FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

A Hybrid Algorithm for the Integrated Production Planning in the Pulp and Paper Industry

Luís Gonçalo Rodrigues Reis Figueira

Master in Industrial Engineering and Management

Supervisor: Bernardo Sobrinho Simões de Almada-Lobo (Assistant Professor of Faculdade de Engenharia da Universidade do Porto)

Co-supervisor: Maristela Oliveira dos Santos (Assistant Professor of Universidade de São Paulo)

July 2011

Abstract

Within a capital and energy intensive industry, material closed loops in integrated production processes is the most effective way for pulp and paper (P&P) mill companies to reduce energy and resource consumption and hence, tackle both economical and environmental issues. This integration makes the production planning process much more complex, with more conflicting objectives and constraints and eventually multiple bottlenecks. Production planning is therefore a major challenge P&P companies have to face in order to streamline their operation costs and survive in an increasing competitive market.

Motivated by a Portuguese company case study, this dissertation tackles the optimization of the short-to-medium term productivity in the most cost-effective manner by dealing with lotsizing and scheduling under an integrated approach, taking into account various critical and interdependent production units. Such industrial features have been ignored by researchers not only because of modelling complexity, but also for solvability reasons.

In this dissertation, a novel mathematical formulation, which do accomplish an important integration level, is refined and a solution approach that combines heuristics with mathematical programming is developed. The heuristic algorithms work with the paper machine setup variables to find promising schedules of production campaigns, while the fixed mixed integer programming (MIP) formulation optimizes the continuous part. A specific heuristic is also developed to determine the discrete digester's speeds. The solution method is tested against a state-of-the-art exact solver, in instances based on real data from the case study.

The final results are satisfactory, since the method proves to be superior in real world sized instances, both in solution quality and computational time. Future work may include the study of other neighbourhood structures and the development of ways to speed up the evaluation of neighbours, as well as the validation with the company of the Algorithm and its generated production plans.

Resumo

Numa indústria de capital e energia intensivos, os ciclos fechados de matérias na integração de processos produtivos representam a forma mais eficaz das empresas de produção de celulose e papel reduzirem o consumo de energia e recursos e, assim, responderem às problemáticas económica e ambiental. Esta integração torna o processo de planeamento da produção muito mais complexo, com mais objectivos conflituosos e eventualmente múltiplos gargalos. O planeamento da produção é portanto um desafio central que as empresas de celulose e papel têm de enfrentar, de modo a poderem reduzir ao máximo os seus custos produtivos e permanecerem num mercado cada vez mais competitivo.

Motivada por um caso de estudo de uma empresa portuguesa, esta dissertação visa a optimização da produtividade de curto-médio prazo da forma mais económica possível, lidando com o dimensionamento e sequenciamento de lotes numa abordagem integrada, tendo em linha de conta várias unidades produtivas críticas e interdependentes. Este tipo de características industriais tem sido ignorado pelos investigadores não apenas devido à complexidade de modelação, mas também por questões de solvabilidade.

Na presente dissertação, é refinada uma formulação matemática que efectivamente alcança um nível de integração significativo e é desenvolvida uma abordagem que combina heurísticas com programação matemática. Os algoritmos heurísticos trabalham com as variáveis de *setup* para encontrar sequências promissoras de campanhas produtivas, enquanto a formulação fixa de programação inteira mista (PIM) optimiza a parte contínua. É também desenvolvida uma heurística específica para determinar as velocidades discretas do digestor de celulose. O método de solução é testado em relação a um *solver* de referência, em instâncias baseadas em dados reais provenientes do caso de estudo.

Os resultados finais são satisfatórios, já que o método demonstra ser superior em instâncias de dimensão real, quer em termos de qualidade da solução, quer em termos de tempo computacional. Trabalhos futuros poderão incluir o estudo de novas estruturas de vizinhança e o desenvolvimento de formas de acelerar o processo de avaliação de vizinhos, assim como a validação, com a empresa, do Algoritmo e dos planos por ele gerados.

Acknowledgements

First and foremost, I would like to thank my advisors Professor Bernardo Almada-Lobo and Professor Maristela Santos for their assistance during this work. Professor Maristela Santos provided me with great support at the hardest part of the project: the beginning. Professor Bernardo Almada-Lobo guided me with his valuable insights, strong motivation and high optimism.

I owe also thanks to my lab colleagues, who are actually true friends, for their help in many problems a project of this type necessarily involves.

Finally, I would like to thank my family. Thank you João and Francisco for helping me building the attitude I have towards study and work. Thank you Mãe and Pai, for your unconditional support.

"Good management is the art of making problems so interesting and their solutions so constructive that everyone wants to get to work and deal with them."

Paul Hawken

Contents

List of Figures

List of Tables

Acronyms

CSLP	Continuous Setup Lotsizing Problem
DLSP	Discrete Lotsizing and Scheduling Problem
DSS	Decision Support System
GDP	Gross Domestic Product
GHG	Greenhouse Gas
GLSP	General Lotsizing and Scheduling Problem
GRASP	Greedy Randomized Adaptive Search Procedure
KPI	Key Performance Indicator
LSS	Lotsizing and Scheduling
LS	Local Search
LP	Linear Programming
MIP	Mixed Integer Programming Problem
P&P	Pulp and Paper
PLSP	Proportional Lotsizing and Scheduling Problem
PPS	Production Planning and Scheduling
SA	Simulated Annealing
TS	Tabu Search
VLSN	Very Large-Scale Neighbourhood
VNS	Variable Neighbourhood Search

Chapter 1

Introduction

1.1 Motivation and Objectives

In an increasing competitive global economy, companies have to constantly improve and develop central skills that match the critical success factors of the markets where they belong to. Process industries are characterized by high capital intensity, low flexibility and make-to-stock policy and its products have high demand volumes, where price and delivery guarantee are the order winners. Production planning is thereby the core of competitiveness in these type of companies, since costs reduction, by means of maximizing the production resources utilization, is the primary way to increase competitiveness. Being in the middle of continuous and batch process industries, the pulp and paper (P&P) industry adds a higher introduction rate of new products and a more complicated scheduling. The P&P industry is therefore one of the most challenging from the production planning perspective.

As a result, the market strategy success strongly depends on the production efficiency, since supplying a wide range of products that meet specific customers demands, with competitive prices and strict due dates can only be achieved with superior planning. Moreover, to really have the market strategy guiding production decisions, an effective planning system that considers the set of defined KPIs (Key Performance Indicators) is required. This can be accomplished with a decision support system (DSS), which incorporates efficient algorithms for solving the production planning problem.

Many papers in the literature have tackled optimization problems in the P&P industry. However, almost all of them adopted non-integrated approaches, which result in sub-optimal plans. The work [1] is one of the few that tried an integrated approach and the only one known for doing it with three of the most important and interrelated production stages: pulping, papermaking and chemical recovery. This approach is thus the only one to model the dynamic mass and material balance of an integrated P&P mill and to analyse the trade-off that arises when solving pulp planning and paper scheduling problems. However, the computational complexity of the proposed formulation is too high for exact methods to obtain good solutions in large real-world instances and within reasonable time. Even iterative MIP-based heuristics, such as relax-and-fix and fix-and-optimize, may fail to achieve the desired goals.

The objective of this dissertation is then to create an efficient solution procedure for the integrated production planning (of pulping, papermaking and chemical recovery), combining exact methods with metaheuristics in order to achieve good results within reduced computational time. Keeping an exact module in the algorithm makes it more flexible and easier to be adapted to model extensions. This is an important feature, since companies are constantly changing their requirements and thus constraints may be added or removed frequently. Moreover, different plants may have different specificities, with different constraints. Therefore, a flexible algorithm can be easily adapted for solving different problems. Additionally, as the solution procedure becomes faster, more detail may be simply introduced into the formulation, in order to obtain more realistic plans.

The algorithm that results from this work may be incorporated in a DSS, which should help P&P companies in their production planning process. This dissertation is motivated in particular by a real Portuguese company, which will be presented as a case study in this work. Nevertheless, the algorithm should have a generic structure in order to be suitable for different companies, with the possibility to add problem-specific heuristics or constraints so as to cope with the specificities of each company or plant. For this reason it was decided to combine metaheuristics with exact methods.

The exact part is based on a mathematical formulation. Therefore, the model proposed by [1] has to be validated and modified, if necessary. Then, a solution procedure based on the resulted model is to be developed.

The project will allow integrated P&P companies to fully optimize short-to-medium term productivity in the most cost-effective manner by dealing with production planning and scheduling problems (PPS) under an integrated approach. Additionally, by running the optimization procedure with different input data, they will be able to perform what-if analysis to evaluate alternative strategies in acquisition and expansion of production resources.

The algorithm may further integrate additional modules, such as the optimization of the cooking parameters in pulping or the cutting stock optimization, cooperating with them in their specific problems.

1.2 Dissertation Synopsis

The motivation and objectives of this dissertation were presented in the previous section. The work developed to achieve these objectives is divided into six chapters, presented below.

Chapter 2 introduces the pulp and paper industry. The production process is detailed and explored in order to understand its complexity and difficulty for the planning process. Then, on the one hand, the new challenges for production planning are presented and, on the other hand, the work developed by research community is reviewed. Basing on both needs (arising with the new challenges) and current developed work, the opportunity and relevance of developing an integrated approach for the production planning in P&P industry is analysed.

Chapter 3 gives some theoretical basis for the optimization approach of production planning problems. It is thus presented an overview of various lotsizing and scheduling models and then, generic solution methods (exact methods, heuristics and metaheuristics and hybrid methods) are briefly explored and discussed. The approach to be adopted, for solving the planning problem proposed in this dissertation, is built on top of the theoretical concepts introduced in this chapter.

In Chapter 4 a case study of a real Portuguese P&P company is discussed. Its planning problem is the focus of this dissertation, although the aim is to conceive it as flexible as possible. The various specificities of both the process and the planning are explored, as well as the model already developed by [1].

The whole solution procedure for the defined problem is addressed in Chapter 5. A new formulation is developed, based on the model proposed by [1], validating existing constraints, adding new constraints that better represent company's reality and modifications that turn the model more efficient for exact methods. Then, the approach for creating the algorithm is presented, as well as its structure and its different modules, which are explored in terms of their behaviour and the way they answer to the requirements. The modules include a constructive heuristic, a heuristic to constrain digester's speed variations and the local search procedure. Finally, the overall algorithm framework is presented.

Chapter 6 presents the computational results of all the tests performed with the different models and the proposed solution method, in order to check and validated them. The input instances are derived from real data from the Portuguese company of the study case. The instances vary in size (of some elements such as the number of products, planning periods and sub-periods), available capacity scenario and type of objective function (weighting more setups, backlog or overall productivity). Also, the production plan that results from the proposed solution method is analysed.

Finally, in Chapter 7 conclusions regarding the dissertation are made and directions for future work are pointed out. The suggested future work include both academic and practical activities.

Chapter 2

The Pulp And Paper Industry

2.1 An Overview

The pulp and paper industry is a major process industry in the current global economy. In Portugal, it accounts for 7% of the country's industrial production and 1.5% of Gross Domestic Product (GDP), being a modest production volume compared to some European countries [2].

Different plant categories exist depending on whether they only produce pulp (pulp mills) for further processing or only paper out of purchased pulp and/or recycled waste paper (paper mills). The third category, the so-called integrated pulp and paper mills, combines the two processes and is most common in the paper industry [3].

Pulp and paper (P&P) production is not only capital intensive, but 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. Therefore the energy content of the different paper grades is comparable to that of other energy intensive products, such as cement or steel [4]. 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 [5]. Therefore, it is of particular interest in the context of both local and global environmental discussions [3].

Over the past few years, this industry has considerably reduced its GHG emissions by introducing energy conservation projects and by increasing its use of biomass as an energy source, assumed by internationally accepted definitions to be CO_2 neutral. In an integrated P&P mill this energy is provided by the chemical recovery stage, where a sub-product of the pulping stage is concentrated and then incinerated to generate steam and to recover chemicals used in pulp production. It is seen that this type of closed loops represent the most effective approach to save both energy and resource consumption and at the same time to decrease all kind of waste production [5]. These loops allow also to reduce costs and increase efficiency, which are key aspects in a capital intensive industry. However, the production planning becomes more complex with this process integration, as there are more conflicting objectives and constraints and eventually multiple bottlenecks.

Besides chemicals recovery, the closed loop approach may include paper recycling, different types of substances re-use during production processes (e.g. water) and co-production [5]. These closed loops exist in a greater extent in integrated P&P mills, so this type of plants are more capable to improve their competitiveness, while satisfying the increasing environmental constraints. On the other hand, a higher level of competitiveness can only be achieved with satisfactory planning, which can deal with the greater system complexity.

Another challenge that is arising in the P&P industry is the incorporation of an increasing portion of recycled fibres in paper products. The current pressures to reduce both supply costs and environmental impact make customers demand lighter products (lower grades), with a much higher incorporation of recycled pulp. Therefore, P&P companies have to develop new technologies and processes to produce types of paper that can meet these requirements, while keeping paper properties, such as tightness and resilience, in acceptable levels. On the other hand, the way the different resources are balanced, as well as the standard sequence of campaigns used in the paper machine may no longer produce satisfactory (if feasible) results. New bottleneck resources may arise and capacity investments in some of them may become mandatory [1].

2.2 The Pulp and Paper Production Process

The P&P industry converts fibrous raw materials into pulp, paper and paperboard and may produce energy to be sold to electrical companies. In a first step raw materials are processed into pulp and in a second step paper and paper products are produced out of this pulp. As seen before, these two steps can be processed in separate plants, but the combination of these two steps on site is the most frequent scenario worldwide [3]. The workflow of the integrated production consists of seven interdependent sub-processes: wood preparation, pulping, bleaching, chemical recovery, papermaking, paper recycling and cutting. As in most of process industries, the first stages are continuous, ending in a discrete production phase (in this case the cutting stage).

In the first stage the wood is stripped and reduced to small chips, suitable for subsequent pulping operations and which, after screening, are sent to storage for later use. These debarking and chipping processes require little energy [3]. From the storage, the selected pieces of wood are transported into one or several continuous digesters to go through the pulping process. The pieces of wood discarded in the screening process are often crushed and sent to a boiler that burns them to produce steam. The wood used in P&P mills mainly comes from softwood, such as pine, but hardwood (as eucalyptus) can also be used [1].

Pulping is the process by which the wood is ground and reduced to a fibrous mass, separating the fibres from each other and suspending them in water. Pulping breaks apart the wood fibres and cleans them of unwanted residues. The ratio of wood to other materials used for pulp depends on the resources available. The remaining fibre is provided by recycled materials or by non-wood plant sources [3].

There are three main types of pulping: chemical, mechanical and chemical-mechanical. In chemical pulping wood chips are cooked inside the digester in an aqueous solution with reagents at high temperature and pressure. Chemical processes dissolve most of the glue that holds the fibres together (lignin) while leaving the cellulose fibres with relatively low detriment of their properties. The Kraft process, which is the most common (representing 68% of all pulp production), uses a sodium hydroxide and sodium sulphide solution. This process results in high quality paper with less impurities, while a wider variety of wood species may be used [5]. However, the pulp yield is only 40%-60% of the weight of the dry wood. The sulphite process uses a mixture of sulphurous acid and bisulphate iron, typically from sodium sulphite [3].

The most common mechanical pulping technique involves separating the cellulose fibres by pressing logs against wet grindstones or by passing wood chips between counter revolving grooved metal disks (refiners). Lignin and other residues are not removed. This results in a higher yield, but there is more damage to the fibres. In addition, lignin will degrade in time. The lower quality fibre limits the use of this process to less expensive grades of paper, such as newsprint [3]. Still, its use correspond to 22% of pulp production [5].

Combined chemical and mechanical pulping can produce varying grades of paper depending on the particular process used. These processes include thermo-mechanical, chemical thermo-mechanical, and semi-chemical [3].

Due to the complexities of the chemical reactions inside the digester (in chemical and chemical-mechanical pulping), the variability of the type of wood, and changes in the wood feed or rotation (speed) of the digester, regulating the pulping quality is a non-trivial task [1]. Therefore, the digester is considered one of the most critical pieces of equipment in a P&P mill [6].

In the pulping process, another by-product may be produced: the weak black liquor (which contains wood lignin, organic materials and inorganic compounds). The production of this liquor is proportional to the amount of pulp produced. From this point onwards, both products go through various transformation processes separately from each other. The weak black liquor goes to an intermediate buffer, ending in a chemical recovery process. The (virgin) pulp is stocked in tank volumes, waiting for being pulled by the paper machine. The intermediate tanks, with certain capacities, act as buffers to smooth downstream variations, both in chemical recovery and paper-making stages [1].

In the manufacture of writing, printing, and decorative papers a bleaching phase is required to whiten the pulp. The process alters or removes the lignin attached to the wood fibre before pulp is sent to the volume tanks. Chemical pulps are bleached through the use of alternating treatments of oxidizing agents and alkali solutions. The Kraft process produces a darker pulp which requires more bleaching. Mechanical pulps are treated with hydrogen peroxide or sodium hydrosulphite to reduce the light absorption of the lignin rather than remove it [3].

From the tanks, the virgin pulp is diluted and sent to the paper machine where it is joined with recycled pulp to produce paper. The recycled pulp is produced in the recycled pulp mill that refines recovered newspaper, magazines and paper lost during paper machine setups. The recycled pulp does not require a preparation process as complicated as pulping because the cellulose has already been removed from the raw material. Similarly to the virgin pulp, the recycled pulp is stocked in tanks and later diluted and sent to the paper machine [1]. It may also go through a bleaching process.

The virgin and recycled pulp (diluted in water) are fed together into a single or a set of parallel paper machine where the paper is prepared and formed and the water is removed from it by pressing and drying the sheet using air or heat. Preparation and drying are the most energy intensive processes. During preparation, the pulp is made more flexible through beating (mechanical pounding and squeezing process). Pigments, dyes, filler materials, and sizing materials are added at this stage. Forming involves spreading the pulp on a screen. The water is largely removed by pressing and the paper is left to dry [3]. Finally, the drying paper of a fixed grade-dependent width is wound into a master reel (called jumbo) with a known diameter. Each type of paper is characterized by the grammage expressed in grams per square meter (g/m^2), the incorporation rate of virgin and recycled pulp (in percentage) and the type of virgin pulp. The higher the level the virgin pulp is incorporated, the better the quality of the paper is. The speed of the machine (meters of paper per minute) is dependent on the type of paper being produced, but can be adjusted if necessary [1]. However, as this stage is usually the bottleneck, the paper machines are wanted to be at full speed.

