Lotsizing and Scheduling in the Glass Container Industry

Bernardo Sobrinho Simões de Almada-Lobo

September 2007

Bernardo Sobrinho Simões de Almada-Lobo

Lotsizing and Scheduling in the Glass Container Industry

Submitted to Faculdade de Engenharia da Universidade do Porto in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Industrial Engineering and Management, supervised by José Fernando da Costa Oliveira, Associate Professor of Faculdade de Engenharia da Universidade do Porto and by Maria Antónia Carravilla, Assistant Professor of Faculdade de Engenharia da Universidade do Porto

Department of Mechanical Engineering and Industrial Management
Faculdade de Engenharia da Universidade do Porto
2007

This research was partially supported by the company BAVidro, SA and by the PhD grant SFRH/BD/23987/2005 awarded by the Portuguese Foundation for Science and Technology.

“Plans are nothing. Planning is everything.” – Dwight D. Eisenhower

Acknowledgements

First and foremost, I would like to thank my advisors Maria Antónia Carravilla and José Fernando Oliveira for their guidance, support and motivation throughout my PhD. It has been a great pleasure to work with them. Their enthusiasm is contagious and their commitment to the academic job and their dedication to the well-being of their students will inspire me throughout my career.

I am deeply grateful to Professor Rui Guimarães. He initiated me to Operations Research, supervised me during my internship in BaVidros, SA, developed my (still scarce, he would say) abilities to write, made me discover the academic world, encouraged me to pursue a doctoral degree and appointed me José Fernando Oliveira as an outstanding potential supervisor.

I am much indebted to Carlos Moreira da Silva, chairman of BaVidros, SA, for the opportunity to collaborate on an extremely interesting industrial problem. His belief in the project is much appreciated. A very special thanks goes out to Ricardo Pinho from the production planning department, who helped me overcome some of the challenges of problem-driven operations research.

I would also like to acknowledge Stephen C. Graves from the Sloan School of Management for hosting me at the Operations Research Center at MIT. His clear managerial insights are worth of note. At MIT I had the chance to meet Diego Klabjan. Diego has significantly improved the quality of this work and have opened directions for future research.

I would further like to thank my brother, Francisco, for teaching me most of what I know about programming and for helping me on several codes.

I owe thanks to Kostas, Theo, Ilan and Ruben for making me believe that one can find non-nerd good friends at MIT.

The daily lunches with my GEIN/DEMEGI friends Sarsfield, Henriqueta, Almacinha, Teresa and Fonseca are unforgettable and made (still make) each day more pleasant, giving me the opportunity to learn a lot on different subjects.

Thank you Mãe, thank you Pai, for instilling me discipline and eagerness to know more, for motivating and supporting me continuously and for being who you are and who you have made me.

Thank you Inesinha... for everything.

Abstract

Manufacturing organizations are keen to improve their competitive position in the global marketplace by increasing operational performance. Production planning is crucial to this end and represents one of the most challenging tasks managers are facing today. Among a large number of alternatives, production planning processes help decision-making by trading-off conflicting objectives in the presence of technological, marketing and financial constraints. Two important classes of such problems are lotsizing and scheduling. Proofs from complexity theory supported by computational experiments clearly show the hardness of solving lotsizing and scheduling problems.

Motivated by a real-world case, the glass container industry production planning and scheduling problem is studied in depth. Due to its inherent complexity and to the frequent interdependencies between decisions that are made at and affect different organizational echelons, the system is decomposed into a two-level hierarchically organized planning structure: long-term and short-term levels.

This dissertation explores extensions of lotsizing and scheduling problems that appear in both levels. We address these variants in two research directions. On one hand, we develop and implement different approaches to obtain good quality solutions, as metaheuristics (namely variable neighborhood search) and Lagrangian-based heuristics, as well as other special-purpose heuristics. On the other hand, we try to combine new stronger models and valid inequalities based on the polyhedral structure of these problems to tighten linear relaxations and speed up the solution process.

Resumo

As empresas industriais estão focadas em melhorar a sua competitividade no mercado global através de um aumento da performance operacional. O planeamento de produção é crucial para este fim e representa um dos principais desafios com que os gestores se confrontam hoje. De entre um número elevado de alternativas, o processo de planeamento de produção auxilia a tomada de decisão ao estabelecer compromissos entre objectivos conflituosos na presença de restrições tecnológicas, comerciais e financeiras. O dimensionamento e o escalonamento de lotes representam duas classes importantes deste tipo de problemas. Demonstrações da teoria da complexidade suportadas por experiências computacionais mostram claramente a dificuldade de resolução de problemas de dimensionamento e escalonamento de lotes.

Motivado por um caso real, esta dissertação aborda em profundidade o problema de planeamento e escalonamento de produção da indústria de embalagens de vidro. Devido à sua complexidade e às frequentes interdependências entre decisões que são tomadas em diferentes níveis organizacionais, o sistema é decomposto numa estrutura de planeamento de dois níveis hierárquicos: longo-prazo e curto-prazo.

Exploram-se extensões de problemas de dimensionamento e escalonamento de lotes que surgem em ambos os níveis. Estas variantes são analisadas em duas linhas de investigação. Por um lado, desenvolvem-se e implementam-se abordagens eficientes para gerar soluções de boa qualidade, tais como metaheurísticas (nomeadamente pesquisa de vizinhança variável, heurísticas Lagrangeanas, bem como outras mais específicas). Por outro lado, combinam-se modelos mais fortes com desigualdades válidas baseadas na estrutura poliédrica destes problemas, para estreitar as relaxações lineares e tornar mais célere a sua resolução.

Résumé

Les entreprises industrielles sont très intéressées par l'amélioration de leur compétitivité sur le marché global via l'augmentation de la performance opérationnelle. La planification de production est cruciale pour arriver à ce but et représente une des taches les plus stratégiques auxquelles les managers sont aujourd'hui confrontés. Grâce à un large panel d'alternatives, les processus de planification de la production aident à la prise de décision via l'arbitrage d'objectives conflictuels ayant des contraintes technologiques, marketing ou financières. Deux des principaux types de ces problèmes sont le dimensionnement des lots et la programmation des taches. La théorie de la complexité appuyée par des expériences computationnelles démontre clairement la difficulté de résolution de ces deux enjeux majeurs.

Motivé par un cas réel, la planification de production et la programmation horaire de l'industrie des conteneurs (emballage) en verre sont étudié en profondeur. De fait de sa complexité inhérente, et des interdépendances fréquentes entre niveau de décision et niveau d'exécution des directives, le système est décomposé en une structure de planification à deux niveaux hiérarchiques : le long terme et le court terme.

Cette dissertation tend à analyser les problématiques de dimensionnement de lots et de programmation des taches qui apparaissent dans les deux niveaux. Nous adressons ces éléments par deux approches différentes. D'un coté, nous développerons et implémenterons différentes stratégies pour arriver à de bonnes solutions qualitatives: les métaheuristiques ou recherche de variable de voisinage, les heuristiques basé le multiplicateur de Lagrange, ou d'autres heuristiques particulières. De l'autre, nous essayons de combiner des nouveaux modèles plus fiables et les "valid inequalities" de la structure polyhédrale de ces problèmes, de façon à serrer les relaxations linéaires et à accélérer le processus de solution.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

In an increasingly competitive global marketplace, production planning lies at the very heart of the performance of industrial enterprizes. 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. Apart from resource acquisition, decisions are typically operational (short-term) to tactical (medium-term) planning problems, such as work force level, overtime assignment, production lot sizes and sequencing of production runs. The focus of the thesis will be the short/medium-term scope, namely two of the most important and challenging production planning problems: lotsizing and scheduling.

The lotsizing problem consists in finding production orders or lots in order to satisfy customer demand at minimum costs. This problem has been intensively studied and, therefore, many models were proposed to describe it, involving different features and assumptions. Naturally, the complexity of lotsizing problems depends on the features taken into account by the model which can be inferred from practical need: planning horizon, type of demand, number of facilities and number of levels of the production system, number of items, capacity or resource constraints, setup structure, supply process, service policy and consideration of time consuming activities. Regarding the objective function, besides setup and holding costs, some variants include production and capacity related costs. Further possible objectives are the maximization of service level or smoothing the production load. An extended presentation of these characteristics can be found in Haase [1994], Rizk and Martel [2001] and Almada-Lobo [2005]. The set of models ranges from the continuous time scale, constant demand and infinite time horizon lotsizing problems —with the well known economic order quantity model (EOQ) and the economic lot scheduling problem (ELSP)—, to the discrete time scale, dynamic demand and finite time horizon lotsizing models.

Regarding production scheduling, it focuses on the allocation of finite resources to execute tasks at different time points. Scheduling (also known as programming) implies assigning and sequencing jobs (start and finish times of jobs are determined).

Research community tended to be divided between researchers investigating lotsizing problems and researchers interested in scheduling problems. Pure lotsizing models only determine the lot size per period, and are not concerned with the sequence of lots in a period. Furthermore, scheduling decisions are taken afterwards for each period separately. However, the high degree of interrelation between lotsizing and scheduling decisions enhances the importance of an integrated decision making. In many practical situations, potential inconsistency between medium- and short-range planning may cause infeasibility. Recently, due to good opportunities of research on the interface between scheduling and inventory theory, some effort has been made to bridge the gap between the two research communities. Models to determine lot sizes have incorporated more detail. In addition to (pure) lotsizing model decisions, lotsizing and sheduling models also determine the sequence of lots and generate a schedule.

This dissertation focuses on deterministic single-level dynamic lotsizing and scheduling problems for multiple items that compete for finite machine capacity. Eppen and Martin [1987] classified these problems into either big-bucket or small bucket problems. Big-bucket models allow for the production of many items in a single period, whereas in small-bucket problems only one setup can be performed on a machine during a given period and, therefore, depending on the model at most one or two items may be produced per period. In practice, many production planning problems involve multiple production lines or machines, in the presence of sequence dependent changeovers, that further complicate lotsizing and scheduling problems considerably.

The work described in this thesis was motivated by the glass container production planning problem. Glass container is essentially a commodity industry, working under a make-to-stock policy and serving extremely dynamic markets. Due to the long-term slow growth of glass making and to the nature of this intensive capital industry, companies have focused on improving efficiencies on the use of glass plant resources and reducing costs to remain competitive. Our case study stimulates us in two directions: on one hand, its manual scheduling is to a great extent subject for improvement; on the other hand, the complexity of its manufacturing system poses sound scientific challenges. Here, it is necessary to make planning decisions at a hierarchy of levels, thus the concept of hierarchical production planning comes into play.

The need for decision support systems that rely upon analytical methods to address production planning problems is growing. In recent years, many companies have implemented enterprize resource planning systems (ERP) interfaced with other advanced systems to support production planning and scheduling tasks. However, these systems are largely ignored by academia in operations research and decision sciences (McKay and Wiers [2003]), leading to a significative gap between theory and practice regarding decision support for production control tasks. We find the integration of the scientific breakthroughs within the sponsoring organization as fundamental.

1.2 Objectives

Motivated by a case study, the goal of this dissertation is to tackle the glass container industry production planning problem with quantitative tools to support decision-making procedures.

The challenges we encounter in practice rely upon lotsizing and scheduling decisions. We intend to give new insights into current literature by addressing more realistic and practical variants of lotsizing and scheduling models, and by using different solution techniques. Some attention will be devoted to the polyhedral study of the convex hull of a problem by the addition of valid inequalities and through the development of strong formulations, to tighten linear relaxations and speed up the solution process. Since these problems are NP-hard, we propose to develop common-sense heuristics, relaxation-based heuristics, as well as to implement metaheuristics given their ability and flexibility to cope with complex industrial settings.

Additionally, the fact that we aim to conduct problem-driven operations research raises a set of additional challenges that should not be put aside from the overall objectives of this work. From a scientific point of view, there is a need: to judge whether there is an opportunity for new models, methods and/or applications; to be flexible to accommodate the curve balls of the real world (e.g. poor data, moving targets,...); and to understand publication risks, since solving an "one-off messy problem" is often not considered publishable research. From a practical point of view, one needs to judge whether there is sufficient patience and tolerance from the company for academic research undertaking, which often triggers (despite the longer horizon) a need to spin off short-term results to maintain the partnership. In the chapters to come, it is our purpose to show several achievements and rewards from this problem-driven operations research.

1.3 Thesis Synopsis

The work is divided into seven chapters. The reminder of the thesis is as follows.

Chapter 2 provides a somewhat self-contained introduction to various multi-item lotsizing and scheduling models, with discrete, time varying demand in a finite planning horizon. It is the aim of the chapter to show the reader some particularities of big-bucket and small-bucket models necessary to tackle the more complicate extensions of the following chapters. The literature review on each model aims to enhance some research opportunities addressed throughout the thesis and does not seek to be exhaustive.

Chapter 3 presents the overall features of the glass container industry and of the manufacturing process that regulate and constrain the production planning and scheduling process. We claim that the complexity of the whole production planning system cannot be globally optimized through a monolithic model. We decompose the system into a two-level hierarchical planning structure, inline with the organization's functions. Since the two levels of the hierarchical production planning system are strongly coupled, we try to guarantee consistency and coordination between them by anticipation, instruction and reaction mechanisms. The chapter ends pointing out some opportunities related to the way the production planning process is carried out in our case study, which we believe to be common ground in other companies (of other sectors, as well). These opportunities will be tackled in further chapters.

In Chapter 4, the long-term production planning level is modeled as a capacitated lot-sizing problem with sequence dependent setup times and costs. We first analyze a recently published model for this problem and prove that it is not a completely accurate formulation. A new set of constraints is added into this model to provide an exact formulation. We then present two novel linear mixed integer programming formulations for this problem, incorporating all the necessary features of setup carryovers, which are strengthened by some valid inequalities. We also present a five-step heuristic, which is effective both in finding a feasible solution (even for tightly capacitated instances) and in producing good solutions to these problems. Computational experiments are run for randomly generated instances.

Chapter 5 reports an application of the variable neighborhood search (VNS) metaheuristic devised to address the NP-hard long-term production planning problem. Since the neighborhoods used are not nested, they are not ordered by increasing sizes, but by means of a new metric developed to measure the distance between any two solutions. Neighborhood sizes decrease significantly throughout the search, thus suggesting the use of a scheme in which efficiency is placed over effectiveness in a first step, and the opposite in a second step. We test a new VNS variant, as well as other two, with a real-world problem instance from our case study. We conclude the chapter with some considerations on how the validation process of the production plans was conducted, and on the integration of an advanced planning system (that contains this algorithm) with a standard ERP.

Chapter 6 is devoted to the glass container short-term production planning and scheduling problem. We present two mathematical models for this problem: an exact formulation and a simplified one, which is an extension of the continuous setup lotsizing problem. We rely on a Lagrangian decomposition based heuristic (that reduces these models into a network-flow type problem) for generating good feasible solutions. We report computational experiments for both randomly generated instances and data from the case study.

Finally, Chapter 7 summarizes the work and suggests directions for future research.

Chapter 2

Lotsizing and scheduling models: An overview

Summary. This chapter reviews research studies on multi-item lotsizing and on multi-item lotsizing and scheduling problems with discrete, time varying demand in a finite planning horizon. The reader is referred to Brahimi et al. [2006] for the state-of-the-art of uncapacitated and capacitated single-item lotsizing problem. Continuous time lotsizing and scheduling problems, such as batching and scheduling problem (e.g. Jordan [1996] and Jordan and Drexl [1998]), are also not taken care here. We start with the basic capacitated lotsizing problem, which is a pure lotsizing model that defines the size of production lots within each time period. Here, scheduling decisions that determine the sequence of lots, are left to a lower planning level. Recently, mainly motivated by industrial settings, research community has addressed lotsizing and scheduling simultaneously, instead of hierarchically. The additional computational complexity has been reduced by the development of tighter formulations and of more efficient solution procedures. We address three lotsizing and scheduling models that allow for at most one setup per period, and a more general lotsizing and scheduling model that uses a two-level time structure.

2.1 Capacitated lotsizing problem

We first consider the problem of determining an optimal production plan for multi items with sequence independent setup costs and no setup times under single-machine capacity constraints, referred to as capacitated lotsizing problem (CLSP).

Throughout the exposition, t denotes time periods, which range from 1 to T, and i and j index products, which are labeled from 1 to N. We denote by $[M]$ the set $\{1, 2, \ldots, M\}$. CLSP data consist of:

h_i	cost of carrying one unit of stock of product i from one period to the next,
p_i	processing time of one unit of product i,
d_{it}	demand for product i at the end of period t,
C_t	capacity of the machine in period t (measured in time units),
c_i	cost incurred to set up the machine for product i.

In addition, let $M_{it} = \min\left\{\frac{C_t}{p_i}, \sum_{u=t}^{T} d_{iu}\right\}$ be an upper bound on the quantity of product i to be produced in period t.

The total cost consists of the holding cost and the machine setup cost. Stockouts are not allowed, which is common in the deterministic demand setting.

In order to capture the lotsizes and the resulting inventory we need the following decision variables:

X_{it}	quantity of product i produced in period t,
I_{it}	stock of product i at the end of period t.

In order to capture the setup cost, we introduce

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

Using this notation, CLSP can be written as follows:

$$\min \sum_i \sum_t c_i \cdot Y_{it} + \sum_i \sum_t h_i \cdot I_{it} \tag{2.1}$$

$$I_{it} = I_{i(t-1)} + X_{it} - d_{it} \qquad i \in [N], t \in [T] \tag{2.2}$$

$$\sum_i p_i \cdot X_{it} \leq C_t \qquad t \in [T] \tag{2.3}$$

$$X_{it} \leq M_{it} \cdot Y_{it} \qquad i \in [N], t \in [T] \tag{2.4}$$

$$(X_{it}, I_{it}) \geq 0, Y_{it} \in \{0, 1\}. \tag{2.5}$$

The objective is to minimize the sum of setup and holding costs, expressed by (2.1), whereby only the inventory at the end of each period, obtained by constraints (2.2), is valued. Constraints (2.3) ensure that the total production in each period does not exceed the capacity. Due to constraints (2.4), production of an item can only take place if the machine is set up for that particular item. Constraints (2.5) determine the suitable domains of the variables.

Literature Review

CLSP is considered to be a large-bucket problem, because several products/setups may be produced/performed per period. Such a period typically represents a time slot of one week or of one month. CLSP can be seen as an extension of the Wagner-Whitin model (Wagner and Whitin [1958]), to take into account capacity constraints. It is well known that CLSP is a NP-hard problem. Even single-item CLSP has been shown by Bitran et al. [1982] to be NP-hard. If positive setup times are incorporated into the model, Maes et al. [1991] have shown that even finding a feasible solution is NP-complete. Hence, there are only a few attempts to solve CLSP optimally for small size instances (e.g. Barany et al. [1984]).

Standard CLSP does not schedule products within a period. In addition, it assumes that setup costs occur for each lot in a period, even if the last product to be produced in a period is the first one in the period that follows (the setup state at the beginning of a period is disregarded). For manufacturing environments with considerable setup times with respect to the length of a period and with tight capacity, which is the case in the glass container industry, this model may not provide any feasible production plans. Therefore, setup carryovers need to be incorporated into CLSP. Setup carryover occurs when a product is the last one to be produced in a period and the first one in the following period. In this case, no setup is required in the latter period. Moreover, setups are preserved over idle periods and an idle time at the end of a period may be used to perform a setup for the first product to be produced at the beginning of the next period.

Despite CLSP having been intensively studied, modeling setup carryover in CLSP has not received much attention due to model complexity and computational difficulty (Sox and Gao [1999]). The efficiency of solving CLSP depends on the structure of setups. Different studies have demonstrated that proper accounting for setup times and carryovers decreases the number of setups and also frees a significant amount of production capacity, Porkka et al. [2003]. Gopalakrishnan et al. [1995] develop a modeling framework for formulating CLSP with sequence and product independent setup times and setup carryovers. In their model, a fixed charge is incurred whenever a product is produced in a period. The authors conclude that the model complexity could be reduced by looking for alternative ways to model setup carryover. Gopalakrishnan [2000] presents a modified framework for modeling setup carryover in CLSP, incorporating sequence independent and product dependent setup times and costs. Nevertheless, setup carryover is modeled in a similar way as in Gopalakrishnan et al. [1995] and decision variables have the same interpretation. Sox and Gao [1999] present a more efficient mixed integer linear programming model to CLSP with sequence independent setup costs and no setup times. Kang et al. [1999] propose a different approach, consisting of dividing the entire schedule into smaller segments, called split-sequences. A drawback of this model is that the number of split-sequences L_t in period t is a parameter. Therefore, multiple runs with different values for L_t are needed for optimization. Moreover, setup times are not considered in this model. Porkka et al. [2003] modify the mixed integer linear programming-based setup carryover model of Sox and Gao [1999] by using sequence independent setup times. To avoid the complexity of separate setup costs and times, explicit setup costs are excluded from the model. Suerie and Stadtler [2003] present a new model formulation for CLSP with sequence independent setup costs and times. The authors start with the standard CLSP model, use the standard facility location reformulation, and introduce new sets of variables and constraints to model the setup carryover. Gopalakrishnan et al. [2001] also address CLSP with setup carryover and present two effective tabu-search heuristics. Jans and Degraeve [2007c] present a MIP Dantzig-Wolfe reformulation for CLSP with sequence independent setup costs and times. In this new formulation, the integer setup and the continuous production quantity decisions are separated. The authors also describe an implementation of a branch-and-price algorithm.

All the aforementioned manuscripts address CLSP with constant sequence independent setup times and/or setup costs, with a setup carryover. In fact, these models do not consider lot sequencing within a period. They focus only on determining the products produced last and first in two consecutive periods, and also the configuration of the machine at the end of the period. Due to the "correct" calculation of the setup costs by linking lots of adjacent periods, these models are usually referred to as CLSP with linked lot-sizes (Haase [1994]).

Notwithstanding, there are several examples of process industries in which sequence-dependent setup times and costs are considerable, such as glass container and some chemical industries. Clark and Clark [2000] model CLSP with sequence-dependent setup times using a new mixed integer programming formulation. They assume that a given number of setups occur in a time period between any two given products, independently of their demand patterns. Haase and Kimms [2000] propose a model for CLSP with sequence dependent setup times and costs in which efficient product sequences are pre-determined. Therefore, it has to be decided which sequences will be used in each period. The authors also assume that the inventory of an item must be null at the beginning of a period to allow its production in that period. Gupta and Magnusson [2005] extend the framework proposed by Gopalakrishnan [2000] to incorporate sequence dependent setup times and setup costs. The setup carryover is modeled in the same way as in Gopalakrishnan et al. [1995] and Gopalakrishnan [2000].

