

LUIS GONALO RODRIGUES REIS FIGUEIRA



Operational Production Planning and Scheduling in the Pulp and Paper Industry

Submitted to Faculdade de Engenharia da Universidade do Porto in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Industrial Engineering and Management, supervised by Bernardo Almada-Lobo, Associate Professor of Faculdade de Engenharia da Universidade do Porto

DEPARTMENT OF INDUSTRIAL ENGINEERING AND MANAGEMENT
FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO
2014

This research was partially supported by the PhD grant SFRH/BD/80109/2011 awarded by the Portuguese Foundation for Science and Technology and by the private company Europa&c Kraft Viana, S.A. under the project *Decision support system for the integrated planning of pulp, paper and energy*, supported by the National Strategic Reference Framework (NSRF).

“Things should be made as simple as possible, but not any simpler.”
Albert Einstein

Acknowledgments

First and foremost, I would like to thank my advisor Bernardo Almada-Lobo. He is the reason I have decided to enrol in a PhD project right after finishing my master's degree. The opportunity to work and learn from Bernardo has been extremely valuable to me. He can be highly supportive and motivating, but at the same time provides trust and the required freedom to grow and improve in different dimensions. I am thankful for his guidance, optimism, patience and above all for his friendship.

I would like to express a word of appreciation and gratitude to Maristela Santos, Reinaldo Morabito and Christian Almeder for having welcomed me in their universities and providing their valuable insights.

I am thankful to Europa&c Kraft Viana, S.A. for believing in the project, supporting it and invested time in multiple meetings at the plant. In particular, I would like to acknowledge Mário Amaral, for the opportunity to collaborate on an industrial problem that is so interesting, and Pedro Campos, Marco Moura, Balbina Fernandes, Raquel Simões and Ricardo Mendes, for their inputs and validations.

I owe also thanks to my colleagues and friends of FEUP/INESC, Brazil and Germany, for the pleasant atmosphere that has ever surrounded me. Their cheer was essential to my morale. Specially, I want to thank Marcos, Pedro, Mário and Luís, who have directly contributed to my work. For Luís a very indebted word of gratitude for helping me literally from the beginning to the end of this project.

Finally I would like to acknowledge the commitment of my family to my well-being and the values they have instilled on me. Thank you João and Francisco for helping me being who I am. Thank you Mãe and Pai, for your unconditional support.

Abstract

Production planning is especially important for capital intensive industries, where the high investments in capacity have to be diluted in the production of typically low profit margin products. Therefore the utilization of the installed productive capacity needs to be maximized, while the equipment is operated efficiently and the market requirements are satisfied.

The operational production planning is essential to adapt medium-term plans according to the actual state of the system at each time. This activity considerably depends on the specific industry and is particularly important in highly stochastic production environments. This thesis thus focus on the short-term planning in a particular industry, the pulp and paper (P&P), in order to grasp and incorporate the specificities of the production process and hence develop useful quantitative models and methods. These models need to be accurate enough to provide detailed and realistic plans.

The P&P industry is not only capital, but also energy intensive and the price of paper products is determined by the market, which may squeeze profit margins. In addition, the plants are subject to process variability and disturbances. Therefore, the operational planning is a critical activity. The planning task poses a variety of challenges, since the complex production flow is characterized by shifting bottlenecks, sequence-dependent set-ups and the variability inherent to the process.

The study conducted in the thesis is driven by a real-world case of a P&P producer, which not only provides valuable inputs regarding the challenges to be addressed, but also allows assessing the practical impact of the scientific contributions. The thesis explores both reactive and proactive planning approaches to the stochastic production system. On the reactive side, realistic models and efficient methods are proposed and a decision support system is implemented in the company. On the proactive perspective, the powerful simulation-optimization combination is studied in depth and a framework is devised for tackling production planning and scheduling problems.

Although the primary focus of the thesis is the P&P industry, other continuous process industries may also benefit from the contributions here provided.

Resumo

O planeamento da produção é especialmente importante em indústrias de capital intensivo, onde os altos investimentos em capacidade produtiva têm de ser diluídos na produção de produtos caracterizados por margens de lucro reduzidas. A utilização da capacidade produtiva instalada deve assim ser maximizada, enquanto o equipamento deverá ser manuseado de forma eficiente e os requisitos do mercado satisfeitos.

O planeamento operacional da produção é essencial para adaptar os planos de médio-prazo à realidade do sistema produtivo em cada momento. Esta actividade depende consideravelmente da indústria específica abordada e é particularmente importante em ambientes produtivos altamente estocásticos. Esta tese procura por isso focar-se no planeamento de curto-prazo numa indústria particular, a pasta e papel (P&P), de forma a compreender e incorporar as especificidades do processo produtivo e, assim, desenvolver modelos e métodos quantitativos de relevância e utilidade prática. É necessário que estes modelos sejam suficientemente fiéis à realidade para que disponibilizem planos detalhados e realistas.

A indústria de P&P é não só de capital intensivo, mas também de uso intensivo de energia, enquanto o preço do papel é determinado no mercado (levando por vezes ao esmagamento das margens de lucro). Para além disso, as fábricas apresentam variabilidade no processo e distúrbios. O planeamento operacional é então uma actividade crítica. Estas tarefas colocam uma série de desafios, visto o fluxo de produção ser caracterizado por gargalos dinâmicos, preparações do equipamento dependentes da sequência de produtos e a variabilidade inerente ao processo.

O estudo realizado nesta tese é motivado por um caso real de um produtor de P&P, que não só enriquece o trabalho com informação relativa aos desafios a abordar, mas também permite avaliar o impacto prático das contribuições científicas. A tese explora abordagens reactivas e proactivas ao planeamento da produção deste sistema estocástico. Na perspectiva reactiva, modelos realistas e métodos de solução eficiente são propostos e um sistema de apoio à decisão é implementado na empresa. Relativamente à abordagem proactiva, a poderosa combinação simulação-optimização é estudada em profundidade e um método para abordar o planeamento e escalonamento da produção é proposto.

Embora o foco principal seja a indústria de P&P, outras indústrias de processo contínuo poderão beneficiar das contribuições aqui propostas.

Contents

1	Motivation and framework	1
1.1	Paper industry and case study	2
1.1.1	Production process and specificities	2
1.1.2	Planning and scheduling	4
1.2	Research objectives	4
1.3	Thesis synopsis	7
	Bibliography	9
2	Solution approach for production planning and scheduling	11
2.1	Introduction	12
2.2	The case study	14
2.2.1	Production process and problem specificities	14
2.2.2	Short-term planning and scheduling	15
2.3	Hybrid VNS approach	17
2.3.1	Problem and notation	17
2.3.2	Solution representation and decoding	18
2.3.3	GVNS design	20
2.3.4	Neighbourhood structures	22
2.3.5	Constructive heuristic	23
2.3.6	<i>Speeds Constraint Heuristic (SCH)</i>	24
2.4	Computational tests and results	26
2.4.1	Tests design	27
2.4.2	Results	28
2.5	Conclusions	31
	Bibliography	32
2.A	Integrated P&P mathematical formulation	34
2.A.1	Pulp mill	34
2.A.2	Paper mill	36
2.A.3	Recovery plant	38
2.A.4	Objective function and overall model	39
3	Decision support system for production planning and scheduling	41
3.1	Introduction	41
3.2	The challenge	43
3.3	Solution approach	46
3.4	Mathematical model	48
3.4.1	Time slots	49
3.4.2	Production system	50
3.4.3	Demand fulfilment	52
3.4.4	Objective function	53

3.4.5	Valid inequalities	54
3.4.6	Practical constraints	54
3.5	Solution method	59
3.6	Decision support system	61
3.6.1	System architecture	61
3.6.2	System's usage	63
3.7	Results	66
3.7.1	Overall results of the system's usage	66
3.7.2	Optimized plan vs. manual plan – an example	67
3.7.3	Plan considering stoppages	68
3.8	Conclusions and future work	69
	Bibliography	70
4	Simulation-optimization taxonomy	77
4.1	Introduction	77
4.2	S-O methods	80
4.2.1	Solution Evaluation approaches	81
4.2.2	Solution Generation approaches	86
4.2.3	Analytical Model Enhancement approaches	88
4.2.4	Methods related to S-O	90
4.3	S-O taxonomy: a new unified framework	91
4.3.1	Interaction between simulation and optimization	91
4.3.2	Search design	93
4.3.3	Complete classification	95
4.4	S-O strategies discussion	95
4.4.1	Simulation purpose	96
4.4.2	Hierarchical structure	96
4.4.3	Search method	97
4.4.4	Search scheme	98
4.5	Conclusion and future lines of research	99
4.A	List of acronyms	101
	Bibliography	102
5	Simulation-optimization for proactive planning and scheduling	109
5.1	Introduction	109
5.2	Problem description	111
5.3	Mixed integer programming model	113
5.3.1	First stage (reactor) and intermediate tank	113
5.3.2	Second stage (machine)	114
5.3.3	Rates variation	115
5.3.4	Objective function	116
5.4	Solution method: VND-LP	116
5.5	Discrete-event simulation model	117
5.6	Hybridizing simulation and optimization	119

5.6.1	Simulation purpose	119
5.6.2	Refinement process	120
5.6.3	S-O structures	121
5.6.4	Variance reduction	121
5.7	Case study and preliminary experiments	122
5.8	Conclusions and further research	123
	Bibliography	125
6	Conclusions and future work	129

Motivation and framework

In increasingly competitive markets, managers need to actively look for opportunities to achieve operational efficiency and consumer satisfaction gains. Supply chain planning is thus an essential activity in the quest for competitive advantage. Production planning lies at the very heart of supply chain management and is specially important for capital intensive industries, where the high investments in capacity have to be diluted in the production of typically low-margin products. In that context a careful planning of production resources is crucial for maximizing capacity utilization, while operating efficiently and satisfying market requirements.

Production planning is performed at different hierarchical levels, which consider distinct planning horizons, scopes and levels of detail (Fleischmann et al., 2008). The longer the horizon, the greater the extent of the model (including more supply chain stages) and the lower the level of detail, not only because of model complexity issues, but also because uncertainty prevents a detailed planning for a long horizon.

In the short-term on the other hand, it is important to have more accurate models that provide detailed and realistic plans. Moreover, at this operational level, planning tasks vary considerably depending on particular industries, since the specificities of the production process become more relevant. Advanced Planning System (APS) providers, although increasingly aware of this situation, do not offer solutions with the required level of customization to accurately model the specific work flows of particular industries. As a result, managers typically rely on spreadsheets and their expertise to manually devise production plans. These procedures are time-consuming and sub-optimal.

In addition, uncertainty at the short-term level is a great challenge. In a disturbance situation, managers will be under more pressure and the available time to devise plans is further reduced. On the other hand, preventing disturbances is a complex subject which requires more elaborated techniques and becomes even more time-consuming. These reactive and proactive approaches thus need to be properly addressed.

This thesis tackles the operational production planning problem and focuses on a particular industry, the pulp and paper (P&P), in order to grasp and incorporate the specificities of the production process and hence develop useful models and methods. The research is driven by a case study from a Iberian P&P company, which not only provides valuable inputs regarding the challenges to be addressed, but also allows assessing the practical impact of the scientific contributions of the thesis. The first part of the thesis completes this cycle, starting with the case study motivation and ending with the final assessment in practice. The second part dives more into quantitative methods to deal with uncertainty and other complexities (in the context of proactive planning), having also its motivation in the case study, but focusing on a more general framework. Nevertheless, all chapters aim at

contributing not only to the P&P industry, but also to other similar (continuous) process industries, such as chemicals, industrial metals (e.g. iron, steel), oil refining and sewage treatment.

The remainder of this chapter is organized as follows. Section 1.1 provides some context of the P&P industry and the case study, specifically in what concerns the main challenges of the production planning and the current practice. Then, Section 1.2 presents the research questions and objectives related to those challenges and the methodology followed in the PhD project. Finally, the last section gives an overview of the thesis, by describing the chapters composing this document.

1.1. Paper industry and case study

The pulp and paper (P&P) industry is not only capital intensive, but also energy intensive. In addition, the price of paper products is determined by the market, which may squeeze profit margins. Therefore, companies need to maximize the utilization of their equipment, keeping production costs as low as possible, while trying to provide a good customer service. Effective production planning is thus imperative. However, the planning process poses a variety of challenges, both to practical and scientific fields. This section starts with a brief description of the production process and case study specificities. Then, the operational planning is depicted.

1.1.1 Production process and specificities

The P&P industry converts fibrous raw materials into pulp, paper and paper-board. In a first step raw materials are processed into pulp (in the digester) and in a second step different paper grades are produced out of this pulp (in the paper machine). These two steps can be performed on separate plants or combined into an integrated mill. Figure 1.1 illustrates the production process of an integrated P&P mill. It should be noted that at each stage there can be one or several resources in parallel and the equipment may work in continuous or batch production.

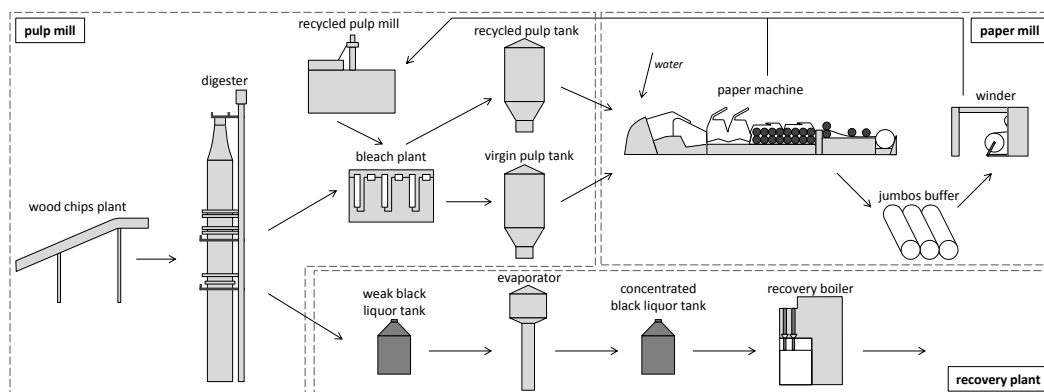


Figure 1.1: The integrated pulp, paper and recovery plant.

The quest for improvements in process efficiency, resource consumption, wastes production and energy balance has led companies to an integration of their processes and introduction of closed loops. Accordingly, integrated P&P mills represent 65% of the industry (CEPI, 2013) and are more capable of achieving high levels of both energy and economic efficiency, due to:

- energy conservation projects (e.g. direct use of steam in the paper drying process);
- absence of additional processes (e.g. pulp drying);
- material closed loops (e.g. recycling of paper machine setup loss and trim loss) which allow reducing waste production and the energy spent on it;
- tightly integrated equipment, which reduces the required capacity.

Nevertheless, an integrated P&P mill like that of our case study poses several challenges:

- important sequence-dependent setups, which break production stability and cause wear to equipment;
- variable production rates, constrained to maximum and minimum values, and with variation costs or limitations;
- tightly integrated production and storage capacity and distinct product recipes, which result in multiple and shifting bottlenecks;
- complex production system with serial, divergent, convergent and loop flows;
- process variability and disturbances, propagating through multiple stages.

The company of the case study produces two main variants of kraftliner (KLB and VLB), which is a type of paper for the corrugated cardboard boxes market. Both products (that essentially differ in their quantities of virgin and recycled fibres) may be produced in a set of different grammage values. For the sake of simplicity, hereafter the term “grade” is used for the combination of the thickness of the paper (related to its grammage) and the proportion of virgin and recycled fibres in the mixture. The company provides to its clients a total of 29 different grades.

All the production stages depicted in Figure 1.1 are performed on site, with the exception of bleaching, which is not necessary for this type of paper. In each stage there is only one production resource and they work in continuous. In addition to the aforementioned challenges, this paper company has to deal with multiple distribution channels, which include the schedule of containers for ship deliveries. All the production is made to order (MTO), which puts pressure on production to deliver the required amounts in the shortest possible lead-time.

1.1.2 Planning and scheduling

In order to minimize setups and maximize capacity utilization, orders are accepted based on a standard sequence of production which is repeated in cycles. In this way orders can be accepted well in advance. Looking at the available slots in the production sequence, managers commit to a due date (available-to-promise). At the operational level, the production sequence is adapted according to the evolving state of the system.

The operational production planning is typically a manual process which follows a hierarchical scheme:

- the top-level focuses on the paper machine, dealing with the sizing and scheduling of paper campaigns to fulfil current demand and backlog;
- the base-level tries to schedule and control all the other production resources, subject to the input from the top-level.

Each level relies only on spreadsheets and the inputs or feedback from the other level. Not only this process requires multiple iterations to produce a feasible plan, but it is also sub-optimal. Moreover, it does not allow for a careful trade-off between the different objectives, given the very limited time available at this operational level.

In addition, the system is subject to process variability (e.g. in production yield) and disturbances (e.g. equipment failures). Disturbances in one process tend to propagate to other processes, resulting in production losses, unnecessary rate changes (which further cause quality disturbances and undesired wear on equipment) and increase of the environmental load of the mill. Therefore, planners need also to consider slacks in their plans, in order to accommodate this uncertainty (proactive planning). Also, in the presence of disturbances a quick plan must be generated (reactive planning).

1.2. Research objectives

This research intends to give new insights into the resolution of the operational production planning problem in the P&P industry. As mentioned before, a case study is used to motivate and identify the main challenges of this problem and also to assess the performance of the models and methods proposed. Santos and Almada-Lobo (2012) had already approached this case study, by proposing an integration of the pulp, paper and chemical recovery processes, in order to consider the multiple bottlenecks of the system. The model however was too complex for exact solvers and even the approximate method proposed only achieved solutions of moderate quality. On the other hand, although used to generate and validate some plans, the model was not used by practitioners and therefore it did not address all the relevant industrial issues. Furthermore, it did not consider any uncertainty in the process. That work is thus the starting point of this thesis, which aims at contributing to both the literature and practice, in two main branches: reactive and proactive planning.

Branch 1 – Reactive planning

Reactive planning focuses on generating good quality plans in a short period of time. These methods can be used to face great disruptions, as well as situations which deviate only slightly from the original plan. As in both cases, especially in the former, the available time to devise a plan is very short, the models and methods assume all parameters to be deterministic. In this context two objectives were defined:

1. *Develop an efficient solution method for short-term planning and scheduling problems*

As the method proposed by Santos and Almada-Lobo (2012) could not generate good quality solutions in feasible time, the first objective is to develop an efficient solution method that can be used to tackle that problem. The method should be flexible enough to deal with specific issues of the problem, but also general so as to approach other lot-sizing and scheduling problems. Metaheuristics are general frameworks that can include specific heuristics or even exact methods. The former take advantage of the problem structure to increase the efficiency of the method, whereas the latter keep its generality. Therefore different components of the metaheuristic can be used to address distinct features of the problem, providing the flexibility in adapting to multiple problems. A metaheuristic is thus to be developed and its performance assessed. In particular, the following questions are to be answered:

- *What decisions should be solved optimally and what should be determined heuristically?*
- *How should campaigns be manipulated?*
- *How can discrete speed rates be addressed?*

2. *Develop a mathematical model to be used in practice, as well as an appropriate solution approach with the required pre- and post-optimization phases*

The efficient resolution of the aforementioned problem is an important matter, since many lot-sizing and scheduling problems will share similar features. However, to validate its suitability to the specific case study it has to be tested in practice. Also, the model needs to be extended, adapted or even reformulated, in order to deal with the industrial issues of a practical implementation. The second objective thus aims at formulating a mathematical model to be incorporated in a decision support system (DSS), which will help managers in their planning activities. The solution approach also needs to be revised (considering the new features) and upgraded with pre- and post-optimization steps. The DSS will be the interface used to test and assess the model and method in practice.

This objective includes the following research questions:

- *What industrial features have to be considered in a model to be used in practice?*

- *How should (continuous) production campaigns be intersected with (discrete) due dates?*
- *How can such a model be approached and incorporated in a DSS? How to implement the interfaces with users and existing systems?*

Branch 2 – Proactive planning

The second branch of this dissertation focuses on proactive approaches. Dealing with uncertainty is a complex issue, especially when the deterministic problem is already intractable. Therefore, it is imperative to devise intelligent ways of modelling and solving these complex systems. Simulation is not only able to easily address uncertainty, but also accurate representations of different types of uncertainty. For instance, stochastic programming allows assigning probability distribution to certain parameters. Simulation on the other hand can emulate disturbances (with specific starting and ending times) with no mathematical sophistication or complexity. Therefore, simulation-optimization (S-O) appears as a powerful approach, since it can combine the ease of simulation for dealing with complexity and the focus of optimization in finding good quality solutions. Two main objectives were thus defined:

1. *Study and classify the different ways of combining simulation and optimization and relate them to problem characteristics*

Simulation and optimization can be combined in numerous ways and the selection of the appropriate design can have a huge impact on the efficiency of the method. That choice will entirely depend on the characteristics of the problem to be tackled. Therefore, the first objective of this second research line is to study S-O methods and the way they relate to problem characteristics. Since the literature in S-O does not have a classification that covers the full spectrum of these methods, this thesis seeks also to propose a taxonomy with the most relevant classifying dimensions.

The main research questions associated with this objective are then the following:

- *What are the key dimensions in which S-O methods should be classified?*
- *What types of problems are best addressed by each S-O design?*

2. *Design and develop a simulation-optimization framework for production planning and scheduling problems under uncertainty*

After having an overview of the full spectrum of S-O methods, the objective is to develop an S-O framework that is suitable to production planning and scheduling problems. According to the general analysis performed in the previous objective, some S-O designs will be immediately disregarded, while other may need further attention.

The aim here is to explore the following questions:

- *What features should be kept in optimization and what could be transferred to simulation?*

- What method should be used to approach the optimization part?
- What type of simulation is most appropriate?
- How should the interaction be designed (what feedback and when should simulation provide to optimization)?

Figure 1.2 shows the overall structure of the thesis, framing the four main objectives in the two aforementioned research lines. It should be noted however that the approaches proposed for reactive planning can also address proactive issues and vice-versa. For instance, considering slacks in reactive models will help accommodating uncertainties. Also, an efficient metaheuristic addressing uncertainty may still be able to produce solutions of reasonable quality in feasible time (quickly reacting to a disruption). Nevertheless, a proactive approach will be focused on determining appropriate slacks, while a reactive method will be more efficient if it does not expend so much effort tackling uncertainty. Therefore, different methods should be used for reactive and proactive planning and both should be applied according to the state of the system.

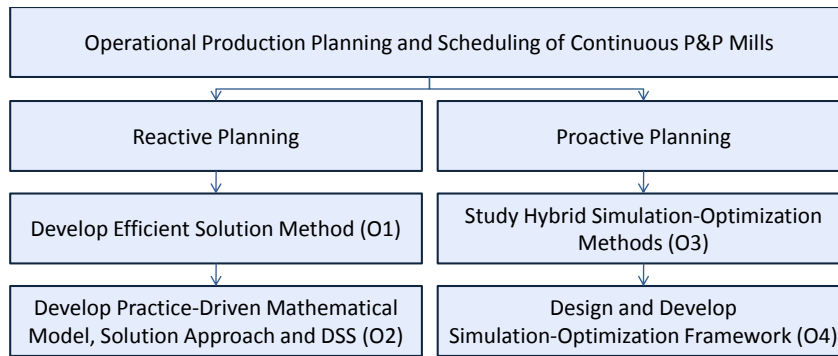


Figure 1.2: Research objectives.

1.3. Thesis synopsis

The chapters of this thesis consist of a collection of papers. Each paper (or chapter) is aligned with each of the research objectives previously defined. This section provides an overview of the main aspects covered by these articles.

Chapter 2 approaches the planning and scheduling that appears at the P&P industry with a hybrid approximate method. A variable neighbourhood search (VNS) is hybridized with an exact linear programming solver and a specific heuristic to deal with the binary variables related to the discrete digester's speed rates. The multiple neighbourhood structures were specifically designed to optimize lot-sizing and scheduling problems, including the discrete, continuous setup, proportional and general lot-sizing and scheduling problems (DLSP, CSLP, PLSP and GLSP, respectively). Those structures focus on the setup pattern, adding / removing / moving entire campaigns to / from certain positions of the schedule. Then, the exact solver and the specific heuristic are used to decode these patterns into final

plans, so that they can be properly assessed. While the heuristic is specific for cases where the speed rate is defined by a discrete grid (due to operational constraints or modelling issues), the solver is generic and allows for an easy incorporation of any linear constraint. Moreover, it fully optimizes continuous variables, which is important when there is still a degree of freedom after the binary variables have been fixed. The resulting method shows a very interesting performance on real world sized instances and seems to be promising for other lot-sizing and scheduling problems.

In Chapter 3 the mathematical model and solution method are revised to be incorporated in a DSS to be used in practice. The model is extended to deal with a variety of issues, such as different priority levels for orders, variable production rates in all units, scheduled stoppages and fixed campaigns. This last feature motivated also the reformulation of time slots, so that a feasible plan could always be generated. The continuous slots are able to cross the discrete periods used for demand fulfilment, which avoids having to (heuristically) determine the number of slots for each campaign. Like the previous model, it considers the sequence-dependent setup times and costs of the paper machine and the integration with the other stages. Therefore, an efficient solution method is even more necessary. However, the new formulation of time slots required including additional binary variables, which would compromise the performance of the VNS, since the decoding procedure would become drastically more complex. Therefore, a MIP-based heuristic was developed, which provides good enough solutions for practical use. The method comprises a main optimization phase and a post-optimization step. The former generates an initial solution and improves it, solving sub-MIP's, first in a forward pass and then with a biased selection. The post-optimization smooths production rates by fixing slots lengths. This separation is conducted to avoid non-linearity. Additionally, the chapter describes the interfaces of the DSS with the existing systems in the company and with the final users, as well as important pre- and post-processing steps. The system is used not only to generate optimized plans, but also to test and assess manually devised plans.

Chapters 4 and 5 then focus on proactive planning. The former conducts a general study of simulation-optimization methods, whereas the latter discusses an S-O framework for production planning and scheduling problems.

In Chapter 4 the full spectrum of S-O methods is reviewed, in order to provide an overview for a proper classification of these methods. The key classifying dimensions are identified. The purpose of simulation in the whole procedure appears to be the most relevant, since it distinguishes the main streams of research in S-O. The Simulation community typically uses simulation models to give feedback with respect to the quality of the solutions being simulated. The Optimization community tends to use simulation combined with analytical models, so as to take advantage of the problem structure, where simulation's output is used to enhance or complement the optimization model. Other key dimensions concern the hierarchical structure that relates simulation with optimization, the search method and the way the probability space is explored. Every category of each dimension is related to specific problem characteristics, thus providing insights regarding the types of problems that are best approached by each S-O design.

Chapter 5 starts from the previous general study and explores S-O methods in the context of a production planning and scheduling problem. The problem is inspired by the

case study and generalized keeping just the most relevant features. In particular, only two stages (and the tank in between) are considered. As previously discussed, the production system of the case study is subject to a variety of uncertainties, both in the form of process variability and as disturbances. To properly model those occurrences, discrete-event simulation is applied. The deterministic analytical model appears to be useful for the optimization of continuous variables, while a metaheuristic, based on that proposed in Chapter 2, is developed for intelligently manipulating the discrete variables. The combination of simulation and optimization is then discussed in its main dimensions, namely the purpose of simulation and the hierarchical structure.

Finally, Chapter 6 summarizes the most substantial results obtained with this thesis and gives directions for future research.

Bibliography

CEPI. Key statistics – european pulp and paper industry 2012. Technical report, June 2013.

Bernhard Fleischmann, Herbert Meyr, and Michael Wagner. Advanced planning. In Hartmut Stadtler and Christoph Kilger, editors, *Supply Chain Management and Advanced Planning*, pages 81–106. Springer Berlin Heidelberg, 2008.

Maristela Oliveira Santos and Bernardo Almada-Lobo. Integrated pulp and paper mill planning and scheduling. *Computers & Industrial Engineering*, 63(1):1–12, 2012. ISSN 0360-8352.

Solution approach for production planning and scheduling

A hybrid VNS approach for the short-term production planning and scheduling: A case study in the pulp and paper industry

Gonçalo Figueira* · Maristela Oliveira Santos[†] · Bernardo Almada-Lobo*

Published in *Computers & Operations Research*, 2013
<http://dx.doi.org/10.1016/j.cor.2013.01.015>

Abstract Mathematical formulations for production planning are increasing complexity, in order to improve their realism. In short-term planning, the desirable level of detail is particularly high. Exact solvers fail to generate good quality solutions for those complex models on medium and large-sized instances within feasible time. Motivated by a real-world case study in the pulp and paper industry, this paper provides an efficient solution method to tackle the short-term production planning and scheduling in an integrated mill. Decisions on the paper machine setup pattern and on the production rate of the pulp digester (which is constrained to a maximum variation) complicate the problem. The approach is built on top of a mixed integer programming (MIP) formulation derived from the multi-stage general lotsizing and scheduling problem. It combines a Variable Neighbourhood Search procedure which manages the setup-related variables, a specific heuristic to determine the digester's production speeds and an exact method to optimize the production and flow movement decisions. Different strategies are explored to speed-up the solution procedure and alternative variants of the algorithm are tested on instances based on real data from the case study. The algorithm is benchmarked against exact procedures.

Keywords Multi-stage lotsizing and scheduling · Production rates · Pulp and paper industry · Mixed integer programming · Variable Neighborhood Search · Hybrid methods

*INESC TEC, Faculdade de Engenharia, Universidade do Porto, Porto, Portugal

[†]Universidade de São Paulo - Instituto de Ciências Matemáticas e de Computação, São Carlos-SP, Brasil

2.1. Introduction

Research on the optimization of production planning has been more and more focused on increasing the detail and realism of mathematical formulations. [Meyr \(2002\)](#) pinpointed the need to integrate lotsizing and scheduling (LSS) in production environments with sequence-dependent setup times, since the final available capacity is only known after the definition of both the size and the sequence of lots. Different LSS models have been proposed, either big- or small-bucket (a classification given by [Eppen \(1987\)](#)). Since small-bucket models allow for only one changeover to be performed in each period, they always involve a larger number of periods when compared to big-bucket formulations. On the other hand, a higher level of detail can be incorporated into the former, as time periods are shorter. This is of particular importance in short-term planning.

Besides the integration of lotsizing and scheduling, researchers have been trying to extend traditional models, in order to include more specificities of the production environment ([Jans and Degraeve, 2007](#)). One of those extensions is the multi-stage scenario, where the interdependencies between resources in different stages are taken into account. The production flow may have different structures (serial, divergent, convergent, loop) or combinations of them. In any of these cases, the respective models are naturally larger and more complex than the corresponding single stage variants.

The general lotsizing and scheduling problem (GLSP), originally proposed by [Fleischmann and Meyr \(1997\)](#), attempts to reduce the size of models, considering micro-periods (or subperiods) of variable length embedded in macro-periods (or simply periods). However, the model's complexity considerably suffers in case the production rate of a given resource is a decision variable constrained to a maximum variation.

[Clark et al. \(2011\)](#) identified the process industries as a promising research area for lotsizing and its extensions, in spite of the current changes in the philosophy of production planning and control (e.g. lean manufacturing, shift from make-to-stock to make-to-order, etc.), due to their very specific features. This paper investigates a case study in the pulp and paper (P&P) industry, where the short-term production planning is tackled. The P&P industry converts fibrous raw materials into pulp, paper and paperboard and may produce energy (in a chemical recovery process) for internal use or to be sold to electrical companies. In a first step raw materials are processed into pulp (in the digester) and in a second step paper and paper products are produced out of this pulp, in different grade runs to be sized and scheduled on the paper machine ([Schumacher and Sathaye, 1999](#)). These two steps can be processed on separate plants (within the same company or belonging to different companies) or combined into a single mill. The latter can be formulated as a multi-stage lotsizing and scheduling problem (addressed here).

Research community tended to focus on a single production stage: either pulping (e.g. [Bredstrom et al. \(2004\)](#)), papermaking (e.g. [Murthy et al. \(1999\)](#) and [Rizk et al. \(2008\)](#)) or chemical recovery (e.g. [Axelsson et al. \(2006\)](#) and [Jonsson et al. \(2008\)](#)). As these approaches do not consider the interdependencies between the different stages, when applied to integrated P&P mills they may generate plans that are not only suboptimal, but also unrealistic, obliging managers to modify them manually to guarantee feasibility.

The work proposed by [Santos and Almada-Lobo \(2012\)](#) is indeed the only, to the best

of our knowledge, to integrate all the three main production stages, although it does not consider the cutting of the reels. This integrated production planning approach aims the synchronization of material flow as it moves through pulp, liquors and paper, responding to the need of integrating production decisions. The model describes a combination of all the aforementioned flow structures (serial, divergent, convergent and loop) and deals with the issue of production speeds constrained to maximum variations (in the digester). As the underlying problem is NP-hard, the authors implemented a stochastic MIP-based heuristic that provides feasible solutions. However, solutions are only of moderate quality. Thus, although the model can effectively reflect reality, useful plans may not be generated within feasible time.

In our paper we tackle the same problem using a different solution method. We propose a hybrid approach that combines approximate methods (heuristics and metaheuristics) with exact methods. In fact, metaheuristics are more tailored for combinatorial problems, whereas exact methods are relatively efficient in the optimization of continuous variables. Therefore, the latter are frequently used to decode (i.e. translate into a full production plan) a representation composed of only integer variables, which is managed by the metaheuristic.

Metaheuristics are high level frameworks that combine basic heuristics in order to efficiently and effectively explore the search space (Blum and Roli, 2003). One of the main issues that metaheuristics have to deal with is the entrapment in local optima. Variable Neighbourhood Search (VNS) is based on a systematic change of neighbourhoods, which may be performed in different ways, resulting in several variants (Hansen et al., 2008). The basic VNS combines a standard local search with a stochastic shaking phase, thus escaping from local optima while avoiding cycles. Variable Neighbourhood Descent (VND) is a deterministic best improvement descent method, where the local search is shifted to another neighbourhood structure if a local optimum is reached in the current one. Replacing in the VNS the local search step by the VND results in the General VNS (GVNS). The reduced VNS (RVNS) is a pure stochastic method where random points in the neighbourhood are picked and the incumbent solution is updated in case of an improvement.

Guimaraes et al. (2012) proposed a hybridization of an RVNS and an exact method to approach the tactical planning problem in the beverage industry. Other papers (such as Hindi (1996), Hung et al. (2003) and Almeder (2010)) applied this kind of hybridization with other metaheuristics (either neighbourhood- or population-based) to extensions of the capacitated lot sizing problem (CLSP), which is a big-bucket model. Almada-Lobo et al. (2008) has also approached an extension of the CLSP. The authors proposed a new VNS variant (combining GVNS and RVNS) for solving a production planning problem in the glass container industry. However, the literature is scarce for hybrid methods on small-bucket models. Meyr (2000) is one of the few to address a small-bucket formulation with a metaheuristic combined with an exact decoding procedure. Nevertheless, the author has not focused his work on the neighbourhood structures and on the constructive heuristic and the underlying problem is of single-stage and has fixed production rates.

Our algorithm considers the pattern of setups on the paper machine as the representation to be managed by the metaheuristic and decodes the remaining variables using an exact procedure and a specific heuristic to determine the discrete digester's speeds. Broad neigh-

bourhood structures are proposed in a GVNS design and different techniques to speed-up the expensive local search are employed. Tests performed on generated instances based on real data attested the superiority of the algorithm over a state-of-the-art commercial solver for large sized instances.

The main contributions of our work are as follows. We propose here a hybrid solution method that is the first to deliver good quality solutions within feasible time for a real-world problem in the P&P industry. Both pure exact methods and MIP-based heuristics provide only feasible solutions of low or moderate quality. We also expect to give insights on the efficient resolution of other LSS problems, where the integer part consists of a setup pattern and/or a grid of production speeds (constrained to a maximum variation).

The remainder of the paper is as follows. In Section 2.2 the case study is discussed (including the general production and planning processes and the characteristics of the specific problem). Section 5.4 is dedicated to the solution method proposed for this problem. Computational experiments are given in Section 2.4. The paper ends with some conclusions and directions for further research.

2.2. The case study

2.2.1 Production process and problem specificities

The workflow of the integrated production system consists of seven interdependent sub-processes: wood preparation, pulping, paper recycling, bleaching, chemical recovery, papermaking and cutting.

In the first stage the wood is debarked and reduced to small chips, which are later cooked in one or several (batch or continuous) digesters in aqueous solutions with reagents at high temperature and pressure (chemical pulping), producing virgin pulp. Different types of pulp are characterized by different types of wood and incorporation ratios of recycled materials or even of non-wood plant sources.

In the pulping process a by-product is produced: the weak black liquor. From this point onwards, both products go through various transformation processes separately from each other (divergent flow). The weak black liquor goes to an intermediate tank before being concentrated in an evaporator. After this stage, the concentrated black liquor flows through another intermediate buffer and then to a capacitated recovery boiler, where it is burnt, providing high-pressure steam and regenerating the spent chemicals applied in pulping. The steam can either be used for the paper drying process or be led to counter-pressure turbines which produce electrical energy to be sold afterwards.

The virgin and recycled pulps may go through a bleaching phase and then stocked in their respective tanks, waiting for being pulled by the paper machine(s). From the tanks, these primary pulps are diluted and fed together into the paper machine (convergent flow), where the paper is formed and the water is removed. The dried paper, characterized by its grammage (measured in g/m^2) and type of pulp, is wound into a master reel (called jumbo). The configuration of the machine to produce a different type of paper (different grammage or pulp mixture) is sequence-dependent. Each setup leads to a loss in the production process in terms of time and quantity of a lower quality paper produced (as the

machine is never idle, even during the switchover). The wasted paper is fed again into the recycled pulp mill (loop flow).

Finally, the jumbo follows to the winder where it is cut into smaller reels. The wasted paper in the cutting process is also fed into the recycled pulp mill. Customers may place orders for reels of different widths and paper types. The reels can also be cut into sheets of different sizes, which are wrapped into reams. Converted paper products, such as paper-board boxes, may also be produced.

Figure 5.1 shows a simplified scheme of the manufacturing process in a P&P mill. It should be noted that at each stage there can be one or several resources in parallel. This figure depicts the main decision variables of the mathematical model of the respective production planning and scheduling problem (presented in 2.A).

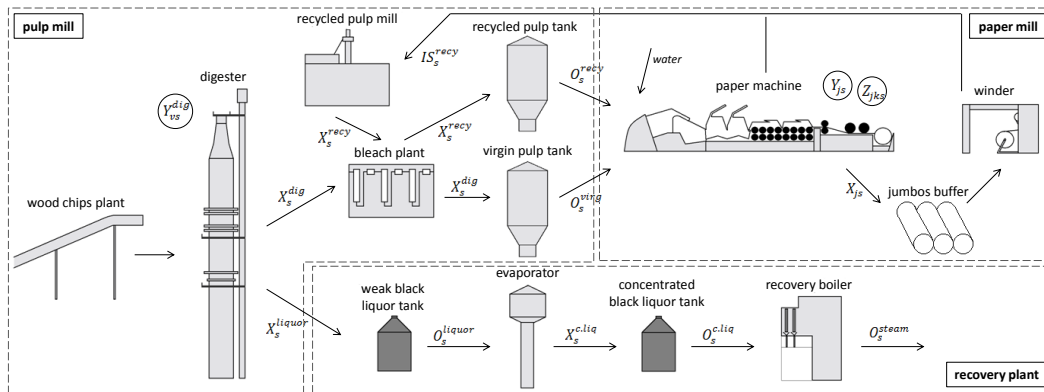


Figure 2.1: The integrated pulp, paper and recovery plant.

The company of this case study produces two main variants of kraftliner (KLB and VLB), which is a type of paper for the corrugated cardboard boxes market. Both products (that essentially differ in their quantities of virgin and recycled fibres) may be produced in a set of different grammage values. For the sake of simplicity, hereafter the term *grade* is used for the combination of the thickness of the paper (related to its grammage) and the proportion of virgin and recycled fibres in the mixture. The company provides to its clients a total of 29 different grades.

All the aforementioned production stages are performed on site, with the exception of bleaching, which is not necessary for this type of paper (note that in Figure 5.1, the input and output flow variables in the bleaching stage are the same), and sheet cutting, as customers place orders for reels. In each stage there is only one production resource (e.g. one continuous digester, one paper machine, etc.). Between the papermaking and the cutting stage, there is a buffer that makes it possible the production planning to consider aggregated demand for jumbos, instead of demand for final items.

