ($LC(NW * SL * n_w * d_w)$).

The proposed operational planning model provides the key decisions that can be used to guide the scheduling model decisions. These decisions include:
• The processing targets for each task $j \in J$ - $\mathbf{B_{jd}}$ (this includes the processing of regular samples and backlogged samples, i.e., $\mathbf{B_{jd}} = B^w_{jd} + BB^w_{jd}$)
• The number of workers available to perform the plant operations - NW

5.2.2 Modified Scheduling Model

As mentioned above the scheduling model is similar to the one presented in section 3.1.3. In this chapter, the presented model in section 3.1.3 is modified to account for the planning decisions so that scheduling model can provide schedules for daily plant operations while attempting to follow the plan and achieve the planning targets. In this section, the focus is on those modifications followed by the complete scheduling model that accounts for the planning decisions.

Recall that scheduling model accounts for the detailed process information and sequence of tasks in the path of a job i via specific resource constraints (3.1), capacity constraints (3.2) and flow balance constraints (3.3),(3.4) to provide decisions including the task and

batch allocation at any time over the scheduling horizon. The key scheduling model decisions include the amount of units processed in task k of job i at time t (y_{ikt}), the amount of units waiting to be processed in task k of job i at time t (x_{ikt}) and the number of machines to be operated for task j at time t (z_{jt}).

The additional constraints included in the modified scheduling model are:

Labour Time Constraint

Constraint (5.13) specifies that the total labour time cannot exceed the available labour hours.

$$\sum_{j \in J} \sum_{t \in 1..|\varepsilon(j)|} z_{jt} L_j \leq NW * SL * 60 \tag{5.13}$$

Processing Target Constraint

Constraint (5.14) accounts for the processing target supplied by the operational planning model. UO_{jd} represents the difference between the processing target ($\mathbf{B_{jd}}$) obtained from the planning model and the processing rate achieved by the scheduling model ($\sum_{i \in I} \sum_{k:P_k^i=j} \sum_{t \in 1..|\varepsilon(j)|} y_{ijt}$) for a task $j \in J$ on a $d \in 1..n_d$.

$$\mathbf{B_{jd}} - \sum_{i \in I} \sum_{k:P_k^i=j} \sum_{t \in 1..|\varepsilon(j)|} y_{ikt} = UO_{jd} \tag{5.14}$$

Objective Function

The objective function of the scheduling model is modified as follows.

$$Max \sum_{i \in I} \sum_{k \in 1..q_i} \sum_{t \in 1..|\varepsilon(P_k^i)|} \frac{k}{q_i} y_{ikt} - \omega \sum_{j \in J} UO_{jd}$$

With the above modification, any unsatisfied processing targets (UO_{jd}) are penalized in the objective function with a penalty factor $\omega > 0$. Note that the modified objective function incentivizes any surplus processing rate. In this way, the scheduling model would possess more flexibility in accounting for the actual processing capabilities of the plant. If the processing target obtained from the planning model is above the plant capacity, the

scheduling model would still aim to achieve the target and any difference will be penalized whereas if the planning model provides a lower target while the plant has enough capacity to process more available units, then the scheduling model can account for the additional processing as the difference will be incentivized by the objective function. Note that possibility of planning model providing a lower target is a feature particular for multijob plants. As there are no fixed products and since the job paths (recipes) vary depending upon the customer specifications, it is possible that some tasks may receive more units to process in actual than what was anticipated or predicted while solving the planning model. Therefore, providing flexibility for the scheduling model to account for the actual processing capabilities of the plant becomes relevant when accounting for a multijob batch plant.

The final scheduling model considered in this study is as follows:

$$\begin{aligned} Max \quad & \sum_{i\in I}\sum_{k\in 1..q_i}\sum_{t\in 1..|\varepsilon(P_k^i)|}\frac{k}{q_i}y_{ikt} - \omega\sum_{j\in J}UO_{jd} \\ \text{s.t.} \quad & (3.1)-(3.4),(5.13)-(5.14) \\ & x, y \geq 0 \\ & z \geq 0 \text{ and integer} \end{aligned} \tag{S1}$$

5.3 Integration of planning and scheduling models

As mentioned in section 5.1, one key challenge in breaking the problem into two and supplying decisions from the long term planning model as inputs to the detailed scheduling model is that it may not be most effective unless the planning decisions were accurate. The approximations considered in the planning model often results in inaccurate capacity estimations and the resulting planning decisions could be unreasonable or even infeasible. In order to address this challenge and ensure that the planning decisions are reasonable and achievable by the scheduling model, a calibration scheme is considered in this study.

Once the planning model is calibrated and consists of estimates that more accurately reflect the actual plant capacity, the calibrated planning model and the scheduling models

are integrated using an iterative framework. An iterative integration framework is an approach that allows two-way interaction between the planning and scheduling models and also allow different objectives for the two models. Moreover, as discussed previously, one of the main challenges of this multijob batch plant problem is that the future job demand and specifications are unknown and difficult to predict. An iterative integration scheme that allows solving planning and scheduling models separately also provides better prospects in accounting for the variations in the job arrivals. Hence, the current study considers a rolling horizon approach [85] for solving the integrated planning and scheduling models.

5.3.1 Calibration scheme

In order to obtain reasonable planning decisions, it is key to ensure that the approximations considered in the planing model are reasonable. As this study considers a multijob batch plants with no fixed products/recipes, the processing demand of each task depends upon the job arrivals and its specifications including the associated samples and job sequences. This makes it quite challenging to account for the actual sequence effects in the planning model. Even though approximated sequence effects are considered in the planning problem, due to the nature of the multijob problem, further measures have to be incorporated to ensure that the planning decisions are reasonable and achievable. The approximations considered in the planning model could affect the estimation of the capacity bounds of the tasks and result in processing targets that are not achievable by the scheduling model. There are studies available in the literature that consider different approaches to derive more accurate capacity constraints for the planning model that closely reflect the actual plant capacity [69, 87]. However, as the current study considers a multijob batch plant where there are no fixed products or recipes, these approaches would not be applicable in this case. Hence, in this study, a calibration scheme is applied to update the capacity bounds of the tasks for a given job distribution. In order to update the bounds in the planning model with respect to the actual capacity from the scheduling model, the calibration scheme allows the two models to interact with each other.

The detailed steps involved in the calibration scheme are presented next.

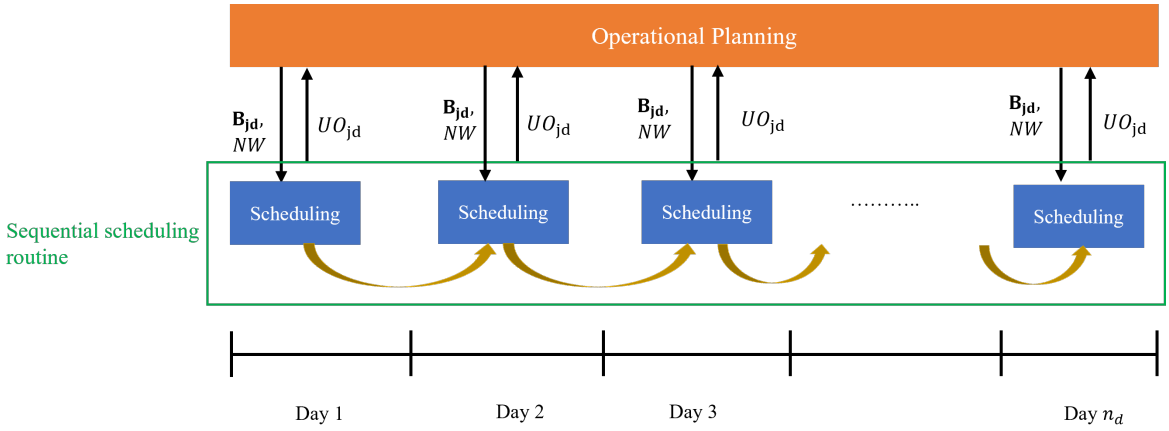


Figure 5.2: Illustration of the Iterative Calibration Scheme

For a given job distribution for the planning horizon, the following steps are performed for calibrating the planning model:

(C1) Solve the planning model presented in section 5.2.1 for the entire planning horizon and obtain the key decisions,i.e., daily processing target ($\mathbf{B_{jd}}$) and number of required workers (NW).

(C2) Solve the scheduling model (S1) for entire planning horizon using the *sequential scheduling routine*. This involves solving each scheduling sub-horizon (day) and carrying over any incomplete jobs to the following scheduling horizon.

The *sequential scheduling routine* can be considered as partitioning the entire planning horizon into n_d smaller sub-horizons and then solving each of these sub-horizons sequentially. Let α_d denotes the d^{th} sub-horizon that needs to be scheduled, where $d \in 1...n_d$. The sequential procedure begins by solving the first sub-horizon ($d = 1$) assuming the state of the plant as having initial set of jobs, denoted by JB_d, available at the beginning of the sub-horizon with all the resources available and empty. Then, solve the model (S1) over the sub-horizon α_d to generate a schedule $Schedule_d$ and obtain the corresponding unsatisfied processing targets per task for the sub-horizon d, i.e., UO_{jd} (see equation (5.14)). The state of the plant is updated

using $Schedule_d$ and JB_d to reflect the current jobs and current machine usage at the beginning of sub-horizon α_{d+1}. Any incomplete jobs from sub-horizon α_d are transferred to the sub-horizon α_{d+1} and accounted in the current jobs for α_{d+1}. If there are further sub-horizons to schedule, increment d and repeat the process. When $d = n_d$, stop and the concatenating $(Schedule_1, Schedule_2, \ldots, Schedule_{n_d})$ gives a feasible schedule for the entire horizon P. Note that each schedule, $Schedule_d$ maybe optimal with respect to the corresponding sub-horizon, but the concatenated schedule may not necessarily be optimal with respect to P as each sub-horizons are scheduled individually.

(C3) The unsatisfied processing targets from each sub-horizon d are used to update the upper bounds (UB_{jd}) of processing targets in the planning model using constraint 5.15:

$$UB_{jd} \leq \mathbf{B_{jd}} - UO_{jd} \quad \forall j \in J, d \in 1..n_d \tag{5.15}$$

Solve the updated planning model for the entire horizon and obtain the modified planning decisions including the processing targets and the number of employees.

(C4) Continue steps (C2) - (C3) until $UO_{jd} \leq tol_j$ or $iter \leq ITER_{NUM}$.

The termination criteria considered here include the value of unsatisfied processing target for the tasks satisfying a given tolerance limit (tol_j) or a limit on the number of iterations ($ITER_{NUM}$). Two termination criteria are used due to the possibility that in some cases a large number of iterations may be required to meet the tolerance limit. As this study considers a multijob plant and that the processing demands for each task depend on the specifications of the jobs arrived, there may be tasks that have to process thousands of units and there may be tasks that have to process only tens of units. Then, the tolerance limit for these tasks have to be set differently and it may result in large number of iterations. Hence, a second termination criteria involving the iteration number is also employed in the calibration scheme. The discussion on how these parameters are set for the current studies are presented in section 5.4.