2.2 Small-time bucket models

CLSP partitions the planning horizon into a small number of lengthy time periods, allowing for the set up of several products within the same bucket. Now consider the case where 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. This is the case of discrete lotsizing and scheduling problem, continuous setup lotsizing problem and proportional setup lotsizing problem to be tackled in the following sections.

2.2.1 Discrete lotsizing and scheduling problem

In discrete lotsizing and scheduling problem (DLSP), demand for each item is dynamic and backlogging is not allowed. Production serves to meet present or future demand. At most one item can be produced per period. In each period, the machine either produces at full capacity or is idle. This assumption is called "all-or-nothing" production. It is clear that this assumption easies the solution procedures, nevertheless there are many practical situations where this assumption is reasonable (e.g. when production quantities are integer multiples of some minimum batch size). In order to capture the start up costs, a new variable Z_{it} is introduced. Z_{it} equals one if the machine is set up for an item for which it was not set up in the previous period, otherwise zero. DLSP formulation takes the form:

$$\min \sum_i \sum_t c_{it} \cdot Z_{it} + \sum_i \sum_t h_i \cdot I_{it} \tag{2.6}$$

$$\begin{aligned}
I_{it} =& I_{i(t-1)} + X_{it} - d_{it} && i \in [N], t \in [T] && (2.7)\\
\sum_i Y_{it} \leq& 1 && i \in [N], t \in [T] && (2.8)\\
p_i \cdot X_{it} =& C_t \cdot Y_{it} && i \in [N], t \in [T] && (2.9)\\
Z_{it} \geq& Y_{it} - Y_{i(t-1)} && i \in [N], t \in [T] \setminus \{1\} && (2.10)
\end{aligned}$$

$$(X_{it}, I_{it}, Z_{it}) \geq 0, Y_{it} \in \{0, 1\}. \tag{2.11}$$

We remark that traditionally DLSP does not consider time varying capacity, i.e. $C_1 = C_2 = \ldots = C_T$. The objective function is expressed by (2.6) and, as in CLSP, penalizes holding and setup costs. As opposed to CLSP, DLSP only charges a setup cost in the period which the production batch starts (if the same product i is produced in periods $t-1$ and t, i.e. $Y_{i(t-1)} = Y_{it} = 1$, then there is no setup cost incurred for production in period t). The discrete production policy is enforced by constraints (2.9). These constraints also 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). Constraints (2.8) ensure that at most one item is produced in a period.

Literature Review

Lasdon and Terjung [1971] are among the first who contributed to the research on DLSP, describing a column generation procedure for DLSP with zero setup times and sequence independent setup costs arising in a production scheduling system for a tire company. Magnanti and Vachani [1990] describe a solution procedure based on polyhedral methods for DLSP with zero setup times. Fleischmann [1990] proposes a generic model for DLSP with zero setup times and presents a branch-and-bound algorithm using Lagrangian relaxation on the machine capacity constraints to determine lower bounds, whereas feasible solutions are derived by successive approximation techniques. The complexity of a number of special variants of DLSP (such as start-up times and multiple machines) is addressed in Salomon et al. [1991]. The authors show that determining the feasibilities of both one-machine problems with nonzero setup times and parallel-machine problems with zero setup times are NP-complete. In addition, the single-machine optimization problem with nonzero setup costs is proved to be NP-hard. Complexity results for other DLSP variants can be found in Bruggemann and Jahnke [1997], Webster [1999] and Bruggemann and Jahnke [2000]. Bruggemann and Jahnke [2000] develop a simulated annealing heuristic for DLSP with batch availabiliy, where items only become available after the whole batch is produced. Cattrysse et al. [1993] extend the work of Fleischmann [1990] by incorporating setup times for DLSP. The authors describe a heuristic based on dual ascent and column generation techiques in which the master problem is formulated as a Set Partitioning Problem. Fleischmann [1994] considers DLSP with sequence dependent setup costs and zero setup times and transforms into a traveling salesman problem with time windows (TSPTW). A Lagrangian relaxation of TSPTW into a shortest path problem with time windows is developed to compute tight lower bounds. Salomon et al. [1997] extend Fleischmann's reformulation of DLSP as TSPTW with sequence dependent setup times and solve it to optimality using a dynamic programming algorithm. The authors show empirically that the performance of this approach is sensitive to problem dimension, inventory holding costs, setup times and production capacity utilization.

Dematta and Guignard [1994] study the multi-machine DLSP without setup times (no production losses) arising at a tile manufacturing company. They develop an efficient procedure to obtain strong lower bounds by solving a Lagrangian relaxation problem, and upper bounds by utilizing a cost minimizing forward scheduling algorithm. Miller and Wolsey [2003] develop tight formulations for different variants of DLSP, such as DLSP with backlogging and safety stocks. Jans and Degraeve [2004] propose an extension of DLSP to tackle an industrial production planning problem at a tire manufacturer, accounting for start-up times (which can be a fraction of a time period, instead of an integer multiple of the time bucket), multiple machines and backlogging. The authors decompose the problem into a master program and a subproblem for every tire type, and describe a column-generation-based heuristic.

2.2.2 Continuous setup lotsizing problem

Continuous setup lotsizing problem (CSLP) is closely related to DLSP. CSLP relaxes the discrete production policy, as here lotsizes are continuous quantities up to capacity. CSLP can be formulated as follows:

$$\min \sum_i \sum_t c_{it} \cdot Z_{it} + \sum_i \sum_t h_i \cdot I_{it} \tag{2.12}$$

$$I_{it} = I_{i(t-1)} + X_{it} - d_{it} \qquad i \in [N], t \in [T] \tag{2.13}$$

$$\sum_i Y_{it} \leq 1 \qquad i \in [N], t \in [T] \tag{2.14}$$

$$p_i \cdot X_{it} \leq C_t \cdot Y_{it} \qquad i \in [N], t \in [T] \tag{2.15}$$

$$Z_{it} \geq Y_{it} - Y_{i(t-1)} \qquad i \in [N], t \in [T] \setminus \{1\} \tag{2.16}$$

$$(X_{it}, I_{it}, Z_{it}) \geq 0, Y_{it} \in \{0,1\}. \tag{2.17}$$

Constraints (2.15) allow the system to produce under its full capacity. The relaxation of the “all-or-nothing” assumption of DLSP enables the preservation of the setup state during idle periods.

Literature Review

Karmarkar and Schrage [1985] present the single-machine version of this problem without setup costs and labeled it the *production cycling problem*. This paper uses a Lagrangian capacity constraint relaxation approach to decouple the problem that provides lower bounds used in a branch and bound algorithm. Karmarkar et al. [1987] study the single item version of CSLP, for both uncapacitated and capacitated cases. Wolsey [1989] derives a family of valid inequalities for the uncapacitated single item CSLP and tightens multi-item CSLP presented by Karmarkar and Schrage. Sandbothe [1996] tackles single-machine multi-item CSLP with sequence independent setup costs and no setup times with a three-step heuristic. Hindi [1995] develops a tabu-search procedure to single-item CSLP with startup costs. Wolsey [1997] surveys some work that can be used to strengthen the formulations of single-machine multi-item CSLP with both sequence independent and sequence dependent changeovers. Vanderbeck [1998] solves single-machine multi-item CSLP with sequence independent setups using an integer programming column generation algorithm and develops a dynamic programming procedure for the single-item subproblem. Constantino [2000] derives valid inequalities for single-machine multi-item CSLP with sequence independent setups and implements a branch and cut algorithm. Dastidar and Nagi [2005] discuss the parallel machine CSLP with sequence dependent setups in presence of multiple capacitated resource constraints that appears in injection molding facilities. A two-phase workcenter-based decomposition scheme is proposed, which helps to decompose large dimension problems into a sequence of subproblems (each one addresses a group of workcenters), solved as MIP. Dematta and Guignard [1995] study the performance of rolling production in a pharmaceutical company. This problem is modeled as multi-machine CSLP with sequence dependent setup costs and no setup times. The authors dualize the demand constraints and implement a langrangean heuristic. Marinelli et al. [2007] formulate as an hybrid CLSP-CSLP the parallel-machine lotsizing and scheduling problem with shared buffers arising in a packaging company producing yoghurt. The authors decompose a relaxed version of the overall problem in a lotsizing problem on tanks (buffers, that store the product mixture) and in a scheduling problem on production lines (machines), only possible due to the non-consideration of setup times (for both tanks and machines) and setup costs for machines.

We emphasize that the majority of the aforementioned manuscripts addresses single-machine CSLP.

2.2.3 Proportional setup lotsizing problem

By relaxing the "all-or-nothing" assumption of DLSP, CSLP wastes capacity in case a period capacity is not fully used. We now introduce proportional lotsizing and scheduling problem (PLSP) as an attempt to avoid this drawback, by scheduling a second item in a period to use its remaining capacity. As other small-time bucket models, at most one setup may occur within a period. However, contrarily to DLSP and CSLP in which setups are performed at the beginning of a period, here the setup may take place at any point in time. Consequently, at most two items for which a setup state exists may be produced per period. The name of the model is motivated by the fact that machine capacity is potentially split for the production of two items proportional to the quantities needed (Haase [1994]). We now present PLSP formulation:

$$\min \sum_i \sum_t c_{it} \cdot Z_{it} + \sum_i \sum_t h_i \cdot I_{it} \tag{2.18}$$

$$I_{it} = I_{i(t-1)} + X_{it} - d_{it} \qquad i \in [N], t \in [T] \tag{2.19}$$

$$\sum_i Y_{it} \leq 1 \qquad i \in [N], t \in [T] \tag{2.20}$$

$$p_i \cdot X_{it} \leq C_t \cdot \left(Y_{i(t-1)} + Y_{it}\right) \qquad i \in [N], t \in [T] \tag{2.21}$$

$$\sum_i p_i \cdot X_{it} \leq C_t \qquad t \in [T] \tag{2.22}$$

$$Z_{it} \geq Y_{it} - Y_{i(t-1)} \qquad i \in [N], t \in [T] \setminus \{1\} \tag{2.23}$$

$$(X_{it}, I_{it}, Z_{it}) \geq 0, Y_{it} \in \{0, 1\}. \tag{2.24}$$

In case two items are produced in a period, the first item of period t corresponds to the second item produced in period $t-1$. Therefore, contrarily to DLSP and CSLP, variables Y_{it} state whether the machine is set up for item i at the end of period t or not. We note that it is impossible for four items to be produced within two time periods. Constraints (2.21) couple the production decisions with the setup state of the machine, allowing the production of item i in period t if the machine is set up for item i either at the beginning or at the end of period t. Constraints (2.22) ensure that the total production in each period does not exceed the capacity. As in CSLP, observe that the setup state is preserved over idle periods.

Literature Review

Drexl and Haase [1995] present a backward-oriented regret-based solution methodology and outline various types of generalizations of the basic PLSP, such as PLSP with sequence independent setup times and multi-machine PLSP. Belvaux and Wolsey [2001] present a tighter model for PLSP basis model. Suerie [2005] derives a new MILP for campaign planning problems (which often arises in the processes industries) based on PLSP and presents some valid inequalities. Suerie [2006] proposes two formulations to model PLSP with period overlapping setup times, allowing for the representation of "long" setup times with respect to the length of a planning period. Contrarily to other lotsizing and scheduling models, the majority of PLSP literature addresses the multi-level variant. Kimms [1996] presents two heuristic approaches for multi-level, single-machine PLSP. The author represents the solution by a graph structure on which a tabu search is performed and the production plan is generated by a specific construction scheme. Kimms and Drexl [1998b] provide some insights into multi-level, multi-machine PLSP, namely a set of valid inequalities. Kimms and Drexl [1998a] also tackle the same problem, providing MILP for several extensions which differ in the allocation of resources. A generic solution method derived from that of Drexl and Haase [1995] is presented. Kimms [1999] presents a genetic algorithm for multi-level multi-machine PLSP. Instead of representing a solution with the general bit strings, the chromosome is a two-dimensional matrix and each entry represents a rule for selecting the set up state for a machine at the end of the period.

2.2.4 Comparison of the models

In order to highlight the differences between the three small-bucket models, we present in Appendix A the optimal solutions of DLSP, CSLP and PLSP for the same instance.

By memorizing the setup state during idle periods, CSLP and PLSP only charge a setup cost between two batches of the same item if another item has been produced meanwhile.

PLSP can be seen as a generalization of CSLP and DLSP since they can be derived from the former by adding constraints (2.15) and (2.9), respectively. Having these equations introduced, other constraints become redundant and can be dropped (see Kimms and Drexl [1998b]). Clearly, the set of feasible solutions of DLSP is a subset of the set of feasible solutions of CSLP, which in turn constitutes a subset of the set of feasible solutions of PLSP. Consequently, for the same data set, $\nu(DSLP) \geq \nu(CSLP) \geq \nu(PLSP)$. An advantage of DLSP over CSLP is that DLSP algorithms are able to solve rather large problems in reasonable time as compared to CSLP algorithms. The computational advantages of DLSP over CSLP explains why the latter has attracted only little research interest. By allowing two items per period, the length of each period in PLSP can be increased, and thus decreasing the number of periods and the size of the model. On the other hand, it is more difficult to find a tight formulation.

2.3 General lotsizing and scheduling problem

A criticism to small-bucket models is that for real-world instances they require a prohibitive number of periods, specially if mathematical programming approaches are to be implemented. This fact motivated the research community to take into account big-bucket models that deal with lotsizing and scheduling simultaneously (see Section 4.2). In this section, a more flexible lotsizing and scheduling formulation is introduced: general lotsizing and scheduling problem (GLSP), first proposed by Fleischmann and Meyr [1997].

As opposed to previous models, GLSP makes use of a two-level time structure. The planning horizon is divided into large buckets (also denoted as macro-periods), with a given length. Each macro-period is divided into a fixed number of non-overlapping micro-periods with variable length.

Let S_t denote the set of micro-periods that belong to macro-period t, $|S_t|$ the number of micro-periods of macro-period t and $f_t(l_t)$ the first(last) micro-period of macro-period t. Clearly,

$$f_t = 1 + \sum_{\tau=1}^{t-1} |S_\tau| \text{ and } l_t = f_t + |S_t| - 1.$$

A sequence of consecutive micro-periods defines a lot and, therefore, a lot may continue over several micro- and macro-periods. The length of a micro-period is a decision variable. However, to allow MILP-modeling, $|S_t|$ is fixed and represents the maximum number of lots that can be scheduled in (macro-)period t.

GLSP can be formulated as follows:

$$\min \sum_i \sum_s c_{is} \cdot Z_{is} + \sum_i \sum_t h_i \cdot I_{it} \tag{2.25}$$

$$I_{it} = I_{i(t-1)} + \sum_{s \in S_t} X_{is} - d_{it} \qquad i \in [N], t \in [T] \tag{2.26}$$

$$\sum_i Y_{is} \leq 1 \qquad i \in [N], t \in [T], s \in S_t \tag{2.27}$$

$$p_i \cdot X_{is} \leq C_t \cdot Y_{is} \qquad i \in [N], t \in [T], s \in S_t \tag{2.28}$$

$$\sum_i \sum_{s \in S_t} p_i \cdot X_{is} \leq C_t \qquad t \in [T] \tag{2.29}$$

$$Z_{is} \geq Y_{is} - Y_{i(s-1)} \qquad i \in [N], t \in [T], s \in S_t \tag{2.30}$$

$$(X_{it}, I_{it}, Z_{is}) \geq 0, Y_{is} \in \{0, 1\}. \tag{2.31}$$

Observe that demand and holding costs (external data) refer to macro-periods, while controllable decisions (setups and productions) link to micro-periods. Note, by constraints (2.28), that the length of micro-period s is determined by the capacity consumption of the respective macro-period (proportional to the quantity produced in that micro-period). Thus, idle micro-periods are allowed. Moreover, the objective function does not distinguish how production is distributed within a sequence of micro-periods assigned to the same item. Fleischmann and Meyr [1997] introduce two additional sets of constraints forcing idle micro-periods to be placed at the end of macro-periods. After an idle micro-period, the machine keeps the setup of the last item produced. The formulation of the variant GLSP with loss of setup state is straightforward.

The fact that other models (such as CLSP and 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. For instance, if $|S_t| = 1$ then GLSP equals CSLP, where macro-periods become constant length small time buckets. In case sequence dependent setups are present and the triangle inequality holds, the models developed in Section 4.2 for a variant of CLSP and the extension of GLSP to account for setups have identical optimal solutions. In case triangular inequality does not hold, GLSP enables an item to be produced several times in the same macro-period. The conversions of GLSP into other lotsizing and scheduling models is straightforward. Modeling concerns have dictated the development of those models and therefore, in general, their additional constraints enable more efficient solution procedures.

Literature Review

GLSP with non-zero minimum lot sizes is NP-hard, and even the problem of finding a feasible solution is NP-complete. Three heuristics based on the threshold accepting local search algorithm are developed in Fleischmann and Meyr [1997] for the single-machine GLSP when setup state is preserved after idle periods. This variant does not account for setup times, but considers sequence dependent setup costs. The algorithm fixes the setup pattern first and determines the production quantities next. Meyr [2000] extends GLSP to deal with sequence dependent setup times. Two solution procedures based on threshold accepting and simulated annealing local search heuristics are presented. The setup pattern is fixed applying local search, and the resulting minimum cost flow problem is solved by using dual network flow reoptimization. Meyr [2002] adapts this solution procedure to multi-machine GLSP. After fixing the setup pattern, the remaining problem is formulated as a generalized network flow problem tackled by dual reoptimization. Koclar and Sural [2005] propose a simple modification on the minimum batch size constraints for the last micro-period (l_t) of each macro-period t presented in Fleischmann and Meyr [1997]. In case the remaining capacity of a macro-period it is not enough to reach the minimum batch size, this new model allows for a solution where production starts in that macro-period and extends over to the following one (the model of Fleischmann and Meyr [1997] would force the production to start in the second macro-period). Araujo et al. [2007] tackle the lotsizing and scheduling problem from a small size foundry with a model closely related to GLSP with sequence dependent setup times and costs. An alloy is processed from scrap metal and other components in a furnace, feeding a single machine that molds products. In contrast to GLSP where a setup relates to only one product, a setup here is associated with a specific alloy. Hence, products sharing the same alloy do not incur extra setup, known in the literature as joint-setups. The authors develop a rolling horizon model and associated relax-and-fix procedure, that benefits from three local search methods: descent heuristic, diminishing neighborhood and simulated annealing.

2.4 Final Remarks

Jans and Degraeve [2007b] give an overview of recent developments in the field of modeling deterministic single-level dynamic lotsizing problems. The authors point out interesting areas for future research, such as lotsizing on parallel machines and an increasing attention to model specific characteristics of the production process, which is valuable in solving real-life planning problems. Actually, our work is inline with these research directions.

Motivated by the long-term glass container production planning problem, in Chapter 4 we study single-machine CLSP with sequence dependent setup times and costs and setup carryovers, and its extension to the case of multi-machine is addressed in Chapter 5. The short-term production planning and scheduling motivates us to tackle an extension of CSLP to multi-machine with sequence dependent setups and production loss costs. We develop a random instance generator that enables us to test settings where production upper bounds equal production lower bounds, thus closely related to DLSP. Despite relying upon a hierarchical system for glass container planning, a GLSP monolithic model is discussed in Appendix D.

Chapter 3

Glass Container Industry

Summary. Inspired by a case study, we deal with the production planning and scheduling problem of the glass container industry. We first present the overall features of the glass container production environment and analyze its main constraints. Two distinct approaches to production planning have been adopted in literature, namely monolithic and hierarchical approaches. We claim that we can not globally optimize the entire system since the monolithic approach would result in a very large and complex optimization problem of difficult resolution. Thus we attempt to achieve a good solution by solving the problem hierarchically. We address the design of the hierarchical system for glass container industry and propose a new two-level hierarchical structure to overcome the conceptual inadequacies of the traditional three-level approach. Consistency and coordination between the two levels are guaranteed by anticipation, instruction and reaction mechanisms.

3.1 Glass industry overview

The glass industry is extremely diverse, both in the products made and in the manufacturing techniques employed. According to the Directive 96/61/EC EuropeanComission [2001], glass industry has six main sectors and each sector is a separate industry in its own right, each producing very different products and facing different challenges.

The glass container is the largest sector of the European Union (EU) glass industry, representing around 60% of the total glass production. This sector, that covers the production of glass packaging, i.e. bottles and jars (mass-production items), is detailed afterwards. Flat glass, fueled by demand for building and automotive glass, is the second largest sector of the glass industry in the EU (22%). Float glass is the main type of flat glass. Final products are not directly produced after the melting stage. Instead, ribbons of glass are cut into different sized panels that are stocked and afterwards cut into smaller rectangular items according to the requirements. Therefore, as far as production planning is concerned, a two-dimensional cutting stock problem arises in this sector (e.g. Al-Khayyal et al. [2001]). Among the smallest sectors of the glass industry in terms of tonnage are the continuous filament glass fibre (1.8%) and the domestic glass (3.6%). The items of the former are converted into other products or are used in different applications (e.g. reinforcement of composite materials), have a relative high value to mass ratio and are readily transported. Thus, this sector has a wide diverse customer base when compared to the container one. Furthermore, melters are smaller (see Randhawa and Rai [1995]). The products of the latter cover ovenware, drinking glasses and giftware (high value ware), and here operations range from automatic processing to hand made relatively labor intensive processing. Energy costs constitute a much lower percentage of overall costs than for the case of glass container larger melters. The molding machines can be automatic or manual. This industry sector works under a make-to-order policy (see Alvarez-Valdes et al. [2005] and He et al. [1996]). The special glass sector (5.8%) is the most diverse in terms of production processes and capacities, and encompasses products such as lighting, cathode ray tubes for televisions and laboratory and technical glassware. Finally, mineral wool sector (6.8%) covers the production of glass wool and stone wool for insulating materials.

Driven by a real industrial case, we will focus hereafter on the glass container sector, specifically on its production planning problem.

3.2 The glass container production process

3.2.1 An overview

The glass container manufacturing process includes three main sub-processes (namely the glass production, the containers manufacturing and the palletizing) and one supporting sub-process (the decoration).