2.2.2 Short-term planning and scheduling

The P&P industry is characterized on the one hand by high capital and energy intensity and, on the other hand, by a relatively high introduction rate of new products. As a result, the

definition of the desirable efficient production sequences (remark that setups are sequence-dependent) and the work-load balance between different resources are frequently perturbed by the accommodation of new products in the production system. Moreover, the expansion of the product range demands for an increased flexibility, which has to be conjugated with high productivity levels, that are crucial in peak demand periods. The objective function of the model in appendix thus weights inventory, backloging and setup costs, as well as the productivity of each resource.

This trade-off between lead-times and productivity is especially difficult when demand is subject to rapid change (as it is in our case). In such cases, the tactical planning level seeks to derive a periodic scheduling, in which a production cycle (e.g. one week) is repeated over time. This periodic scheme facilitates the acceptance of customer orders well in advance. Then, in the operational level the sequence of paper grades (defined for the cycle) is adapted according to the evolving state of the system. The short-term problem is the focus of our work. For tactical approaches, see [Bouchriha et al. \(2007\)](#) and [Castro et al. \(2009\)](#).

In most companies, the short-term production planning process is completely manual and typically follows a hierarchical scheme. The top-level focuses on the paper machine, dealing with the sizing and scheduling of paper runs to fulfil current demand and backlog. The paper sequence may come from the tactical level and generally consists of an ascending (or descending) order of grades, followed by a descending (or ascending) order, hence minimizing setups. Some grades may be repeated in the same cycle, due to market issues (concerning customer order due dates and backlog) and production issues (temporary bottlenecks and virgin/recycled pulp tank levels). The base-level of the hierarchical planning tries to schedule and control all the other production resources, subject to the input from the top-level.

This industry-practice planning seeks the maximization of the mill's throughput, focusing mainly on the paper machine (typically the bottleneck) and minimizing the time lost in setups of that resource. However, backloging and inventory costs are not properly weighted. Also, the complexity of the production flow (including divergent, convergent and loop structures) makes the base-level planning incredibly hard to optimize. Moreover, various iterations between the two levels may be needed to find a reasonable solution, since for example, a lack of capacity of the recovery boiler may force the digester to reduce its speed, not allowing it to provide the required pulp for a given papermaking schedule. In addition, as the pulp quality strongly depends on the digester's stability, its synchronization with other resources, such as the paper machine, is crucial. In fact, different paper grades produced on the paper machine pull more or less pulp from the digester (forcing the change of its speed) according to their respective grammage. Therefore, the close interrelation between these two stages must be carefully managed.

Facing several constraints, conflicting objectives and eventually multiple and shifting bottlenecks, planners are led to place feasibility over optimality. The resulting suboptimal plans may not only compromise production costs and company's efficiency, but also the service level provided to its clients. Optimization models focusing on a comprehensive P&P production planning are thus imperative, as well as efficient solution methods capable of solving these complex and realistic problems within feasible time. The latter are the

focus of this paper.

2.3. Hybrid VNS approach

2.3.1 Problem and notation

The model proposed by Santos and Almada-Lobo (2012) is based on the GLSP with sequence-dependent setup times and costs and extended to the multi-stage scenario with additional constraints. The length of each subperiod s is a continuous decision variable denoted by N_s (in this case expressed in hours) and the time grid to be defined (i.e. number of non-empty micro-periods) is common for all the resources and production stages (similar modelling techniques are used by Camargo et al. (2012)).

One of the main issues of this problem is that the digester's rotation speed (V_s^{dig}) is a decision variable that is constrained to a maximum variation ($\Delta \cdot \Phi$) in each subperiod (to ensure the stability and smoothness of the cooking process) and hence, it has to be represented explicitly in the model. Therefore, in order to avoid a non-linear formulation, a grid of 15 discrete values for the digester's speed (sp_v) had to be defined (in steps of 0.25 rpm), with a binary variable ($Y_{v,s}^{dig}$) associated to each of them. The other binary variables of the model are related to paper machine's setups and changeovers (Y_{js} and Z_{jks} , respectively, which are mutually dependent). All the remaining variables are continuous and basically consist of the flow variables represented in Figure 5.1 and the inventory levels throughout the process. Although backlog is allowed, it should not build up for a long period of time and hence, in each cycle, the initial backlog of each grade has to be fulfilled until the end of that cycle (backlog coverage). This requirement is common in practice.

A refined version of Santos and Almada-Lobo's model is presented in 2.A, with additional constraints and objectives that better represent company's reality and with modifications that turn the model more efficient for exact methods. Throughout this section, the notation used for variables is the same as that of the model. Additionally, a complete solution is denoted by x and each of its decision variables dv can be assessed as a function $dv(x)$.

In order to quickly understand the key features of the model, a production plan was generated for a small instance, with 8 products (aggregated into 7 grades), 5 periods (totalizing a cycle in this case) and 3 subperiods per period. The model was solved to optimality by a commercial solver. Figure 2.2 illustrates the plans generated for both the digester and paper machine, as well as their synchronization through the common time grid (which is also shared by the other production resources). The vertical dashed lines divide the subperiods, whereas the continuous lines represent the digester's speed (expressed in rotations per minute) and its maximum value.

Although small instances of this (NP-hard) problem can be solved to optimality by commercial solvers in reasonable time, real-world sized instances clearly need more efficient methods (see Section 2.4 for computational times and gaps). For that purpose, an approximate algorithm was developed.

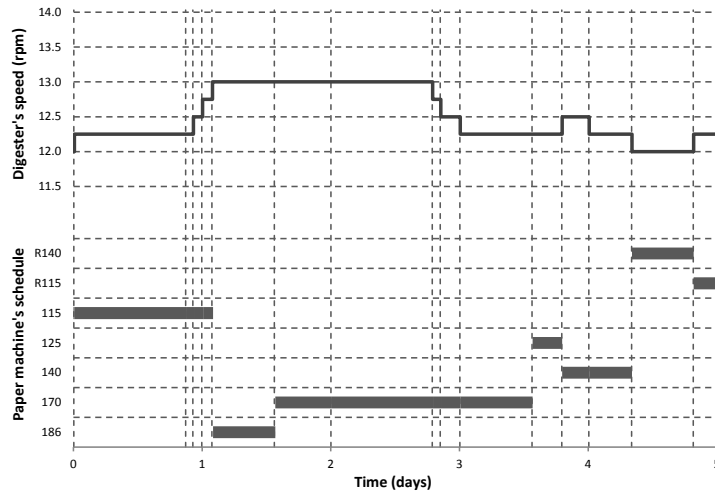


Figure 2.2: Partial production plan: synchronization between the digester and the paper machine.

2.3.2 Solution representation and decoding

The algorithm proposed in this paper combines heuristic and exact methods in a GVNS design (see Algorithm 11 in Subsection 2.3.3 for the high level procedure). Each solution is represented by the setup pattern (encoding), i.e. the number of subperiods assigned to each paper grade and their sequence (which is the information given by the setup variables Y_{js}) – see Figure 2.4 for graphical representations. The GVNS iterates through different patterns and uses a decoding procedure to translate each pattern into a final production plan (as that represented in Figure 2.2, with subperiods of different length and production decisions related to the other resources). Figure 2.3 illustrates this general algorithm structure. The decoding procedure is a combination of an exact method and a specific heuristic (the *Speeds Constraint Heuristic* – SCH). The exact method optimizes the continuous variables, so that good quality solutions are not incorrectly rejected and the algorithm keeps its simplicity and flexibility to cope with further model extensions and modifications. The heuristic is used to determine the digester's speeds. The complete decoding is presented in Algorithm 12.

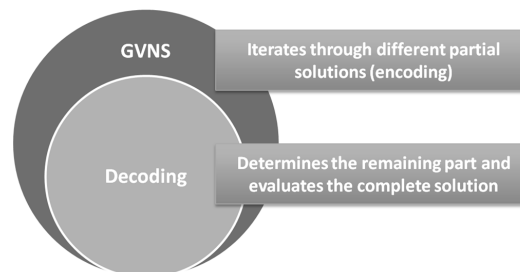


Figure 2.3: General structure of the hybrid algorithm.

The Decode function decodes gradually the solution while the objective value is better than that of the incumbent (f^{inc}), in order to save computational time. This is particularly important in local search, where the number of neighbours to evaluate can be very large. If the solution is worse than the incumbent, it is rejected by considering it as infeasible. In the shaking procedure of the GVNS, however, as the solution can deteriorate, an infinity is assigned to f^{inc} so that worse solutions are not rejected.

Decode starts by calling an exact solver to optimize the linear programming (LP) problem that results from fixing the setup pattern Y and relaxing the speed variation constraints (line 1). This relaxation is achieved by removing (2.3)–(2.6) and replacing the digester's production output constraints (2.9) by the following:

$$X_s^{dig} \leq \alpha \cdot v_{max}^{dig} \cdot N_s \quad s \in [S]. \quad (2.1)$$

$$X_s^{dig} \geq \alpha \cdot v_{min}^{dig} \cdot N_s \quad s \in [S]. \quad (2.2)$$

The new constraints just impose upper and lower bounds on the digester's output (and therefore to its speeds), leaving the choice of the speed as a continuous decision variable. In fact, the speed is now an implicit variable defined by $V_s^{dig} = X_s^{dig} / (\alpha \cdot N_s)$.

To ensure that the digester's speeds satisfy the maximum variation constraints (2.4), the SCH is called to check, correct and fix the speeds (line 3). In SCH (detailed in Section 2.3.6), the LP is solved again to update the values of the continuous variables, according to the modified speeds. If the solution returns as infeasible, the solver is called to definitely validate the setup pattern (solving the fixed MIP with the original constraints until a first solution is found – line 5). The solver is drastically more demanding in terms of computational time (over 100 times, according to some preliminary testing) and the SCH enables a quick validation of the feasibility of the majority of the neighbour solutions.

Algorithm 1: Decode(Y, f^{inc})

```

1  $x' \leftarrow \text{Solver}(Y, \text{RelaxSpeeds}, \text{Optimality})$ 
2 if  $f(x') < f^{inc}$  then
3    $x'' \leftarrow \text{SCH}(x')$ 
4   if  $x''$  is infeasible then
5      $x'' \leftarrow \text{Solver}(Y, \text{IncludeSpeeds}, \text{FirstSol})$ 
6   if  $x''$  is feasible and  $f(x') < f^{inc}$  then
7      $x \leftarrow x'$ 
8   else
9      $x \leftarrow inf$ 
10 else
11    $x \leftarrow inf$ 
12 return  $x$ 

```

The gradual decoding of solutions is reflected not only in the number of steps performed in Algorithm 12, but also in the first LP solving procedure. In fact, in that procedure a speed-up strategy called *Dual Re-optimization* is used. This technique interrupts the LP optimization procedure (which is solved with the dual algorithm), in case the solution

gets worse than the incumbent (Meyr, 2002; Guimaraes et al., 2012).

2.3.3 GVNS design

The GVNS design is presented in Algorithm 11. A solution is initialized by `ConstructiveHeuristic` and then a series of `Shake` and `VND` executions try to improve this solution by exploring different regions of the space. In the shaking phase, the search shifts to the next neighbourhood if an improvement is not achieved (line 10) and it returns to the first one, otherwise (line 8). Neighbourhoods are sequenced in increasing order of the strength of the corresponding moves. In this way, small perturbations are applied after a local optimum is obtained, so that its close region is well explored. Then, the search diversifies more to explore different regions. The algorithm ends after the last neighbourhood has been tried in shaking. A time limit is also imposed (and checked throughout the algorithm). The final setup pattern is exactly decoded, running the solver until optimality with the Y variables fixed (line 11).

Algorithm 2: $GVNS(k_{max}, k'_{max})$

```

1  $x \leftarrow \text{ConstructiveHeuristic}()$ 
2  $k \leftarrow 1$ 
3 while  $k \leq k_{max}$  do
4    $x' \leftarrow \text{Shake}(x, k)$ 
5    $x'' \leftarrow \text{VND}(x', k'_{max})$ 
6   if  $f(x'') < f(x)$  then
7      $x \leftarrow x''$ 
8      $k \leftarrow 1$ 
9   else
10     $k \leftarrow k + 1$ 
11  $x \leftarrow \text{Solver}(Y(x), \text{IncludeSpeeds}, \text{Optimality})$ 

```

In the search performed by the GVNS, critical constraints are relaxed and their violation is penalized in the evaluation function, so that the `ConstructiveHeuristic` is always able to generate an initial solution. The `VND` is then expected to correct this violation, which is related to backlog covering (constraints (2.22)) and the virgin pulp and black liquors tank levels (constraints (2.15), (2.31) and (2.35)).

The neighbourhood structures to be used in `VND` are very broad in this case (as we will see in the next subsection). Moreover, as the evaluation of each neighbour requires calling the expensive `Decode` function, the `VND` design (see Algorithm 22) is focused on speeding-up the neighbourhood search.

The neighbourhood search is applied to each production cycle, one at a time (the verification in lines 6-7 allows not to repeat cycles already explored since the last move). Also, when an improvement move is found, it is immediately selected and implemented (first-improvement strategy), avoiding an exhaustive exploration of neighbourhoods. In addition, to find improving solutions more quickly, the neighbours are ranked according to their likelihood of improving the incumbent solution (line 8). Since the last ranked moves are not so likely to improve, it is probably not worth (and is even impractical on real-sized problem

instances) to explore the entire neighbourhoods. Thus, a given parameter (ps_i) determines the percentage of the size of each neighbourhood i to be explored. It may also balance the different neighbourhood reduced sizes. The change of neighbourhoods is performed in a cyclic way: the search continues in the same neighbourhood while it finds improving solutions and moves to the following otherwise (except when the last neighbourhood is explored – see lines 15-21). The VND is executed until the number of non-improving consecutive iterations (k'') reaches the number of neighbourhoods (k'_{max}).

Algorithm 3: VND(x', k'_{max})

```

1  $k'' \leftarrow 0$ 
2 while  $k'' < k'_{max}$  do
3    $k \leftarrow 1$ 
4    $x'' \leftarrow x'$ 
5   for  $c \leftarrow 1$  to  $\lceil C \rceil$  do
6     if  $c = c' + 1$  and  $x' = x''$  then
7       break
8     Sort( $N_k^c(Y(x'))$ )
9     while  $f(x) \geq f(x')$  and  $k \leq \lceil ps_{k'} \cdot |N_k^c(Y(x'))| \rceil$  do
10       $x \leftarrow \text{Decode}(Y, f(x'))$ ,  $Y \in N_k^c(Y(x'))$ 
11       $k \leftarrow k + 1$ 
12      if  $f(x) < f(x')$  then
13         $x' \leftarrow x$ 
14         $c' \leftarrow c$ 
15      if  $f(x) < f(x'')$  then
16         $k'' \leftarrow 0$ 
17      else
18         $k'' \leftarrow k'' + 1$ 
19         $k' \leftarrow k' + 1$ 
20        if  $k' > k'_{max}$  then
21           $k' \leftarrow 1$ 
22 return  $x'$ 

```

The neighbourhood structures consider compound moves, i.e., each move is composed of different partial moves. The ranking procedure is then performed for each of these partial moves. The exact impact on the changeovers cost is evaluated and the changes in inventory and backlogging are estimated, thus providing an indicator of the quality of the move. In order to avoid premature convergence, a stochastic selection is applied, choosing randomly in each iteration one of the $pr\%$ best ranked moves to be evaluated. The deterministic variant is also tested.

Finally, an early rejection strategy is implemented in order to further reduce computational time. The strategy consists on examining moves feasibility regarding backlog covering constraints. In fact, if the only production run of a grade with backlog to fulfil in a given cycle is removed (and no other run of that grade is inserted), the move will result in an infeasible solution and thus, it can be rejected before executing the Decode procedure.

The following subsections explore in more detail the different ingredients of the algorithm (neighbourhood structures, ConstructiveHeuristic and SCH).

2.3.4 Neighbourhood structures

A neighbour of an incumbent solution x is obtained by slightly changing the setup pattern and decoding it (see line 10 of Algorithm 22). We have defined four different types of neighbourhood structures, which share a common feature: the total number of subperiods keeps constant, which is important when the LP model is solved. Figure 2.4 illustrates the neighbourhood structures, each with an example of a move applied to a given setup pattern. It is important to note that time is discretized in subperiods, which after the decoding procedure may have different lengths.

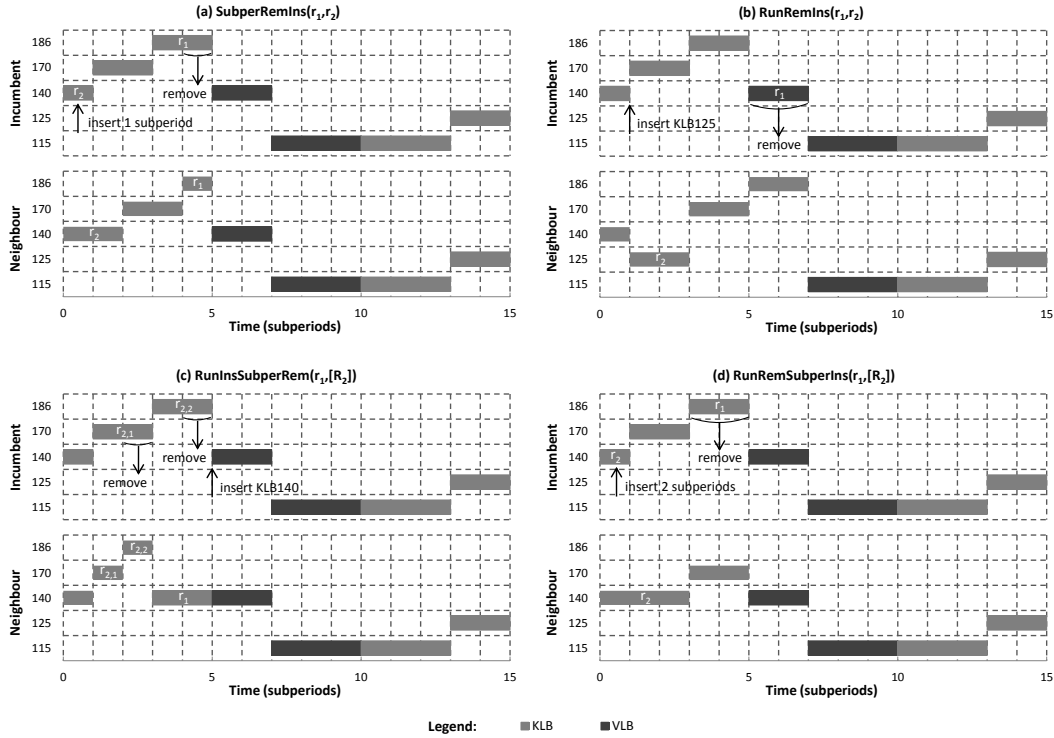


Figure 2.4: Illustration of the neighbourhood structures.

Given a set of production runs $r = 1, \dots, R$ and a set of grades $j = 1, \dots, K$, let $j(r)$ denote the grade being produced on run r , $s(r)$ be its number of subperiods and $o(r)$ its position in the sequence. The definition of the neighbourhood structures then follow:

- *SubperRemIns*(r_1, r_2) (Figure 2.4(a)) consists on removing one subperiod from a run r_1 (in which $s(r_1) > 1$, so that the run itself is not removed) and inserting another one into a run r_2 , such that $r_2 \neq r_1$.
- *RunRemIns*(r_1, r_2) (Figure 2.4(b)) removes a complete production run r_1 and inserts another run r_2 , such that $s(r_2) = s(r_1) \wedge (j(r_2) \neq j(r_1) \vee o(r_2) \neq o(r_1))$.
- *RunInsSubperRem*($r_1, [R_2]$) (Figure 2.4(c)) inserts a production run r_1 and consecutively removes single subperiods (in a total of $s(r_1)$) from a set $[R_2]$ of any runs

$r_2 \in [R] \cup \{r_1\}$, such that $s(r_2) > 1$.

- *RunRemSubperIns*($r_1, [R_2]$) (Figure 2.4(d)) removes a production run r_1 and consecutively inserts single subperiods (in a total of $s(r_1)$) into a set $[R_2]$ of any runs $r_2 \in [R] \cup \{r_1\}$, such that $\exists r_2 \neq r_1$.

The first neighbourhood (based on *SubperRemIns*(r_1, r_2)) is the smoothest and only explores the distribution of production subperiods among the existing runs. The other three are very broad in the sense that they extend traditional neighbourhoods (such as transfer, splitting and aggregation of runs).

Some preliminary tests were performed in order to select the neighbourhood structures for the local search phase, as well as their sequence. The results led to the selection of the last three, in the order they were presented here.

For the shaking phase, six derived neighbourhoods are used in the following order: three neighbourhoods based on *SubperRemIns*(r_1, r_2), consisting of one, two and three random moves of that type, respectively; three other based on *RunRemIns*(r_1, r_2), consisting of one, two and three random moves, respectively.

2.3.5 Constructive heuristic

Finding an initial solution for this problem consists on determining a setup pattern for the paper machine, which is then decoded by Decode function (as seen in Subsection 2.3.2). The constructive heuristic proposed here can be divided in two main steps: sequencing grades and assigning a number of production subperiods to each grade. To make it simple, only one run is considered per grade and per cycle, at this stage. The procedure is separately executed for each cycle and decodes in each iteration the partial solution, composed of all the considered setup variables (whose integrality was relaxed in the beginning of the algorithm). Algorithm 18 exhibits the main procedure.

In each cycle, sequencing is executed by consecutively choosing, among the grades with demand or backlog to fulfil in that cycle, the one (within the same product type) of the following grade either in ascending or descending order. For instance, grades may start increasing the grammage value within a given product type (e.g. KLB). Then, a changeover to another type (VLB) is performed, starting at the highest value (for the shift to be smooth) and decreasing until the lowest. Finally, it returns to the original type (KLB), if there are any grades left, starting at the lowest and continuing in ascending order. In this way, the amount of time lost in setups is relatively small. The first setup pattern in Figure 2.4(a) is an illustrative example of a sequence of paper runs starting in ascending order. The procedure can be executed in the opposite way (starting with decreasing grades), which will lead to the mirrored sequence. Both sequences (defined by $order_l$, $l = \{1, 2\}$) are generated (see line 5) and evaluated in the *ConstructiveHeuristic* and the one that provides the best results is selected (lines 16-17). As this sequencing procedure is deterministic and may lead towards premature convergence, a stochastic variant is also implemented.

For each of the generated sequences, the assignment of subperiods to paper grades has to be determined. The heuristic starts assigning an amount of subperiods that guarantees

the backlog coverage of the corresponding cycle (lines 6-7). From that assignment, three different cases may occur:

- The number of assigned subperiods is the same as the total available to schedule and thus, the procedure has finished;
- The assigned subperiods are more than those available and consequently some subperiods have to be removed from production runs;
- There are still subperiods to be assigned, requiring a procedure to determine the additional assignment.

The assignment of subperiods in the first step (to fulfil backlog) is performed assuming that all superperiods have equal lengths. Therefore, the removal of subperiods in the second case does not necessarily result in an infeasible solution, since the slack of some paper grades can be used by other grades adjusting the length of the subperiods. Hence, an iterative procedure is executed, where in each iteration a subperiod is removed from the grade with the greatest slack (line 9).

When the number of assigned subperiods is less than the total available (third case), a utility measure u_j is used to determine the portion of the remaining subperiods to assign to each grade (line 12). The greater the production needs (considering demand, backlog and inventory on the one hand, and the already assigned subperiods on the other hand) and the faster the production rate of a given grade are, the higher the utility will be. However, the number of subperiods to assign is constrained by the actual production requirements S_{addReq_j} . Then, if there are still subperiods to be assigned, an iterative procedure is used, choosing in each iteration the grade with the greatest production needs.

2.3.6 Speeds Constraint Heuristic (SCH)

The *Speeds Constraint Heuristic* has to check, correct and fix the speed values that result from the optimization of the first LP (in which the speed variables and constraints were removed) – see Algorithm 12. The SCH iterates through subperiods, comparing the digester's speed to that of the previous subperiod. The main steps are presented in Algorithm 13.

In order to get more control over the behaviour of the speed pattern, the (guiding) objective function is slightly modified. The virgin pulp production output of early subperiods is favoured over the output of those that follow. Therefore, at first only the speed shift upper limit may be violated and then, when one of the downstream tanks (either the virgin pulp or the weak black liquor) gets full, the speed may be abruptly reduced.

The correction of the upper limit violation becomes trivial: it is just necessary to reduce the speed value to the previous speed incremented by the maximum variation (see line 4). The same is not true for the lower limit, since increasing the speed value will necessarily result in overflow of the tank levels (as the digester's production was being maximized), unless the speed has been reduced in a previous subperiod. In case of overflow, the speeds of previous subperiods have also to be reduced. Thus, after a set of consecutive subperiods

Algorithm 4: ConstructiveHeuristic()

```

1  $x' \leftarrow inf$ 
2 for  $l \leftarrow 1$  to 2 do
3   Relax integrality of  $Y$ 
4   for  $c \leftarrow 1$  to  $\lceil C \rceil$  do
5     Sequencing( $c, order_l$ )
6      $S_{assignCalc_j} \leftarrow average\left\{\frac{|S_t|}{cap_t} \mid t \in [T_c]\right\} \cdot (p_j \cdot IG_{j,t_{1c}-1}^-(x) + st_{kj})$ , where  $k, j \in [K]$  and  $k$  is
       before  $j$  in the sequence
7      $S_{assign_j} \leftarrow \lceil S_{assignCalc_j} \rceil, j \in [K]$ 
8     while  $\sum_j S_{assign_j} > |S_{cycle_c}|$  do
9        $k \leftarrow argmax\{S_{assign_j} - S_{assignCalc_j} \mid j \in [K]\}$ 
10       $S_{assign_k} \leftarrow S_{assign_k} - 1$ 
11     if  $\sum_j S_{assign_j} < |S_{cycle_c}|$  then
12        $S_{assign'_j} \leftarrow min\left\{\left[u_j \cdot (|S_{cycle_c}| - \sum_j S_{assign_j})\right], S_{addReq_j}\right\}, j \in [K]$ 
13       Iteratively assign the remaining subperiods according to production needs
14     Determine  $Y$  according to the defined sequence and subperiods assignment
15      $x \leftarrow Decode(Y, f(x'))$ 
16     if  $f(x) < f(x')$  then
17        $x' \leftarrow x$ 
18 return  $x'$ 

```

$\{s_i, \dots, s_f\}$ where the speed reduction has been violated, the iterative procedure is interrupted and the previous speeds are reduced until all the cumulative overflow has been eliminated (line 9). The iteration then restarts after all the assessed speeds have been rounded down to the next value of the grid, in order to guarantee discrete values (which is important for the results to be comparable to those obtained when solving the complete MIP exactly). At the end, the speeds are fixed (in addition to the setup pattern) and the solver is called to update the continuous variables (in an LP).

Algorithm 5: SCH(x)

```

1 while  $\exists s \in S : |V_s^{dig} - V_{s-1}^{dig}| > \Delta \cdot \Phi$  do
2   for  $s \leftarrow 1$  to  $S$  do
3     if  $V_s^{dig} > V_{s-1}^{dig} + \Delta \cdot \Phi$  and there is no speed reduction to correct then
4        $V_s^{dig} \leftarrow V_{s-1}^{dig} + \Delta \cdot \Phi$ 
5     else if  $V_s^{dig} < V_{s-1}^{dig} - \Delta \cdot \Phi$  then
6        $V_s^{dig} \leftarrow V_{s-1}^{dig} - \Delta \cdot \Phi$ 
7       Update  $s_i$  and  $s_f$ 
8     if there is any speed reduction to correct and (the speed reduction in  $s$  does not need correction or  $s = S$ ) then
9       LayerRemoval( $s_i, s_f$ )
10      break
11   Round down all the assessed speeds
12  $x' \leftarrow Solver(Y(x), FixSpeeds, Optimality)$ 
13 return  $x'$ 

```

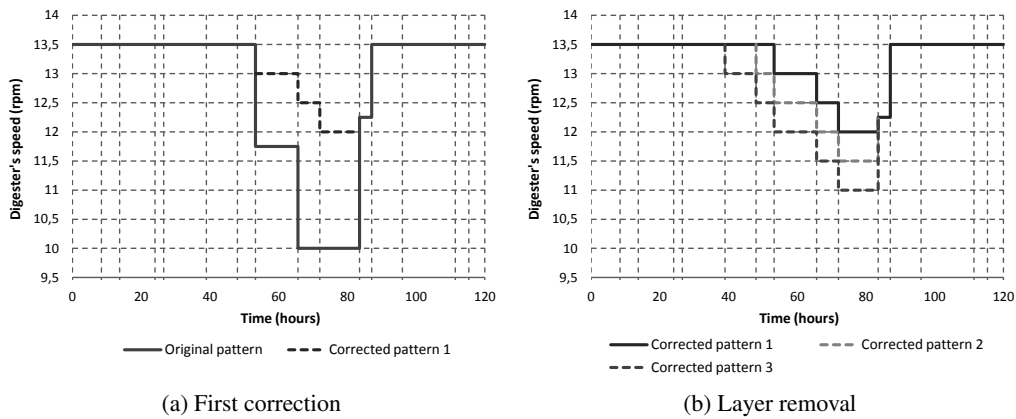


Figure 2.5: Speed pattern correction procedure.

The reduction of the previous speeds consists on an iterative procedure based on the idea of ‘layer removal’. This procedure is illustrated in Figure 2.5b. In Figure 2.5a the speed pattern starts at its maximum value (13.5 rpm in this case) and abruptly decreases in a given subperiod, due to one of the tanks being completely filled. Therefore, a correction is made to the speed pattern in order to eliminate the violations of the smooth speed shifts (*corrected pattern 1*) – see line 6 in Algorithm 13. However, this adjustment causes a violation in the tank levels. Thus, the speed pattern has to be lowered, i.e., *layers* have to be removed, as in the *corrected pattern 2* (where one layer was removed) and the *corrected pattern 3* (where two layers were removed). The layer removal procedure has to be executed until the area beneath the corrected pattern and above the original pattern (that corresponds to additional content inside the tanks) is less than the area above the former and under the latter (that corresponds to the content that is removed from the tanks). The speed reductions do not have to be performed through entire layers (this would lead to unnecessary extra reductions). Hence, they are executed subperiod by subperiod, in a forward move.

2.4. Computational tests and results

In this section we present the experimental results that validate the solution method design and show how it behaves for a varied set of instances. We start by presenting how the computational tests were designed. Then, the final results, which compare different variants of the proposed method, the MIP-based heuristic (*MIPBH*) by Santos and Almada-Lobo (2012) and IBM ILOG Cplex 12.1 (for the complete MIP), are exhibited. All the methods were implemented in C++, compiled using GCC and run on an Intel Xeon CPU E5504 @ 2.0 GHz, with 3 Gb of random access memory. Cplex was also used as the LP solver (within our algorithm) and it was limited to one thread in all methods to have a fair comparison.

2.4.1 Tests design

2.4.1.1 Instances

A set of different instances were generated based on real data from the company of the case study. The aim here is on the one hand to test the methods on instances as close as possible to real-world data and, on the other hand, to compare and analyse their behaviour on a variety of scenarios, which should represent different planning systems and strategies. Also, it is important to assess the robustness regarding different moments in time and KPIs (*Key Performance Indicators*) policies.

The configuration set of test classes is defined by all the combinations of $T = \{8, 15\}$ (where the number of periods per cycle $|T_c|$ is 5 or 10, respectively, corresponding to 1.5 cycles), $|S_t| = \{3, 4\}$ and $K = \{8, 16\}$. It is important to note that in spite of the total number of grades of the studied company being 29, only about a half of them has demand or backlog to fulfil in each production cycle and therefore is considered to be produced in that cycle. The instance class that matches the company's reality is defined by: $T = 15$ ($|T_c| = 10$ and $C = 1.5$) and $K = 16$. The number of subperiods is yet to be defined and it will depend on the quality of the final production plans (the trade-off between the accuracy of the model and the efficiency of the method is to be assessed in practice).

For each class, 10 different instances were generated, varying some parameters that appear to be more dynamic, namely demand intensity, initial tank levels and company's KPIs. For the demand intensity (which is defined here as the required time to produce all demand and backlog – excluding setups – divided by the total available time), a uniform distribution $U[0.65, 0.85]$ was used. Regarding the initial tank levels, a uniform $U[0.8, 1.2]$ was multiplied by the original values. If any of the lower or upper bounds were violated, the corresponding value was corrected to that bound. The KPIs (represented by w_i – see 3.4.4) were randomly varied between half and double of the original values provided by the decision maker, but considering the same probability of being less or greater than those values. Therefore, either $U[0.5, 1]$ or $U[1, 2]$ were used with a probability of 50% each.

2.4.1.2 Algorithm variants and general parameters

Variants of the main procedure of our algorithm mentioned in Section 5.4 are tested here. We choose not to combine variants, in order to have a reasonably small amount of methods to test. Still, the potential benefits of each alternative design can be assessed by comparing the results of the corresponding variant to those of the *Standard Algorithm*. The alternative variants are as follows:

- *DetMoves* uses a deterministic strategy for the sequence of neighbour solutions to be evaluated. Instead of choosing randomly one of the $pr\%$ best ranked moves in each iteration (where pr was chosen to be 10, based on preliminary tests), it selects the absolute best.
- *RelaxBCov* allows temporarily some violation of the backlog covering constraint (2.22) in the middle of the algorithm's procedure, in order to make the change of

setup patterns more flexible. Using adaptive penalties, the method alternately explores the feasible and the infeasible regions, in a balanced way. However, the exploration of the infeasible region begins only after the first local optimum has been reached. At that stage, the penalty starts being equal to the backlogging cost multiplied by the number of periods in a cycle and by a given coefficient: $w_{coef} \cdot \lambda_2 \cdot |T_c|$. Then, after a maximum number N_{max} of consecutive iterations in a given region (either feasible or infeasible), the penalty changes (increases if the search was performing in the infeasible region or decreases otherwise), according to a given rate w_{delta} . Some preliminary tests lead us to choose the following values for these parameters: $w_{coef} = 0.5$, $N_{max} = 5$ and $w_{delta} = 1.5$.

- *RandCH* applies a stochastic procedure to the first part of the *ConstructiveHeuristic*. This variant consists on selecting at each step of the sequencing phase not the following grade, but one of the $N_{restricted}$ closer, within the same product type. As a result, the conventional smooth sequence is replaced by a semi-random construction. A value of 2 is assigned to $N_{restricted}$, so that the randomness is not too strong.

The time limit for the computational experiments is set to one hour, which seems to be appropriate for a real-world application. The stochastic nature of the proposed methods (with the exception of *DetMoves*) makes it necessary to run each algorithm more than once, in order to obtain a reliable measure of its performance. Therefore, each stochastic method was run 5 times on each of the 10 different randomly generated instances of each class.

We choose to capture the quality of the final solutions (in stochastic variants, the average of the runs is used) by their gaps to the best obtained solution in all runs of all methods (including Cplex MIP) on each instance. The *Gap* is computed as $Gap = (z_i - z_{best}) / z_i$, where z_i is the average final solution of method i and z_{best} is the best obtained solution in all runs on a given instance. Results are aggregated (through arithmetic mean) by instance class, in order to provide a sense of the average performance of each method on each planning configuration. We also report the minimum and maximum gaps obtained by each method on every instance class.

2.4.2 Results

Table 2.1 shows the average gaps and running times (in seconds) and Table 2.2 exhibits the minimum and maximum gaps of every method on each instance class (for stochastic versions, the average of the runs on each instance was used). Comparing the different variants of the proposed algorithm, one can see that on small- to medium-sized instances (the first six instance classes) the *RelaxBCov* is the overall best, whereas on large-sized instances (the last two classes) the *Standard* version provides the best average gaps, for similar running times.

The flexibility of moving through the solution space provided by *RelaxBCov* appears to be more effective than a simple shaking, on instances of moderate size. In the last two classes of instances, the *Standard* version exceeds that variant. The reason for that may be in the fact that on larger instances the space is not so constrained and hence, crossing the infeasible region is not so important. Nevertheless, the effectiveness of that strategy is

Table 2.1: Average gaps and times of the four algorithm versions and Cplex on each instance class. Best results are in boldface.

K	T	$ S_t $	Standard		DetMoves		RelaxBCov		RandomCH		Cplex	
			AvgGap	Time	AvgGap	Time	AvgGap	Time	AvgGap	Time	AvgGap	Time
8	8	3	1.9%	339	2.0%	143	1.7%	933	6.8%	718	0.0%	1619
		4	2.8%	921	2.6%	500	2.8%	1189	5.6%	1026	0.2%	2792
	15	3	2.3%	3005	1.4%	3296	1.4%	3561	17.0%	3431	8.3%	3594
		4	3.4%	3539	4.8%	3336	2.0%	3636	18.2%	3576	18.8%	3597
16	8	3	14.1%	1284	17.9%	845	8.1%	3444	11.5%	1312	6.7%	3593
		4	5.3%	2931	13.0%	1561	3.7%	3561	4.2%	3113	14.3%	3598
	15	3	5.8%	3584	8.1%	3284	6.0%	3582	11.9%	3467	21.1%	3598
		4	6.9%	3435	12.0%	3506	7.4%	3631	15.6%	3592	71.5%	3595

Table 2.2: Minimum and maximum gaps of the four algorithm versions and Cplex on each instance class.

K	T	$ S_t $	Standard		DetMoves		RelaxBCov		RandomCH		Cplex	
			Min	Max	Min	Max	Min	Max	Min	Max	Min	Max
8	8	3	0.0%	4.8%	0.0%	3.7%	0.0%	4.6%	0.9%	18.2%	0.0%	0.2%
		4	0.0%	7.7%	0.0%	6.4%	0.0%	7.7%	1.6%	17.4%	0.0%	1.3%
	15	3	0.0%	8.0%	0.0%	6.0%	0.4%	3.7%	4.2%	26.2%	1.8%	14.1%
		4	0.3%	9.4%	0.0%	36.0%	0.3%	5.6%	4.9%	27.5%	3.7%	33.3%
16	8	3	2.5%	48.6%	4.2%	55.1%	0.3%	47.6%	2.5%	31.5%	0.0%	43.7%
		4	0.4%	13.6%	2.0%	28.7%	0.3%	9.7%	0.8%	9.9%	1.5%	33.3%
	15	3	0.8%	10.2%	0.9%	15.8%	2.9%	10.4%	7.2%	17.1%	11.4%	38.0%
		4	3.5%	13.6%	6.2%	17.1%	1.6%	12.8%	5.7%	27.6%	20.9%	98.1%

often rather impressive, even on large-sized instances. Figure 2.6 illustrates the behaviour of the optimization process (switching between feasible and infeasible spaces) on one of the instances of the seventh class ($K = 16$, $T = 15$ and $|S_t| = 3$). The progress of the penalty value (properly adjusted to the scale of the figure) is also exhibited. The importance of the backlog covering relaxation is evident in the figure, where the algorithm is able at four different times to further improve the first local optimum. For effectively switching between feasible and infeasible solutions, the behaviour of the penalty function is determinant.

The *DetMoves* variant, on the other hand, proves to be very efficient just on the smallest instances (the first two classes), obtaining high quality solutions within remarkably short times. On larger instances, the gaps are consistently worse (on average) than those of *Standard*, showing that it is not beneficial to use a completely greedy strategy in the raking system when exploring neighbourhoods. Therefore, we can conclude that the random component in this exploration prevents premature convergence towards local optima in relatively large solution spaces.

The *RandCH* can achieve rather good results when the number of products is large in comparison to the number of periods (in the fifth and sixth instance classes). However, that is not true on the remaining instances, where it is often the worst variant. Indeed, it seems that even a semi-random sequencing procedure in the *ConstructiveHeuristic* spoils too much the setup pattern and the algorithm cannot fix it afterwards. Hence, a fairly

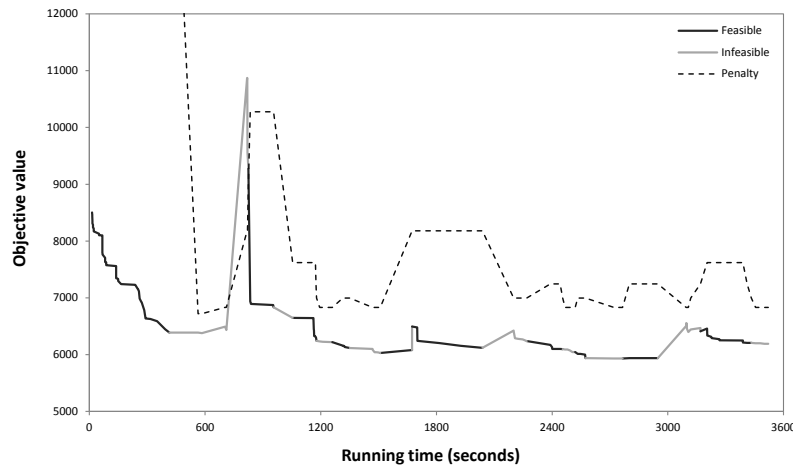


Figure 2.6: Algorithm's behaviour (*RelaxBCov* variant): switching between feasible and infeasible spaces.

greedy constructive heuristic is desirable for this problem.