The illustrative representation of the calibration scheme is shown in figure 5.2.

With the above calibration procedure, a planning model that consists of modified processing bounds is obtained based on the processing rates of the scheduling model, thereby more accurately reflecting the actual processing capabilities of the plant.

Note that the calibration procedure is performed for a given job distribution. However, in reality, when planning horizons over months are considered, it is difficult to forecast the future job arrivals. Particularly, when a multijob batch plant is considered where the processing demands for each task depends on the job specifications, it is important to be able to account for the new job arrivals. This can be taken into account using a rolling horizon approach.

5.3.2 Rolling Horizon Approach

Rolling horizon (RH) methods solve the integrated planning and scheduling models in a sequence of iterations, each of which models only part of the time horizon in detail, while the rest of the horizon is represented in an aggregate manner. As shown in figure 5.3, in the first iteration of the rolling horizon approach, a part of the time horizon is solved using the detailed formulation (scheduling model) while the rest is represented in an aggregate manner (planning model) and in the second iteration, the second part of the horizon is solved while decisions for the already solved first part is fixed and the later parts are represented in an aggregate manner. These iterations continue until the entire time horizon is scheduled using the detailed formulation. In principle, this approach may produce close to optimal solutions with a significant reduction of the computational requirements.

When applying the rolling horizon approach for the integration purposes, the overall time horizon is discretized into periods having endpoints which represent the moments in time when the planning and scheduling layers directly interact with one another. Note that the discretization of periods should be in line with the discretization employed in the planning model. Since the proposed operational planning problem considers the discretization of the planning horizon into weeks and days, there is flexibility in the choice of when

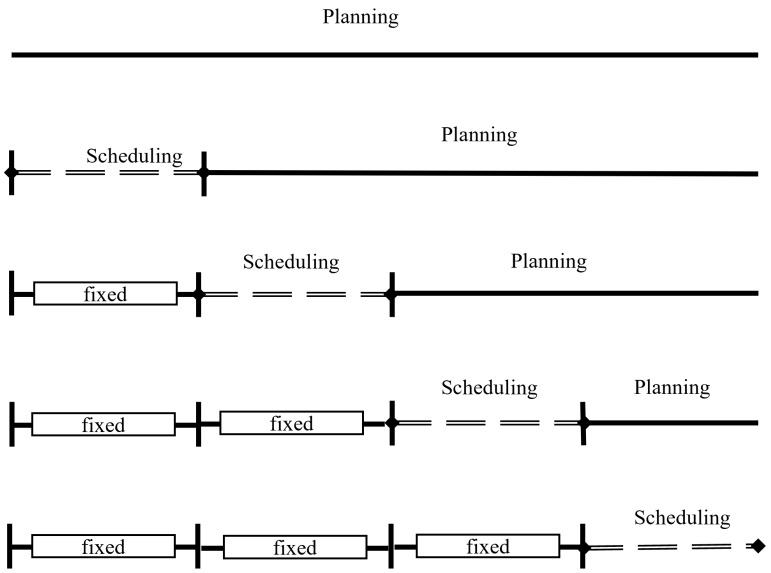


Figure 5.3: Schematic Representation of Rolling Horizon Framework

to allow the planning and scheduling models to interact via rolling horizon approach. In this study, the end of the week is considered to be the moments in time when the planning and scheduling layers directly interact with one another, i.e, for rolling horizon purposes, the planning and scheduling layers are allowed to interact at the end of every week. Therefore, the number of periods is the same as the number of weeks considered in the planning horizon (n_w).

The detailed steps in the rolling horizon approach are as follows:

Step 1 Set the first week as the current period, l and solve the calibrated planning problem with the predicted job distribution. Let the set of jobs based on the predicted distribution be represented as $I_w^1 \ \forall w \in n_w$. Obtain the processing targets ($\mathbf{B_{jd}}$) and number of required workers (NW).

Step 2 Solve the scheduling model for the current period, l using the sequential scheduling routine explained in **(C2)** of the calibration scheme. Note that unlike **(C2)**, here, instead of n_d sub-horizons, the scheduling model is solved only for the sub-horizons

in the current period, l.

As described in **(C2)**, by using the sequential scheduling routine, the scheduling model can accommodate for new set of jobs JB_d for every scheduling sub-horizon. This allows the scheduling model to accommodate any variations to the predicted job distribution used in the planning model.

If the current period l is the last week of the planning horizon, stop. Otherwise, go to Step 3.

Step 3 Now that the current period l is scheduled, update the state of the plant using the latest information. That is, based on the scheduling results from the current period l, fix the processing rates (B^w_{jd} and BB^w_{jd}) and update the bounds for the current period, l using the following:

$$\begin{aligned} B^w_{jd} &= \sum_{i \in I_w} \sum_{k:P^i_k=j} \sum_{t \in 1..|\varepsilon(j)|} y_{ikt} \\ BB^w_{jd} &= \sum_{i \notin I_w} \sum_{k:P^i_k=j} \sum_{t \in 1..|\varepsilon(j)|} y_{ikt} \\ UB_{jd} &\leq \mathbf{B_{jd}} - UO_{jd} \end{aligned} \tag{5.16}$$

The job distribution is also updated using the latest information available at that point and also considering the information from the last scheduling sub-horizon of the current period, l. That is, update the set of jobs to I^{l+1}_w to reflect any known variations in the predicted job distribution.

• For all $w \in n_w : w \leq l, \quad I^{l+1}_w = I^l_w$; That is, for all the weeks until the period l, the distribution considers the jobs that have already been scheduled.

• For all $w \in n_w : w > l+1, \quad I^{l+1}_w = I^1_w$; That is, for all weeks later than period $l+1$, the distribution remains the same as that of the initial predicted distribution.

• For $w \in n_w : w = l+1, \quad I^{l+1}_w$ includes the new jobs arrived for period $l+1$ along with any incomplete jobs from sub-horizon α_d, where d refers to the last sub-horizon

of the current period l. Fix the required number of workers (NW) to be the same as that of obtained in Step 1.

With these updated information solve the planning model again; update the current period index, l; go to Step 2.

Note that in the above algorithm, the required number of workers (NW) remains a fixed value as obtained from the first iteration of the planning model and are not modified in the later iterations because changing the number of required workers frequently would not be an ideal scenario for the industry.

5.4 Computational Study

The integration framework proposed in section 5.3.2 is used to solve the actual industrial case study from the ASI sector. The description of the industrial plant and specifications is provided in section 3.4. As mentioned in the section 3.4, the plant consists of over 180 tasks and multiple identical resources to perform each task. During a typical one month timespan, the plant receives jobs comprising of over 200 unique paths, with over 100 unique tasks. Over this timespan, they receive several hundred jobs comprising of more than 20,000 samples. As mentioned previously, the capacities and processing times of the individual tasks vary greatly. The largest capacity among all tasks 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. These features differentiate the plant and results in a large problem to tackle. The economic data included in the computational studies - the revenue from jobs, utility cost and the labour cost were provided by the industrial plant. Due to a non-disclosure agreement, the actual data cannot be presented here. Hence, the normalized process data is presented in the Appendix. The goal is to maximize the profit by completing the sample analysis within the due date. The due date is one week from the day the samples were received.

In the next sections, results from multiple computational experiments using this industrial case study are presented. In order to conduct the experiments, the historical data

from the plant was used. The plant data was collected for months and the paths of the arrived jobs and the samples associated with the jobs were recorded. The detailed discussion on the experiments and data used are given in the later sections. All the computational experiments were implemented using Julia 1.0.5 [99] and solved using CPLEX solver in a i7-3.40GHz Windows machine with 16GB RAM.

5.4.1 Performance Metrics

Before discussing the various experiments and results,the performance metrics used to evaluate the results are presented here. Throughout this section, to analyse the performance of the plant and to evaluate the benefits in using the proposed framework, two criteria are considered: percentage completion analysis and profit. Percentage completion analysis determines the amount of samples that completed analysis within one week of arrival (d_w represents the days in a week). $SamplesArrived_{id}$ denotes the samples in job i arrived on a day d and $SamplesCompletedAnalysis_{id}$ denote the samples in job i that completed processing all the tasks in the path of job i on day d. Then, the percentage completion analysis is defined as follows:

$$\text{Percentage completion Analysis} = \frac{\sum_{i \in I_d} \sum_{h=d}^{d+d_w} \text{Samples Completed Analysis}_{ih}}{\sum_{i \in I_d} \text{Samples Arrived}_{id}} \tag{5.17}$$

Total profit is calculated by combining the daily profits (Equation (5.13)).

For detailed analysis of the benefits in using the proposed planning model and the integration framework, the results obtained from the computational experiments are compared against the direct scheduling approach. The direct scheduling approach corresponds to solving the scheduling model presented in section 3.1.3 with an additional constraint to account for the available workers (constraint (5.13)) using a *sequential scheduling routine* (section 5.3.1). To make a fair comparison, the number of available workers in constraint (5.13) for the direct scheduling approach is fixed to be the same as the value used in the cor-

responding instances of experiments performed using the proposed framework. Throughout this section, the results obtained from the instances of the direct scheduling approach are presented under the column 'Scheduling Direct'.

5.4.2 Iterative Integration Framework - Analysis

The first part of the integration framework constitutes the calibration of the planning model. For a planning horizon of c_m months (for this case study, the planning horizon is considered as 2 months), the historical industrial plant data was observed and the jobs arrived and its specifications were recorded. Using the recorded data, the job distribution for the planning horizon is obtained and the calibration of the planning horizon is performed as per section 5.3.1. In order to obtain the job distribution for the 2 months, job paths were sampled based on the information recorded from the historical data that included the job paths and their observed frequencies. Sampling the job paths based on their observed frequencies helps ensure that the considered instance resembles an actual scenario in the batch plant by accounting for higher number of high frequent job paths compared to the low frequent paths. The number of samples in each job was selected uniformly at random between 10 and 50, which was also determined based on the observations from the historical data. The scheduling horizon considered in the experiments is 8hrs, i.e., in step **(C1)** of the calibration procedure, $N_j \ \forall j \in J$ in the planning problem is calculated using $S = 8$hrs and the processing time of tasks Φ_j. Similarly, the length of scheduling sub-horizons, α_d in the step **(C2)** of the calibration procedure were set equal to 8hrs.