The manufacturing process begins with the mixing of raw materials (mainly sand), including recycling glass (“cullet”, from both domestic and foreign sources) in the batch house. The mixture of different quantities of raw materials determines the glass color (typically amber, flint or green). The mixture is transported into the furnace where it is melted at around 1500°C. The “cullet” reduces the temperature at which the mix melts. Since the batch material takes about 24 hours to pass through the melting stage, the furnace capacity is measured in melted tonnes per day. Individual furnace capacities range from under 100 tonnes per day to those with a capacity of over 650 tonnes per day. All the activities within the factory are entirely dependent upon its output. The energy source in this process is natural gas. The glass paste is cut into gobs (according to the size of the container being manufactured) and distributed by feeders to a set of parallel independent section (IS) glass molding machines that shape the finished product at 600°C. Each molding machine has four main characteristics:

- the number of individual sections, i.e container making units assembled side by side (typically IS machines are made up of from 6 to 20 sections);
- the number of mold cavities per section, i.e. the number of gobs to be formed in parallel, ranging from one to four. For instance, in a double-gob machine two gobs are shaped at the same time within a section;
- the center distance, i.e. the distance between the molds in a double-gob, triple-gob or quadruple-gob machine (either 41/4 inches, 5in, 51/2in or 61/4in);
- the type of manufacturing process. There are two main processes: the “blow and blow” (BB) technique, in which compressed air is used in a pre-mold to give the container its

general form, before being, in the final mold, blown to its final shape, and the "press and blow" (PS) technique, that differs from BB in the first stage, where the initial shape is obtained with a press by a metal plunger before being subsequently blown to its final profile (see Figure 3.2). There are some variants of these standard techniques. For example, the well known "narrow-neck press-and-blow" (NNPB) is a variant of the PS process for the production of lightweight glass containers.

Figure 3.1 shows the IS machine, which is the accepted industry standard worldwide for glass container production.

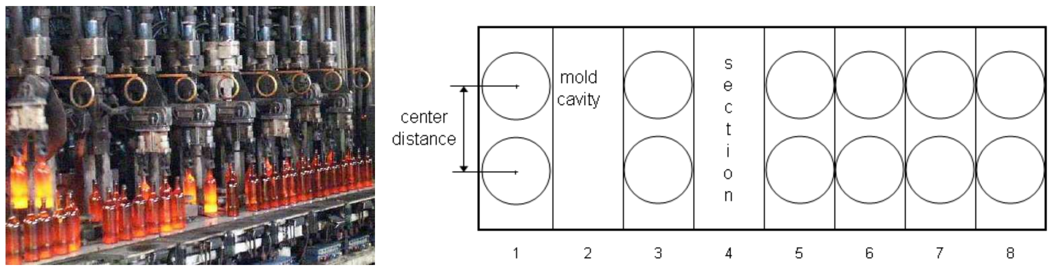


Figure 3.1: Eight section double-gob IS machine

The formed containers are then passed through a reheating/annealing kiln ("lehr") used to cool the glass evenly, in order to improve its strength. With a surface finishing process, the containers are given some additional protection from scratches and their resistance to break is also improved. A conveyor belt then moves the containers through a strict automatic inspection. Containers found to be defective are discarded and melted down in the furnace as "cullet". Once they have been quality approved, the containers are packed on pallets (the final product to be warehoused or shipped) at the end of the production lines. The term "good tonne" refers to an output of containers of a satisfactory quality weighing one tonne. Figure 3.3 illustrates a typical production facility layout.

The value added decoration sub-process (such as labeling) doesn't fit the core business of the glass container companies and usually takes place in dedicated facilities adjacent to the glass container production lines. Hence, hereafter we will ignore it.

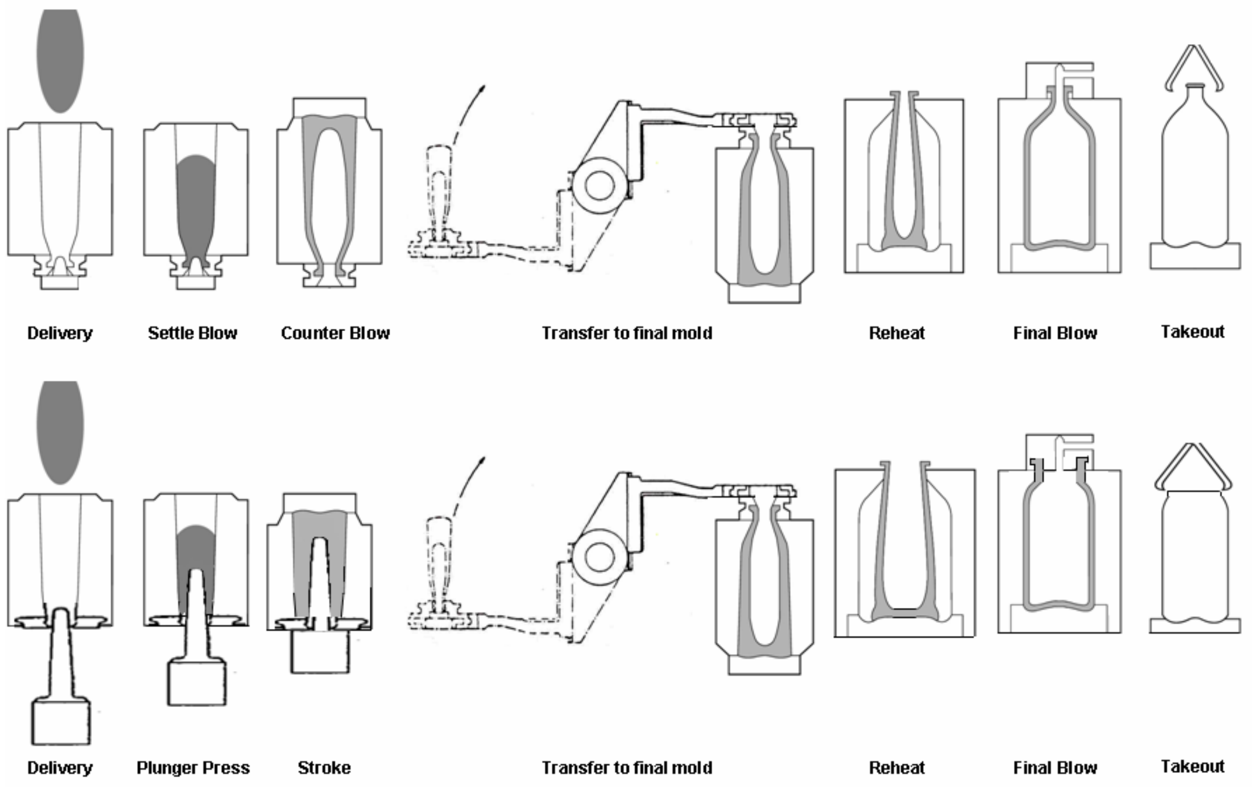


Figure 3.2: Blow and blow (above) and press and blow (below) container forming

3.2.2 The main constraints

Glass container industry is essentially a commodity industry, working under a make-to-stock policy (MTS), serving extremely dynamic markets. Sales of glass containers have two main characteristics: a high seasonality and a high variability (Paul [1979]). Since production capacity remains almost constant, the high seasonality leads to occasional production incapacities to face demand. Figure 3.4 stresses the imbalance between demand and production capacity of our case study for the year 2007. Capacity remains fairly constant since neither furnace repairs nor new furnaces and molding machines were planned for 2007.

The level of stock of containers held by this industry is of significance in considering the economics of manufacture, and fluctuates with changes in demand. In order to meet peak requirements, stocks are increased when demand is low. Figure 3.5 plots the inventory levels

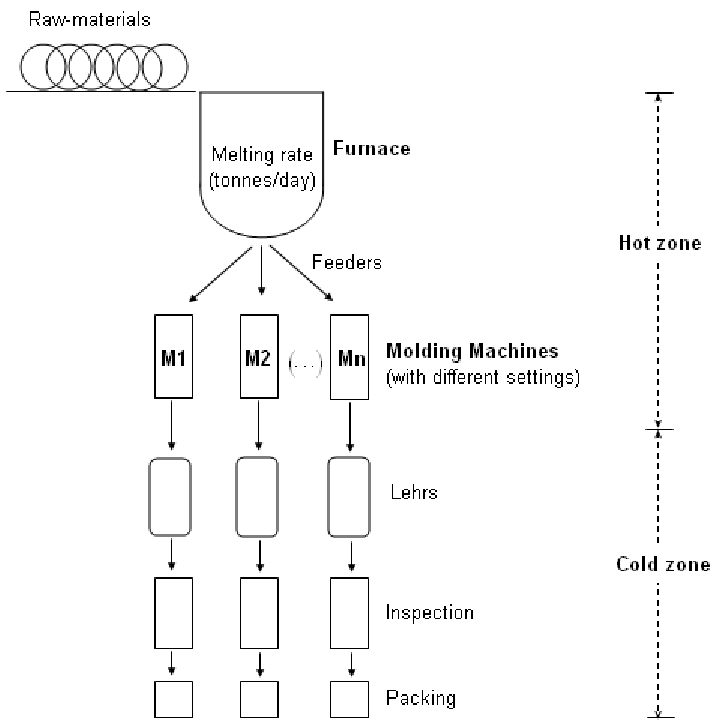


Figure 3.3: Glass container production process layout

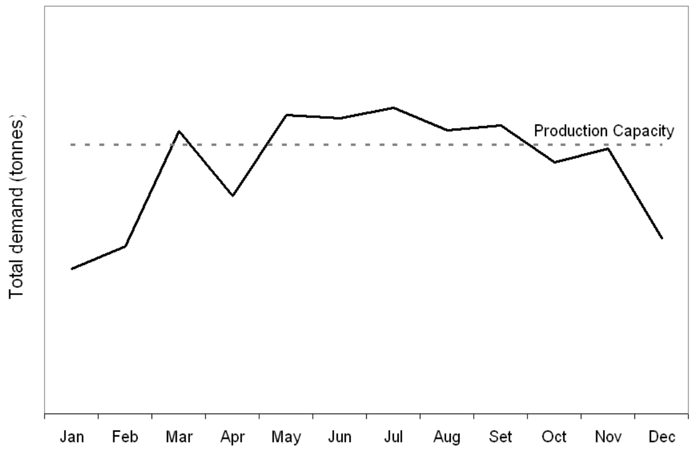


Figure 3.4: Imbalance between demand and production capacity for 2007

of our case study from 2002 to 2006, measured as the number of months of demand that are covered from stock. Figures between two to three months are standard in this industry.

The high demand variability for any container is caused by the nature of the final end

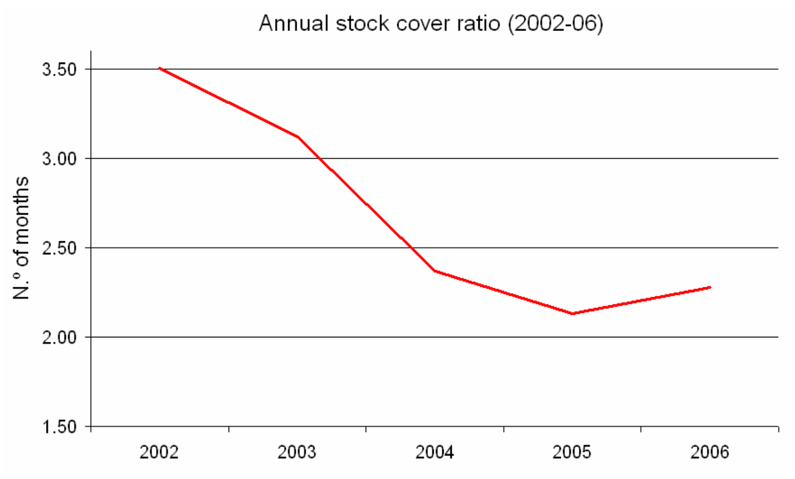
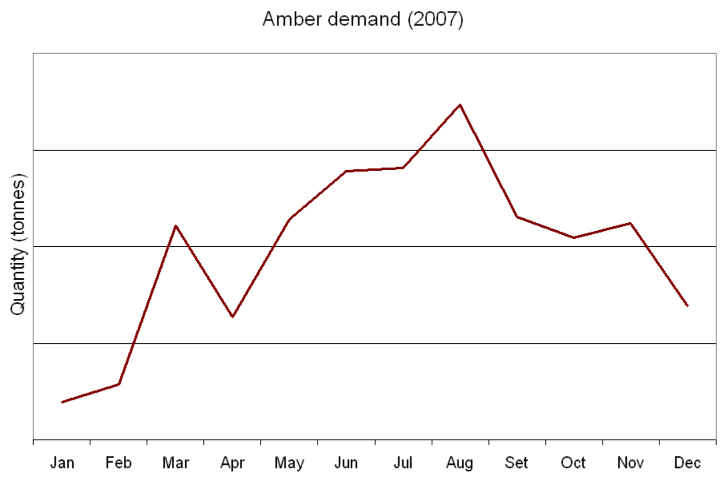


Figure 3.5: Glass container inventory levels

product with the further complication of being an intermediary process. Glass containers are sold to customers who then pack or bottle their own products in them. The volatility of demand for any particular brand is projected to the use of glass containers. Additionally, seasonal variations in demand of these products lead to corresponding variations in demand for many types of glass containers. Figure 3.6 presents the monthly demand of amber and ultraviolet green glass colors for 2007. The amber glass color is mainly used for bottling beer, whilst ultraviolet green for wines. Wines are typically bottled between March and May. Beer consumption peaks in the Summer, but demand for beer also increases before Christmas and Easter. We note that most of demand quantities are based on monthly demand forecasts for each product. Nevertheless, larger customers tend to indicate more often their requirements.

Usually, a glass container company has several plants (spread over one or more countries). Since (empty) containers are low value, heavy and large in volume, they are typically manufactured closer to the end use due to transportation costs. Therefore, the location of the production sites constraints the glass container potential market. Imports and exports tend to be fairly limited within EU since glass containers are produced in almost all Member States, and are typically driven by a policy of using the surplus installed capacity. The plants distinguish themselves by the technologies employed and by the number and capacity of their furnaces. The number of furnaces and the color campaign schedules strangulate the production flexibility. Only one color of glass can be produced at any time in each furnace



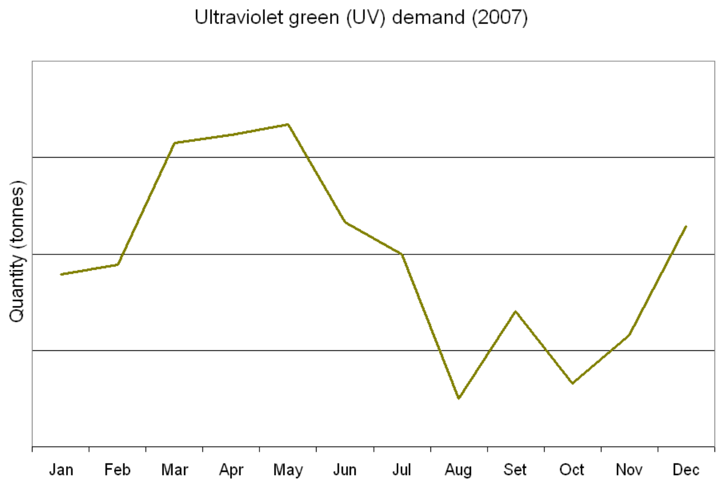


Figure 3.6: Amber (above) and Ultraviolet (below) demand in 2007

and machines served by the same furnace produce only one color of glass at a time. In addition, there are high sequence dependent setup times involved in a color changeover (e.g. the color changeover from cobalt blue to emerald green takes about 120 hours), clearly inducing color long runs and, therefore, furnace color's specialization. Consequently, MTS policy is not only driven by demand seasonality, but also by the need to meet customer's needs until the next production run of the container in order to minimize lost production capacity due to changeovers. The furnace color's specialization can be, sometimes, relaxed due to commercial demands. Nevertheless, color is likely to remain constant in the short-term.

Furnaces are operated continuously (except when they are being repaired) and machine lines operate on a 24 hour, seven days a week basis. New furnaces and forming machines cost millions of euros and require at least 18 months of planning. Therefore, there is little freedom for varying output to match fluctuations in demand. Furnace output can vary to a limited extent: it is possible to run furnaces somewhat above the rated output by boosting (heating by electricity to supplement the normal natural gas heating process) and below maximum capacity, but the savings in costs are minimal. The emptying of furnaces for short periods is impracticable as the refractory material is likely to be damaged in the process. Due to economies of scale in natural gas consumption and to feeders' setup, machine idleness is not allowed. This type of machine balancing constraint forces machines fed by the same furnace to process for the same amount of time. Each machine can only run one product at a time. Furthermore, the number of mold equipments may also limit the number of machines on which a product can be allocated at the same time.

From the four characteristics of the molding machines referred previously, the number of both individual sections and gobs to be formed in parallel determine the maximum throughput of the machine (the daily output of containers), whilst the center distance and the type of manufacturing process restrict the set of products that can be assigned to a machine.

The production rate of a product on a machine cannot be pre-specified to a pair machine-product, because it is not fixed. It depends not only on the characteristics of both product and machine, but also on the products that are produced on neighbor machines (i.e. that are fed by the same furnace). One major advantage of IS machines is the possibility of independently stop sections. Moreover, some flexible machines allow different number of gobs to be formed in parallel (e.g. the same machine may run a double or triple gob). If the mix of products demands too much from the furnace output (above its daily capacity), at least one machine has to be either sectioned (stopping some machine sections) or, if possible, its number of gobs to be formed in parallel decreased and, consequently, changing the processing time of that job. Thus, we are dealing with controllable discrete processing times. The interrelation between machines of the same furnace cannot be neglected. However, notice that the processing time of each product per mold cavity of each machine is constant (known in the glass terminology as cavity rate). The production of an IS machine can be above 400 containers/minute (lightweight round beer bottles are produced at up to 750/minute on a 12 section quadruple-gob machine).

There is also a sequence dependent setup time in a product changeover on a machine (even if it is minor when compared with the major setups of a color changeover) that has to do with time lost in setting up the production of a container on a machine and the subsequent running-in time to attain optimum production of “good tonne”. The reader is referred to Almada-Lobo [2002] for a thorough analysis of sequence dependent setups in glass container industry.

The overall efficiency of the production is measured as a “pack to melt” ratio, i.e. the tonnage of containers packed for shipment (“good tonne”) as a percentage of the tonnage of glass melted in the furnace (generally varies from 85% to 94%).

Moreover, there are other industrial constraints concerning the scheduling of product changeovers, since they are undertook by teams of highly skilled workers. Therefore, depending on the plants, there might be restrictions to the maximum number of changeovers allowable per day or per week, or to the period in which a changeover might occur (for instance, only being possible on working days or morning shifts).

3.3 The production planning system

Glass making long term slow growth (total sales generally follow population growth) and the fact that we are dealing with an intensive capital mature industry, lead to a strong focus on improving efficiencies on the use of glass plant resources (a furnace runs continuously up to an average of 12 years, before it is demolished and rebuilt) and reducing costs in order to remain competitive and, subsequently, on the production planning process. In general terms, production can be defined as a process of converting raw materials into finished products (Bukh [1994]). Although the production system is multi-site, for the sake of managerial efficiency, the production planning and scheduling functions are centralized and performed by a single scheduler. The complexity of the glass container production system (incorporates almost all the traditional lotsizing and scheduling constraints in addition to those related to furnaces), the dynamic and stochastic nature of this production environment, as well as the frequent interdependencies between decisions that are made at and affect different organizational echelons (the person in charge of sequencing products on a machine is not the one that will decide the assignment of glass colors to plants) make it difficult to manage the whole production planning decision-making process with a single quantitative model. Despite the strong dependency of these decision levels requiring an integrated procedure that would minimize sub-optimization, the result of a monolithic entity would be a large scale model, difficult to assemble, optimize and interpret, neglecting established organizational structures (Vicens et al. [2001]). A classical approach to handle this multi-level decision making process is the Hierarchical Production Planning – HPP (Hax and Meal [1975]). HPP recognizes the differences and contrasts of the decision categories introduced by Anthony [1965], namely strategic planning, tactical planning and operational control –, and partitions a large scale production planning problem into smaller manageable sub-problems. HPP uses the aggregation and the desegregation of information along the several linked hierarchical levels: decisions primarily taken pose constraints on the subsequent detailed decisions that, in turn, give feed-back to evaluate the quality of higher level action. By breaking the production planning problem into subproblems results in a system suboptimization, even if one can optimize each of these subproblems. On the other hand, by fitting the organizational structure, HPP procedure is more realistic and easier to implement. The reduction of complexity (in both data and computational requirements) and the gradual absorption of random events (e.g. due to error compensation, aggregate data forecasts are more reliable than that related to detailed data) are other major advantages of HPP. The design of a HPP is determined by both the number of hierarchical levels and the aggregation level of the variables at stake. Three factors are typically identified in deciding on the distinct decision levels to use within HPP: lead time and planning horizon, similarity and decision making procedure. Three aggregation categories are distinguished: products, resources and time. Different levels of aggregation vary the amount of detail in the problem description (and, naturally, the number of variables and constraints of the model).

After a brief overview of HPP literature, we concentrate on the design of an hierarchical system to support tactical and operational decisions pertinent to glass container production planning.

Literature Review

Hax and Meal [1975] consider a multi-plant firm with four decision levels. The highest level distributes the products to individual plants, and is solved once. Three further decision levels that address the management of a single plant are a consequence of recognizing three aggregation levels for products: items (end products), families (group of items sharing similar setups) and types (group of families having similar seasonal demand patterns and production costs). Using a hierarchical approach requires that attention be paid to the linking mechanisms between the different levels. Bitran and Hax [1977] explore the interaction mechanisms among the different hierarchical levels of Hax and Meal and suggest the use of convex knapsack problems to disaggregate the product type and families run quantities into families and item run quantities. Some modifications to this algorithm that clearly improve its performance (specially for settings where the setup costs are significant) are introduced in Bitran et al. [1981]. Boskma [1982] studies the effects of aggregation over products and over time. Bitran et al. [1982] discuss extensions of HPP systems to support two-stage production processes, as the case of environments involving fabrication and assembly operations. It must be remarked that the only interaction in the hierarchical system of Hax and Meal is through the constraints that higher-echelons impose on lower-echelons. Graves [1982] introduces a feedback mechanism between two subproblems, sending back information (by means of a Lagrange multiplier formulation) which reflects the cost penalties at the lower level due to the constraints imposed by the higher level subproblems. Erschler et al. [1986] focus on the consistency of decisions in a two-level structure and present necessary and sufficient conditions for a disaggregation procedure to be consistent. Merce et al. [1997] analyze the interactions between two successive levels of a HPP based on time aggregation and present a set of analytical conditions providing the robustness and consistency of decisions. An aggregate decision is said to be robust if it can be disaggregated into a feasible detailed solution, which in turn is said to be consistent if it is compatible with all the aggregate decisions. Fontan et al. [2002] tackle a two-level scheduling structure based on time aggregation and show that the robustness of an aggregate plan can be achieved by adjusting some upper level parameters that provide autonomy to the lower-level scheduling necessary to meet requirements imposed upwards. Dauzere-Peres and Lasserre [2002] study the production planning and scheduling hierarchical system. The authors claim that in many production contexts the scheduling decision level should not be considered a “slave” of the planning decision level and propose an iterative approach that incorporates in planning models considerations on how scheduling is performed in the shop-floor.