Regarding robustness, the *RelaxBCov* appears to be the best variant, since its ranges of gaps seem to be relatively short. In the fifth instance however the maximum gap is around 50% for all the variants (except for *RandCH*, which presents a fairly good performance on those instances, but the worst robustness overall). It seems that the algorithm finds it hard to perform consistently when dealing with several products within few periods.

Looking at the Cplex results on the small-sized instances (namely when $K = 8$ and $T = 8$), one can see that, as expected, it is not only able to find provably optimal solutions within the time limit for almost all the instances, but it also proves to be superior to the approximate algorithms. However, for medium and large-sized instances, that superiority is inverted and Cplex deteriorates its performance sharply, specially as the number of periods increases. Thus, we can conclude that in real-world applications like that of our case study, exact methods fail to obtain good quality solutions.

Figure 2.7 shows the progress of the best solution throughout the search of the *Standard* version and Cplex, on the same instance as that of Figure 2.6. It is clear from the figure the incredibly fast convergence of the algorithm towards good quality solutions. Indeed, it is able to provide within few seconds a solution that is drastically better than that obtained by Cplex running a complete hour.

The results of the MIP-based heuristic (proposed in the previous paper) demonstrate that, although this kind of approaches is more effective than Cplex for medium- to large-sized instances, its performance also suffers significantly as the instance size increases (contrary to the algorithm proposed in the present paper). Furthermore, it is similar to Cplex in what concerns the gradual generation of good solutions. Therefore, for reduced time limits the advantage of the new method would be even more clear.

The final plans generated by our method were validated with the company. They are naturally very similar to those provided by Santos and Almada-Lobo (2012), as we use virtually the same constraints, but respond better to the defined objective function criteria,

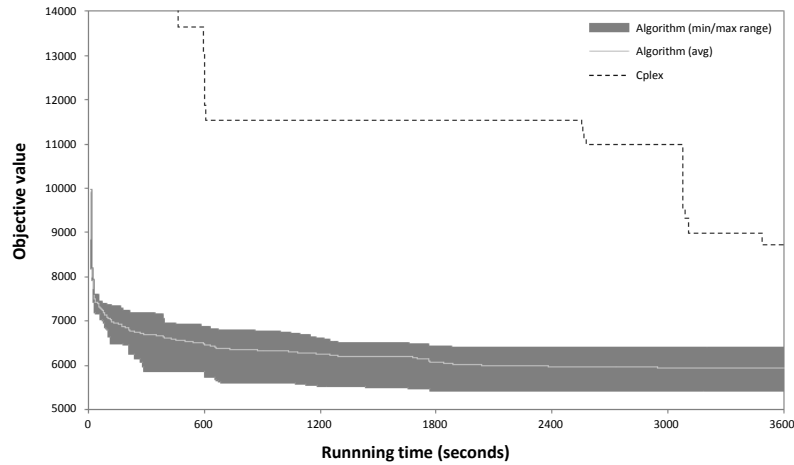


Figure 2.7: Algorithm's behaviour (*Standard* version compared to Cplex): best solution throughout the optimization process.

since the algorithm is more effective in finding good solutions.

2.5. Conclusions

Exact solvers fail to generate good quality solutions for complex multi-stage lotsizing and scheduling models on large real-world sized instances within feasible time. In this paper we explore a case study in the P&P industry, where different production stages are integrated into a single formulation. We propose a hybrid approach that combines the GVNS metaheuristic (which manages the paper machine setup pattern), a specific heuristic to determine the digester's speeds (the SCH) and an exact solver for the continuous variables (material flow and inventory levels).

Computational tests have demonstrated that our method is significantly more efficient than both pure exact methods and MIP-based heuristics on real-sized instances. Different algorithm variants were tested on a variety of instances. Results suggest that on small-to medium sized instances the relaxation of backlog covering constraints appears to be a more effective strategy to explore the solution space, rather than supporting the whole diversification scheme in the shaking phase. On the largest instances (which correspond to our case study), the *Standard* method provides better results.

Our method is driven by a real-world case study of the short-term production planning in an integrated P&P mill company. However, it can be applied to many other LSS problems (in or outside the industry) formulated by a small-bucket MIP model where the integer part consists of a setup pattern and/or a grid of production speeds (constrained to a maximum variation). The notion of cycles is also taken into account by our solution method.

The plans generated by our algorithm were validated with the company and their quality was confirmed. However, they are still subject to several changes and adaptations during

their implementation and cannot guarantee feasibility in the presence of disturbances. Indeed, the system is stochastic, since processes have an uncertain behaviour and production resources, such as the paper machine, are subject to unpredicted interruptions. Therefore, further research regarding the generation of more robust production plans is desirable.

Currently, the propagation of disturbances is avoided by considering slacks in the tank limits when creating plans. We can try to optimize those slacks, instead of leaving them as predefined decisions. In this way, both quality and robustness of production plans would be correctly weighted.

Bibliography

- B. Almada-Lobo, J.F. Oliveira, and M.A. Carravilla. Production planning and scheduling in the glass container industry: A vns approach. *International Journal of Production Economics*, 114(1):363–375, 2008.
- Christian Almeder. A hybrid optimization approach for multi-level capacitated lot-sizing problems. *European Journal of Operational Research*, 200(2):599–606, 2010. ISSN 0377-2217.
- E. Axelsson, M.R. Olsson, and T. Berntsson. Increased capacity in kraft pulp mills: Lignin separation and reduced steam demand compared with recovery boiler upgrade. *Nordic Pulp and Paper Research Journal*, 21(4):485–492, 2006.
- C. Blum and A. Roli. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Comput. Surv.*, 35:268–308, 2003. ISSN 0360-0300.
- H. Bouchriha, M. Ouhimmou, and S. D’Amours. Lot sizing problem on a paper machine under a cyclic production approach. *International Journal of Production Economics*, 105(2):318–328, 2007.
- David Bredstrom, Jan T. Lundgren, Mikael Ronnqvist, Dick Carlsson, and Andrew Mason. Supply chain optimization in the pulp mill industry – ip models, column generation and novel constraint branches. *European Journal of Operational Research*, 156(1):2–22, 2004. ISSN 0377-2217.
- V. Camargo, F. Toledo, and B. Almada-Lobo. Three time-based scale formulations for the two-stage lot sizing and scheduling in process industries. *Journal of the Operational Research Society*, 63:1613–1630, 2012.
- Pedro M. Castro, Joakim Westerlund, and Sebastian Forssell. Scheduling of a continuous plant with recycling of byproducts: A case study from a tissue paper mill. *Computers & Chemical Engineering*, 33(1):347–358, 2009.
- A. Clark, B. Almada-Lobo, and C. Almeder. Lot sizing and scheduling: Industrial extensions and research opportunities. *International Journal of Production Research*, 49(9): 2457–2461, 2011.

- C.D. Eppen. Solving multi-item capacitated lot-sizing problems using variable redefinition. *Operations Research*, 35(6):832–848, 1987.
- B. Fleischmann and H. Meyr. The general lotsizing and scheduling problem. *OR Spectrum*, 19:11–21, 1997. ISSN 0171-6468.
- Luis Guimaraes, Diego Klabjan, and Bernardo Almada-Lobo. Annual production budget in the beverage industry. *Engineering Applications of Artificial Intelligence*, 25(2):229–241, 2012. ISSN 0952-1976.
- Pierre Hansen, Nenad Mladenovic, and Jose Moreno Perez. Variable neighbourhood search: methods and applications. *4OR: A Quarterly Journal of Operations Research*, 6:319–360, 2008. ISSN 1619-4500.
- K.S. Hindi. Solving the clsp by a tabu search heuristic. *Journal of the Operational Research Society*, 47(1):151–161, 1996.
- Yi-Feng Hung, Ching-Ping Chen, Chia-Chung Shih, and Ming-Hsiau Hung. Using tabu search with ranking candidate list to solve production planning problems with setups. *Computers & Industrial Engineering*, 45(4):615–634, 2003. ISSN 0360-8352.
- Raf Jans and Zeger Degraeve. Meta-heuristics for dynamic lot sizing: A review and comparison of solution approaches. *European Journal of Operational Research*, 177(3):1855–1875, 2007. ISSN 0377-2217.
- J. Jonsson, I.L. Svensson, T. Berntsson, and B. Moshfegh. Excess heat from kraft pulp mills: Trade-offs between internal and external use in the case of sweden—part 2: Results for future energy market scenarios. *Energy Policy*, 36(11):4186–4197, 2008. ISSN 0301-4215.
- H. Meyr. Simultaneous lotsizing and scheduling by combining local search with dual re-optimization. *European Journal of Operational Research*, 120(2):311–326, 2000. ISSN 0377-2217.
- H. Meyr. Simultaneous lotsizing and scheduling on parallel machines. *European Journal of Operational Research*, 139(2):277–292, 2002. ISSN 0377-2217.
- Sesh Murthy, Rama Akkiraju, Richard Goodwin, Pinar Keskinocak, John Rachlin, Frederick Wu, James Yeh, Robert Fuhrer, Santhosh Kumaran, Alok Aggarwal, Martin Sturzenbecker, Ranga Jayaraman, and Robert Daigle. Cooperative multiobjective decision support for the paper industry. *Interfaces*, 29:5–30, 1999. ISSN 0092-2102.
- N. Rizk, A. Martel, and S. D’Amours. Synchronized production-distribution planning in a single-plant multi-destination network. *Journal of the Operational Research Society*, 59(1):90–104, 2008.
- Maristela Oliveira Santos and Bernardo Almada-Lobo. Integrated pulp and paper mill planning and scheduling. *Computers & Industrial Engineering*, 63(1):1–12, 2012. ISSN 0360-8352.

K. Schumacher and J. Sathaye. India's pulp and paper industry: Productivity and energy efficiency. Technical report, LBNL-41843, 1999.

Appendix 2.A Integrated P&P mathematical formulation

The presentation of the MIP model is divided into four parts: one for each of the three main production steps (related to pulp mill, paper mill and chemicals recovery plant) and the last dedicated to the objective function. The indices, parameters and decision variables are defined along the presentation of the model.

2.A.1 Pulp mill

Parameters, indices and sets:

t	Index for period ($t \in [T] = \{1, \dots, T\}$)
s	Index for subperiod ($s \in [S] = \{1, \dots, S\}$)
$[S_t]$	Set of subperiods s belonging to period t
Φ	Minimum possible speed variation (in rpm)
Δ	Maximum possible speed step
V	Number of speeds, that is equal to $\frac{v_{max}^{dig} - v_{min}^{dig}}{\Phi} + 1$
v_{max}^{dig} (v_{min}^{dig})	Maximum (minimum) speed of the digester (in rpm)
v	Index for digester's rotation speed ($v \in [V] = \{1, \dots, V\}$)
sp_v	Value of the speed indexed by v (e.g. $sp_1 = v_{min}^{dig}$)
cap_t	Capacity of period t (in hours)
N_{min}	Minimum subperiod length (in hours)
α	Digester's conversion parameter from rpm to tonnes
$Y_{v,0}^{dig}$	$\begin{cases} 1 & \text{if the digester is running at speed } v \text{ at the beginning of the planning horizon,} \\ 0 & \text{otherwise} \end{cases}$
xr_{max}^{recy} (xr_{min}^{recy})	Maximum (minimum) production rate of the recycled pulp plant (in tonnes per hour)
k_{cap}^{recy}	Capacity coefficient of recycled fibre plant
I_0^{virg} (I_0^{recy})	Initial inventory of virgin (recycled) pulp at the beginning of the planning horizon (in tonnes)
I_{max}^{virg} (I_{min}^{virg})	Upper (lower) bounds on the virgin pulp stocked in the respective tank (in tonnes)
I_{max}^{recy} (I_{min}^{recy})	Upper (lower) bounds on the recycled pulp stocked in the respective tank (in tonnes)

Decision variables:

Y_{vs}^{dig}	$\begin{cases} 1 & \text{if the digester runs at speed } v \text{ in subperiod } s, \\ 0 & \text{otherwise} \end{cases}$
N_s	Length of subperiod s (in hours)
Nh_{vs}	Time that the digester works at speed v in subperiod s (in hours)
X_s^{dig}	Output of the digester in subperiod s (in tonnes of virgin pulp)
X_s^{recy}	Output of the recycled pulp mill in subperiod s (in tonnes of recycled pulp)
I_s^{virg} (I_s^{recy})	Inventory of virgin (recycled) pulp in the respective tank at the end of micro-period s (in tonnes)
O_s^{virg} (O_s^{recy})	Output of virgin (recycled) pulp tank in micro-period s (in tonnes)
IS_s^{recy}	Quantity of pulp fed back to the recycled pulp mill during the paper grade changeover in s (in tonnes)

$$\sum_v Y_{vs}^{dig} = 1, \quad s \in [S] \quad (2.3)$$

$$Y_{vs}^{dig} \leq \sum_{w=\max(v-\Delta,0)}^{\min(v+\Delta,V)} Y_{w,s-1}^{dig}, \quad v \in [V], s \in [S] \quad (2.4)$$

$$Nh_{vs} \leq cap_t \cdot Y_{vs}^{dig}, \quad v \in [V], s \in [S] \quad (2.5)$$

$$N_s = \sum_v Nh_{vs}, \quad s \in [S] \quad (2.6)$$

$$\sum_{s \in [S_t]} N_s = cap_t, \quad t \in [T] \quad (2.7)$$

$$N_s \geq N_{min}, \quad s \in [S] \quad (2.8)$$

$$X_s^{dig} = \alpha \cdot \sum_v sp_v \cdot Nh_{vs}, \quad s \in [S] \quad (2.9)$$

$$X_s^{recy} \geq xr_{min}^{recy} \cdot N_s, \quad s \in [S] \quad (2.10)$$

$$X_s^{recy} \leq xr_{max}^{recy} \cdot N_s, \quad s \in [S] \quad (2.11)$$

$$\sum_{s \in [S_t]} X_s^{recy} \leq k_{cap}^{recy} \cdot xr_{max}^{recy} \cdot cap_t, \quad t \in [T] \quad (2.12)$$

$$X_s^{dig} + I_{s-1}^{virg} = O_s^{virg} + I_s^{virg}, \quad s \in [S] \quad (2.13)$$

$$X_s^{recy} + I_{s-1}^{recy} + IS_s^{recy} = O_s^{recy} + I_s^{recy}, \quad s \in [S] \quad (2.14)$$

$$I_{min}^{virg} \leq I_s^{virg} \leq I_{max}^{virg}, \quad s \in [S] \quad (2.15)$$

$$I_{min}^{recy} \leq I_s^{recy} \leq I_{max}^{recy}, \quad s \in [S] \quad (2.16)$$

The digester runs at the same speed throughout each subperiod s (constraints (2.3)). Naturally, the speed of the digester in subperiod s is implicitly determined by $V_s^{dig} = \sum_v sp_v \cdot Y_{vs}^{dig}$. Constraints (2.4) ensure a smooth shift of the speed of the digester between two consecutive subperiods. The working speed of the digester and the length of each subperiod s (which is common to all resources) are defined by constraints (2.5) and (2.6). The sizes of the micro-periods of the same period altogether cannot exceed the capacity of the respective period (constraints (2.7)), but each of them has a minimum value (constraints (2.8)), so that the digester keeps its speed during at least a given amount of time. The amount of virgin pulp in subperiod s is proportional to the speed of the digester and to the length of s , as in constraints (2.9).

The production of the recycled pulp plant (where the waste collected is refined) in each subperiod, X_s^{recy} , is limited to lower (xr_{min}^{recy}) and upper bounds (xr_{max}^{recy}) of production rate. Additionally, the whole production in a period cannot exceed k_{cap}^{recy} of the total capacity of the corresponding period. These constraints are expressed in (2.10) - (2.12).

Both the virgin and recycled pulp tanks have limited storage capacity and must be filled with stock over a minimum limit (constraints (2.15) and (2.16)). The inventory balancing equations of virgin and recycled pulp are given by constraints (2.13) and (2.14). In the case of recycled pulp tanks, it is necessary to consider the recovery of the paper loss during grade changeovers and the cutting stage (see constraints (2.14)).

2.A.2 Paper mill

Parameters, indices and sets:

j, k	Indices for paper grades ($j, k \in [K] = \{1, \dots, K\}$)
j_1	Common first paper grade of all the cycles
c	Index for cycle ($c \in [C^-] = \{1, \dots, [C]\}$ or $c \in [C^+] = \{1, \dots, [C]\}$, where C is the number of cycles, which can be non-integer)
$[T_c]$	Set of periods t belonging to cycle c ($T = \lceil T_c \rceil \cdot C, c \in [C^-]$)
$[S_{cycle_c}]$	Set of subperiods s belonging to cycle c
t_{1c}	First period of cycle c
s_{1c}	First subperiod of cycle c
sl_{kj}	Paper lost (setup cost) in a changeover from grade k to j (in tonnes)
st_{kj}	Time lost (setup time) in a changeover from grade k to j (in hours)
$b_j^{virg} (1 - b_j^{virg})$	Percentage of virgin (recycled) pulp used in the production of paper grade j
f_j	Percentage of water in paper grade j
p_j	Minimum processing time to produce one tonne of paper grade j
M_j	Large number for each paper grade j
m_j	Minimum lot size of paper grade j
D_{jt}	Demand for paper grade j in period t (in tonnes)
ι	Average trim loss (in percentage)
Y_{j0}	$\begin{cases} 1 & \text{if the paper machine is set up for grade } j \text{ at the beginning of the planning horizon,} \\ 0 & \text{otherwise.} \end{cases}$

Decision variables:

Z_{kjs}	$\begin{cases} 1 & \text{if a changeover from grade } k \text{ to } j \text{ takes place at the beginning of subperiod } s, \\ 0 & \text{otherwise.} \end{cases}$
Y_{js}	$\begin{cases} 1 & \text{if the paper machine is set up for grade } j \text{ in subperiod } s, \\ 0 & \text{otherwise.} \end{cases}$
X_{js}	Quantity of paper grade j produced in subperiod s (in tonnes)
$IG_{j,t}^+$	Inventory of paper grade j at the end of period t (in tonnes)
$IG_{j,t}^-$	Quantity of paper grade j backlogged at the end of period t (in tonnes)

$$\sum_j b_j^{virg} \cdot (1 - f_j) \cdot \left(X_{js} + \sum_k sl_{kj} \cdot Z_{kjs} \right) = O_s^{virg}, \quad s \in [S] \quad (2.17)$$

$$\sum_j (1 - b_j^{virg}) \cdot (1 - f_j) \cdot \left(X_{js} + \sum_k sl_{kj} \cdot Z_{kjs} \right) = O_s^{recy}, \quad s \in [S] \quad (2.18)$$

$$\sum_j \sum_k (1 - f_j) \cdot sl_{kj} \cdot Z_{kjs} + \sum_j \iota \cdot X_{js} = IS_s^{recy}, \quad s \in [S] \quad (2.19)$$

$$\sum_j \left(p_j \cdot X_{js} + \sum_k st_{kj} \cdot Z_{kjs} \right) = N_s, \quad s \in [S] \quad (2.20)$$

$$(1 - \iota) \cdot \sum_{s \in [S_t]} X_{js} + IG_{j,t-1}^+ - IG_{j,t-1}^- - D_{jt} = IG_{jt}^+ - IG_{jt}^-, \quad j \in [K], t \in [T] \quad (2.21)$$

$$(1 - \iota) \cdot \sum_{s \in [S_{cycle_c}]} X_{js} \geq IG_{j,t_{1c}-1}^-, \quad j \in [K], c \in [C^-] \quad (2.22)$$

$$X_{js} \leq M_j \cdot Y_{js}, \quad j \in [K], s \in [S] \quad (2.23)$$

$$X_{js} \geq m_j \cdot (Y_{js} - Y_{j,s-1}), \quad j \in [K], s \in [S] \quad (2.24)$$

$$\sum_j Y_{js} \leq 1, \quad s \in [S] \quad (2.25)$$

$$Y_{j_1, s_{1c}} = 1, \quad c \in [C^+] \quad (2.26)$$

$$\sum_k Z_{jks} = Y_{j,s-1}, \quad j \in [K], s \in [S] \quad (2.27)$$

$$\sum_k Z_{kjs} = Y_{js}, \quad j \in [K], s \in [S] \quad (2.28)$$

Constraints (2.17) and (2.18) keep track of the usage of virgin and recycled pulp, respectively, during production and setup tasks in each subperiod (where water is added to it). In the P&P mill, the amount of paper lost during grade switchovers and reels cutting is fed back into the recycled pulp mill and is traced by constraints (2.19).

It should be remembered that there is a common time grid for all production resources. Constraints (2.20) define the size of each micro-period as a function of the production and setup times of the paper machine performed in that subperiod. These requirements, together with (2.6), synchronize the time grid of the digester with that of the paper machine in each subperiod.

Constraints (2.21) represent the balance of inventory of paper grade j in period t . The initial inventory of each grade is null ($IG_{j,0}^+ = 0 \forall j$). The term $(1 - \iota) \cdot \sum_{s \in [S, t]} X_{js}$ deducts the trim loss out of the overall quantity produced. Constraints (2.22) ensure that in each cycle, the initial backlog of each grade is fulfilled until the end of that cycle.

Constraints (2.23) establish that production of grade j occurs in subperiod s if the machine is set up for that grade in subperiod s . In that case, constraints (2.24) force a minimum lot size for the whole run. From (2.25) at most one grade can be produced per subperiod. Constraints (2.26) impose that the first grade of each cycle is always the same and thus, they create the perception of cycles (which is important for planners). Finally, constraints (2.27) and (2.28) relate the changeover variables to the setup state variables.

2.A.3 Recovery plant

Parameters:

I_0^{liquor}	Initial inventory of weak black liquor (in m^3)
$I_{max}^{liquor} (I_{min}^{liquor})$	Maximum (minimum) inventory of weak black liquor in the buffer (in m^3)
C^{evap}	Capacity of the evaporator to process the black liquor (in m^3 per hour)
ρ	Ratio between digester's pulp and weak black liquor production
β	Conversion parameter from weak black liquor to concentrated black liquor
$I_0^{c.liq}$	Initial inventory of concentrated black liquor (in m^3)
$I_{max}^{c.liq} (I_{min}^{c.liq})$	Maximum (minimum) holding stock of concentrated black liquor in the buffer (in m^3)
$C^{r.boiler}$	Capacity of the recovery boiler to produce steam (in tonnes per hour)
σ	Conversion factor of concentrated black liquor to steam
$C_{burn}^{r.boiler}$	Burning capacity of the recovery boiler (in m^3 per hour)

Decision variables:

X_s^{liquor}	Quantity of weak black liquor produced by the digester in subperiod s (in m^3)
O_s^{liquor}	Quantity of weak black liquor evaporated in subperiod s by the evaporator (in m^3)
I_s^{liquor}	Inventory of weak black liquor at the end of subperiod s (in m^3)
$X_s^{c.liq}$	Quantity of concentrated black liquor produced in subperiod s (in m^3)
$I_s^{c.liq}$	Inventory of concentrated black liquor at the end of subperiod s (in m^3)
$O_s^{c.liq}$	Quantity of concentrated black liquor to be burnt (in m^3)
X_s^{steam}	Quantity of steam produced in the recovery boiler (in tonnes)

$$X_s^{liquor} = \rho \cdot X_s^{dig}, \quad s \in [S] \quad (2.29)$$

$$X_s^{liquor} + I_{s-1}^{liquor} = O_s^{liquor} + I_s^{liquor}, \quad s \in [S] \quad (2.30)$$

$$I_{min}^{liquor} \leq I_s^{liquor} \leq I_{max}^{liquor}, \quad s \in [S] \quad (2.31)$$

$$O_s^{liquor} \leq C^{evap} \cdot N_s, \quad s \in [S] \quad (2.32)$$

$$X_s^{c.liq} = \beta \cdot O_s^{liquor}, \quad s \in [S] \quad (2.33)$$

$$X_s^{c.liq} + I_{s-1}^{c.liq} = O_s^{c.liq} + I_s^{c.liq}, \quad s \in [S] \quad (2.34)$$

$$I_{min}^{c.liq} \leq I_s^{c.liq} \leq I_{max}^{c.liq}, \quad s \in [S] \quad (2.35)$$

$$O_s^{c.liq} \leq C_{burn}^{r.boiler} \cdot N_s, \quad s \in [S] \quad (2.36)$$

$$X_s^{steam} = \sigma \cdot O_s^{c.liq}, \quad s \in [S] \quad (2.37)$$

$$X_s^{steam} \leq C_{steam}^{r.boiler} \cdot N_s, \quad s \in [S] \quad (2.38)$$

The amount of weak black liquor, X_s^{liquor} , expressed in m^3 , is proportional to the digester's virgin pulp output, and is given by constraints (2.29). The buffer between the digester and the evaporator is constrained by lower and upper bounds (constraints (2.31)). Constraints (2.32) establish the evaporator's capacity and the inventory balance equations on the weak black liquor are given by (2.30). The output of the evaporation process is the concentrated black liquor, $X_s^{c.liq}$ that is proportional to the input O_s^{liquor} , as given by constraints (2.33).

The mass balance equations of the concentrated black liquor tank are given by constraints (2.34), while the storage capacity of the buffer is defined by (2.35). Constraints

(2.36) ensure that the burning capacity is limited in each subperiod. Finally, the steam generated by the recovery boiler is proportional to the amount of dry solid content that is burnt at each time, as indicated in (2.37). The steam output cannot exceed an upper limit – see constraints (2.38).

2.A.4 Objective function and overall model

$$\begin{aligned}
 Obj = & \sum_j \sum_t \lambda_1 \cdot IG_{jt}^+ + \sum_j \sum_t \lambda_2 \cdot IG_{jt}^- + \sum_j \sum_k \sum_s \lambda_3 \cdot sl_{kj} \cdot Z_{kjs} - \sum_s \lambda_4 \cdot X_s^{steam} \\
 & - \sum_s \lambda_5 \cdot O_s^{liquor} - \sum_s \lambda_6 \cdot X_s^{recy} - \sum_s \lambda_7 \cdot X_s^{dig}
 \end{aligned} \tag{2.39}$$

Expression (2.39) represents the weighted sum of the objective function. Its coefficients (λ_i 's) were determined based on the preferences of the decision-maker. To make this process easier, the following expression was used: $\sum_i (w_i \cdot F_i / F_i^{max})$, where w_i 's are the weights to be parametrized, F_i 's are the sub-functions and F_i^{max} 's are the maximum values these functions can take. Therefore, the sub-functions are normalized and hence, easier to be weighted. The weights to be applied to the output flow variables (the last four terms) should increase as we go downstream the production process, so that stages closer to the overall mill's output are not hindered by upstream stages. Therefore, a factor of 2 was applied between the weights of two consecutive stages and hence, we have: $w_4 = 2 \cdot w_5 = 4 \cdot w_6 = 4 \cdot w_7$. The values of the λ_i 's can be determined by dividing the corresponding w_i 's by the F_i^{max} 's.

The overall model then reads:

$$\begin{aligned}
 & \min Obj \\
 & \text{subject to (2.3)–(2.38),} \\
 & N_s, Nh_{vs}, X_s^{dig}, X_s^{recy}, I_s^{virg}, I_s^{recy}, O_s^{virg}, O_s^{recy}, IS_s^{recy}, X_{js}, IG_{jt}^+, IG_{jt}^-, X_s^{liquor}, O_s^{liquor}, \\
 & I_s^{liquor}, X_s^{c.liq}, I_s^{c.liq}, O_s^{c.liq}, X_s^{steam} \geq 0, \\
 & Y_{vs}^{dig}, Z_{kjs}, Y_{js} \in \{0, 1\}.
 \end{aligned}$$

Decision support system for production planning and scheduling

A decision support system for the operational production planning and scheduling of an integrated pulp and paper mill

Gonçalo Figueira* · Pedro Amorim* · Luís Guimarães* · Mário Amorim Lopes* · Bernardo Almada-Lobo*

Submitted to *Computers & Chemical Engineering*, 2014

Abstract Production planning and scheduling in the process industry in general and in the pulp and paper (P&P) sector in particular can be very challenging. Most practitioners, however, address those activities relying only on spreadsheets, which is time-consuming and sub-optimal. The literature has reported some decision support systems (DSSs) that are far from the state-of-the-art with respect to optimization models and methods, and several research works that do not address industrial issues. We contribute to reduce that gap by developing and describing a DSS that resulted from several iterations with a P&P company and a thorough review of the process systems engineering literature. The DSS incorporates relevant industrial features (which motivated the development of a specific model), exhibits important technical details (such as the connection to existing systems and user-friendly interfaces) and shows how optimization can be integrated in real world applications, enhanced by key pre- and post-optimization procedures.

Keywords Decision Support System · Lot-sizing and Scheduling · MIP-based heuristics · Pulp and Paper Industry · Continuous Production

3.1. Introduction

The pulp and paper (P&P) industry is highly capital intensive, which means that investments in capacity can represent very long-term decisions. For instance, the modification of a single paper machine requires a planning horizon of at least five years (Martel et al.,

*INESC TEC, Faculdade de Engenharia, Universidade do Porto, Porto, Portugal

2005). Paper products are commodities, with their price determined in the market and characterized by small margins. Hence, companies must differentiate themselves by improving customer satisfaction indicators, while keeping production costs as low as possible. Furthermore, the P&P production process is also energy intensive. Producing one tonne of paper requires 5–17 GJ of process heat, depending on the paper type and on the technology applied (Szabó et al., 2009). The P&P industry uses 84% of the fuel energy consumed by the forest products industry as a whole and it is one of the largest producers of greenhouse gas (GHG) emissions. Therefore, it is of particular interest in the context of environmental discussions.

These three main factors (capital intensity, energy intensity and competitive market) make the production planning an essential activity in the quest for improvements in operational efficiency and consequently economic gains. However, the planning process poses a variety of challenges, both to practical and scientific fields. If these challenges are successfully addressed, companies can achieve true competitive advantage.

In most cases, production planning is addressed manually by practitioners, even in modern mills with sophisticated automated systems. That applies not only to the P&P industry, but to the process industry in general (see Harjunkski et al. (2014)). Companies may use advanced tools for particular tasks, such as the planning of the paper reels' cutting, but when approaching the overall planning activity (e.g. size and sequence of paper campaigns, production rates, etc.) most practitioners rely only on spreadsheets. This manual process is time-consuming, sub-optimal (as only few alternatives are considered) and completely dependent on the planners' expertise.

Therefore, there is thus the need for optimization-based tools that support decision making in the operational production planning of these mills. Some decision support systems (DSS) for this particular industry were reported in the literature (e.g. Murthy et al. (1999); Respício and Captivo (2008)). Although considering relevant industrial features and practical issues, the underlying approaches are far from the state-of-the-art in production planning. On the other hand, more research-oriented works do not address the issues of an industrial implementation.

Our work helps closing the gap between research and practice, proposing an optimization-based DSS, which improves on manual planning and contributes to the literature in the following ways:

- exploring desirable characteristics of analytical models and methods;
- identifying relevant industrial features and how they should be included in the solution approach;
- extending models of the literature, considering practical constraints and objectives;
- exhibiting important data processing in both pre- and post-optimization phases;
- illustrating required connections to existing systems and desirable aspects in the user interface.

We start by describing the industrial system and the planning challenge in Section 3.2. This motivates the discussion in Section 5.6.1 on the different approaches in the literature

for the operational production planning. In that section we explore some desirable characteristics of optimization models and identify relevant features for industrial practice. Based on that, we choose an appropriate mathematical model and extend it in order to include the identified practical issues. Our formulation is presented in Section 3.4. Section 5.4 motivates and explores the solution method for an efficient yet simple and flexible resolution of this complex problem. Section 3.6 details the integration of the optimization in the decision support system, describing pre- and post-optimization steps, as well as the interfaces to the existing information systems and to the final user. The usage of the system is also discussed. In Section 5.7 we compare a plan obtained with the system to one manually generated by production planners and give further details of the performance and usage of the system by the company. Finally, the last section summarizes the benefits of our DSS, discusses its applicability to other environments and shows possible directions for further improvement.

3.2. The challenge

The P&P production process is illustrated in Figure 5.1. The variables depicted in this Figure will be introduced later in Section 3.4. In a first step, both virgin and recycled pulps are produced out of wood and recycled paper, respectively. These pulps are then stored in tanks, waiting for being pulled by the paper machine. The machine can produce different types of paper. Each type (or grade) is characterized by its grammage (measured in g/m^2) and pulp mixture. The configuration of the machine to produce a different grade is sequence-dependent. Each setup leads to a loss in the production process in terms of time and quantity of a lower quality paper produced (as the machine is never idle and the paper produced will not be homogeneous nor satisfy completely the customer requirements). The wasted paper (setup loss) is fed again into the recycled pulp mill and a loss pulp tank.

The master reel that results at the end of the paper machine, the jumbo, is cut into smaller reels. The wasted paper in the cutting stage (trim loss) is also fed back to the production process. Customers place orders for reels of different widths and grades. The orders may have different priority levels. The maximum priority is given to those that travel by ship, since the company has to schedule containers in advance and commit to a given due date. Then, the remaining is divided into normal and priority orders.

In parallel, a by-product of the digester's pulping process, the weak black liquor, is concentrated and burnt to provide high-pressure steam and regenerate the spent chemicals applied in the pulping stage. The steam can either be used for the paper drying process or be led to counter-pressure turbines which produce electrical energy to be sold afterwards.

In other companies, the paper mill can be physically distant from the pulp and recovery plants. However, integrated plants, like our case study, represent 65% of the industry (CEPI, 2013) and are more capable of achieving high levels of both energy and economic efficiency, due to:

- energy conservation (e.g. direct use of steam in the paper drying process);
- absence of additional processes (e.g. pulp drying);

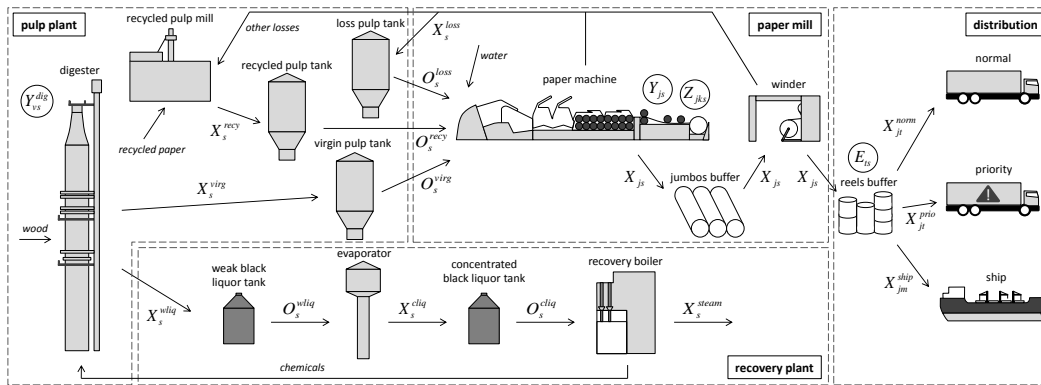


Figure 3.1: The integrated pulp, paper and recovery plant, with multiple distribution channels.

- material closed loops (e.g. recycling of paper machine setup and trim losses) which allow reducing waste production and the energy spent on it;
- tightly integrated equipment, which reduces the required capacity.

Nevertheless, an integrated mill poses additional difficulties. The complex production process described above (with convergent, divergent and loop flows) and the tightly integrated equipment, with limited intermediate storage space, result in multiple and shifting bottlenecks. For instance, the company under study produces two main products (KLB and VLB), which have major differences with respect to the incorporation of virgin and recycled fibres. Hence, the bottleneck stage clearly shifts according to the mixture being produced (for example, VLB quickly exhausts the recycled pulp mill, whereas KLB is typically restricted by the recovery boiler).

On the other hand, the orders placed by customers put a great pressure on production, as the system operates in a make-to-order (MTO) policy. Still, adjusting the production sequence to better meet market needs has to be done carefully, since the desired production cycles will be disrupted and some stages may be forced to drastically reduce their production rate or even to shut-down. The start-up of the process after these interruptions is typically problematic and requires a large expenditure of energy, increasing the environmental load of the mill. The rates of the various resources are wanted to be as steady as possible, as rough changes can cause quality deviations and undesired wear of equipment. Therefore, the variation that results from production cycles is conveyed as much as possible to the tank levels. Nonetheless, the system is subject to process variability (e.g. in production yield, required pulp mixture, etc.) and disturbances (e.g. paper breaks, equipment failures, etc.). Thus, it is important to keep some slacks in the intermediate buffers, in order to prevent under and overflows.

In this capital intensive industry in general, and in this plant in particular, the exploitation of existing capacity is a major concern. Therefore, the plant works on a 24/7 basis and backlog is tolerable. When accepting orders, managers use a base sequence of grades, where they can check the earliest available slot for that grade and hence commit to a due

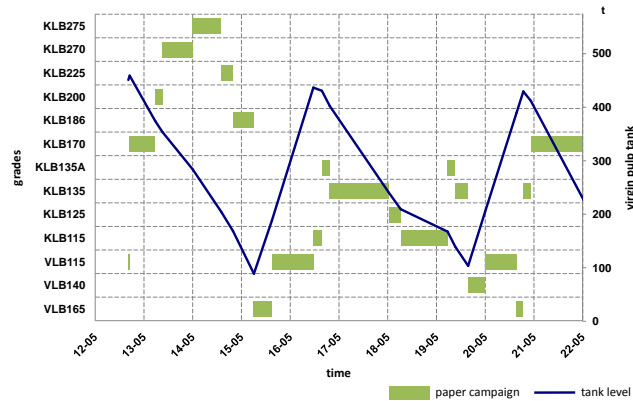


Figure 3.2: Example of a production plan – cycle of grades (left axis) and cycle of virgin pulp tank (right axis).

date (available-to-promise). The base sequence minimizes changeovers, since not only they consume time (i.e. capacity), but can also be highly disruptive to the system, breaking the production stability and causing wear to equipment. Still, at the operational level, the sequence has to be adapted according to the evolving state of the system.

The operational planning consists of determining the sequence and size of paper campaigns (where each campaign represents one of the 29 commercial grades) and all the production rates of the different resources for the following 10 to 15 days. This horizon is important to effectively check production cycles and the impact on every unit. An example of a production plan is illustrated in Figure 3.2. The cycle of grades is conveyed to a cycle of the virgin pulp tank level (as the digester's rate should be as smooth as possible). As VLB-type grades consume more recycled pulp, the tanks of virgin pulp increase their stock throughout the VLB campaigns. This stock is consumed during the majority of the KLB-type campaigns. The slacks of the tanks are kept to some desired limits that will not impact the feasibility of the subsequent plans.

This planning process is conducted by two main entities: Sales and Production departments. The former aims to fulfil the orders respecting due dates, whereas the latter is focused on operational costs and throughput maximization. As stated above, aligning these two main objectives is problematic. Therefore, when devising a plan, several interactions are needed between both departments. The lack of a holistic view and of an optimization-based DSS makes the generation of feasible plans a hard task and their quality is compromised. This is especially true in the presence of disturbances, scheduled maintenance or rush customer orders. The latter prevent the reels cutting from being planned for the same horizon. Indeed, one late order can completely spoil the cutting patterns. Therefore, managers choose to define those patterns just for the following 2 to 3 days.

3.3. Solution approach

Considering the aforementioned challenges to the production planning activity, we propose an optimization-based DSS. Our system supports decision making regarding the size and sequence of paper campaigns, the production rates for every unit and the fulfilment of individual orders (with different priority levels). The planning of cutting patterns is not included for two reasons: (i) companies already have software packages for that task; (ii) the planning horizon is radically different, as the cutting is only scheduled for a couple of days ahead; (iii) as there are buffer areas between the paper machine and the cutting reels, jumbos storage is allowed to some extent, enabling the decoupling of the two processes.