As discussed previously in section 5.3.1, the goal of calibration procedure is to ensure that the planning model consists of processing estimates that more accurately represent the actual plant specifications. Note that the calibration is performed using the job distribution obtained from the historical data. Referring to the discussion in section 5.1, the task demands could vary from the historical projection with respect to the specifications of the jobs received. Hence, while performing the calibration procedure, rather than focusing on modifying the processing bounds in the planning problem until the planning processing targets ($\mathbf{B_{jd}}$) are perfectly satisfied by the scheduling model, a certain level of tolerance is

allowed for the unsatisfied processing demands, UO_{jd} (the difference between the processing target for a task supplied by the planning model and the rate achieved by the scheduling model) and are defined as tolerance limit, tol_j for each task.

In order to set the tolerance limit, tol_j and the limit on the iteration number ($ITER_{NUM}$) discussed in section 5.3.1, a set of preliminary experiments were carried out. Based on the results of the preliminary experiments and the historical data, the bottleneck tasks, the tasks with higher processing demands and the tasks with lower processing demands were identified. Bottleneck tasks are those that are processing at a full capacity through out the scheduling sub-horizon. Tasks with high processing demands are considered those that process more than 40% of the weekly demands in the plant; for instance, if the plant receives 5000 samples in a week, a task that has to process more than 2000 samples are considered as high processing demand task. Similarly, low processing demand tasks are those that process less than 10% of the weekly sample arrivals in the plant. The tolerance limit (tol_j) set for these tasks in the current study are presented in Table 5.1. The values of tol_j for these tasks were chosen considering the following aspects. Even though the task demands could vary from the historical projection with respect to the specifications of the jobs received, the tasks with higher processing demands (as per the historical data) represents those tasks that are commonly present in the job paths and are more likely to follow a similar trend in terms of processing demands. For such tasks, a lower tolerance limit of 4% was set. That is, for a task that has to process 5000 samples in a week, the unsatisfied demands could be at most 200 samples. For any week, it is highly likely that there is a variation of 200 samples between the historical projection and the actual samples based on jobs received. Hence, further reducing the tolerance limit to a lower value may not be significant. Similarly, considering the observable variations in samples with respect to the historical projections and actual job specifications, a tolerance limit of 10% and 20% were defined for the medium processing demand tasks and the low processing demand tasks, respectively. In addition, for the bottleneck tasks, a finer tolerance limit of 2% was defined as they could have higher impact on the profit function. Based on the preliminary experiments, it was observed that at least 75% tasks, including the high processing tasks and the bottleneck tasks satisfied the tolerance limit within 5 iterations. Recall that the

actual job distributions could vary from the historical distribution and these variations could be predominant for the low processing demand tasks. Hence, increasing the number of iterations for achieving the tolerance limit for the low processing demand tasks may not ensure increased benefits. Therefore, the iteration number limit ($ITER_{NUM}$) was set at 5 for the calibration procedure. Another parameter that was defined based on the preliminary experiments is the penalty factor ω defined in the scheduling model S1 in section 5.2.2. The value of ω was varied from 0.001 to 1 and the sensitivity of the results to the variations in the value was analysed. Based on the observations, the value was chosen as 0.1.

Table 5.1: Computational parameters - tolerance limit

Specification of tasks	Processing Demand	Tolerance (tol_j)
Low processing demand	$< 10\%$ of the weekly samples arrived	20%
Medium processing demand	$\geq 10\% \leq 40\%$ of weekly samples arrived	10%
High processing demand	$> 40\%$ of weekly samples arrived	4%
Bottleneck Tasks		2%

Using the above information and the task specifications (the normalized data for the task specifications are presented in the Appendix), the planning model was calibrated. The calibrated planning model thus obtained was used in the second part of the integration framework, the rolling horizon method.

The calibrated planning model and scheduling model were solved as per the steps included in section 5.3.2 for a 2 month horizon. The time horizon was decomposed into periods, with each period being a week. Note that, in the rolling horizon approach, as the planning model is solved again after scheduling each period, the planning model can accommodate for any new job arrivals. That is, in the first iteration, the planning problem is solved with a predicted job distribution (I_w^1) for the entire planning horizon. Once the first week ($l = 1$) is scheduled, the predicted job distribution (I_w^l) is updated to account for the latest available information. Thus the second job distribution (I_w^2) for the planning problem considers the jobs that has already been scheduled in the first week (I_1^1), and the incomplete jobs from the first week along with the newly arrived jobs that are

available to process in the beginning of the second week, and the predicted jobs for the weeks later than second week ($I^1_{w:w>2}$). In the second iteration of the RH method, the second week ($l = 2$) is scheduled and this is followed by updating the job distribution for the planning horizon a third time. In this third job distribution (I^3_w) for the planning problem, for the first two weeks, it considers the scheduled jobs for the respective weeks (I^1_1, I^2_2), and the incomplete jobs from the second week along with the newly arrived jobs are considered for the third week, and the predicted jobs are considered for the later weeks ($I^1_{w:w>3}$). This process continues until all weeks are scheduled. Due to the updating of the job distribution after scheduling every week, the planning problem can take into account the variations in job arrivals as opposed to the predicted job distribution and therefore provide more effective planning decisions. Note that the above integration procedure was carried out for a 2 month horizon. The job data may vary when the data for a different 2 month horizon is considered. For instance, the observed job distribution for the months of June and July could be different from the observed job distribution for March and April. Hence, to obtain a detailed analysis, the integration procedure (including the calibration and the rolling horizon method) was repeated for five different 2 month planning horizons. The evaluating criteria, percentage completion analysis and profit were calculated and the obtained results are presented in table 5.2 under the column 'Integrated Framework', where CP1, CP2, ... ,CP5 represents a different 2 month planning horizon. The results obtained from the corresponding instance using the direct scheduling approach are also presented in the table 5.2 under the column 'Scheduling Direct'.

Figure 5.4 plots the percentage completion rates for the 5 instances presented in the table 5.2. For the direct scheduling approach, where there are no processing targets supplied from the planning model, the average rate of samples that completed the analysis within a week of its arrival in the plant is 0.794. For the integrated approach, that accounts for the planning decisions in the scheduling model and allows two-way interaction between the models, the average rate of samples that completed the analysis within a week of its arrival in the plant is 0.893. The results show an average increase of 9.8% in the analysis completion rates when the planning and scheduling models are integrated. Figure 5.5 plots the profits for the 5 instances presented in the table 5.2. The average profit gained

Table 5.2: Iterative Integration Framework - Results

Instance	Percentage Completion Analysis	Profit
Instance 1 (CP1)		
Scheduling Direct	0.783	3,034,612.2
Integrated Framework	0.876	3,317,631.4
Instance 2 (CP2)		
Scheduling Direct	0.801	3,158,945.6
Integrated Framework	0.894	3,412,204.4
Instance 3 (CP3)		
Scheduling Direct	0.779	2,891,738.9
Integrated Framework	0.881	3,186,696.2
Instance 4 (CP4)		
Scheduling Direct	0.812	3,091,673.9
Integrated Framework	0.903	3,354,466.1
Instance 5 (CP5)		
Scheduling Direct	0.796	3,088,946.5
Integrated Framework	0.91	3,370,040.6

with direct scheduling is $\$3,053,183.4$, while the profit is $\$3,328,207.7$ from the integrated framework, implying a 8.27% increase on average in profit. The results shows that there is benefit in integrating the planning and scheduling models via proposed approach.

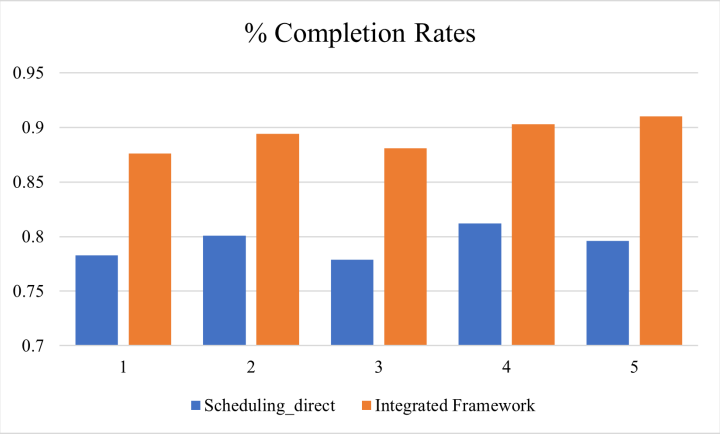


Figure 5.4: Percentage completion rates for the different instances

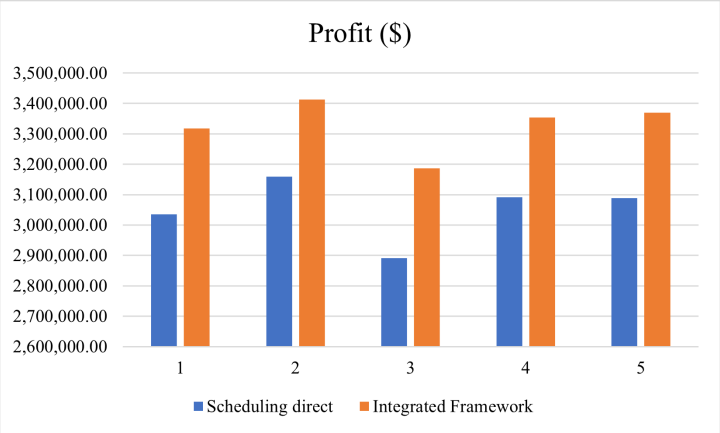


Figure 5.5: Profit graphs for the different instances

With respect to the number of jobs arrived, the size of the planning and scheduling models also vary. The average number of variables/constraints in the planning and scheduling model instances considered in the study and the computational time involved are summarized in table 5.3. Note that, as mentioned in section 5.3.2, for each period, the scheduling model is solved in a sequential scheduling routine, hence, the average number of variables for the scheduling model given in the table represents the variables for each sub-horizon (day).

Table 5.3: Rolling Horizon Approach - Problem size specifications

Avg. number of variables:	
Planning	103141
Scheduling (per sub-horizon)	68490
Avg. number of constraints:	
Planning	103710
Scheduling (per sub-horizon)	48658
Avg. solving time (sec)	3285

To evaluate the benefits in incorporating calibration of the planning model as a key part of the integration framework, the integration was also carried out using an uncalibrated planning model for the same planning horizon as of those presented in table 5.2

and the results are compared. As the planning models are not calibrated, the corresponding instances are represented as P1, P2,..,P5. The results from the computational studies are presented in table 5.4, where integrated framework (P1) represents the results from integration of an uncalibrated planning and scheduling models for a 2 month horizon and the Integrated framework (CP1) represents the results from the calibrated planning and scheduling models for the same 2 month horizon. Both corresponding instances consider the same job distribution. The results show that there is an average increase of 2.8% in the analysis completion rates and 6.6.% increase in terms of profit when the integration is carried out using the calibrated planning model. Note that the increase in the profit is significantly higher than the increase in the analysis completion rates. This is due to the inaccurate estimation of the processing rates and the corresponding required labor hours. As mentioned in section, 5.3.2 the number of required workers is fixed as the value obtained from the planning model and is not updated during the rolling horizon iterations. Inaccurate processing estimates in the planning model resulting from overestimation of the processing capacity of the batch plant results in increased estimate of the required workers. However, when actual plant capacity and job sequences are accounted in the scheduling model, the available labor hours is surplus and affects the profits earned. Regarding the computational time requirements, the total CPU time required to perform integration of calibrated planning and scheduling model is 15525 seconds on average, whereas the total computational time required for integration of an uncalibrated planning and scheduling models is 3285 seconds on average. However, note that the calibration is performed using historical data. Therefore, calibration can be considered as offline simulations that can be performed *a priori* to the rolling horizon method. That is, when solving the integrated model via rolling horizon method, even if the planning model is calibrated or uncalibrated, the average solving time is same as that of reported in the table 5.3. Hence, by performing the calibration of planning model beforehand, the benefits obtained through integration via rolling horizon can be enhanced.