The configuration of a machine to produce a new type of paper is sequence-dependent. Each setup leads to a loss in the production process in terms of time and quantity of a lower quality paper (as the machine is never idle, even during the switchover). The wasted paper is fed again into the recycled pulp mill [1].

Finally, the jumbo follows to winder where it is cut into smaller reels. Customers place orders for reels of different widths, grades and given percentages of virgin fibres. Converted paper products, such as paperboard boxes, may also be produced.

The chemical recovery process regenerates the spent chemicals applied in Kraft pulping, using the waste stream of inorganic chemicals and wood residues, known as black liquor. This stage can also provide steam, which may be converted into energy, needed by the paper production process, that is, the chemical recovery acts as a co-generation system that uses the biomass (concentrated liquors) as bio-fuel. Indeed, parallel to the paper forming, the weak black liquor coming from digesters goes to an intermediate tank before being concentrated in evaporators of given capacity. After this stage, the concentrated black liquor goes through another intermediate buffer and then to capacitated recovery boilers, where it is burnt, providing high-pressure steam. This steam can either be used for the drying process of the paper or be led to counter-pressure turbines which produce electrical energy, sold afterwards [1]. The wood residues provide the fuel and the chemicals are separated as smelt, which is then treated to produce sodium hydroxide. Sodium sulphide is also recovered [3]. Figure 2.1 shows a simplified scheme of the manufacturing process in a P&P mill. It should be noted that at each stage there can be several resources in parallel.

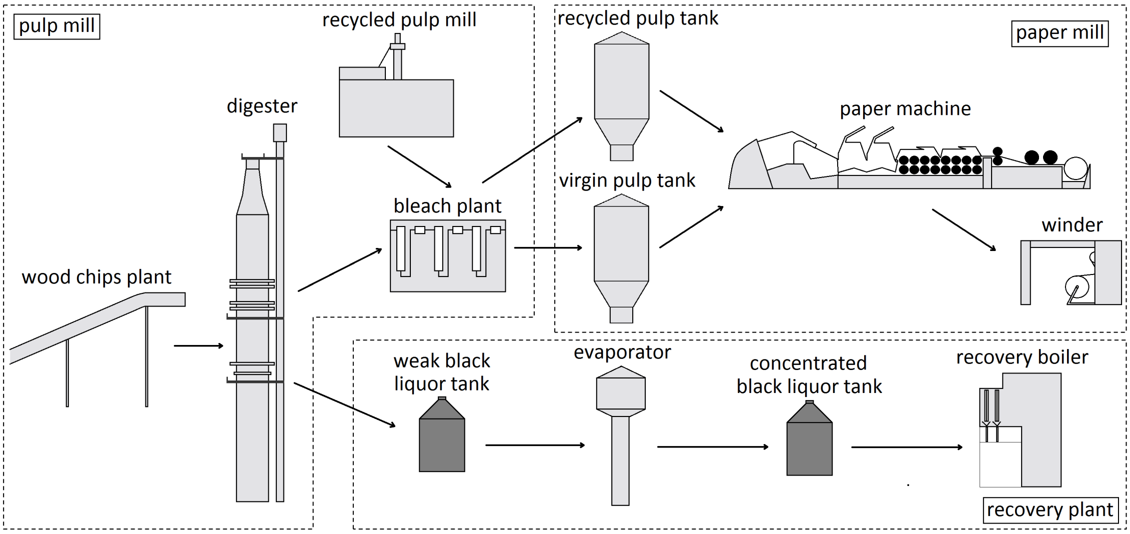


Figure 2.1: The integrated pulp, paper and recovery plant.

2.3 The Pulp and Paper Production Planning

2.3.1 The Planning Process and New Challenges

Process industries can be divided into two classes: continuous process industries and batch process industries. The second one can be viewed as positioned halfway between continuous and manufacturing industries. Batch processing is defined as a manufacturing technique in which parts are accumulated and processed together in a lot, while continuous flow production is lotless production in which products flow continuously rather than being divided. Examples of process industries include oil, steel, beverages, paper, major chemicals, rubber and specialty chemicals, among others. While the first industries of this set are very close to pure continuous flow, the last ones are more close to batch flow [7]. The paper industry is therefore in the middle of continuous and batch processes, being characterized by high capital intensity, low flexibility and make-to-stock policy. Its products have high demand volumes and the order winners are price and delivery guarantee. On the other hand, when compared to pure continuous process industries, the introduction of new products is higher and scheduling is more complicated. All these features make the P&P industry one of the most challenging from the production planning perspective.

The production planning process plays then a central role in the competitiveness strategy of P&P companies. Indeed, in this type of companies the production resources have an important share in the final products cost. On the other hand, the profit margins for this type of products are low, making costs reduction, by maximizing the production resources utilization, the primary way to increase competitiveness.

Production planning is thereby the core of competitiveness in P&P companies. The market strategy success strongly depends on the production efficiency, since supplying a wide range of products that meet specific customers demands, with competitive prices and strict due dates can only be achieved with superior planning. Moreover, to really have the market strategy guiding production decisions, an effective planning system that considers the set of defined KPIs (Key Performance Indicators) is required. These KPIs must be defined taking into account both market and production conflicting objectives, weighting them in a proper way.

However, this procedure is not always followed. Indeed, in most of the cases the planning process is completely manual and planners, having several constraints to satisfy, are led to place feasibility over optimality. Even when there is significant slack in constraints, it is difficult for planners to properly weight the different KPIs (in case they have been defined) in every decision along the manual planning process. This manual process usually follows a hierarchical scheme, where the top-level focus on the paper machine, dealing with the lotsizing and scheduling of paper campaigns, and the base-level tries to schedule and control all the other production resources, subject to the input from the top-level. To find a feasible solution, various iterations between the two levels may be needed [1], since for example, the sequence in the paper machine may force the digester to considerably change its speed and thus deteriorate paper quality. The resulting suboptimal plans may not only compromise production costs and company's efficiency, but also the service level it provides to its clients.

Moreover, new challenges are arising in what concerns production planning. The quest for improvements in process efficiency, resource consumption, wastes production and energy balance lead companies to an integration of their processes and introduction of closed loops, such as chemical recovery, paper recycling and substances re-use. All this integration makes the production planning much more complex, with more conflicting objectives and constraints and eventually multiple bottlenecks.

Additionally, the current pressures to reduce both supply costs and environmental impact make customers demand lighter products (lower grades), with higher incorporation of recycled pulp. Therefore, the way the different resources are balanced, as well as the standard sequence of campaigns used in the paper machine may no longer produce satisfactory (if feasible) results. New bottleneck resources may arise and capacity investments in some of them may become mandatory [1].

The optimization of the planning process is thereby extremely important in the current context of the P&P industry. However, it has not yet received the proper attention by companies. In most of the cases the planning process until the cutting stage is completely manual and the few cases where optimization is applied consist of local specific problems, such as the pulping stage. There are decision support systems both implemented and in development, some of them with a holistic approach. Still, although they have a strong information systems module, the optimization part is poorly defined or non-existent.

2.3.2 Literature Review

One of the main stages in this industry is the pulp production in digesters. Because of its capital intensive feature, producing an even quality pulp at consistently high production rate facing a given demand is the goal of this stage. The production rate shifts of the digester have to be managed efficiently to guarantee pulping process stability (better papermaking properties). Many papers have tackled the optimization of digester's cooking parameters. They are based on mathematical models to simulate the transient behaviour of the digester in terms of temperature and compound concentrations [6]. Peuhkuri [8] takes into account changes in production rates and wood species and even disturbance detection and management. Other papers have solved supply chain problems for pulp companies (e.g. [9]). However, all these works have their focus on the digester, not considering its great dependency on other resources, such as pulp silos or paper machines, which do affect pulp production in P&P plants. In fact, different paper types produced in paper machines pull more or less pulp from digester (forcing the change of its speed), according to their respective grades.

On the other hand, there are works in the literature that aim to optimize the papermaking step, where the definition of quantity of paper to produce in order to satisfy a set of ordered reels must be done. The works [10] and [11] have tackled this problem, assuming a fixed production sequence of paper grades. In [11] the authors determine first the cycle length, assuming constant demand and production rate, and then solve a lotsizing problem with the computed cycle being the elementary period. Finally, they solve the same lotsizing problem, but being constrained neither by a cycle nor by a fixed production sequence. It is concluded in this paper that optimizing the sequence would allow managers to save only 2% of setup and inventory costs. However, this must be seen in the context of a long-term planning problem, where the bucket, of one or two weeks, is too large to consider the exact inventory and backlogging costs. Therefore, the optimization of the sequence may actually have a much greater impact in the company's efficiency.

Again, these papers focus just on a part of the problem, in this case the paper machine, completely disregarding the pulp production. This approach can be entirely suitable, if the paper machine is actually the absolute bottleneck. However, the importance of digester's stability may constrain the papermaking stage, if the intermediate buffers have not sufficient capacity to effectively smooth variations. Moreover, the recovery stage may also limit the papermaking, since it may force the digester to reduce its speed, due to an eventual lack of capacity of the recovery boilers or evaporators. In fact, according to [12], the recovery boiler is often a bottleneck. The authors have then investigated two different approaches to debottleneck the recovery boiler: upgrade it or extract lignin from the black liquor in order to keep the boiler load constant. In [13], the authors focus on the heat cooperation between P&P industries and heating companies and analyse the effects of this cooperation in economical terms. The work [14] investigates the tradeoff between internal and external use of excess heat from a Kraft pulp mill for four different future energy market scenarios.

Capacity utilization is thus a major aspect for P&P companies, not only in digesters and paper machines, but also in the recovery stage resources, particularly the recovery boiler. Indeed, this stage is of maximum importance for the company's energy efficiency, so it should be managed simultaneously with the other major stages.

The last stage, where the cutting patterns used to produce smaller rolls have to be determined, has also been a focus of attention in the literature. The cutting stock problem is largely studied in and outside the P&P industry. Most of the authors approach this single problem without any integration with the upstream stages, i.e. the cutting stock problem is solved after determining the lots of products and their sequence in the paper machine [15]. The integration between papermaking optimization and cutting stock is particularly important when there are different parallel paper machines. Indeed, the maximum width and weight of each jumbo may depend on the machine in which it is produced [16]. Therefore, the allocation of jumbos to paper machines will determine their maximum widths and weights and then it will affect the cutting stock optimization. Despite its applicability in several industries, the combined cutting stock and lot-sizing problem has not been sufficiently studied because of its complexity. The work [16] is one of the few that has tried to couple lotsizing decisions with cutting stock problem. Still, the authors have ignored pulp production and finite capacities of several resources [1].

2.3.3 The Integrated Production Planning Approach

Research community has neglected the high degree of interrelation between the digester, the recovery boiler, the paper machine and the various silos that warehouse virgin and recycled pulp and other by-products, such as the weak black liquor. As the throughput of the paper mill (tonnes of related paper) may be constrained by different production resources in each instant, integrated production planning of the pulp and paper mill, as well as the chemical recovery plant is imperative. In fact, non-integrated approaches are not only suboptimal but also unrealistic, obliging managers to modify manually plans suggested by planning systems to obtain feasible ones. It is then mandatory that standard models are extended to effectively reflect reality and planners perform an integrated planning of the resources to identify, for instance, the need to debottleneck the recovery boiler or to increase silos capacity [1].

This integrated production planning approach aims the synchronization of material flow as it moves through pulp, liquors and paper, optimizing the company's efficiency and maximizing its KPIs. It is then a global approach, which can comprise or cooperate with other modules that solve specific problems, such as the optimization of digester's cooking parameters and the cutting stock problem. In fact, the optimization of the cooking parameters at the pulping stage, for instance, can be much more effective with a planning system that coordinates the different resources in order to avoid large variations in the digester's production speed.

Chapter 3

Optimization Approaches For Production Planning Problems

3.1 Models for Lot Sizing and Scheduling Problems

3.1.1 Introduction

Production planning addresses the acquisition, utilization and allocation of production resources required to transform raw materials into finished products in order to satisfy customer requirements in the most efficient and economical way [17].

Among the most important production planning problems are the sizing and scheduling of production lots. Researchers tended to be divided between those investigating lotsizing and those interested in scheduling problems. These two problems are usually solved by different decision makers in different planning horizons. However, in many industrial applications the close relationship between lotsizing and scheduling (LSS) forces both decisions to be made simultaneously in order to efficiently use capacity [17]. This is the case of the pulp and paper (P&P) industry, due to significant sequence dependent setups in paper type changeovers.

Therefore, while in long-term planning, models may include only lotsizing decisions, in short-to-medium term the lotsizing and scheduling integration should be considered. For instance, [18] overviews recent developments in modelling lotsizing and points out a further integration of LSS as an interesting area for future research, as well as an increasing attention to model specificities of the production process. On the other hand, [19] highlights the scarcity of literature devoted to LSS with sequence-dependent setups despite its relevance, and underlines the need for faster and more efficient heuristics [1].

Eppen [20] classifies production planning models into either big-bucket or small-bucket. Big-bucket models allow for the production of many items in a single period (suitable for lotsizing problems), whereas in small-bucket models only one setup can be performed on a machine during a given period and, therefore, as referred before, depending on the model at most one or two items may be produced per period [17]. Thus, the production sequence is visible in the solution.

Naturally, it is by far more straightforward to include additional requirements and detail into small-bucket formulations than in big-bucket ones.

In this chapter some basic single-machine lotsizing and scheduling models (small-bucket) are presented and discussed. A more general model is also explored. These models contain basic modelling ideas and can be extended to incorporate more detail, such as sequence-dependent setups, setup times, multiple machines (in parallel or in sequence) and other more specific features of the production process, in order to better represent complex real problems. Therefore, the section ends with some possible extensions to the basic models.

3.1.2 Small-bucket Models

In small-bucket models the planning horizon is divided into many short periods (such as days, shifts or hours), in which at most one setup may be performed. Therefore, depending on the models, we are limited to producing at most one or two items per period. Such models are useful for developing short-term production schedules. Lotsizing and scheduling decisions are taken simultaneously, as here a lot consists in the production of the same product over one or more consecutive periods [17]. This is the case of discrete lotsizing and scheduling problem (DLSP), continuous setup lotsizing problem (CSLP) and proportional lotsizing and scheduling problem (PLSP), to be presented below.

Before presenting the models, it is introduced first some notation, which is going to be used in some or all the models.

Indices

i, j Indices for item ($i \in 1...N$)

t Index for time period ($t \in 1...T$)

Parameters

sc_i Cost incurred to set up the machine for product i

h_i Cost of holding one unit of product i during one period

d_{it} Demand for product i in period t

p_i Processing time of one unit of product i

C_t Capacity of the machine in period t (measured in time units)

Y_{i0} $\begin{cases} 1 & \text{if the machine is configured for product } i \text{ at the beginning of the planning horizon,} \\ 0 & \text{otherwise.} \end{cases}$

Decision variables

X_{it} Quantity of product i produced in period t

I_{it} Inventory of product i at the end of period t

Y_{it} $\begin{cases} 1 & \text{if a setup occurs on the machine configuration state for product } i \text{ in period } t, \\ 0 & \text{otherwise.} \end{cases}$

Z_{it} $\begin{cases} 1 & \text{if the machine is set up for product } i \text{ in period } t \text{ and it was not in } t-1, \\ 0 & \text{otherwise.} \end{cases}$

In DLSP, demand for each item is dynamic and backlogging is not allowed. Thus, production serves to meet only present or future demand. In each period, the machine either produces at full capacity or is idle ("all-or-nothing") and at most one item can be produced per period [17]. DLSP (with time varying capacity) can be formulated as follows:

$$min \sum_i \sum_t sc_i \cdot Z_{it} + \sum_i \sum_t h_i \cdot I_{it} \tag{3.1}$$

$$I_{it} = I_{i,t-1} + X_{it} - d_{it} \qquad i = 1 \ldots N, \qquad t = 1 \ldots T. \tag{3.2}$$

$$\sum_i Y_{it} \leq 1 \qquad t = 1 \ldots T. \tag{3.3}$$

$$p_i \cdot X_{it} = C_t \cdot Y_{it} \qquad i = 1 \ldots N, \qquad t = 1 \ldots T. \tag{3.4}$$

$$Z_{it} \geq Y_{it} - Y_{i,t-1} \qquad i = 1 \ldots N, \qquad t = 1 \ldots T. \tag{3.5}$$

$$(X_{it}, I_{it}, Z_{it}) \geq 0, Y_{it} \in \{0,1\} \qquad i = 1 \ldots N, \qquad t = 1 \ldots T. \tag{3.6}$$

The objective function is expressed by (3.1) and it consists of a minimization, which does the tradeoff between setup and holding costs. (3.2) ensure the inventory balance and (3.3) that at most one item is produced in each period. In (3.4) the production of each period is equalled to capacity and (3.5) determine changeovers, based on whether the setup variable changes its value or not between two consecutive periods.

In spite of the "all-or-nothing" assumption may be reasonable in many practical situations (e.g. when production quantities are integer multiples of a minimum batch size and the period duration is sufficiently short), in many others it will lead to unrealistic plans. Moreover, constraints (3.4) impose Y_{it} to be zero in case the machine is idle (thus, $X_{it} = 0$). Consequently, the setup state is not kept up during idle periods and the first production after an idle period incurs always a setup cost, even if the product is the same as the last one produced [17]. Furthermore, when there is overcapacity the planned production is going to be greater than demand (since it always produces at full capacity), leading to unnecessary additional production and inventory costs.

To overcome these issues, constraints (3.4) may be relaxed, as in (3.7), in order to allow lotsizes being continuous and holding quantities below period capacity. The resulted model is the already mentioned continuous setup lotsizing problem (CSLP).

$$p_i \cdot X_{it} \leq C_t \cdot Y_{it} \qquad i = 1 \ldots N, \qquad t = 1 \ldots T. \tag{3.7}$$

On the other hand, proportional lotsizing and scheduling problem (PLSP) introduces a further improvement: being possible to schedule a second item in a period where capacity is not fully used. This can be achieved replacing constraints (3.7) by (3.8) and (3.9) in the CSLP formulation.

$$p_i \cdot X_{it} \leq C_t \cdot (Y_{i,t-1} + Y_{it}) \qquad i = 1 \ldots N, \qquad t = 1 \ldots T. \tag{3.8}$$

$$\sum_i p_i \cdot X_{it} \leq C_t \qquad t = 1 \ldots T. \tag{3.9}$$

(3.8) allow item i to be produced in period t, if the machine is set up for item i at the end of period $t-1$. In (3.9) it is assured that capacity is not exceeded by all the production of the scheduled items in each period.

Since PLSP is a generalization of CSLP (and the same between CSLP and DLSP), the former can have a better solution quality but requires more computational time. Nevertheless, PLSP allows for longer periods, since two items can be produced in the same period, and hence a model with less periods may be used [17]. The choice between CSLP and DLSP depends on whether the already mentioned issues that result in DLSP are relevant to the specific problem or instance to be solved.