However, at this planning level it is important to integrate lot-sizing and scheduling. Indeed, the sequence dependent setups on the paper machine, with important costs related to production stability, must be taken into account when defining production plans. This is not possible in plain lot-sizing models. On the other hand, pure scheduling approaches are not appropriate either, since this (capital intensive) continuous production plant has high utilization ratios. Therefore, choosing the amounts to be produced and the respective sequence (leaving orders unmet or to be met later) is a key decision.

The literature on this industry has mainly focused on either lot-sizing (e.g. Rizk et al. (2008), Poltroniere et al. (2008)) or scheduling (e.g. Akkiraju et al. (2001), Correia et al. (2012)). Moreover, those works have not included the pulping and recovery stages. This is crucial in integrated plants due to the tightly integrated equipment and the existence of multiple and shifting bottlenecks. Santos and Almada-Lobo (2012) were the first to integrate those stages. The proposed model was based on the general lot-sizing and scheduling problem (GLSP), presented by Fleischmann and Meyr (1997), and considered sequence dependent setup times and costs. Figueira et al. (2013) then improved the resolution of the problem with a variable neighbourhood search.

GLSP is a small-bucket model (also known as time slot formulation, see Amorim et al. (2013)), it allows for at most one setup in each (micro) period. Having only one product in each period is a key aspect in our case, since it allows the accurate evaluation of mass balances. This is crucial when the intermediate tanks have tight capacities. In big-bucket models (also known as precedence-based), multiple products with different rates are produced in the same period, whereas the mass balances are only checked at the end of each period. Therefore, the tank limits may be violated within a period, while being respected at the beginning and at the end.

Contrary to other small-bucket models, GLSP combines continuous- and discrete-time grids. Continuous-time grids are important to accurately track the production of non-standard amounts, essential in this MTO system. Discrete-time grids allow assessing the fulfilment of demand (at discrete time points) during the planning horizon. This mixed grid thus makes the interface between Production and Sales requirements.

Other works in the literature, in both the operations research (OR) and the process systems engineering (PSE) communities, sought to combine the advantages of continuous and discrete-time grids. Maravelias (2005) and Westerlund et al. (2007) proposed models that use a discrete grid, but where campaigns do not need to start or end at the exact grid points. This aspect was also explored by the continuous setup lot-sizing problem (CSLP),

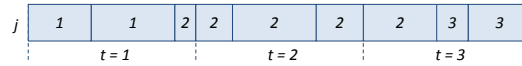


Figure 3.3: GLSP – each period t is divided into multiple continuous slots, where a grade j is produced.

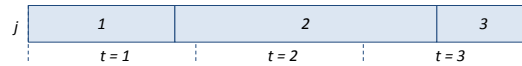


Figure 3.4: CGLSP – the discrete grid (of periods t) is crossed by continuous slots, where a grade j is produced.

presented by [Karmarkar and Schrage \(1985\)](#), and the proportional lot-sizing and scheduling problem (PLSP), proposed by [Drexler and Haase \(1995\)](#). However, in these models obtaining continuous amounts imply idle times and/or multiple products per period, which undermines the evaluation of intermediate tank levels.

Models based on GLSP have been extensively explored in the literature (e.g. [Lim and Karimi \(2003\)](#), [Erdirik-Dogan and Grossmann \(2008\)](#), [Castro et al. \(2009\)](#), [Amorim et al. \(2011\)](#)). Still, all these models divide each (macro) period in multiple slots (micro-periods). Thus, if a campaign crosses a period, it will need more than one slot (see Figure 3.3). This increases the difficulty in incorporating some practical constraints. For instance, in our case we need to fix the amounts of the first campaigns, as they were already programmed for the cutting stage. In those models we have to determine the number of slots for each of these campaigns. Since the production rates are variable, and will have to be adapted according to the state of the system, the appropriate number of slots can only be approximately estimated. This may compromise the feasibility of the plans generated. Other constraints include maximum and minimum lot sizes and a minimum distance between campaigns of the same grade.

[Karimi and McDonald \(1997\)](#) and [Camargo et al. \(2012\)](#) proposed other mixed grids formulations, where (continuous) production slots may cross (discrete) time periods (see Figure 3.4). This feature allows assigning only one slot to each campaign, and hence easily accommodating the aforementioned issues. The authors concluded however that these crossed grids models (here denoted by CGLSP) are harder to solve than the GLSP. There is thus a trade-off between modelling accuracy and performance. For the P&P industrial problem, the CGLSP was found to be the appropriate choice. In fact, the accuracy of plans is a critical issue. Moreover, the flexibility in incorporating the numerous practical constraints, avoiding specific heuristics for the allocation of slots, was crucial in this industrial implementation. In addition to the multi-stage environment and the sequence dependent setup times and costs modelled by [Santos and Almada-Lobo \(2012\)](#), our system comprises a set of features that are important for industrial practice, such as:

- fixing the size of certain campaigns (due to cutting programs);
- fixing the sequence of certain campaigns, critical when the production system is not

responding well to certain grades;

- giving an initial sequence of campaigns, which can then be improved;
- scheduling stoppages in the paper machine and digester (for maintenance purposes or in a disturbance situation);
- variable production rates for all the units, including the paper machine;
- production rates smoothness (to be performed in post-optimization, in order to keep model's linearity);
- different priority levels for the clients orders;
- demand beyond the planning horizon (important in low demand periods).

The system was designed to support both Sales and Production staff, weighting their conflicting objectives. Other works in the literature (e.g. Murthy et al. (1999)) proposed multi-objective approaches, where managers can evaluate different plans and choose the one that seems more appropriate. However, in our case this evaluation appears to be time consuming. Indeed, at the operational level managers do not have the time to properly evaluate different plans. Therefore, they prefer to define *a priori* the trade-offs between objectives, according to the company's policies, and then use them in every plan generation.

3.4. Mathematical model

We present the mathematical formulation in four main parts:

- continuous time slots and their intersection with the discrete time grid;
- production system, namely the P&P production;
- demand fulfilment, discriminated by period;
- objective function, with all the components of the production and marketing objectives;

An illustrative example is provided to help the understanding of the model. Two additional sections are then presented to foster the applicability of the model, namely:

- valid inequalities that help improving the tightness of the formulation;
- practice issues, such as smoothed production rates (essentially performed in post-optimization), undesired sequence of campaigns and stoppages.

3.4.1 Time slots

The production slots are of arbitrary lengths and are independent of the discrete time grid (they may cover one or more periods). A slot contains only one paper campaign and starts always with the changeover from the grade produced in the previous slot to the grade of the current slot. This continuous-time representation of the formulation avoids the time slot splitting, which will be of great advantage to incorporate in a straightforward manner industrial features, such as lower and upper bounds on the quantity produced. On the other hand, (external) demand is given for each (discrete) time period, and therefore there is a need to define the quantity produced in each time period. The following set of constraints allow for the coherence of this two-level time grid.

Indices, sets and parameters	
t	Index for period ($t \in [T] = \{1, \dots, T\}$)
s, u	Indices for slot ($s, u \in [S] = \{1, \dots, S\}$)
j, k	Indices for paper grades ($j, k \in [K] = \{1, \dots, K\}$)
cap	Period's capacity (in hours)
$hcap$	Planning horizon's capacity (in hours): $hcap = T \cdot cap$
$slots_{max}$	Maximum number of slots crossing one period
Decision variables	
S_s^{start}	Starting time of slot s
S_s^{end}	Ending time of slot s
U_{js}^{start}	Starting time of slot s for grade j
U_{js}^{end}	Ending time of slot s for grade j
E_{ts}	$\begin{cases} 1 & \text{if slot } s \text{ intersects period } t \\ 0 & \text{otherwise} \end{cases}$
Y_{js}	$\begin{cases} 1 & \text{if the paper machine is set up for grade } j \text{ in slot } s \\ 0 & \text{otherwise} \end{cases}$
N_s	Length of slot s (in hours)

The maximum number (S) of slots is given. The planning horizon starts with the first slot ($S_1^{start} = 0$) and ends with the last $S_S^{end} = hcap$. In case the slots are not all used, phantom slots can be moved to the end of the horizon as it will be shown later.

$$S_s^{start} = S_{s-1}^{end}, s \in [S] \setminus \{1\} \quad (3.1)$$

$$N_s = S_s^{end} - S_s^{start}, s \in [S] \quad (3.2)$$

$$U_{js}^{end} - hcap \cdot Y_{js} \leq U_{js}^{start} \leq U_{js}^{end}, j \in [K], s \in [S] \quad (3.3)$$

$$S_s^{start} - hcap \cdot (1 - Y_{js}) \leq U_{js}^{start} \leq S_s^{start} + hcap \cdot (1 - Y_{js}), j \in [K], s \in [S] \quad (3.4)$$

$$S_s^{end} - hcap \cdot (1 - Y_{js}) \leq U_{js}^{end} \leq S_s^{end} + hcap \cdot (1 - Y_{js}), j \in [K], s \in [S] \quad (3.5)$$

Constraints (3.1) forbid idleness by blocking the starting time of a slot to the ending time of the precedent one. The length of each slot is computed in (3.2). The production

time of each campaign is limited to the respective bucket – see (3.3)–(3.5). When these constraints are active ($Y_{js} = 1$), clearly $U_{js}^{start} = S_s^{start}$ and $U_{js}^{end} = S_s^{end}$.

Recall that the production resources have the slots independent of the time period's boundaries. Now, the intersection between slots and time periods is assured by variables E . Naturally, the first slot takes place in time period one ($E_{11} = 1$) and the last in the last period ($E_{TS} = 1$).

$$slots_{max} \cdot T \cdot (1 - E_{ts}) \geq \sum_{t'=1}^{t-1} \sum_{s'=s+1}^S E_{t's'}, t \in [T] \setminus \{1\}, s \in [S] \setminus \{S\} \quad (3.6)$$

$$E_{t-1,s} + E_{t+1,s} \leq E_{ts} + 1, t \in [T] \setminus \{1, T\}, s \in [S] \quad (3.7)$$

The proper assignment of slots to time periods is established by (3.6). Given that a slot s is assigned to period t , a subsequent slot can not be assigned to any precedent time period, which guarantees the order correctness of the slots. In case the slot spans from period $t - 1$ to period $t + 1$, then (3.7) ensures that period t is also crossed by the same slot.

3.4.2 Production system

Here we focus on the pulp plant and paper mill of the P&P manufacturing system (recall Figure 5.1).

Parameters	
$V_{min}^{dig} (V_{max}^{dig})$	Minimum (maximum) rate of the digester (in rpm)
α	conversion factor of digester's throughput (from rpm to tonnes per hour)
$I_{min}^{virg} (I_{max}^{virg})$	Minimum (maximum) level of virgin pulp stocked in the tanks (in tonnes)
f_j	Percentage of water in paper grade j
tl_j	Average trim loss (in percentage)
b_j^{virg}	Percentage of virgin pulp used in the production of paper grade j
b_{loss}^{virg}	Percentage of virgin pulp in the loss pulp
sl_{kj}	Paper lost in a changeover from grade k to j (in tonnes)
st_{kj}	Time lost in a changeover from grade k to j (in tonnes)
p_j^{min}	minimum time (at machine's maximum rate) to produce one tonne of paper grade j (in hours)
p_j^{max}	maximum time (at machine's minimum rate) to produce one tonne of paper grade j (in hours)
Decision variables	
X_s^{virg}	Production of virgin pulp in slot s (in tonnes)
O_s^{virg}	Amount of virgin pulp fed into the paper machine in slot s (in tonnes)
I_s^{virg}	Inventory of virgin pulp at the end of slot s (in tonnes)

X_s^{loss}	Production of loss pulp in slot s (in tonnes)
O_s^{loss}	Amount of loss pulp used in slot s (in tonnes)
X_{jts}	Production of paper grade j in period t and slot s (in tonnes)
Z_{kjs}	$\begin{cases} 1 & \text{if a changeover from grade } k \text{ to } j \text{ occurs at the beginning of slot } s \\ 0 & \text{otherwise} \end{cases}$

In what concerns the pulp production, the amount of virgin pulp produced by the digester in each slot is bounded by its speed limits, according to (3.8). Produced pulp is stocked in tanks and then fed into the paper machine (3.9). In (3.10), this stock is also limited to lower and upper bounds. The reader is referred to Santos and Almada-Lobo (2012) for a thorough analysis of the constraints of these two areas.

$$\alpha \cdot V_{min}^{dig} \cdot N_s \leq X_s^{virg} \leq \alpha \cdot V_{max}^{dig} \cdot N_s, \quad s \in [S] \quad (3.8)$$

$$X_s^{virg} + I_{s-1}^{virg} = O_s^{virg} + I_s^{virg}, \quad s \in [S] \quad (3.9)$$

$$I_{min}^{virg} \leq I_s^{virg} \leq I_{max}^{virg}, \quad s \in [S] \quad (3.10)$$

The requirements of the recycled pulp and recovery plant are similar to those that appear in the virgin pulp plant, and therefore will not be presented here (such as the minimum and maximum production of recycled pulp, liquors and steam, the inventory balance for recycled and loss pulps and liquors, and the inventory limits for recycled and loss pulps and liquors).

$$\sum_j Y_{js} = 1, \quad s \in [S] \quad (3.11)$$

$$\sum_k Z_{jks} = Y_{j,s-1}, \quad j \in [K], s \in [S] \quad (3.12)$$

$$\sum_k Z_{kjs} = Y_{js}, \quad j \in [K], s \in [S] \quad (3.13)$$

$$X_{jts} \leq cap \cdot Y_{js}, \quad j \in [K], t \in [T], s \in [S] \quad (3.14)$$

$$p_j^{min} \cdot X_{jts} \leq cap \cdot E_{ts}, \quad j \in [K], t \in [T], s \in [S] \quad (3.15)$$

$$\sum_j b_j^{virg} \cdot (1 - f_j) \cdot \left(\sum_t X_{jts} + \sum_k sl_{kj} \cdot Z_{kjs} \right) = O_s^{virg} + b_{loss}^{virg} \cdot O_s^{loss}, \quad s \in [S] \quad (3.16)$$

$$X_s^{loss} = \sum_j \sum_k (1 - f_j) \cdot sl_{kj} \cdot Z_{kjs} + \sum_j \sum_t (1 - f_j) \cdot tl_j \cdot X_{jts}, \quad s \in [S] \quad (3.17)$$

$$\sum_t p_j^{min} \cdot X_{jts} + \sum_k st_{kj} \cdot Z_{kjs} \leq U_{js}^{end} - U_{js}^{start}, \quad j \in [K], s \in [S] \quad (3.18)$$

$$\sum_t p_j^{max} \cdot X_{jts} + \sum_k st_{kj} \cdot Z_{kjs} \geq U_{js}^{end} - U_{js}^{start}, \quad j \in [K], s \in [S] \quad (3.19)$$

Regarding the paper production, from (3.11) at most one grade can be produced per slot.

Constraints (3.12) and (3.13) link the grade changeover variables to the setup state variables on the paper machine. A grade j is only produced in a given bucket in case the machine has been appropriately set up to (3.14). Naturally, variable X_{jts} only takes positive values in case slot s crosses period t , as ensured by equations (3.15). Note that the production quantity (X_{jts}) is described per period, i.e., the portion of the slot s that crosses period t is taken into account to define the quantity that meets the demand of that period. The virgin pulp used by the paper machine, defined in (3.16, comes from the virgin pulp stored in tanks together with (part of) the pulp that is recovered from the production losses due to the grade setup changeovers and trim sub-process – which are computed in (3.17).

Finally, according to (3.18)-(3.19) the time confined to the setup and production operations of each grade on the paper machine in each slot is given by the length of the slot, where the machine operates within its minimum and maximum speed limits. Note that in case grade j is not assigned to slot s , from (3.3), $U_{js}^{end} = U_{js}^{start}$ and therefore no operations take place. The definition of the consumption of recycled pulp is also modeled in a way similar to the consumption of the virgin pulp presented before, and it is not presented here.

3.4.3 Demand fulfilment

Demand is fulfilled at the end of the discrete periods. The following equations look at the intersection of the continuous production slots with those discrete periods.

Parameters	
d_{jt}	Demand for paper grade j in period t (in tonnes)
I_{j0}	Initial inventory of paper grade j (in tonnes)
IB_{j0}	Initial backlog of paper grade j (in tonnes)
Decision variables	
W_{jts}^{start}	Starting time of slot s for grade j in period t
W_{jts}^{end}	Ending time of slot s for grade j in period t
I_{jt}	Inventory of paper grade j at the end of period t (in tonnes)
IB_{jt}	Backlog of paper grade j at the end of period t (in tonnes)

$$p_j^{min} \cdot X_{jts} \leq W_{jts}^{end} - W_{jts}^{start}, j \in [K], t \in [T], s \in [S] \quad (3.20)$$

$$p_j^{max} \cdot X_{jts} \geq W_{jts}^{end} - W_{jts}^{start}, j \in [K], t \in [T], s \in [S] \quad (3.21)$$

$$W_{jts}^{end} \leq \min(t \cdot cap; U_{js}^{end} + hcap \cdot (1 - E_{ts})), j \in [K], t \in [T], s \in [S] \quad (3.22)$$

$$W_{jts}^{start} \geq \max\left((t-1) \cdot cap; U_{js}^{start} + \sum_k st_{kj} \cdot Z_{kjs} - hcap \cdot (1 - E_{ts})\right), j \in [K], t \in [T], s \in [S] \quad (3.23)$$

$$\sum_t (W_{jts}^{end} - W_{jts}^{start}) + \sum_k st_{kj} \cdot Z_{kjs} = U_{js}^{end} - U_{js}^{start}, j \in [K], s \in [S] \quad (3.24)$$

$$\sum_s (1 - tl_j) \cdot X_{jts} + I_{j,t-1} - IB_{j,t-1} = d_{jt} + I_{jt} - IB_{jt}, j \in [K], t \in [T] \quad (3.25)$$

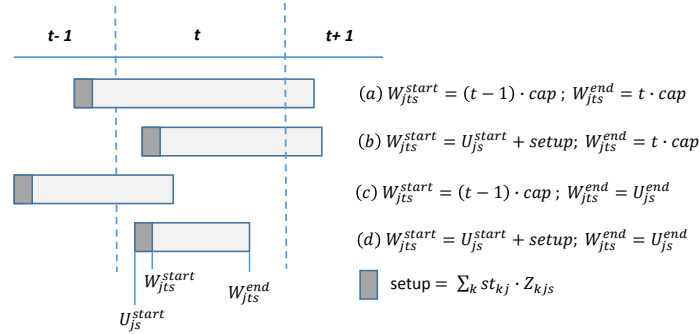


Figure 3.5: Production window of a slot definition in each time period.

Variables W_{jts}^{start} and W_{jts}^{end} define the respective production windows, which in turn limit the production amounts (X_{jts}) – see requirements (3.20)–(3.21). These variables are upper bounded by (3.22) and lower bounded by (3.23). Figure 3.5 provides some examples to explain how these constraints work and to show different production windows definitions. In case 3.5(a), the slot cross entirely period t , and therefore the production window bounds in period t match that period's boundaries. In case the slot ends in the middle of time period t , cases 3.5(c) and (d), the ending date of the production window is the same of the ending time of the slot. Note that the production window is contained within the slot, as it discards the setup time. In case slot s does not intersect period t (i.e. $E_{st} = 0$), requirements (3.22) and (3.23) are non-active. The relation between the production window of grade j in slot s and the slot length is established in (3.24) – the slot contains the production window and the setup times consumed in a grade changeover. Then, the fulfilment of demand is obtained by (3.25), which allows backlogging orders to be met in later periods.

3.4.4 Objective function

Given that on the one hand the system works with an MTO policy and on the other hand the plant operates continuously on a 24/7 basis (no idle or extra times), orders may backlog at some points in time. The goal of the operational production planning and scheduling is then to minimize that backlog at the minimum possible cost. Operational costs include grade setup, production and holding costs. Production costs are mainly related to diluting the costs of the capital intensive equipment, which is achieved by maximizing its utilization (sometimes producing beyond fixed orders). The setup costs, which are sequence dependent, concern not only the time and product lost in that process, but also the wear imputed to equipment. Equation (5.17) represents a cost function that aggregates these main objectives.

Parameters	
bc	Cost of backlogging one tonne of paper per period
hc	Cost of holding one tonne of paper in stock per period
sc_{kj}	Setup cost of changing from grade j to k
po	Benefit (negative cost) of producing paper

$$\min \sum_j \sum_t bc \cdot IB_{jt} + \sum_j \sum_t hc \cdot I_{jt} + \sum_j \sum_k \sum_s sc_{kj} \cdot Z_{kjs} - \sum_j \sum_t \sum_s po \cdot X_{jts} \quad (3.26)$$

Other objectives are considered in the complete model. The driver of the P&P company is not only to produce paper, but also to generate and sell energy based on the steam produced (which is somehow proportional to the throughput of the digester). Therefore, any plan also aims at maximizing the production of steam. Further terms of this objective function are revealed in Subsection 3.4.6, where some practical issues are addressed.

3.4.5 Valid inequalities

A few sets of valid inequalities are hereby introduced to strengthen the aforementioned mathematical formulation.

$$\sum_s E_{ts} \geq 1, t \in [T] \quad (3.27)$$

$$\sum_t E_{ts} \geq 1, s \in [S] \quad (3.28)$$

$$\sum_s E_{ts} \leq slots_{max}, t \in [T] \setminus \{T\} \quad (3.29)$$

$$Z_{jjs} \leq 1 - Z_{jku}, j, k \in [K] : k \neq j, s, u \in [S] : u > s \quad (3.30)$$

$$Z_{jjs} \leq E_{Ts}, j \in [K], s \in [S] \quad (3.31)$$

Each time period is intersected by at least one slot, as pointed out by (3.27). Clearly, each slot has to be assigned to at least one time period (3.28). The maximum number of slots per period is reinforced by (3.29). Two consecutive non-empty slots produce different grades. In case there is no need to use all the pre-defined slots, the null slots should be placed at the end of the horizon. Requirements (3.30) and (3.31) force the phantom grade changeovers (Z_{jjs}) to be moved to the end.

3.4.6 Practical constraints

In order to be used in practice, the model presented above needs to incorporate real-world extensions. Some of the new features are related to customers to maximize their satisfaction, other have to do with the hurdles of managing the resources in the smoothest way

possible.

3.4.6.1 Sales requirements

To consider different types of orders with different priority levels it is necessary to disaggregate production, inventory and backlog variables. In that way it is possible to introduce different penalties for backlogging specific orders. The suffix *norm* of the new variables refers to the normal priority level, whereas *ship* to ship orders and *prio* to the highest priority orders. Given that multiple ships can be scheduled within the planning horizon, an index $m \in [M]$ is introduced for them. For instance, variable X_{jtm}^{ship} denotes the amount of grade j produced in period t to be distributed in ship m . The latest production time of an order to meet the departure of ship m is given by dd_m .

$$I_{j0} = I_{j0}^{norm} + I_{j0}^{prio} + \sum_m I_{jm}^{ship}, j \in [K] \quad (3.32)$$

$$\sum_s (1 - tl_j) \cdot X_{jts} = X_{jt}^{norm} + X_{jt}^{prio} + \sum_m X_{jtm}^{ship}, j \in [K], t \in [T] \quad (3.33)$$

$$\sum_{t=1}^{dd_m} X_{jtm}^{ship} + I_{jm}^{ship} - IB_{jm}^{ship,ini} = d_{jm}^{ship} - IB_{jm}^{ship}, j \in [K], m \in [M] \quad (3.34)$$

$$I_{jT}^{norm} - IB_{jT}^{norm} = d_{j,T+1}^{norm} + I_{j,T+1}^{norm} - IB_{j,T+1}^{norm}, j \in [K] \quad (3.35)$$

The disaggregation of the initial inventory is performed by (3.32), where it is allocated to normal, priority or ship orders (the three priority levels described in Section 3.2). Constraints (3.33) disaggregate production. Note however that there is also aggregation, regarding the different slots that cross the period, since at this point we just need to know the completed amounts at the end of each period.

The demand fulfilment equations for normal and priority orders are similar to constraints (3.25), hence they are not detailed here. On the other hand, for ships they are slightly different – see (3.34). Indeed, each ship m has a single due date (dd_m). Thus, demand fulfilment is not checked in each period, but only in the ship's due date. Therefore, production is summed from the first period until the due date and no inventory is left after that since there is no other chance to fulfil that demand.

Finally, in (3.35) the fulfilment of demand beyond the planning horizon is verified. Given that the machine is never idle and paper is not made to stock, if all the demand within the horizon has been met, future demand should be considered. All this demand is then aggregated in an additional term corresponding to period $T + 1$. Naturally, not fulfilling this demand will have a lower penalty in the objective function.

3.4.6.2 Paper campaigns

Here, we describe some constraints related to operational issues in executing paper campaigns. Parameters N_{min} and N_{max} define the lower and upper bounds for the length (in time units) of each grade campaign, respectively. Furthermore, L_{min}^{VLB} refers to the minimum

production amount of each *VLB*-type campaign, $|K_{VLB}|$ to the number of *VLB* grades, and $dist_{min}$ to the minimum distance between campaigns of the same grade.

$$Z_{jks} = 0, \quad s \in [S], j \in [K], k \in [K_j^{na}] \quad (3.36)$$

$$N_s \leq N_{max} \cdot \sum_k \sum_{j \neq k} Z_{kjs}, \quad s \in [S] \quad (3.37)$$

$$N_s \geq N_{min} - N_{max} \cdot \sum_j Z_{jjs}, \quad s \in [S] \quad (3.38)$$

$$(1 - tl_j) \cdot \sum_t X_{jts} \geq L_{min} \cdot (Y_{js} - Z_{jjs}), \quad j \in [K], s \in [S] \quad (3.39)$$

$$\sum_{j \in [K_{VLB}]} \sum_t \sum_{s'=s}^{s+|K_{VLB}|-1} X_{jts'} \geq L_{min}^{VLB} \cdot \sum_{k \notin [K_{VLB}]} \sum_{j \in [K_{VLB}]} Z_{kjs}, \quad j \in [K], s \in [S] : s \leq S - |K_{VLB}| + 1 \quad (3.40)$$

$$\sum_{s'=s}^{s+dist_{min}} (Y_{js'} - Z_{jjs'}) \leq 1 + viol^{dist}, \quad j \in [K], s \in [S] : s \leq S - dist_{min} \quad (3.41)$$

The first constraints regard sequences that cannot be executed. In our case, quality “A” grades should not be produced after basic ones. Indeed, if they do not meet the required quality level, but there is a basic quality campaign following, they can still be used as basic quality products and the “A” quality has another opportunity to be produced. Therefore, for each grade j , there is a list of forbidden grades $[K_j^{na}]$, which correspond to the better quality versions (if any).

Constraints (3.37) and (3.38) define maximum and minimum lengths for campaigns, which should be respected for the sake of plant’s stability. Note that for all fictitious changeovers (Z_{jjs}), the length is zero. Minimums are also defined in term of the amounts produced (L_{min} and L_{min}^{VLB}) – see (3.39) and (3.40). The former applies a minimum lot to individual campaigns, whereas the latter does that to a set of campaigns of the same type (*VLB* grades).

Finally, (3.41) forces a minimum distance ($dist_{min}$) between campaigns of the same grade. The violation of this soft constraint ($viol^{dist}$) is penalized in the objective function. This issue is somehow avoided with setup costs. However, these constraints provide a more explicit penalization of producing the same grade twice in a short period of time. This type of constraints help better tailoring the plans to meet industrial requirements, which would otherwise need non-linear cost functions.

Figure 3.6 provides an example of a plan that does not comply with the operational constraints here described. The corrected version of the plan is that represented in Figure 3.2. Note that the first part is fixed, since those campaigns were already programmed in the cutting stage. The first campaign, which seems not to comply with the minimum lot size, is indeed in progress. The above constraints are thus only applied after the last fixed campaigned.

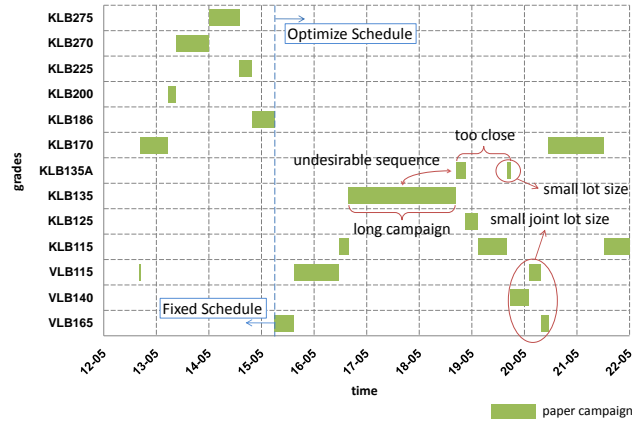


Figure 3.6: Example of a plan not following the operational requirements.

3.4.6.3 Production rates

The production rates of all resources should be as smooth as possible to avoid large energy expenditures and wear of the equipment. Here we illustrate how the digester's rates can be smoothed in the optimization process. Variables V_s^{dig} introduce the speed of the digester in slot s .

$$V_s^{dig} - V_{s-1}^{dig} = \delta_s^{dig+} - \delta_s^{dig-}, s \in [S] \quad (3.42)$$

$$X_s^{virg} = \alpha \cdot V_s^{dig} \cdot N_s, s \in [S] \quad (3.43)$$

The rates variations are assessed with equations (3.42). Minimizing the variables δ_s^{dig+} and δ_s^{dig-} in the objective function will smooth rates variations. The former variable will then assess the rates' increase, whereas the latter will assess their decrease. These equations require an explicit variable for the digester's rate (V_s^{dig}), as opposed to the implicit definition in constraints (3.8). That results in a non-linear formulation, since the rate variable is multiplied by the slot's length – see (3.43). Therefore, in order to avoid the complexity of the non-linear formulation, these constraints are only added in a post-optimization step where the slots are fixed.

3.4.6.4 Recommended tank levels

In practice, the stock should not approach the physical limits of the tank, since it may trigger operational difficulties and would leave the system more vulnerable to disturbances. For those reasons, recommended levels (below the capacity upper limit and above than the lower limit) are established. The stock may in some points in time range between the desirable and physical limits, but the violation from the recommend levels is penalized in the objective function. The stock of virgin pulp in the tank in slot s is now given by $I_s^{virg} + I_s^{virg,rec+} - I_s^{virg,rec-}$, where variables $I_s^{virg,rec+}$ denote the amount of stock above the

recommended upper limit, and $I_s^{virg,rec-}$ the amount of stock below the recommended lower limit. In addition, parameter I_{slack-}^{virg} (I_{slack+}^{virg}) refers to the slack between the upper (lower) recommended level and the upper (lower) physical limit.

Equations (3.9)–(3.10) should then be replaced by the following:

$$I_{min}^{virg} + I_{slack-}^{virg} \leq I_s^{virg} \leq I_{max}^{virg} - I_{slack+}^{virg}, \quad s \in [S] \quad (3.44)$$

$$I_{min}^{virg} \leq I_s^{virg} + I_s^{virg,rec+} - I_s^{virg,rec-} \leq I_{max}^{virg}, \quad s \in [S] \quad (3.45)$$

$$X_s^{virg} + I_{s-1}^{virg} + I_{s-1}^{virg,rec+} - I_{s-1}^{virg,rec-} = O_s^{virg} + I_s^{virg} + I_s^{virg,rec+} - I_s^{virg,rec-}, \quad s \in [S] \quad (3.46)$$

Equations (3.44) force the regular variable (I_s^{virg}) to respect the recommended levels, while extra level variables ($I_s^{virg,rec+}$ and $I_s^{virg,rec-}$) are included in the physical limits verification and material balance equations – constraints (3.45) and (3.46), respectively.

3.4.6.5 Fixed schedule

A critical feature of the model, and one of the most important reasons that led to the choice of CGLSP as the basic formulation, is the possibility of fixing production amounts without the need of heuristically define the number of slots needed for each campaign. Indeed, as slots are able to cross periods, only one slot is required per campaign. Therefore, both the sequence of campaigns and their lot sizes can be easily fixed. The former is a simple assignment ($Y_{js} = 1$, for the respective grade and slot), while the latter is represented in constraints (3.47). These constraints are essential to freeze campaigns in the DSS that have already been programmed in the cutting stage, i.e. have cutting patterns optimized for them.

$$\sum_t X_{jts} = X_s^{fix}, \quad s \in [S] < prodS, j \in [K] : Y_{js} = 1 \quad (3.47)$$

X_s^{fix} is the amount to be produced in slot s and $prodS$ is the number of slots with fixed production. This latter parameter has to be taken into account in some of the constraints previously presented.

3.4.6.6 Stoppages

Finally, in this last subsection we show the way stoppages were modelled. These stoppages can occur in different production resources, such as the digester and the paper machine. We consider all the stoppages' starting and ending times as parameters $stop_r^{start}$ and $stop_r^{end}$, where r is the index of the stoppage. Then, we use binary variables P_{rs}^{start} and P_{rs}^{end} to state if stoppage r starts at the beginning or ends at the end of slot s .

$$stop_r^{start} - hcap \cdot (1 - P_{rs}^{start}) \leq S_s^{start} \leq stop_r^{start} + hcap \cdot (1 - P_{rs}^{start}), r \in [R], s \in [S] \quad (3.48)$$

$$stop_r^{end} - hcap \cdot (1 - P_{rs}^{end}) \leq S_s^{end} \leq stop_r^{end} + hcap \cdot (1 - P_{rs}^{end}), r \in [R], s \in [S] \quad (3.49)$$

$$\sum_s P_{rs}^{start} = 1, r \in [R] \quad (3.50)$$

$$\sum_s P_{rs}^{end} = 1, r \in [R] \quad (3.51)$$

$$\sum_{s'=1}^s P_{rs'}^{start} \geq \sum_{s'=1}^s P_{rs'}^{end}, r \in [R], s \in [S] \quad (3.52)$$

Constraints (3.48) and (3.49) align the starting and ending times of slots with the starting and ending times of stoppages, respectively. Then, (3.50) and (3.51) ensure that only one slot is aligned with the start and/or the end of each stoppage. The last constraints forbid the end of a stoppage to occur before its start.

Variables P_{rs}^{start} and P_{rs}^{end} are then used to constraint the output of the corresponding resources. In the case of the digester, for instance, equations (3.8) would include these variables to enable or disable the maximum and minimum production rates.

3.5. Solution method

The solution strategy followed is based on a heuristic solution of the problem's mathematical formulation. The reasons behind this choice come both from practical implications and algorithmic aspects. First and foremost the large scale and complexity of the model described in the previous section for a regular real-world instance prohibited the use of a commercial solver on the complete model formulation given its difficulty in even finding a feasible solution in the time limit imposed by the DSS usage. The model suffers from its computational intractability especially when the number of slots defined increases. Second, heuristics based on mathematical programming techniques, also known as matheuristics, are a powerful framework to explore the problem's structure and take advantage of today's computational and commercial solvers power. Matheuristics trade-off the solution quality obtained by solving the complete model formulation in a commercial solver with the solution search efficiency of heuristics and meta-heuristics. Moreover, these algorithms yield substantial advantages when compared to traditional heuristics, as they require less parameters and consequently a much lower effort in parameter tuning, and can adapt to changes in the mathematical formulation with limited or zero adjustments. Matheuristics often provide quasi-optimal solutions for a variety of problems and for the majority of the companies having a very good solution is sufficient as optimality can have a marginal improve for a high additional effort.

The matheuristic designed for the purpose of generating good quality solutions for this

complex problem has three main phases: initial solution, forward-pass and neighborhood search. All the phases decompose the problem or reduce its dimension by working on the set \mathcal{Y} defined by the variables Y_{js} of the original problem. The overall procedure is illustrated in Figure 3.7.

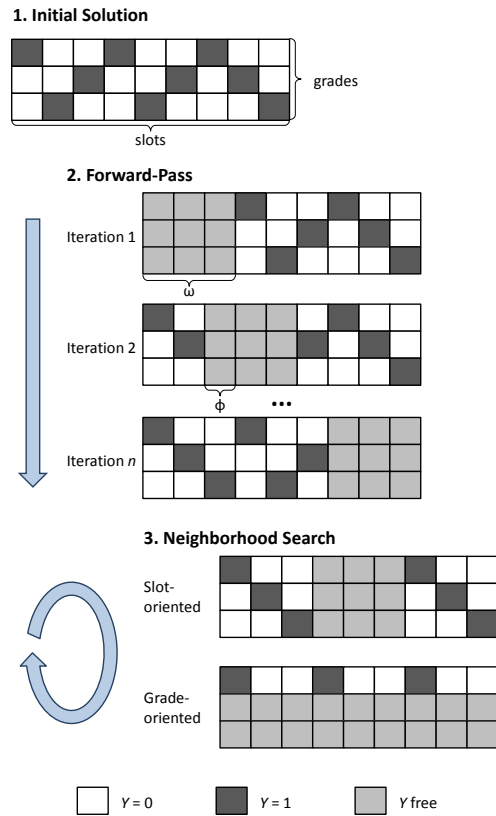


Figure 3.7: Solution method consisting of three phases: initial solution, forward-pass and neighborhood search. A sub-MIP is solved in each iteration.

In the first phase the goal is to generate an initial feasible solution to the problem. To this aim the variables on set \mathcal{Y} are fixed in the mathematical formulation either to zero or as following: an initial sequence provided by a company’s manager or a standard (pre-defined) sequence considering the current grade at the paper machine. The value for the remaining variables in the problem is obtained by solving the resulting sub-MIP.

The idea of the second phase is to improve the quality of the initial solution using a forward pass over the continuous time slots. At each iteration of the second phase a total of ω adjacent slots are re-optimized in a sub-MIP were their corresponding Y_{js} are set as binary while the remaining Y_{js} variables keep their current value. We start at the beginning of the planning horizon and re-optimize the solution corresponding to the first ω slots. Between iterations the selection of the slots also uses ϕ as the number of overlapping slots allowing for a smother schedule. Thus, in next iteration the solution for the first $\omega - \phi$ slots is fixed and we re-optimize the flowing ω slots. This process is repeated until the final slot

in reached.

The third phase is a neighborhood search that iteratively decomposes the overall problem. Two neighborhoods were defined: slot and grade oriented. The slot oriented neighborhood is similar to the second phase where all the Y_{js} variables associated with a given sub-set of slots are “free” in the model formulation keeping the other slots unchanged. The main difference relies on the fact that the re-optimization step does not follow a forward pass search. The grade oriented neighborhood optimizes the variables Y_{js} of a given sub-set of grades over the entire planning horizon. The choice of the slots and grades is guided by a procedure which uses the frequency and recency of selection to bias the sub-sets. The more frequent or recently a slot or grade has been selected the less probability has to be part of the sub-set to re-optimize. The exploration of a neighborhood corresponds to the sub-set creation and model re-optimization being performed until a maximum number of consecutive iterations without improvement has been reached. The neighborhood search alternates between the two neighborhoods stopping when both are unable to yield any improvement or the user defined time limit is reached.

3.6. Decision support system

This section starts by describing the system architecture and how the optimization component is integrated in that framework. Then, the focus is shifted to the system’s usage, where the features and information relevant to managers are detailed. In particular, specific activation and deactivation rules are discriminated, as well as key performance indicators (KPIs) of the plant.

3.6.1 System architecture

The DSS was built following a typical setup of full-fledged “Software as a Service” (SaaS) applications. SaaS applications are centrally hosted and by porting this feature into a service-oriented DSS, we address many problems related to the maintenance and evolution of software, being especially helpful when phasing out older versions (Gold et al., 2004). Updates and bug fixes are made available in a transparent and immediate way, requiring no further action from the user.

The architecture of this SaaS DSS is comprised of three main layers: the Data Layer, the Optimization Layer and the Web Presentation Layer. The layers communicate and work in a seamless way. Also, the DSS infrastructure interacts with the business ERP and related systems, following the best practices in the field (Harjunkoski et al., 2014). Figure 3.8 gives an overview of the system’s architecture.

The Data Layer component is in control of establishing the necessary connections with third-party systems, namely with the manufacturing execution system (MES) to retrieve information about the shop floor; and with the enterprise resource planning (ERP), that contains typical customer and order management figures. The connection is established over an on-demand VPN link to the data warehouse, after which data is retrieved and mapped to a local data model schema stored in an SQL database (SQL Server).