Table 5.4: Integration with uncalibrated planning model - Results

Instance	Percentage Completion Analysis	Profit
Instance 1		
Integrated Framework (P1)	0.868	3,092,172.3
Integrated Framework (CP1)	0.876	3,317,631.4
Instance 2		
Integrated Framework (P2)	0.847	3,209,164.8
Integrated Framework (CP2)	0.894	3,412,204.4
Instance 3		
Integrated Framework (P3)	0.856	2,968,080.8
Integrated Framework (CP3)	0.881	3,186,696.2
Instance 4		
Integrated Framework (P4)	0.878	3,161,545.7
Integrated Framework (CP4)	0.903	3,354,466.1
Instance 5		
Integrated Framework (P5)	0.887	3,177,908.2
Integrated Framework (CP5)	0.91	3,370,040.6

5.4.3 Sensitivity Analysis

In order to analyse the sensitivity of the integration approach with the length of the planning horizon, the variation in results were analysed with respect to change in the length of planning horizon. The results from the computational experiments with variation in the planning horizon are presented in the table 5.5. The results show that the benefits of using the integrated approach increases with increase in the length of the planning horizon. It can be noted that the benefits in terms of profit increases from a 3.84% to 8.53% when the length of planning horizon increases from 2 weeks to 8 weeks. A similar trend can be noted for the percentage completion increases where the average benefits increases from 2.7% to 9%. Increase in the length of time horizon enables more interaction between the planning and scheduling models and provide increased scope for modifying the planning decisions to reflect the actual plant specifications and thus the benefits increases with increase in the length of the horizon.

Table 5.5: Sensitivity Analysis with the length of planning horizon

Instance	Percentage Completion Analysis	Profit
8 weeks		
Scheduling Direct	0.783	3,034,612.19
Scheduling With RH-Planning	0.876	3,317,631.42
6 weeks		
Scheduling Direct	0.788	2,546,742.60
Scheduling With RH-Planning	0.861	2,746,998.81
4 weeks		
Scheduling Direct	0.791	1,491,863.20
Scheduling With RH-Planning	0.82	1,579,192.55
2 weeks		
Scheduling Direct	0.785	842,221.87
Scheduling With RH-Planning	0.812	875,854.69

5.5 Chapter Summary

An iterative framework to integrate an operational planning and scheduling models for large scale multijob industrial operations was presented. The key feature of the study is that the proposed framework considers multijob batch plants were there are no fixed products or recipes and the job recipes vary with respect to the customer specifications. The proposed planning model considered the jobs arrived and supplied daily processing profile including decisions such as labour time requirements and the processing targets to the scheduling model and the models are integrated to obtain realizable and profitable plant/system performance using a iterative integration approach - Rolling Horizon method. Before implementing the rolling horizon approach, a calibration scheme was employed for the planning model to ensure that the model utilizes reasonable estimates of plant information to obtain the key planning decisions. An iterative framework involving the rolling horizon method was incorporated to integrate the calibrated planning model and the scheduling models that account for the variations in the job distribution assumed for the planning model. The computational results show an average increase of 9.8% in the analysis completion rates and 8.27% in terms of profit when the models are integrated via

RH approach. An analysis of the benefits in calibrating the planning model was conducted by comparing the results when integrated the scheduling model with calibrated and uncalibrated planning model respectively. The obtained results show that there is an average increase of 2.8% in the analysis completion rates and 6.6% increase in terms of profit when the integration is carried out using the calibrated planning model. A sensitivity analysis was also conducted to analyse the performance of the proposed framework with variations in the length of the planning horizon. The analysis results show that the benefits (in terms of profits and process completion rates) increases with increase in the length of the planning horizon.

Chapter 6

Conclusion

In this PhD thesis, novel approaches and models were developed to address some of the existing gaps in the literature. In this section, the summary of the main contributions of this thesis is presented along with a discussion on the possible directions for future research.

In chapter 3, a novel two-stage stochastic programming approach was developed for scheduling of batch operations under type II endogenous uncertainty. One of the key challenges in literature when modelling type II endogenous uncertainties were associated with non-anticipativity enforcement. As these uncertainties belong to the endogenous category where the time of realization of the uncertain parameter is dependent on the model decisions, enforcing non-anticipativity usually required introduction of auxiliary binary variables and defining explicit non-anticipativity constraints. Due to the modelling and computational complexity resulting from the introduction of auxiliary binary variables, the studies available in the literature that accounted for such type II uncertainties were limited. The proposed two-stage approach account for such type II uncertainties without introducing any auxiliary binary variables or defining explicit non-anticipativity constraints. Along with the stochastic model, this chapter also includes the proof which shows that careful formulation of the constraints enables implicit non-anticipativity enforcement in the proposed approach.

In chapter 4, the two-stage stochastic approach was modified and a node-based mul-

tistage stochastic approach for scheduling batch operations under type II endogenous uncertainty was developed. The proposed multistage approach enforces non-anticipativity implicitly while also addressing two major limitations of the previously proposed two-stage approach. Unlike the two-stage approach, the multistage approach allows multiple realizations for the uncertain parameter throughout the time horizon and also consider the possibility of multiple tasks that are uncertain in the same sequence (job)). The proposed approach was validated using multiple case studies including an actual industrial case study and two case studies from the literature. The computational studies using these case studies depicts significant benefits in terms of VSS (Value of Stochastic Solution). Each case study exhibited significant increments in the VSS as the number of stages was increased. A comparison study was also conducted between the node-based approach and the conventional binary variable approach (from the literature). The results from the study shows up to 85% reduction in computational time while using the node-based approach in comparison to the binary variable approach.

In chapter 5, An iterative integration framework was proposed based on rolling horizon approach to enable the interaction between the planning and scheduling models for a multijob multitasking batch plant. The effective integration of a planning and scheduling model greatly depends on a planning model that easily interfaces with the scheduling model and provide it with the required input parameters. The main drawbacks of the existing planning models for process industries in the literature includes that it often ignores the job sequence effects and model the plant information using the bottleneck tasks and also that the planning models often fail to provide daily processing profiles to the scheduling model. Considering these limitations in the literature, a long term planning model was developed for a multijob multitasking batch plant that considers approximated sequence constraints and provides key planning decisions including daily processing profiles to the scheduling model. The study further proposes a calibration scheme and an iterative integration scheme to ensure that the estimated information used in the planning model are reasonable and to ensure that the latest job arrivals are accounted for in the planning model. The proposed framework was validated using an actual industrial case study and the computational results show an average increase of 9.8% in the analysis completion rates and 8.27% in

terms of profit when the models are integrated via rolling horizon (RH) approach.

Regarding the future work, there are a few potential aspects that can be explored. One of those aspects include exploring the possibilities for extending the proposed stochastic approach for scheduling under type II endogenous uncertainty to other endogenous uncertainties with unknown time of uncertainty realization that cannot be accounted for in the model using the flow balance constraints (e.g. task processing times). As the proposed approach does not require auxiliary binary variables or explicit non-anticipativity constraints (NACs), if this approach could be adapted to other type II and type I endogenous uncertainties, it would be vastly beneficial in terms of computational costs and applications.

The main goal of the studies conducted in chapters 3 and 4 was to exhibit the applicability and benefits of the proposed two-stage/multistage approach to an industrial application. It is possible that if the considered number of jobs, stages and the nodes were further increased, there maybe an even larger increase in terms of the VSS. However, increasing the number of jobs or the number of stages or number of realizations (nodes) considered in each time-period by a large number may also increase the computational cost substantially. In such cases, efficient decomposition or relaxation strategies may be required to solve such larger instances. Therefore, as future work, developing efficient decomposition strategies/relaxation strategies for solving even larger industrial instances would allow widen the adaptability of the proposed approach.

Chapter 5 focuses on developing an integration framework to solve the long term planning and scheduling models for a large scale multijob batch plant where there are no fixed products/recipes and where the job specifications including recipe depends on the customer specifications. Some of the features that could be considered in the scheduling model in order to increase the efficiency of the plant operations include manual allocation. Current framework provides decisions such as number of required workers to operate the tasks for the considered planning horizon; incorporating manual allocation and assigning the workers to the tasks could help provide better insights of the operational requirements of the plant and also better analysis of the economic aspects of the plant. In addition, incorporating uncertainty modelling into the integration framework for considering uncertainty in

scheduling/planning parameters would further enhance the adaptability of the proposed approach.

References

[1] Christos T Maravelias and Charles Sung. Integration of production planning and scheduling: Overview, challenges and opportunities. *Computers & Chemical Engineering*, 33(12):1919–1930, 2009.

[2] Saman Lagzi, Ricardo Fukasawa, and Luis Ricardez-Sandoval. A multitasking continuous time formulation for short-term scheduling of operations in multipurpose plants. *Computers & Chemical Engineering*, 97:135–146, 2017.

[3] E Kondili, CC Pantelides, and RWH Sargent. A general algorithm for short-term scheduling of batch operations—i. milp formulation. *Computers & Chemical Engineering*, 17(2):211–227, 1993.

[4] Sara Velez and Christos T Maravelias. Multiple and nonuniform time grids in discrete-time mip models for chemical production scheduling. *Computers & Chemical Engineering*, 53:70–85, 2013.

[5] Aurélien Froger, Michel Gendreau, Jorge E Mendoza, Eric Pinson, and Louis-Martin Rousseau. Maintenance scheduling in the electricity industry: A literature review. *European Journal of Operational Research*, 251(3):695–706, 2016.

[6] Martin Steinrücke. Integrated production, distribution and scheduling in the aluminium industry: a continuous-time milp model and decomposition method. *International Journal of Production Research*, 53(19):5912–5930, 2015.

[7] Martijn AH Van Elzakker, Edwin Zondervan, Neha B Raikar, Ignacio E Grossmann, and Peter MM Bongers. Scheduling in the fmcg industry: An industrial case study. *Industrial & engineering chemistry research*, 51(22):7800–7815, 2012.