3.1.3 General Lotsizing and Scheduling Problem

A criticism to small-bucket models is that for real-world instances they require a prohibitive number of periods, especially if mathematical programming approaches are to be implemented. This fact motivated the development of a more flexible model: the general lolsizing and scheduling problem (GLSP), which makes use of a two-level time structure. In this model the planning horizon is divided into macro-periods t, with a given length. Each macro-period is divided into a fixed number of non-overlapping micro-periods with variable length, where S_t denotes the set of micro-periods s belonging to the macro-period t. All micro-periods are ordered in the sequence $s = 1...S$, where $S = \sum_t |S_t|$ [21].

Since micro-periods are of variable length, they can be adjusted either to entire lots or just to a part of them. In both cases, the micro-periods do not have to represent a fixed time unit. Therefore, the number of micro-periods may be drastically decreased, making the model size much more small. On the other hand, the complexity of the model increases.

The GLSP model (with conservation of setup state) is presented below:

$$min \sum_i \sum_s sc_i \cdot Z_{is} + \sum_i \sum_t h_i \cdot I_{it} \tag{3.10}$$

$$I_{it} = I_{i,t-1} + \sum_{s \in S_t} X_{is} - d_{it} \qquad i = 1 \dots N, \qquad t = 1 \dots T. \tag{3.11}$$

$$\sum_i Y_{is} \leq 1 \qquad s = 1 \dots S. \tag{3.12}$$

$$p_i \cdot X_{is} \leq C_t \cdot Y_{is} \qquad i = 1 \dots N, \qquad t = 1 \dots T, \qquad s \in S_t. \tag{3.13}$$

$$\sum_i \sum_{s \in S_t} p_i \cdot X_{is} \leq C_t \qquad t = 1 \dots T. \tag{3.14}$$

$$Z_{is} \geq Y_{is} - Y_{i,s-1} \qquad i = 1 \dots N, \qquad s = 1 \dots S. \tag{3.15}$$

$$(X_{is}, I_{it}, Z_{is}) \geq 0, Y_{is} \in \{0,1\} \qquad i = 1 \dots N, \qquad t = 1 \dots T, \qquad s = 1 \dots S. \tag{3.16}$$

This model is very similar to the previous ones, except that here the decision variables are indexed to micro-periods. However, parameters are kept related to macro-periods. The length of a micro-period is now a decision variable, implicitly defined by constraints (3.13) and (3.14) and being proportional to the quantity produced in it.

The fact that other models (such as the previously presented small-bucket problems) can be obtained from GLSP by adding additional constraints motivates the name of this model. Therefore, the set of solutions of each of these models is a subset of the solutions of GLSP [17].

3.1.4 Model Extensions

The previously presented models are very simple and do not represent most of real problems. Therefore, extensions may be applied to these models in order to include specific features and hence result in more realistic plans. Some of these extensions are explored below.

In many industrial processes, setup times (*st*) consume a significant part of the available capacity. Thus, it is imperative to include them in the capacity constraints, as in (3.17).

$$p_i \cdot X_{it} + st_i \cdot Z_{it} \leq C_t \cdot Y_{it} \qquad i = 1 \ldots N, \qquad t = 1 \ldots T. \tag{3.17}$$

In order to consider sequence-dependent setups, the Z variable must have an additional index (for the previous item) and must be defined by (3.18), where Z is forced to be one if the machine is set up for product i in the previous subperiod and for j in the current one (i.e. in case there is a changeover from i to j).

$$Z_{ijt} \geq Y_{i,t-1} + Y_{jt} - 1 \qquad i = 1 \ldots N, \qquad j = 1 \ldots N, \qquad t = 1 \ldots T. \tag{3.18}$$

When triangle inequalities do not hold, as in some chemical industries, minimum lotsizes (m) must be defined to avoid set-up changes without product changes [21], as follows:

$$X_{it} \geq m_i \cdot (Y_{it} - Y_{i,t-1}) \qquad i = 1 \ldots N, \qquad t = 1 \ldots T. \tag{3.19}$$

In some cases, backlogging (taken into account by variable I^-) may be necessary, so it has to be included in both the inventory balance (constraints (3.20)) and objective function.

$$I_{it} = I_{i,t-1} + X_{it} - d_{it} - I^-_{t-1} + I^-_t \qquad i = 1 \ldots N, \qquad t = 1 \ldots T. \tag{3.20}$$

Finally, he production process may either be multi-stage (with machines in sequence) or occur in a single stage but with parallel machines (which may be identical, uniform or unrelated) or both. The second case implies the capacity constraints to be as in (3.21), as well as the inventory balance to include the production of all the machines (indexed by $m = 1 \ldots M$). Multi-stage scenario imposes the definition of additional inventory balance constraints, for the intermediate products, which have their demand dependent on the lotsizes in the downstream levels [21].

$$\sum_i p_{im} \cdot X_{imt} \leq C_{mt} \qquad m = 1 \ldots M, \qquad t = 1 \ldots T. \tag{3.21}$$

3.2 Solution Methods

Once the problem is modelled, a solving procedure is required to generate a "good" solution for the problem. Depending on its complexity, it may be solved by an exact method or an approximate method. Exact methods obtain optimal solutions and guarantee optimality. For NP-complete problems, exact algorithms are non-polynomial time. Approximate methods can generate high-quality solutions in a reasonable time, but there is no guarantee of finding a global optimal solution [22]. These methods include problem-specific heuristics, metaheuristics and mathematical programming-based heuristics. Specific heuristics are tailored to solve specific problems or instances, while the other are general-purpose, being able to solve almost any optimization problem. The following sections present and explore exact methods, mathematical programming-based heuristics and metaheuristics, as well as hybrid methods.

3.2.1 Exact Methods

There is a good variety of exact methods, such as dynamic programming, branch and bound, branch and cut, branch and price and constraint programming.

One of the most popular is branch and bound, which is based on an implicit enumeration of all the solutions. It involves solving several linear relaxations of the problem and branching integer variables. Different strategies can be applied in its two main operators (branching and pruning). These operators intervene in each iteration of the algorithm. The branching strategy determines the order in which the branches are explored. Different strategies comprise depth-first, breadth-first and best-first. The pruning strategy eliminates the partial solutions that do not lead to optimal solutions, computing their lower bound [22].

In spite of intelligent strategies can be applied in branch and bound, the computational time in real-world large instances of NP-complete problems is prohibitive. Indeed, in many real-world problems exact methods spend too much time just to find a feasible solution. In these cases, more efficient algorithms are necessary.

Moreover, an interesting question may occur: why solve exactly real-life optimization problems that are fuzzy by nature? In fact, as seen in Section 3.1, the optimization models are simplifications of reality. They involve approximations and sometimes they skip processes that are complex to represent [22].

3.2.2 Mathematical Programming-based Heuristics

Two major mathematical programming-based heuristics are the linear programming (LP) based and the mixed integer programming (MIP) based. The former are based on LP relaxation of the MIP model. The solution is then rounded up according to different strategies. This simple method may work well in some problems, but it tends to fail in more complicated problems. The MIP-based heuristics do not solve LP relaxations, but MIP sub-problems that may be partially relaxed from the original MIP formulation.

The computational time of the exact methods increases in a non-linear way with the increase of the amount of discrete variables. Thus, decreasing the number of these variables results in great savings of computational effort. MIP-based heuristics lie on this idea. They subdivide or partially relax and/or fix the overall mixed integer programming (MIP) problem, which results in easier MIPs that are solved much more quickly.

The so-called progressive interval heuristics (such as the relax-and-fix heuristic and internally rolling schedule heuristics) are MIP-based heuristics that solve the overall MIP problem as a sequence of partially relaxed MIPs, always starting from the first period on a forward shifting move. The setup variables are progressively fixed at their optimal values obtained in earlier iterations. The algorithm has two main parameters: the number of periods to optimize in each iteration (which are neither relaxed nor fixed) and the number of overlapping periods (which are the common periods to optimize between two consecutive iterations). The tuning of these parameters allow a trade-off between optimality and run-time [23].

The authors in [24] describe an improvement heuristic similar to the relax-and-fix heuristic, which they called "exchange". The method differs in that it starts with a feasible solution and does not relax the binary setup variables in the sub-problems. At each step the integer variables are fixed at their best value found in the previous iterations, except for a limited set of integer variables that are related to the sub-problem at hand [23].

3.2.3 Metaheuristics

Metaheuristics are high level frameworks that combine basic heuristics, in order to efficiently and effectively exploring a search space [25]. One of the most important aspects in a metaheuristic is the trade-off between two major strategies: intensification and diversification. Intensification consists of exploring specific regions of the search space, identified as promising by the solution procedure. In diversification there is an effort to evenly explore all the regions of the search space. This strategy is of particular importance when the best solutions are scattered throughout the search space.

Metaheuristics can be divided into two main categories: single-solution based and population-based. Single-solution based metaheuristics may be viewed as trajectories performed by a single solution that moves through its consecutive neighbourhoods in the search space. Population-based metaheuristics use a population of solutions and are not based on neighbourhood structures. Instead, they generate new populations, which are integrated into the current one using some selection procedures [22]. In spite of both categories being flexible enough to be adapted to a very large set of problems, population-based metaheuristics seem to be more suitable for cases where diversification is a major issue (when compared to intensification) or when it is required to generate a set of different optimal solutions, as in multi-objective optimization.

Taking into account the type of the problem approached in this dissertation (single-objective production planning and scheduling), it focuses just on single-solution based procedures and thus, they are going to be explored in the current section. Besides the number of solutions carried out simultaneously by the algorithm, other classification criteria may be used, such as:

- memory usage (extracted during the search) or memoryless;
- stochastic (random or probabilistic decisions applied during the search) or deterministic;
- and iterative, starting with a complete solution and transforming it in each iteration, or greedy, starting from an empty solution and assigning decision variables until a complete solution is obtained [22].

For a solution to describe a trajectory in the search space, two important components are required: a constructive heuristic to generate an initial solution and a neighbourhood structure that defines the possible moves a solution can perform in each iteration. The constructive heuristics can be random or greedy. The former are easier to implement and do not converge so quickly to a local optimum. The greedy heuristics speed up the subsequent search and are more likely to generate initial feasible solutions.

According to [22], local search (LS) can be seen as the simplest metaheuristic method. It starts with a given initial solution and, in each iteration, it replaces the current solution by a neighbour that improves the objective function. The search stops when all candidate neighbours are worse than the current solution, meaning a local optimum is reached. For large neighbourhoods, the candidate solutions may be a subset of the neighbourhood, in order to speed up the search. Variants of LS can be distinguished according to the order in which neighbours are generated (deterministic or stochastic) and the selection strategy (best improvement, first improvement or random selection). In practice, on many applications, it has been observed that the first improvement strategy leads to the same solution quality as the best improvement, while using less computational time and a smaller probability of premature convergence to a local optimum [22].

In order to escape from local optima, different strategies were developed into more sophisticated metaheuristics, such as simulated annealing (SA), tabu search (TS), variable neighbourhood search (VNS) and greedy randomized adaptive search procedure (GRASP). In SA, non-improving neighbours are accepted with a given probability. TS chooses in each iteration the best non-tabu neighbour (even if it does not improve). The tabu list, which determines the tabu neighbours, is intended to avoid visiting the same solutions repeatedly. VNS uses a set of neighbourhoods to shake the solution and take it out of local optima. In its general version, it uses also a set of neighbourhoods to do the local search and find different local optima. In GRASP, a multi-start is performed with a greedy randomized heuristic, followed by a local search.

3.2.4 Hybrid Methods

The various optimization approaches can be combined into hybrid methods. The idea is to enhance algorithms using the core advantages of different methods and benefit from eventual synergies among them. In this section, hybrid methods involving metaheuristics are explored. Indeed, hybridization of metaheuristics with other techniques has recently become an important focus of research, since it consists of a fruitful cross-fertilization of approaches that leads to superior results [22, 26]. However, according to [27], hybrid metaheuristics are still in their infancy and need substantial further research to deliver superior solutions in reasonable computational times.

The work [26] provides a survey of some of the most important lines of metaheuristics hybridization. It divides the hybrid methods into five major classes: metaheuristics with (meta-)heuristics, metaheuristics with constraint programming, metaheuristics with tree search techniques, metaheuristics with problem relaxation and metaheuristics with dynamic programming. The combined metaheuristics and the exact tree search techniques are explored in the following subsections.

3.2.4.1 Hybridization of metaheuristics with (meta-)heuristics

According to [26], hybridization of metaheuristics with heuristics or other metaheuristics was the most popular hybrid approach when they started rising. Nowadays the hybridization of different metaheuristics is very widespread, especially for what concerns the use of local search methods within population-based methods. In fact, population-based methods have a strong diversification strategy, but may lack in intensification. The local search procedure addresses this issue, refining the generated solutions and converging quickly to local optima. This is an example of two completely different procedures that are complementary and thus create a positive synergy.

Nevertheless, similar metaheuristics may create positive synergies too. Talbi [22] refers to homogeneous hybrids as combined algorithms that use the same metaheuristic. These metaheuristics may differ in the initialization of their parameters (e.g. tabu list size in TS hybridization) and search components, such as the objective function or the search operators (neighbourhood, mutation, crossover, etc.). In fact, there is a wide range of possibilities to create different initializations that provide good synergies.

Another popular hybridization is the use of greedy heuristics to create initial solutions both for single solution-based and population-based metaheuristics. In the last one, the greedy heuristic has to be stochastic in order to generate a population of different solutions. Indeed, as greedy heuristics are in general of less computing complexity than iterative heuristics, they are suitable for finding quickly feasible solutions of fair quality.

3.2.4.2 Hybridization of metaheuristics with exact tree search techniques

The hybridization of metaheuristics with tree search techniques is probably one of the most popular lines of combining different algorithms for optimization [26]. A common approach is to search neighbourhoods in local search based metaheuristics by means of exact algorithms. If the neighbourhoods are chosen appropriately, they can be relatively large and nevertheless an efficient search for the best neighbour is still reasonable. In fact, exact methods may exhaustively search relatively large promising regions of the solution space within a reasonable amount of time. Such techniques are known as Very Large-Scale Neighbourhood (VLSN) search [27].

Another approach is to use exact algorithms as decoders. As seen in Subsection 3.2.2, the main issue of exact methods is the discrete variables. On the other hand, metaheuristics do not perform well if continuous variables have their own degrees of freedom, i.e., if they can change their value without changing the value of any discrete variable (as it occurs in GLSP formulations). In fact neighbourhood operators are suitable for discrete variables, not to continuous ones. Moreover, exact methods are more flexible to cope with model extensions, as they do not incorporate problem-specific heuristics. Therefore, an hybridization of these two optimization methods has a lot of potential. The authors in [28], for example, present an effective approach for solving a lot-sizing and scheduling problem, where the solution is represented in a metaheuristic by the setup pattern and the decoding is optimally performed by an exact algorithm, which has just to solve a linear programming problem.

In the previous two examples the exact method was incorporated into the metaheuristic, i.e., the exact method was a subordinate embedded component of the metaheuristic [27]. The opposite also exists: metaheuristics incorporated into exact methods. In fact, heuristics and metaheuristics are often used to determine bounds and incumbent solutions in branch and bound approaches. In some nodes of the tree, it may be beneficial to run a metaheuristic procedure to eventually find a better incumbent solution [27].

3.3 General Comments on Literature Review

GLSP proves to be a very flexible formulation in what concerns the sizing of productions lots with a given fixed number of planning (sub)periods. However, for the same number of periods, its computational complexity is obviously higher. The choice between the different basic models (analysed in previous sections) depends thus on the type of production system that is being studied. In the P&P industry, where the lotsizes may considerably vary, GLSP should be a wise choice, since the subperiods length can be adjusted to production lotsizes.

The flexibility of GLSP consists of an additional challenge for solution methods. Indeed, the length of the subperiods has its own degree of freedom and hence, it becomes too hard for a pure metaheuristic to obtain good quality solutions. Hybrid approaches combining metaheuristics and exact methods emerge as an effective response to that challenge. Moreover, they allow the solution method to easily cope with further model extensions, since they do not incorporate so many problem-specific heuristics, as they leave most of the optimization for the exact part. This is an important feature, since not only companies are constantly changing their planning requirements and constraints, but also different plants may have different specificities and thus, a more flexible algorithm could be easily adapted to solve different problems. In fact, one of the aims of this dissertation is to create a solution method as flexible as possible to be able to be adapted or extended to other companies of the industry.

Chapter 4

The Case Study

4.1 The Company and its Production Planning

EUROPA&C KRAFT VIANA is a Portuguese P&P mill and is the only kraftliner plant in Iberian Peninsula. It is the fourth biggest manufacturer in Europe and sells its products worldwide. The kraftliner is a type of paper for the corrugated cardboard boxes market, having high physical and moisture resilience.

The company currently produces two main products: "Portoliner" and "Vianaliner". The former is for the market of boxes for food and industrial products. In its composition it incorporates a great portion of virgin pulp, but also pulp made from recycled paper. "Vianaliner" does not belong to the traditional kraftliner type. It incorporates a greater quantity of recycled fibres and is more suitable for the industrial market. Still, it is close to the resilience properties of "Portoliner". Both products may be produced in a set of different grammages and variants. For the sake of simplicity, hereafter the term "grammage" is used for the combination of the thickness of the paper (the grammage itself or grade, measured in g/m^2) and the proportion of virgin and recycled fibres.

As an integrated P&P mill, EUROPA&C KRAFT VIANA does all the major production stages: pulping, papermaking, cutting and chemicals recovery. Wood preparation and paper recycling are also performed on site. Only bleaching is not made, since the final products do not have to be whitened.

There is only one digester and one single paper machine. These two resources are the focus of the planning process, which is currently done manually by managers. This manual process follows a hierarchical scheme, where the top-level focus on the paper machine, dealing with the lotsizing and scheduling of paper campaigns, and the base-level tries to schedule and control all the other production resources, particularly the digester, subject to the input from the top-level. To find a feasible solution, various iterations between the two levels may be needed [1], since for example, the sequence in the paper machine may force the digester to considerably change its speed and thus deteriorate paper quality. The other resources, such as the recovery boiler, the evaporator and the various tanks may also constraint the paper machine scheduling, since they affect the digester's production rate and they have upper and lower capacity bounds to be satisfied.