Numerous HPP applications are reported in the literature for different production environments, such as tile industry (Liberatore and Miller [1985]), shoe production (Carravilla and Sousa [1995]), agricultural machinery manufacturing (Ozdamar et al. [1998]), paper industry (Respício et al. [2002]) and steel fabrication (Neureuther et al. [2004]). McKay et al. [1995] discuss the fundamental principles and assumptions of HPP regarding the types of manufacturing situations it is suitable for. As far as glass container industry is concerned, most of the publications encountered in literature tackles the production planning at a detailed level (short-term) and gives little insight into how the respective HPP is designed. Nevertheless, from the formulation of the short-term problem, we are able to infer upon the structure of the respective HPPs.

Richard [1997] and Chevalier et al. [1996] present similar planning systems for glass container industry with the three classical levels: strategic, tactical and operational, making use of the three aggregation categories: products (group technology), resources and time. At a strategic level, monthly demand is previously aggregated by color of glass and then assigned to furnaces for an entire seasonal cycle (one year), with the objective of minimizing the sum of raw material, melting process, transportation and holding costs. This problem is solved through a simple transportation problem algorithm. At an intermediate level, products are scheduled on machines in a 12-month rolling horizon within each color campaign, and product lot sizes are determined. The presence of production rates turn the problem into a generalized transportation problem, which is solved heuristically based on economic quantities. At a lower level, products are scheduled on machines on a daily basis for a short-term period: 3 to 6 months. At this point there are so many constraints that product scheduling is made manually. The main goal is to satisfy demand with minimum stocks. We note that this hierarchical system induces a scheduling of color campaigns situated between strategic and tactical levels, since they are assigned to furnaces in the former and their length is calculated in the latter. However, this interrelationship is not fully analyzed since significant high color changeover costs would need to be considered. According to Richard [1997], the introduction of those costs would result in a non-linear problem, most unlikely to be solved until optimality.

In Daheur and Jacquet-Lagreze [1996], a production scheduling system is proposed for 3 successive months, and it consists of two stages: a “lot-sizing procedure”, in which the scheduler calculates the appropriate product quantity to be manufactured and determines a target period (regarding inventory levels), followed by a “machine-loading and scheduling” procedure, that tries to find an available machine with the right characteristics to process each lot previously defined, giving the precise date for its production. The second stage is modeled and then solved as a linear programming problem. Furnaces are hereby considered to be mono-color for a long period. The timetables for the colors of the furnaces are input data of this HPP. This architecture suggests the use of at least 3 hierarchical levels to analyze the whole production planning.

Richard and Proust [2000] also focus on short-term planning. The authors consider that the middle term production plan output fixes color campaigns, assigns a set of commercial demands to plants and computes the production load of each furnace. The objective of the short-term planning step is to assign products to machines, maximizing a financial criterion: “total production margin”. The authors argue that they deal with a lotsizing problem type, despite not including inventory levels. Moreover, this level does not cope with sequencing lots on machines, and the authors limit the analysis to one furnace. The size of an instance of “industrial problems” led the authors to propose an hierarchical approach to short-term planning. In a first stage products are aggregated in macro-products (the aggregation process is based on minimizing the changeover costs) that are assigned to machines to minimize sequence independent changeover costs. In a second stage, the assignment of products to machines aims to maximize a financial criterion, satisfying the aggregate plan output.

T’Kindt et al. [2001] also deal with the short-term production planning problem. The set of inputs of this system are the same considered in Richard and Proust [2000]. In addition to the financial criterion, another criterion based on the idle time of a machine (that tries to minimize the difference in machine workload, as a way of tackling balancing constraints) is incorporated in the model to schedule products on parallel machines. An algorithm to compute all the strict Paretto optima is offered, and is easily extended to an interactive one. In Paul [1979] jobs are scheduled and sequenced on machines using different dispatching rules, towards the minimization of both the mean tardiness of jobs and the number of stockouts. Lot dimension is calculated based on the gross requirements to meet the following two-periods of forecast demand. The input data of this problem are the color campaign timetables and the assignment of products to furnaces. The author concludes that a shortest processing time based dispatching rule provides the most efficient operating policy.

Paul [1979], Daheur and Jacquet-Lagreze [1996], Richard and Proust [2000] and T'Kindt et al. [2001] assume the same input data for the short-term planning process: color campaigns scheduled per furnace and products assigned to furnaces. Such an input is used to schedule products on machines in the short-term planning. [Richard, 1997] and Chevalier et al. [1996] also propose a HPP whereas in one level products are assigned to furnaces and, downstream, products are scheduled on machines.

New design of HPP

In our opinion, none of these HPPs reflects entirely the container glass inherent production dynamics. We believe that, with those designs, the feasibility of an upstream level plan may lead to the infeasibility of a downstream plan. The high setup times involved in color and manufacturing process changeovers make both the number of machines in each furnace and the technologies employed on each machine vital to the desired production flexibility. The adequacy of machine equipment to face demand is critical in this industry due to the balancing constraints that do not allow idle times on machines. Therefore, the color campaign scheduling decision is highly related to the assignment of products to machines. Even if the machine balancing constraint is slightly relaxed, good local solutions in both subproblems would lead to poor overall performance. Naturally, integrating both subproblems in the same level increases the difficulty to compute and optimize it.

Based on the above considerations, the production planning process will be decomposed into two broad levels: long-term planning, with a tactical nature, and short-term operational planning. The factors that determine the contents of each decision level are the following:

- planning horizon: 12 months for long-term level and 8 weeks for short-term one,
- lead time: short-term level demands a daily plan making, whilst long-term a few plans per month,
- decision making: different decision-makers involved in both levels, and
- production process environment.

Regarding the manufacturing process, the production planning is only constrained by the hot area (see Figure 3.3) of this semi-continuous manufacturing process, which includes glass production (the continuous part) and containers manufacturing (the discrete part). Usually, there is enough capacity downstream of the molding machines to process all the upstream work. Even if some problems arise at the end of the production line (packaging area), the conveyor belt has some buffer areas to temporarily stock intermediate products, avoiding the stoppage of any molding machine. The hot production process area can be considered as a single operation type (single level), the transformation of the molten glass into a finished product. This feature enables us hereafter to use the term machine to refer to the production line.

3.3.1 Long-term level

The output of the long-term planning is a 12-month rolling horizon plan that assigns colors to furnaces, schedules color campaigns and does a monthly assignment of products to machines (determining the number of production days). At this level, monthly product demand forecasts are provided. Figure 3.7 illustrates a partial plan for the first month of a multi-facility production system. For instance, product with reference P1 will be produced on machine C1 for three days within the first color campaign of the first period (January).

Clearly, this output is ambitious, in a sense that we are potentially dealing at the same level with transports routing, color scheduling, lotsizing and machine loading procedures. At this level we are not tackling yet the product sequencing problem and even the lotsizing procedure is not complete since, at a lower level, it is possible to schedule two or more

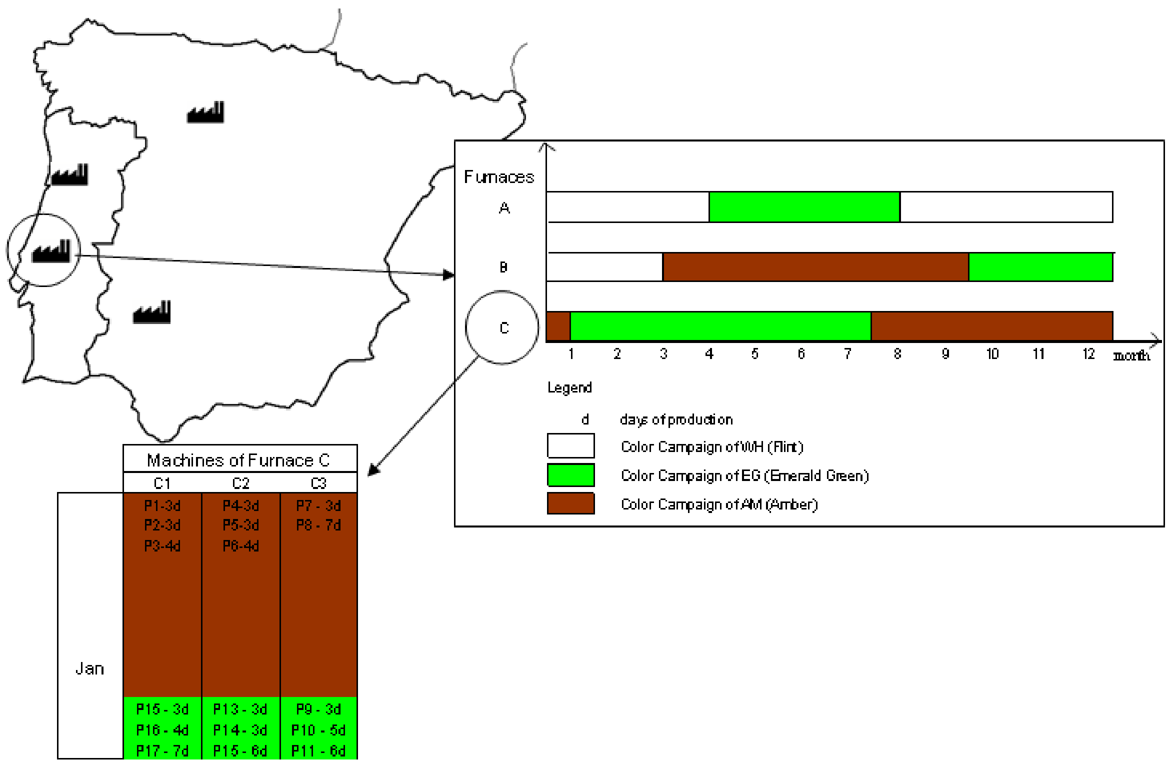


Figure 3.7: Long-term production planning output

production runs of the same product within the same color campaign (known as job splitting), or to join two lots to be produced in one run (job batching).

The specific management objectives at this level include meeting customer demands, minimizing inventory investment (above safety stocks) and transportation costs, and maximizing the utilization of production facilities. Firstly, customer satisfaction has to do with delivery commitments. When production capacity is not enough to meet gross requirements, it is necessary to manage priorities by ranking customers' importance, typically ABC categorized. Secondly, regarding transportation costs, customers are aggregated in pre-defined geographic areas. Thirdly, the minimization of the costly holding stocks can be achieved by shortening color campaigns. Finally, the maximization of throughput (furnace utilization) can be achieved through the minimization of sequence dependent setups. Considering that the only characteristic that is intrinsic to a container is its color[1], it is assumed that, at this

[1]We remark that some containers may be produced with more than one manufacturing process, thus not making it completely intrinsic to a container. The manufacturing process changeovers also require high

level, products sharing the same color have negligible setup times. Along with the gob setup (e.g. switchover from double-gob to single-gob settings), the color setup is the most difficult changeover and requires significant (major) setup time (and consequently setup cost). Since glass industry furnaces are usually dedicated to a small set of colors, it is essential to define color campaigns. Manufacturing process sub-campaigns (within each color campaign) on each machine are not defined yet at this level, however the number of manufacturing processes required to produce products on each machine within each color campaign may be easily taken into account as constraints. Machines are assumed to be producing at their fastest production rate (consequently, gob changeovers are not considered here).

A final remark about the granularity of information used in this problem: from the three aggregation categories, we use a time aggregation mechanism, aggregating micro-periods (days) in macro-periods (months). As far as resources are concerned, it is not advisable, as referred before, to aggregate machines of the same furnace due to strong machine-balancing constraints, that would generate unfeasible solutions downstream. Since we are planning finished products (the end product of the hot zone) instead of final products (decorated and palletized finished products), there is a slight product aggregation (the same finished product may be decorated or palletized in different ways, generating more than one final product). A broader product aggregation is tempting due to intrinsic and common technology features, such as color and manufacturing process families. However, a certain family could contain products to be distributed to customers from different regions, not allowing the definition of logistic costs to that family. Products could be filtered by customers' geographical clusters though. However, those sub-families could contain products to be shipped to customers ranked differently according to their importance to the company (used to prioritize production in case of scarce capacity). In addition to color, manufacturing process and geographical cluster filters, a customer ranking filter would need to be added. With these sequential filters, the total number of families would be large, not reducing as intended the problem complexity and dimension (for our case study, we would have 8 colors * 3 manufacturing processes * 10 geographical clusters * 3 customers importance levels = 720 families). Nevertheless, as we are only considering major setups here, the concept of product family is disseminated.

setups, not with the same magnitude of that of color's though.

We are not taking advantage of the fact that product families demand forecasts are more realistic and reliable than the related detailed data though.

While performing the rolling horizon approach, notice that the first two months of every iteration should be frozen. Although the long-term level solution is computed over a year, only a part of it (over the short-term horizon) is implemented. Such period is forced by the expected time that takes from the requisition to the availability of a new set of molds in the facilities (clearly, a hard constraint).

3.3.2 Short-term level

The output of this stage is a daily production plan for the following 8 weeks. This plan sequences products on processors for a time interval that is somewhat equivalent to the frozen period of the long-term planning, and gives the precise date (day) for each lot to be produced. It is impracticable (or, at least, not desirable) to change production plans for the current week, therefore it is always frozen. Figure 3.8 illustrates a partial short-term production plan. As an example, product P6 is produced on machine C2 for 4 days starting on January 1st. In this production intention, machine C2 runs a 10-section single-gob.

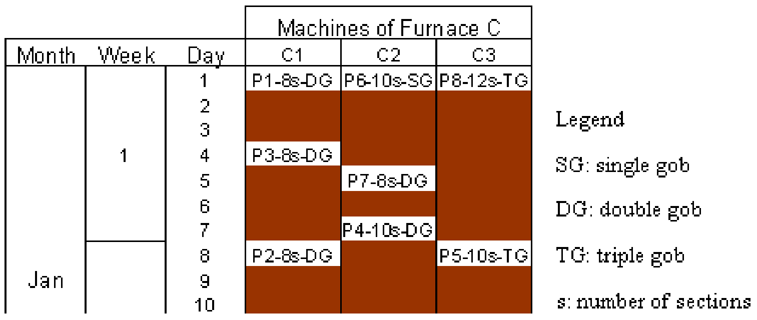


Figure 3.8: Short-term level (partial) output

The input data of this level consist of timetables for color campaigns on each furnace and monthly assignment of products to furnaces. Despite the long-term output suggesting the number of production days per month of products on machines, short-term sequencing procedure may anticipate or postpone start or finish dates of a lot to a different month and only considers the furnace where the product is produced. Therefore, the machine loading long-term output is not considered as an input of this level in order to increase the flexibility of the search for feasible plans (strangulated by machine balancing constraint). This short-term analysis is conducted furnace by furnace.

The short-term management objectives encompass the satisfaction of customers' due dates (on a weekly basis), the minimization of holding costs and the maximization of resources utilization. The last goal can be attained by maximizing the furnaces' daily melted tonnage and minimizing product (major and minor) setup times (not negligible even for products sharing the same color and manufacturing process). During a product changeover on a machine, the furnace keeps feeding the machine, however, the gobs are discarded and melted down again in the furnace. Hence, sequence dependent setups consume part of the furnace capacity. This operation is performed by specialized workers in the first shift of the day, thus the minimum time slot considered by the planner is the day (i.e. a machine can only be assigned one product per day).

At this level, production plan has to satisfy additional constraints. Due to the semi-continuous nature of this manufacturing process, the daily throughput of the furnace is determined by the daily output of its associated machines. Thus, machines of the same furnace are not independent of one another since they share (consume) the same resource. If the products mix on neighbor machines demands too much from the furnace (above its daily capacity), it may be necessary to stop some machine sections and/or change (if possible) the number of gobs to be formed in parallel. The processing time of products to be manufactured in those situations is naturally larger. On the other hand, if the mix of products in a certain day pulls too little melted glass from the furnace (this usually happens when products are lightweight), the natural gas consumption economies of scale are minimum and may lead to prohibitive industrial costs. Remember that at the long term level we have assumed that each machine is working at its full speed. The number of mold equipments may also limit the number of machines on which a product can be allocated at the same time (job splitting requires the duplication of expensive equipments). Moreover, additional commercial constraints have to be taken into account, like the need for safety stocks imposed by large customers. Products may also be prioritized per customer importance (A, B or C).

In addition to industrial and marketing constraints, the short-term planning process has also to respect management rules of different production sites. For instance, changing of a lot on a machine is only possible on working days or there is a limit number of daily or weekly changes per facility.

3.3.3 Conceptual Overview on the new HPP design

HPP structures are grouped into three main classes:

- hierarchical planning: where production quantities are determined at each decision level;
- hierarchical scheduling: tasks are scheduled on resources that are aggregated according to the decision level at stake;
- Production planning and scheduling (PPS).

PSS is definitely one of the most prominent hierarchical systems. It contains two hierarchical levels, in which the "top-level" deals with the production planning problem (more precisely with lotsizing) and the "base-level" entails the scheduling (programming) problem. Despite also having two levels, HPP hereby introduced does not entirely fit PPS framework, since the higher level already integrates planning (the number and dimension of lots) and scheduling decisions (color campaigns programming). The lower level is similar to PPS "base-level", despite also dealing with lots (it is equivalent to batching sequencing problem). Figure 3.9 summarizes the interactions of both levels, indicating on the left hand side the different criteria of levels and on the right hand side different information situations.

Production planning is usually performed on a rolling horizon basis. As mentioned before, despite the long-term level being computed over 12 months, only a part of it (over the 8-week short-term horizon) is implemented. Figure 3.9 uses the conceptual framework proposed in [Schneeweiss, 1995], distinguishing three different stages of hierarchical interdependencies: anticipation, instruction and reaction. In finding a feasible solution, the long-term level takes into account the most relevant characteristics of the short-term level, namely the machine balancing constraint and the average machine efficiency (considering average setup times for

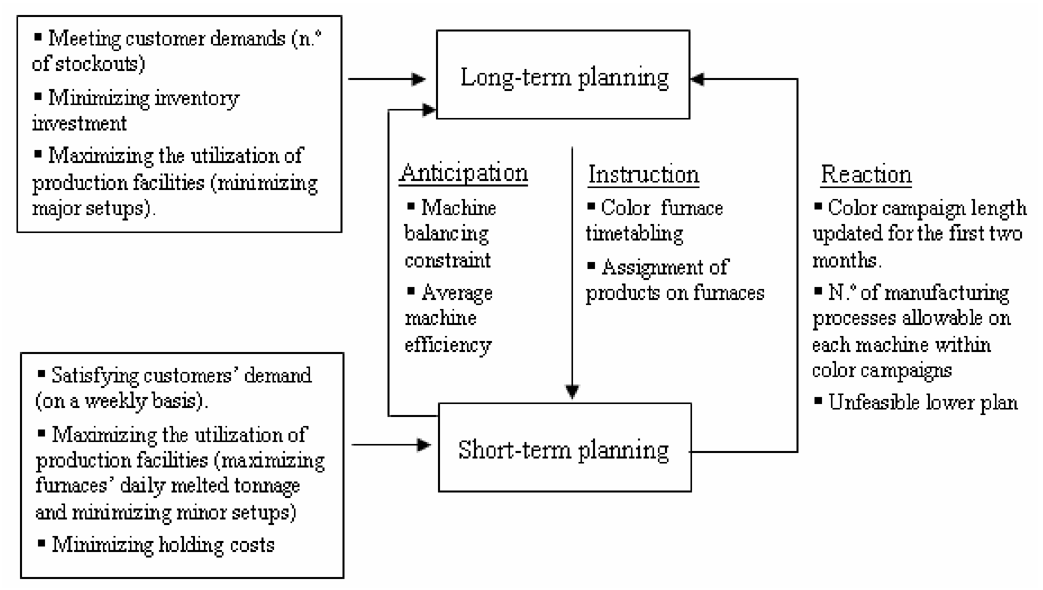

▪ Meeting customer demands (n.º of stockouts)
▪ Minimizing inventory investment
▪ Maximizing the utilization of production facilities (minimizing major setups).
Long-term planning
Anticipation
▪ Machine balancing constraint
▪ Average machine efficiency
Instruction
▪ Color furnace timetabling
▪ Assignment of products on furnaces
Reaction
▪ Color campaign length updated for the first two months.
▪ N.º of manufacturing processes allowable on each machine within color campaigns
▪ Unfeasible lower plan
▪ Satisfying customers' demand (on a weekly basis).
▪ Maximizing the utilization of production facilities (maximizing furnaces' daily melted tonnage and minimizing minor setups)
▪ Minimizing holding costs
Short-term planning

Figure 3.9: Glass container hierarchical planning structure

product changeovers). This bottom-up influence is crucial to the future feasibility of short-term production plans. The output of the long-term level will naturally influence the short-term level (top-down influence). Remember that base level analysis is conducted furnace by furnace. Finally, the base-level reacts to the top-level's instruction (also a bottom-up influence, but in this case as a feedback one). The length of the color campaign is updated at the lower level. In addition, the number of manufacturing process sub-campaigns on each color campaign may have to be limited iteratively at this stage. This information is given back to the upper level, that has to recompute its long-term plans based on the actual state of the system. If there is no feasible solution in the short-term planning, the decision maker has to manipulate the input data of the long-term level (intensifying the *feedforward* influence of the anticipation).