[8] Dimitris C Paraskevopoulos, Gilbert Laporte, Panagiotis P Repoussis, and Christos D Tarantilis. Resource constrained routing and scheduling: Review and research prospects. *European Journal of Operational Research*, 263(3):737–754, 2017.

[9] Bhushan P Patil, Ricardo Fukasawa, and Luis A Ricardez-Sandoval. Scheduling of operations in a large-scale scientific services facility via multicommodity flow and an optimization-based algorithm. *Industrial & Engineering Chemistry Research*, 54(5):1628–1639, 2015.

[10] Saman Lagzi, Do Yeon Lee, Ricardo Fukasawa, and Luis Ricardez-Sandoval. A computational study of continuous and discrete time formulations for a class of short-term scheduling problems for multipurpose plants. *Industrial & Engineering Chemistry Research*, 56(31):8940–8953, 2017.

[11] Zachariah Stevenson, Ricardo Fukasawa, and Luis Ricardez-Sandoval. A dynamic approach to selecting time points for short-term scheduling with application to multipurpose facilities. *Industrial & Engineering Chemistry Research*, 59(19):9180–9197, 2020.

[12] Nikolaos V Sahinidis. Optimization under uncertainty: state-of-the-art and opportunities. *Computers & Chemical Engineering*, 28(6-7):971–983, 2004.

[13] M. G. Ierapetritou, J. Acevedo, and E. N. Pistikopoulos. An optimization approach for process engineering problems under uncertainty. *Computers & Chemical Engineering*, 20(6-7):703–709, 1996.

[14] John R Birge and Francois Louveaux. *Introduction to stochastic programming*. Springer Science & Business Media, 2011.

[15] Alexander Shapiro and Andy Philpott. A tutorial on stochastic programming. *Manuscript. Available at www2. isye. gatech. edu/ashapiro/publications. html*, 17, 2007.

[16] Vikas Goel and Ignacio E Grossmann. A class of stochastic programs with decision dependent uncertainty. *Mathematical programming*, 108(2-3):355–394, 2006.

[17] Robert M Apap and Ignacio E Grossmann. Models and computational strategies for multistage stochastic programming under endogenous and exogenous uncertainties. *Computers & Chemical Engineering*, 103:233–274, 2017.

[18] Vijay Gupta and Ignacio E Grossmann. Solution strategies for multistage stochastic programming with endogenous uncertainties. *Computers & Chemical Engineering*, 35(11):2235–2247, 2011.

[19] Matthew Colvin and Christos T Maravelias. A stochastic programming approach for clinical trial planning in new drug development. *Computers & Chemical Engineering*, 32(11):2626–2642, 2008.

[20] Matthew Colvin and Christos T Maravelias. Modeling methods and a branch and cut algorithm for pharmaceutical clinical trial planning using stochastic programming. *European Journal of Operational Research*, 203(1):205–215, 2010.

[21] Thomas Staeblein and Katsuki Aoki. Planning and scheduling in the automotive industry: A comparison of industrial practice at german and japanese makers. *International Journal of Production Economics*, 162:258–272, 2015.

[22] Peter Michael Verderame. *Planning and scheduling of batch processes under uncertainty*. princeton university, 2011.

[23] Peter M Verderame, Josephine A Elia, Jie Li, and Christodoulos A Floudas. Planning and scheduling under uncertainty: a review across multiple sectors. *Industrial & engineering chemistry research*, 49(9):3993–4017, 2010.

[24] Kavitha G Menon, Ricardo Fukasawa, and Luis A Ricardez-Sandoval. A novel stochastic programming approach for scheduling of batch processes with decision dependent time of uncertainty realization. *Annals of Operations Research*, pages 1–28, 2021.

[25] Hojae Lee and Christos T Maravelias. Combining the advantages of discrete-and continuous-time scheduling models: Part 1. framework and mathematical formulations. *Computers & Chemical Engineering*, 116:176–190, 2018.

[26] Sara Velez and Christos T Maravelias. Theoretical framework for formulating mip scheduling models with multiple and non-uniform discrete-time grids. *Computers & Chemical Engineering*, 72:233–254, 2015.

[27] Christodoulos A Floudas and Xiaoxia Lin. Continuous-time versus discrete-time approaches for scheduling of chemical processes: a review. *Computers & Chemical Engineering*, 28(11):2109–2129, 2004.

[28] Christos T Maravelias and Ignacio E Grossmann. New general continuous-time state-task network formulation for short-term scheduling of multipurpose batch plants. *Industrial & engineering chemistry research*, 42(13):3056–3074, 2003.

[29] Carlos A Méndez, Jaime Cerdá, Ignacio E Grossmann, Iiro Harjunkoski, and Marco Fahl. State-of-the-art review of optimization methods for short-term scheduling of batch processes. *Computers & Chemical Engineering*, 30(6-7):913–946, 2006.

[30] Ignacio E Grossmann, Robert M Apap, Bruno A Calfa, Pablo García-Herreros, and Qi Zhang. Recent advances in mathematical programming techniques for the optimization of process systems under uncertainty. *Computers & Chemical Engineering*, 91:3–14, 2016.

[31] Mina Rafiei and Luis A Ricardez-Sandoval. New frontiers, challenges, and opportunities in integration of design and control for enterprise-wide sustainability. *Computers & Chemical Engineering*, 132:106610, 2020.

[32] Yun Ye, Jie Li, Zukui Li, Qiuhua Tang, Xin Xiao, and Christodoulos A Floudas. Robust optimization and stochastic programming approaches for medium-term production scheduling of a large-scale steelmaking continuous casting process under demand uncertainty. *Computers & Chemical Engineering*, 66:165–185, 2014.

[33] Aharon Ben-Tal and Arkadi Nemirovski. Selected topics in robust convex optimization. *Mathematical Programming*, 112(1):125–158, 2008.

[34] Amir Beck and Aharon Ben-Tal. Duality in robust optimization: primal worst equals dual best. *Operations Research Letters*, 37(1):1–6, 2009.

[35] Xiaoxia Lin, Stacy L Janak, and Christodoulos A Floudas. A new robust optimization approach for scheduling under uncertainty:: I. bounded uncertainty. *Computers & chemical engineering*, 28(6-7):1069–1085, 2004.

[36] Nikolaos H Lappas and Chrysanthos E Gounaris. Robust optimization for decision-making under endogenous uncertainty. *Computers & Chemical Engineering*, 111:252–266, 2018.

[37] Aharon Ben-Tal, Alexander Goryashko, Elana Guslitzer, and Arkadi Nemirovski. Adjustable robust solutions of uncertain linear programs. *Mathematical programming*, 99(2):351–376, 2004.

[38] Hanyu Shi and Fengqi You. Adjustable robust optimization for scheduling of batch processes under uncertainty. In *Computer Aided Chemical Engineering*, volume 38, pages 547–552. Elsevier, 2016.

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

[40] Frans JCT de Ruiter, Jianzhe Zhen, and Dick den Hertog. Dual approach for two-stage robust nonlinear optimization. *Available on Optimization-Online*, 2018.

[41] Julia L Higle. Stochastic programming: Optimization when uncertainty matters. In *Emerging Theory, Methods, and Applications*, pages 30–53. Informs, 2005.

[42] Sebastian Engell, Andreas Märkert, Guido Sand, and Rüdiger Schultz. Aggregated scheduling of a multiproduct batch plant by two-stage stochastic integer programming. *Optimization and Engineering*, 5(3):335–359, 2004.

[43] J Balasubramanian and IE Grossmann. Approximation to multistage stochastic optimization in multiperiod batch plant scheduling under demand uncertainty. *Industrial & engineering chemistry research*, 43(14):3695–3713, 2004.

[44] Zhengyang Hu and Guiping Hu. A multi-stage stochastic programming for lot-sizing and scheduling under demand uncertainty. *Computers & Industrial Engineering*, 119:157–166, 2018.

[45] John R Birge. State-of-the-art-survey—stochastic programming: Computation and applications. *INFORMS journal on computing*, 9(2):111–133, 1997.

[46] Vikas Goel and Ignacio E Grossmann. A stochastic programming approach to planning of offshore gas field developments under uncertainty in reserves. *Computers & chemical engineering*, 28(8):1409–1429, 2004.

[47] Miloš Kopa and Tomáš Rusỳ. A decision-dependent randomness stochastic program for asset–liability management model with a pricing decision. *Annals of Operations Research*, 299(1):241–271, 2021.

[48] Fengqiao Luo and Sanjay Mehrotra. Distributionally robust optimization with decision dependent ambiguity sets. *Optimization Letters*, 14(8):2565–2594, 2020.

[49] Omid Nohadani and Kartikey Sharma. Optimization under decision-dependent uncertainty. *SIAM Journal on Optimization*, 28(2):1773–1795, 2018.

[50] Zukui Li and Marianthi Ierapetritou. Process scheduling under uncertainty: Review and challenges. *Computers & Chemical Engineering*, 32(4-5):715–727, 2008.

[51] Thomas Eiter and Georg Gottlob. On the computational cost of disjunctive logic programming: Propositional case. *Annals of Mathematics and Artificial Intelligence*, 15(3):289–323, 1995.

[52] Pedro M Castro and Ignacio E Grossmann. Generalized disjunctive programming as a systematic modeling framework to derive scheduling formulations. *Industrial & Engineering Chemistry Research*, 51(16):5781–5792, 2012.

[53] Tore W Jonsbråten, Roger JB Wets, and David L Woodruff. A class of stochastic programs withdecision dependent random elements. *Annals of Operations Research*, 82:83–106, 1998.

[54] Can Li and Ignacio E Grossmann. A review of stochastic programming methods for optimization of process systems under uncertainty. *Frontiers in Chemical Engineering*, 2:622241, 2021.

[55] Bora Tarhan, Ignacio E Grossmann, and Vikas Goel. Stochastic programming approach for the planning of offshore oil or gas field infrastructure under decision-dependent uncertainty. *Industrial & Engineering Chemistry Research*, 48(6):3078–3097, 2009.

[56] Vijay Gupta and Ignacio E Grossmann. Multistage stochastic programming approach for offshore oilfield infrastructure planning under production sharing agreements and endogenous uncertainties. *Journal of Petroleum Science and Engineering*, 124:180–197, 2014.

[57] Vijay Gupta and Ignacio E Grossmann. A new decomposition algorithm for multistage stochastic programs with endogenous uncertainties. *Computers & Chemical Engineering*, 62:62–79, 2014.

[58] Bora Tarhan and Ignacio E Grossmann. A multistage stochastic programming approach with strategies for uncertainty reduction in the synthesis of process networks with uncertain yields. *Computers & Chemical Engineering*, 32(4-5):766–788, 2008.