The sequence in the paper machine is programmed in cycles, i.e. during a certain period of time (two weeks in this case), all the grammages with demand or backlog to be fulfilled must be scheduled in the paper machine with a given sequence. This sequence is determined in order to minimize setups and hence it consists of an increasing (or decreasing) order of grades, followed by a decreasing (or increasing) order. This occurs within each type of paper ("Portoliner" and "Vianaliner") and the changeover between them is planned to be as smooth as possible (with similar grades). In this way, the time lost in setups is minimal and the backlog orders of each grammage should not increase, since all grammages are guaranteed to be produced every cycle.

However, this planning system does not assure that backlogging is minimized. Moreover, the fact that the planning is manual implies various iterations to find a feasible solution, which results in suboptimal plans that may not only compromise production costs and company's efficiency, but also the service level it provides to its clients. Santos and Almada-Lobo [1] developed a multi-stage lotsizing and scheduling formulation for the integrated production planning of pulp, paper and energy (obtained in chemicals recovery), based on the plant of EUROPA&C KRAFT VIANA. An updated version of this model is presented and analysed in the next section and its generality is inferred.

4.2 Exact Formulation for the Integrated Production Planning Problem

The mixed-integer programming (MIP) model is based on the general lot sizing problem (GLSP) with setup times and sequence-dependent setups and extended for the multi-stage scenario with additional constraints. As seen in Section 3.1.3, GLSP uses the concept of the combined small-bucket-big-bucket approach, where the planning horizon is divided into macro-periods t ($t = 1...T$), each, in its turn, divided into a fixed number of non-overlapping micro-periods with variable length. S_t denotes the set of micro-periods s belonging to the macro-period t. All micro-periods are ordered in the sequence $s = 1...S$. The lengths of the subperiods are decision variables 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 and fixed for all the resources and production stages (similar modelling techniques are used in [29] and [30]). This assumption makes the synchronization of the various production units easier at a cost of needing a higher number of micro-periods per period to gain enough flexibility to derive good-quality plans [1]. Nevertheless, the total number of micro-periods is much less than it would be in a small-bucket formulation. In this feature lies the advantage of GLSP, sharply decreasing the size of the model when lotsizes are continuous and too much variable. This is the case of the production planning problem in the P&P industry.

The presentation of the MIP model is divided into four subsections: 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.

4.2.1 Pulp Mill

As in this case there is always sufficient wood chips in house to process downstream activities, the wood preparation is not considered in the model. In fact, this assumption could be extended to most of the industry, since it is not economically convenient for this stage to be the bottleneck.

The quantity of virgin pulp to be produced by the digester in each subperiod s is proportional to the rotation speed of the digester, V_s^{dig} (expressed in rotations per minute - rpm). This speed is limited to minimum and maximum thresholds and is only allowed to vary each time a maximum number of rpm to ensure the stability and smoothness of the cooking process. Hence, $V_s^{dig} = V_{s-1}^{dig} \pm \Delta \cdot \Phi$, where Φ is the minimum possible speed variation and Δ the maximum step. Φ and Δ define the granularity of the digester's speed grid in each micro-period. For example, if $V_0^{dig} = 12.0$, $\Delta = 1$ and $\Phi = 0.5$, V_1^{dig} is limited to taking one of the following values: 11.5, 12.0 or 12.5. This constraint may be not so hard in other plants, where the equipment or the control system can accommodate large changes in the digester's speed. Nevertheless, if that is the case, the corresponding constraints just need to be removed. The part of the model related to the pulp production is following presented.

Parameters and indices	
s	Index for subperiod ($s \in 1...S$)
S_t	Set of subperiods s belonging to the period t
v	Index for digester's rotation speed ($v \in 1...V$)
Φ	Minimum possible speed variation
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)
sp_v	Value of the speed indexed by v (e.g. $sp_1 = v_{min}^{dig}$)
cap_t	Capacity of period t (given 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 tank (in tonnes)
$I_{max}^{recy}(I_{min}^{recy})$	Upper (lower) bounds on the recycled pulp stocked in the tank (in tonnes)
Decision variables	
Y_{vs}^{dig}	$\begin{cases} 1 & \text{if the digester runs at speed } v \text{ in subperiod } s, \\ 0 & \text{otherwise} \end{cases}$
Ns	Length of subperiod s
Nh_{vs}	Number of hours that the digester works at speed v in subperiod s
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 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 tank during the grammage changeover in s (in tonnes)

$$\sum_{v} Y_{vs}^{dig} = 1 \qquad s = 1 \ldots S. \tag{4.1}$$

$$Y_{vs}^{dig} \leq \sum_{k=(v-\Delta)k\geq 0}^{v+\Delta} Y_{k,s-1}^{dig} \qquad s = 1 \ldots S. \tag{4.2}$$

$$Nh_{vs} \leq cap_t \cdot Y_{vs}^{dig} \qquad v = 1 \ldots V, \quad s = 1 \ldots S. \tag{4.3}$$

$$N_s = \sum_{v} Nh_{vs} \qquad s = 1 \ldots S. \tag{4.4}$$

$$\sum_{s \in S_t} N_s = cap_t \qquad t = 1 \ldots T. \tag{4.5}$$

$$X_s^{dig} = \alpha \cdot \sum_{v} sp_v \cdot Nh_{vs} \qquad s = 1 \ldots S. \tag{4.6}$$

$$X_s^{recy} \geq xr_{min}^{recy} \cdot N_s \qquad s = 1 \ldots S. \tag{4.7}$$

$$X_s^{recy} \leq xr_{max}^{recy} \cdot N_s \qquad s = 1 \ldots S. \tag{4.8}$$

$$\sum_{s \in S_t} X_s^{recy} \leq K_{cap}^{recy} \cdot xr_{max}^{recy} \cdot cap_t \qquad t = 1 \ldots T. \tag{4.9}$$

$$X_s^{dig} + I_{s-1}^{virg} = O_s^{virg} + I_s^{virg} \qquad s = 1 \ldots S. \tag{4.10}$$

$$X_s^{recy} + I_{s-1}^{recy} + IS_s^{recy} = O_s^{recy} + I_s^{recy} \qquad s = 1 \ldots S. \tag{4.11}$$

$$I_{min}^{virg} \leq I_s^{virg} \leq I_{max}^{virg} \qquad s = 1 \ldots S. \tag{4.12}$$

$$I_{min}^{recy} \leq I_s^{recy} \leq I_{max}^{recy} \qquad s = 1 \ldots S. \tag{4.13}$$

The digester runs at the same speed throughout each subperiod s (constraints (4.1)). 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 (4.2) ensure a smooth shift of the speed of the digester between two consecutive subperiods. The number of hours that the digester works at speed v is computed by constraints (4.3) and (4.4). The latter also define the length of each subperiod s that is common to all resources. The sizes of the micro-periods of the same period altogether can not exceed the capacity of the respective period (constraints (4.5)). It should be mentioned that the digester is never idle (24-7 production system) and thus, the period capacity cannot be extended with overtime. As the pulp is continuously produced in the digester, the amount of virgin fibres in subperiod s is proportional to the speed of the digester and to the length of s, as in constraints (4.6).

The production of the recycled pulp plant (where the waste collected is refined) in each subperiod, X_s^{recy}, is limited to the lower (xr_{min}^{recy}) and upper bounds (xr_{max}^{recy}), expressed in tonnes per time unit. 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 (4.7) - (4.9).

The tanks are used as intermediate buffers to stock virgin and recycled pulp separately coming from the digester and refiner, respectively, to prevent bottlenecks. Both tanks have limited storage capacity and must be filled with stock over a minimum limit (constraints (4.12) and (4.13)). The inventory balancing equations of virgin and recycled pulp are given by constraints (4.10) and

(4.11). In the case of recycled pulp tanks, it is necessary to consider the recovery of the paper loss during the grammage changeover (see constraints (4.11)).

4.2.2 Paper Mill

The paper mill is responsible for converting the pulp (mixture of recycled and virgin pulp) into final paper that is wound into the jumbo and then cut into smaller reels. Only the forming of the kraft paper in the paper machine is considered in this model. The cutting stage is then disregarded and hence, demand for the final products (ordered reels) is aggregated by type of grammage. To anticipate the downstream average waste of the cutting process, an average trim loss of ι is considered. The non-integration of the cutting stage should not be important in this case, since there is a single paper machine. In fact, it is in a multiple-machine environment that the coupling of machine scheduling with cutting stock may be significant.

The paper grammage determines the maximum production rate of the paper machine, the necessary incorporation of water and the width of the jumbo (L_j). The speed of the machine can be adjusted if necessary. However, as this stage is usually the bottleneck, the paper machines are wanted to be at full speed. Thus, and for a matter of simplicity, the model considers the paper machine speed as a fixed parameter. The part of the model that represents the paper production is introduced below.

Parameters and indices

j,k	Indices for grammages ($j,k \in 1...K$)
t	Index for period ($t \in 1...T$)
sl_{kj}	Paper lost (setup cost) in a changeover from grammage k to j (in tonnes)
st_{kj}	Time lost (setup time) in a changeover from grammage k to j (in hours)
b_j^{virg} $(1-b_j^{virg})$	Percentage of virgin (recycled) pulp used in the production of grammage j
f_j	Percentage of water in grammage j
p_j	Processing time to produce one tonne of grammage j
M_j	Large number for each grammage j
m_j	Minimum lot size of grammage j
D_{jt}	Demand for grammage j in period t (in tonnes)
$K_{backlog}$	Percentage of the initial backlog to be fulfilled at the end of the planning horizon for each grammage
ι	Average trim loss (in percentage)
Y_{j0}	$\begin{cases} 1 & \text{if the machine is set up for grammage } j \text{ at the beginning of the planning horizon,} \\ 0 & \text{otherwise.} \end{cases}$

Decision variables

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

$$\sum_j b_j^{virg} \cdot (1-f_j) \cdot (X_{js} + \sum_k sl_{kj} \cdot Z_{kjs}) = O_s^{virg} \qquad s = 1 \ldots S. \quad (4.14)$$

$$\sum_j (1-b_j^{virg}) \cdot (1-f_j) \cdot (X_{js} + \sum_k sl_{kj} \cdot Z_{kjs}) = O_s^{recy} \qquad s = 1 \ldots S. \quad (4.15)$$

$$\sum_j \sum_k (1-f_j) \cdot sl_{kj} \cdot Z_{kjs} + \sum_j \iota \cdot X_{ks} = IS_s^{recy} \qquad s = 1 \ldots S. \quad (4.16)$$

$$\sum_j \left(p_j \cdot X_{js} + \sum_k st_{kj} \cdot Z_{kjs} \right) = N_s \qquad s = 1 \ldots S. \quad (4.17)$$

$$(1-\iota) \cdot \sum_{s \in S_t} X_{js} + IG_{j,t-1}^{+} - IG_{j,t-1}^{-} - D_{jt} = IG_{jt}^{+} - IG_{jt}^{-} \qquad j = 1 \ldots K, \quad t = 1 \ldots T. \quad (4.18)$$

$$(1-\iota) \cdot \sum_s X_{js} \geq K_{backlog} \cdot IG_{j0}^{-} \qquad j = 1 \ldots K. \quad (4.19)$$

$$X_{js} \leq M_j \cdot Y_{js} \qquad j = 1 \ldots K, \quad s = 1 \ldots S. \quad (4.20)$$

$$X_{js} \geq m_j \cdot Y_{js} \qquad j = 1 \ldots K, \quad s = 1 \ldots S. \quad (4.21)$$

$$\sum_j Y_{js} \leq 1 \qquad s = 1 \ldots S. \quad (4.22)$$

$$Z_{kjs} \geq Y_{k,s-1} + Y_{js} - 1 \qquad j,k = 1 \ldots K, \quad s = 1 \ldots S. \quad (4.23)$$

$$|Y_{vs}^{dig} - Y_{v,s-1}^{dig}| \leq \sum_j \sum_{k,k \neq j} Z_{kjs} \qquad v = 1 \ldots V, \quad s = 1 \ldots S. \quad (4.24)$$

Constraints (4.14) and (4.15) keep track of the usage of virgin and recycled pulp, respectively, during the production and setup tasks in each subperiod (where water is added to it). In the P&P mill, the amount of pulp lost during a grammage switchover is fed back into the tank of recycled pulp and is traced by constraints (4.16).

It should be remembered that there is a common time grid for all production resources. Constraints (4.17) 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 (4.4), synchronize the time grid of the digester with that of the paper machine in each subperiod.

Constraints (4.18) represent the balance of inventory of grammage j in period t. Demand that cannot be met by its due date may be backlogged. The initial inventory of each grammage 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. In spite of backlogging is allowed, orders should not be unfulfilled for a long period of time. Therefore, constraints (4.19) ensure that $K_{backlog}$ of the initial backlog of each grammage is fulfilled until the end of the planning horizon (a cycle).

Constraints (4.20) establish that production of grammage j in subperiod s occurs if the machine is set up for that grammage in subperiod s. In that case, constraints (4.21) force a minimum lot size. From (4.22) at most one type of grammage can be produced per subperiod. Constraints (4.23) relate the changeover variables to the setup state variables. Furthermore, in case the production of two adjacent subperiods is of the same grammage, the speed of the digester remains constant. This assumption is motivated by industrial practice and given by constraints (4.24).

4.2.3 Recovery Plant

Besides the virgin pulp, another by-product is produced during the digester's cooking process: the black liquor. Through a set of intermediate steps (in the evaporator and recovery boiler), the liquor is converted into steam that is directly used to dry the paper or to produce electrical energy. Next, the main variables and constraints that regulate this process are presented.

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	Conversion parameter from digester's rotation to 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)
$C_{steam}^{r.boiler}$	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)
O_s^{steam}	Quantity of steam produced in the recovery boiler (in tonnes)

$$X_s^{liquor} = \rho \cdot \sum_v sp_v \cdot Nh_{vs} \qquad s = 1 \dots S. \tag{4.25}$$

$$X_s^{liquor} + I_{s-1}^{liquor} = O_s^{liquor} + I_s^{liquor} \qquad s = 1 \dots S. \tag{4.26}$$

$$I_{min}^{liquor} \leq I_s^{liquor} \leq I_{max}^{liquor} \qquad s = 1 \dots S. \tag{4.27}$$

$$0 \leq O_s^{liquor} \leq C^{evap} \cdot N_s \qquad s = 1 \dots S. \tag{4.28}$$

$$X_s^{c.liq} = \beta \cdot O_s^{liquor} \qquad s = 1 \dots S. \tag{4.29}$$

$$X_s^{c.liq} + I_{s-1}^{c.liq} = O_s^{c.liq} + I_s^{c.liq} \qquad s = 1 \dots S. \tag{4.30}$$

$$I_{min}^{c.liq} \leq I_s^{c.liq} \leq I_{max}^{c.liq} \qquad s = 1 \dots S. \tag{4.31}$$

$$O_s^{c.liq} \leq C_{burn}^{r.boiler} \cdot N_s \qquad s = 1 \dots S. \tag{4.32}$$

$$O_s^{steam} = \sigma \cdot O_s^{c.liq} \qquad s = 1 \dots S. \tag{4.33}$$

$$O_s^{steam} \leq C_{steam}^{r.boiler} \cdot N_s \qquad s = 1 \dots S. \tag{4.34}$$

The amount of weak black liquor, X_s^{liquor}, expressed in m^3, is proportional to the digester's rotation speed (and therefore to the virgin pulp output), and is given by constraints (4.25).

The evaporator has a capacitated evaporation flow C^{evap} of volume per unit of time (constraints (4.28)). On the way from the digester to the evaporator, there is a tank of weak black liquor that buffers the differences between the throughput of the digester X_s^{liquor} and the quantity that is evaporated in the same period of time, O_s^{liquor}. On the other hand, I_s^{liquor} represents the amount of weak black liquor in the buffer in each subperiod s, constrained by lower and upper bounds (I_{min}^{liquor} and I_{max}^{liquor}, respectively – see constraints (4.27)). The inventory balance equations on the weak black liquor are given by (4.26). Now, 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 (4.29).

The concentrated black liquor goes through a buffer (see Figure 2.1) to smooth downstream variations, before being burnt in the boiler to recover some chemicals and produce steam. Similarly to the buffer of the evaporator, the tank of concentrated black liquor plays an important role when the recovery boiler does not have sufficient capacity to burn all the black liquor coming from the evaporator. The respective tank mass balance equations are given by constraints (4.30), while the storage capacity of the buffer is ensured by (4.31). The burning capacity of the recovery boiler is expressed in tonnes of dried solid content of black liquor (directly converted from the volumetric variable $O_s^{c.liq}$). Constraints (4.32) ensure that the burning capacity is limited in each time 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 (4.33). The steam output can not exceed an upper limit – see constraints (4.34).

4.2.4 Objective Function

Within a capital intensive industry, the short-term production planning and scheduling problem faced by the integrated P&P companies aims at finding production orders that maximize the virgin pulp produced (which is weighted in economical terms), while meeting as much as possible a deterministic demand at a minimum cost. The latter includes the cost of holding stock of each type of grammage, the cost of switching over grammages in the paper machine and the cost of backlogging unmet demand. Expression (4.35) represents the weighted sum of the objective function.

$$\sum_j \sum_t \psi_{jt}^+ \cdot IG_{jt}^+ + \sum_j \sum_t \psi_{jt}^- \cdot IG_{jt}^- + \sum_j \sum_k \sum_s \lambda_{kj} \cdot Z_{kjs} - \sum_s \vartheta_s \cdot X_s^{dig}, \tag{4.35}$$

where $\psi_{j,t}^+$, $\psi_{j,t}^-$, λ_{kj} and ϑ_s denote the weights that aggregate the multiple objectives into a single monetary one.

Chapter 5

Solution Method

5.1 Refined Formulation

This chapter presents the proposed solution method for the problem of the case study analysed in the previous chapter. The formulation proposed by [1] (see Section 4.2) was based on the same case study and hence, it is an important starting point for this work. In addition, it is the only formulation known for doing an integration approach with three of the most important and interrelated production stages: pulping, papermaking and chemical recovery. Therefore, the main structure of the model is adopted and some changes are proposed and discussed.

One of the issues of the formulation by [1] is that it aims to optimize the production of one planning cycle, but does not consider any periods beyond that cycle. As a result, at the end of the planning horizon, all tanks were either at maximum or at minimum capacity. Thus, the production of the next cycle could be strongly constrained, since there would be no slack to raise those tanks at maximum or to lower those at minimum. Indeed, a rolling planning horizon, where some further periods are included in the model (which are going to appear also in the next planning horizon), in order to prepare subsequent production, provides more reliable plans. It seems reasonable to add half of a cycle to the planning horizon considered by [1], as this should be enough to avoid the aforementioned issue. Constraints (4.19) relate only to the first cycle, i.e., the backlog at the beginning of that cycle must be fulfilled until the end of it, since it is the only complete cycle.