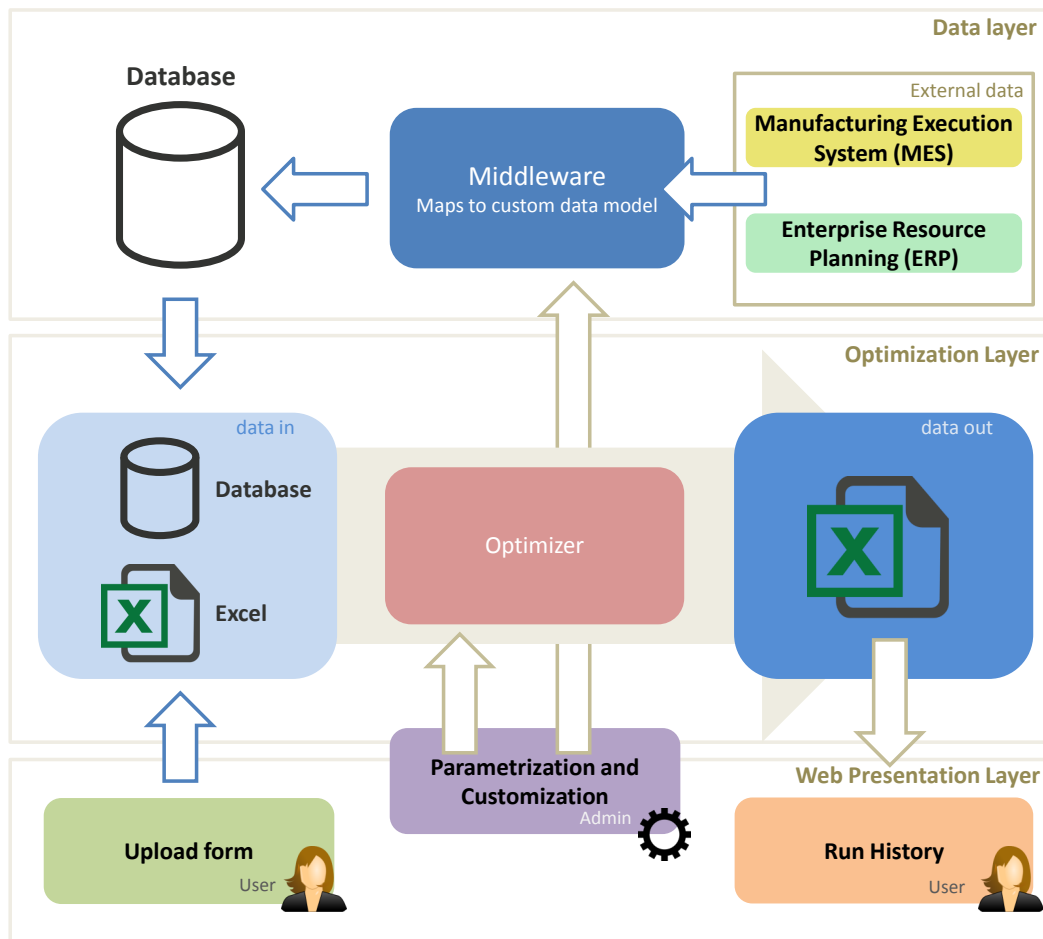


Figure 3.8: Overview of the DSS architecture, composed by three main layers: Data, Optimization and Web Presentation.

Data retrieved from the external systems is then made available to the Optimization Layer, providing critical input parameters to the optimization algorithm. Optionally, data can also be provided through an Excel spreadsheet. Excel is still a popular and ubiquitous tool for assembling data from several sources and for assisting in such an iterative and time-consuming process (Harjunkski et al., 2014), despite being potentially more error-prone since users can manipulate it freely. Regardless of the input source chosen, the data is fed to the Optimizer, which will then generate a solution and provide it as an Excel spreadsheet file.

The optimizer block (illustrated in Figure 3.9) generates a production plan by executing the optimization algorithm described in the previous section, followed by a post-optimization phase, where the slots' lengths are fixed and the production rates are smoothed (cf. Subsection 3.4.6.3). The overall optimization process is wrapped by pre- and post-processing stages. In the former the size of the mathematical model is reduced, by removing paper grades with no demand or backlog to meet. In addition, orders are aggregated

by grade and priority level to match the model's parameters. After the optimization the production amounts have to be disaggregated again, in order to check the fulfilment of individual orders, and KPIs have to be computed. The allocation of production to orders is achieved by applying the "earliest due date" rule.

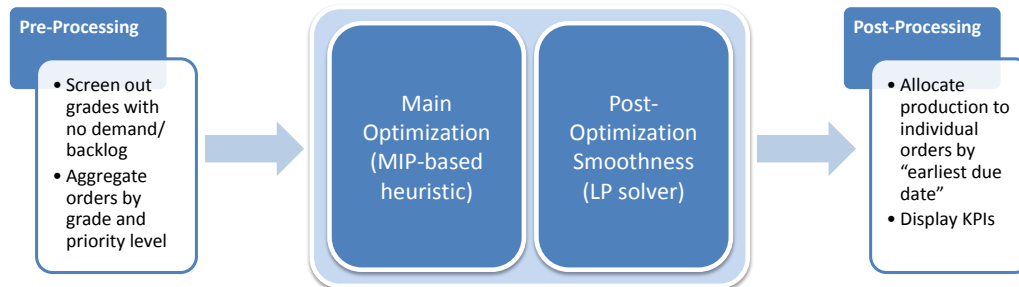


Figure 3.9: The optimizer component includes the main and post-optimizations, as well as pre- and post-processing stages.

None of the layers is directly accessible by the operator of the DSS. Instead, the user interacts with the system through the Web Presentation Layer, where he can upload Excel files to be later used by the Optimization Layer; consult the run history; download output files; or check the log files. Moreover, advanced users with administrator privileges will also be able to change certain configuration parameters that may interfere directly with the optimizer. In the limit, the web interface could be used only as a gate to the algorithm, a situation in which the user would only upload the Excel input file and then download the Excel output file generated. However, the full potential of the web interface comes with the important information directly retrieved from the ERP systems along with the tools developed and incorporated in the DSS to assist in the analysis. One of such tools is an advanced table editor (*Handonstable*) that allows for in-place editing and direct copy-paste from and to Excel. The main advantage being that critical business logic constraints and error-checking can be implemented at the interface level, preventing potential mistakes from propagating to the Optimization Layer or to be stored in the local database.

3.6.2 System's usage

From a user perspective, our DSS assists both production planners and production managers in obtaining the best production plans for a short-term planning horizon. The process to obtain such plans is presented in Figure 3.10. As already mentioned, to start the user needs to select which type of data to use – Excel or database.

By choosing to use the database, the user further needs to select a profile (production planner or production manager) and he may also fine-tune some parameters. This parametrization is made directly on the system's interface, which is divided in two main sections. One displays the main steps to generate a plan and comprises only the frequently changed parameters, in order to expedite the process (Figure 3.11). The other section contains more structural parameters, such as capacities and processing times, which are more

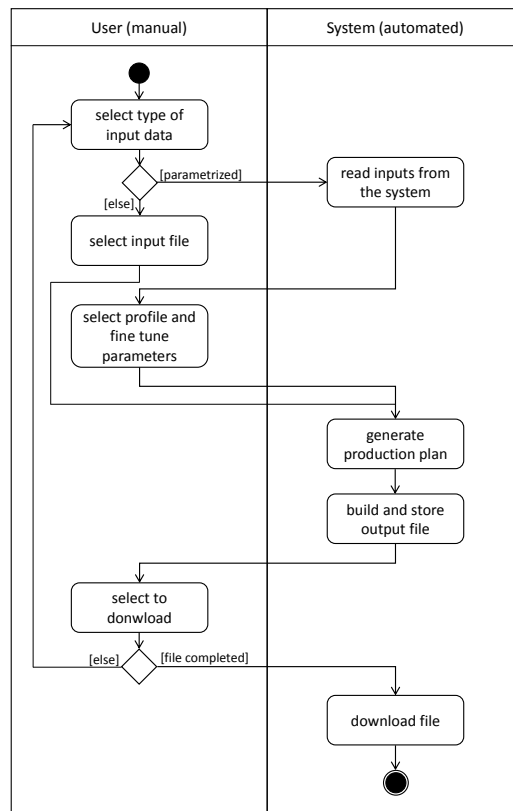


Figure 3.10: Process of generating a production plan as an activity diagram. The activities on the right-hand side are automated in the decision Support system, and the actions on the left-hand side require an input by the user.

stable and hence accessed only sporadically (Figure 3.12). The advanced editor tool for plan generation is particularly interesting for listing and customizing the orders (e.g. assigning priorities) and checking production quantities on the spot, as well as fixing the initial part of the plan (essential for rescheduling purposes) and scheduling stoppages (cf. Subsection 3.4.6.6). In addition, a variety of rules can be activated or deactivated, depending on the user's judgement of the current state of the plant. These options include:

- distinct priority levels (cf. Subsection 3.4.6.1);
- required quality sequence (cf. Subsection 3.4.6.2)
- maximum and minimum lot sizes (cf. Subsection 3.4.6.2);
- maximum and minimum campaigns' lengths (cf. Subsection 3.4.6.2);
- minimum distance between campaigns (cf. Subsection 3.4.6.2);
- production rates smoothness (cf. Subsection 3.4.6.3);

- initial backlog clearance (imposing the fulfilment of the initial backlog orders until the end of the planning horizon).

STP Planning

New Process

Please choose your profile: Planner Execute Process

Planning start date: 30-10-2014 15:55

Orders | Ships | **Plan** | Other settings | Uniformance

Paper Machine

Grades	Production (Tons)	Start time (estimated)	End time (estimated)	Optimizer
KLB225	47.44	30-10-2014 15:55	30-10-2014 16:53	✓
KLB200A	56.78	30-10-2014 16:53	30-10-2014 18:16	✓
KLB200	434	30-10-2014 18:16	31-10-2014 03:11	✓
KLB170	851.91	31-10-2014 03:11	31-10-2014 20:56	✓
KLB125	324.12	31-10-2014 20:56	01-11-2014 05:03	✓
KLB115	401.61	01-11-2014 05:03	01-11-2014 15:50	✓
VLB165	436.65	01-11-2014 15:50	02-11-2014 01:22	✓
VLB115	460.53	02-11-2014 01:22	02-11-2014 13:36	✓
VLB140	180.42	02-11-2014 13:36	02-11-2014 17:33	✓

Fixed sequence

Last update to Optimizer 17-09-2014 00:22:20 Update

Figure 3.11: Web interface of the DSS for generating a production plan (only frequently changed parameters are displayed).

When finishing the setup of the input data, the system starts to generate the production plan by running the Optimizer, which includes the MIP-based heuristic (cf. Section 5.4). After running this module, the system builds and stores the output excel file that will serve as basis for the decision process. Figure 3.13 is an example of an output. The excel output file allows some adjustment of the resulting schedule to address unsolved business/production issues or perform *what-if* analysis. This spreadsheet has different sheets with Gantt charts depicting the sequence of operations across each resource, trend charts displaying the forecasted evolution of inventories of intermediate and final products; and a comprehensive reporting on order fulfilment with expected dates of delivery, as well relevant KPIs such as average and final backlog of orders, equipment utilization ratios and total production output. At the end, if the user wants to perform some sensitivity analysis he may run another production plan or he may download the output file to his computer.

The system's usage described can be performed in three main contexts:

- periodic planning – the managers typically determine an optimized plan on a weekly basis and revise it daily;

- reactive scheduling – in the presence of a disturbance situation which causes a serious disruption to the original plan, recover from that disturbance using an appropriate sequence of campaigns (sometimes not obvious to plant managers);
- *what-if*-analysis – supporting planning in hierarchically upper levels by checking the effectiveness of practical rules, expanding storage and production capacities, changing weights of different objectives and scheduling maintenance.

3.7. Results

In this section we start by looking at the usage of the system during a period of four months and then delve into one plan generated by the system and its differences to the manually generated counterpart. In the end, a plan considering a long stoppage is analysed.

3.7.1 Overall results of the system's usage

Production planners and managers are not willing to wait more than 15 to 20 minutes for an optimized plan in normal conditions. When facing an operational issue, such as a disturbance, that time can be reduced to 5 to 10 minutes. With that time limit, state-of-the-art solvers like Cplex are not even able to find a solution for a regular instance. The focus here is thus much more on feasibility and improvement over current practice than it is on optimality. For that reason, a considerable amount of constraints are relaxed (and put as soft constraints) so that our constructive heuristic is always able to find an initial solution. Then, the MIP-based heuristic improves that solution to a point that it satisfies plant's managers.

Our system was made available online for the company to use. Various iterations were performed with the users in order to fine tune all the parameters and introduce additional constraints to the mathematical model, such that the resulting plans meet their requirements. Table 3.2 contains information about 22 runs executed during four months after those iterations. In spite of the total number of grades being 29, the number of grades with demand or backlog to satisfy at each moment is reduced to 16/17. This shows the importance of the pre-processing steps described in the previous section. The maximum number of campaigns (i.e. slots) is set to 35 (for an horizon of 15 days) and the model can use them all or leave some empty. The fixed (frozen) campaigns are dependent on what was already scheduled in the cutting stage and additional considerations managers might want to take.

The table reports the improvements achieved over the initial solution in runs executed by the planners. We can see that those values are highly variable. Essentially, they will depend on whether the initial solution is already good or not, and on the time limit given, which is defined by the user in every run. Nevertheless, an average improvement of 34% was obtained, with an average running time of 11 minutes, which shows the ability of the heuristic in correcting and improving production plans in short periods of time.

To validate the quality of those plans, they need to be compared to those manually generated by the company. In this regard, the system can also help. Indeed, the system can be used not only to optimize a plan from scratch, but also to test certain schedules by fixing

Table 3.2: Company's usage of the system from June to September 2014.

#	Date	Grades	Fixed campaigns	Total campaigns	Improv over ini sol	Time (min)
1	09/06/2014	17	4	35	75%	13
2	17/06/2014	17	7	34	81%	12
3	19/06/2014	17	9	28	26%	8
4	20/06/2014	17	8	33	51%	10
5	20/06/2014	17	20	27	5%	4
6	27/06/2014	17	10	30	29%	10
7	04/07/2014	17	14	28	0%	7
8	09/07/2014	16	7	35	4%	10
9	14/07/2014	16	3	31	1%	18
10	18/07/2014	17	4	35	45%	12
11	18/07/2014	17	27	28	3%	2
12	22/07/2014	17	9	32	8%	11
13	24/07/2014	17	4	30	6%	12
14	24/07/2014	17	7	31	7%	15
15	24/07/2014	17	13	29	63%	9
16	29/07/2014	17	5	30	87%	18
17	06/08/2014	17	5	30	93%	14
18	06/08/2014	16	5	32	46%	16
19	27/08/2014	17	9	33	55%	11
20	03/09/2014	17	7	29	6%	10
21	15/09/2014	16	6	31	18%	15
22	25/09/2014	17	9	29	48%	14
Avg	-	17	9	31	34%	11

the sequence and amounts of paper campaigns. The table clearly shows that the company has applied these two usages, comparing the output of both (e.g. runs 4 and 5, with a very different number of fixed campaigns - run 4 refers to the optimum plan and run 5 to the so-called manual solution). When fixing most campaigns, the algorithm is naturally much faster to finish. However, that should not be a reason for not letting the system optimize the plans, since just providing the initial sequence guarantees a good solution in a few minutes, in case the sequence is good, and gives the opportunity to improve on that solution.

The plan of run 4 (taking full advantage of the DSS) outperforms that of run 5 (manual plan) by 59% of the optimized. Regarding runs 10 (optimized one) and 11 (manual), that improvement was of 71%. These values are unreasonably high since some soft constraints were violated in the company's plans, whereas our system has addressed them. A more insightful comparison is performed in the next subsection, where we look at the schedules and the main KPIs.

3.7.2 Optimized plan vs. manual plan – an example

Two production plans, one optimized by the system and the other manually defined (although still generated by the system), are now examined. Figures 3.14 and 3.15 depict these two plans, in particular the schedule of campaigns and the virgin pulp stock level. As usual the initial part of the plans is fixed. The differences thus start in the middle, where the optimized plan includes KLB275A and KLB170A (which were ignored by the manual plan) and an additional campaign of KLB225, on the descending part of the of the

sequence. Note that plans tend to follow this smooth ascending and descending pattern of grades, in order to avoid costly setups. Then, the cycle of the virgin pulp level is extended in the optimized plan, to produce a larger amount of KLB170 and avoid a further campaign of this grade. After that, the VLB campaigns have to start in order to rise the virgin pulp level again. Indeed, these grades incorporate considerably more recycled fibres and hence are used to balance the KLB grades, which have the opposite composition - more virgin fibers. The manual plan ends with low KLB grades, which were not repeated in the optimized plan.

To better understand the different planning approaches, we need to look at their impact on the main KPIs. The relative improvement of the optimized plan over the manual counterpart is given in Table 3.3.

Table 3.3: Comparison of the manual and optimized plans, with respect to main KPIs.

Backlog	Setups	Production	Stock	Overall
-6.4%	-10.0%	-1.8%	-2.6%	-7.4%

From those figures, one may conclude that the additional campaigns in the optimized plan helped improving the backlog in more than 6%. Moreover, the smoother grade changeovers and absence of the second wave of KLB grades at the end of the manual plan provided a reduction of 10% in setup costs. Furthermore, inventory was reduced by almost 3%. The only advantage of the manual plan is a 2% additional production. The optimization method made a trade-off when extending the production cycle, which forced a slight reduction of the paper machine's rate, but allowed a better fulfilment of demand and improvement of operational costs. This kind of trade-offs are hard to grasp in manual planning and practitioners are forced to use simple rules, such as a given production cycle, to generate feasible plans. Still, when programming stoppages for equipment maintenance or when stoppages arise in disturbance situations, it is more difficult to devise rules. In those cases, the DSS can bring additional value.

3.7.3 Plan considering stoppages

To end this section, a plan considering a long stoppage in the whole plant is inspected. The plan is illustrated in Figure 3.16, where the stoppage of the paper machine is represented as another paper grade. The paper machine stops after the digester in order to consume all the pulp left in the tanks. At the start-up the digester is activated before the machine to create some inventory of pulp. The machine starts with VLB165, which consumes significantly more recycled than virgin pulp. In this way the pulp inventory continues to rise until reaching a level where the KLB products (which consume more virgin pulp) can be produced.

Planning a stoppage takes the resources to their actual limits. Therefore, the rates have to be carefully managed. In the generation of the illustrated plan, some simplifying assumptions were made, such as the ability of the production rates to rise from (or fall to) zero instantly. The aim there was just to obtain a rough idea of what the plan would be.

Nevertheless, the system allows considering maximum rates variations, as well as easily redefining maximum and minimum rates. Hence, after this initial plan managers could generate a more accurate one by further customizing the system inputs.

3.8. Conclusions and future work

The DSS reported in this paper has been developed to help a P&P company deal with the various challenges of their production planning and scheduling activities. Companies in this sector have to differentiate themselves by providing the best customer service at minimum cost, since paper prices are determined by the market. Therefore, multiple criteria have to be considered and properly weighted when devising a production plan. However, managers do not have the time at the operational level to perform a thorough evaluation of plans, which must satisfy a multitude of complex constraints. Hence, the DSS supports the automatic generation of plans with different levels of customization (e.g. fixing the sequence and/or amounts of paper campaigns).

Several iterations with the company were performed in order to include a set of essential features for their planning tasks and to design user-friendly interfaces. The immediate updates (due to the “Software as a Service” setup), in-place editing, direct copy-paste from and to Excel, input validation procedures, automatic connection to existing systems and separation of plan generation and fine tuning interfaces are examples of technical details that were crucial for the usability of the system. On the other hand, the optimization model provided the required flexibility in incorporating operational constraints, such as fixing campaigns and scheduling stoppages. The method then makes use of the model to generate good quality solutions in the short periods of time available at the operational level.

As future work, there are additional features that can be included, such as improving the interactivity of the final schedules or providing more customization of the input data. These enhancements should be done according to company’s actual needs. In addition, the modular design of the system, as well as the general-purpose of the optimization method, would facilitate its adaptation or extension to other similar plants.

In scientific terms, the formulation may still be strengthened by additional valid inequalities or reformulations. Comparing the formulations of [Karimi and McDonald \(1997\)](#) and [Camargo et al. \(2012\)](#) could give some insights in this regard. Moreover the facility plant location reformulation could also improve the model’s solvability ([Amorim et al., 2011](#)). Another important aspect to be examined is the impact of the soft constraints in the optimization performance and how that could be addressed by the heuristic method. The latter could also be improved with respect to the neighborhoods used, possibly allowing it to add/remove campaigns to/from certain positions, as suggested by [Figueira et al. \(2013\)](#).

Finally, as computational power increases and the efficiency of the methods improve, the production planning problem can integrate more details of other activities (up or downstream in the supply chain) or address important issues, such as process variability and disturbances. The latter may have an important impact on the production system and therefore its consideration in a proactive planning approach is highly relevant.

Bibliography

- R. Akkiraju, P. Keskinocak, S. Murthy, and F. Wu. An agent-based approach for scheduling multiple machines. *Applied Intelligence*, 14(2):135–144, 2001.
- P. Amorim, T. Pinto-Varela, B. Almada-Lobo, and A. Barbósa-Póvoa. Comparing models for lot-sizing and scheduling of single-stage continuous processes: Operations research and process systems engineering approaches. *Computers and Chemical Engineering*, 52:177–192, 2013.
- Pedro Amorim, Carlos H Antunes, and Bernardo Almada-Lobo. Multi-objective lot-sizing and scheduling dealing with perishability issues. *Industrial & Engineering Chemistry Research*, 50(6):3371–3381, 2011.
- V. Camargo, F. Toledo, and B. Almada-Lobo. Three time-based scale formulations for the two-stage lot sizing and scheduling in process industries. *Journal of the Operational Research Society*, 63:1613–1630, 2012.
- Pedro M Castro, Iiro Harjunoski, and Ignacio E Grossmann. New continuous-time scheduling formulation for continuous plants under variable electricity cost. *Industrial & engineering chemistry research*, 48(14):6701–6714, 2009.
- CEPI. Key statistics – european pulp and paper industry 2012. Technical report, June 2013.
- MH Correia, J.F. Oliveira, and J.S. Ferreira. Integrated resolution of assignment, sequencing and cutting problems in paper production planning. *International Journal of Production Research*, 50(18):5195–5212, 2012.
- Andreas Drexl and Knut Haase. Proportional lotsizing and scheduling. *International Journal of Production Economics*, 40(1):73–87, 1995.
- Muge Erdirik-Dogan and Ignacio E Grossmann. Simultaneous planning and scheduling of single-stage multi-product continuous plants with parallel lines. *Computers & Chemical Engineering*, 32(11):2664–2683, 2008.
- Gonçalo Figueira, Maristela Oliveira Santos, and Bernardo Almada-Lobo. A hybrid vns approach for the short-term production planning and scheduling: A case study in the pulp and paper industry. *Computers & Operations Research*, 40(7):1804–1818, 2013.
- B. Fleischmann and H. Meyr. The general lotsizing and scheduling problem. *OR Spectrum*, 19:11–21, 1997. ISSN 0171-6468.
- Nicolas Gold, Andrew Mohan, Claire Knight, and Malcolm Munro. Understanding service-oriented software. *Software, IEEE*, 21(2):71–77, 2004.
- Iiro Harjunoski, Christos Maravelias, Peter Bongers, Pedro Castro, Sebastian Engell, Ignacio 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.

- Iftekhhar A Karimi and Conor M McDonald. Planning and scheduling of parallel semi-continuous processes. 2. short-term scheduling. *Industrial & Engineering Chemistry Research*, 36(7):2701–2714, 1997.
- Uday S Karmarkar and Linus Schrage. The deterministic dynamic product cycling problem. *Operations Research*, 33(2):326–345, 1985.
- May-Fong Lim and IA Karimi. Resource-constrained scheduling of parallel production lines using asynchronous slots. *Industrial & engineering chemistry research*, 42(26):6832–6842, 2003.
- Christos T Maravelias. Mixed-time representation for state-task network models. *Industrial & engineering chemistry research*, 44(24):9129–9145, 2005.
- Alain Martel, Nafee Rizk, Sophie D’Amours, and Hanen Bouchriha. Synchronized production-distribution planning in the pulp and paper industry. In Andre Langevin and Diane Riopel, editors, *Logistics Systems: Design and Optimization*, pages 323–350. Springer US, 2005. ISBN 978-0-387-24977-3.
- Sesh Murthy, Rama Akkiraju, Richard Goodwin, Pinar Keskinocak, John Rachlin, Frederick Wu, James Yeh, Robert Fuhrer, Santhosh Kumaran, Alok Aggarwal, Martin Sturzenbecker, Ranga Jayaraman, and Robert Daigle. Cooperative multiobjective decision support for the paper industry. *Interfaces*, 29:5–30, 1999. ISSN 0092-2102.
- S. Poltroniere, K. Poldi, F. Toledo, and M. Arenales. A coupling cutting stock-lot sizing problem in the paper industry. *Annals of Operations Research*, 157:91–104, 2008. ISSN 0254-5330.
- A. Respício and M.E. Captivo. Marketing-production interface through an integrated dss. *Journal of Decision Systems*, 17(1):119–132, 2008.
- N. Rizk, A. Martel, and S. D’Amours. Synchronized production-distribution planning in a single-plant multi-destination network. *Journal of the Operational Research Society*, 59(1):90–104, 2008.
- Maristela Oliveira Santos and Bernardo Almada-Lobo. Integrated pulp and paper mill planning and scheduling. *Computers & Industrial Engineering*, 63(1):1–12, 2012. ISSN 0360-8352.
- L. Szabó, A. Soria, J. Forsström, J.T. Keränen, and E. Hytönen. A world model of the pulp and paper industry: Demand, energy consumption and emission scenarios to 2030. *Environmental Science & Policy*, 12(3):257–269, 2009.
- Joakim Westerlund, Mattias Hästbacka, Sebastian Forssell, and Tapio Westerlund. Mixed-time mixed-integer linear programming scheduling model. *Industrial & engineering chemistry research*, 46(9):2781–2796, 2007.



Figure 3.12: Web interface of the DSS for fine-tuning structural (not frequently changed) parameters.

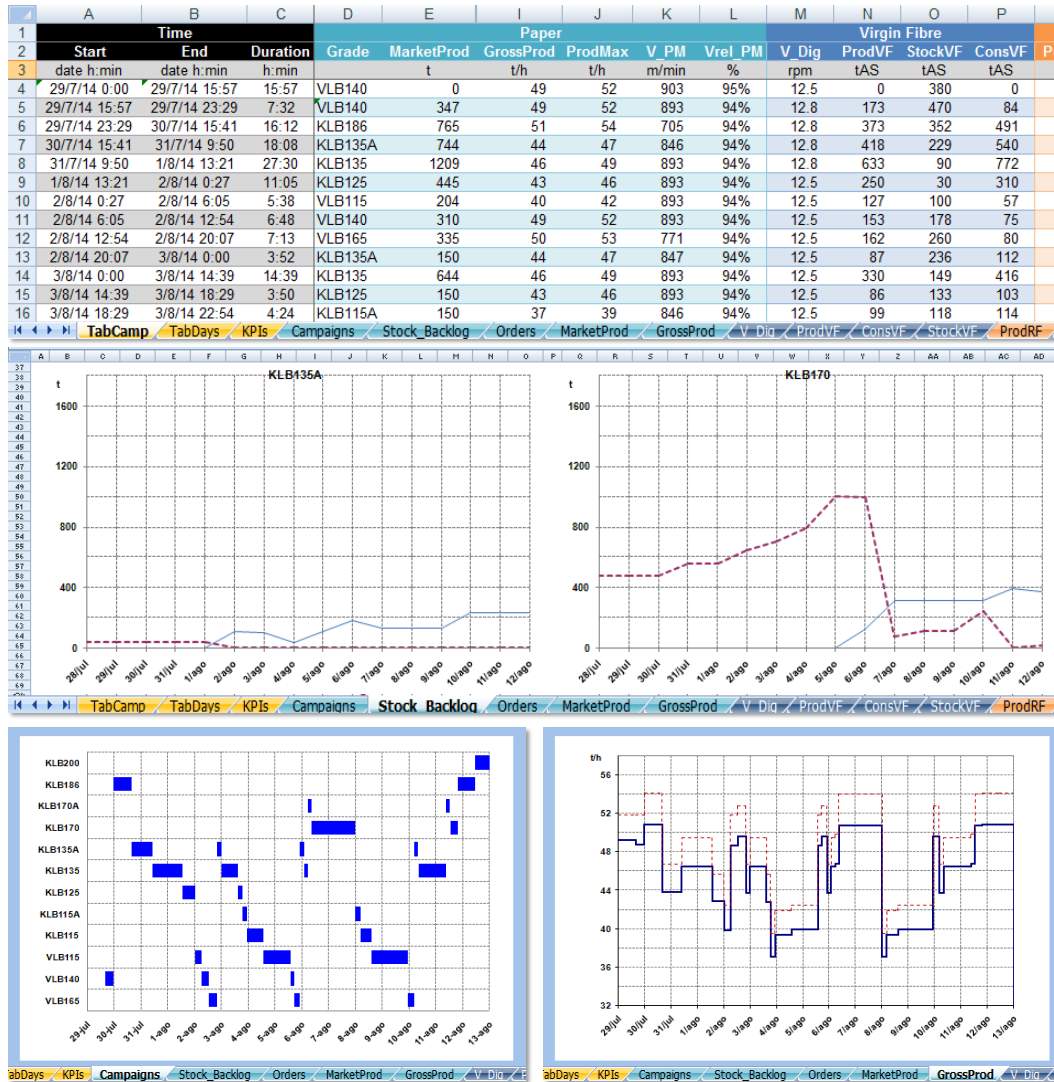


Figure 3.13: Output Excel file with tables and charts detailing the production plan (complete table, evolution of final products stock / backorders, paper campaigns and machine's production rate, respectively).

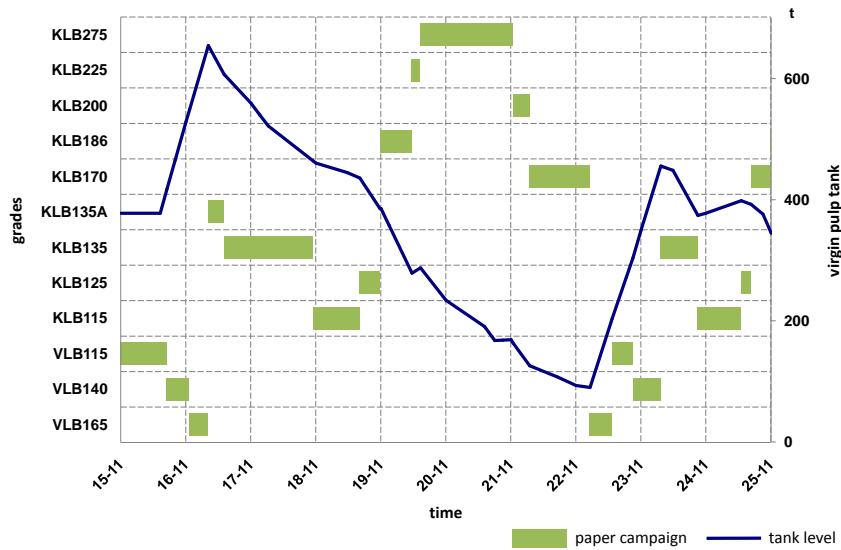


Figure 3.14: Manual plan – generated by the system, fixing the sequence and size of paper campaigns (grades on the left and virgin pulp on the right axis).

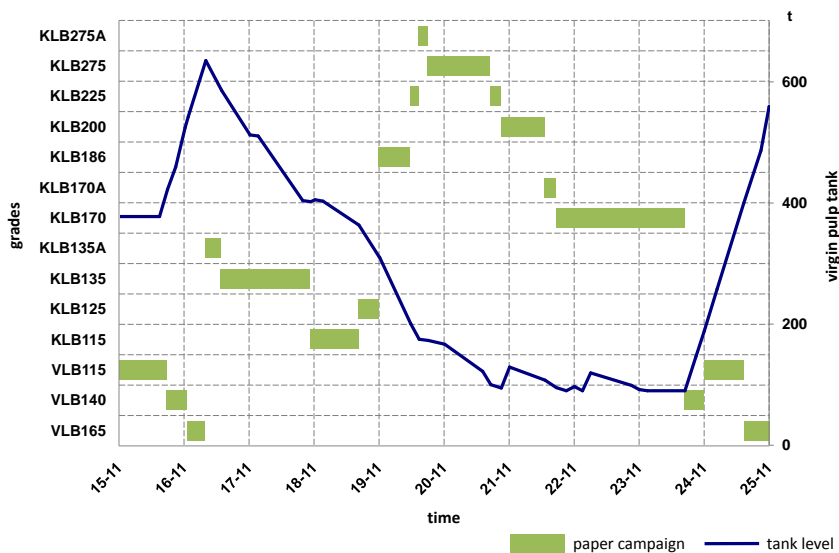


Figure 3.15: Optimized plan – generated by the system, optimizing the sequence and size of paper campaigns (grades on the left and virgin pulp on the right axis).

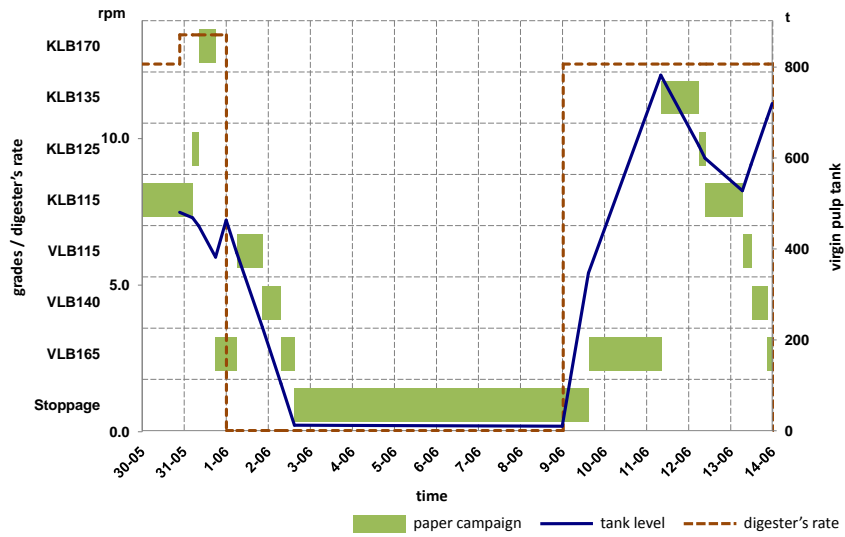


Figure 3.16: Plan considering a long long stoppage – paper grades (bars) and digester’s rate (dashed line) represented on the left axis and virgin pulp tank (solid line) on the right axis.

Simulation-optimization taxonomy

Hybrid Simulation-Optimization Methods: A Taxonomy and Discussion

Gonçalo Figueira* · Bernardo Almada-Lobo*

Published in *Simulation Modelling Practice and Theory*, 2014
<http://dx.doi.org/10.1016/j.simpat.2014.03.007>

Abstract The possibilities of combining simulation and optimization are vast and the appropriate design highly depends on the problem characteristics. Therefore, it is very important to have a good overview of the different approaches. The taxonomies and classifications proposed in the literature do not cover the complete range of methods and overlook some important criteria. We provide a taxonomy that aims at giving an overview of the full spectrum of current simulation-optimization approaches. Our study may guide researchers who want to use one of the existing methods, give insights into the cross-fertilization of the ideas applied in those methods and create a standard for a better communication in the scientific community. Future reviews can use the taxonomy here described to classify both general approaches and methods for specific application fields.

Keywords Simulation-Optimization · Taxonomy · Classification · Review · Hybrid Methods

4.1. Introduction

Simulation and Optimization were traditionally considered separate (or alternative) approaches in the operational research field. However, tremendous leaps in computational power promoted the appearance of methods that combined both. Simulation-based approaches started involving the optimization of the model inputs (also called controllable parameter settings). On the other hand, optimization-based approaches started using simulation for the computation of parameters (e.g. in queuing systems) or the sampling of scenarios for mathematical programming models. Nevertheless, this dichotomy is gradually vanishing, as other approaches are applying a balanced use of simulation and optimization (e.g. ROSA – see Subsection 4.2.3 – for a complete list of acronyms see 4.A). The

*INESC TEC, Faculdade de Engenharia, Universidade do Porto, Porto, Portugal

idea is to explore simultaneously the great detail provided by simulation and the ability of optimization techniques to find good or optimal solutions.

One of the main challenges hybrid simulation-optimization tries to answer is uncertainty. This aspect is addressed by a variety of (more conventional) approaches, such as stochastic programming, fuzzy programming and stochastic dynamic programming. The accuracy and detail of these models are however much lower when compared to simulation approaches. Furthermore, the difficulty in dealing with pure mathematical models leads in most cases towards the use of simulation for some computations. For instance, stochastic programming is most common in the form of scenarios (which may apply Monte Carlo simulation to perform the sampling), since the mathematical manipulation of probability distributions easily becomes intractable. Stochastic dynamic programming makes also use of simulation, when solving large complex models with the so-called reinforcement learning algorithms. A good overview of stochastic, fuzzy and stochastic dynamic programming is given by Sahinidis (2004).

Another major challenge is the consideration of nonlinear relationships, qualitative aspects or even processes hardly modelled by analytical expressions. The problem tackled by Albey and Bilge (2011) is an example of a completely deterministic problem that requires simulation. Indeed, the core advantage of simulation is its ability to deal with complex processes, either deterministic or stochastic, with no mathematical sophistication.

Still, combining simulation and optimization typically results in highly demanding methods in terms of computational effort, even for today's standards. Hence, the design of a good interaction is crucial. For that reason, and because the possibilities of combining them are so vast, it is very important to have a good overview of the different approaches. There is thus the need for a taxonomy which covers the full spectrum of these hybrid approaches and launches the discussion on the different strategies (their advantages and limitations).

A number of taxonomies and classifications have been proposed in the literature using different criteria. Some distinguished simulation-optimization methods by the applied techniques (e.g. statistical procedures, gradient approaches, heuristics, etc.) or their properties of convergence, optimality and correct selection (Carson and Maria, 1997; Banks et al., 2000; Fu, 2001). Other frameworks focused on the optimization problem, i.e. the solution space and objective function (Fu, 1994; Tekin and Sabuncuoglu, 2004; Barton and Meckesheimer, 2006; Ammeri et al., 2011). Shanthikumar and Sargent (1983) suggested a classification schema, according to the hierarchical structure of both simulation and optimization models. The authors have further distinguished between "hybrid models" (which they classify) and "hybrid modelling" (previously classified by Hanssmann et al. (1980)). In the former both analytic and simulation models are combined into one single model, whereas in the latter each model is able to generate a complete solution, but the final solution results from information exchanges between their executions.

While helpful for understanding the extent of research and practice in the field, these classifications have focused only on particular streams of methods. The last two considered only the cases where an analytical model exists *a priori* (which does not happen in several simulation-optimization approaches, such as stochastic approximation). The remaining papers addressed only the optimization of simulation model inputs, commonly known as

“simulation optimization” (SO). Here, we refer to “hybrid simulation-optimization” (or simply “simulation-optimization” – S-O) as any combination of these two major OR approaches.

Another aspect of those classifications that is subject to improvement is the possibility of combining different criteria. In fact, relating different dimensions and perspectives in a single classification can be critical when trying to grasp the essence of S-O methods and discover new opportunities for the cross-fertilization of ideas or the exploration of new approaches.

Finally, important criteria were overlooked. The first concerns the purpose of the simulation component in the overall design. This is the criterion that distinguishes the main streams of research in S-O. Fu (2002) outlined this dimension in two main categories (“optimization for simulation” and “simulation for optimization”), but the author has not developed it further. Another key dimension is the search scheme with respect to the series of solutions and realizations considered for evaluation. We refer here to “realization” as a short sample path (or simulation run) or part of a long path. The search scheme not only separates methods that tackle deterministic problems from those that address stochastic settings, but also discriminates the different strategies for dealing with the latter.

Two other papers (Pasupathy and Henderson, 2006; Han et al., 2011) sought to create taxonomies for simulation optimization problems and methods, in order to facilitate numerical comparisons and code reuse. Nevertheless, the interaction between simulation and optimization was not discussed and their studies were confined solely to SO methods.