3.3.4 Case Study production planning process - opportunities

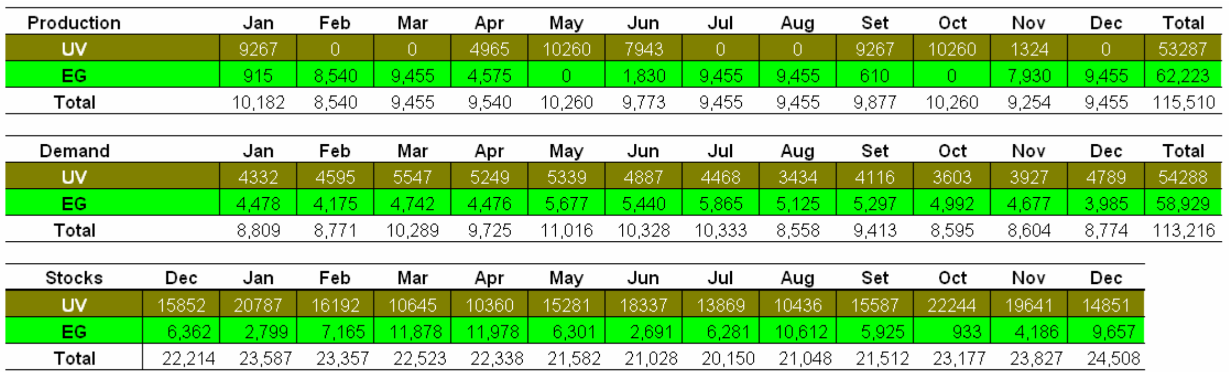
The two main stages of the HPP detailed previously are supported, in our case study, by some auxiliary ones, namely sales budgeting, revision and continuous improvement stages.

The sales budgeting is conducted every year in the beginning of October for the following year (hereafter referred as target year). It contains the commercial strategy of the company, and it is materialized in a sales budget proposal. By commercial strategy we refer to the definition of the market segments to privilege, the quantity of containers by glass color to be produced and the inventory level goal at the end of the target year (given a certain production capacity). Such decisions are established by the Executive Board (EB), with some inputs of the production planning department (PPD) regarding the estimated level of stocks in the beginning of the target year. To compute the "good tonnage" production capacity, EB also defines the goals for the overall efficiency of each plant and the capacity utilization of each furnace for the target year. The colors are then assigned to furnaces. The sales budget determines, per product and client, the expected sales in both quantity and value for the entire target year.

The sales budget is then disaggregated monthly, and it has to be validated in terms of production capacity. Given the input of EB regarding the assignment of colors to furnaces, the number of color changeovers and the planned maintenance, the first step of PPD is the color campaign scheduling. By taking into account each color initial stock and demand forecasts for the target year, the gross requirements are computed and production intentions are decided to avoid stockouts. Naturally, some months have production capacity surplus, whereas others lack capacity. Production intentions are anticipated or postponed to balance the capacity utilization. On furnaces that were assigned more than one color by EB, the production intentions have to obey to color campaigns that are scheduled iteratively at this stage. Figure 3.10 depicts stocks, demand forecasts and production intentions in 2007 for a two-color furnace –emerald green (EG) and ultraviolet (UV). The outcome of this procedure is a production plan with 5 color changeovers, illustrated in Figure 3.11.

The planner then introduces the output of this step in a transaction of SAP R/3 enterprize resource planning software (ERP), which is depicted in Figure 3.12. For instance, this report shows an EG color campaign scheduled on Furnace 5 of Avintes plant with a capacity of 370 tonnes/day, starting on January 29st and finishing on April 15th.

The following step is the assignment of products to machines. The adequacy of the machinery equipment to face demand is at stake here. The placement of production intentions is carried out in the form of Figure 3.13. On the top, the used capacity (%) of each machine of the furnace to manufacture EG containers is displayed. On the bottom, stocks, demand and



Production	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Set	Oct	Nov	Dec	Total
UV	9267	0	0	4965	10260	7943	0	0	9267	10260	1324	0	53287
EG	915	8,540	9,455	4,575	0	1,830	9,455	9,455	610	0	7,930	9,455	62,223
Total	10,182	8,540	9,455	9,540	10,260	9,773	9,455	9,455	9,877	10,260	9,254	9,455	115,510

Demand	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Set	Oct	Nov	Dec	Total
UV	4332	4595	5547	5249	5339	4887	4468	3434	4116	3603	3927	4789	54288
EG	4,478	4,175	4,742	4,476	5,677	5,440	5,865	5,125	5,297	4,992	4,677	3,985	58,929
Total	8,809	8,771	10,289	9,725	11,016	10,328	10,333	8,558	9,413	8,595	8,604	8,774	113,216

Stocks	Dec	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Set	Oct	Nov	Dec
UV	15852	20787	16192	10645	10360	15281	18337	13869	10436	15587	22244	19641	14851
EG	6,362	2,799	7,165	11,878	11,978	6,301	2,691	6,281	10,612	5,925	933	4,186	9,657
Total	22,214	23,587	23,357	22,523	22,338	21,582	21,028	20,150	21,048	21,512	23,177	23,827	24,508

Figure 3.10: Placement of furnace production intentions for 2007

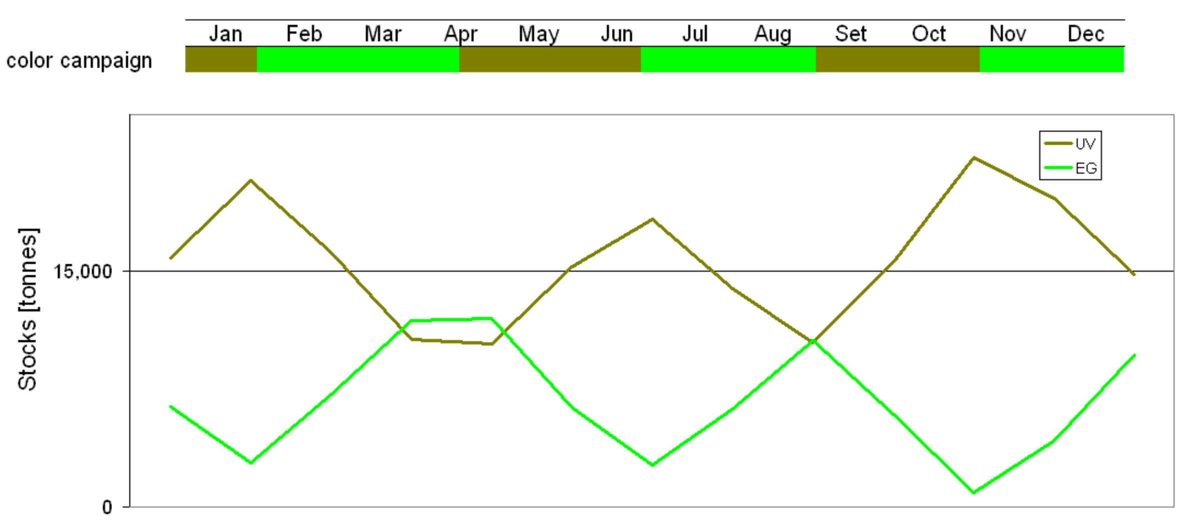
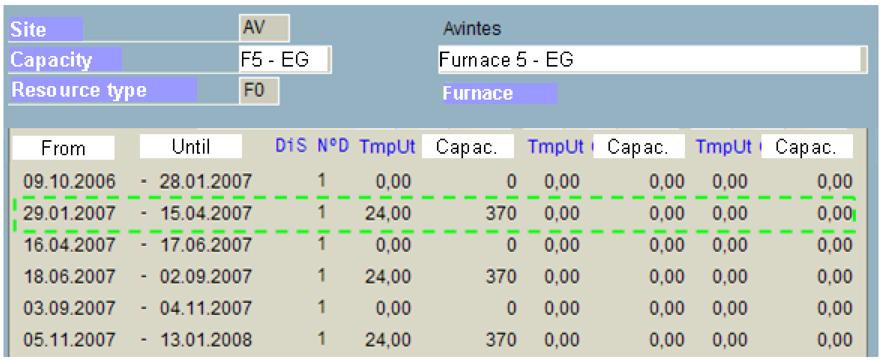


Figure 3.11: Color campaign scheduling for a furnace in 2007

production intentions are shown for product with reference 0018L019EG. To decide on how much to produce, PPD relies on a stock rotation policy defined by glass color. For instance, the production orders for flint containers must face 3 months of demand. If EB decides that stocks have to decrease, PPD has to plan smaller production runs (higher rotation).

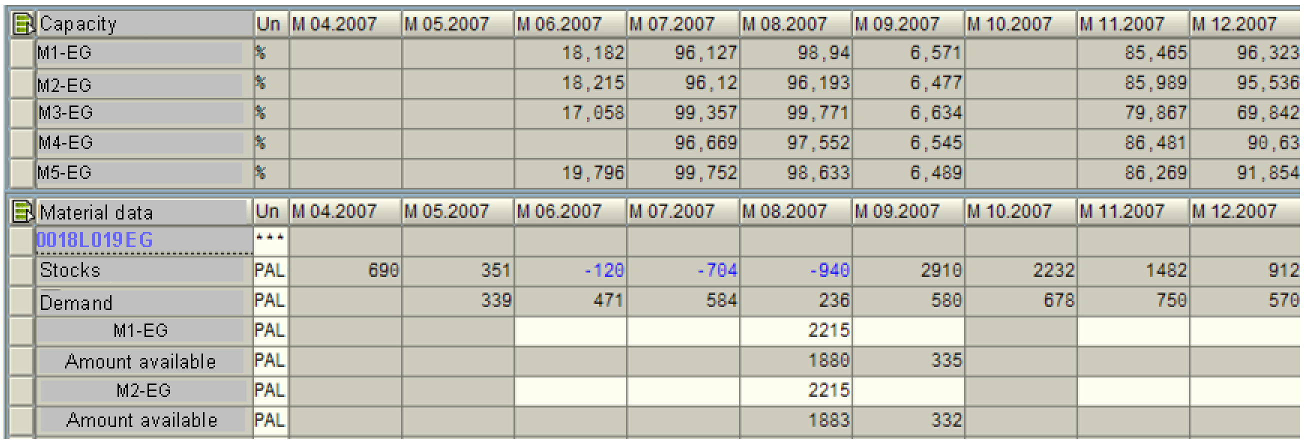
We note that the introduction of these orders is done manually, for more than 500 products and 28 machines. This hard time-consuming task benefits from using a preferential machine to introduce each product production intentions. The choice of the preferential machine respects the assignment of colors to furnaces (decided by EB) and, despite constraining the planning flexibility downwards, it most likely decreases the difficulty level of changeovers



Site	AV	Avintes	
Capacity	F5 - EG	Furnace 5 - EG	
Resource type	F0	Furnace	

From		Until	DiS	N°D	TmpUt	Capac.	TmpUt	Capac.	TmpUt	Capac.
09.10.2006	-	28.01.2007		1	0,00	0	0,00	0,00	0,00	0,00
29.01.2007	-	15.04.2007		1	24,00	370	0,00	0,00	0,00	0,00
16.04.2007	-	17.06.2007		1	0,00	0	0,00	0,00	0,00	0,00
18.06.2007	-	02.09.2007		1	24,00	370	0,00	0,00	0,00	0,00
03.09.2007	-	04.11.2007		1	0,00	0	0,00	0,00	0,00	0,00
05.11.2007	-	13.01.2008		1	24,00	370	0,00	0,00	0,00	0,00

Figure 3.12: Color Campaign SAP R/3 report



Capacity	Un	M 04.2007	M 05.2007	M 06.2007	M 07.2007	M 08.2007	M 09.2007	M 10.2007	M 11.2007	M 12.2007
M1-EG	%			18,182	96,127	98,94	6,571		85,465	96,323
M2-EG	%			18,215	96,12	96,193	6,477		85,989	95,536
M3-EG	%			17,058	99,357	99,771	6,634		79,867	69,842
M4-EG	%				96,669	97,552	6,545		86,481	90,63
M5-EG	%			19,796	99,752	98,633	6,489		86,269	91,854

Material data	Un	M 04.2007	M 05.2007	M 06.2007	M 07.2007	M 08.2007	M 09.2007	M 10.2007	M 11.2007	M 12.2007
0018L019EG	***									
Stocks	PAL	690	351	-120	-704	-940	2910	2232	1482	912
Demand	PAL		339	471	584	236	580	678	750	570
M1-EG	PAL					2215				
Amount available	PAL					1880	335			
M2-EG	PAL					2215				
Amount available	PAL					1883	332			

Figure 3.13: Assignment of products on machines

and makes the act of planning easier.

Whenever the demand does not match the production resources (revealed by unbalanced furnace/machine workloads), constant negotiations between PPD, marketing/commercial and plant managers enable the convergence of an (exhausting) iterating procedure. In the end, the company manufacturing machinery must suit the sales budget proposal. After being validated, the sales budget remains unchanged. Additionally, the annual sales plan, that incorporates real sales up to the current month and updated demand forecasts for the rest of the year, is revised monthly.

The sales budgeting takes place once a year, while the long-term planning process is performed throughout the year in a rolling horizon fashion. Apart from the first step of sales budgeting, the making-procedure of this plan is similar to that described right before.

Along with the long-term process, a short-term planning is triggered and aims to suggest the sequence of products on machines on a daily basis for a minimum of 4 weeks. Each plant operation director analyzes the plan from an industrial point of view, verifying its validity and frequently presents counterproposals.

Opportunities

Among the main decisions made at a strategic level, we highlight the assignment of colors to furnaces and the inventory management policy. Some furnaces have been exclusively dedicated to some colors for several years. It should be questioned whether such decision (that strangulates the planning process) has followed the market trend. As far as stocks are concerned, it is not straightforward that the rise of stock rotation endeavors a better customer response and strengthens the company key performance indicators (KPIs, leveraged by smaller holding costs), since setups absorb a considerable amount of capacity. This tradeoff is not accurately analyzed (remember that rotation stock policy is decided by EB). In addition, the consideration of a preferential machine to process each product ignores a huge amount of plans that would most likely perform better. Nevertheless, the allocation of products on other machines would demand a decision support system.

Even with these simplifications, PPD shows difficulties in looking-ahead for the entire 12-month rolling horizon. The fact that account managers usually inflate their predictions to ensure that PPD plans their “expected” amounts, stresses those troubles. In order to show it, we rely on data of Figure 3.14 that displays the annual sales plan revision throughout 2005. In January 2005 PPD analyzed a sales plan clearly above the 2005 sales budget (approved in November 2004), that to be feasible would have to benefit from a significant decrease on inventory level (since the total demand amount was well above the overall capacity). Such procedure could make the planning of farther months harder (due to the absence of capacity). The lack of visibility and planning for 12 months made PPD to accept such plan, postponing hurdles. In fact, those difficulties were detected and tackled on the new versions of the sales plan. As time went by, the planner could simulate more precisely the production planning for the last months of 2005.

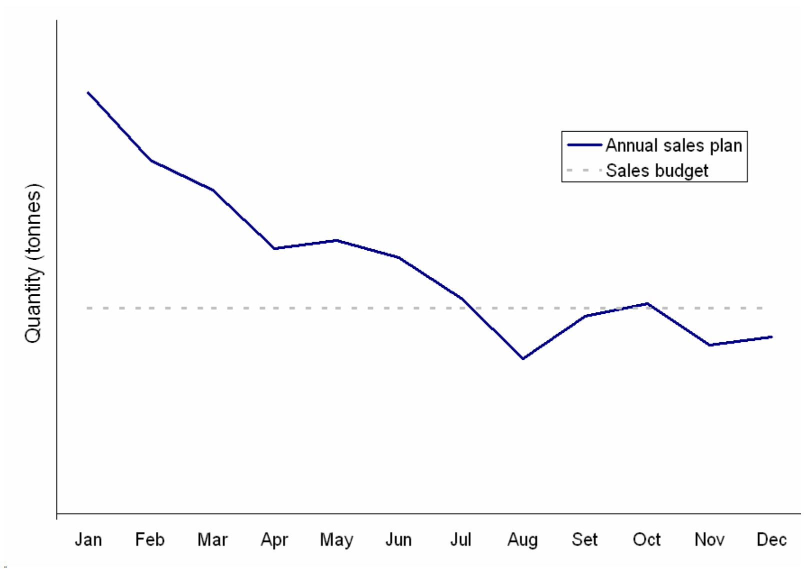


Figure 3.14: Annual sales plan revision throughout 2005

A typical deviation pattern between real sales and demand forecast throughout the year has been observed (see Figure 3.15). The pressure to accomplish the sales budget at the end of a year affects necessarily the first months of the following year. While accepting new orders, PPD should warn the marketing department that by forcing too much sales at the end of a season, the lack of both capacity and stocks will compromise the near future. These pronounced deviations roll over to the following year and they are not healthy for the company internal procedures and may compromise the customers' response and KPIs.

Finally, we would like to remark that the absence of a DSS may not be crucial when the company production resources are relatively stable and, consequently, the main strategic planning decisions are quite the same. However, strategic actions such as furnace (re)built and plant acquisition, do not allow PPD to rely on past decisions and, here, quantitative management methods are of a great help in carrying out the planning process. In Chapter 5 we will come back to this point.

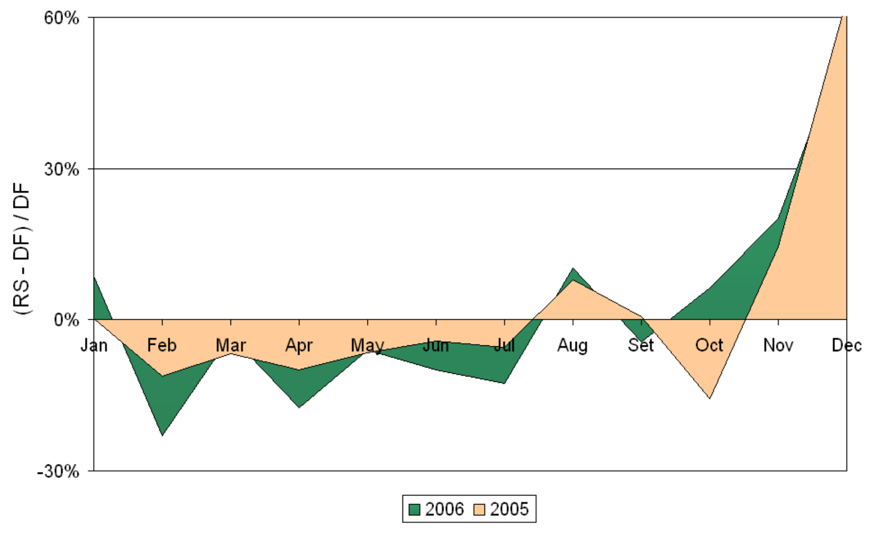


Figure 3.15: Deviations(%) between real sales (RS) and demand forecast (DF)

Chapter 4

Long-term production planning and scheduling: A first approach

Summary. In production planning in the glass container industry, machine dependent setup times and costs are incurred for switchovers from one product to another. The resulting multi-item capacitated lotsizing problem has sequence-dependent setup times and costs. We present two novel linear mixed integer programming formulations for this problem, incorporating all the necessary features of setup carryovers. The compact formulation has polynomially many constraints, while, on the other hand, the stronger formulation uses an exponential number of constraints that can be separated in polynomial time. We also present a five-step heuristic, which is effective both in finding a feasible solution (even for tightly capacitated instances) and in producing good solutions to these problems. We report computational experiments.

4.1 Introduction

Long-term production planning arising in the glass container industry is a complex process. It is a semi-continuous manufacturing process, which includes the glass production (the continuous part) and the container manufacturing (the discrete part). Furnaces are operated continuously and machine lines operate on a twenty four hour seven days a week basis.

During a product changeover on a molding machine, the furnace keeps feeding the machine with molten glass, however, the gobs are discarded and melted down again in the furnace (wasting a huge amount of energy, the main production cost). Fast job changes with fewer job change parts lower the energy costs. Therefore, there is a sequence dependent setup time (and proportional cost) in a product changeover. Since machine setups consume the furnace capacity, it is essential to carry the setup from one period to the next. This production planning problem results in a capacitated multi-item deterministic lotsizing problem.

4.2 Single-Machine Multi-product Capacitated Lotsizing with Sequence-dependent Setups

We now study single machine CLSP with sequence dependent setup times and costs. In each time period several products can be produced. Whenever a switch of a product is performed, a certain amount of time is consumed and the setup cost is incurred. This time and cost depend on the two products. In each time period at most a given amount of time can be allotted to switchover times and to the actual production time. This leads to sequence dependent setup times and costs. We first analyze the model proposed by Gupta and Magnusson [2005], prove that it is actually not a completely accurate formulation and add a new set of constraints into their model to provide an exact formulation for this problem (Almada-Lobo et al. [2008]). We then present two novel linear mixed integer programming formulations for single machine CLSP with sequence-dependent setup times and costs, incorporating all the necessary features of setup carryovers. The compact formulation has a polynomial number of constraints. The efficiency of this model is benchmarked against that presented by Gupta and Magnusson [2005]. The second formulation uses an exponential number of constraints, which makes it impractical to be solved directly. However, we generate constraints dynamically by presenting a polynomial time separation algorithm. We compare the strength of the linear programming relaxations of both formulations. The two formulations are improved by adding valid inequalities that lead to good lower bounds. Since this problem is NP-hard, in order to have upper bounds we also develop a five-step heuristic that is able to solve large problem instances. The flexibility of the heuristic's first two steps allows us to find feasible solutions even under tightly capacitated scenarios. The last three steps focus on solution quality. We compare the heuristic solution with the lower bound generated by the strongest model and with other known heuristics. The main contributions of our work are as follows. To the best of our knowledge, the two presented formulations are the first exact formulations for the problem under consideration. We also formally prove a relationship between the two linear programming relaxations. Another major contribution of our work is the underlying heuristic.

4.2.1 A note on Gupta and Magnusson's model

Gupta and Magnusson [2005] develop a model for CLSP with sequence dependent setup times and setup costs (see Appendix B). This formulation ensures that setups are preserved over idle periods, that idle time at the end of a period may be used to perform a setup for the first product to be produced at the beginning of the following period, and that the setup state of the machine at the end of a period is carried into the following period. Notwithstanding, this model does not eliminate disconnected subtours and, therefore, may generate infeasible solutions.

As before, t denotes time periods, which range from 1 to T, i and j index products, which are labeled from 1 to N, d_{it} denotes the demand of product i in period t, and h_i is the capacity-unit inventory carrying cost for product i. Moreover, we are given the following data:

s_{ij}	time needed to set up the machine from product i to product j,
c_{ij}	cost incurred to set up the machine from product i to product j.

As far as decision variables are concerned, X_{it} denote the quantity of product i produced in period t, I_{it} the stock of product i at the end of period t and Y_{it} a binary variable that equals one if product i is produced in period t. Production quantities are denoted in terms of fractions of available capacity, that is normalized to one unit per period ($C_t = 1 \ \forall t$). In addition, the following 0/1 decision variables are defined in [Gupta and Magnusson, 2005]: ψ_{it}, β_{it}, γ_{it}, δ_t, ω_{it} and T_{ijt}. ψ_{it} (β_{it}) equals one if product i is produced first (last) in period t. γ_{it} equals one if the machine is set up for product i at the end of period t. δ_t (ω_t) is defined to be zero (one) if exactly (at least) one product is produced in period t. Finally, T_{ijt} equals one if a setup occurs on the machine configuration state from product i to product j in period t.