In addition, to really have a visible bound separating different cycles, constraints (5.1) and (5.2) were introduced. They impose that the first grammage of each cycle is always the same and thus, they slightly force some standardization of the scheduling patterns.

$$Y_{j_{initial},start_c} = 1 \qquad c = 1 \ldots C. \tag{5.1}$$

$$Y_{j,start_c} = 0 \qquad j \in K \setminus \{j_{initial}\}, \qquad c = 1 \ldots C. \tag{5.2}$$

where C is the set of cycles, $start_c$ is the first subperiod of cycle c and $j_{initial}$ is the common first grammage of all the cycles.

Another issue is related to constraints (4.21). They force the production of a minimum lot size in each subperiod, regardless of being the first subperiod of its campaign or not. Indeed, as seen in Section 3.1.4, minimum lot sizes should be required just in the first subperiod of a given production campaign. On the other hand, if that change is implemented, some subperiods may have null length and thus, the constraints regarding the smooth shift of the digester's speeds would lose their purpose. Therefore, besides the modification of constraints (4.21), it is introduced new constraints to force a minimum subperiod length (see constraints (5.3) and (5.4)).

$$X_{js} \geq m_j \cdot (Y_{js} - Y_{j,s-1}) \qquad j = 1 \ldots K, \qquad s = 1 \ldots S. \tag{5.3}$$

$$N_s \geq N_{min} \qquad s = 1 \ldots S. \tag{5.4}$$

where N_{min} is the minimum subperiod length.

These new constraints should provide a better global optimum, since now there is flexibility to use smaller lot sizes. However, they may also increase the computational time required to exactly solve the model. The same may not be true when removing constraints (4.24). Indeed, not only they shrink the solution space and may worsen the global optimum, but also the tests performed in Section 6.1.2 indicate that they turn the model more demanding regarding computational effort and hence, they are removed.

Constraints (4.23) may also be improved. In fact, there is an alternative formulation for these constraints (see [31]) that provides better computational results. This is also tested (see Section 6.1.2), validated and included into the formulation. The new constraints are as follows:

$$\sum_k Z_{jks} = Y_{j,s-1} \qquad j = 1 \ldots K, \qquad s = 1 \ldots S. \tag{5.5}$$

$$\sum_k Z_{kjs} = Y_{js} \qquad j = 1 \ldots K, \qquad s = 1 \ldots S. \tag{5.6}$$

The objective function was also subject to reform, both in the terms it includes and the weights it considers. Its presentation and discussion is made in Section 6.3, where the impact of different weighting polices is analysed. Hence, the overall modified model (Model 2) reads: min (6.1) , subject to: (4.1) - (4.20), (4.22), (4.25) - (4.34), (5.1), (5.2), (5.3), (5.4), (5.5) and (5.6).

After all these changes and improvements, the model is tested on a real size instance in order to check whether there is a need for more efficient methods or not. As it can be seen in Section 6.1.3, the exact state-of-the-art solver takes too much time in the large instances to deliver a poor solution quality, where the optimality gap is still too high. This was expected since the model is NP-hard [1]. However, there are other problems with much larger instances and additional complexity (e.g. multiple parallel paper machines). Therefore, the use of more efficient methods is mandatory not only for this particular case, but for the P&P industry in general.

5.2 Algorithm Solution Approach

Increasing realism turns the mathematical models that represent industrial environments larger and more complex. This added complexity and the need to increase the size of instances solvable to near-optimality leads the research towards alternative ways (besides exact methods) for solving such problems [32]. As mentioned in the previous section, a state-of-the-art optimization solver fails to generate good quality solutions in a reasonable amount of time for the problem at hand. It is then proposed in this dissertation a more efficient solution approach, based on both (meta)heuristics and exact methods.

In Section 3.2.4.2, the advantages of combining metaheuristics and exact (tree search) methods were discussed, namely how they complement each other. Moreover, mathematical programming-based metaheuristics are more flexible than simple metaheuristics in coping with model extensions, since they do not incorporate so many problem-specific heuristics, as they leave most of the optimization for the exact module.

The easiness with which exact methods deal with continuous variables makes them suitable for the optimization of those variables. On the other hand, discrete variables heavily affect the computational time of this type of method. Metaheuristics are then more appropriate to deal with discrete variables. In the current problem (see Chapter 4) there are two types of discrete variables: those regarding setups and changeovers at the paper machine (Y and Z, which are mutually dependent) and those related to digester's rotation speeds (Y^{dig}). Creating a neighbourhood search based on both would probably result in a non-efficient method, since the number of possible neighbours would be too large. Hence, dividing these two sets of variables into two different algorithm modules should be a wiser approach. As determining the digester's speeds seems to be easier than scheduling grammages in the paper machine, it was developed a heuristic procedure to solve the former and scheduling was left for the metaheuristic. The sizing of the production lots and the other continuous variables (related to the various manufacturing resources) are determined by a solver.

The procedure to generate a neighbour consists of an iterative repetition of the following steps:

1. A setup pattern is determined and the corresponding discrete variables are fixed.
2. An exact solver optimizes the linear programming problem that results from the fixed MIP.
3. The digester's speeds are corrected and fixed.
4. The solver is called again to correct the continuous variables.

It should be noted that in the second step, for the solver to optimize the fixed MIP as a LP problem, the digester's speeds, as well as constraints (4.1) to (4.4), must be removed from the model and constraints (4.6) and (4.25) need to be replaced. The new constraints are as follows:

$$X_s^{dig} \leq \alpha \cdot v_{max}^{dig} \cdot N_s \qquad s = 1 \ldots S. \tag{5.7}$$

$$X_s^{dig} \geq \alpha \cdot v_{min}^{dig} \cdot N_s \qquad s = 1 \ldots S. \tag{5.8}$$

$$X_s^{liquor} = \frac{\rho}{\alpha} \cdot X_s^{dig} \qquad s = 1 \ldots S. \tag{5.9}$$

These constraints just impose upper and lower bounds to the digester's production output (and hence to its speeds), leaving the choice of the speed as a continuous variable decision. In fact the speed is now an implicit continuous decision variable defined by:

$$V_s^{dig} = \frac{X_s^{dig}}{\alpha \cdot N_s} \qquad s = 1 \ldots S. \tag{5.10}$$

To speed up the neighbourhood search, an efficient constructive heuristic was developed. Therefore, an initial solution with reasonable quality is quickly created and the local search should converge faster. However, it should not be too fast, as a premature convergence to a local optimum may result in poor final solution quality. The constructive heuristic, as well as the local search, deals just with the setup pattern (which actually may be seen as the solution representation). Then, the remaining variables are decoded by both the heuristic that determines the speeds of the digester and the exact solver (for continuous variables). In LS, the decoding is performed in each neighbour evaluation. The exact solver chosen was Cplex, which is a state-of-the-art optimization solver.

The next sections explore the various modules that compose the algorithm: the Constructive Heuristic, the heuristic for determining the digester's speeds (Speeds Constraint Heuristic) and the Local Search. Finally, the overall structure of the algorithm is presented.

5.3 Constructive Heuristic

The aim of the constructive heuristic is to create an initial solution quickly and, if possible, with a reasonable quality. As stated in the previous section, the solution representation may be seen as a pattern of the setup variables regarding the paper machine. Therefore, the constructive heuristic algorithm has just to create an initial schedule for the grammages production campaigns. When the schedule is created, the exact solver is called to determine the remaining (continuous) variables. Then, the *Speed Constraint Heuristic* makes the necessary corrections to the digester's speeds and calls again the solver to update the continuous variables.

The algorithm creates two different schedules, with two different sequences, both using ascending and descending orders of grades. For instance, grammages start increasing within a given type of product (e.g. KLB). Then, a changeover to another type (VLB) is done, starting at the highest grade (for the shift to be smooth) and decreasing until the lowest. In this way, the time lost in setups is minimal. Figure 5.1 shows an illustrative example of a sequencing of campaigns starting in ascending order and Algorithm 1 presents the required steps to create it. The opposite sequence may also be implemented, starting in a descending order and then, ascending with the other product type.

From the two created sequences, the one that gives the best results is selected. In this way, it is not necessary to know *apriori* how the sequence was like before the starting period. Regardless of the sequence being increasing or decreasing paper grades, the best direction to follow is always chosen. For each of the sequences, the constructive heuristic creates n schedules, one for each of the n cycles of the planning horizon (in this case $n = 2$, since the planning horizon consists of 1.5 cycles). The schedules are defined, besides the sequence, by the number of subperiods (in this case, shifts) assigned to each production campaign (i.e. grammage).

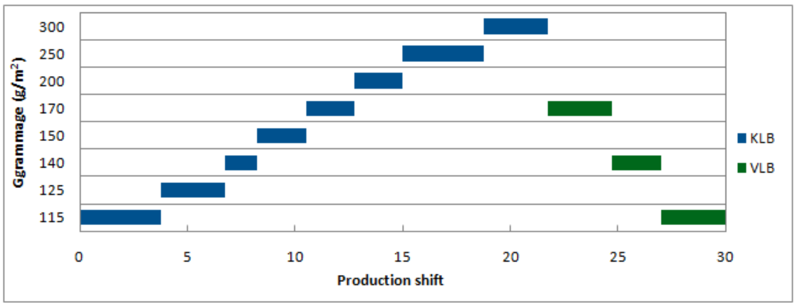


Figure 5.1: Example of a scheduling of paper production campaigns.

Algorithm 1 Sequencing of grammages (starting in ascending order)

Current set of grammages of the same type as $j_{initial}$ with demand or backlog to fulfil: Θ_1
Current set of grammages of the other type with demand or backlog to fulfil: Θ_2
Current schedule: $\pi \leftarrow \emptyset$
$\pi(1) = j_{initial}$
$i = 2$
while $\exists j \in \Theta_1 : grade_j > grade_{\pi(i-1)}$ **do**
    Select the lowest grade of those j: k
    $\Theta_1 \leftarrow \Theta_1 \setminus k$
    $\pi(i) = k$
    $i += 1$
end while
while $\exists j \in \Theta_2$ **do**
    Select the highest grade of those j: k
    $\Theta_2 \leftarrow \Theta_2 \setminus k$
    $\pi(i) = k$
    $i += 1$
end while
while $\exists j \in \Theta_1$ **do**
    Select the lowest grade of those j: k
    $\Theta_1 \leftarrow \Theta_1 \setminus k$
    $\pi(i) = k$
    $i += 1$
end while

The algorithm starts assigning first just the required subperiods that guarantee the fulfilment of the initial backlog (see constraints (4.19) in Section 4.2). If, for instance a given grammage needs 40% of the capacity of a period to produce its backlog quantity, then two subperiods must be assigned to it, in case the period is divided into three subperiods, since one would represent just 33%. In spite of the length of the subperiods may vary in GLSP, if all production campaigns have their number of subperiods assigned according to this procedure, then feasibility (of backlogging covering constraints) is guaranteed, since there is at least one feasible configuration, which is all the subperiods being of equal length. After this first allocation of subperiods, there can be three possibilities. The first is that the number of assigned subperiods is the same as the total available to schedule and thus, the process is finished. The second is that the number of assigned subperiods is greater that those available and, in that case, some campaigns have to have their number of subperiods reduced. The last case, where there are still subperiods to be assigned, requires a procedure to determine how many additional subperiods to assign to each grammage.

As seen before, the number of assigned subperiods in the first step is enough to guarantee feasibility of the backlog covering, but it may have some unnecessary slack ($2 \cdot 33\% - 40\% = 27\%$, in the example previously given). Moreover, even if one or two subperiods are removed from one campaign, it may still fulfil its backlog, as the length of subperiods can vary in GLSP and hence, they can be adjusted in order to create a feasible plan. Therefore, if there have been assigned more subperiods than those available, it is just necessary to iteratively remove one subperiod from one campaign at a time, so as to have the number of assigned subperiods equal to the available ones. In each iteration, the grammage with more "slack" (even if negative) is chosen to be reduced in one subperiod. Thus, the probability of obtaining a feasible solution is kept relatively high.

When the number of assigned subperiods is less than the total available, a procedure is required to determine the additional subperiods to assign to each grammage. This is done by computing and assigning a utility for each grammage, which represents their share of the total remaining subperiods. The utility is computed in three steps:

1. Simple utility of each grammage j: $u_j = max(0, (IG^-_{j,t_0} + \sum_{t \in c} D_{jt} - IG^+_{j,t_0}) \cdot \frac{1}{p_j})$, where t_0 is the last period of the previous cycle.

2. Relative utility of each grammage j: $ru_j = max(0, \frac{u_j}{\sum_i u_i} - \frac{NSubPerB_j}{NTotalAvailable)}$, where $NSubPerB_j$ is the number of previously assigned subperiods (to fulfil backlog) and $NTotalAvailable$ is the total number of available subperiods in the cycle.

3. Adjusted relative utility of each grammage j: $aru_j = \frac{ru_j}{\sum_i ru_i}$.

The notation used is the same as in Section 4.2. In the first step, priority is given to grammages with more backlog and demand to fulfil, considering the initial stock. The result is divided by p_j (processing time of grammage j), since the faster the production is, the greater the percentage of demand that can be fulfilled will be. This is true for campaigns that cannot fulfil all the demand (the other cases are considered later). In the second step, the previous utility is divided by the sum of all utilities in order to obtain a relative value. As backlog had already been taken into account in the first assignment step, the assigned share (number of assigned subperiods over the available ones) is subtracted. Finally, in the last step, the relative utility is divided by the sum of all relative utilities, in order to get values whose sum is 1. It should be noted that in the first two steps, it is computed the maximum value between the corresponding expression and 0, hence assuring non-negative values.

The computed utilities are then multiplied by the remaining subperiods, in order to determine the additional subperiods to assign to each campaign. The result is rounded down, so that the total number of assigned subperiods is not greater than the available. For the grammages whose the assigned subperiods are more than the required to fulfil all the demand and backlog, it is assigned just the necessary amount. Then, if there are still remaining subperiods, they are assigned to the other grammages, according to the gap between the original value and the rounded one.

Finally, in the case all grammages have the required amount of subperiods to fulfil their demand and backlog and there are still subperiods to be assigned, they are distributed according to the demand of the next cycle (if it is not the last) or to the average demand of all cycles (if it is the last cycle).

A simplified version of the constructive heuristic (starting in ascending order) is presented in Algorithm 2. The overall procedure consists of running it for the opposite sequence too.

5.4 Speeds Constraint Heuristic

The removal of the digester's speeds variables from the model, including the constraints that required smooth speed shifts, oblige the use of a heuristic to correct the speeds that result from the fixed MIP optimization. This heuristic goes through each subperiod, computes the digester's speed and compares it to the one of the previous subperiod. If either the upper or the lower limits are violated, the speed value is adjusted and the consequences of these changes to the levels of both virgin pulp and weak black liquor tanks are estimated. Then, the necessary corrections to eliminate the violations of the tank levels are implemented.

In order to get more control over the behaviour of the speed pattern, a small term is added to the objective function: $-\sum_s 0.0001 \cdot (S - s) \cdot X_s^{dig}$. This term has a very low weight in the final objective value, so it will not change the overall optimal plan, except in one aspect: the pulp production (and consequently the virgin pulp and weak black liquor tank levels) will be maximized in cases of indifference. In addition, early subperiods weight more than the subsequent ones. Therefore, the digester's speed will be maximized in the first subperiods, until one of the tanks reaches its maximum value. Thus, at first only the speed's upper limit may be violated and then, when one of the tanks gets full, the speed may be abruptly reduced.

The upper limit violation is easy to correct. It is just necessary to reduce the speed value to the previous speed incremented by the maximum variation ($V_s^{dig} = V_{s-1}^{dig} + \Delta \cdot \Phi$). The problem is the lower limit. In fact, when the speed is abruptly reduced (with the digester's production being maximized), the increase of the speed value will necessarily result in tank levels violation. In these cases, the speeds of previous subperiods have also to be reduced. The defined procedure is as follows:

1. Going through each of the subperiods in a forward move, if the upper limit is violated, the speed is corrected (adding $\Delta \cdot \Phi$ to the previous one) and the tank levels are updated (according to the new speed).

2. If a lower limit violation occurs, the speed is corrected (subtracting $\Delta \cdot \Phi$ to the previous one) and it is further incremented if the tank levels allow for (indeed, eventual previous corrections of the upper limit violation may have reduced tank levels and thus, allow for a

Algorithm 2 Constructive Heuristic (starting in ascending order)

Relax all binary variables
for $c = 1 \to NumCycles$ **do**
Sequencing of grammages (starting in ascending order): consider all grammages with demand or backlog to fulfil in cycle c
Assign to each grammage a number of subperiods that guarantees feasibility
Compute the number of assigned subperiods: $NTotalAssigned$
Compute the number of available subperiods: $NTotalAvailable$
if $NTotalAssigned > NTotalAvailable$ **then**
while $NTotalAssigned > NTotalAvailable$ **do**
Reduce by one subperiod the grammage with more slack
$NTotalAssigned - = 1$
end while
end if
if $NTotalAssigned < NTotalAvailable$ **then**
for $j = 1 \to K$ **do**
Compute the amount of remaining subperiods to assign to grammage j (according to a given utility): $NRemAssignCalc_j = u_j \cdot (NTotalAvailable - NTotalAssigned)$
Compute the integer amount to assign: $NRemAssign_j = int(NRemAssignCalc_j)$
Compute the additional required amount to fulfil demand and backlog: $NReqDemBac_j$
end for
for $j = 1 \to K$ **do**
if $NRemAssign_j > NReqDemBac_j$ **then**
$NRemAssign_j = NReqDemBac_j$
end if
$NTotalAssigned + = NRemAssign_j$
end for
while $NTotalAssigned < NTotalAvailable$ **and** $\exists j \in Grammages : NRemAssign_j < NReqDemBac_j$ **do**
From grammages j where $NRemAssign_j < NReqDemBac_j$, do $NRemAssign_j + = 1$ to the one where $(NRemAssignCalc_j - NRemAssign_j)$ is maximum
end while
while $NTotalAssigned < NTotalAvailable$ **do**
if $c = NumCycles$ **then**
Allocate the remaining subperiods according to the average demand of all cycles
else
Allocate the remaining subperiods according to the demand of the next cycle
end if
end while
end if
Fix setup variables, according to defined sequence and subperiods distribution
Optimize the fixed MIP
Call $SpeedsConstraintHeuristic$
end for

greater speed than it was expected); tank levels are updated too (their eventual violation is estimated and added to the cumulative violation).

3. When the first non-violation of the lower limit occurs after a violation of the lower limit, the process is interrupted and the correction of previous speeds takes place, in order to eliminate all the cumulative violation of the tank levels; the resulted speeds are rounded down (as they must be discrete values and rounding up would result in tank levels violation) and fixed; the model is updated.