[59] Matthew Colvin and Christos T Maravelias. Scheduling of testing tasks and resource planning in new product development using stochastic programming. *Computers & Chemical Engineering*, 33(5):964–976, 2009.

[60] Matthew Colvin and Christos T Maravelias. R&d pipeline management: Task interdependencies and risk management. *European Journal of Operational Research*, 215(3):616–628, 2011.

[61] Natashia Boland, Irina Dumitrescu, and Gary Froyland. A multistage stochastic programming approach to open pit mine production scheduling with uncertain geology. *Optimization online*, pages 1–33, 2008.

[62] Boris Defourny, Damien Ernst, and Louis Wehenkel. Multistage stochastic programming: A scenario tree based approach to planning under uncertainty. In *Decision theory models for applications in artificial intelligence: concepts and solutions*, pages 97–143. IGI Global, 2012.

[63] SJ Wilkinson, N Shah, and CC Pantelides. Aggregate modelling of multipurpose plant operation. *Computers & chemical engineering*, 19:583–588, 1995.

[64] Christian H Timpe and Josef Kallrath. Optimal planning in large multi-site production networks. *European journal of operational research*, 126(2):422–435, 2000.

[65] Josef Kallrath. Combined strategic and operational planning–an milp success story in chemical industry. *Or Spectrum*, 24(3):315–341, 2002.

[66] Josef Kallrath. Solving planning and design problems in the process industry using mixed integer and global optimization. *Annals of Operations Research*, 140(1):339–373, 2005.

[67] Matthew H Bassett, Joseph F Pekny, and Gintaras V Reklaitis. Decomposition techniques for the solution of large-scale scheduling problems. *AIChE Journal*, 42(12):3373–3387, 1996.

[68] Peter M Verderame and Christodoulos A Floudas. Integrated operational planning and medium-term scheduling for large-scale industrial batch plants. *Industrial & Engineering Chemistry Research*, 47(14):4845–4860, 2008.

[69] Charles Sung and Christos T Maravelias. An attainable region approach for production planning of multiproduct processes. *AIChE Journal*, 53(5):1298–1315, 2007.

[70] Peter M Verderame and Christodoulos A Floudas. Integration of operational planning and medium-term scheduling for large-scale industrial batch plants under demand and processing time uncertainty. *Industrial & engineering chemistry research*, 49(10):4948–4965, 2010.

[71] Oswaldo Andrés-Martínez and Luis A Ricardez-Sandoval. Integration of planning, scheduling, and control: A review and new perspectives. *The Canadian Journal of Chemical Engineering*.

[72] Yanshuo Peng and Luis Ricardez-Sandoval. Integration of planning, scheduling, and control for multi-product chemical systems under preventive maintenance. *IFAC-PapersOnLine*, 54(3):536–541, 2021.

[73] Héctor Uriel Rodríguez Vera and Luis A. Ricardez-Sandoval. Integration of scheduling and control for chemical batch plants under stochastic uncertainty: A back-off approach. *Industrial & Engineering Chemistry Research*, 61(12):4363–4378, 2022.

[74] Arnoldo C Hax and Harlan C Meal. Hierarchical integration of production planning and scheduling. 1973.

[75] Dan Wu and Marianthi Ierapetritou. Hierarchical approach for production planning and scheduling under uncertainty. *Chemical Engineering and Processing: Process Intensification*, 46(11):1129–1140, 2007.

[76] ACS Amaro and Ana Paula FD Barbosa-Póvoa. Planning and scheduling of industrial supply chains with reverse flows: A real pharmaceutical case study. *Computers & Chemical Engineering*, 32(11):2606–2625, 2008.

[77] Susara A van den Heever and Ignacio E Grossmann. A strategy for the integration of production planning and reactive scheduling in the optimization of a hydrogen supply network. *Computers & Chemical Engineering*, 27(12):1813–1839, 2003.

[78] Zukui Li and Marianthi G Ierapetritou. Production planning and scheduling integration through augmented lagrangian optimization. *Computers & Chemical Engineering*, 34(6):996–1006, 2010.

[79] Nikisha K Shah and Marianthi G Ierapetritou. Integrated production planning and scheduling optimization of multisite, multiproduct process industry. *Computers & Chemical Engineering*, 37:214–226, 2012.

[80] Arthur M Geoffrion. Generalized benders decomposition. *Journal of optimization theory and applications*, 10(4):237–260, 1972.

[81] Monique Guignard and Siwhan Kim. Lagrangean decomposition: A model yielding stronger lagrangean bounds. *Mathematical programming*, 39(2):215–228, 1987.

[82] T Majozi and) XX Zhu. A novel continuous-time milp formulation for multipurpose batch plants. 1. short-term scheduling. *Industrial & Engineering Chemistry Research*, 40(25):5935–5949, 2001.

[83] Muge Erdirik Dogan and Ignacio E Grossmann. A decomposition method for the simultaneous planning and scheduling of single-stage continuous multiproduct plants. *Industrial & engineering chemistry research*, 45(1):299–315, 2006.

[84] Yunfei Chu, Fengqi You, John M Wassick, and Anshul Agarwal. Integrated planning and scheduling under production uncertainties: Bi-level model formulation and hybrid solution method. *Computers & Chemical Engineering*, 72:255–272, 2015.

[85] AD Dimitriadis, N Shah, and CC Pantelides. Rtn-based rolling horizon algorithms for medium term scheduling of multipurpose plants. *Computers & Chemical Engineering*, 21:S1061–S1066, 1997.

[86] Alistair R Clark. Rolling horizon heuristics for production planning and set-up scheduling with backlogs and error-prone demand forecasts. *Production Planning & Control*, 16(1):81–97, 2005.

[87] Zukui Li and Marianthi G Ierapetritou. Rolling horizon based planning and scheduling integration with production capacity consideration. *Chemical Engineering Science*, 65(22):5887–5900, 2010.

[88] Jian Fang and Yugeng Xi. A rolling horizon job shop rescheduling strategy in the dynamic environment. *The International Journal of Advanced Manufacturing Technology*, 13(3):227–232, 1997.

[89] Aldo Bischi, Leonardo Taccari, Emanuele Martelli, Edoardo Amaldi, Giampaolo Manzolini, Paolo Silva, Stefano Campanari, and Ennio Macchi. A rolling-horizon optimization algorithm for the long term operational scheduling of cogeneration systems. *Energy*, 184:73–90, 2019.

[90] Ronald L Graham. Bounds for certain multiprocessing anomalies. *Bell system technical journal*, 45(9):1563–1581, 1966.

[91] Egon Balas, Neil Simonetti, and Alkis Vazacopoulos. Job shop scheduling with setup times, deadlines and precedence constraints. *Journal of Scheduling*, 11(4):253–262, 2008.

[92] Paolo Serafini. Scheduling jobs on several machines with the job splitting property. *Operations Research*, 44(4):617–628, 1996.

[93] Chris N Potts and Mikhail Y Kovalyov. Scheduling with batching: A review. *European journal of operational research*, 120(2):228–249, 2000.

[94] Fernando Santos, Ricardo Fukasawa, and Luis Ricardez-Sandoval. An integrated machine scheduling and personnel allocation problem for large-scale industrial facilities using a rolling horizon framework. *Optimization and Engineering*, 22(4):2603–2626, 2021.

[95] Jonathan L Gross, Jay Yellen, and Ping Zhang. *Handbook of graph theory*. CRC press, 2013.

[96] Suvrajeet Sen and Julia L Higle. An introductory tutorial on stochastic linear programming models. *Interfaces*, 29(2):33–61, 1999.

[97] Andrzej Ruszczyński. Parallel decomposition of multistage stochastic programming problems. *Mathematical Programming*, 58:201–228, 1993.

[98] Josef Kallrath. Planning and scheduling in the process industry. *OR spectrum*, 24(3):219–250, 2002.

[99] Jeff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B Shah. Julia: A fresh approach to numerical computing. *SIAM review*, 59(1):65–98, 2017.

[100] Michel Goossens, Frank Mittelbach, and Alexander Samarin. *The LaTeX Companion*. Addison-Wesley, Reading, Massachusetts, 1994.

[101] Donald Knuth. *The TeXbook*. Addison-Wesley, Reading, Massachusetts, 1986.

[102] Leslie Lamport. *LaTeX — A Document Preparation System*. Addison-Wesley, Reading, Massachusetts, second edition, 1994.

[103] Christos T Maravelias and Ignacio E Grossmann. On the relation of continuous and discrete time models for the state-task network formulation. *AIChE J*, 52(2):843–849, 2006.

[104] Muge Erdirik-Dogan and Ignacio E Grossmann. Planning models for parallel batch reactors with sequence-dependent changeovers. *AIChE Journal*, 53(9):2284–2300, 2007.

[105] Sebastian Terrazas-Moreno and Ignacio E Grossmann. A multiscale decomposition method for the optimal planning and scheduling of multi-site continuous multiproduct plants. *Chemical Engineering Science*, 66(19):4307–4318, 2011.

[106] Spas B Petkov and Costas D Maranas. Multiperiod planning and scheduling of multiproduct batch plants under demand uncertainty. *Industrial & engineering chemistry research*, 36(11):4864–4881, 1997.

[107] Xin Chen and Yuhan Zhang. Uncertain linear programs: Extended affinely adjustable robust counterparts. *Operations Research*, 57(6):1469–1482, 2009.

[108] Laurence A Wolsey and George L Nemhauser. *Integer and combinatorial optimization*, volume 55. John Wiley & Sons, 1999.

[109] Francisco Trespalacios and Ignacio E Grossmann. Improved big-m reformulation for generalized disjunctive programs. *Computers & Chemical Engineering*, 76:98–103, 2015.

[110] H Paul Williams. *Model building in mathematical programming*. John Wiley & Sons, 2013.

[111] P Castro, APFD Barbosa-Póvoa, and H Matos. An improved rtn continuous-time formulation for the short-term scheduling of multipurpose batch plants. *Industrial & engineering chemistry research*, 40(9):2059–2068, 2001.

[112] Iiro Harjunkoski, Christos T Maravelias, Peter Bongers, Pedro M Castro, Sebastian Engell, Ignacio E Grossmann, John Hooker, Carlos Méndez, Guido Sand, and John Wassick. Scope for industrial applications of production scheduling models and solution methods. *Computers & Chemical Engineering*, 62:161–193, 2014.

[113] Marianthi Ierapetritou and Zukui Li. Modeling and managing uncertainty in process planning and scheduling. In *Optimization and logistics challenges in the enterprise*, pages 97–144. Springer, 2009.

[114] Laureano F Escudero, Araceli Garín, María Merino, and Gloria Pérez. The value of the stochastic solution in multistage problems. *Top*, 15(1):48–64, 2007.

[115] Karl G Kempf, Pınar Keskinocak, and Reha Uzsoy. Planning production and inventories in the extended enterprise: A state of the art handbook, volume 1. 2011.