In light of the above discussion, we propose a comprehensive taxonomy for S-O methods. Our classifying framework comprises four key dimensions: Simulation Purpose, Hierarchical Structure, Search Method and Search Scheme. The first two are related to the interaction between simulation and optimization, whereas the other two concern the search algorithm design. Considering these four dimensions (and their full spectrum), we are able to cover the complete range of S-O methods and distinguish virtually all of them in at least one dimension. The categories of each dimension had to be created from scratch, even for those already considered in the literature, since the confrontation of multiple criteria so required. The range of S-O methods includes: “simulation optimization” (already mentioned); “simulation for optimization”, where simulation helps enhancing an analytical model; and “optimization-based simulation”, where simulation generates the solution based on the optimization output (optimization does not need any simulation feedback).

One may question whether a so ambitious taxonomy is reasonable, or if it would make more sense studying and discussing those main streams of methods separately. The issue is that in many applications, even the choice of the main approach is not straightforward and consequently requires the consideration of methods that are entirely different in spirit. Moreover, some of these methods are more similar than it might appear at first sight.

This paper makes a clear distinction between the characteristics of the problem and those of the method and suggests connections between both. Our work has therefore a threefold contribution:

- give an overview of the full spectrum of simulation-optimization approaches, providing some guidance for researchers who want to use one of the existing techniques;

- explore the characteristics of these methods, giving insights into the cross-fertilization of their ideas and showing gaps that may result in new approaches;
- create a standard for a better communication in the scientific community, either when comparing existing S-O methods or when proposing a new one.

As opposed to other papers, we start by reviewing a variety of well-known methods (in Section 4.2) and only then propose our taxonomy (in Section 4.3). We do not intend to do an extensive review. Our aim is just to provide an overview of the main methods in the literature, referring to specific examples. Nevertheless, we seek to cover their full spectrum, in order to create a comprehensive taxonomy. Section 4.4 presents a discussion on the different S-O strategies and their relationship with the characteristics of the problem. Finally, Section 5.8 summarizes the conclusions and future lines of research. Since the review and taxonomy comprise a wide range of methods and categories, we provide a list of acronyms in appendix.

4.2. S-O methods

We divide this section in three main parts, each corresponding to one of the three major streams of simulation-optimization research: Solution Evaluation (SE), Analytical Model Enhancement (AME) and Solution Generation (SG) approaches, plus a fourth part for related methods. The first approach corresponds to SO and is more popular in the simulation community. Here, simulation is used to evaluate solutions and hence perceiving the landscape (or response surface). The other two combine simulation with analytical models, thus being classified in the literature as “hybrid simulation-analytic models/modelling”. These are mostly adopted within the optimization community.

S-O methods can be applied to a wide range of problems in areas such as manufacturing, warehousing, transportation, logistics, services, finance, among others. In this paper we present an illustrative example of a manufacturing system, which will be used to give context to the explanation of the S-O methods.

Consider the job shop consisting of four machines and three buffers (or queues) exhibited in Figure 4.1. Three different products can be produced in this system and each has its own sequence of machines to visit. Each machine can only process one product at a time. Whenever a machine finishes one job, the product moves to the buffer immediately before the following machine, waiting for being processed. However, the machine cannot discharge it if the succeeding buffer is full. The size of a buffer b is determined by its number of positions S_b , each having a certain capacity a (the overall buffer capacity is then $S_b \cdot a$). The total number of positions is limited. Processing times are stochastic and follow given probability distributions. Each product has a certain demand in each day and backlog is not allowed. The aim is then to determine the (continuous) production amounts X_{it} , for each product i in each day t , and appropriate (discrete) buffer sizes S_b (with $b \in \{2, 3, 4\}$) in order to optimize the expected aggregated costs, which include production, inventory and shortage (unmet demand) costs. Two variants (simplifications) of this problem will also be considered:

- lot sizing (determine \mathbf{X}) assuming infinite buffer capacities (infinite \mathbf{S});
- buffer space allocation (determine \mathbf{S}) assuming fixed production lots (fixed \mathbf{X}).

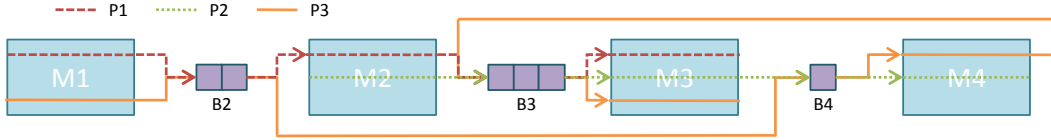


Figure 4.1: Production system: a job shop with three products (P1, P2 and P3), four machines (M1, M2, M3 and M4) and three buffers (B2, B3 and B4).

4.2.1 Solution Evaluation approaches

The first major S-O approach consists of developing a comprehensive simulation model to represent the system (such as the manufacturing system presented above) and use that model to evaluate the performance of various solutions. For instance, we could configure the system to produce given lot sizes and include certain buffers. Alternative solutions, with different lots and buffer spaces would then be tested and compared, in order to find a good (or optimal) solution. This experiment is thus the optimization of a simulation model, which in the literature is called “simulation optimization” (SO). Here, we refer to them as Solution Evaluation (SE) methods, since the purpose of simulation is to evaluate the performance of solutions.

The general SO problem setting is as follows:

$$\min f(\theta) \quad (4.1)$$

$$\text{s.t. } g(\theta) \geq 0, \quad (4.2)$$

$$\theta \in \Theta, \quad (4.3)$$

where θ represents the vector of input variables, Θ its potentially feasible domain (before running any simulation), $f(\theta) = Y[F(\theta, \omega)]$ and $g(\theta) = Z[G(\theta, \omega)]$ the objective and constraint functions (respectively) determined by a simulation model, ω a sample path, F and G the direct performance measures of the simulation output (depending on both the controllable vector and stochastic effects) and Y and Z the corresponding statistics (typically the expected value).

In our problem example:

- θ includes both \mathbf{X} and \mathbf{S} (in each variant however it includes either one or the other);
- $f(\theta) = E[F(\theta, \omega)]$, which can be represented by

$$E \left[\sum_{t=1}^T \sum_{i=1}^3 \left(c_{it} \cdot RX_{it}(\theta, \omega) + h_{it} \cdot RI_{it}^+(\theta, \omega) + q_{it} \cdot RI_{it}^-(\theta, \omega) \right) \right] \quad (4.4)$$

where RX_{it} , RI_{it}^+ and RI_{it}^- are the observed production, inventory and shortage amounts, and c_{it} , h_{it} and q_{it} are the production, inventory and shortage costs, respectively;

- $g(\theta) = G(\theta, \omega)$ would include constraints such as the inventory balance equations

$$RX_{it}(\theta, \omega) + RI_{i(t-1)}^+(\theta, \omega) + RI_{it}^-(\theta, \omega) - RI_{it}^+(\theta, \omega) = d_{it} \quad i = 1 \dots 3 \quad t = 1 \dots T \quad (4.5)$$

where d_{it} is the demand;

- Θ incorporates non-negativity and other *a priori* constraints, such as the total available buffer positions

$$\sum_{b=2}^4 S_b \leq N_{pos} \quad (4.6)$$

- ω is the set of values obtained for the (stochastic) processing times in a given realization.

Note the difference between X_{it} and RX_{it} . The former is the predefined amount, whereas the latter is that observed in the simulation (which will also depend on the buffer sizes S and the stochastic effects ω). There is a wide variety of methods approaching this problem. Some of the most important are following presented.

Statistical Selection Methods (SSM)

Consider the buffer space allocation problem stated above (second variant). If the number of possible solutions is not very high (let say each buffer can have one, two or three spaces, resulting in 27 combinations), then all those combinations can be evaluated in order to determine the optimal solution. However, the solution space is not the only to be explored. In fact, the problem is stochastic and thus one replication should not be enough to accurately evaluate the performance of each solution. Therefore, the number of replications for each solution (i.e. the way the probability space is explored) is also to be determined. Statistical Selection Methods, which include the well-known Ranking and Selection (R&S) and Multiple Comparison Procedures (MCP), focus on this aspect. These methods compare and select solutions applying statistical analysis, so that a given confidence is achieved in that process. The solution search typically consists of an exhaustive enumeration and is thus limited to a relatively small set of solutions (Θ) with discrete input variables, such as in this buffer space allocation problem. At the end, R&S provides the best solution, while MCP quantify the differences in performance ($f(\theta)$) among the solutions. Traditional procedures (here referred to as SSM^a) can be enhanced (to SSM^b) by the method of common random numbers (CRN) which induces a positive correlation among the solutions (Swisher et al., 2003). In practical terms CRN favours a fairer comparison between solutions, resulting in a reduction in the necessary number of replications.

Metaheuristics (MH)

To explore large or infinite solution spaces (such as larger instances of the buffer space allocation or the lot sizing problem), “true” search algorithms are needed. Metaheuristics are high level frameworks that combine basic heuristics in order to efficiently and effectively explore the search space (Blum and Roli, 2003). These algorithms may be single-solution based (e.g. simulated annealing (Alkhamis et al., 1999)), population-based (e.g. genetic algorithms (Truong and Azadivar, 2003)) or set-based (e.g. nested partitions (Shi and Ólafsson, 2000)) and can tackle both combinatorial (MH^a) and continuous optimization (MH^b). They were originally developed to address deterministic problems, even though the algorithm itself may be stochastic. Nevertheless, metaheuristics have also been applied to stochastic simulation optimization, where the evaluation of solutions is performed (multiple times for each) by the simulation model (Ólafsson, 2006). Indeed, given their flexibility to tackle any type of solution space and their ability to quickly achieve good quality solutions, these methods dominate the optimization routines of the discrete-event simulation software.

Memory-based Metaheuristics (MMH)

The difference to the previous is that while moving in the solution space, these methods construct and memorize a perception of the landscape. This memory structure is typically a probability distribution on the space of solutions which provides an estimate of where the best solutions are located. The generation of the following solutions then uses this distribution to perform a biased selection towards promising regions. Some examples include Swarm Intelligence (e.g. ant colony optimization (Dorigo and Blum, 2005)), Estimation of Distribution Algorithms (Larrañaga and Lozano, 2002), Cross-Entropy Method (Rubinstein and Kroese, 2004) and Model Reference Adaptive Search (Hu et al., 2008). These methods are also known as *model-based* methods, as opposed to the previous methods, which are *instance-based* (Fu et al., 2005), and may work either with discrete (MMH^a) or continuous variables (MMH^b).

Random Search (RS)

RS is very close to MH (and may even be seen as the same method), where a neighbourhood can be defined for each incumbent solution. However, the next move is probabilistically chosen, based on a given probability distribution. These methods have also been originally designed for conventional deterministic problems, but have been extended afterwards to the stochastic setting. In the latter, an additional feature has to be defined: how the best solution is selected. Possible alternatives include the incumbent solution at the end of the procedure, the most visited solution and the solution with the best sample mean. The choice is closely related to the approach for dealing with noise in the stochastic setting, which can range from expending a significant amount of computer effort on each point visited (RS^a) to deciding on how to proceed based only on limited information (RS^b). Further details on RS are given by Andradóttir (2006) and recent work is reported by Hong et al. (2010) and Xu et al. (2013).

Stochastic Approximation (SA)

In contrast to the previous method, SA is a gradient-based procedure and is therefore targeted at continuous variables (such as the lot sizing decisions of the problem described before). Under appropriate conditions, convergence towards local optima can be guaranteed and its rate is dramatically enhanced with the availability of direct gradients, which are problem-specific. Their absence is solved by computing naive estimations, such as finite differences or simultaneous perturbations. SA typically uses a short simulation run in each iteration, but can also perform gradient steps at intervals during a (single) long run (Single-Run Optimization – see [Suri and Leung \(1987\)](#)). For a comprehensive discussion on SA see [Kushner and Yin \(2003\)](#).

Sample-Path Optimization (SPO)

SPO ([Gurkan et al., 1994](#); [Plambeck et al., 1996](#)) is based on the idea of going out far enough along the sample path (to have a good estimate of the limit function), fixing it for every solution and solving the resulting problem applying deterministic optimization. In our illustrative problem a large set of processing times would be sampled and used in a long simulation run. Applying the same set of processing times to every solution (the CRN method extended to the whole domain) makes it a deterministic problem, which should be easier to solve. Indeed, the existence of effective methods for constrained deterministic optimization can be exploited here. In spite of the typical implementations (here denoted by SPO^a) being gradient-step methods, SPO is a general procedure and can be applied to many SE and AME methods.

Metamodel-based Methods

Instead of iterating through different solutions and running simulations to evaluate each of them, another approach is to first perceive the landscape in multiple points (using simulation) and then approximate a metamodel for those points. Deterministic optimization methods can then be applied to the obtained objective function, which is inexpensive to compute. This is the idea behind Metamodel-based Methods. A significant advantage of these methods, as [Barton and Meckesheimer \(2006\)](#) advocate, is the “reduction in prediction variance by extending the effect of the law of large numbers over all points in the fitting design”. However, “this advantage comes at a cost: bias that is introduced when the metamodel fails to capture the true nature of the response surface”.

Two main strategies exist: Global Metamodel-based Methods (GMM) and Local Metamodel-based Methods (LMM). The former are typically plain SO sequential procedures and require a global optimization strategy. The latter, which include the well-known Response Surface Methodology (RSM) – see [Khuri and Mukhopadhyay \(2010\)](#), alternate between model construction (from simulation responses) and deterministic optimization. For both, solutions may be evaluated with single (GMM^a and LMM^a) or multiple realizations/replications (GMM^b and LMM^b).

Gradient Surface Methods (GSM)

Combining techniques is frequently a promising avenue of research. One of these fruitful results is the GSM, which combines RSM and SA. Proposed by [Ho et al. \(1992\)](#), this method implicitly utilizes a second-order design for the fitting surface (in RSM), by considering the gradient surface modelled by a first-order design. However, only a single replicate is used to obtain each estimate. The procedure switches to a pure SA when the optimum is approached ([Fu, 1994](#)).

Surrogate Management Framework (SMF)

There are approaches that make also use of metamodels (or surrogate models), but do not apply a deterministic search directly on them. Instead, the surrogate model is used just to guide the search, screening out poor solutions, which do not get to be evaluated by simulation, or even selecting only high-quality solutions for examination. The latter is the case of SMF. This methodology was proposed by [Booker et al. \(1999\)](#) and is built on top of pattern search methods – see [Torczon \(1997\)](#). SMF creates a grid of points to serve as a basis for the recalibration of a surrogate model. The points are chosen to be spatially disperse, in order to improve the accuracy of the approximation, but at the same time be promising solutions. The surrogate model then predicts points at which improvements are expected. The recalibration procedure may use single (SMF^a) or multiple evaluations (SMF^b) in each point. The overall method does not require the existence of derivatives and still converges towards local optima.

Reverse Simulation Technique (RST)

The idea of RST is to specify in advance desired target values or ranges of values for particular simulation variables and run the model with expert systems guiding those variables towards their corresponding targets ([Wild and Pignatiello, 1994](#)). For instance, in our lot sizing problem we could specify that the time each machine m spends on each product i in each day t could not be higher than a given percentage of the total available time of that day. If this share was reached, then the machine would shift to another product. The adjustments thus occur during the simulation execution. RST appears to be an interesting approach to generate an initial solution of reasonable quality. Both continuous (RST^a) and discrete variables (RST^b) can be addressed by this method. In [Lee et al. \(1997\)](#) the authors report an RST algorithm that finds the steady state of a system and an optimal state.

Approximate Dynamic Programming (ADP)

The parametric optimization problem defined in the beginning of this section can be extended to a control optimization problem (often called dynamic optimization), which may be stated as: $\min_{U(\cdot)} E \left[\sum_{t=1}^T C(V(t, \omega), U(V, t)) \right]$, where $V(t, \omega)$ is the state of the system and $U(V, t)$ is the corresponding control action. The solution to this problem is not a finite dimensional vector (like θ), but a function ($U(V, t)$). The optimal control may be solved via dynamic programming (DP). However, according to [Gosavi \(2003\)](#), stochastic DP “suffers

from the curses of modelling and dimensionality”. The author states also that “these two factors have inspired researchers to devise methods that generate optimal or near-optimal solutions without having to compute or store transition probabilities”. ADP (also known as reinforcement learning or neuro-dynamic programming, in the artificial intelligence and control theory communities, respectively) is one such method. Combining DP and simulation, ADP is based on a learning agent which selects actions according to its knowledge of the environment. The latter responds by an immediate reward (the reinforcement signal). This procedure is similar to RST (in the sense that there is an agent controlling the system over the simulation), but here the agent learns and improves its actions over the process. For a brief review of ADP see [Powell \(2009\)](#).

Retrospective Simulation Response Optimization (RSRO)

All the aforementioned methods (with the exception of RST) deal with stochastic problems performing either one or multiple realizations for each solution. A different approach, which is the essence of RSRO ([Healy and Schruben, 1991](#)), is to consider one common realization for every solution. Note the difference to SPO, where the latter fixes a long sample path (i.e. multiple realizations for each solution). In the buffer space allocation problem, RSRO would simulate the system with an infinite (or very large) number of spaces until the total number of used spaces reached the real availability. This would be actually the optimal solution for that realization if the objective was to minimize the time to buffer exhaustion, but not necessarily a good solution for other scenarios. Hence, the method is repeated for a number of realizations. Each realization results in a distinct solution. Therefore, the expected value of the relevant performance measure cannot be evaluated. Instead, it provides for instance the solution with the maximum likelihood of being optimal (the poor behaviour when it is not optimal is not assessed though). Like RST, RSRO seems to be appropriate for generating good solutions for further evaluation and refinement. The most natural use of RSRO applies exact methods (RSRO^a), although heuristics can also be used. The RSRO approach should not be mistaken for the more recent retrospective optimization ([Pasupathy, 2010](#)), the latter being closer to SPO.

4.2.2 Solution Generation approaches

Using simulation to evaluate different solutions (as in every method of the previous section) can be very computationally intensive. For some particular problems, the feedback from simulation may not even be important to the choice of the solution. In those cases analytical models can be formulated and solved and their solutions simulated (optimization-based simulation), in order to compute all the variables of interest. The purpose of simulation here is not to verify the advantage of one solution over another, but simply to compute some variables and hence be part of the whole solution generation (SG).

The lot sizing problem described in the beginning of this section could be formulated

as:

$$\min \sum_{t=1}^T \sum_{i=1}^3 (c_{it} \cdot X_{it} + h_{it} \cdot I_{it}^+ + q_{it} \cdot I_{it}^-) \quad (4.7)$$

$$\text{s.t. } X_{it} + I_{i(t-1)}^+ + I_{it}^- - I_{it}^+ = d_{it} \quad i = 1...3 \quad t = 1...T, \quad (4.8)$$

$$\sum_{i=1}^3 p_{im} \cdot X_{it} \leq cap_{mt} \quad m = 1...4 \quad t = 1...T, \quad (4.9)$$

$$X_{it}, I_{it}^+, I_{it}^- \geq 0 \quad i = 1...3 \quad t = 1...T, \quad (4.10)$$

where p_{im} is the processing time (here assumed deterministic) of product i on machine m , cap_{mt} its capacity in period t and the remaining terminology is the same as that used in the previous subsection. Buffer spaces are assumed to be infinite. Note that production, inventory and shortage amounts are all decision variables, not observations of simulation. Simulation is then used to compute more realistic values for these variables.

In order to close the gap between predefined and simulated values, it would be important to reduce the production capacities (cap_{mt}) in the analytical model, since there will be waiting times between operations which are only considered in simulation. If defining these capacity reductions appears to be difficult and critical to the overall optimization, this SG approach might not be the best choice. However, if the analytical model is able to generate good solutions, this should be the best approach, especially when simulation is very expensive, since it only needs to run once. The optimization process can be performed either before or during that simulation run. These two schemes are described below.

Solution Completion by Simulation (SCS)

In this first method simulation is used to compute some variables, which will complete or correct the solution generated by optimization. In our example the model stated by (4.7)–(4.10) is solved for the entire horizon, resulting in certain production (X_{it}), inventory (I_{it}^+) and shortage (I_{it}^-) amounts. This solution is fed to simulation, which gives more accurate values for these variables. The literature applies SCS to different types of problems. [Briggs \(1995\)](#) approached the problem of mass tactical airborne operations. The analytical model generates a solution under ideal conditions. Simulation then considers the inherent variability and provides the expected, best and worst outcomes. Other applications of SCS somehow hybridize it with SE methods. For instance, [Lim et al. \(2006\)](#) determine a production-distribution plan with an analytical model and simulate that plan to obtain more realistic values. However, if the plan is not acceptable, the replenishment policy is changed and simulation is run again. [Truong and Azadivar \(2003\)](#) embed an SCS into an MH to solve a supply chain design problem. Their genetic algorithm determines some qualitative decisions. Each chromosome is decoded by an SCS method, where first an analytical model is solved to determine other decisions and then simulation is used to evaluate the complete solution. The popular commercial optimization module OptQuest also allows applying this type of hybridization.

Iterative Optimization-based Simulation (IOS)

Instead of a simply sequential procedure, optimization may be called during simulation execution. For instance the above analytical model can be solved in a rolling-horizon basis, simulating one period (t) at a time, updating the inventories and shortages of that period and solving the analytical model from that period until the end of the horizon. The new solution is then re-primed in simulation, which advances another period (to $t + 1$). In this way the output of optimization should be closer to that of simulation, since the former keeps track of the latter over the horizon. This iterative process is applied by [Subramanian et al. \(2000\)](#). However, in their case optimization is not called in a periodic scheme, but in an event-driven basis. Whenever the simulation module encounters a need for a control action, it momentarily suspends itself and communicates the state of the system to the optimization module, which resolves it. This approach may resemble RST. However, here the optimization phase is based on an analytical model and does not need simulation to evaluate its actions. In fact, RST obtains feedback from simulation, whereas IOS only receives feedforward. Furthermore the need for adjustments does not result from the specification of targets.

4.2.3 Analytical Model Enhancement approaches

Solution Generation methods can be very efficient, but if the feedback from simulation proves to be important, they may not be effective. Therefore, as mentioned above, they often end up being hybridized with Solution Evaluation methods. Another possibility is to enhance the analytical model using the simulation results. This analytical model enhancement (AME) can be conducted in different ways.

Stochastic Programming Deterministic Equivalent (SPDE)

Consider the lot sizing problem previously stated, but with only one machine. The waiting times issue is now simple to address. The only difficult aspect that remains is the uncertainty of processing times. In this situation, stochastic programming appears to be appropriate. Still, since mathematically manipulating probability distributions can be difficult, one may resort to Monte Carlo simulation to perform the sampling of scenarios, which are later embedded in a single analytical model. The final model is then the same as before, but where constraints (4.9) are replaced by:

$$\sum_{i=1}^3 p_{iw} \cdot X_{it} \leq cap_t \quad t = 1 \dots T \quad w = 1 \dots W, \quad (4.11)$$

where w represents a given scenario, for which all processing times p_{iw} are sampled from their corresponding distributions. This model is called a large-scale deterministic equivalent, since introducing the scenario index may increase the number and size of constraints by multiples. The general procedure is known as sample average approximation (SAA – see [Kleywegt et al. \(2002\)](#)) and includes SPO (which fixes a long sample path), but also the aggregation of realizations of multiple paths. The method starts with a given sample size,

which increases until required confidence and variance levels are achieved. An example is given in [Santoso et al. \(2005\)](#). These SPDE approaches typically require some assumptions / simplifications to be made, when compared to S-O using discrete-event simulation. Nevertheless, the work by [Schruben \(2000\)](#) may change this scenario (see Subsection 4.2.4).

Recursive Optimization-Simulation Approach (ROSA)

If the SPDE method was applied to the original system, consisting of four machines, it would probably fail to provide a good solution, since it would not consider waiting times. In those cases the output of a discrete-event simulation model should be helpful to assess those parameters and thus correctly estimate a proper capacity reduction for the analytical model. That is the idea behind ROSA. This approach was firstly proposed by [Nolan and Sovereign \(1972\)](#) and consists of running alternately a (typically linear) deterministic analytical model, such as (4.7)–(4.10), and a (typically stochastic discrete-event) simulation model. Simulation uses the solution generated by optimization and computes particular performance measures (in our case, throughput times). The values of these measures are then introduced again into the analytical model, refining its parameters (e.g. capacity reductions, processing times). The iterative process ends after a stopping criterion is met (e.g. convergence of the solution, parameters, objective). It should be noted that even if the processing times were deterministic, simulation would still be needed, due to the waiting times.

There are several implementations of this approach, applying different stopping criteria and refinement strategies. The variants in [Albey and Bilge \(2011\)](#), [Bang and Kim \(2010\)](#) and [Almeder et al. \(2009\)](#) are some interesting examples. The first two tackle pure deterministic problems (ROSA^a), whereas the last one approaches a stochastic environment (ROSA^b). In the latter case results from multiple simulation runs have to be aggregated (typically by quantiles, which may reflect different risk-averse levels). Some authors have criticized these iterative procedures, claiming that convergence can be problematic ([Irdem et al., 2008](#)). Still, most of the studies have found convergence in practice after a few iterations. [Acar et al. \(2009\)](#) propose a ROSA framework close to SE methods. The authors define an analytical model with an optimistic objective function. Simulation evaluates each solution generated (under more realistic conditions) and corrects the objective function for that particular solution. This ROSA variant is one of the few S-O methods that is optimal. Nevertheless, it is geared to discrete solution spaces.

Function Estimation based Approach (FEA)

An alternative to the iterative procedure of ROSA is the initial estimation of (often nonlinear) functions that describe the relationship between particular input and output variables. In our lot sizing problem the relationship between lot sizes X_t and production lead-times $lt(X_t)$, which include processing and waiting times, is not trivial and needs simulation to be estimated. By performing a set of simulations, this dependency can be characterized and included in the analytical model, specifically in the capacity constraints:

$$lt(X_t) \leq cap_{mt} \quad m = 1..4 \quad t = 1..T, \quad (4.12)$$

All the other constraints and the objective function remain the same. In this way, the method is not dependent on the convergence between simulation and optimization, like ROSA. However, this comes at the cost of a more difficult analytical model to be solved. An example of FEA is implemented by [Asmundsson et al. \(2006\)](#). The authors conduct an initial simulation study to specify the nonlinear relationship between the expected work-in-progress and the expected throughput. This relationship is reproduced in what are called “clearing functions”. The final analytical formulation incorporates an approximation of these concave functions by outer linearization.

Optimization-based Simulation with Iterative Refinement (OSIR)

ROSA assumes that once the decisions are made, they cannot be revised. However, in practice they often can. If that is to be considered, SPDE for instance has to be extended to a multi-stage stochastic program. This increases considerably the scale of the model to be solved, which easily becomes intractable. An alternative is proposed by [Jung et al. \(2004\)](#), who approach a supply chain management problem under demand uncertainty. The authors have devised a framework similar to IOS, but that performs refinements to the analytical model. The latter is thus solved in rolling-horizon within the simulation model (optimization-based simulation). This procedure is repeated multiple times, performing (every n iterations) the appropriate refinements to the safety stock levels, in order to accommodate the uncertainty of demand. The refinement process, however, is not straightforward in their case. Indeed, unlike the deterministic problems where capacity reductions directly result from simulated waiting times, in this stochastic problem the right adjustments are to be determined. For this purpose, the authors apply a gradient-based search. Their method can thus be seen as an IOS embedded in an SA method. The input variables of the latter are the refinement parameters in the former (the safety stocks).

4.2.4 Methods related to S-O

Before concluding this section, we would like to make a reference to methods that are closely related to S-O, but may not be exactly a hybridization between simulation and optimization. [Schruben \(2000\)](#) proposed modelling sample paths from event relationship graphs, which are used to simulate discrete event systems (DES) as the solutions to mathematical programming (MP) models. In [Chan and Schruben \(2008\)](#) the authors provide a procedure that generates these optimization models directly from the explicit event relationships and examine several examples and potential applications. The objective of the MP models, as in the simulation, is simply to execute events as early as possible. Hence, they are seen as simulation models. The solution of each MP model represents a single simulation run. This procedure does not fit in the S-O definition given in Section 5.1, since it does not combine simulation and optimization. Instead, it replaces the former by the latter. Thus, it can be called “simulation by optimization” (SbO).

SbO has two main advantages over conventional simulation. First, linear programming duality can be used to perform sensitivity analysis, which is simpler than the traditional way of computing sample path derivatives. Moreover, as [Chan and Schruben \(2008\)](#) ex-

plain, the solution obtained from linear programming may provide more information for a single simulation run because other perturbed sample paths can be reached from the current sample path by some additional computation (pivots), which may be easier than running a new simulation.

The other key advantage is related to the application of this approach to S-O. SbO promotes a deeper integration between simulation and optimization. Indeed, the constraints that describe the system dynamics can be embedded within an optimization model that also determines decisions (in addition to simulating the system). If the concept of SAA is also applied, a single optimization model can be enough to perform the whole S-O. This can be seen as an SPDE, but less restricted by assumptions that oversimplify the problem.

Nevertheless, SbO has a major issue to be addressed: computational time. If one SbO run can provide more information than one conventional simulation run, it is also true that it may consume significantly more time. The MP model may have integer variables which highly increase the computational burden. To overcome this difficulty, [Alfieri and Matta \(2012\)](#) propose approximate representations for simulation, based on time buffers. In [Alfieri and Matta \(2013\)](#), the authors apply a time-based decomposition algorithm to further reduce the computational effort.

4.3. S-O taxonomy: a new unified framework

The proposed framework for classifying simulation-optimization methods is composed of four dimensions:

1. Simulation Purpose,
2. Hierarchical Structure,
3. Search Method and
4. Search Scheme.

The first two are related to the interaction between simulation and optimization, whereas the other two concern the search algorithm design. In the following subsections we describe all the categories of each dimension and combine the four dimensions, two by two. The two matrices that result are then used to classify all the previously reviewed methods.

4.3.1 Interaction between simulation and optimization

The categories for Simulation Purpose correspond to the main streams of research in S-O:

- Evaluation Function (EF) – iterative procedures that use simulation to evaluate solutions and hence guide the search, validating its moves;
- Surrogate Model Construction (SMC) – methods which apply simulation for the construction of a surrogate model, which is either used to guide the search or is directly searched;

- Analytical Model Enhancement (AME) – approaches making use of simulation to enhance a given analytical model, either by refining its parameters or by extending it (e.g. for different scenarios);
- Solution Generation (SG) – methods where a simulation model generates the solution, using optimization for some computations.

The first two categories compose the Solution Evaluation (SE) approaches. The large variety of these methods and the core difference of having (or not) a surrogate model leads to this distinction.

Some of these streams of methods may in some situations appear quite similar. In both SMC and AME an analytical model is improved with the output of simulation. However, the key difference is that the former uses only evaluations of the objective function to update a problem-independent model, whereas the latter starts from a problem-specific model which is modified according to the simulation output. Similarly, SG may be very close to SE approaches (e.g. IOS vs. RST – see Subsection 4.2.2), although belonging to different streams of research. The importance of integrating all S-O approaches into one single classification is therefore evident.

For the Hierarchical Structure dimension, four other categories were defined:

- Optimization with Simulation-based Iterations (OSI) – in all (or part) of the iterations of an optimization procedure, one or multiple complete simulation runs are performed;
- Alternate Simulation-Optimization (ASO) – both modules run alternately and in each iteration, either both run completely or both run incompletely;
- Sequential Simulation-Optimization (SSO) – both modules run sequentially (either optimization following simulation or the opposite);
- Simulation with Optimization-based Iterations (SOI) – in all (or part) of the iterations of a simulation process, a complete optimization procedure is performed.

Figure 4.2 exhibits the combination of these two dimensions and the corresponding classification of each method. According to the categories previously defined, there is a trend along both axes of the matrix: the autonomy of the optimization procedure tends to grow and the total number of simulations, compared to the number of optimization iterations, tends to decrease. There are however cells that are not-applicable, particularly at two corners:

- top right corner – calling the simulation process multiple times (either in OSI or ASO), when only one solution is to be generated and no feedback is necessary;
- bottom left corner – simulation being the evaluation function of the optimization procedure and not being (either completely or partially) in the iterations of the latter.

The number of categories in both dimensions could be larger. For instance, SMC methods could be further divided into those that base all their search in the surrogate model (e.g.

LMM, GMM) and those which use it as a supporting tool to guide the search (e.g. MMH, SMF). Likewise, AME and SG methods could be divided according to the extent of the analytical and simulation models (which may result in a hybrid model or hybrid modelling – see [Shanthikumar and Sargent \(1983\)](#)). That is the essential difference between SPDE and ROSA, for instance. Also, the ASO and SSO structures include more than one possibility, as described above. Nevertheless, this level of aggregation in both dimensions was kept in order to simplify the analysis and favor a global view over the whole spectrum of methods.

Finally, another aspect worth mentioning is the fact that GSM appears in more than one cell. Indeed, hybrid methods, such as this, may embed different S-O interaction schemes and hence, fill more than one cell in the matrix. Subsection 4.3.3 explains how these methods can be fully classified.


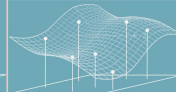
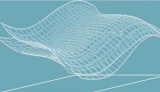

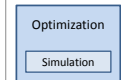
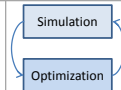
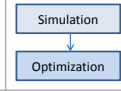
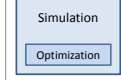
		Simulation Purpose			
		Solution Evaluation (SE)		Analytical Model Enhancement (AME)	Solution Generation (SG)
		Evaluation Function (EF)	Surrogate Model Construction (SMC)		
					
Hierarchical Structure	Optimization with Simulation-based Iterations (OSI) 	SSM, MH, RS, SA, SPO ^a , GSM	MMH		<i>Not-applicable</i>
	Alternate Simulation-Optimization (ASO) 	RST, RSRO	LMM, GSM, SMF, ADP	SPDE, ROSA	
	Sequential Simulation-Optimization (SSO) 	<i>Not-applicable</i>	GMM	FEA	SCS
	Simulation with Optimization-based Iterations (SOI) 				OSIR

Figure 4.2: Classification according to the interaction between simulation and optimization: simulation purpose and hierarchical structure.

4.3.2 Search design

Four types of methods are distinguished:

- Exact (E) – methods which achieve and prove global optimality (e.g. exhaustive and implicit enumeration, Dantzig’s simplex algorithm, branch-and-bound, column generation, etc.);
- Derivative-Based Heuristic (DBH) – heuristic methods (i.e. procedures that cannot guarantee an optimal solution) which are based on derivative information, thus targeting continuous variables, to make progress in their search (e.g. gradient-step methods, Hessian-based methods, etc.);

- Other Continuous Heuristic (OCH) – heuristic methods that tackle continuous variables without requiring derivative information (e.g. pattern search, Nelder-Mead’s simplex method, particle swarm optimization, etc.);
- Discrete Heuristic (DH) – heuristic methods that tackle discrete variables (e.g. neighbourhood-based metaheuristics, greedy heuristics, etc.).

When defining these four categories, we have looked at the major dichotomies in Optimization: exact vs. heuristic; and continuous vs. discrete (or combinatorial). Continuous can be further divided into derivative-based or other. Given the difficulty of most problems approached by S-O, the exact methods are not so frequently applied and were thus aggregated into one single category.

The search scheme concerns the sequence of solutions and realizations considered for computation. Four schemes exist:

- One realization for each solution (1R1S) – moving in the solution and probability spaces simultaneously (i.e. changing the realization whenever a new solution is generated);
- Different realizations for each solution (DR1S) – successively considering multiple realizations for each solution that is generated;
- Common realizations for each solution (CR1S) – successively considering multiple realizations for each solution that is generated (the realizations are the same for all solutions);
- One realization for multiple solutions (1RMS) – repeatedly applying the same realization to multiple solutions (the former may change after a series of solutions).

The last scheme is the framework of RSRO and includes methods that approach deterministic problems. If that is the case, the realization does not change during the entire procedure, as there is one single scenario. The second and third schemes (the latter being the essence of SAA and SPO) involve two variants each: the consideration of a constant or a variable (typically increasing) number of realizations. Fully sequential procedures in SSM, where in each iteration a single observation is added to each alternative solution that is still in the running, are examples of the second variant.

Figure 4.3 presents the matrix that combines these two dimensions and the corresponding classification of each method. An important observation to make is the appearance of many S-O methods in multiple cells each. Indeed, some of the methods proposed in the literature have a clearly specified procedure (e.g. SA), while other methods are more general approaches (e.g. ROSA). Nevertheless, the different variants identified in the literature were properly discriminated with different labels in the previous section. The identified variants are not exhaustive though. For instance, SPDE may use metaheuristics to find solutions, which should not be confused with MH methods (in the latter the metaheuristic is the S-O method, whereas in the former it concerns just the optimization part).

		Search Method			
		Exact (E)	Continuous-space Heuristic (CH)		Discrete-space Heuristic (DH)
			Derivative-Based Heuristic (DBH)	Other Continuous-space Heuristic (OCH)	
Search Scheme	One realization for each solution (1R1S)	IOS, OSIR	SA, GSM, LMM ^a , GMM ^a	SMF ^a	RS ^b , ADP
	Different realizations for each solution (DR1S)	SSM ^a , SCS, ROSA ^b	LMM ^b , GMM ^b	MH ^b , MMH ^b , SMF ^b	MH ^a , MMH ^a , RS ^a
	Common realizations for each solution (CR1S)	SSM ^b , SPDE, FEA	SPO ^a		
	One realization for multiple solutions (1RMS)	RSRO ^a , ROSA ^a		RST ^a	RST ^b

Figure 4.3: Classification according to the search design: method and scheme.

4.3.3 Complete classification

Our taxonomy was used to categorize well-known methods proposed in the literature, as well as some of their specific variants. However, the purpose is that it may be used to classify any S-O framework that appears, either being an adaptation of the existing methods or a new approach. The complete classification would then connect the four dimensions, resulting in [SimulationPurpose]-[HierarchicalStructure]/[SearchMethod]-[SearchScheme]. For instance, a genetic algorithm (GA) that evaluates individuals (represented by discrete solutions) by running simulations is a MH method and can be classified as EF-OSI/DH-DR1S. Some approaches apply however a hybridization of interactions and methods. An example is the GSM, which combine SA (EF-OSI) and RSM (SMC-ASO). GSM can then be classified as <EF-OSI,SMC-ASO>/DBH-1R1S. Some hybridizations are less evident. For instance, a GA that tackles both discrete and continuous variables is also considered a hybrid method. In this case the classification would be EF-OSI/<DH,OCH>-DR1S.

4.4. S-O strategies discussion

In this section we examine and discuss different S-O strategies in each of the dimensions of the proposed taxonomy. The strategies depend on the characteristics of the problem that is approached. Therefore, we try to understand these relationships. The following subsections explore every classifying dimension. Figure 4.4 summarizes some evident connections that result from the analysis of the taxonomy. Other connections are possible though, since there is a wide range of aspects that can influence the choice of an S-O design. Moreover, the diversity of S-O methods and the possibilities of modifying and adjusting them are vast, which makes it difficult to provide definite statements in this regard. Nevertheless, we believe this discussion can contribute to a better understanding of this field, similarly to the classifications proposed in the literature that relate S-O methods to problem characteristics (see Section 5.1). Indeed, the schema in Figure 4.4 can be seen as an extension of those

classifications (considering additional dimensions and categories).

4.4.1 Simulation purpose

Looking at the first dimension, the major streams of research in S-O can be distinguished. The SE approach assumes that the landscape of solutions is not known (and has to be perceived), whereas in AME it is partially known or known with uncertainty (and has to be improved) and in SG it is completely known (it naturally results from the problem and possibly some simulation). The first is applied in the optimization modules included in commercial simulation software, since it considers the simulation model as a black-box (see [April et al. \(2003\)](#) for further details). Nevertheless, when designing an S-O algorithm those assumptions do not need to reflect the real context.

It is evident that SE approaches, relying their search only on the output from a (detailed) simulation model, should perform a better selection of candidate solutions. That is longer true for EF than for SMC, since the latter may compute interpolations / extrapolations. Therefore, one may opt for SE approaches, rather than AME, even if the landscape is partially known. The downside of this strategy is that a great amount of simulation runs may be needed, hence increasing computational time. The issue of selecting the appropriate approach is thus related to a trade-off between solution quality and efficiency and how the consideration of an initial analytical model responds to that. This is what we mean by “useful analytical model” in the first question of Figure 4.4.