4. After the correction procedure, it continues to analyse further subperiods until the last one (where correction is also implemented, if necessary).

5. Finally, the solver is called to update the continuous variables.

Algorithm 3 presents the structured overall procedure.

In order to reduce the speeds of previous subperiods (see line 33 of Algorithm 3), an iterative procedure based on the idea of "layer removal" takes place. For a convenient explanation, take a look at Figure 5.2, which shows an infeasible speed pattern (black line) that needs correction. The horizontal axis represents the time (in hours) over five working days and the vertical grid lines divide some consecutive production shifts (the subperiods). The speed pattern starts at its maximum value (13.5 in this case) and abruptly decreases in a given subperiod, due to one of the tanks has been completely filled. Therefore, a correction is made to the speed pattern in order to eliminate the violations of the smooth speed shifts (red dashed line). However, this adjustment causes a violation in the tank levels, up to the blue area (multiplied by a given factor). Thus, the speed pattern has to be lower than the red dashed line, i.e., some "layers" have to be removed, as in the green dashed line (where one layer was removed) and the purple line (where two layers were removed). The speed pattern has to be lowered until the area beneath it and above the original pattern (that corresponds to additional content inside the tanks) is less than the area above it and under the original pattern (that corresponds to the content that is removed from the tanks). The reductions do not have to be performed through entire layers (this would lead to unnecessary extra reductions). Thus, they are executed subperiod by subperiod, which corresponds to one small rectangle in the figure. The procedure is presented in Algorithm 4.

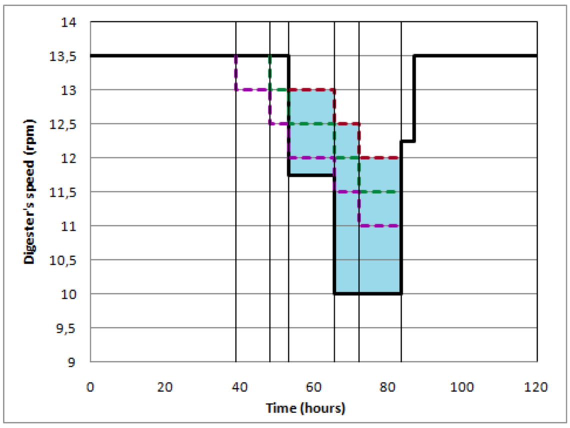


Figure 5.2: Speed pattern correction procedure.

Algorithm 3 Speeds Constraint Heuristic

while $feasible = false$ **do**
Feasibility in all speed shifts: $feasible = true$
First subperiod to reduce its speed: $redFirstSubPer = -1$
for $s = 1 \to NumSubPeriods$ **do**
Digester speed in subperiod s: $V_s^{dig} = \frac{X_s^{dig}}{\alpha \cdot N_s}$
Violation of the lower limit of speed shifts: $violated = false$
if $V_s^{dig} > V_{s-1}^{dig} + \Delta \cdot \Phi$ **and** $redFirstSubPer < 0$ **then**
$feasible = false$
$V_s^{dig} = V_{s-1}^{dig} + \Delta \cdot \Phi$
Update $VirginPulpTankLevel$ and $BlackLiqTankLevel$
end if
if $V_s^{dig} < V_{s-1}^{dig} - \Delta \cdot \Phi$ **then**
$feasible = false$
$V_s^{dig} = V_{s-1}^{dig} - \Delta \cdot \Phi$
$violated = true$
Last subperiod to reduce its speed: $redLastSubPer = s$
if $VirginPulpTankLevel < VirginPulpTankMax$ **and** $BlackLiqTankLevel < BlackLiqTankMax$ **then**
while $V_s^{dig} \leq V_{s-1}^{dig} + \Delta \cdot \Phi$ **and** $V_s^{dig} \leq v_{max}^{dig}$ **and** $VirginPulpTankLevel < VirginPulpTankMax$ **and** $BlackLiqTankLevel < BlackLiqTankMax$ **do**
$V_s^{dig} += \Phi$
end while
$V_s^{dig} -= \Phi$
Update $VirginPulpTankLevel$ and $BlackLiqTankLevel$
else
if $redFirstSubPer < 0$ **then**
$redFirstSubPer = s$
end if
end if
end if
if $redFirstSubPer > 0$ **and** $(violated = false$ **or** $s = NumSubPer)$ **then**
if $s = NumSubPer$ **then**
$feasible = true$
end if
Correct speeds of previous subperiods: Call $LayerRemovalProcedure(redFirstSubPer, redLastSubPer)$
break
end if
end for
Round down the non-discrete speeds and fix all the speed values until s
Update the model with the new speed limits
end while
Optimize the fixed MIP

Algorithm 4 Layer Removal Procedure (redFirstSubPer,redLastSubPer)

Start a new layer: $newlayer = true$
First subperiod of the active layer: $layerStartSubPer = redFirstSubPer$
Active subperiod of the active layer: $layerCurrentSubPer = -1$
while $VirginPulpTankLevel > VirginPulpTankMax$ **or** $BlackLiqTankLevel > BlackLiqTankMax$ **do**
  if $newlayer = true$ **then**
    $layerStartSubPer -= 1$
    if $V^{dig}_{layerStartSubPer} > V^{dig}_{layerStartSubPer-1} - \Delta \cdot \Phi$ **then**
      if $V^{dig}_{layerStartSubPer-1} - \Delta \cdot \Phi > V^{dig}_{0}$ **then**
        Speed decrease: $speedDec = V^{dig}_{layerStartSubPer} - (V^{dig}_{layerStartSubPer-1} - \Delta \cdot \Phi)$
      else
        Speed decrease: $speedDec = V^{dig}_{layerStartSubPer} - V^{dig}_{0}$
      end if
      $V^{dig}_{layerStartSubPer} -= speedDec$
      $VirginPulpTankLevel -= speedDec \cdot \alpha \cdot N_{layerStartSubPer}$
      $BlackLiqTankLevel -= speedDec \cdot \rho \cdot N_{layerStartSubPer}$
      $layerCurrentSubPer = layerStartSubPer + 1$
      $newlayer = false$
    else
      $newlayer = true$
    end if
  else
    if $layerCurrentSubPer \leq redLastSubPer$ **and** $V^{dig}_{layerCurrentSubPer} \geq V^{dig}_{0}$ **then**
      $V^{dig}_{layerCurrentSubPer} -= speedDec$
      $VirginPulpTankLevel -= speedDec \cdot \alpha \cdot N_{layerCurrentSubPer}$
      $BlackLiqTankLevel -= speedDec \cdot \rho \cdot N_{layerCurrentSubPer}$
      $layerCurrentSubPer += 1$
    else
      $newlayer = true$
    end if
  end if
end while

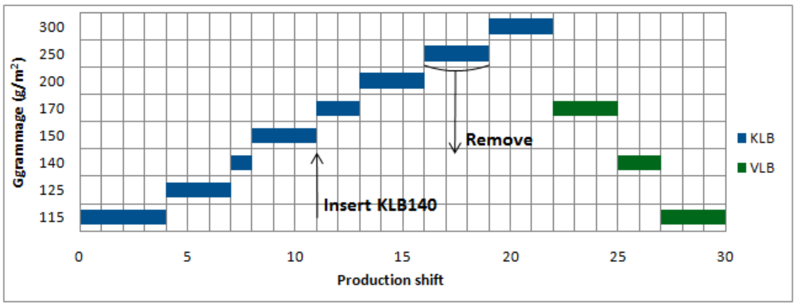
5.5 Local Search

After generating an initial solution, an iterative metaheuristic procedure starts improving the solution. This local search consists of generating neighbours, evaluating them and selecting them if an improvement is achieved. As seen in Section 5.2, the local search deals only with the binary variables regarding the setups of the paper machine. Then, the solution is decoded by both the Speeds Constraint Heuristic and the exact solver. Neighbours are thus generated doing modifications to the setup pattern. The type of modifications to perform defines the neighbourhood structure.

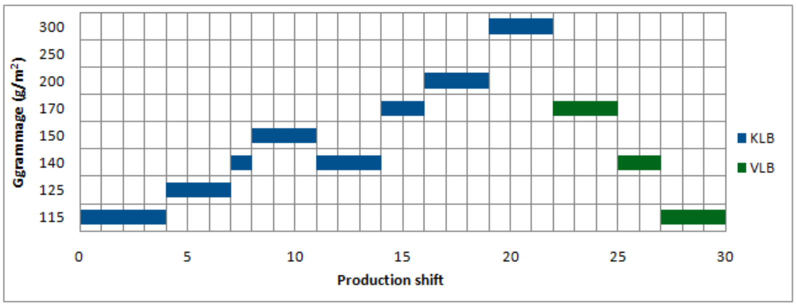
For this problem, three different neighbourhood structures were created. All of them have a feature in common: they are based on combined moves, i.e., it is executed more than one simple move at each time. Indeed, if one setup is inserted into the pattern, the number of total subperiods becomes greater than it was supposed to be. Therefore, for each implemented insertion, a removal has also to be performed, so that the total number of subperiods remains unchanged after the applied move. In the evaluation of each neighbour, a combined move is thus executed and the solver is called twice (before and after the Speeds Constraint Heuristic).

Classic moves include insertion, removal, transfer and swap. These moves can be either applied to single setups or to sets of them. However, the former tend to be somewhat myopic. For instance, a transfer of a single setup, which belongs to a production campaign with two subperiods, to another position in the pattern adds an additional changeover cost that hardly turns the move profitable. It would be necessary to execute two transfers, in order to actually move the whole campaign and hence eliminate the original changeover. This myopia of the single setup moves leads this work towards the multiple setup moves. The defined neighbourhood structures are following presented.

The first consists of a removal of an entire production campaign, followed by the insertion of another campaign with the same number of subperiods. The transfer move is a particular case of this neighbourhood structure, where the selected campaign to insert is the same as that which is removed. However, it includes other possibilities, since a different grammage can be inserted in the same position (where the removal took place) or even a different grammage in a different position. Figure 5.3 shows an example of a move in this neighbourhood.



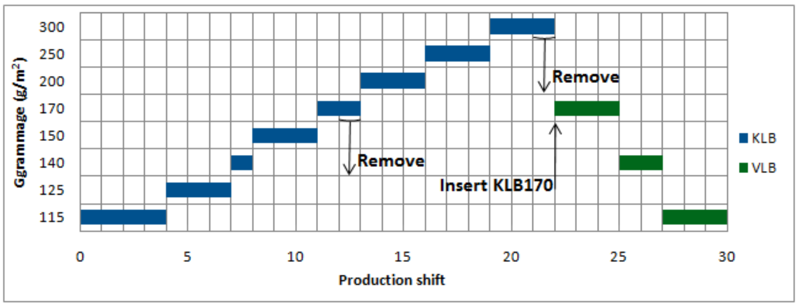
(a) Schedule before the move



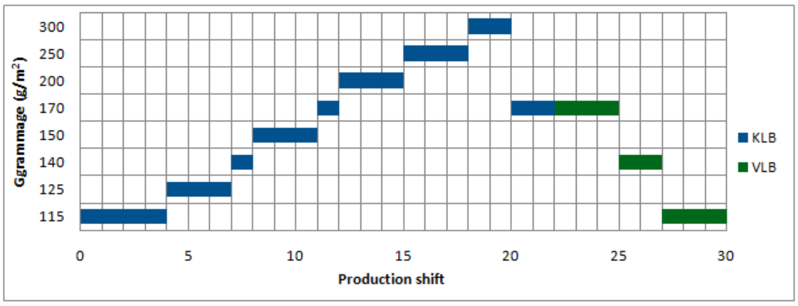
(b) Schedule after the move

Figure 5.3: Example of a move in the first neighbourhood structure (removal and insertion of campaigns).

Another neighbourhood combines the insertion of a production campaign with the removal of single setups, which may belong to different campaigns. Again, this neighbourhood is quite broad, since it includes the splitting of production campaigns, among other cases. Indeed, if the removed subperiods are of the same grammage as the those inserted, it is actually a campaign splitting. To reduce the possibilities, it was defined that the insertion would always be a campaign with two subperiods. This constraint seems reasonable, since it is roughly the average number of subperiods of a production campaign. In Figure 5.4, an example of a move of this type is presented.



(a) Schedule before the move



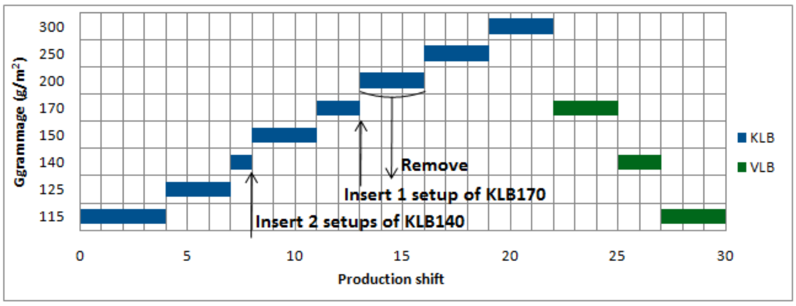
(b) Schedule after the move

Figure 5.4: Example of a move in the second neighbourhood structure (campaign insertion and setup removals).

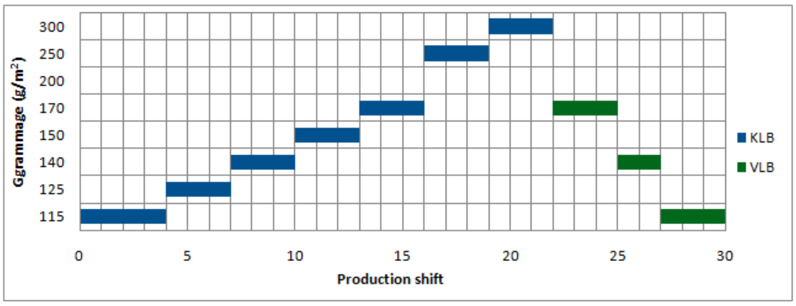
The last defined neighbourhood is the opposite of the previous one: it combines the removal of a production campaign with the insertion of single setups into existing campaigns. It includes, among other possibilities, the union of campaigns, where the inserted setups are of the same grammage as those removed. Figure 5.5 illustrates an example of a move in this neighbourhood.

The aforementioned neighbourhoods are quite extensive, as they include many possibilities and hence, the number of actual neighbours is very large. Moreover, for each neighbour evaluation, it is required to call the solver twice, which strongly increases the computational effort. Therefore, some strategies to speed up the search are adopted.

For the selection strategy, it was chosen the first improvement, since preliminary computational results have shown that it usually provides the same solution quality as the best improvement, but within less computational time. Furthermore, it seems that the probability of premature convergence to a local optimum is smaller (see Section 3.2.3).



(a) Schedule before the move



(b) Schedule after the move

Figure 5.5: Example of a move in the third neighbourhood structure (campaign removal and setup insertions).

In addition, some sorting criteria were implemented, in order to rank the various neighbours according to their likelihood of improving the current (incumbent) solution. Thus, the search should find more quickly an improving solution. On the other hand, if the last ranked moves are not so likely to improve the solution, it is probably not worth to explore the entire neighbourhood, as it is quite large. Therefore, a given parameter determines the percentage of the neighbourhood to explore. In order to rank the different moves, the following criteria are used for each partial movement (i.e., for each removal and insertion that compose the complete move):

- Changeover balance (eliminated setup costs minus additional ones). The higher the gain is, the better the removal/insertion will be.
- Overproduction (current production subtracted by the sum of demand and backlog) of the corresponding grammage. The higher the overproduction is, the better the removal (or the worse the insertion) will be.
- Production rate of the corresponding grammage (applicable only to grammages with negative overproduction, i.e., with underproduction). The faster the production is in grammages with underproduction, the worse the removal (or the better the insertion) will be.
- Average length of the subperiods of the campaign (applicable only to single setup partial moves). The longer the subperiods are, the more likely is for the campaign to need for more subperiods and hence, the better the insertion (or the worse the removal) will be.

Finally, two additional improvements are implemented in order to save computational time. The first is to call the Speeds Constraint Heuristic only if the solution can potentially be better than the incumbent one. Indeed, after fixing the setup pattern and calling the LP solver, the solution is not constrained regarding speed shifts and hence, it can just get worse. Therefore, if the solution is already worse than the incumbent before calling the Speeds Constraint Heuristic, it is not necessary to proceed. The second improvement consists of using dual reoptimization, which consists of interrupting the LP optimization process, if the solution gets worse than the incumbent [29, 28]. This method provides many early rejections of low quality solutions and thus, it can effectively save computational time.

A simplified version of the neighbourhood search (for the case of campaign removal and insertion) is presented in Algorithm 5. The search in the other neighbourhoods follows a similar procedure. It should be noted that the selected move may be invalid (for instance, removing the only campaign of a grammage with backlog to fulfil) and thus, it may be skipped.

Algorithm 5 Neighbourhood Search (campaign removal and insertion)

Best solution: $BestSolution$
Incumbent solution: $IncumbentSolution$
Total number of neighbours: N
Percentage of the neighbourhood to be explored: α
Load ranked production campaign removals
while $k \leq int(\alpha \cdot N)$ **do**
    Select the removal move to evaluate
    Implement the removal
    Load ranked production campaign insertions
    Select the insertion to evaluate
    if the complete move is not valid **then**
        Restore incumbent solution
        break
    end if
    Implement the insertion
    Optimize the fixed MIP
    Objective value returned: $Solution$
    if $Solution < IncumbentSolution$ **then**
        Call $SpeedsConstraintHeuristic$
        Objective value returned: $Solution$
        if $Solution < IncumbentSolution$ **then**
            Store incumbent solution
            if $Solution < BestSolution$ **then**
                Store best solution
            end if
            break
        end if
    end if
    Restore incumbent solution
end while

5.6 Algorithm Framework

The way the previously presented optimization modules are integrated and used in the algorithm consists of its framework and it is critical for both the efficiency and effectiveness of the solution procedure. In this section, the defined framework is presented and discussed.

Some of the combinations between the optimization modules were already seen in the previous sections, namely the incorporation of both the exact solver and the Speeds Constraint Heuristic into the Constructive Heuristic and Local Search. However, it has not been explained how the different local searches (which are built on different neighbourhood structures) are combined. First, it has to be stated that the local searches are applied just to one production cycle at a time. Thus, a loop that goes through each cycle is executed and a local search is performed for each of them.

Different neighbourhood searches lead towards different local optima. Therefore, the basic idea is to escape from the local optimum of a given neighbourhood, changing the local search to another neighbourhood. For this method to be effective, the various neighbourhoods have to be different enough to be able to escape from the local optima of each other.