In order to show that Gupta and Magnusson's model is not exact, we test this formulation on a small data set provided in Table 4.1 for a four-product, three-period problem.

Table 4.1: Data for the four product, three-period problem

	d_{it}			s_{ij}				c_{ij}				
	$t=1$	$t=2$	$t=3$	$j=1$	$j=2$	$j=3$	$j=4$	$j=1$	$j=2$	$j=3$	$j=4$	h_i
$i=1$	0.15	0.17	0.12	0	0.03	0.02	0.02	0	490	465	390	9
$i=2$	0.30	0.15	0.13	0.02	0	0.02	0.02	390	0	430	385	4
$i=3$	0.25	0.12	0.14	0.03	0.02	0	0.02	480	340	0	360	9
$i=4$	0.15	0.17	0.17	0.02	0.02	0.02	0	355	455	447	0	4

We solved the model using OPL Studio with CPLEX-MIP Solver. Table 4.2 shows the optimal solution to this instance, when machine is set up for product $i = 1$ at the beginning of the planning period, i.e., $\gamma_{10} = 1$.

Table 4.2: Optimal solution obtained by the original model

$t=1$	$t=2$	$t=3$
$X_{11} = 0.15$, $Y_{11} = 1$	$X_{12} = 0.29$, $Y_{12} = 1$	
$X_{21} = 0.30$, $Y_{21} = 1$	$X_{22} = 0.28$, $Y_{22} = 1$	
$X_{31} = 0.25$, $Y_{31} = 1$	$X_{32} = 0.12$, $Y_{32} = 1$	$X_{33} = 0.14$, $Y_{33} = 1$
$X_{41} = 0.24$, $Y_{41} = 1$	$X_{42} = 0.25$, $Y_{42} = 1$	
$I_{41} = 0.09$	$I_{12} = 0.12$, $I_{22} = 0.13$, $I_{42} = 0.17$	
$\omega_1 = 1$, $\delta_1 = 1$	$\omega_2 = 1$, $\delta_2 = 1$	$\omega_3 = 1$
$\psi_{11} = 1$, $\beta_{21} = 1$, $\gamma_{21} = 1$	$\psi_{22} = 1$, $\beta_{32} = 1$, $\gamma_{32} = 1$	$\psi_{33} = 1$, $\beta_{33} = 1$, $\gamma_{33} = 1$
$T_{141} = 1$, $T_{431} = 1$, $T_{321} = 1$	$T_{232} = 1$, $T_{142} = 1$, $T_{412} = 1$	

The objective value is 2354.64. However, note that this solution is not feasible, since a disconnected subtour of products 1 and 4 occurs in period 2. In fact, $T_{142} = T_{412} = 1$ and the machine is set up neither for product 1 nor for product 4 at both the beginning and the end of period 2, i.e., $\gamma_{11} = \gamma_{41} = \gamma_{12} = \gamma_{42} = 0$.

A feasible solution does not entail both disjoint paths and disjoints cycles, in order to sequence the production. It follows from constraints (B.7), (B.8), (B.13) and (B.14) of Gupta and Magnusson's model that a feasible solution has at most one path, but potentially many cycles (subtours). A disconnected subtour is a cycle that is not linked to γ's. The model of Gupta and Magnusson only links the configuration state of the machine at the beginning of a period with the first setup to be performed in that period (i.e., links $\gamma_{i(t-1)}$ to T_{ijt}), and the configuration state of the machine at the end of a period with the last setup to be performed in that period (i.e., links γ_{it} to T_{jit}).

To ensure feasibility, a new set of constraints must be added to the original model to eliminate disconnected subtours. This set makes use of an auxiliary continuous variable (V_{it}) that represents the machine state throughout any product sequence. The subtour elimination constraints can be expressed as

$$V_{jt} \geq V_{it} + N \cdot T_{ijt} - (N-1) - N \cdot \gamma_{i(t-1)} \quad , \quad \forall i \neq j, t \tag{4.1}$$

where N is the total number of products. These constraints apply whenever one subtour occurs in a period, forcing the machine to be set up at the beginning of that period to one of the products that are part of the subtour. Consider the following subtour Φ (i_1, i_2, ..., i_j, i_1), with $j < N$, in period t. The sum up of all the constraints (4.1) listed for every product belonging to that subtour, yields the following constraint:

$$N \cdot \sum_{i \in \Phi} \gamma_{i(t-1)} \geq j \tag{4.2}$$

It is clear that this constraint is only satisfied when the machine is set up at the beginning of period t for one of the products of the subtour.

We note that connected subtours are not avoided by constraints (4.1), however they will never occur in the optimal solution due to the triangle inequality of setup costs ($\forall i, k, l$: $c_{ki} + c_{il} \geq c_{kl}$).

The optimal solution of the modified model for the instance described earlier is represented in Table 4.3.

Table 4.3: Optimal solution obtained by the modified model

$t = 1$	$t = 2$	$t = 3$
$X_{11} = 0.15$, $Y_{11} = 1$	$X_{12} = 0.29$, $Y_{12} = 1$	
$X_{21} = 0.30$, $Y_{21} = 1$	$X_{22} = 0.28$, $Y_{22} = 1$	
$X_{31} = 0.25$, $Y_{31} = 1$	$X_{32} = 0.12$, $Y_{32} = 1$	$X_{33} = 0.14$, $Y_{33} = 1$
$X_{41} = 0.24$, $Y_{41} = 1$	$X_{42} = 0.25$, $Y_{42} = 1$	
$I_{41} = 0.09$	$I_{12} = 0.12$, $I_{22} = 0.13$, $I_{42} = 0.17$	
$\omega_1 = 1$, $\delta_1 = 1$	$\omega_2 = 1$, $\delta_2 = 1$	$\omega_3 = 1$
$\psi_{11} = 1$, $\beta_{21} = 1$, $\gamma_{21} = 1$	$\psi_{22} = 1$, $\beta_{32} = 1$, $\gamma_{32} = 1$	$\psi_{33} = 1$, $\beta_{33} = 1$, $\gamma_{33} = 1$
$T_{141} = 1$, $T_{431} = 1$, $T_{321} = 1$	$T_{242} = 1$, $T_{412} = 1$, $T_{132} = 1$	
$V_{21} = 2$, $V_{31} = 1$	$V_{12} = 1$, $V_{32} = 2$	

The objective function of the optimal solution is 2384.64. This optimal solution entails the production sequence $1 \rightarrow 4 \rightarrow 3 \rightarrow 2$ in period 1, and the production sequence $2 \rightarrow 4 \rightarrow 1 \rightarrow 3$ in period 2. In the last period only product 3 is produced and no setup is performed. Note that this solution is feasible and not unique. As an example, since both products 2 and 4 are produced in periods 1 and 2 and have identical holding costs, instead of just carrying inventory of product 4 from period 1 to period 2, we can also carry inventory of product 2. If we increase (decrease) the quantity of product 2 produced in period 1 (period 2) by any $\theta \in [0, 0.09]$, and decrease (increase) the quantity of product 4 produced in period 1 (period 2) by the same amount θ (with all other production quantities remaining the same as shown in Table 4.3), it is easy to see that this new solution has the same objective function value as the solution depicted in Table 4.3.

4.2.2 The First Formulation

We now present our first novel formulation for the general single-stage model involving multiple items to be scheduled on a single machine and assume that the triangle inequality with respect to the setup cost and time holds, i.e., $c_{ik} \leq c_{ij} + c_{jk}$ and $s_{ik} \leq s_{ij} + s_{jk}$ for all products i, j, and k. This assumption holds in many practical settings, and definitely, as we will discuss later on, in our glass container case.

In order to capture the setup cost and time, we need the decision variables T_{ijt}. We assume that $T_{iit} = 0$ for every product i. We allow the machine to be set up to product i at the end of a time period and then start the next time period with the same product but not incurring any extra setup cost. Since such a setting involves two consecutive time periods, a new decision variable needs to be introduced. Let

$$\alpha_{it} = \begin{cases} 1, & \text{if the machine is set up for product } i \text{ at the beginning of period } t, \\ 0, & \text{otherwise.} \end{cases}$$

We also need the auxiliary variables V_{it} that assign product i in period t.

Our first formulation (F_1) for CLSP with sequence-dependent setup costs and times, and setup carryover is as follows:

$$\min \sum_i \sum_j \sum_t c_{ij} \cdot T_{ijt} + \sum_i \sum_t h_i \cdot I_{it} \tag{4.3}$$

$$I_{it} = I_{i(t-1)} + X_{it} - d_{it} \qquad i \in [N], t \in [T] \tag{4.4}$$

$$\sum_i p_i \cdot X_{it} + \sum_i \sum_j s_{ij} \cdot T_{ijt} \le C_t \qquad t \in [T] \tag{4.5}$$

$$X_{it} \le M_{it} \cdot \left(\sum_j T_{jit} + \alpha_{it} \right) \qquad i \in [N], t \in [T] \tag{4.6}$$

$$\sum_i \alpha_{it} = 1 \qquad t \in [T] \tag{4.7}$$

$$\alpha_{it} + \sum_j T_{jit} = \alpha_{i(t+1)} + \sum_j T_{ijt} \qquad i \in [N], t \in [T] \tag{4.8}$$

$$V_{it} + N \cdot T_{ijt} - (N-1) - N \cdot \alpha_{it} \le V_{jt} \qquad i \in [N], j \in [N] \setminus \{i\}, t \in [T] \tag{4.9}$$

$$(X_{it}, I_{it}, \alpha_{it}, V_{it}) \ge 0, T_{ijt} \in \{0,1\}. \tag{4.10}$$

The objective function (4.3) minimizes the sum of sequence-dependent setup costs and the holding cost. Constraints (4.4) represent the inventory balances and (4.5) ensure that the total production and setup time in each period does not exceed the available capacity.

Requirement (4.6) guarantees that a product is produced only if the machine has been set up for it. Constraints (4.7)-(4.9) determine the sequence of products on the machine in each period and keep track of the machine configuration state by recording the product that a machine is ready to process (setup carryover information is thereby tracked). We next discuss these constraints further. Hereafter we refer to T_{jit} as the input setup for product i in period t and to T_{ijt} as the output setup for product i in period t.

The model does not explicitly have binary variables representing whether a product is produced or not in a certain period. Production of product i can occur in period t if the machine is set up for i at the beginning of t or if at least one input setup is performed for product i. These conditions are guaranteed by constraints (4.6).

Constraints (4.7) ensure that α_{it} is one for exactly one product in a given period, i.e., that the machine is set up for exactly one product at the beginning of each period.

Constraints (4.8) ensure a balanced network flow of the machine configuration states and carry the setup configuration state of the machine into the next period. If no setup is performed in period t, which happens when $T_{ijt} = 0$ for every i and j, configuration state of the machine is carried from period $t-1$ to period $t+1$. The idle time of a period t may also be used to perform an empty setup for the first product to be produced at the beginning of the next period. If there is an input setup and no output setup for product i in period t, it means this setup was the last one to be performed on the machine in period t and, consequently, the machine is configured for product i at the beginning of the next period, i.e., $\alpha_{i(t+1)} = 1$. On the other hand, if there is an output setup and no input setup, this means that the machine is configured for product i at the beginning of period t, i.e., $\alpha_{it} = 1$.

In general, constraints (4.8) impose that flow in equals flow out. Let us define a digraph $G = ([N], A = [N] \times [N])$, where nodes correspond to products and arc $a = (i, j)$ corresponds to the setup from product i to product j. It follows that the set $\{(i, j)|T_{ijt} = 1\}$ corresponds to a collection of disjoint paths and cycles. Constraints (4.4)-(4.8) and (4.10) admit solutions that have at most one path, but potentially many cycles, also called subtours, see Figure 4.1. The left configuration represents a solution that entails one path from product i_1 to product i_4 and three disconnected subtours. The right configuration illustrates a solution with a cycle that starts and ends in product i_1 and two disconnected subtours.

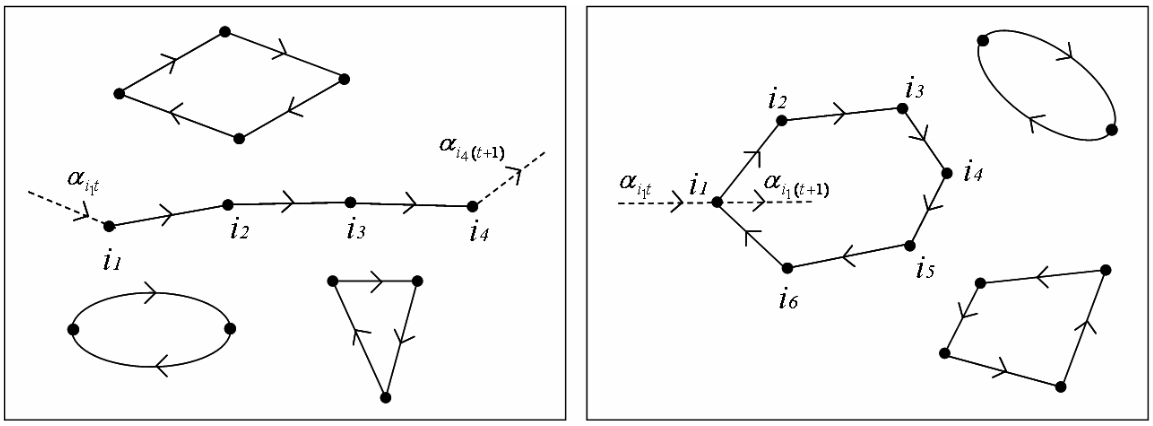


Figure 4.1: Possible configurations

In order to obtain a feasible solution, we must impose that we have only a single connected component linked to α's. Constraints (4.9) achieve this by using auxiliary variables that value the machine state through any sequence. Consider the following subtour $\Phi = (i_1, i_2, \ldots, i_m, i_1)$, with $m \leq N$, in period t. If constraints (4.9) are listed for every product belonging to the subtour, we get:

$$\begin{aligned}
V_{i_2t} \geq & V_{i_1t} + 1 - N \cdot \alpha_{i_1t} \\
V_{i_3t} \geq & V_{i_2t} + 1 - N \cdot \alpha_{i_2t} \\
& \vdots \\
V_{i_1t} \geq & V_{i_mt} + 1 - N \cdot \alpha_{i_mt}.
\end{aligned}$$

Summing them up, we obtain $N \cdot \sum_{i\in\Phi} \alpha_{it} \geq m$. If the subtour is not linked to α's then $\sum_{i\in\Phi} \alpha_{it} = 0$, which contradicts the previous statement. Therefore, constraints (4.9) avoid the occurrence of disconnected subtours. We note that constraints (4.4)-(4.10) allow solutions with cycles that are not disjoint, as that illustrated in Figure 4.2. However, such a scenario will never occur in an optimal solution since it is easy to see that the triangle inequality assumption implies that the in flow of every node is at most 1.

Now we argue that no feasible solution is excluded by (4.9). To this end, let α, T satisfy all of the remaining constraints. As before, let $\Phi = (i_1, i_2, \ldots, i_m, i_1)$ be the cycle corresponding to all T's at 1, and $\alpha_{i_1t} = \alpha_{i_1(t+1)} = 1$. Then we define $V_{it} = 0$ for all remaining nodes i. We also define $V_{i_jt} = j$ for $j = 1, \ldots, m$. It is easy to see that this V satisfies (4.9). We can

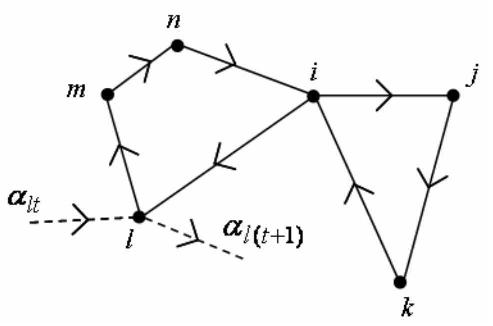


Figure 4.2: Connected subtours

argue similarly in the case of a path. A similar modeling trick is known in the context of the traveling salesman problem, see, e.g., Nemhauser and Wolsey [1988].

We conclude that the underlaying subgraph associated with an optimal T is either a cycle or a path. Thus it looks like that depicted in Figure 4.1 except that there are no disconnected subtours.

Note that the integrality condition of α_{it} is not necessary. Let us assume that $\alpha_{i1} = 1$. From (4.8) it follows that α_{j2} is an integer for every j. Together with (4.7) we obtain that α_{j2} is binary. This reasoning rolls over to the subsequent periods. If the machine is not set up for any product in the first time period, then the integrality requirements of α_{j1}'s need to be added. As an alternative to avoid it, the machine can be set up for a virtual product at the beginning of the planning period. No setup times or costs would be incurred if a setup occurs between this product and any other one.

The presented model assumes the single-machine case. The extension of this model to multiple machines is straightforward, also considering other usual features, such as shortages, backlogging costs and lower and upper bounds on lot sizes.

Example 1. Consider the following small data set provided in Table 4.4 for a three-product, three-period problem.
Setup times are $s_{ij} = 5$ time units for $i \neq j$ and zero otherwise. Moreover, $C_t = 100$ for every t and $p_i = 1$ for every i.

We solved the model by using CPLEX 9.0. Figures 4.3 and 4.4 show the optimal solution

Table 4.4: Data for the three product, three-period problem

	d_{it}			c_{ij}			
	$t=1$	$t=2$	$t=3$	$j=1$	$j=2$	$j=3$	h_i
$i=1$	15	5	10	0	3	3	10
$i=2$	20	35	20	4	0	3	15
$i=3$	0	110	40	5	5	0	20

for this instance, when the machine is set up for product 3 at the beginning of the planning period, i.e., $\alpha_{31} = 1$. Note that there is idle time at the end of period 3.

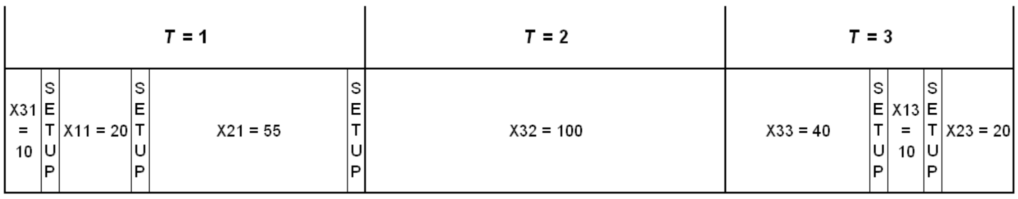


Figure 4.3: Gantt chart of the optimal solution

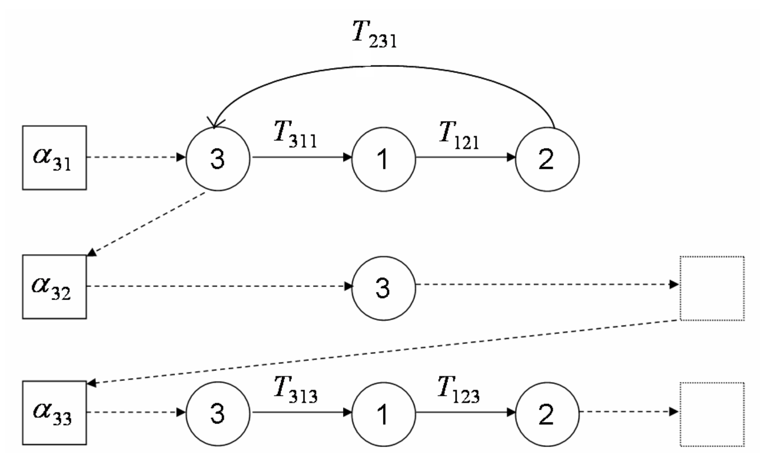


Figure 4.4: Network representation of the optimal solution

It entails seven periods of production and five setups. The last setup in period 1 is an empty one, since the idle time at the end of that period is used to set up for product 3, the first one to be produced at the beginning of period 2. In period 2 only product 3 is produced and no setup is performed. Therefore, the setup state of the machine for product

3 is carried from period 1 to period 3. At the end of the planning horizon the machine is ready to produce product 2. □

4.2.2.1 Benchmarking the First Formulation

In this section we compare our model F_1 with Gupta and Magnusson's model. The experiments were conducted on an IBM machine with a 3.2 GHz processor and 1GB of random access memory and CPLEX 9.0 as a mixed integer programming solver.

In Table 4.5 we conduct a benchmark on the complexity (in terms of the number of binary variables, continuous variables and constraints) of F_1 with respect to the models presented in Clark and Clark [2000] and Gupta and Magnusson [2005]. Here S is the maximum allowable number of setups per period used in Clark and Clark's model.

Table 4.5: Benchmark of models' complexity

Model	Binary variables	Continuous variables	Constraints
F_1	N^2T	$4NT$	$NT(2+N)+2T$
Clark and Clark [2000]	$N^2(ST-1)+N$	SNT	$NT(SN+2S+2)$ $+1-N$
Gupta and Magnusson [2005]	N^2T+4NT	$2NT$	$NT(5+3N)+8T$

We can observe that F_1 requires fewer binary variables than the other two models. In Example 1, F_1 has 27 binary variables, whereas Clark and Clark's model has 75 (S has to be set to three to reach the same optimal solution) and the Gupta and Magnusson's model has 63 binary variables.

In order to evaluate the models computationally, we generated random instances based on the parameters used by Haase and Kimms [2000]. The machine is always set up for product 1 at the beginning of the planning horizon. The number of products N ranges from 3 to 10, the number of periods were 3, 4, 5, 7, 10, and 15, and the capacity utilization (Cut) defined as $\sum_i d_{it}/C_t$ takes the values 0.6 and 0.8. Processing times (p_i) are unitary for all products. Demand (d_{it}) is selected based on the uniform distribution $U[40, 60]$, holding costs (h_i) based on $U[2, 10]$, and setup times (s_{ij}) based on $U[5, 10]$ (the triangle inequalities hold).

The setup costs c_{ij} are proportional to the setup times, i.e., $c_{ij} = \theta \cdot s_{ij}$ with $\theta = 50$. Note that implicity θ defines a relationship between the setup and holding costs. In the case of the company producing glass containers that inspired this work, θ falls in the interval $[50, 75]$.

For each triplet N, T, and Cut, three different instances were generated and the average running times (in seconds) were calculated. Tables 4.6 and 4.7 give the computational times to obtain an optimal solution using model F_1. An empty field means that at least one of the instances could not be solved optimally within a 3600 seconds limit.