Typically, when a linear relationship between most input and output variables exists and is known, but there are particular aspects difficult or undesirable to be included in the analytical model, AME can be very efficient and effective (see ROSA and OSIR implementations). If no relationship is known and simulation is costly, either because of the need to perform many replications or because each replication is expensive, then SMC may be adequate (see for instance [Kleijnen et al. \(2010\)](#), where the authors reported a GMM implementation that outperformed OptQuest, in terms of number of replications and final solution quality). In the case of simulation being relatively modest, EF may fit well. The selection of SG approaches is more clear: they are suitable when the analytical model does not depend on any simulation feedback (although it may depend on simulation feedforward – see Subsection 4.2.2). The trend (along the axis of this dimension) of decreasing number of simulation runs validates these observations.

4.4.2 Hierarchical structure

The hierarchical structure between simulation and optimization concerns the dependence of one component on the other. This dependence is related to the previous dimension, since for each category of the latter, there are some structures that are typical and others which may not make sense (see Subsection 4.3.1). For instance, EF methods are usually associated with OSI structures, but not with SSO or SOI, since simulation works as an evaluation function and hence needs to be called in the iterations of optimization. AME approaches, on the other hand, are more flexible and may involve any structure. The choice will typically regard the dependence of the analytical model on the refinements performed

by simulation. For example, FEA fully enhances the analytical model just in the beginning, while ROSA does it recurrently, thus being suitable for applications where the parameters depend on the solution. The former is also able to deal with that issue, but they typically complicate the model introducing nonlinear relationships. OSIR requires the optimization problem to be solved multiple times during simulation. This analysis suggests that along this axis the optimization problem should be of decreasing complexity, since it may be further complicated or called more times. Still, it is difficult to make a global conclusion and hence, we chose to leave it as an open (and more conservative) question in Figure 4.4, which does not directly relate to problem characteristics. Indeed, it only questions about the dependence of one component on the other.

4.4.3 Search method

The search method focuses on the optimization problem (but not on the interaction with simulation). The categories defined for this dimension are aligned with the characteristics of the target optimization problem. Therefore, the correspondence is straightforward. Exact methods are appropriate when the problem is relatively easy (e.g. simple combinatorial, linear, quadratic). Large complex problems naturally require the use of heuristics. If the entire solution space is discrete, DH methods are suitable. For continuous optimization, either DBH or OCH methods can be used, although the former are more effective if derivatives exist. In mixed integer-continuous problems a hybrid solution method, combining DH with DBH/OCH methods, can be necessary. An example is given by [Gray et al. \(2010\)](#), where the authors propose an algorithm that hybridizes a GA with a pattern search method. Alternatively, heuristics may be combined with exact methods: DH+E, E+DBH or E+OCH. The first is suitable when the combinatorial part is difficult and the continuous part is linear (or simple nonlinear), which typically occurs in AME or SG approaches. The remaining are more appropriate when the combinatorial part is simple and the continuous part is complex nonlinear, most common in SE. The distinction between simple and difficult combinatorial problems typically concerns the existence of polynomial-time optimal algorithms and the problem size.

In the discrete-space case another distinction could be made. The decision variables can be integer or binary. The latter are typically related to qualitative decisions, which may activate sub-options that also need to be optimized. These “simulation configuration” problems ([Pierreval and Paris, 2003](#)) can be seen as an extension of SO, since not only the simulation input parameters need to be optimized, but also its configuration (entities or layout of the simulation model). For such problems, optimization becomes more difficult because mathematical programming, hill climbing and stochastic approximation methods are not usually applicable. In addition, there is the need for automatic generation of simulation models according to a systematic process ([Azadivar, 1999](#)). To tackle these problems, [Azadivar \(1999\)](#), [Truong and Azadivar \(2003\)](#) and [Pierreval and Paris \(2003\)](#) suggest evolutionary / genetic algorithms (a particular type of DH).

4.4.4 Search scheme

The selection of the search scheme is mainly related to the relative difficulty in travelling in both solution and probability spaces and to the trade-off between convergence and diversification. Therefore, if it is harder to move in the solution space, either DRIS or CRIS should be the correct choice. This is the case of approaches like ROSA, where each iteration requires the entire resolution of an analytical model. The difference between DRIS and CRIS is that the former focuses more on diversification whereas the latter provide faster convergence. On the other hand, if moving in the probability space appears to be more difficult (or non-existent, in the case of deterministic problems), then IRMS would be more appropriate. This is the foundation of the RSRO approach: for problems such as the well-known Newsvendor problem (see [Ruszczynski and Shapiro \(2003\)](#)), it can be more convenient to consider all the solutions before changing the realization. Finally, if moving in both spaces is similar in terms of effort, then IRIS may be the best option. This is the approach of some methods which are very focused on diversification issues in stochastic settings, such as SA and RS^b. In SMC methods (such as LMM or GMM) greater diversification is achieved with DRIS and IRIS. Both obtain a greater variety of realizations and the latter allows simulating more solutions. There is thus a trend of decreasing diversification (or increasing convergence) along the axis of this dimension, as suggested in Figure 4.4. Nevertheless, there is another aspect that should be taken into consideration when choosing the search scheme: the use of variance reduction techniques (VRTs) or SSM ideas, which require DRIS/CRIS schemes.

VRTs are methods that aim at reducing the variance of results for a given number of replications. This means that for the same confidence level, the application of VRTs may allow reducing the number of replications and hence, the computational effort. Several VRTs exist, such as the common random numbers ([Schruben, 2010](#)), antithetic variates ([Hammersley and Morton, 1956](#)), control variates ([Glynn and Szechtman, 2000](#)), importance sampling ([Glynn and Iglehart, 1989](#)) and stratified sampling ([McKay et al., 2000](#)). The first is the most simple and widely used VRT and consists on applying the same sets of random number streams to every solution. If this is applied to the complete optimization procedure, we are in the presence of CRIS. Still, it can be applied to each iteration separately, within a DRIS scheme. For instance, in a single-solution based MH all neighbours in a given iteration would be evaluated with the same realizations, but these would change in the following iterations.

In addition to VRTs, SSM principles may be applied in SE approaches. The idea of discarding simulation replications for solutions which quickly prove (statistically) to be of poor quality is very attractive. It is not necessary though that the statistical proof involves a good confidence level (such as 90% or higher). Indeed, it may even be desirable working with a low level, in order to promote diversification in the search procedure. Still, solutions of too low quality would be quickly rejected, saving computational time.

1	How difficult is it to conceive <i>a priori</i> a useful analytical model?		Simulation Purpose
	a	Impracticable/impossible	
	1.1	How expensive is simulation?	
	a.a	Modest	⇒ EF
	a.b	Very expensive	⇒ SMC
	b	Practicable, but the model needs to be further enhanced	⇒ AME
c	Practicable, even before any simulation feedback	⇒ SG	
2	How frequently does one component need the feedback of the other?		Hierarchical Structure
	a	Optimization highly depends on simulation	⇒ OSI
	b	Both need each other recurrently	⇒ ASO
	c	One needs the other once	⇒ SSO
	d	Simulation highly depends on optimization	⇒ SOI
3	What are the optimization problem characteristics?		Search Method
	a	Discrete decision space	
	a.a	Simple combinatorial problem	⇒ E
	a.b	Difficult combinatorial problem	⇒ DH
	b	Continuous decision space	
	b.a	Linear (or simple nonlinear) problem	⇒ E
	b.b	Complex nonlinear problem and existence of derivatives	⇒ DBH
	b.c	Complex nonlinear problem and inexistence of derivatives	⇒ OCH
	c	Mixed decision space	
	c.a	Simple combinatorial linear (or simple nonlinear) problem	⇒ E
	c.b	Difficult combinatorial linear (or simple nonlinear) problem	⇒ DH + E
	c.c	Simple combinatorial complex nonlinear problem	⇒ E + CH
	c.d	Difficult combinatorial complex nonlinear problem	⇒ DH + CH
4	How hard is it to travel in both solution and probability spaces and how important is convergence/diversification in the former?		Search Scheme
	a	Same difficulty and/or diversification is very important	⇒ 1R1S
	b	More difficult in solution space and/or diversification is important	⇒ DR1S
	c	More difficult in solution space and/or convergence is critical	⇒ CR1S
	d	More difficult in the probability space (or deterministic problem)	⇒ 1RMS

Figure 4.4: Evident connections between the problem characteristics and the classifying dimensions and categories.

4.5. Conclusion and future lines of research

In the S-O research field the dichotomy between “simulation-based” and “optimization-based” approaches is vanishing. Some of the main streams of methods may in some situations appear quite similar, as previously shown. Thus, the choice of the main approach may not be straightforward and require the consideration of methods that are different in spirit.

Indeed, the boundaries of these methods can be fuzzy. Some categorization is important though. It is also crucial that it covers all S-O approaches, so that a comprehensive view is obtained. That was the aim of our taxonomy.

The classifications used so far in the literature focused on particular streams of methods and typically utilized one single criterion. We propose a classification which relates multiple dimensions and perspectives in an attempt to grasp the essence of S-O methods and discover new opportunities for the cross-fertilization of ideas and the exploration of new approaches. Two new key criteria were considered, which greatly contributed to the discussion. The organization of criteria in two matrices allowed the examination of both the interaction between simulation and optimization and the search design. We were also able to explore the characteristics of each method and distinguish virtually all of them in at least one dimension. Additionally, common trends along different axes were identified. The discussion of each classifying dimension has resulted in some guidelines which can be used when selecting appropriate S-O designs for given problems.

The variety of methods reviewed in Section 4.2 reveals an immense range of possibilities for combining simulation and optimization. Some methods have a clearly specified procedure, while others are more general approaches. Nevertheless, research activities in this promising area are likely to increase significantly in the near future. We provide here some directions for further research:

- Embedding ideas from RS^b into MH frameworks. The former are very focused on the stochasticity issues, moving simultaneously in both solution and probability spaces. The latter contain a large variety of diversification and intensification strategies. An interesting framework is proposed by [Andradóttir and Prudius \(2009\)](#), where exploration, exploitation and estimation are carefully balanced. Another example is the work of [Xu et al. \(2010\)](#) whose method consists of a global-search phase, followed by a local search phase, and ending with a “clean-up” (selection of the best) phase. Nevertheless, these approaches are still in their infancy.
- Applying VRTs in DR1S/CR1S schemes. As previously discussed, these techniques may allow reducing computational effort when performing simulation replications. Contrary to what might appear, VRTs can be applied not only in SE approaches, but also in AME. Indeed, even when the aim of simulation is not evaluating solutions, the feedback of the former may take advantage of a reduced variance to better enhance the analytical model.
- Using SSM principles in other SE approaches. As opposed to the previous point, this idea can only be employed in SE approaches. The increase of efficiency that result from SSM may improve the performance of DH methods, such as RS and MH. This promising combination is being explored both in the literature (e.g. [Ólafsson \(2004\)](#)) and in practice (e.g. by OptQuest). Nevertheless, there are still some questions to be answered. For instance, [Almeder and Hartl \(2013\)](#) show that in their case a statistical selection with 95% of confidence performs worse than a sample average of 10 replications, since diversification is spoiled in the former. Therefore, the choice of a good confidence level is still an open question.

- Filling gaps in the matrices. Two obvious gaps in the Interaction matrix exist: SMC-SOI and AME-OSI. The former could be some kind of hybridization between GMM and ADP, where during simulation a global metamodel would be constructed (instead of simply reinforcing a function). The latter might result in a progressive refinement strategy, where an analytical model would be refined during its resolution. Regarding the Search Design matrix blanks, CRIS schemes could be an interesting approach for metaheuristics with difficult convergence (which is more likely in the stochastic setting).

To conclude, our taxonomy fully classifies any S-O method, including hybridizations. Therefore, it can contribute to a better communication in the scientific community. Future reviews can use the dimensions here described to classify additional S-O methods or extend them for particular streams of research. For instance, in ROSA approaches it may be important to distinguish different refinement strategies and stopping criteria. More in-depth studies may also allow better characterizing the relationship between S-O problems and methods (or strategies). There is a wide avenue of research in simulation-optimization, particularly regarding the study of specific application fields.

Appendix 4.A List of acronyms

General:

OR	Operations Research
SO	Simulation Optimization
S-O	Simulation-Optimization
SbO	Simulation by Optimization
DES	Discrete Event System
MP	Mathematical Programming

Simulation purposes:

SE	Solution Evaluation
EF	Evaluation Function
SMC	Surrogate Model Construction
AME	Analytical Model Enhancement
SG	Solution Generation

Hierarchical structures:

OSI	Optimization with Simulation-based Iterations
ASO	Alternate Simulation-Optimization
SSO	Sequential Simulation-Optimization
SOI	Simulation with Optimization-based Iterations

Search methods:

E	Exact
CH	Continuous-space Heuristic
DBH	Derivative-Based Heuristic
OCH	Other Continuous-space Heuristic
DH	Discrete-space Heuristic

Search schemes:

IRIS	One realization for each solution
DRIS	Different realizations for each solution
CRIS	Common realizations for each solution
IRMS	One realization for multiple solutions

Simulation-optimization methods:

SSM	Statistical Selection Methods
R&S	Ranking and Selection
MCP	Multiple Comparison Procedures
MH	Metaheuristics
GA	Genetic Algorithm
MMH	Memory-based Metaheuristics
RS	Random Search
SA	Stochastic Approximation
SPO	Sample Path Optimization
GMM	Global Metamodel-based Methods
LMM	Local Metamodel-based Methods
RSM	Response Surface Methodology
GSM	Gradient Surface Methods
SMF	Surrogate Management Framework
ADP	Approximate Dynamic Programming
RST	Reverse Simulation Technique
RSRO	Retrospective Simulation Response Optimization
SPDE	Stochastic Programming Deterministic Equivalent
SAA	Sample Average Approximation
ROSA	Recursive Optimization-Simulation Approach
FEA	Function Estimation based Approach
OSIR	Optimization-based Simulation Iterative Refinement
SCS	Solution Completion by Simulation
IOS	Iterative Optimization-based Simulation

Stochastic techniques:

VRT	Variance Reduction Technique
CRN	Common Random Numbers

Bibliography

- Yavuz Acar, Sukran N. Kadipasaoglu, and Jamison M. Day. Incorporating uncertainty in optimal decision making: Integrating mixed integer programming and simulation to solve combinatorial problems. *Computers & Industrial Engineering*, 56(1):106–112, 2009.
- Erinç Albey and Ümit Bilge. A hierarchical approach to fms planning and control with simulation-based capacity anticipation. *International Journal of Production Research*, 49(11):3319–3342, 2011.
- Arianna Alfieri and Andrea Matta. Mathematical programming formulations for approximate simulation of multistage production systems. *European Journal of Operational Research*, 219(3):773–783, 2012.
- Arianna Alfieri and Andrea Matta. Mathematical programming time-based decomposition algorithm for discrete event simulation. *European Journal of Operational Research*, 231(3):557–566, 2013.
- Talal M Alkhamis, Mohamed A Ahmed, and Vu Kim Tuan. Simulated annealing for discrete optimization with estimation. *European Journal of Operational Research*, 116(3):530–544, 1999.
- Christian Almeder and Richard F. Hartl. A metaheuristic optimization approach for a real-world stochastic flexible flow shop problem with limited buffer. *International Journal of Production Economics*, 145(1):88–95, 2013. ISSN 0925-5273.
- Christian Almeder, Margaretha Preusser, and Richard Hartl. Simulation and optimization of supply chains: alternative or complementary approaches? *OR Spectrum*, 31:95–119, 2009.
- A. Ammeri, W. Hachicha, H. Chabchoub, and F. Masmoudi. A comprehensive literature review of mono-objective simulation optimization methods. *Advances in Production Engineering & Management*, 6:291–302, 2011.
- S. Andradóttir. Chapter 20 an overview of simulation optimization via random search. In Shane G. Henderson and Barry L. Nelson, editors, *Simulation*, volume 13 of *Handbooks in Operations Research and Management Science*, pages 617–631. Elsevier, 2006.
- Sigrún Andradóttir and Andrei A. Prudius. Balanced explorative and exploitative search with estimation for simulation optimization. *INFORMS J. on Computing*, 21(2):193–208, 2009. ISSN 1526-5528.
- Jay April, Fred Glover, James P. Kelly, and Manuel Laguna. Simulation-based optimization: practical introduction to simulation optimization. In *Proceedings of the 2003 Winter Simulation Conference*, pages 71–78, 2003. ISBN 0-7803-8132-7.

- J. Asmundsson, R.L. Rardin, and R. Uzsoy. Tractable nonlinear production planning models for semiconductor wafer fabrication facilities. *Semiconductor Manufacturing, IEEE Transactions on*, 19(1):95–111, 2006.
- Farhad Azadivar. Simulation optimization methodologies. In *Proceedings of the 31st conference on Winter simulation: Simulation—a bridge to the future-Volume 1*, pages 93–100, 1999.
- June-Young Bang and Yeong-Dae Kim. Hierarchical production planning for semiconductor wafer fabrication based on linear programming and discrete-event simulation. *IEEE Transactions on Automation Science and Engineering*, 7(2):326–336, 2010.
- Jerry Banks, John S. Carson, Barry L. Nelson, and David M. Nicol. *Discrete-Event System Simulation*. Prentice Hall, 3 edition, 2000.
- Russell R. Barton and Martin Meckesheimer. Chapter 18 metamodel-based simulation optimization. In Shane G. Henderson and Barry L. Nelson, editors, *Simulation*, volume 13 of *Handbooks in Operations Research and Management Science*, pages 535–574. Elsevier, 2006.
- C. Blum and A. Roli. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Comput. Surv.*, 35:268–308, 2003. ISSN 0360-0300.
- A. J. Booker, J. E. Dennis, P. D. Frank, D. B. Serafini, V. Torczon, and M. W. Trosset. A rigorous framework for optimization of expensive functions by surrogates. *Structural and Multidisciplinary Optimization*, 17:1–13, 1999.
- David D Briggs. A hybrid analytical/simulation modeling approach for planning and optimizing mass tactical airborne operations. Technical report, DTIC Document, 1995.
- Yolanda Carson and Anu Maria. Simulation optimization: methods and applications. In *Proceedings of the 1997 Winter Simulation Conference*, pages 118–126, 1997.
- Wai Kin Victor Chan and Lee Schruben. Optimization models of discrete-event system dynamics. *Operations Research*, 56(5):1218–1237, 2008.
- Marco Dorigo and Christian Blum. Ant colony optimization theory: A survey. *Theoretical Computer Science*, 344(2):243–278, 2005.
- M. C. Fu. Simulation optimization. In *Proceedings of the 2001 Winter Simulation Conference*, volume 1, pages 53–61, 2001.
- M. C. Fu, F. W. Glover, and J. April. Simulation optimization: a review, new developments, and applications. In *Proceedings of the 2005 Winter Simulation Conference*, pages 83–95, 2005.
- Michael C. Fu. Optimization via simulation: A review. *Annals of Operations Research*, 53:199–247, 1994.

- Michael C. Fu. Feature article: Optimization for simulation: Theory vs. practice. *INFORMS J. on Computing*, 14(3):192–215, 2002.
- Peter W. Glynn and Donald L. Iglehart. Importance sampling for stochastic simulations. *Management Science*, 35(11):1367–1392, 1989.
- Peter W. Glynn and Roberto Szechtman. Some new perspectives on the method of control variates. In K.T. Fang, F.J. Hickernell, and H. Niederreiter, editors, *Monte Carlo and Quasi-Monte Carlo Methods 2000*, page 27–49. Springer-Verlag, 2000.
- Abhijit Gosavi. *Simulation-Based Optimization: Parametric Optimization Techniques and Reinforcement Learning*. Kluwer Academic Publishers, 2003.
- G.A. Gray, K. Fowler, and J.D. Griffin. Hybrid optimization schemes for simulation-based problems. *Procedia Computer Science*, 1(1):1349–1357, 2010. ISSN 1877-0509.
- G. Gurkan, A. Y. Ozge, and T. M. Robinson. Sample-path optimization in simulation. In *Proceedings of the 1994 Winter Simulation Conference*, pages 247–254, 1994.
- J. M. Hammersley and K. W. Morton. A new monte carlo technique: antithetic variates. *Mathematical Proceedings of the Cambridge Philosophical Society*, 52(03):449–475, 1956.
- J. Han, J. A. Miller, and G. A. Silver. Sopt: Ontology for simulation optimization for scientific experiments. In *Proceedings of the 2011 Winter Simulation Conference*, pages 2909–2920, 2011.
- F Hanssmann, G Diruf, W Fischer, and S Ramer. Analytical search models for optimum seeking in simulations. *Operations Research Spektrum*, 2(2):91–97, 1980.
- Kevin Healy and Lee W. Schruben. Retrospective simulation response optimization. In *Proceedings of the 1991 Winter Simulation Conference*, pages 901–906, 1991.
- Y. Ho, Leyuan Shi, Liyi Dai, and Wei-Bo Gong. Optimizing discrete event dynamic systems via the gradient surface method. *Discrete Event Dynamic Systems*, 2:99–120, 1992.
- L. Jeff Hong, Barry L Nelson, and Jie Xu. Speeding up compass for high-dimensional discrete optimization via simulation. *Operations Research Letters*, 38(6):550–555, 2010.
- Jiaqiao Hu, Michael C Fu, and Steven I Marcus. A model reference adaptive search method for stochastic global optimization. *Communications in Information & Systems*, 8(3): 245–276, 2008.
- D.F. Irdem, N.B. Kacar, and R. Uzsoy. An experimental study of an iterative simulation-optimization algorithm for production planning. In *Proceedings of the 2008 Winter Simulation Conference*, pages 2176–2184, 2008.
- June Young Jung, Gary Blau, Joseph F. Pekny, Gintaras V. Reklaitis, and David Eversdyk. A simulation based optimization approach to supply chain management under demand uncertainty. *Computers & Chemical Engineering*, 28(10):2087–2106, 2004.

- André Khuri and Siuli Mukhopadhyay. Response surface methodology. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(2):128–149, 2010.
- Jack P.C. Kleijnen, Wim van Beers, and Inneke van Nieuwenhuyse. Constrained optimization in expensive simulation: Novel approach. *European Journal of Operational Research*, 202(1):164–174, 2010.
- A.J. Kleywegt, A. Shapiro, and T. Homem-De-Mello. The sample average approximation method for stochastic discrete optimization. *SIAM Journal on Optimization*, 12(2):479–502, 2002.
- Harold J Kushner and George Yin. *Stochastic approximation and recursive algorithms and applications*, volume 35. Springer, 2003.
- Pedro Larrañaga and Jose A Lozano. *Estimation of distribution algorithms: A new tool for evolutionary computation*, volume 2. Springer, 2002.
- Young Hae Lee, Kyoung Jong Park, and Yun Bae Kim. Single run optimization using the reverse-simulation method. In *Proceedings of the 1997 Winter Simulation Conference*, pages 187–193, 1997.
- Seok Jin Lim, Suk Jae Jeong, Kyung Sup Kim, and Myon Woong Park. A simulation approach for production-distribution planning with consideration given to replenishment policies. *The International Journal of Advanced Manufacturing Technology*, 27:593–603, 2006.
- M. D. McKay, R. J. Beckman, and W. J. Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 42(1):55–61, 2000. ISSN 0040-1706.
- Richard L. Nolan and Michael G. Sovereign. A recursive optimization and simulation approach to analysis with an application to transportation systems. *Management Science*, 18(12):B676–B690, 1972.
- S. Ólafsson. Two-stage nested partitions method for stochastic optimization. *Methodology And Computing In Applied Probability*, 6(1):5–27, 2004.
- Sigurdur Ólafsson. Chapter 21 metaheuristics. In Shane G. Henderson and Barry L. Nelson, editors, *Simulation*, volume 13 of *Handbooks in Operations Research and Management Science*, pages 633–654. Elsevier, 2006.
- R. Pasupathy and S. G. Henderson. A testbed of simulation-optimization problems. In *Proceedings of the 2006 Winter Simulation Conference*, pages 255–263, 2006.
- Raghu Pasupathy. On choosing parameters in retrospective-approximation algorithms for stochastic root finding and simulation optimization. *Operations Research*, 58(4-Part-1): 889–901, 2010.

- Henri Pierreval and Jean Luc Paris. From ‘simulation optimization’ to ‘simulation configuration’ of systems. *Simulation Modelling Practice and Theory*, 11(1):5–19, 2003.
- Erica L. Plambeck, Bor-Ruey Fu, Stephen M. Robinson, and Rajan Suri. Sample-path optimization of convex stochastic performance functions. *Mathematical Programming*, 75:137–176, 1996.
- Warren B Powell. What you should know about approximate dynamic programming. *Naval Research Logistics (NRL)*, 56(3):239–249, 2009.
- Reuven Y Rubinstein and Dirk P Kroese. *The cross-entropy method: a unified approach to combinatorial optimization, Monte-Carlo simulation and machine learning*. Springer, 2004.
- Andrzej Ruszczyński and Alexander Shapiro. Stochastic programming models. In A. Ruszczyński and A. Shapiro, editors, *Stochastic Programming*, volume 10 of *Handbooks in Operations Research and Management Science*, pages 1–64. Elsevier, 2003.
- Nikolaos V. Sahinidis. Optimization under uncertainty: state-of-the-art and opportunities. *Computers & Chemical Engineering*, 28:971–983, 2004.
- Tjendera Santoso, Shabbir Ahmed, Marc Goetschalckx, and Alexander Shapiro. A stochastic programming approach for supply chain network design under uncertainty. *European Journal of Operational Research*, 167(1):96–115, 2005.
- Lee W. Schruben. Mathematical programming models of discrete event system dynamics. In *Proceedings of the 32nd conference on Winter simulation*, pages 381–385, 2000.
- Lee W. Schruben. *Common Random Numbers*. John Wiley & Sons, Inc., 2010. ISBN 9780470400531.
- J. G. Shanthikumar and R. G. Sargent. A unifying view of hybrid simulation/analytic models and modeling. *Operations Research*, 31(6):1030–1052, 1983.
- Leyuan Shi and S. Ólafsson. Nested partitions method for stochastic optimization. *Methodology and Computing in Applied Probability*, 2(3):271–291, 2000.
- Dharmashankar Subramanian, Joseph F Pekny, and Gintaras V Reklaitis. A simulation-optimization framework for addressing combinatorial and stochastic aspects of an r&d pipeline management problem. *Computers & Chemical Engineering*, 24(2):1005–1011, 2000.
- Rajan Suri and Ying Tat Leung. Single run optimization of a siman model for closed loop flexible assembly systems. In *Proceedings of the 1987 Winter Simulation Conference*, pages 738–748, 1987.
- James R. Swisher, Sheldon H. Jacobson, and Enver Yücesan. Discrete-event simulation optimization using ranking, selection, and multiple comparison procedures: A survey. *ACM Trans. Model. Comput. Simul.*, 13(2):134–154, 2003.

- Eylem Tekin and Ihsan Sabuncuoglu. Simulation optimization: A comprehensive review on theory and applications. *IIE Transactions*, 36(11):1067–1081, 2004.
- Virginia Torczon. On the convergence of pattern search algorithms. *SIAM J. on Optimization*, 7(1):1–25, 1997.
- Tu Hoang Truong and Farhad Azadivar. Simulation optimization in manufacturing analysis: simulation based optimization for supply chain configuration design. In *Proceedings of the 2003 Winter Simulation Conference*, pages 1268–1275, 2003.
- Rosemary H. Wild and Joseph J. Pignatiello, Jr. Finding stable system designs: a reverse simulation technique. *Commun. ACM*, 37(10):87–98, 1994.
- Jie Xu, Barry L Nelson, and JEFF Hong. Industrial strength compass: A comprehensive algorithm and software for optimization via simulation. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 20(1):1–29, 2010.
- Jie Xu, Barry L Nelson, and L Jeff Hong. An adaptive hyperbox algorithm for high-dimensional discrete optimization via simulation problems. *INFORMS Journal on Computing*, 25(1):133–146, 2013.

Simulation-optimization for proactive planning and scheduling

Simulation-optimization for planning and scheduling of two-stage continuous systems subject to variability and disturbances

Gonçalo Figueira* · Bernardo Almada-Lobo*

Working paper, 2014

Abstract Continuous production systems are frequently subject to high variability and disturbances in the process. The literature tends to focus on reactive planning to avoid the burden of dealing with stochastic models. Simulation-optimization (S-O) appears as a promising technique for proactive planning, since it combines the ease of simulation for dealing with complexity and the focus of optimization in finding good quality solutions. However, selecting the appropriate S-O design is critical to the efficiency of the final method. This paper studies different ways of combining simulation and optimization when approaching the production planning and scheduling of a two-stage continuous system subject both to process variability and disturbances. An S-O method is then implemented and tested against a non-proactive approach. The first preliminary results validate the effectiveness of the proposed S-O method in generating robust production plans.

Keywords Simulation-optimization · Lot-sizing and Scheduling · Process Industry · Continuous Production · Proactive Planning · Process Uncertainty

5.1. Introduction

Production planning and scheduling are important activities when trying to reduce costs and provide a good service level to customers. In process industries that is particularly relevant, since the daily operation has to return the investments made on expensive machinery. The literature on process systems engineering (PSE) has been active in providing models and algorithms to improve planning and scheduling in a variety of industries and for both batch and continuous processes. However, most of them assume all the input data to be known

*INESC TEC, Faculdade de Engenharia, Universidade do Porto, Porto, Portugal

in advance, despite these processes being typically characterized by high variability levels (e.g. in production yield) and the occurrence of disturbances (such as equipment failures). These uncertainties in the process create a gap between plans and their execution.

The approaches that attempt to close that gap can be classified as reactive or proactive. The former are essential in the presence of serious disturbances. However, they can also be used in a periodic scheme (rolling-horizon). Lindholm et al. (2013) apply hierarchical scheduling and control in this rolling-horizon scheme, where the lower levels update the schedule of upper levels. Subramanian et al. (2012) use ideas from model predictive control (MPC) in a deterministic scheduling approach. Nevertheless, these reactive approaches do not optimize production plans, but rather their execution. The optimization criteria are thus related to the minimization of deviations from the plan. Therefore, if those plans are not robust, their implementation can be problematic. Hence, proactive approaches are crucial. Some efforts were made in order to integrate planning and control. Harjunkoski et al. (2009) reviewed some contributions in this context. Still, the authors concluded that the integrated approaches essentially start from the control's point of view and have only been able to address smaller instances of the planning problem.

At the planning level, there are three main methodologies to obtain robust solutions and thus mitigate the effects of variability and disturbances:

- accommodate uncertainty, considering buffers in the form of (predefined) safety stocks or safety times;
- incorporate uncertainty, in the so-called stochastic programming, fuzzy logic or robust optimization;
- consider uncertainty in a lower level, simulating the execution of plans.

The first branch is the most simplistic and results in the plain deterministic models that abound in the literature. If buffers are defined blindly, they may result in too conservative or too risky plans. This has motivated the formulation of models that incorporate stochastic elements (second branch of approaches). Several works have devised a variety of approaches, based on different well-known techniques. Balasubramanian and Grossmann (2002) and Engell et al. (2004) proposed stochastic programming models for scheduling problems under process and demand uncertainty, respectively. To avoid the exponential complexity that such models can take, decomposition methods were devised. These approaches however require simplifying assumptions that reduce their applicability to a wide range of problems. Janak et al. (2007) and Wittmann-Hohlbein and Pistikopoulos (2013) have also tackled scheduling problems under uncertainty. They have chosen robust optimization techniques, which are very efficient, but frequently too conservative. Some works have further combined scheduling with planning in a stochastic environment, using either fuzzy set theory (Majozi and Zhu, 2005), stochastic programming (Wu and Ierapetritou, 2007) or robust optimization (Verderame and Floudas, 2010). However, these methods have not integrated planning and scheduling in a single model, but used procedures which iterated between these two levels. A full integration of lot-sizing and scheduling is imperative in the presence of significant sequence-dependent setups. This integration has essen-

tially been achieved with deterministic approaches and for problems of moderate complexity and size (e.g. [Erdirik-Dogan and Grossmann \(2008\)](#)). Simulation-optimization (S-O) appears to be a promising technique to extend them to stochastic settings, while keeping the required level of detail.

Indeed, S-O provides a good framework for dealing not only with uncertainty (see for example [Chu et al. \(2014\)](#), who were able to keep a deterministic planning model, while the agent-based simulation incorporated the stochastic part), but also with elements which are hard to include in analytical models, such as non-linearities and qualitative aspects. [Castro et al. \(2011\)](#), for instance, combined mathematical programming with discrete-event simulation to address the issue of having a shared transportation resource in a multi-stage batch plant. In our case simulation provides accurate representations of different types of uncertainty. While stochastic programming allows assigning probability distribution to certain parameters, such as daily capacity, simulation is able to emulate disturbances (with specific starting and ending times) with no mathematical sophistication or complexity. S-O is hence very powerful both for increasing the realism and detail of models/methods and for improving their solution time. Nevertheless, simulation and optimization can be combined in numerous ways and the selection of the appropriate design can have a huge impact on the efficiency of the method.

This paper provides a study on different ways of combining simulation and optimization when approaching the production planning and scheduling of a two-stage continuous system, which is subject both to process variability and disturbances. Previous research ([Figueira et al. \(2013a\)](#)) started to devise an S-O framework for this type of problems. In the current paper that problem is generalized and the S-O method is improved.

The paper is organized as follows: Section 5.2 describes the general problem to be tackled. That will further motivate the integration of lot-sizing and scheduling and the use of S-O. Section 5.3 exhibits the (deterministic) analytical model, while Section 5.4 summarizes the ideas of the solution method for that problem. The stochastic part of the problem is then addressed in Section 5.5, where the simulation model is depicted. The exploration of the combination of simulation and optimization is conducted in Section 5.6. First, the main S-O approaches are discussed in the context of the underlying problem. Then, an S-O framework is proposed. Section 5.7 provides preliminary results of the application of that framework in a real case study. The paper ends with some conclusions and directions for future research.

5.2. Problem description

The industrial problem under investigation was inspired by the short-term planning and scheduling of a pulp and paper plant, described in ([Figueira et al., 2013b](#)). Here, some specificities of the problem are removed, so that it becomes more general and we can focus on the most relevant features, which appear in other production environments.

The plant operates in continuous production, on a 24/7 basis and has two main production stages:

- a production unit, which can represent a reactor, mill or furnace;

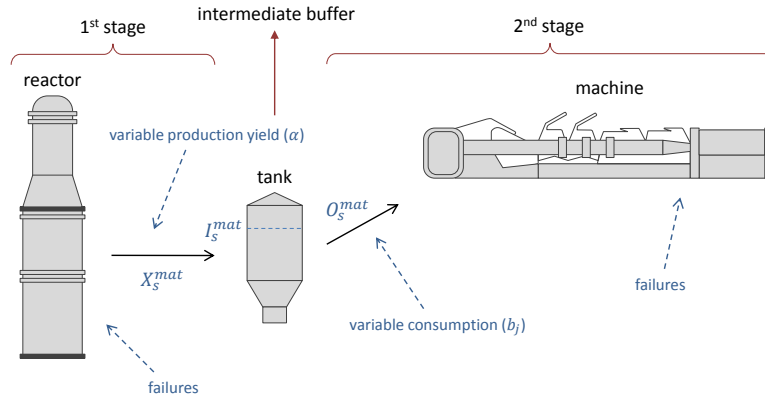


Figure 5.1: Two-stage continuous system, subject to variability and disturbances.

- and a machine, which can produce multiple products (with sequence-dependent setups) from the material that comes from the previous stage.

These stages have maximum and minimum rates (no idle time exists). Between them there is an intermediate tank, which works as a buffer, allowing the machine to consume at a higher or lower rate than the output rate of the reactor. This is crucial, as different products have different consumption levels, which makes the bottleneck shifting between the two stages. On the other hand, changing production rates or setting up products involves important costs, due to wear and stability issues. It is thus desirable to use the tank to its limits, in order to make the most of production cycles, while rate and product changes are minimized.

Nevertheless, the devised plans for this system should not consider the physical limits of the tank, but more constrained levels, so that there is some slack to face disturbances and process variability. The latter include variations in production and consumption. Disturbances regard any type of disruption that has a beginning and an end (transitory change). In this study, we consider failures on both resources (from the two production stages), which will block their production for some time, hence the need for the slacks in the tank. Keeping track of this tank level is therefore essential and that, in addition to the sequence-dependent setup times and costs, motivates the full integration of lot-sizing and scheduling. The planning process should be conducted for a 15 day horizon, so that production cycles are effectively checked.

The sequence of products has to take into account not only the operational constraints and objectives, but also the commitments to the final clients. Indeed, the system operates in a make-to-order (MTO) policy and has certain due dates to take into account. However, given that there is neither extra, nor idle time, backlog can be hard to avoid sometimes. Figure 5.1 illustrates the system being approached.

5.3. Mixed integer programming model

To model the system described in the previous section, we use the general lot-sizing and scheduling problem (GLSP), proposed by [Fleischmann and Meyr \(1997\)](#), which is extended to capture the multi-stage setting. This formulation allows considering continuous production amounts, tracing the tank level whenever rates or products change and keeping a moderate number of periods. In terms of computational performance, it proves to be superior to other similar models (see [Camargo et al. \(2012\)](#)).

We present the formulation in four parts. The first two concern the two production stages, the third regards production rates variation and the last one the objective function.

5.3.1 First stage (reactor) and intermediate tank

Indices, sets and parameters:	
t	Index for time period ($t \in [T] = \{1, \dots, T\}$)
s	Index for time slot ($s \in [S] = \{1, \dots, S\}$)
$[S_t]$	Set of slots s belonging to period t
cap_t	Capacity of period t
N_{min}	Minimum slot length
α	Material's production yield
$V_{max}^{reac} (V_{min}^{reac})$	Reactor's maximum (minimum) rate
I_0^{mat}	Initial inventory of material
$I_{max}^{mat} (I_{min}^{mat})$	Upper (lower) limit on material stocked in the tank
Decision variables:	
N_s	Length of slot s
X_s^{mat}	Output of the reactor in slot s
I_s^{mat}	Inventory of material in the tank at the end of slot s
O_s^{mat}	Output of material from the tank in slot s

$$\sum_{s \in [S_t]} N_s = cap_t, \quad t \in [T] \quad (5.1)$$

$$N_s \geq N_{min}, \quad s \in [S] \quad (5.2)$$

$$\alpha \cdot V_{min}^{reac} \cdot N_s \leq X_s^{mat} \leq \alpha \cdot V_{max}^{reac} \cdot N_s, \quad s \in [S] \quad (5.3)$$

$$X_s^{mat} + I_{s-1}^{mat} = O_s^{mat} + I_s^{mat}, \quad s \in [S] \quad (5.4)$$

$$I_{min}^{mat} \leq I_s^{mat} \leq I_{max}^{mat}, \quad s \in [S] \quad (5.5)$$

The production slots within each time period are of variable length. On the other hand, the period time grid is fixed. This two-level time grid is quite advantageous to address simultaneously the features of the operational process and the requirements of external demand. Constraints (5.1) ensure that the time slots respect the capacity of each period. In addition, each of these slots has a minimum length (constraints (5.2)), so as to account for a stabilization period after a possible change in rates or products. It should be highlighted that a common time grid is used for both production stages (i.e. the slot boundaries cross

over the reactor and machine time grids). The defined time slots are then used to determine reactor's production, which also depends on its (maximum and minimum) rates and a yield factor – see (5.3). The material balance is established by (5.4), whereas its storage limitations are enforced by (5.5).