The defined neighbourhoods of the present work seem to be quite different from each other and it is rather intuitive to observe their complementarity. For instance, the second neighbourhood structure (that combines the insertion of a production campaign with the removal of setups) allows increasing the number of campaigns, which is not possible with the first neighbourhood (that for each insertion, performs a removal of an entire campaign). On the other hand, in order to effectively reduce the number of campaigns, the third neighbourhood (that removes one campaign and inserts single setups) is required. Nevertheless, the neighbourhood structures were tested in sequences of all of them or just part of them. The sequence that has presented the best results was the same as they were presented in this dissertation.

The whole presented method was designed in order to escape from local optimum convergence as most as possible. The non-myopic and very large neighbourhoods, the first improvement strategy and the exploration of just a part of the neighbourhoods are great efforts in doing that. However, the ranking system, which had to be introduced to speed up the search, goes somewhat against that effort. Therefore, it was implemented the initialization of the local searches with different values for the percentage of the neighbourhood to explore. Indeed, if the percentage is great at the outset of the algorithm, the procedure should more likely converge earlier, since it becomes closer to a best improvement strategy, when it is hard to find an improving solution in a given neighbourhood. Thus, initializing the local searches with low values of this parameter may allow the algorithm to avoid premature convergence. Then, when a local optimum is reached, the neighbourhoods can be initialized again, but now with greater values in order to explore additional neighbours. This strategy proved to be quite effective in the instances in which it was tested.

The complete algorithm may be seen as an hybridization at three different lines. First, it is an hybridization between an exact method and a metaheuristic. Second, it combines various local searches (with different neighbourhood structures). Finally, it uses different initializations of local search procedures. The last two hybridizations may be seen as hybrids of metaheuristics with (meta-)heuristics (see Section 3.2.4). The overall procedure is presented in Algorithm 6.

Algorithm 6 Algorithm Framework

```
ConstructiveHeuristic
Percentage of each neighbour m to explore: α_m
Number of consecutive non-improving iterations: n = 0
First neighbour to be explored: m = 1
Number of neighbourhoods: NumNeighbourhoods
while n < NumNeighbourhoods do
    LocalSearch(m, α)
    if an improvement is found then
        n = 0
    else
        n+ = 1
        m+ = 1
        if m > NumNeighbourhoods then
            m = 1
        end if
    end if
    if n = NumNeighbourhoods and any α < 1 then
        n = 0
        m = 1
        for all m do
            α_m = min(2 · α_m, 1)
        end for
    end if
end while
```

Chapter 6

Computational Tests And Generated Results

6.1 Computational Tests

In this section, some computational tests, performed both with an exact solver and the developed algorithm, are presented. The aim is to check and validate the implemented method and to know how it behaves and under what conditions it is efficient. First, some of the proposed changes to the original formulation are tested and validated. Then, the algorithm is compared, both in results and behaviour, to a state-of-the-art exact solver, the IBM Ilog Cplex 12.1, which is used both as mixed integer and linear programming solver. The whole method was implemented in C++, compiled using Microsoft Visual Studio 2008 and run on an Intel Core 2 Quad Q6600 2.40 GHz processing unit with 4 GB of random access memory. The experiments are carried out in some generated instances, derived from real data from the case study.

6.1.1 Test Instances

The case study has provided data for six days of demand, as well as the initial tank levels and all the existing resources parameters. The aim is to optimize an entire production cycle (plus the additional subperiods of the rolling plan horizon), which corresponds to a total of fifteen days. Therefore, additional demand had to be generated for the other nine periods. This new demand was created using the same rate of the first periods, as well as the same temporal spacing.

In addition, other instances were generated based on the first one. Indeed, in order to know the behaviour of the algorithm under different conditions, it has to be tested on different instances. Four main parameters were chosen to vary between them (number of grammages, periods and subperiods and capacity ratio) and two cases were analysed for each parameter, as follows:

- Number of grammages: $K = \{8, 16\}$.
- Number of periods: $T = \{8, 15\}$, with the number of periods per cycle being 5 and 10, respectively.

- Number of subperiods: $|S_t| = \{3,4\}$.

- Capacity ratio: $cap_{ratio} = \{0.6, 0.8\}$.

The original instance may be classified as having: $K = 16$, $T = 15(10+5)$, $|S_t| = 3$ and $cap_{ratio} = 0.745$. The instances with $K = 8$ were then generated selecting randomly eight grammages of the original instance. The capacity ratio was used for the computation of the capacity parameter (cap_t), dividing the total required production time (excluding setups) to fulfil backlog and demand, by the number of periods (multiplied by the capacity ratio).

These instances may represent different moments in time of one mill (for instance, the capacity ratio varies over time, depending on the current demand and backlog) or even different mills (with different number of grammages). The numbers of periods and subperiods are clearly a planning decision. However, they strongly depend on the type of production at hand (which is related to the size and complexity of the plant) and they heavily affect the computational time.

6.1.2 Formulation Alternatives

Two formulation changes proposed in Section 5.1 are here tested and validated with Cplex. First, the reformulation of constraints (4.23) (*model2*) is tested against the original one (*model1*), with both models having all the other proposed modifications, except the removal of constraints (4.24). Then, *model2* is tested without constraints (4.24) (*model3*), in order to verify an eventual improvement. These experiments are performed for the two smallest generated instances ($K = 8$, $T = 8$ and $|S_t| = 3$), as they do not require an extensive test, since they are executed with the same solution method. A time limit of 3600 seconds was imposed for the experiments. Table 6.1 presents the obtained results. The gap denotes the deviation between upper and lower bounds. In order to compare the quality of the solutions provided by different models, the UBGap computes the deviation of the upper bound of each model against the overall best lower bound. The results show a clear superiority of *model2* over *model1* on both instances. The same occurs between *model3* and *model2*.

Table 6.1: Computational results of the formulation alternatives, in generated instances with $K = 8$, $T = 8$ and $|S_t| = 3$. Time is measured in seconds.

	model1			*model2*			*model3*		
cap_{ratio}	Gap	Time	UBGap	Gap	Time	UBGap	Gap	Time	UBGap
0.6	75.1%	3600	9.9%	0.0%	880	0.0%	0.0%	234	0.0%
0.8	56.2%	3600	4.2%	0.5%	3600	0.1%	0.0%	633	0.0%

6.1.3 Algorithm's Results

The developed algorithm is compared with Cplex exact method for the real-based instance and for all the other artificial instances. For practical reasons, it was imposed a one hour time limit for the Algorithm, while for Cplex two hours instead. However, the Algorithm may be executed for a longer time, since it does a complete iteration before checking the stopping criteria. The computational results for the real-based instance and for the generated ones are presented in Table 6.2 and Table 6.3, respectively. The presented gap of the algorithm is computed between its final solution and the lower bound of Cplex.

Table 6.2: Computational results of the Algorithm (compared to Cplex), for the real-based instance. Time is measured in seconds. Best results are in boldface.

Cplex			Algorithm		
Objective	Gap	Time	Objective	Gap	Time
2901	55,6%	7201	**2632**	**51,1%**	**3608**

Table 6.3: Computational results of the Algorithm (compared to Cplex), for generated instances. Time is measured in seconds. Best results are in boldface.

			$K = 8$				$K = 16$			
			Cplex		Algorithm		Cplex		Algorithm	
T	$\|S_t\|$	cap_{ratio}	Gap	Time	Gap	Time	Gap	Time	Gap	Time
8	3	0.6	**0.0%**	**245**	7.0%	258	**5.6%**	7200	11.4%	**396**
		0.8	**0.0%**	656	4.8%	**92**	**9.3%**	**7200**	-	-
	4	0.6	**0.0%**	4483	10.7%	**159**	**37.5%**	7201	38.7%	**1580**
		0.8	**2.9%**	7200	7.8%	**139**	21.1%	7201	**19.8%**	**604**
15	3	0.6	**42.7%**	7200	48.0%	**1742**	62.6%	7201	**59.6%**	**3606**
		0.8	**21.9%**	7200	24.1%	**1522**	44.3%	7201	**43.1%**	**3686**
	4	0.6	**60.6%**	7200	63.4%	**2755**	69.4%	7201	**67.1%**	**3877**
		0.8	41.3%	7200	**41.2%**	**1533**	55.6%	7201	**49.4%**	**3977**

The Algorithm's results for the real-based instance are superior in quality and for half of the running times. The same occurs in the last quadrant of the generated instances (see Table 6.3), which corresponds to the set of the largest ones. Although the differences in quality may seem small, as the gaps are somewhat close to each other, it should be noted that the lower bound of this model is very poor (as it can be seen in Table 6.1). Therefore, small differences in gap may correspond to great differences in the final solution quality. In addition, none of the computational experiments performed for these larger sized instances has proved to reach a local optimum, as they all were still exploring neighbourhoods when they were stopped.

For the medium and small sized instances, the Cplex obtains a better solution quality, although the Algorithm performs in a much shorter time. In some medium instances, the Algorithm can provide a very close (or even better) solution, while using much less time. However, this is not true for the small instances. Indeed, in for smallest one, the Cplex proves to be superior both in quality and time. In fact, the advantage of the Algorithm lies in its performance in large sized instances. The small generated instances (and even the medium ones) do not represent real world sized problems of most of the companies in the industry. Indeed, some of them can have significantly larger problems than the case study at hand, both in complexity and size of the instances. Therefore, the Algorithm may be a helpful tool for many companies.

6.1.4 Algorithm's Behaviour

The behaviour of the algorithm is analysed for the real-world based instance of our case study. Its improvements along the time are compared to Cplex's performance for the same instance. Figure 6.1 shows this comparison along their total active time (3600s for the Algorithm and 7200s for the Cplex). The improvements are plotted and it is highlighted the module of the Algorithm that has performed it (both the Constructive Heuristic and the neighbourhood searches). *Neighbourhood1* corresponds to the one that combines a campaign removal with a campaign insertion and *Neighbourhood2* consists of inserting one campaign and removing single setups from others.

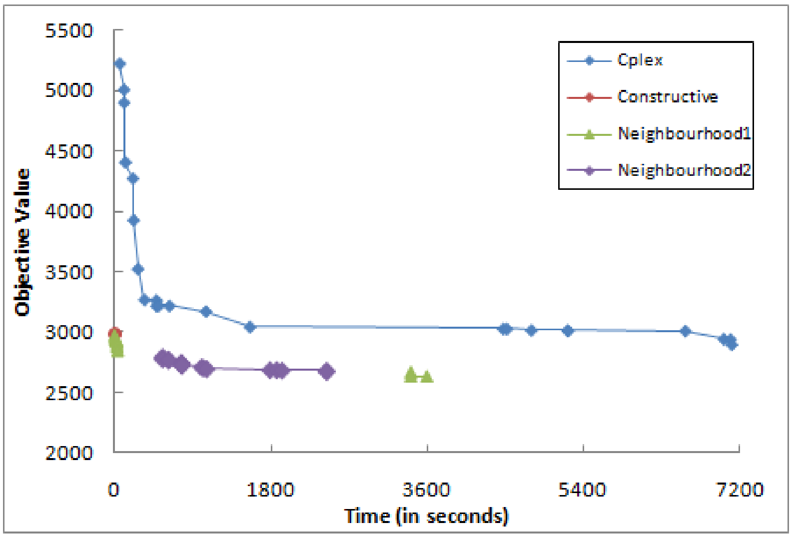


Figure 6.1: Algorithm's behaviour (compared to Cplex), for the real-based instance.

The most obvious aspect of this comparison is the initial solution which both methods start with. Indeed, the Constructive Heuristic provides a much better solution quality in a very short computational time. The Algorithm actually does several improvements before the Cplex even generates an initial solution. Additionally, it is clear that when a local optimum of a given neighbourhood is reached, another neighbourhood search is able to continue the improvement procedure along some further iterations.

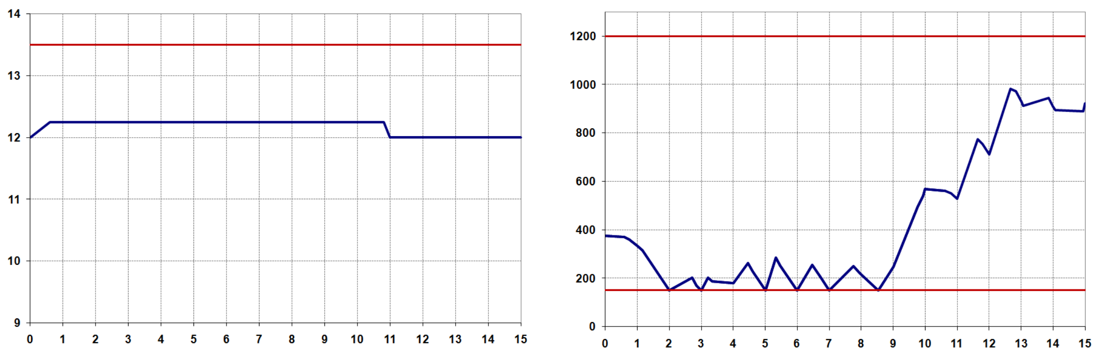
In Cplex's performance, it can be seen that there is a first stage, where large improvements are achieved. Afterwards, it steadies in a given solution for a very long time. Further significant improvements are only achieved at the end of the experiment. Still, within the two hours of the experiment, it reaches a solution only slightly better than that generated by the Constructive Heuristic.

In this particular case, one of the neighbourhood structures is not used. However, as the Algorithm's local optimum has not been reached, it could still be used in further iterations. In other instances, all the neighbourhood structures are used within the imposed time limit.

6.2 Generated Production Plan Analysis

The proposed solution method was used to generate a complete production plan for the real-based instance. The resulted plan is analysed in this section. Figures 6.2 to 6.6 present the planned state for the main production resources. Some outputs are discretized in days (e.g. Figure 6.3 (a) consists of the daily aggregated production of the recycled pulp mill), while in other variables the dynamics within the day are illustrated (e.g. Figure 6.2 (a)). In most of the figures, a blue line represents the quantitative variable that is being presented in the corresponding figure, while the red horizontal lines are the upper and/or lower physical limits constraining that variable.

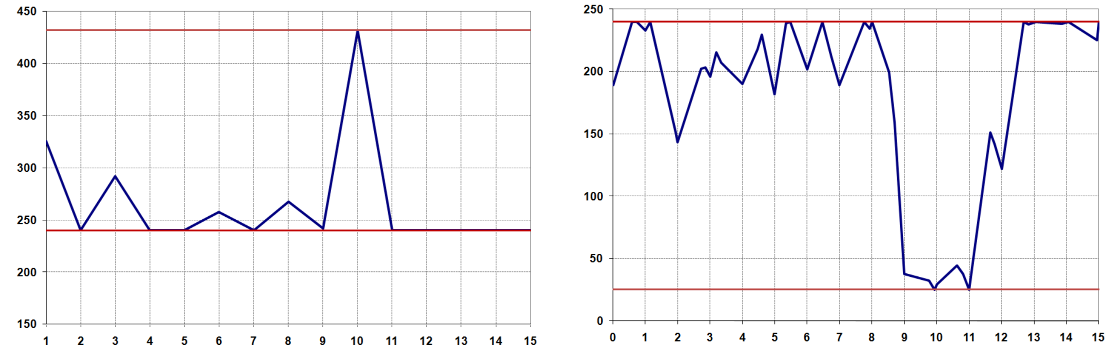
In Figure 6.2, it can be observed that the digester's speed is very steady, as it only changes its value twice. However, it does not reach its maximum at any point in time and hence, it is constrained by one of the downstream resources. Since the virgin pulp tank is never full (see 6.2 (b)), it is the recovery plant that is constraining the pulp mill production output. In fact, Figure 6.5 (b) shows that the weak black liquor tank level reaches its maximum more than once. It should be also noted that there are instants where the virgin pulp tank is at minimum and the weak black liquor tank is at maximum (e.g. end of period 6). This may explain why the digester's speed has to be steady and set at a moderate value. Indeed, it has to conciliate those two opposite cases and hence, it must not create imbalances by changing its rate from a given equilibrium value.



(a) Speeds (*rpm*) (b) Inventory (*tonnes*)

Figure 6.2: Virgin pulp mill: digester's speeds (a) and virgin pulp inventory (b).

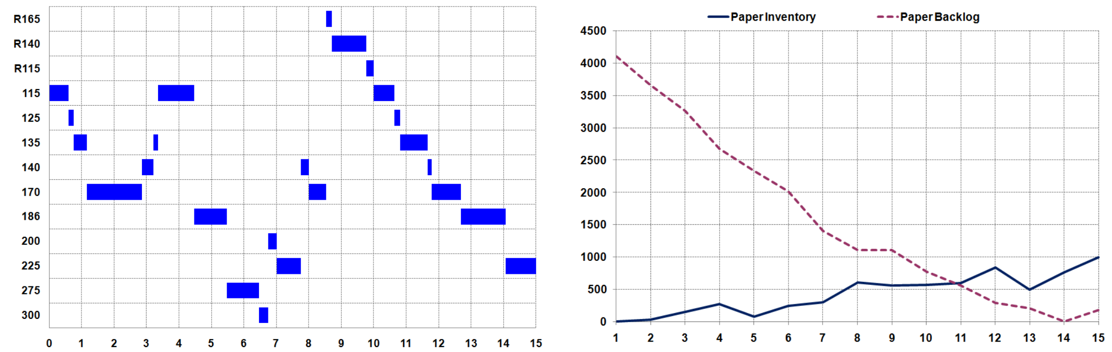
The company produces two main types of paper, which differ in the quantity of incorporated recycled fibres. Figure 6.4 (a) represents some grammages starting with a "R", which correspond to those incorporating a higher portion of recycled fibres. As it can be seen in Figure 6.3, it is during their production that the recycled pulp plant has to maximize its output. However, it is far from being a bottleneck. In fact, one can see in this case that during the 10^{th} period, the inventory of recycled pulp only slightly decreases, although the "R" grammages are being produced along the entire period. Therefore, the campaigns of these grammages could even have started right after the first (fixed) grammage and have had considerably large lengths, that it would not be a problem for this resource. The same may not be true in the opposite case, where the absence of "R" grammages can actually constrain the overall P&P mill. Indeed, it is illustrated in this case that even producing at minimum capacity, the inventory of recycled pulp can sharply increase during the production of low "normal" grammages (see subperiods 11^{th} to 13^{th}). Thus, it might not be possible to produce very long campaigns of these grammages, due to the reycled pulp plant lower capacity bound.



(a) Production ($tonnes/day$) (b) Inventory ($tonnes$)

Figure 6.3: Recycled pulp mill: production (a) and inventory (b) of recycled pulp.