Table 4.6: Average running times for $Cut = 0.6$ and $\theta = 50$ (model F_1)

	T					
N	3	4	5	7	10	15
3	0.1	0.1	0.1	0.2	0.2	14.7
4	0.1	0.1	0.3	2.8	1.3	34.7
5	0.2	0.3	0.7	7.3	7.9	300.1
6	0.2	0.3	0.7	2.3	24.5	593.6
7	0.5	1.2	2.0	5.1	310.7	1628.0
8	0.7	1.1	4.7	37.7	162.9	
9	0.7	2.1	9.4	55.5	967.8	
10	1.4	1.2	39.7	313.1	2875.7	

A comparison between the efficiency of F_1 and the Gupta and Magnusson's model (denoted by GM) for $Cut = 0.6$ and $Cut = 0.8$ is shown in Figures 4.5 and 4.6. We did not compare F_1 to the Clark and Clark's model since multiple runs with different values for S would be needed to reach the same optimal solution. The vertical axis denoted by F_1/GM represents the ratio of the respective computational times.

It is clear that the F_1 model is always more efficient than the GM model. Moreover, as the number of periods increases, the ratio F_1/GM tends to 0, i.e., the performance of F_1 is significantly better with increasing T. It turns out that the number of products does not significantly influence the relative efficiency. The GM model was not able to solve optimally within an hour instances with $T = 15$, $N > 4$ and $T = 10$, $N > 6$ for $Cut = 0.6$, and $T = 15$, $N > 3$, $T = 10$, $N > 6$ and $T = 7$, $N > 9$ for $Cut = 0.8$.

Table 4.7: Average running times for $Cut = 0.8$ and $\theta = 50$ (model F_1)

	T					
N	3	4	5	7	10	15
3	0.1	0.1	0.1	0.3	3.5	47.8
4	0.1	0.1	0.4	1.5	25.1	943.4
5	0.3	0.4	0.9	8.3	103.7	2410.0
6	0.3	0.4	1.4	16.2	323.0	
7	0.8	1.7	4.2	20.7	1080.0	
8	0.9	2.7	7.3	57.6	2390.7	
9	2.1	8.7	13.7	387.1		
10	5.9	12.4	17.2	815.6		

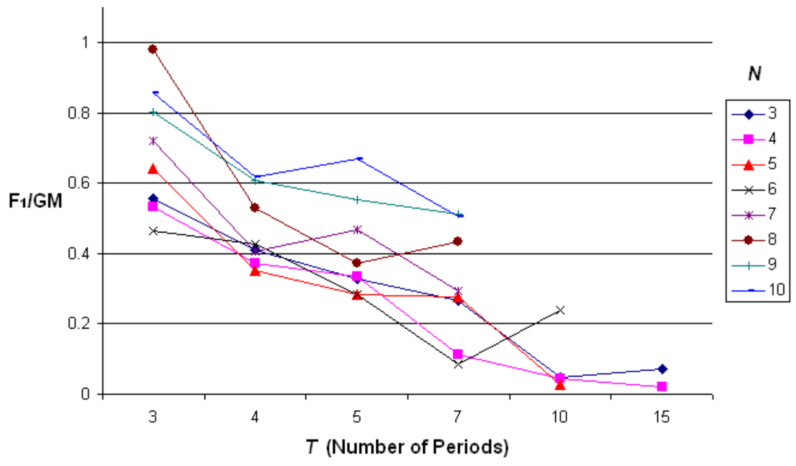


Figure 4.5: Comparison of run-time performance between the F_1 and GM models for $Cut = 0.6$

4.2.3 The Compact Formulation

In this section we propose an alternative equivalent formulation (F_2) to CLSP with sequence dependent setup times and setup costs.

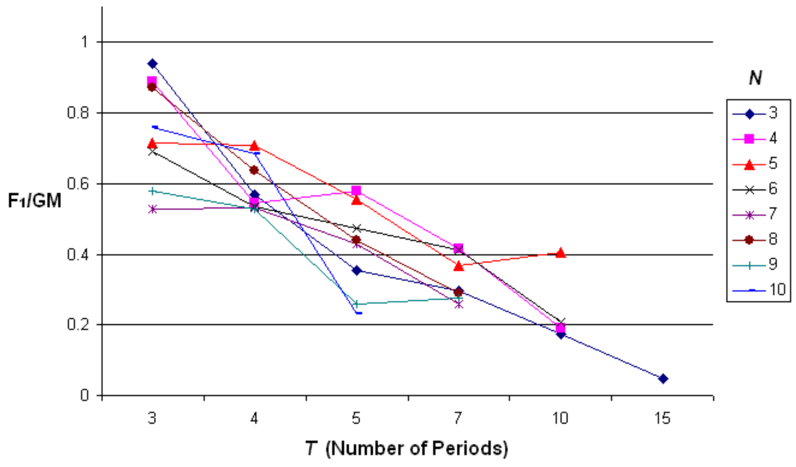


Figure 4.6: Comparison of run-time performance between the F_1 and GM models for $Cut = 0.8$

Since the triangle inequality holds for both setup costs and setup times, constraints

$$\sum_j T_{ijt} \leq 1 \qquad i \in [N], t \in [T] \tag{4.11}$$

$$\sum_j T_{jit} \leq 1 \qquad i \in [N], t \in [T] \tag{4.12}$$

are always satisfied in an optimal solution. Let

$$\vartheta_{it} = \alpha_{i(t+1)} + \sum_j T_{ijt} \qquad i \in [N], t \in [T] \tag{4.13}$$

denote auxiliary variables. The following inequalities are an alternative formulation for constraints (4.9):

$$\sum_{i \in S} \sum_{j \in S} T_{ijt} \leq \sum_{i \in S} \vartheta_{it} - \vartheta_{kt} + \alpha_{kt} \qquad t \in [T], k \in S, S \subseteq [N]. \tag{4.14}$$

By substituting (4.13) in (4.14) and taking into account (4.8), we obtain the equivalent requirement

$$\sum_{i \in S} \sum_{j \notin S} T_{ijt} + \sum_{i \in S} \alpha_{i(t+1)} \geq \sum_j T_{jkt} \qquad t \in [T], k \in S, S \subseteq [N]. \tag{4.15}$$

If there is a subtour on a subset S of nodes, the left-hand side of (4.14) equals to $|S|$. If $\alpha_{it} = 0$ for every $i \in S$, this constraint is violated since its right-hand side equals to $|S| - 1$ for every product k in the cycle. Note that $\vartheta_{it} = 1$ for every $i \in S$.

Now we argue that a feasible solution is not excluded by (4.15). To this end, let (X, I, α, T) be feasible for F_1. It suffices to consider the case where the right-hand side of (4.15) equals one. For a fixed S and $k \in S$, let $h \in [N]$ be such that $T_{hkt} = 1$. We distinguish two cases: $h \in S$ and $h \notin S$.

$h \in S$: If subset S contains the node set of the entire cycle or path corresponding to the solution, then $\sum_{i \in S} \alpha_{i(t+1)} = 1$ and (4.15) is satisfied. On the other hand, if this is not the case, then we have to consider three different scenarios.

(a) If the solution entails a cycle, then clearly $\sum_{i \in S} \sum_{j \notin S} T_{ijt} \geq 1$ since the cycle must "enter" and "leave" S.

(b) If the solution consists of a path that ends at node k, then $\alpha_{k(t+1)} = 1$ and, therefore, $\sum_{i \in S} \alpha_{i(t+1)} = 1$.

(c) If the solution consists of a path that does not end at node k, then either the path ends at another node in S (and $\sum_{i \in S} \alpha_{i(t+1)} = 1$) or it ends at a node not in S and, in this case, $\sum_{i \in S} \sum_{j \notin S} T_{ijt} \geq 1$.

Each of these scenarios yields constraint (4.15).

$h \notin S$: This case is similar to the above case.

The F_2 formulation reads

$$\min \sum_i \sum_j \sum_t c_{ij} \cdot T_{ijt} + \sum_i \sum_t h_i \cdot I_{it}$$
$$\text{subject to } (4.4) - (4.8), (4.11), (4.12), (4.15)$$
$$(X_{it}, I_{it}, \alpha_{it}) \geq 0, T_{ijt} \in \{0, 1\}.$$

Let P_{F_1} and P_{F_2} denote the feasible sets of the linear relaxations of the formulations F_1 and F_2, respectively. In the following theorem we show that F_2 is at least as strong as F_1.

Theorem 1. *We have* $P_{F_2} \subseteq \mathrm{proj}(P_{F_1})$*, where* proj *denotes the projection of* P_{F_1} *onto the space of* $(X_{it}, I_{it}, T_{ijt}, \alpha_{it})$*.*

Proof. We demonstrate the statement by exhibiting the appropriate $V's$. They are defined as shortest path distances with respect to predefined weights. Since shortest path distances are well defined on networks with nonnegative weight cycles, we rely on (4.14) and appropriately selected weights to show this.

Let $(X^*_{it}, I^*_{it}, T^*_{ijt}, \alpha^*_{it}) \in P_{F_2}$ be arbitrary. We find V^*_{it} such that (4.9) hold, i.e.,

$$(X^*_{it}, I^*_{it}, T^*_{ijt}, \alpha^*_{it}, V^*_{it}) \in P_{F_1} \ .$$

Consider time period t and let $w_{ij} = (N-1) + N \cdot \alpha^*_{it} - N \cdot T^*_{ijt}$ for every i, j. Consider network G defined in Section 4.2.2, where each arc has weight w_{ij}. Let us fix an arbitrary source node s and let $dist_i$ denote the length of the shortest path from the source node to node i. We show later that these distances are well defined. They satisfy the following optimality conditions:

$$dist_i + w_{ij} \geq dist_j \qquad (i,j) \in A(G).$$

Then, by setting $\overline{V}_{it} = -dist_i$, we obtain

$$\overline{V}_{it} - (N-1) - N \cdot \alpha^*_{it} + N \cdot T^*_{ijt} \leq \overline{V}_{jt}. \tag{4.16}$$

To ensure that V_{it} are non-negative, we define $V^*_{it} = \overline{V}_{it} + \max_{j,\bar{t}} |\overline{V}_{j\bar{t}}|$. Clearly, from (4.16) it follows that (V^*_{ijt}, T^*_{ijt}) satisfy (4.9). We conclude that $(X^*_{it}, I^*_{it}, T^*_{ijt}, \alpha^*_{it}, V^*_{it}) \in P_{F_1}$.

In order to complete the proof, we need to show that $dist_i$ are well defined. This is equivalent to showing that there are no negative cost cycles with respect to w (see, e.g., Ahuja et al. [1993]). We need to show that $\sum_{(i,j)\in C} w_{ij} \geq 0$ for any cycle C. Let S be the node set of cycle C and $k \in S$. From (4.14) and nonnegativity of T^*, we have

$$\sum_{(i,j)\in C} T^*_{ijt} \leq \sum_{i,j \in S} T^*_{ijt} \leq \sum_{i \in S} \vartheta_{it} - \vartheta_{kt} + \alpha^*_{kt}. \tag{4.17}$$

Since

$$\sum_{(i,j)\in C} w_{ij} = (N-1) \cdot |S| + N \cdot \sum_{i \in S} \alpha^*_{it} - N \cdot \sum_{(i,j)\in C} T^*_{ijt} \tag{4.18}$$

and by using (4.17), we obtain

$$\sum_{(i,j)\in C} w_{ij} \geq (N-1)\cdot|S| + N\cdot\sum_{i\in S}\alpha^*_{it} - N\cdot\sum_{i\in S}\vartheta_{it} + N\cdot\vartheta_{ik} - N\cdot\alpha^*_{ik}. \tag{4.19}$$

Incorporating (4.13) into (4.19), yields

$$\sum_{(i,j)\in C} w_{ij} \geq (N-1)\cdot|S| - N\cdot\sum_{i\in S\setminus\{k\}}\sum_{j} T^*_{jit}.$$

Taking into account (4.12), leads to

$$\sum_{(i,j)\in C} w_{ij} \geq (N-1)\cdot|S| - N\cdot(|S|-1) \geq 0,$$

since $|S| \leq N$. □

Note that F_2 has an exponential number of constraints. To introduce the most violated (t,S,k) inequalities (4.15) we need to solve the separation problem. We next show how to solve this problem.

Let $(X^*_{it}, I^*_{it}, T^*_{ijt}, \alpha^*_{it})$ satisfy (4.4) – (4.8), (4.11) and (4.12). Consider network G with capacity T^*_{ijt} on arc (i,j) in A. Next we define a new network G^0 by adding sink node z to G and arcs (i,z) for every product i with capacity $\alpha^*_{i(t+1)}$. Finally, we solve the min (s,z) cut problem for every product s. The algorithm hinges on the fact that the value of an $s-z$ cut S in G^0 equals to $\sum_{i\in S}\sum_{j\notin S} T^*_{ijt} + \sum_{i\in S}\alpha^*_{i(t+1)}$. If there exists a $k \in S$ such that $\sum_j T^*_{jkt} > Z^*$ with Z^* being the value of the minimum $s-z$ cut, then the (t,S,k) inequality is violated.

Since finding a minimum cut in a directed graph can be performed in polynomial time, this separation problem is polynomial.

4.2.3.1 Valid Inequalities

In this section we present two classes of valid inequalities (cuts) that strengthen F_2.

The first class of valid inequalities is not a family in the original space, but we need to consider a higher dimensional space. Let W_t denote a binary variable that equals 0 if at least one setup is performed in period t and 1 otherwise. Then, the following sets of valid inequalities, denoted (W_t) inequalities, are valid:

$$\begin{aligned}
\alpha_{it} \leq & \sum_j T_{ijt} + W_t && i \in [N], t \in [T] \\
\alpha_{it} \leq & \sum_j T_{ji(t-1)} + W_{t-1} && i \in [N], t \in [T] \\
1 - W_t \leq & \sum_{kj} T_{kjt} && t \in [T] \\
W_t \leq & 1 - \frac{\sum_{kj} T_{kjt}}{N} && t \in [T].
\end{aligned}$$

To prove that they are valid, it suffices to consider the case $\alpha_{it} = 1$. We distinguish the following four cases:

$\alpha_{i(t-1)} = 0, \alpha_{i(t+1)} = 0$: In this case $\sum_j T_{ji(t-1)} \geq 1, \sum_j T_{ijt} \geq 1$ and $W_t = W_{t-1} = 0$, therefore the inequalities are satisfied;

$\alpha_{i(t-1)} = 1, \alpha_{i(t+1)} = 0$: In this case $\sum_j T_{ijt} \geq 1$ and $W_t = 0$. If $\sum_{kj} T_{kj(t-1)} \geq 1$, then also $\sum_j T_{ji(t-1)} \geq 1$ (and $W_{t-1} = 0$) and they are valid. On the other hand, if $\sum_{kj} T_{kj(t-1)} = 0$, then $W_{t-1} = 1$ and again they are valid;

$\alpha_{i(t-1)} = 0, \alpha_{i(t+1)} = 1$: This case is similar to the above case;

$\alpha_{i(t-1)} = 1, \alpha_{i(t+1)} = 1$: It is easy to adapt the second case to this case.

The introduction of W_t does not increase the computational time significantly since there are only T of them, and, furthermore, their integrality is implied.

It is also helpful to use the well known (l, S) inequalities for uncapacitated single-item lotsizing (Barany et al. [1984]). The (l, S) inequalities adjusted for the multi-item case of our problem are expressed as

$$d_{i1l} \leq \sum_{t \in [l] \setminus S} d_{itl} \cdot \left(\sum_j T_{jit} + \alpha_{it} \right) + \sum_{t \in S} X_{it} \qquad l \leq T, S \subseteq [l], \tag{4.20}$$

where $d_{ipg} = \sum_{t=p}^{g} d_{it}$. Since there is an exponential number of these constraints, we introduce them as cutting planes (instead of adding all of them *a priori*). To find the most violated (l, S) inequalities, we use the separation algorithm introduced by Barany et al. [1984], which requires $O(NT^2)$ time steps.

4.2.4 A Heuristic for CLSP

In this section we propose a heuristic for finding good solutions to CLSP, i.e., upper bounds on the optimal value. The heuristic consists of five basic steps. The first two attempt to find an initial feasible solution, whilst the last three are geared toward improving the quality of the solution. In the first step the machine is loaded in each period with production gross requirements. Such a solution typically violates the capacity requirements. In the second step the production is sequenced from period T to 1. Whenever the capacity constraint is violated, we perform a procedure to eliminate overtime, which is the amount of capacity violation. The third step is geared toward eliminating setups by anticipating production, and the forth step tries to reduce holding costs by postponing production. Finally, the last step attempts to find a better coordination between periods, eliminating, whenever possible, empty setups.

The heuristic starts with a lot-for-lot forward pass, allocating to each period the demand for that period ($X_{it} = d_{it}$, for every i, t), without considering the capacity constraints.

The second step works backwards and performs, in each period, both sequencing and amending procedures. In a given time period, the sequencing procedure is basically a *minmax* algorithm; in each pass a new arc (u, v) is selected due to either the variability of the input setup times of its initial node u (ΔI_u) or the variability of the output setup times of its terminal node v (ΔO_u). The variability is measured as the difference between the two smallest setup time values. The node with the maximum variability among the head and tail nodes is chosen. Then, the other node of the arc is the argument that minimizes the setup time of the selected arc. The sequencing procedure tries to minimize the sequence dependent setup times instead of setup costs, therefore, feasibility is placed over optimality in this step. Since cycles are not taken into account and each product is produced, the final sequence (path) of N products has $N - 1$ edges. The sequencing heuristic is given in Algorithm 1, where Θ is a large number, e.g., $\Theta = 1 + \max_{i,j} s_{ij}$.

Figure 4.7 gives an example of the procedure described in Algorithm 1. Instead of setting the time of already considered arcs to Θ, we delete such arcs. The option of considering the difference between the smallest and the second smallest setup time is preferred over the standard greedy approach of basing the selection only on the smallest setup time to

```
loop $N-1$ times
  for $k = 1$ to $N$
    $i_1 = \arg\min_{i \in [N]}(s_{ik} : s_{ik} \neq \Theta)$
    $i_2 = \arg\min_{i \in [N]\setminus\{i_1\}}(s_{ik} : s_{ik} \neq \Theta)$
    $j_1 = \arg\min_{j \in [N]}(s_{kj} : s_{kj} \neq \Theta)$
    $j_2 = \arg\min_{j \in [N]\setminus\{j_1\}}(s_{kj} : s_{kj} \neq \Theta)$
    $\Delta O_k = s_{kj_2} - s_{kj_1}$
    $\Delta I_k = s_{i_2k} - s_{i_1k}$
  end for
  $i' = \arg\max_i(\Delta O_i), j'' = \arg\min_{j \in [N]}(s_{i'j})$
  $j' = \arg\max_j(\Delta I_j), i'' = \arg\min_{i \in [N]}(s_{ij'})$
  if $\Delta O_{i'} > \Delta I_{j'}$ or ($\Delta O_{i'} = \Delta I_{j'}$ and $s_{i'j''} \leq s_{i''j'}$) then
    $T_{i'j''t} = 1$
  else
    $T_{i''j't} = 1, i' = i'', j'' = j'$
  end if
  $s_{j'i''} = \Theta$
  for $k = 1$ to $N$
    $s_{i'k} = s_{kj''} = \Theta$
  end for
end loop
```

Algorithm 1: The Sequencing Procedure in Period t

"hedge" against the possibility of making a bad decision and later having very little room for correction. In addition, computational experiments have revealed this to be a superior criterion. For $t < T$, the last product to be scheduled is the first product in period $t+1$ (the sink node is specified before generating the path in Algorithm 1).