[116] Susara A Van Den Heever and Ignacio E Grossmann. An iterative aggregation/disaggregation approach for the solution of a mixed-integer nonlinear oilfield infrastructure planning model. *Industrial & engineering chemistry research*, 39(6):1955–1971, 2000.

[117] Sriram Subrahmanyam, Joseph F Pekny, and Gintaras V Reklaitis. Decomposition approaches to batch plant design and planning. *Industrial & engineering chemistry research*, 35(6):1866–1876, 1996.

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

APPENDICES

Appendix A

Normalized process data of the actual industrial plant

The normalized data for the 189 tasks in the scientific services facility are presented below in Table A.1.

Table A.1: Normalized process data used in experiments.

Begin of Table				
Tasks	Capacity	Processing time	Resources	Labor time
1	0.09991	0.001389027	2	0.015
2	0.005901	0.005853755	3	0.06
3	0.005401	0.047524556	1	0.45
4	9.90E-05	0.1427721	4500	0
5	0.049905	0.037106856	4	0.375
6	0.0049	0.018553428	10	0.188
7	0.0049	0.099116976	2	1
8	0.0049	0.018553428	6	0.188

Continuation of Table A.1

Tasks	Capacity	Processing time	Resources	Labor time
9	0.0049	0.019744022	2	0.2
10	0.0002	0.000694513	1	0.008
11	0.0049	0.019744022	3	0.15
12	0.0049	0.014783213	1	0.05
13	0.0049	0.004861593	3	0.015
14	0.010901	0.107054271	1	0.1
15	0.0049	0.009822403	1	0.04
16	0.0041	0.003869431	2	0.04
17	0.0041	0.003869431	2	0.06
18	0.0029	0.1011013	4	0.12
19	0.021502	0.029665641	2	0.032
20	0.002	0.014783213	2	0.095
21	0.0007	0.01875186	1	0.045
22	0.0047	0.060918742	1	0.01
23	9.90E-05	0.016469888	1	0.015
24	0.0006	0.1427721	1	0.24
25	0.005801	0.209346165	1	0.182
26	0.0039	0.165095744	2	0.3
27	0.0039	0.744022224	40	0.18
28	0.0029	0.035618613	2	0.07
29	0.0007	0.053477528	1	0.03
30	0.005301	0.050501042	4	0.06
31	0.0029	0.005853755	1	0.045
32	0.0026	0.011310646	1	0.055
33	0.0026	0.040579423	1	0.06
34	0.0025	0.013791051	1	0.06
35	0.0025	0.013791051	1	0.06
36	0.0025	0.013791051	1	0.02

Continuation of Table A.1

Tasks	Capacity	Processing time	Resources	Labor time
37	0.0024	0.012402024	1	0.04
38	0.0023	0.013791051	1	0.025
39	0.0026	0.010814565	1	0.12
40	0.019902	0.011806727	1	0.12
41	0.010101	0.011806727	2	0.01
42	0.008101	0.000892946	2	0.12
43	0.030403	0.011806727	2	0.24
44	0.014901	0.02371267	1	0.06
45	0.010201	0.005853755	1	0.18
46	0.012401	0.017759698	1	0.18
47	0.012401	0.017759698	1	0.06
48	0.0011	0.005853755	1	0.09
49	0.005801	0.008830241	1	0.02
50	0.0019	0.001885108	1	0.06
51	0.005901	0.005853755	1	0.06
52	0.0005	0.005853755	2	0.06
53	0.0039	0.005853755	1	0.06
54	0.005901	0.005853755	1	0.06
55	0.0029	0.005853755	2	0.18
56	0.047905	0.017759698	3	0.24
57	0.043904	0.02371267	2	0.42
58	0.009901	0.041571584	1	0.01
59	0.0049	0.098124814	1	0.045
60	0.0011	0.01329497	2	0.03
61	0.0029	0.00287727	1	0.15
62	0.0007	0.026689156	3	0.12
63	0.021502	0.011806727	4	0.22
64	0.043104	0.021728346	2	0.22

Continuation of Table A.1

Tasks	Capacity	Processing time	Resources	Labor time
65	0.043904	0.021728346	4	0.06
66	0.0029	0.005853755	5	0.005
67	0.007101	0.014287132	1	0.005
68	0.006701	0.026193075	1	0.01
69	0.005101	0.048516718	1	0.02
70	0.0029	0.066871713	1	0.005
71	0.000099	0.024208751	2	0.005
72	0.0017	0.048020637	1	0.015
73	0.0001	0.001885108	2	0.06
74	0.005901	0.005853755	1	0.005
75	0.0041	0.006349836	2	0.01
76	0.0041	0.000892946	2	0.01
77	0.008301	0.005853755	1	0.252
78	0.0005	0.032642127	2	0.042
79	0.0041	0.004067864	2	0.005
80	0.0017	0.001389027	2	0.005
81	0.0041	0.001389027	2	0.01
82	0.0041	0.007838079	2	0.01
83	0.0017	0.007838079	2	0.042
84	0.0041	0.004067864	1	0.042
85	0.0041	0.004067864	2	0.192
86	0.0013	0.047524556	1	0.007
87	0.0029	0.006647485	1	0.02
88	0.012701	0.072923901	4	0.064
89	0.012701	0.00625062	2	0
90	0.000099	0.000892946	1	0
91	0.000099	0.1427721	1	0.06
92	0.014901	0.1427721	2	0.035

Continuation of Table A.1

Tasks	Capacity	Processing time	Resources	Labor time
93	0.0025	0.012798889	1	0.035
94	0.0025	0.012798889	1	0.035
95	0.0024	0.013890267	1	0.025
96	0.0025	0.011806727	1	0.025
97	0.0026	0.042563746	1	0.12
98	0.017902	0.038595099	1	0.06
99	0.023902	0.1427721	1	0.025
100	0.011301	0.314912194	1	0.105
101	0.011501	0.017362834	1	0.03
102	0.071907	0.072824685	10	0.03
103	0.047905	0.04663161	10	0.03
104	0.071907	0.070443496	10	0.03
105	0.016702	0.017759698	10	0.03
106	0.029903	0.024704832	10	0.03
107	0.137914	0.138307372	10	0.195
108	0.035904	0.025498561	2	0.12
109	0.011101	0.124516321	8	0.12
110	0.013401	0.113106459	8	0.15
111	0.008101	0.025101697	8	0.12
112	0.008101	0.142573668	8	0.12
113	0.007301	0.126996726	8	0.024
114	0.051605	0.159142772	8	0.021
115	0.026903	0.077884711	8	0.012
116	0.026303	0.133842643	1	0.9
117	0.020602	0.136819129	5	0.003
118	0.001	0.000198432	6	0.01
119	0.0029	0.000892946	1	0.01
120	0.0029	0.000892946	1	0.03

Continuation of Table A.1

Tasks	Capacity	Processing time	Resources	Labor time
121	0.001	0.00287727	1	0.007
122	0.001	0.000595297	1	0
123	0.047905	0.047524556	1	0
124	0.053205	0.047524556	1	0
125	0.014301	0.047524556	1	0
126	0.035904	1	1	0.42
127	0.071907	1	1	0.3
128	0.009901	0.041571584	1	0.062
129	0.008901	0.029665641	2	0.062
130	0.005301	0.012798889	1	0.062
131	0.0011	0.124913186	1	0.062
132	0.005301	0.102093462	1	0.167
133	0.005301	0.012798889	1	0.167
134	0.0015	0.026193075	1	0.062
135	0.0015	0.029665641	1	0.02
136	0.005301	0.019744022	1	0.08
137	0.0025	0.075404306	1	0.02
138	0.005301	0.007838079	2	0.09
139	0.0025	0.030459371	1	0.003
140	0.005301	0.008830241	4	0.07
141	0.009501	0.064688957	3	0.074
142	0.009501	0.006845917	4	0.264
143	0.014301	0.020438536	1	0.224
144	0.010101	0.090981248	1	0.224
145	0.010101	0.04038099	1	0.015
146	0.010101	0.04038099	1	0.045
147	0.005901	0.005853755	1	0.45
148	0.0013	0.166583987	2	0.15

Continuation of Table A.1

Tasks	Capacity	Processing time	Resources	Labor time
149	0.0013	0.04454807	2	0.005
150	0.0007	0.041571584	2	0.015
151	0.0007	0.018255779	1	0.03
152	0.0007	0.001389027	2	0.025
153	0.005901	0.00287727	2	0.032
154	0.0021	0.002381189	2	0.002
155	0.0007	0.003075702	1	0.036
156	0.000099	0.0000992162	1	0.42
157	0.0017	0.029665641	1	0.06
158	0.0026	0.113007243	1	0.06
159	0.0021	0.005853755	1	0.001
160	0.0019	0.089195357	1	0.001
161	0.000099	0.001488243	1	0.75
162	0.000099	0.000099	1	0.75
163	0.005901	0.105566028	1	0.75
164	0.005901	0.105566028	1	0.06
165	0.005901	0.105566028	1	0.12
166	0.043904	0.160631015	4	0.09
167	0.043904	0.166583987	4	0.16
168	0.0023	0.056454013	1	0.24
169	0.0015	0.158646691	1	0.03
170	0.0023	0.02371267	1	0.06
171	0.0005	0.038595099	1	0.01
172	0.0019	0.006052188	1	0.082
173	0.0011	0.012302808	1	0.001
174	0.0031	0.008830241	1	0.001
175	0.000099	0.000099	6	0.015
176	0.000099	0.000099	1	0.001

Continuation of Table A.1

Tasks	Capacity	Processing time	Resources	Labor time
177	0.0014	0.001389027	1	0.01
178	0.000099	0.000099	1	0.001
179	0.000099	0.00079373	1	0.01
180	0.000099	0.000099	1	0.024
181	0.000099	0.000892946	3	0.015
182	0.0011	0.016569104	1	0.001
183	0.000099	0.001389027	1	0.012
184	0.000099	0.000099	1	0.001
185	0.0005	0.00208354	5	0
186	0.000099	0.000099	2	0.081
187	1	0.000892946	1	0.009
188	0.09991	0.1427721	1	0.076
189	0.09991	0.1427721	1	0.016

End of Table

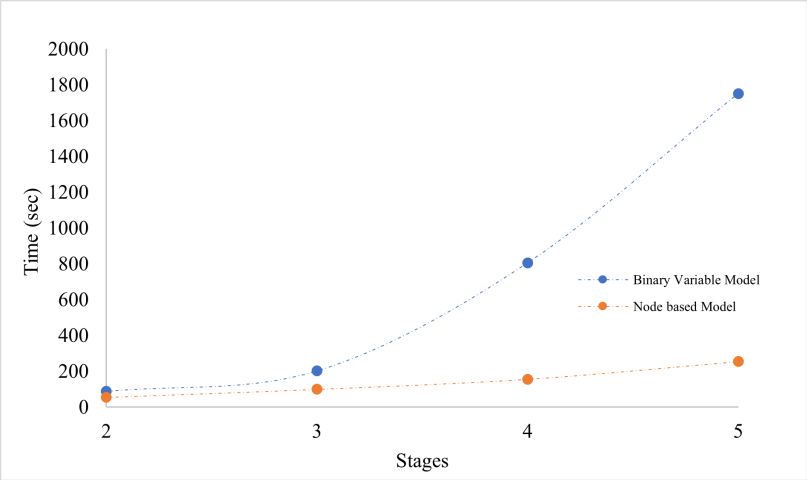
Appendix B

Multistage Approach - Supplementary Information

B.1 Multistage Model - Performance Comparison

To provide better insights on the model performance, detailed discussions on computational time of binary variable model and node based model are provided here. For the comparison purposes, the same instance of 10 jobs and a scheduling horizon of 24hrs are solved using the binary variable approach and node based approach (presented in section 4.3.2). The number of stages are varied from two to five. Variations in the model computational time with increase in the number of stages for the instance are presented in Table 4.4 and the values are plotted in Figure B.1. As shown in Figure B.1, with increase in number of stages, the computational time also increases for both approaches. However, the increase in the computational time for the binary variable approach tend to follow an exponential behavior and the actual CPU times are much higher than the proposed node based approach. The exponential increase in the computational time for the binary variable approach shows that the approach is not preferable for large multistage problems. Hence, the proposed node based approach where the computational time requirements are lesser than the binary variable approach emerge as a potential solution to handle large multistage problems.