5.3.2 Second stage (machine)

Indices and parameters:	
j, k	Indices for products ($j, k \in [K] = \{1, \dots, K\}$)
sl_{kj}	Product lost in a changeover from product k to j
st_{kj}	Time lost in a changeover from product k to j
b_j	Percentage of material in product j
p_j	Minimum processing time of product j
V_{min}^{mach}	Machine's minimum rate (relative to its maximum)
M_j	Upper bound on the lot size of product j
m_j	Minimum lot size of product j
d_{jt}	Demand for product j in period t
vl	Variable production loss
Y_{j0}	$\begin{cases} 1 & \text{if the machine is set up for product } j \text{ at the beginning of the planning horizon,} \\ 0 & \text{otherwise.} \end{cases}$
I_0^+	Initial inventory of of product j
I_0^-	Initial backlog of of product j
Decision variables:	
Y_{js}	$\begin{cases} 1 & \text{if the machine is set up for product } j \text{ in slot } s, \\ 0 & \text{otherwise.} \end{cases}$
Z_{kjs}	$\begin{cases} 1 & \text{if a changeover from product } k \text{ to } j \text{ takes place at the beginning of slot } s, \\ 0 & \text{otherwise.} \end{cases}$
X_{js}	Quantity of product j produced in slot s
$I_{j,t}^+$	Inventory of product j at the end of period t
$I_{j,t}^-$	Quantity of product j backlogged at the end of period t

$$\sum_j b_j \cdot \left(X_{js} + \sum_k sl_{kj} \cdot Z_{kjs} \right) = O_s^{mat}, \quad s \in [S] \quad (5.6)$$

$$V_{min}^{mach} \cdot N_s \leq \sum_j \left(p_j \cdot X_{js} + \sum_k st_{kj} \cdot Z_{kjs} \right) \leq N_s, \quad s \in [S] \quad (5.7)$$

$$(1 - vl) \cdot \sum_{s \in [S_t]} X_{js} + I_{j,t-1}^+ - I_{j,t-1}^- = d_{jt} + I_{jt}^+ - I_{jt}^-, \quad j \in [K], t \in [T] \quad (5.8)$$

$$X_{js} \leq M_j \cdot Y_{js}, \quad j \in [K], s \in [S] \quad (5.9)$$

$$\sum_j Y_{js} = 1, \quad s \in [S] \quad (5.10)$$

$$\sum_k Z_{jks} = Y_{j,s-1}, \quad j \in [K], s \in [S] \quad (5.11)$$

$$\sum_k Z_{kjs} = Y_{js}, \quad j \in [K], s \in [S] \quad (5.12)$$

The production of final products is linked to material consumption with constraints (5.6) and constrained by machine's minimum and maximum rates with (5.7). Note the sequence-dependent setup times and losses in these equations. Indeed, during change overs the machine keeps pulling material from the tank. The products processed on this stage might incorporate other materials rather than that stocked in the tank (e.g. different types of pulp, in the pulp and paper industry). Parameter b_j gives enough flexibility to represent different environments. The processing time p_j used in (5.7) corresponds to machine's maximum rate. Therefore, the right-hand side includes only the length of the slot, while the left-hand side multiplies it by the minimum rate (as a percentage of the maximum). Constraints (5.8) model the demand satisfaction, where production suffers an additional loss (beyond setup losses), which depends on the amount produced. Constraints (5.9) link production to setups, (5.10) enforce only one product per period, and (5.11) and (5.12) link setups to changeovers, guaranteeing the consistency of the production sequence throughout the horizon.

5.3.3 Rates variation

Parameters:	
V_0^{reac}	Rate of the reactor at the beginning of the planning horizon
V_0^{mach}	Rate of the machine at the beginning of the planning horizon
Decision variables:	
V_s^{reac}	Rate of the reactor in slot s
δ_s^{reac+} (δ_s^{reac-})	Positive (negative) variation of reactor's rate in slot s
V_s^{mach}	Rate of the machine in slot s
δ_s^{mach+} (δ_s^{mach-})	Positive (negative) variation of machine's rate in slot s

$$X_s^{mat} = \alpha \cdot V_s^{reac} \cdot N_s, \quad s \in [S] \quad (5.13)$$

$$V_s^{reac} - V_{s-1}^{reac} = \delta_s^{reac+} - \delta_s^{reac-}, \quad s \in [S] \quad (5.14)$$

$$\sum_j \left(p_j \cdot X_{js} + \sum_k st_{kj} \cdot Z_{kjs} \right) = V_s^{mach} \cdot N_s, \quad s \in [S] \quad (5.15)$$

$$V_s^{mach} - V_{s-1}^{mach} = \delta_s^{mach+} - \delta_s^{mach-}, \quad s \in [S] \quad (5.16)$$

To measure rates variation (δ_s^{reac+} , δ_s^{reac-} , δ_s^{mach+} and δ_s^{mach-}), the implicit formulation in equations (5.3) and (5.7) is not sufficient. Explicit decision variables (V_s^{reac} and V_s^{mach}) are needed. However, those equations become non-linear, since the explicit rate variables are multiplied by slots' lengths – see constraints (5.13) and (5.15). This issue has to be addressed by the solution method (see next section).

5.3.4 Objective function

Parameters:	
bc	Backlog cost per unit and period
sc_{kj}	Setup cost of changing from product k to j
rc	Cost of changing reactor's production rate
mc	Cost of changing machine's production rate
po	Benefit (negative cost) of producing final products

$$\begin{aligned}
 \min \sum_j \sum_t bc \cdot I_{jt}^- + \sum_j \sum_k \sum_s sc_{kj} \cdot Z_{kjs} + \sum_s rc \cdot (\delta_s^{reac+} + \delta_s^{reac-}) \\
 + \sum_s mc \cdot (\delta_s^{mach+} + \delta_s^{mach-}) - \sum_j \sum_s po \cdot X_{js}
 \end{aligned} \tag{5.17}$$

The objective is to minimize a cost function which comprises the orders backlog costs, machine sequence-dependent setup costs and penalties for changing production rates (which cause wear to equipment and break plant's stability in the reactor and machine). In addition, the maximization of the plant's output is promoted with a negative term. This last criteria seeks to make the most of installed (capital intensive) equipment, reducing the industrial cost per product unit.

5.4. Solution method: VND-LP

The problem formulated in the previous section has both binary and continuous variables and is further complicated by nonlinearity (the multiplication of the slots' lengths by production rates). Given that for large real-world instances exact solvers fail to produce good quality solutions within reasonable time, even for a linearized version of the model, we rely on heuristic methods.

The solution method is based on that proposed by [Figueira et al. \(2013b\)](#). We combine a metaheuristic which focuses on the binary variables and an exact solver for the continuous part. The metaheuristic thus iterates through different setup patterns (the solution representation, consisting only of the setup and changeover variables) and for each pattern (each solution visited), it calls the exact solver to decode the pattern into a final plan. To avoid the nonlinearity of this problem, the decoding is performed in two steps:

1. the linear programming (LP) solver is called without the rates variation part – equations (5.13)-(5.16);
2. the rates variation part is added to the model, the slots' lengths N_s are fixed (to the values obtained in step 1) and it is solved again.

This decoding procedure is more expensive than that of [Figueira et al. \(2013b\)](#), but also more accurate. Nevertheless, the number of solutions visited will be reduced. Two other factors will also contribute to this reduction. First, simulation will consume additional time. Second, the use of the *dual reoptimization* procedure will no longer be possible, since the

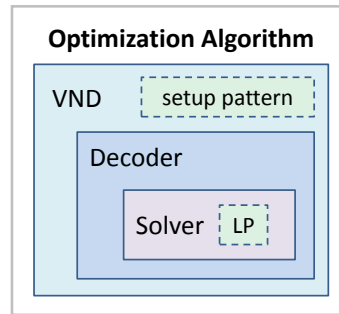


Figure 5.2: Optimization algorithm: the VND iterates through different setup patterns, whereas an exact solver is called to optimize the continuous variables of a linear program.

solution of the analytical part may worsen, even for better (more robust) plans. Therefore, the perturbation of the original variable neighbourhood search (VNS) is discarded here and the method remains a variable neighbourhood descent (VND). This method is still able to escape from local optima, since it alternates between different neighbourhood structures. Figure 5.2 exhibits the main blocks of the algorithm.

5.5. Discrete-event simulation model

The analytical model previously presented does not account for process variability or disturbances, since all the parameters are deterministic. These stochastic elements are addressed now in a simulation model, which is used to analyse the dynamics of the implementation of production plans. These dynamics, mainly those with respect to disturbances, cannot be properly evaluated with (static) Monte Carlo simulations. Therefore, we chose to build a dynamic simulation model.

In dynamic models, time can be represented in a continuous or discrete manner. The process industries are characterized by continuous materials, whose state changes continuously along the time, especially in continuous systems like the one studied here. However, at the level of detail required (and desirable) by the planning process, the system can be seen as changing its state at discrete points in time and hence, modelled by a discrete-event approach.

In our model, events can be production orders to be executed, disturbances beginning or ending, units changing their rate or processes changing their yield. These events then modify the state of the production units (entities), which is accomplished when the production task (activity) is executed. The state of an entity may consist of a given quantity of a specific product being produced, a specified production rate or simply a certain inventory level.

A general overview of the simulation process is provided in Figure 5.3. After, reading the plan, production orders (each corresponding to a time slot of the optimization model) are given one at a time, first to the machine and then to the reactor. During this time, disturbances may appear in the machine. In that case, the completion time of the current slot

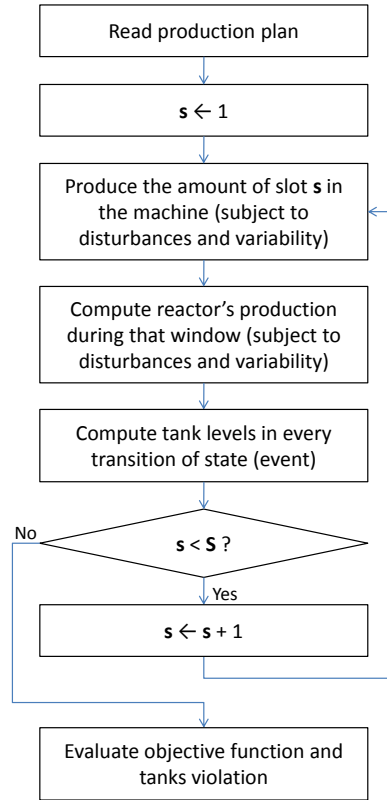


Figure 5.3: Simulation process flowchart.

will be extended, since the amounts to be produced are fixed, to properly satisfy demand. The operation of the reactor uses those time periods (in order to keep synchronization), for which given production rates are to be followed. Disturbances may also arise in the reactor. In addition, both the production and consumption of the material that flows between the two production units is subject to variability. Therefore, each period was divided into sub-periods, independently of the time slots, where those stochastic parameters assume particular values. At the end of each iteration, the tank levels are computed for every elapsed event. Thus, whenever a disturbance started or ended or a stochastic parameter changed its value, the tank level is evaluated. In this way, we make sure that any tank violation is taken into account. These violations, along with all the objectives that may have been impacted (such as production and backlog), are then returned at the end of the simulation.

To model the variability and disturbances, the following parameters were used:

- reactor's production yield and machine's consumption rate (α_i and b_{ji} , respectively – where i is the sub-period);
- time between failures and time to repair of the reactor and machine ($tb^{f^{reac}}$, ttr^{reac} , $tb^{f^{mach}}$ and ttr^{mach} , respectively).

In the presence of stochastic parameters, the simulation model has to run multiple times

and the results have to be aggregated, so that one can have a certain confidence on them. The confidence level is naturally correlated to the number of simulation replications. However, there are techniques to reduce the variance of results without increasing the number of simulations (see Subsection 5.6.4). This is particularly important for simulation-optimization algorithms, where the simulation is executed iteratively.

The model was implemented in C++, since it was of moderate complexity. A general purpose programming language like C++ allows for a very high computational performance, when compared to commercial simulation packages. In addition, it gives more flexibility in the model implementation. For instance, one has control over the sampling from the distributions, hence enabling the application of variance reduction techniques.

5.6. Hybridizing simulation and optimization

5.6.1 Simulation purpose

The simulation model can be hybridized with optimization in different ways. The main S-O approaches of the literature were identified and classified by [Figueira and Almada-Lobo \(2014\)](#). The most distinctive aspect is the purpose of simulation in the whole procedure. The authors divided this dimension in four categories:

- Evaluation Function (EF), where simulation evaluates every single solution visited by an optimization algorithm;
- Surrogate Model Construction (SMC), where simulation evaluations are used to create a surrogate (analytical) model;
- Analytical Model Enhancement (AME), where simulation feedback is applied on the refinement or extension of a problem-specific analytical model;
- Solution Generation (SG), where simulation computes part of the solution, using optimization for other computations.

The selection of the appropriate approach strongly depends on the characteristics of the problem. For small solution spaces and inexpensive simulations, EF should be the right choice, since it will always rely on more accurate simulation evaluations to select a solution. As we move to the other approaches, the number of simulations (and hence the computational time) tends to decrease and the consideration of assumptions tends to grow. SMC assumes the landscape of solutions can be represented by some analytical model, AME assumes that model is known *a priori* (but needs some refinements) and SG that the model is completely known.

In our case the analytical model needs the feedback from simulation to generate robust plans, therefore SG is not suitable. On the other hand, an EF design would result in an extremely large amount of simulations. In spite of the simulation runs being very fast, they would be called hundreds of thousands of times in real world instances. Moreover, using simulation as the evaluation function of the VND (see Figure 5.4a) would give feedback just to the optimization of the setup patterns. Indeed, in order to consider stochastic elements

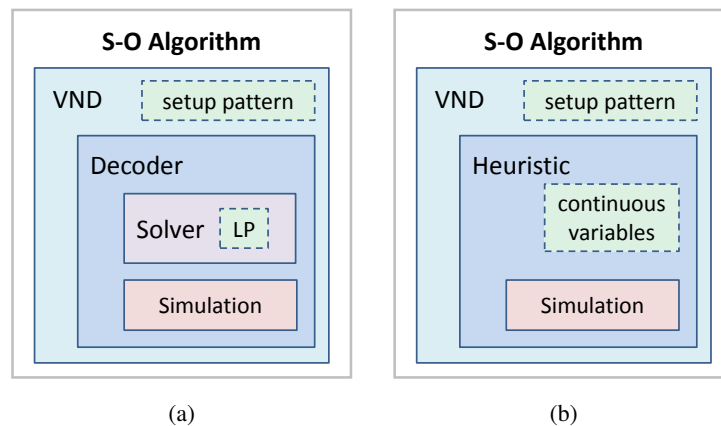


Figure 5.4: Using simulation as the evaluation function of: (a) the VND; (b) the whole optimization process.

in the optimization of the continuous variables, simulation has to be incorporated in that optimization (see Figure 5.4b). The issue of this second approach is that all the linear relationships between input and output variables are lost and the efficient LP solver can no longer be applied, being replaced by a heuristic. The same would happen with SMC methods, which in addition would not use the problem-specific neighbourhood structures of the VND.

Therefore, enhancing the linear programming model with the simulation's feedback seems to be the most appropriate approach. One key aspect of these AME methods is the choice of the parameters to be refined in the analytical model. Indeed, they do not have necessarily to correspond to the stochastic parameters listed in the previous section, especially because there is no direct correspondence of the TBFs and TTRs in the analytical model. The impact of these disturbances (and of the process variability) will be primarily on the tank level. Only if it reaches one of the limits, then it will force the production units to change their rates (or even to stop). Thus, we consider slacks in the tank, so that the system is able to accommodate some variability and disturbances, without the need for drastic actions. The consideration of slacks is indeed an industrial practice. However, here those slacks will be optimized with the simulation's feedback.

5.6.2 Refinement process

Depending on the problem and on the parameters to be refined, the refinement process can be more or less complex. For instance, [Albey and Bilge \(2011\)](#) approached a deterministic, yet complex, production system and used simulation to check if the capacity considered in the analytical model was appropriate. In this case the refinement is straightforward: the capacity is changed to match the simulation result.

Stochastic problems usually require a more sophisticated process. [Almeder et al. \(2009\)](#) aggregated the simulation results of multiple replications not by average, but by quantiles of 50, 70 and 90%. The 90%-quantile resulted better for most of the instances,

but in a few of them the 70% appeared as more suitable. The S-O algorithm was not able however to alternate between these values to evaluate the best refinement strategy. The quantile was a fixed input parameter.

Jung et al. (2004) on the other hand devised an outer optimization layer, which applied a gradient search procedure to refine the safety stock levels used in the analytical model. The refinement of the safety stocks of the different products and facilities was performed in an aggregated way, i.e. all the parameters were changed simultaneously in a specific direction (either increasing or decreasing).

In our case, we apply a heuristic process which refines the upper and lower tank slacks independently. According to the violations observed in the simulation, one of the slacks may increase while the other decreases. In this way, the slacks correctly weigh the variability and disturbances occurring up and downstream.

5.6.3 S-O structures

The refinement process may be executed at different moments. Three alternatives are here identified (illustrated in Figure 5.5):

- refining at the beginning of the optimization (a pure sequential procedure);
- refining at the end of the optimization, in an iterative process;
- refining during optimization (progressive refinement).

The first structure is the most simple. Once the simulation process (including multiple replications) is finished, it is not called again, giving way to the optimization. This procedure is computationally less expensive than the other approaches. However, if the optimal slacks are dependent on the solution (namely the production sequence), the sequential approach may not be effective. In that case either the iterative or the progressive methods would be more suitable. The latter seeks to save computational time by performing simultaneously the convergence to a solution and the refinement of the analytical model.

5.6.4 Variance reduction

In order to increase the confidence on the simulation results without increasing the number of simulations (which results in higher computational effort), variance reduction (VR) techniques are applied. We combine here two of these techniques: Latin hypercube sampling (LHS) (McKay et al., 2000) and antithetic variates (AV) (Hammersley and Morton, 1956).

The first technique consists of defining n strata for each of the m stochastic parameters, where all the strata have the same probability of occurrence. Then, n scenarios are generated, each sampling from a distinct stratum of each parameter. For instance, for $n = 2$ and $m = 3$, two scenarios could be created as follows:

1. sampling from the second, first and second strata of the first, second and third parameters, respectively;

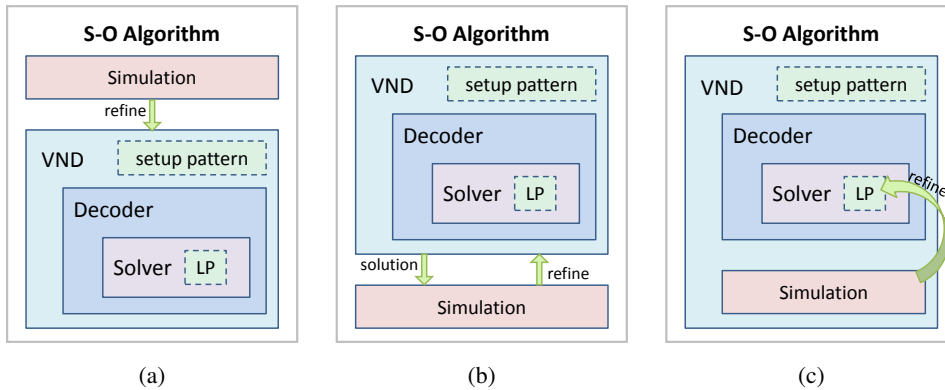


Figure 5.5: Using simulation to refine the LP, giving feedback to the whole optimization process: (a) at the beginning; (b) iteratively; (c) progressively.

2. sampling from the first, second and first strata of the first, second and third parameters, respectively.

In this way we ensure the scenarios are well distributed in the probability space and the sample is more representative. LHS has the advantage over the standard stratified sampling of resulting in only n scenarios (rather than $n \cdot m$, if all the combinations were considered).

On top of LHS, we apply AV, which establishes the use of symmetric random seeds in the sampling process. This means that for every sample path obtained, its antithetic path is also considered, thus requiring the definition of an even number of strata in LHS.

5.7. Case study and preliminary experiments

This section describes a case study where the proposed S-O method is applied. First, the processing of the input data is presented and then, the results of preliminary experiments are exposed. These experiments seek to perform a first assessment of the S-O method proposed.

The case study is conducted on an integrated pulp and paper mill. These plants comprise multiple production stages and a complex production flow structure. We focus on the two most critical production resources: the pulp digester and the paper machine, as well as the intermediate tank in between. This simplified system was indeed the inspiration of the problem described in Section 5.2. Both production units are subject to disturbances and the process is characterized by high variability.

We started by collecting data from the plant regarding all the uncertainty in the process. Then, different probability distributions were approximated to the data. The procedure consisted of the following steps:

1. general characterizing the underlying distribution, by computing summary descriptive statistics and histograms;

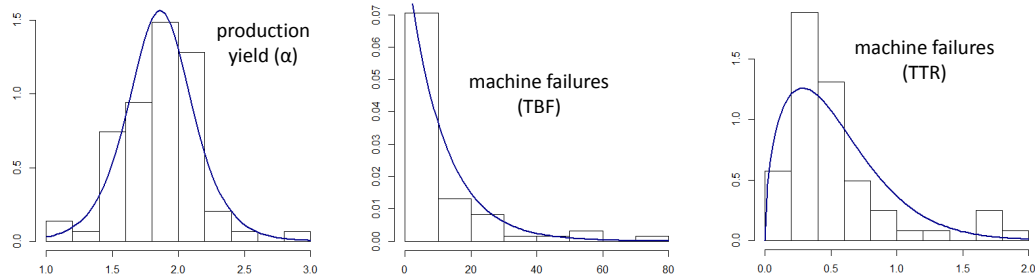


Figure 5.6: Approximation of probability distributions to real data of process variability and disturbances.

2. selecting a set of candidate distributions and fitting them to the data using the method of maximum likelihood;
3. determining which of the fitted distributions best represents the data using Kolmogorov-Smirnov tests.

The final probability distributions and the histograms of the original data are illustrated in Figure 5.6. These distributions were then introduced in the simulation model and the average values (regarding process variability) were used in the analytical model.

In order to have a first validation of the effectiveness of the S-O method in generating robust plans, a comparison is conducted relative to a non-proactive approach. Figure 5.7 shows the performance of both methods in the main KPIs. By sacrificing particular objectives, such as the orders backlog, the S-O algorithm is able to generate plans that in the simulation result in less over and underflows of the tank level. These violations in practice would imply the adaptation of the plan, particularly the execution of abrupt changes to production rates and consequently loss of productivity. Therefore, the penalization given to the tank violation (in the objective function) corresponds to those adaptation costs. Overall, the method was able to improve by 89% the tank violation. Nevertheless, it would be important to extend this computational study to more production settings, in order to obtain a better understanding of the impact of the S-O method.

The production plan used in these experiments is illustrated in Figure 5.8. The non-proactive approach does not consider slacks in the tank level and hence takes advantage of the wider limits to contract the production cycle. The S-O method on the other hand, extends the grades cycle, resulting in a smoother variation of the tank level, which does not approach the limits. Operating with these slacks in the tank level, the system is more prepared to accommodate possible disturbances that may occur either upstream (in the digester) or downstream (in the paper machine).

5.8. Conclusions and further research

Simulation-optimization appears to be a very promising tool to approach planning and scheduling problems subject to process variability and disturbances. Discrete-event sim-

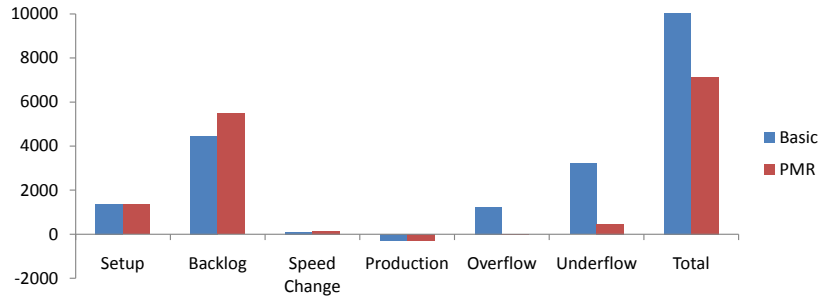


Figure 5.7: Comparison between the S-O algorithm and the non-proactive approach, with respect to the KPIs.

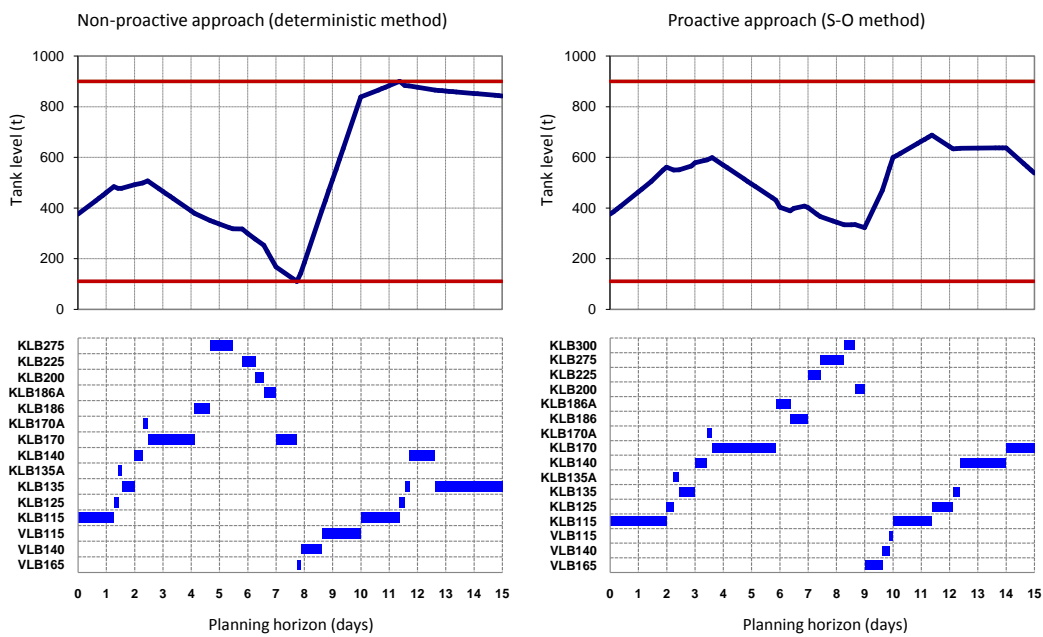


Figure 5.8: Comparison between the S-O algorithm and the non-proactive approach, with respect to the generated plans.

ulation is able to accurately model these different types of uncertainty. Moreover, this is performed with no mathematical sophistication and hence the computational time does not grow exponentially.

Nevertheless, the selection of an appropriate S-O design can have a huge impact on the efficiency of the methods. Having an existing analytical model that represents the system with reasonable detail is crucial for taking advantage of the problem structure and executing simulation only to refine key parameters (instead of using it as a time-consuming evaluation function).

Our preliminary experiments validated the effectiveness of the proposed S-O method in generating robust plans. Still, there is a long research path on proactive planning for continuous systems. First, it would be important to conduct a more comprehensive study, considering different production settings and evaluating the impact of the S-O method. The method should also be compared to proactive approaches that define tank slacks *a priori*.

Furthermore, the particular S-O design should be benchmarked against the other S-O variants identified, namely those refining the slacks only at the beginning or in an iterative way. The advantage of one approach over the others may depend on the problem settings. Therefore, the features of the problem that favour a given strategy should also be identified.

Finally, statistical selection methods can be explored, in order to validate the search of the optimization algorithm and possibly reducing the number of simulations needed. Additional variance reduction techniques may also help enhancing further this aspect.

Bibliography

- Erinç Albey and Ümit Bilge. A hierarchical approach to fms planning and control with simulation-based capacity anticipation. *International Journal of Production Research*, 49(11):3319–3342, 2011.
- Christian Almeder, Margaretha Preusser, and Richard Hartl. Simulation and optimization of supply chains: alternative or complementary approaches? *OR Spectrum*, 31:95–119, 2009.
- J Balasubramanian and IE Grossmann. A novel branch and bound algorithm for scheduling flowshop plants with uncertain processing times. *Computers & chemical engineering*, 26(1):41–57, 2002.
- V. Camargo, F. Toledo, and B. Almada-Lobo. Three time-based scale formulations for the two-stage lot sizing and scheduling in process industries. *Journal of the Operational Research Society*, 63:1613–1630, 2012.
- Pedro M Castro, Adrián M Aguirre, Luis J Zeballos, and Carlos A Méndez. Hybrid mathematical programming discrete-event simulation approach for large-scale scheduling problems. *Industrial & Engineering Chemistry Research*, 50(18):10665–10680, 2011.
- 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*, 2014.

- 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.
- Muge Erdirik-Dogan and Ignacio E Grossmann. Simultaneous planning and scheduling of single-stage multi-product continuous plants with parallel lines. *Computers & Chemical Engineering*, 32(11):2664–2683, 2008.
- G. Figueira and B. Almada-Lobo. Hybrid simulation–optimization methods: A taxonomy and discussion. *Simulation Modelling Practice and Theory*, 46:118–134, 2014.
- Gonçalo Figueira, Marcos Furlan, and Bernardo Almada-Lobo. Predictive production planning in an integrated pulp and paper mill. In *Manufacturing Modelling, Management, and Control*, volume 7, pages 371–376, 2013a.
- Gonçalo Figueira, Maristela Oliveira Santos, and Bernardo Almada-Lobo. A hybrid vns approach for the short-term production planning and scheduling: A case study in the pulp and paper industry. *Computers & Operations Research*, 40(7):1804–1818, 2013b.
- B. Fleischmann and H. Meyr. The general lotsizing and scheduling problem. *OR Spectrum*, 19:11–21, 1997. ISSN 0171-6468.
- J. M. Hammersley and K. W. Morton. A new monte carlo technique: antithetic variates. *Mathematical Proceedings of the Cambridge Philosophical Society*, 52(03):449–475, 1956.
- I. Harjunkoski, R. Nyström, and A. Horch. Integration of scheduling and control — theory or practice? *Computers & Chemical Engineering*, 33(12):1909–1918, 2009.
- Stacy L Janak, Xiaoxia Lin, and Christodoulos A Floudas. A new robust optimization approach for scheduling under uncertainty: Ii. uncertainty with known probability distribution. *Computers & chemical engineering*, 31(3):171–195, 2007.
- June Young Jung, Gary Blau, Joseph F. Pekny, Gintaras V. Reklaitis, and David Eversdyk. A simulation based optimization approach to supply chain management under demand uncertainty. *Computers & Chemical Engineering*, 28(10):2087–2106, 2004.
- Anna Lindholm, Charlotta Johnsson, Nils-Hassan Quttineh, Helene Lidestam, Mathias Henningsson, Joakim Wikner, Ou Tang, Nils-Petter Nytzén, and Krister Forsman. Hierarchical scheduling and utility disturbance management in the process industry. In *Manufacturing Modelling, Management, and Control*, volume 7, pages 140–145, 2013.
- T. Majozi and X.X. Zhu. A combined fuzzy set theory and {MILP} approach in integration of planning and scheduling of batch plants—personnel evaluation and allocation. *Computers & Chemical Engineering*, 29(9):2029 – 2047, 2005.
- M. D. McKay, R. J. Beckman, and W. J. Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 42(1):55–61, 2000. ISSN 0040-1706.

- Kaushik Subramanian, Christos T Maravelias, and James B Rawlings. A state-space model for chemical production scheduling. *Computers & chemical engineering*, 47:97–110, 2012.
- 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.
- Martina Wittmann-Hohlbein and Efstratios N Pistikopoulos. Proactive scheduling of batch processes by a combined robust optimization and multiparametric programming approach. *AIChE Journal*, 59(11):4184–4211, 2013.
- 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.

Conclusions and future work

This thesis approaches the operational production planning and scheduling of continuous integrated pulp and paper mills. The work is driven by a case study from a P&P producer. The research conducted in this thesis is divided in two main branches: reactive and proactive planning. Chapters 2 and 3 are dedicated to the first branch. Here, problems are formulated as deterministic and the focus is on obtaining good quality solutions in short periods of time. Proactive planning on the other hand seeks to address uncertainty in order to produce robust plans, which can accommodate variability and possible disturbances. Given that the deterministic problems approached here are already very complex (exact methods fail to provide reasonable solutions in feasible time), the promising combination of simulation and optimization is studied, first in general and then in the specific context of planning and scheduling in Chapters 4 and 5, respectively. The approaches developed for reactive planning may also be proactive, in case they consider slacks. However, those slacks would have to be defined *a priori*, not reflecting correctly the trade-off between optimality and robustness. Likewise, a proactive approach may be used in reactive planning, but it would not be so effective (since it will need more computational time). The reactive and proactive methods developed in this research may be adapted to other similar process industries, especially those with continuous processes and shifting bottlenecks among production resources. This chapter overviews the main results and contributions of the thesis and pinpoints directions for ongoing and future research.

The PhD project starts by focusing on reactive approaches. All the production parameters are assumed to be deterministic, in order to maximize the efficiency of the methods. Still, the models need to include the relevant features of the production process, such as sequence-dependent setup times and costs, integration of multiple stages (to take into account shifting bottlenecks) and variable production rates. First, an efficient solution method to tackle this complex problem is developed. The method consists of a variable neighbourhood search (VNS), hybridized with a linear programming solver and a specific heuristic for the discrete speed rates. Starting with a constructive heuristic which devises a simple plan, the VNS is able to quickly improve the initial solution, searching alternately in different neighbourhood structures, specifically designed for lot-sizing and scheduling problems. The exact solver plays an essential role, since it efficiently addresses the continuous variables (with the simplex method enhanced by the dual re-optimization technique). Therefore, the method keeps its generality and optimality regarding continuous decisions, while taking advantage of the problem structure for the combinatorial (harder) part of the problem. When compared to Cplex, the method revealed a reasonable performance on the smallest instances, with optimality gaps of 2 to 3%. On the largest (real world sized) instances, the VNS clearly outperformed both Cplex and the original MIP-based heuristic

and was able to find reasonable solutions much faster. The proposed method can be easily adapted to approach other small bucket lot-sizing and scheduling problems (linear constraints are incorporated without any required modification), but will be especially relevant for problems with independent continuous decisions (i.e., after the binary variables have been fixed, continuous decisions still need to be optimized), such as the general lot-sizing and scheduling problem (GLSP) and the proportional lot-sizing and scheduling problem (PLSP). The method is a significant contribution to the literature, since not only it outperformed the existing approaches in that specific problem, but also contains novel ideas for metaheuristics approaching these small-bucket problems.

The second objective of the thesis is the development of a decision support system (DSS) to be used in practice. For that purpose, the mathematical model is completely reformulated. The main issue is the consideration of fixed campaigns. This is one of the most important requirements, since the first campaigns to be produced have been previously optimized for the cutting stage. Moreover, it gives the DSS the ability to test manually devised campaigns or work as a hybrid (consider part of a manual plan and optimize the remaining). The incorporation of this feature suggests the use of a continuous time formulation where production slots can cross discrete time periods. In this way, one campaign is represented by just one slot, thus avoiding the need to heuristically determine the number of slots for each campaign. Other important industrial features include different priority levels for orders, variable production rates in all units and scheduled stoppages. The new formulation needs less binary variables for setups, but introduces additional ones for the intersection of the time grids. Therefore, the adaptation of the proposed VNS is not straightforward and hence, a MIP-based heuristic is applied. The method is wrapped by pre- and post-processing steps, which essentially deal with the aggregation and disaggregation of production orders, and includes a post-optimization phase, where production rates are smoothed. When compared to manual plans, the system's plans have shown all the desirable characteristics and proved to be superior in the most relevant KPIs. The DSS is currently in function in the case study company. The main contribution of this DSS is related to bridging the gap between research and practice, by bringing more insights from the industry (in terms of operational issues), investigating the ability of the existing models in the literature to deal with those issues and showing how state-of-the-art models and methods can be applied in practice. Moreover, it includes features which are essential for reactive planning (and were not considered by the previous method), such as the possibility of modelling stoppages (which can be used to characterize disturbance situations or scheduled maintenance) and fixing campaigns (indispensable for rolling-horizon frameworks).

From the aforementioned work two research papers have resulted:

- G. Figueira, M.O. Santos, B. Almada-Lobo. A hybrid VNS approach for the short-term production planning and scheduling: A case study in the pulp and paper industry. *Computers & Operations Research*, 40(70):1804–1818, 2013.
- G. Figueira, P. Amorim, L. Guimarães, M.A. Lopes, B. Almada-Lobo. A decision support system for the operational production planning and scheduling of an integrated pulp and paper mill. *Submitted to Computers & Chemical Engineering*, 2014.

In the proactive perspective, simulation-optimization (S-O) methods appear to be promising. However, the existence of a vast literature on these methods and the lack of a comprehensive classification motivated an in-depth study of this field. Most of the work on S-O naturally results from two main disconnected communities, which contribute with different methods and approaches. The Simulation community tends to rely more on simulation models and hence uses them as evaluation (or fitness) functions of an optimization algorithm, or at most to evaluate multiple solutions which serve to approximate a metamodel. The Optimization community on the other hand is more willing to formulate *a priori* analytical models which are specific of the problem, thus taking advantage of its structure. Simulation is then used to enhance or complement those models. The purpose of simulation is what distinguishes the main S-O approaches. The second key dimension is the hierarchical structure that relates simulation with optimization. The matrix that results from combining these two dimensions allows to put the main S-O methods in perspective, which may help further understanding their differences and similarities, as well as identifying possible designs not explored yet. Other two dimensions, more related to the search design, also improve the comprehension of the different methods. The first (search method) is indeed very close to the way the literature has been classifying S-O approaches, while the second (search scheme) complements it by examining the alternation between the solution and probability spaces. Exploring each of these dimensions, different S-O designs are related to specific problem characteristics. The study thus contributes to the literature with a taxonomy that contains key dimensions disregarded by previous classifications, general guidelines concerning the use of S-O methods for different problems, insights into the cross-fertilization of the ideas applied in distinct S-O methods and a standard for a better communication in the S-O community.

The overview of S-O methods paves the way for an approach to proactive production planning and scheduling. The main S-O designs are discussed in the context of these problems and a novel combination of simulation and optimization is proposed, where the analytical model is refined during the optimization process. The analytical model is deterministic, but considers the main features of the problem, like the previous reactive approaches. Simulation is then responsible for evaluating how plans behave in a stochastic environment, subject to process variability and disturbances. The latter are accurately modelled with events for their start and end. This discrete-event simulation model provides feedback on the slacks used in the intermediate tank, so that the optimization algorithm is able to generate good quality and robust solutions. The main contribution of this chapter is an S-O framework that is applicable to a range of planning and scheduling problems, especially in multi-stage scenarios, where the bottleneck may shift between stages.

The study on S-O has resulted in two research articles and in two proceedings:

- G. Figueira, B. Almada-Lobo. Hybrid simulation–optimization methods: A taxonomy and discussion. *Simulation Modelling Practice and Theory*, 46:118–134, 2014.
- G. Figueira, B. Almada-Lobo. Simulation-optimization for planning and scheduling of two-stage continuous systems subject to variability and disturbances. *Working paper*, 2014.

- G. Figueira, B. Almada-Lobo. Robust Production Planning and Scheduling of a Pulp Digester and a Paper Machine. *Computer Aided Chemical Engineering*, 33:499-504, 2014.
- G. Figueira, M. Furlan, B. Almada-Lobo. Predictive production planning in an integrated pulp and paper mill. *IFAC Proceedings Conference on Manufacturing Modelling, Management, and Control*, 371–376, 2014.

Ongoing research is comparing the performance of the proposed S-O method to other variants, which call simulation at different points in the algorithm. Running simulation only at the beginning can be more straightforward and efficient. However, the proper values of the model's parameters to be refined (in this case, the tank slacks) may depend on the solution being simulated. Therefore, the values obtained with the initial solution will not be valid for the final one. An iterative procedure should be able to address that issue, but can be time consuming. The progressive refinement strategy aims at combining the best of both worlds, converging to a solution with increasing model's accuracy. Nevertheless, its effectiveness should be validated by comparing to the aforementioned alternatives using different problem settings. The features of the problem that favour a given strategy should also be identified. In addition, statistical selection methods should be applied, in order to validate search moves and possibly reducing the number of simulations.

With respect to reactive approaches, the comprehensive formulation proposed here, when approached with the MIP-based heuristic, is already providing solutions that are good enough for practice. Still, there is a long way to go until a reasonable optimality gap can be achieved. Indeed, even the linear relaxation is problematic for current commercial solvers. The VNS proposed could not be applied directly, due to the additional binary variables that emerged from practical requirements. Therefore, it would be necessary to include in the VNS a heuristic to deal with that issue or adapt the MIP-based heuristic to incorporate some of the ideas of the VNS, such as adding/removing campaigns to/from certain positions. On the other hand, the formulation may still be strengthened with additional valid inequalities or reformulations. In addition, the model can still be enhanced with respect to reactive planning. In a serious disturbance situation, the fixed campaigns may no longer be hard constraints. Reactive planning should then weigh the cost of adapting the original plan to converge the system towards recommended operating levels.

Finally, the coordination of proactive and reactive approaches is also a relevant subject for future research. Determining the periodicity or the events that should trigger reactive approaches and whether there is enough time to properly look at preventive issues are important questions to be investigated. The trade-off between solution quality and computational time is always present, but it is also constantly evolving over time.