In Figure 6.4 (a), it can be observed that the schedule tries to cover all the grammages through a continuous smooth pattern, in order to minimize the time lost in the sequence-dependent setups, while meeting demand and minimizing backlog (see Figure 6.4 (b)). In addition, changeovers from one type of pulp to another are done with similar grades (from 170 to R165 and from R115 to 115).

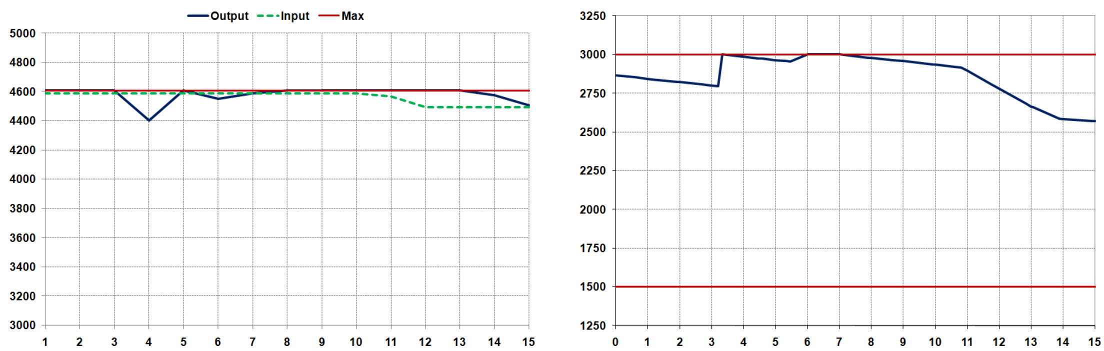


(a) Schedule (b) Inventory and backlog ($tonnes$)

Figure 6.4: Paper mill: campaigns schedule (a) and paper inventory and backlog (b).

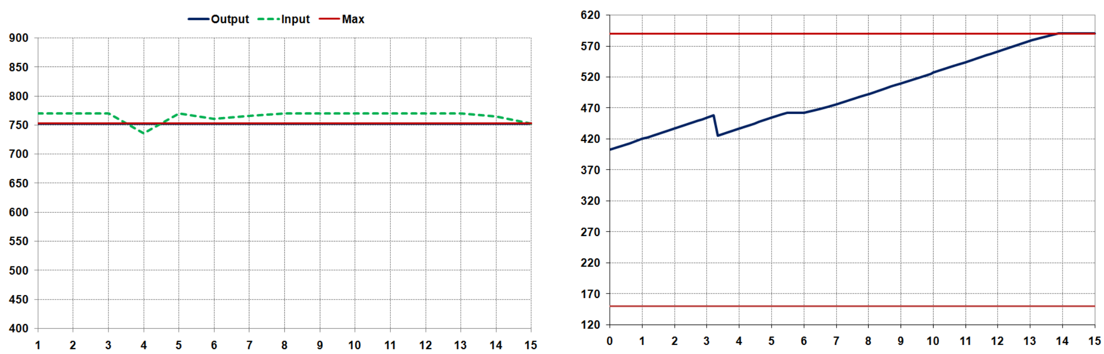
In what concerns the recovery plant, Figures 6.5 and 6.6 show the mass flow through both the evaporator and the recovery boiler, as well as the levels of the buffers between those resources. In spite of the input and the output being different from each other within each of the resources (e.g. concentrated black liquor enters the recovery boiler and steam goes out), they are represented in the same equivalent units, in order to better understand the mass balance.

As seen before, the bottleneck lies at this plant. Indeed, the absolute bottleneck is the recovery boiler, as it can be seen in Figure 6.6 (a), where the output and the maximum capacity lines are overlapping with each other. In addition, the buffer before the recovery boiler (the concentrated black liquor tank) is not able to absorb all the incoming liquor provided by the evaporator. Thus, the last one is forced to reduce its production below its maximum rate at certain times, which causes the complete filling of the weak black liquor tank (note the symmetric behaviour of the black liquor tanks, which is due to their inverse behaviour when responding to the evaporator production). The weak black liquor tank, in its turn, constrains the digester's speeds and hence, the output of both the pulp mill and the paper mill.



(a) Mass flow (m^3/day) (b) Inventory (m^3)

Figure 6.5: Weak black liquor: mass flow at the evaporator unit (a) and inventory before that stage (b).



(a) Mass flow (m^3/day) (b) Inventory (m^3)

Figure 6.6: Concentrated black liquor: mass flow at the recovery boiler (a) and inventory before that stage (b).

6.3 Objective Function Analysis

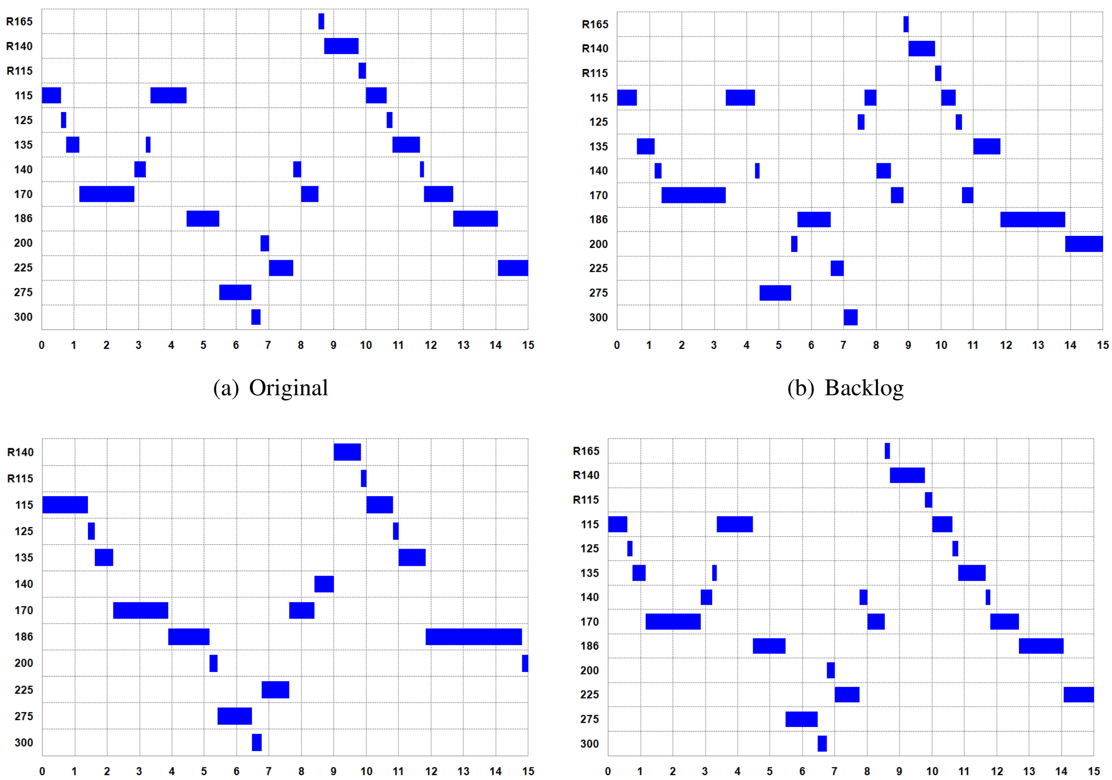
The objective function is one of the most important ingredients of any optimization method. In fact, it guides the optimization procedure and determines the planning policy to be followed. The company's KPIs (and the underlying strategy) should be behind the definition and parametrization of the objective function. In this section, it is proposed a comprehensive objective function and its terms are weighted in an empirical way. The impact of different weighting policies is also analysed, considering the resulting planning schedules.

Including a term regarding digester's production provides the maximization of the overall P&P mill productivity, as all the other resources will produce at their full capacity with a mass pushing strategy. However, it does not do it in a complete direct way, as there can be a lag between the pulp production and the output of the recovery boiler, for example. Thus, in some particular cases, future periods may be constrained more than expected. Therefore, it was added to the objective function proposed by [1] the output of the digester's downstream resources, namely the recovery boiler and the evaporator. The output of the paper mill is not necessary, since it is already measured by the actual backlog. In addition, the output of the recycled pulp plant was included too, as well as the term regarding the digester's output, mentioned in Section 5.4. The final expression is following presented.

$$\sum_j \sum_t 0.01 \cdot IG^+_{jt} + \sum_j \sum_t 0.1 \cdot IG^-_{jt} + \sum_j \sum_k \sum_s 5 \cdot sl_{kj} \cdot Z_{kjs} - \sum_s 0.01 \cdot O^{steam}_s - \sum_s 0.001 \cdot O^{liquor}_s - \sum_s 0.001 \cdot X^{recy}_s - \sum_s 0.0001 \cdot (S-s) \cdot X^{dig}_s \tag{6.1}$$

This objective function has provided an apparently reasonable production plan (seen in the previous section). A sensitivity analysis, where some of the weights of the objective function are modified, is presented here. Figure 6.7 shows the planning schedules for four different objective functions: the original (already presented in the previous section), one weighting more backlog, other weighting more setups and the last one weighting more the overall productivity (measured by the digester's output). In the first two cases, the coefficient in the objective function is multiplied by 3. In the last one, it is introduced the following term: $-\sum_s X^{dig}_s$.

When the objective function weights more backlog, the setup pattern tends to escape from the smooth shape presented by the original one (see 6.7 (a) and (b)). In fact, since backlog becomes much more important than setup costs, grammages are produced as they are required to meet customer's demand. On the other hand, when weighting more setups, the pattern comes closer to a perfect sinusoid, providing very smooth changeovers (see 6.7 (c)). It can be observed that in this schedule, the lengths of the production campaigns are greater, as they are hardly repeated, and that there is no production for grammage R165, since it does not have any backlog at the beginning of the planning horizon (and hence, its production is not mandatory). Finally, the last schedule, which weights more the overall productivity, is the same as the original. Indeed, the original objective function already maximizes productivity, since the recovery boiler is producing at its maximum rate and the other resources are producing the most the recovery boiler allows them to.



(a) Original

(b) Backlog

(c) Setups

(d) Productivity

Figure 6.7: Campaigns schedules for different objective functions: the original (a) and other with higher weights in different parameters: backlog (b), setups (c) and overall productivity (d).

Chapter 7

Conclusions And Future Work

This dissertation investigates the optimization of the short-to-medium term production planning of pulp, paper and steam, under an integrated approach. The interdependency between the various production units has been neglected by researchers not only because of modelling complexity, but also for solvability reasons. The work [1] is actually the only one known for modelling the dynamic mass and material balance in an integrated P&P mill, through the integration of three of the most important and interrelated production stages: pulping, papermaking and chemical recovery. However, the computational complexity of the proposed formulation is too high for exact methods to obtain good solutions in large real-world instances and within reasonable time. Therefore, this dissertation proposes an efficient algorithm, based on both metaheuristics and exact methods, for solving an integrated model of this complexity.

First, the formulation is refined and some improvements are implemented, in order to turn the model easier to solve by exact methods. Nevertheless, the use of more efficient methods remains mandatory, since the computational time to solve real-world instances is still prohibitive.

The solution approach consists of creating an initial solution with a constructive heuristic and then, improving it with local search. These two main optimization modules deal only with the setup pattern, while the remaining variables are decoded by an heuristic to determine the digester's speeds and an exact solver to optimize the continuous variables. The local search explores different neighbourhoods interchangeably and initializes them with a progressive percentage of their domain to be explored. In fact, the defined neighbourhood structures are too large to be completely explored in each move, considering that at least one linear programming problem has to be solved in each neighbour evaluation.

The computational results demonstrate a clear superiority of the Algorithm, compared to Cplex, both in solution quality and computational time, in the real sized instances. Without any diversification strategy besides the exploration of different neighbourhoods (and the way they are switched between them), the Algorithm is not able to find the global optimum in the smallest instances. However, it does not reach a local optimum during one hour of computational tests, in any of the largest instances. In fact, these experiments were still exploring neighbourhoods when they were stopped. Therefore, possible improvements to be applied to the solution method should start with the efficiency of its modules, rather than diversification strategies.

There are two main ways to improve the efficiency of the Algorithm. One is to study and implement new neighbourhood structures, as well as their cooperation along the solution procedure. The other consists of developing ways to speed up the evaluation of neighbours, which weighs too much in computational time, as it requires solving linear programming problems. As stated before, the defined neighbourhood structures are very large, as they comprise a wide variety of possible moves. For instance, in the combined campaign removal and insertion, the transfer move consists of only a small sub-space of the whole neighbourhood. Therefore, more focused small neighbourhood structures should be studied, in order to speed up at least some phase within the solution procedure. The improvement of the efficiency in neighbours evaluation can be done for instance using duality in the ranking process or accelerating the linear programming solving, through a further enhancement of the formulation or finding strategies that reduce computational time.

When the Algorithm becomes efficient enough to converge quickly to a (good) local optimum, then, diversification strategies must be applied. Incorporating a stochastic version of the current method or implementing a shaking phase are two possible ways to explore new areas of the solution space. Having a strong diversification strategy, the Algorithm should reach even better solutions in large instances and probably find the optimum in small instances.

In addition, the solution method has to be presented to the company. It is imperative to fully discuss the formulation in general and the objective function in particular. Indeed, as stated before, the objective function is the guide of the optimization procedure and it should be defined by the company's KPIs. The incorporation of the Algorithm into a multi-objective approach is also to be considered, since it provides different production plans and turns the managers' decisions more practical and intuitive. Nevertheless, the current solution procedure may be already compared with the company's manual planning system and the resulted plans validated.

As seen in Section 6.2, the generated production plans demonstrate a clear bottleneck in the recovery boiler. Also, some tanks prove not to have enough capacity to absorb all the mass flow imbalances. Therefore, the current dissertation may already provide useful information for expansion strategies to be adopted by the company. Indeed, the Algorithm may incorporate a decision support system (DSS), not only to optimize the short-to-medium term production planning, but also to allow managers to perform what-if analysis regarding expansion strategies of the production resources.

This dissertation will thus hopefully motivate extensive further research, as both the P&P industry and the hybrid optimization approaches have a lot of potential to be explored.

References

[1] M.O. Santos and B. Almada-Lobo. Integrated pulp and paper mill planning and scheduling. 2011. Paper submited to *Computers & Industrial Engineering*.

[2] M. Faleiros. Portugal seeks technology in paper and alternative markets, November 2009. *O Papel*, available at http://www.revistaopapel.org.br/noticia-anexos/1259846385_14145506ef06cfdc93d0121c2cc9bda6_2064007827.pdf, accessed in 30-05-2011.

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

[4] 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.

[5] D. Jankunaite. Pulp and paper industry - case study. *SCA Sundsvall*, available at http://www.sca.com/sv/Sundsvall/, accessed in 30-05-2011.

[6] A. Araújo, J.A. Ferreira, N. Fernandes, and P. de Oliveira. Using splitting methods in continuous digester modeling. *Applied Mathematical Modelling*, 32(9):1869 – 1882, 2008.

[7] Y. Crama, Y. Pochet, and Y. Wera. A discussion of production planning approaches in the process industry. CORE Discussion Papers 2001042, Université catholique de Louvain, Center for Operations Research and Econometrics (CORE), October 2001.

[8] J. Peuhkuri. Stabilizing the continuous digester process, October 2009. *RISI*, available at http://www.risiinfo.com/techchannels/pulping/Stabilizing-the-continuous-digester-process.html, accessed in 30-05-2011.

[9] H. Gunnarsson and M. Rönnqvist. Solving a multi-period supply chain problem for a pulp company using heuristics–an application to södra cell ab. *International Journal of Production Economics*, 116(1):75 – 94, 2008.

[10] N. Rizk, A. Martel, and S. D'Amours. Production planning in the pulp and paper industry. *Working Paper DT-2004-AM-3*, pages 1 – 10, 2004.

[11] 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.

[12] 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.

[13] S. Klugman, M. Karlsson, and B. Moshfegh. A swedish integrated pulp and paper mill–energy optimisation and local heat cooperation. *Energy Policy*, 37(7):2514 – 2524, 2009.

[14] J. Jönsson, 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. Transition towards Sustainable Energy Systems.

[15] M.H. Correia, J.F. Oliveira, and J.S. Ferreira. Reel and sheet cutting at a paper mill. *Computers & Operations Research*, 31(8):1223 – 1243, 2004.

[16] 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.

[17] B. Almada-Lobo. *Lotsizing and Scheduling in the Glass Container Industry*. PhD thesis, Faculdade de Engenharia da Universidade do Porto, 2007.

[18] R. Jans and Z. Degraeve. Modeling industrial lot sizing problems: a review. *International Journal of Production Research*, 46(6):1619–1643, 2008.

[19] B. Karimi, S.M. Ghomi, and J.M. Wilson. The capacitated lot sizing problem: a review of models and algorithms. *Omega-International Journal of Management Science*, 31(5):365–378, 2003.

[20] C.D. Eppen. Solving multi-item capacitated lot-sizing problems using variable redefinition. *Operations Research*, 35(6):832–848, 1987.

[21] A.T. Staggemeier and A.R. Clark. A survey of lot-sizing and scheduling models. 2001.

[22] E.G. Talbi. *Metaheuristics: From Design to Implementation*. Wiley Publishing, 2009.

[23] R. James and B. Almada-Lobo. Single and parallel machine capacitated lotsizing and scheduling: New iterative mip-based neighborhood search heuristics. *Computers & Operations Research*, 38(12):1816 – 1825, 2011.

[24] Y. Pochet and L.A. Wolsey. *Production planning by mixed integer programming*. Springer series in operations research and financial engineering, 2006.

[25] C. Blum and A. Roli. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Comput. Surv.*, 35:268–308, September 2003.

[26] C. Blum, J. Puchinger, G.R. Raidl, and A. Roli. Hybrid metaheuristics in combinatorial optimization: A survey. *Applied Soft Computing*, 11(6):4135 – 4151, 2011.

[27] J. Puchinger and G.R. Raidl. Combining metaheuristics and exact algorithms in combinatorial optimization: A survey and classification. 3562:113–124, 2005.

[28] L. Guimarães, D. Klabjan, and B. Almada-Lobo. Annual production budget in the beverage industry. *Engineering Applications of Artificial Intelligence*, In Press, Corrected Proof, 2011.

[29] H. Meyr. Simultaneous lotsizing and scheduling on parallel machines. *European Journal of Operational Research*, 139(2):277 – 292, 2002.

[30] V. Camargo, F. Toledo, and B. Almada-Lobo. Three time-based scale formulations for the two stage lot sizing and scheduling in process industries. 2011. Paper submited to *Journal of the Operational Research Society*.

[31] C. Gicquel, N. Miègeville, M. Minoux, and Y. Dallery. Discrete lot sizing and scheduling using product decomposition into attributes. *Computers & Operations Research*, 36(9):2690 – 2698, 2009.

[32] 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.