Figure B.1: Variations in computational time with increase in stages



B.2 Transformation of STN to Graph representation

Discussions on transformation from an STN representation to graph representation is presented in this section. While transforming a representation from STN to graph, one of the key factor is the components involved in the network representation. As discussed in section 4.4, STN representation also involves explicit representation of the units (states) in contrast to the graph representation that involves representation of tasks and the incoming/outgoing streams. In order to provide better understanding, the transformation is explained using the case study presented in section 4.5.3. The case study in figure 4.9 includes 5 products with 5 unique recipes. Each product is considered as a different job and is represented in Figure B.2. The figure includes the network graph representation of the problem (Figure B.2 (a)) and the job-wise graph representation (Figure B.2 (b)). The graph representation of each job includes the tasks and the incoming/outgoing streams associated with the task.

B.3 Binary Variable Model

In this section, the multistage model is presented that was used to compare against the proposed model in Section 4.5.5. Note that the goal was to develop a basic model using standard integer programming modeling techniques to enforce non-anticipativity. In what follows, a detailed description of such model is presented

The idea is that, from the proposed model, all features added for the purposes of enforcing non-anticipativity are removed, keeping the rest of the variables and constraints. Specifically, the decision variables in the proposed model were retained, with the following modification:

• Variables x^n_{ikt} and y^n_{ikt} are still defined for $n \in N_m$, but only for times in time period m, that is, for $t \in \varepsilon'(P^i_k)_m$.

The main idea is that while taking decisions about time period m and a node in N_m, we should only be looking at the variables for times within that time period.

The objective function and constraints (3.1)-(3.6) also remain the same, since these have no effect on non-anticipativity. Constraints (4.2) and (4.3) are replaced by:

$$W^n_{ikt} = y^n_{ikt} + y_{ikt} \quad ,\forall m \in M, n \in N_m, i \in I, k \in 1, \dots, q_i, t \in \varepsilon'(P^i_k)_m$$

$$V^n_{ikt} = x^n_{ikt} + x_{ikt} \quad ,\forall m \in M, n \in N_m, i \in I, k \in 1, \dots, q_i, t \in \varepsilon'(P^i_k)_m$$

Constraints (4.4) are replaced to reflect the new expression of (W, V) variables as follows.

$$V^n_{ikt} + W^n_{ikt} = V^{n(t-1)}_{ikt-1} + \sum_{k' \in N^-_G(P^i_k)} \sum_{r \in \theta(i,k,k',t)} \rho^n_{ik'k} W^{n(r)}_{ik'r} \quad ,\forall m \in M, n \in N_m, i \in I, k \in 1, \dots, q_i, t \in \varepsilon'(P^i_k)_m$$

Constraints (4.5) and (4.6) are removed since these were the constraints used to enforce non-anticipativity implicitly in the proposed model. Finally, constraints (3.2) are kept.

With only the above constraints, the proposed model would not enforce non-anticipativity. For that purpose, the following variables were added:

• Binary variables (L^n_{ikt}) are introduced to check if non-anticipativity must be enforced for the k-th task in path of job $i \in I$, for a node whose parent node is n at time point t, for all $m \in M$, $n \in N_{m-1}$ (Note that here these variables are defined for every task k as this study considers a non-uniform time discretization, where the total number of time points for each task may vary with respect to the time step defined, as mentioned in Section 3.1.2).

• Binary variables ($E^n_{ij't}$) are also introduced to identify when an imperfect task $P^i_{j'}$ is finished between times $t-1$ and t. Note that j' represent the index of an imperfect task of a job i (i.e., $P^i_{j'} \in \Im_i$)

With these variables, one can define "Big-M type" constraints to say that all the model decision variables must be equal until the time of uncertainty realization. The binary variable L^n_{ikt} takes a value 1 when the uncertainty is realized; hence all the model decisions for task k must be equal when $L^n_{ikt} = 0$.

The constraints that define when these binary variables become true or false are as follows:

$$C_{j'}R_{j'}E^n_{ij't} - W^{n(r)}_{ij'r} \geq 0 \quad \forall m \in M, n \in N_m, i \in I, j' \in q_i : P^i_{j'} \in \Im_i, t \in \varepsilon'(P^i_{j'})_m, r \in \theta(i, j', j', t) \tag{B.1}$$

$$E^n_{ij't} - W^{n(r)}_{ij'r} \leq 0 \quad \forall m \in M, n \in N_m, i \in I, j' \in q_i : P^i_{j'} \in \Im_i, t \in \varepsilon'(P^i_{j'})_m, r \in \theta(i, j', j', t) \tag{B.2}$$

$$\begin{aligned} L^{p(n)}_{ikt} \leq E^n_{ij't'} \quad & \forall m \in M, n, n' \in N_m : p(n) = p(n'), i \in I, k = 1, \ldots, q_i, \\ & j' \in q_i : P^i_{j'} \in \Im_i, t = \left(\varepsilon'(P^i_k)_m\right)_1, t' \in \varepsilon'(P^i_{j'})_m : \varepsilon(P^i_{j'}, t') = \varepsilon(P^i_k, t) \end{aligned} \tag{B.3}$$

$$L_{ikt}^{p(n)} \leq L_{ikt-1}^{p(n)} + E_{ij't'}^{n} \quad \forall m \in M, n, n' \in N_m : p(n) = p(n'), i \in I, k = 1, \ldots, q_i, \\ j' \in q_i : P_{j'}^{i} \in \Im_i, t \in \varepsilon'(P_k^i)_m \setminus \left(\varepsilon'(P_k^i)_m\right)_1, t' \in \varepsilon'(P_{j'}^i)_m : \varepsilon(P_{j'}^i, t') = \varepsilon(P_k^i, t) \tag{B.4}$$

$$L_{ikt}^{p(n)} \geq L_{ikt-1}^{p(n)} \quad \forall m \in M, n, n' \in N_m : p(n) = p(n'), \\ i \in I, k = 1, \ldots, q_i, t \in \varepsilon'(P_k^i)_m \setminus \left(\varepsilon'(P_k^i)_m\right)_1 \tag{B.5}$$

$$E_{ij't'}^{n} \leq L_{ikt}^{p(n)} \quad \forall m \in M, n, n' \in N_m : p(n) = p(n'), i \in I, k = 1, \ldots, q_i, \\ j' \in q_i : P_{j'}^{i} \in \Im_i, t \in \varepsilon'(P_k^i)_m, t' \in \varepsilon'(P_{j'}^i)_m : \varepsilon(P_{j'}^i, t') = \varepsilon(P_k^i, t) \tag{B.6}$$

Constraints (B.1) include the binary variables ($E_{ij't}^{n}$) that are forced to take a value 1 when the imperfect task finishes a batch of units, i.e., when $W_{ij'r}^{n(r)}$ variables for an imperfect task are non-zero. Constraints (B.2) ensure that the variables $E_{ij't}^{n}$ take a value 1 only when the $W_{ij'r}^{n(r)}$ variables are non zero. Constraints (B.3) - (B.5) define that the actual binary variables L_{ikt}^{n} take a value 1 if and only if an imperfect task has already finished processing a batch of samples (i.e., uncertainty has already realized). Constraints (B.3) enforce that the binary variable L_{ikt}^{n} takes a value 1 at the first time point of a time period only if variable $E_{ij't'}^{n}$ is equal to 1. Constraints (B.4) define that variable L_{ikt}^{n} can take a value 1 only when either L_{ikt-1}^{n} or $E_{ij't'}^{n}$ is equal to 1. Constraints (B.5) ensure that once uncertainty is realized the value of L_{ikt-1}^{n} remains 1 for the rest of the time period. Constraints (B.6) define that variable L_{ikt-1}^{n} takes a value 1 whenever variable $E_{ij't'}^{n}$ is equal to 1.

When L_{ikt}^{n} is equal to zero, the nodes n and $n^{'}$ that share the same parent node remain indistinguishable and hence the corresponding decisions are equated through constraints (B.7) - (B.10).

$$W^n_{ikt} \leq W^{n'}_{ikt} + C_k R_k L^{p(n)}_{ikt} \quad \forall m \in M, n, n' \in N_m : p(n) = p(n'), i \in I, k = 1, \ldots, q_i, t \in \varepsilon'(P^i_k)_m \tag{B.7}$$

$$W^n_{ikt} \geq W^{n'}_{ikt} - C_k R_k L^{p(n)}_{ikt} \quad \forall m \in M, n, n' \in N_m : p(n) = p(n'), i \in I, k = 1, \ldots, q_i, t \in \varepsilon'(P^i_k)_m \tag{B.8}$$

$$V^n_{ikt} \leq V^{n'}_{ikt} + A_i L^{p(n)}_{ikt} \quad \forall m \in M, n, n' \in N_m : p(n) = p(n'), i \in I, k = 1, \ldots, q_i, t \in \varepsilon'(P^i_k)_m \tag{B.9}$$

$$V^n_{ikt} \geq V^{n'}_{ikt} - A_i L^{p(n)}_{ikt} \quad \forall m \in M, n, n' \in N_m : p(n) = p(n'), i \in I, k = 1, \ldots, q_i, t \in \varepsilon'(P^i_k)_m \tag{B.10}$$

Figure B.2: Transformation from STN to graph representation