Let $O_t = \max\left\{0, \sum_i p_i \cdot X_{it} + \sum_{ij} s_{ij} \cdot T_{ijt} - C_t\right\}$ denote the amount of overtime in period t. After the sequencing procedure, there can be time periods with positive overtime, i.e., with violated capacity. In the amending pass, we push overtime towards the previous

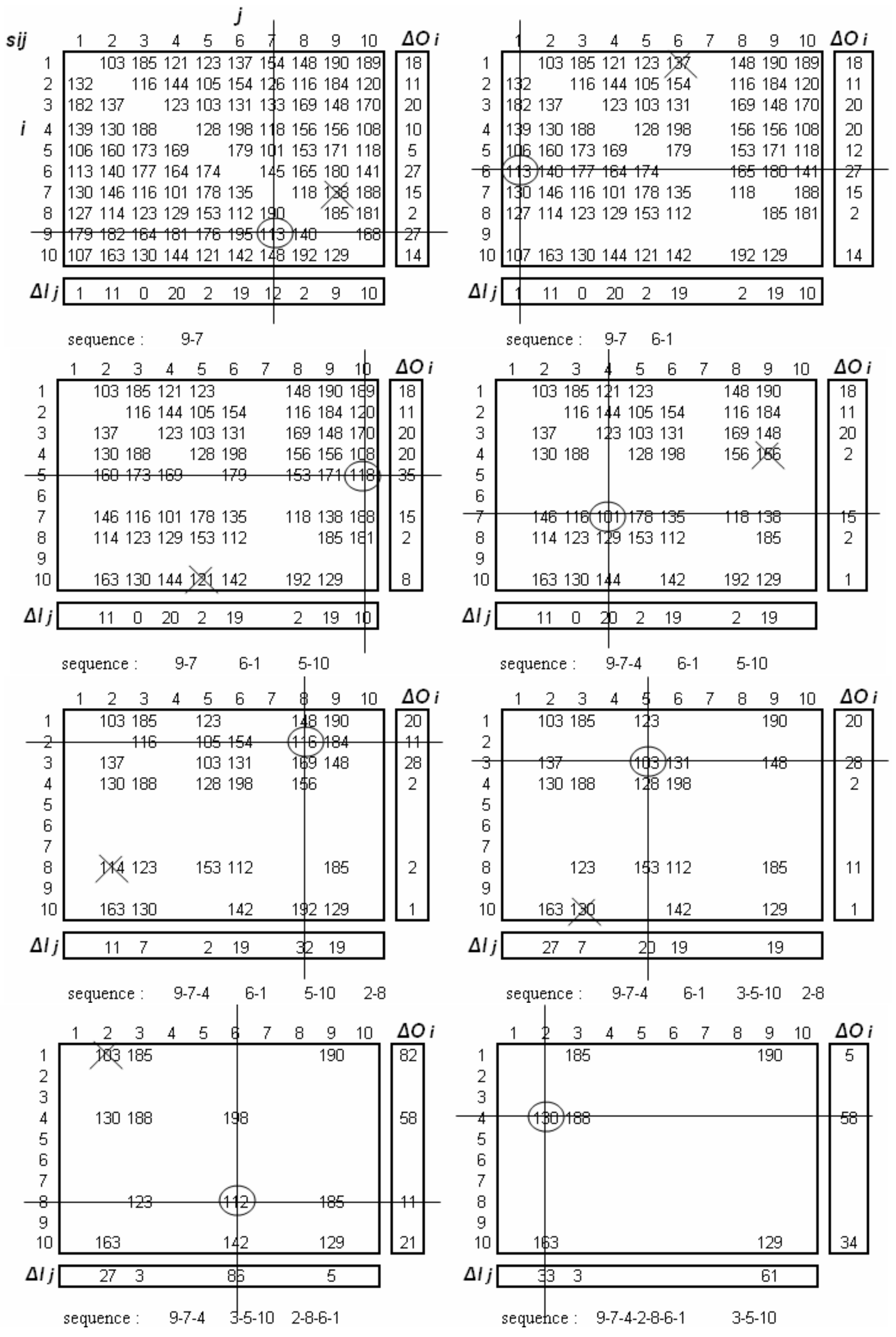
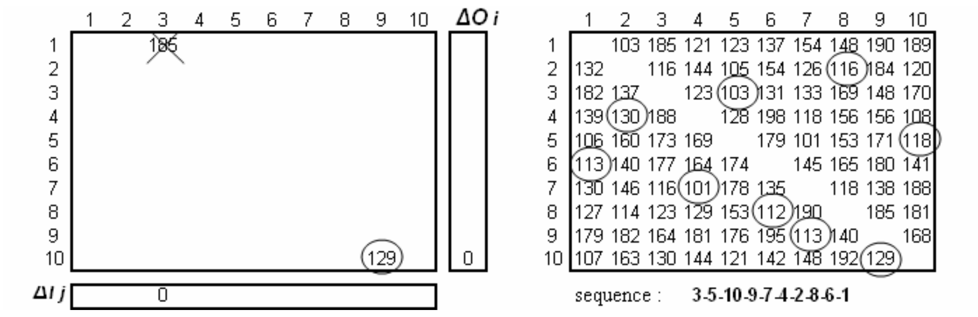

sij
j
i
ΔO i
ΔI j
sequence : 9-7
sequence : 9-7 6-1
sequence : 9-7 6-1 5-10
sequence : 9-7-4 6-1 5-10
sequence : 9-7-4 6-1 5-10 2-8
sequence : 9-7-4 6-1 3-5-10 2-8
sequence : 9-7-4 3-5-10 2-8-6-1
sequence : 9-7-4-2-8-6-1 3-5-10

Figure 4.7: Illustrative example



	1	2	3	4	5	6	7	8	9	10
1		103	185	121	123	137	154	148	190	189
2	132		116	144	105	154	126	116	184	120
3	182	137		123	103	131	133	169	148	170
4	139	130	188		128	198	118	156	156	108
5	106	160	173	169		179	101	153	171	118
6	113	140	177	164	174		145	165	180	141
7	130	146	116	101	178	135		118	138	188
8	127	114	123	129	153	112	190		185	181
9	179	182	164	181	176	195	113	140		168
10	107	163	130	144	121	142	148	192	129	

sequence : **3-5-10-9-7-4-2-8-6-1**

Figure 4.7: Continued

period. While working on period t, we might create new overtime in period $t-1$. In the case $O_t > 0$, in order to determine the product or set of products whose (partial or entire) lots will be moved backwards, the following options are sequentially taken into account in the amending procedure (an option is only considered if the preceding one has not eliminated O_t).

(a) Let $S = \{i \in [N] : p_i \cdot X_{it} \geq O_t \text{ and } X_{i(t-1)} > 0\}$ and $k = \arg\min_{i \in S} h_i$.
Move O_t/p_k of X_{kt} to period $t-1$.

(b) Let $S = \{i \in [N] : 0 < p_i \cdot X_{it} < O_t \text{ and } X_{i(t-1)} > 0\}$, and let fs_{it} denote the minimum savings in setup time by not producing product i in period t. Formally,

$$fs_{it} = \begin{cases} \sum_j T_{ijt} \cdot s_{ij} & \text{if } \alpha_{it} = 1 \text{ and } \alpha_{i(t+1)} = 0 \text{ and } X_{it} > 0, \\ \min\limits_{\substack{X_{kt}>0, X_{jt}>0 \\ k \neq i,\, j \neq i}} [s_{ki} + s_{ij} - s_{kj}] & \text{if } \alpha_{it} = 0 \text{ and } \alpha_{i(t+1)} = 0 \text{ and } X_{it} > 0, \\ 0 & \text{otherwise.} \end{cases}$$

Note that $\alpha_{i(t+1)} = 1$ is not considered because it would modify the production sequence in downstream periods.

Let $k = \arg\min_{i \in S}(p_i \cdot X_{it} + fs_{it} : p_i \cdot X_{it} + fs_{it} \geq O_t)$. If k is not defined, then $k = \arg\max_{i \in S}(p_i \cdot X_{it} + fs_{it})$.
Move all X_{kt} to period $t-1$ and return to (a) if $O_t > 0$.

(c) Let $S = \{i \in [N] : p_i \cdot X_{it} \geq O_t\}$ and $k = \arg\min_{i \in S} h_i$.
Move O_t/p_k of X_{kt} to period $t-1$.

(d) Let $S = \{0 < p_i \cdot X_{it} < O_t\}$ and $k = \arg\min_{i \in S}(p_i \cdot X_{it} + fs_{it} : p_i \cdot X_{it} + fs_{it} \geq O_t)$.

If k is not defined, then $k = \arg\max_{i \in S}(p_i \cdot X_{it} + fs_{it})$.
Move all X_{kt} to period $t-1$, and return to (a) if $O_t > 0$.

We note that steps (c) and (d) are similar to (a) and (b), respectively, however, they lead to a higher capacity consumption in period $t-1$ since a new setup has to be performed. This is the reason for placing step (c) after step (b) as opposed to execute (a) and (c) simultaneously. Proceeding from the last period to the first period, the solution is either feasible or there is overtime in period one (and no feasible solution is generated). We point out that finding a feasible solution is NP-complete, Trigeiro et al. [1989]. If the first two steps do not find a feasible solution, then the entire heuristic fails.

The following steps try to improve the quality of the initial solution. Let $\phi = \{(i,t) : X_{it} > 0\}$, let fc_{it} denote the minimum savings in setup costs by not producing product i in period t (its expression is identical to fs_{it} with setup costs instead of setup times as one of the arguments), and let $Ca_t = \max\left\{0, C_t - \sum_i p_i \cdot X_{it} - \sum_{ij} s_{ij} \cdot T_{ijt}\right\}$ denote the available (idleness) capacity in period t. The third step of the heuristic is a backwards pass in time that seeks to avoid the cost and capacity consumption of a setup, by moving an entire lot of product i^* from period t to the closest previous period t_f^* in which product i^* is produced and there is enough idle capacity ($Ca_{t_f^*} > p_i \cdot X_{i^*t}$) to absorb the new lot, where

$$(i^*, t_f^*) = \underset{\substack{(i,t_f)\in\phi,(i,t)\in\phi \\ t_f<t, Ca_{t_f}>p_i\cdot X_{it}}}{\arg\max} [fc_{it} - h_i \cdot X_{it} \cdot (t - t_f)].$$

Note that these operations may destroy feasibility which needs to be recovered at the end.

The fourth step is a forward pass that seeks to reduce inventory holding costs by shifting forward a fraction or an entire lot of product i from a source period t to a target period t_f in which product i is also produced ($X_{it_f} > 0$). This move is similar to that presented by Hindi et al. [2003], but with a different purpose. Our objective is optimality, while theirs is feasibility. The candidate products have currently a positive inventory level ($I_{it} > 0$). In each iteration, the production quantity $Q^t_{i^*t_f^*}$ of product i^* is carried over from period t into period t_f^*, where

$$(i^*, t_f^*) = \underset{i\in[N], t_f\in\{1,\ldots,T-1\}}{\arg\max} \left[h_i \cdot Q^t_{it_f} \cdot (t_f - t) : X_{it_f} > 0\right],$$

$$Q^t_{it_f} = \min\{X_{it}, I_{it}, I_{i(t+1)}, \ldots, I_{i(t_f-1)}, C_{t_f}\}.$$

Finally, the last step looks for improvements in the links between adjacent periods, in a forward pass. The sequencing procedure of the second step may generate a schedule in which the machine is set up to produce product i at the beginning of period $t+1$ ($\alpha_{i(t+1)} = 1$), without producing i in that period ($X_{i(t+1)} = 0$). If there is a product j such that $X_{jt} > 0$, $X_{j(t+1)} > 0$, $\alpha_{j(t+1)} = 0, j \neq i$, then by assigning $\alpha_{j(t+1)} = 1$, $\alpha_{i(t+1)} = 0$ a setup is eliminated in period $t+1$. Let $S = \{j \in [N] : X_{jt} > 0,\, X_{j(t+1)} > 0,\, \alpha_{jt} = 0,\, \alpha_{j(t+2)} = 0\}$ be the set of candidate products. The last two conditions in the definition of S restrict the current analysis to periods t and $t+1$ (otherwise other periods would have to be rescheduled). To select product i^* from S we compute for every $j \in S$ variables ΔI_j and ΔO_j, in the same way as in the sequencing procedure, except that all quantities are computed with respect to c_{ij}. To compute ΔO_j and ΔI_j we only take into account products that are produced in periods t and $t+1$, respectively. The selected product has the minimum aggregated variability $i^* = \arg\min_{j \in S}(\Delta O_j + \Delta I_j)$. Then, both periods t and $t+1$ are rescheduled with the input $\alpha_{i^*(t+1)} = 1$.

From all the steps described before, only the third and the fifth steps may generate unfeasible schedules due to the creation of overtime in a period. Whenever this occurs, we try to recover feasibility by applying the four options of step 2 from the last period with overtime backwards.

4.2.5 Computational Results

All computational experiments in this section were performed on an IBM personal computer with a 3.2 GHz CPU and 1GB of random access memory. CPLEX version 9.0 from ILOG was used as the mixed integer programming solver. We used the instance generator described in Section 4.2.2.1 to generate random instances. For each type of instance, characterized by the quadruple N, T, Cut and θ, we present the average of the values obtained in 10 randomly generated instances.

We first compare the LP relaxations of models F_1 and F_2, both strengthened with the (l, S) and (W_t) valid inequalities. Let F_{1LP} and F_{2LP} denote the values of the linear relaxations of F_1 and F_2, respectively, and let $F^{\star}_{1LP}$ and $F^{\star}_{2LP}$ denote the values of the linear relaxations of F_1 and F_2 including all the (l, S) and W_t cuts, respectively. The separation algorithms for inequalities (4.15) and (4.20) were coded in OPL version 3.7 from ILOG. We solved the minimum cut problem of the disconnected subtour separation algorithm with the Ford-Fulkerson's algorithm. The running times of the linear relaxation of F_2 strengthened with the (l, S) and (W_t) valid inequalities were always less than 150 seconds for all instances. As an example, with $T = 10$ and $N = 25$, model F_1 has 6760 columns and 4135 rows. Figure 4.8 presents the gap $\frac{F^{\star}_{2LP} - F^{\star}_{1LP}}{F^{\star}_{1LP}}$. As proved in Section 4.2.3, formulation F_2 is at least as strong as formulation F_1. The gap increases with the number of time periods.

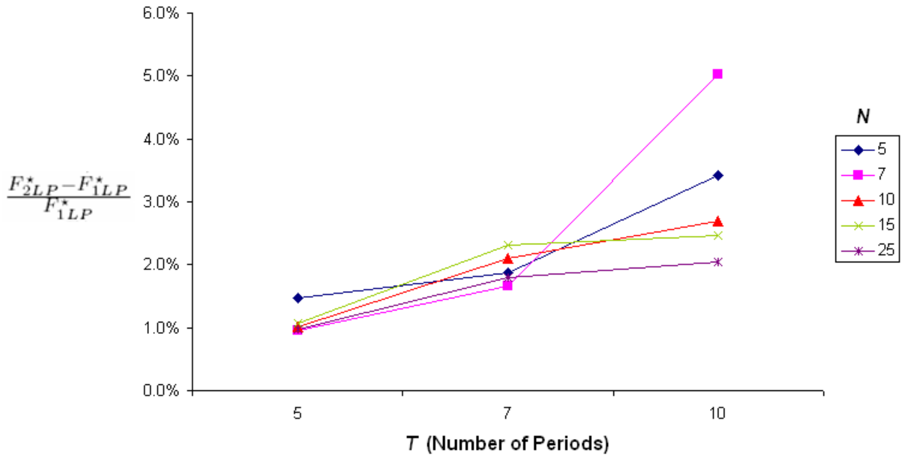


Figure 4.8: Comparison of $F^{\star}_{1LP}$ and $F^{\star}_{2LP}$ for $Cut = 0.6$ and $\theta = 100$

We next determine the impact of the (l, S) and (W_t) inequalities with respect to F_{2LP}. Let $F^{(l,S)}_{2LP}$ be the value of the linear relaxation of F_{2LP} including the (l, S) cuts and let $F^{(W_t)}_{2LP}$ be the value of the linear relaxation of F_{2LP} including the (W_t) cuts. The computational results are displayed in Table 4.8. Both (l, S) and (W_t) inequalities improve the lower bound, with an effect that increases (decreases) with the number of periods (products). Clearly, the improvement provided by the (l, S) cuts is especially noteworthy.

Table 4.9 reports the average deviation of the optimal solution from the lower bound $F^{\star}_{2LP}$, i.e., its integrality gap. As in Section 4.2.2.1, the empty buckets mean that at least one of the instances could not be solved optimally within the 3600 seconds time limit.

It is clear that the quality of the lower bound is significantly improved as the number of products (N) increases, and the quality seems to be independent of the number of periods (T). The results show that, on average, the integrality gap of the instances with $\theta = 50$ is

Table 4.8: Comparison (%) of $F_{2LP}^{(l,S)}$ and $F_{2LP}^{(W_t)}$ with F_{2LP} for $Cut = 0.6$ and $\theta = 100$

	$(F_{2LP}^{(l,S)} - F_{2LP})/F_{2LP}$			$(F_{2LP}^{(W_t)} - F_{2LP})/F_{2LP}$		
	T			T		
N	5	7	10	5	7	10
5	102.1	177.1	276.0	16.1	44.1	86.1
7	96.1	148.8	256.3	3.8	16.4	41.2
10	87.1	142.7	231.9	0.9	3.3	14.3
15	81.5	133.0	204.0	0.4	0.6	2.4
25	79.5	124.5	191.3	0.2	0.2	0.3

Table 4.9: Integrality gap (%) for $Cut = 0.6$

	$\theta = 50$			$\theta = 100$		
	T			T		
N	5	7	10	5	7	10
5	2.7	3.1	2.5	5.6	6.7	5.9
7	1.5	2.1	2.3	4.3	4.5	4.6
10	1.5	0.9	0.9	2.5	3.1	
15	1.0	0.9		1.5		
25	0.8	1.0		0.7		

lower than the integrality gap of the instances with $\theta = 100$. Nevertheless, this difference becomes smaller as the number of products in a problem instance increases. In both cases, from 25 onwards the gap is below 1%.

To generate an upper bound, we implemented the heuristic from Section 4.2.4 in C++ by using Visual Studio 6.0 as the development environment. Tables 4.10 and 4.11 display the minimum, average, and maximum gap of the heuristic solution from the lower bound for $\theta = 50$ and $\theta = 100$, respectively. The running times of the heuristic were always less than 0.4 seconds for all of the instances.

The heuristic found a feasible solution for all problem instances. The flexibility of the

Table 4.10: Gap (%) between the lower and upper bounds for $Cut = 0.6$ and $\theta = 50$

	T		
N	5	7	10
5	2.8/6.4/12.7	2.5/8.3/19.2	2.0/6.4/15.6
7	2.3/6.0/9.9	2.7/7.0/12.7	2.6/6.6/10.3
10	5.6/9.5/18.0	4.7/8.9/13.4	3.3/7.7/14.9
15	4.7/9.7/14.6	4.9/10.0/18.9	6.3/10.1/12.4
25	6.5/9.9/11.9	6.0/11.1/14.9	8.6/12.0/16.3

minimum / average / maximum gap (%)

Table 4.11: Gap (%) between the lower and upper bounds for $Cut = 0.6$ and $\theta = 100$

	T		
N	5	7	10
5	6.8/15.4/22.6	10.3/15.8/18.5	5.1/15.2/26.4
7	4.6/12.9/19.0	6.4/13.5/19.1	9.5/14.5/17.9
10	6.6/13.6/20.1	9.6/13.4/15.5	7.7/13.5/20.7
15	12.9/17.0/20.5	10.4/16.7/21.4	11.9/16.7/23.1
25	20.3/24.0/26.4	21.9/24.1/25.9	20.2/23.6/26.7

minimum / average / maximum gap (%)

four options of step 2 of the heuristic enables us to find feasible solutions, even for tightly capacitated instances. The results indicate that for both $\theta = 50$ and $\theta = 100$ the heuristic performance deteriorates as the number of products increases. This situation is emphasized when the setup costs are considerable higher than the inventory costs ($\theta = 100$). The performance of the heuristic is not influenced by the number of periods. The amplitude of the gap between the lower and upper bounds tends to decrease as the number of product increases. Recall that in our production setting from the glass industry, typically $\theta = 50$ and thus Table 4.10 is more relevant. Based on Table 4.9, we also conclude that the most significant portion of the gap comes from the heuristic (and not from a weak lower bound).

To the best of our knowledge, Gupta and Magnusson [2005] is the only paper in the literature to publish results (gaps between lower and upper bounds) for this problem. However, since inventories and production quantities are expressed in units of capacity (normalized to one), they adapted the parameters of the instance generator used by Haase and Kimms [2000]. By comparing our parameters with the parameters of their generator, their instances are similar to the case with $\theta = 50$ and $Cut = 0.6$. The authors present the performance of their heuristic from $N = 2$ to $N = 15$ for $T = 4$, and from $T = 2$ to $T = 15$ for $N = 4$. For problem instances with 15 products and 4 periods, Gupta and Magnusson [2005] report an average gap between lower and upper bounds of approximately 20%, and for problem instances with 4 products and 15 periods an average gap of approximately 30%. From Table 4.10 it clearly follows that our models and the heuristic are better.

4.2.6 Concluding Remarks

We have presented two novel exact formulations for modeling setup carryover in the challenging CLSP problem with sequence dependent setup times and costs. The models are simpler than others available in the literature. Note that all the models published so far only perform a setup whenever there is sufficient time available to do it entirely. In the case of very large setup times (as the ones observed in the glass container industry that may reach 120 hours), setup operations may start at the end of one period and finish at the beginning of the following period. This new feature has not yet been incorporated in CLSP models. The extension of this work to the case of multiple machines is an interesting area for future research. Jans [2006] looks at how to incorporate parallel machines in a MIP model using commercial optimization software, but the author considers sequence independent setups and no setup carryover. Another extension is to problems with sequence dependent setups that do not satisfy the triangular inequality. Recently, Clark [2006] addresses an animal feed plant production lotsizing and scheduling problem, accounting for holding, backlogging and overtime costs, and incorporating setup times and setup carryovers. The author develops a multi-period model based on the asymmetric traveling salesman problem (ATSP) and implements an ATSP-patching solution approach for relatively small instances (21 products and 4 time periods). The rapid convergence of the algorithm is possible due to the fact that all non-zero setup times yield the same value, therefore more work is this area is necessary to assess this approach.

We have formally shown that one of the formulations is stronger than the other and we have implemented valid inequalities that enable us to reduce the integrality gap of the LP relaxation with a desirable property: the integrality gap decreases as the number of products increases (and it appears to be independent on the number of periods). It is clearly important to find strategies to combine even stronger valid inequalities based on the polyhedral structure of this problem with tighter reformulations. These inequalities should take into account the capacity constraints and the potential cycles and paths of products sequence in each time period.

We have described a five-step heuristic that is effective both in finding a feasible solution and in producing good solutions to these problems (measured by the gap between the lower and upper bounds). The simplicity of this heuristic enables its implementation as a building block of more complex heuristics and metaheuristics. It is worth pointing out that it may effectively generate good initial solutions and repair infeasibilities due to the capacity requirements of solutions generated by any local search procedure.

Chapter 5

Long-term production planning and scheduling: A VNS approach

Summary. This chapter reports a successful application of VNS to the long-term production planning and scheduling problem that arises in the glass container industry. This is a multi-facility production system, where each facility has a set of furnaces where the glass paste is produced in order to meet the demand, being afterwards distributed to a set of parallel molding machines. Since the neighborhoods used are not nested, they are not ordered by increasing sizes, but by means of a new metric developed to measure the distance between any two solutions. Neighborhood sizes decrease significantly throughout the search thus suggesting the use of a scheme in which efficiency is placed over effectiveness in a first step, and the opposite in a second step. We test this variant as well as other two with a real-world problem instance from our case study.

5.1 Introduction

Inspired by a case study, a variant of the Variable Neighborhood Search (VNS) is introduced to tackle the production planning (especially lotsizing) and scheduling problem that arises at the long-term planning level of the glass container industry (Almada-Lobo et al. [2007c]). As we mentioned before, a typical company has several plants equipped with furnaces. Each furnace distributes glass to a set of parallel molding machines. The production planning main constraint is the color of glass melted in the furnace. The goal is to define a 12-month rolling horizon plan that assigns colors to furnaces, schedules color campaigns within each furnace, and assigns products to machines monthly. Due to the very high sequence dependent setup times in color changeovers, color campaigns lotsizing and scheduling have to be done simultaneously. Furthermore, products cannot be aggregated into families, thus increasing the complexity of the problem. Only major setups (multiple family joint setups) are considered, i.e. changeovers between two products sharing the same color are disregarded. The review Karimi et al. [2003] pinpoints the scarce literature devoted to lotsizing and scheduling problems with family joint setups, regarded as an interesting research area to develop heuristics.

Many real world production planning problems are combinatory and multi-objective by nature. Modeling even simplified abstractions of those problems often leads to untractable NP-hard problems. Consequently, several heuristic procedures have been proposed over the years to solve large scale instances. Local search (or neighborhood search) heuristics are improvement algorithms that start with an initial solution and try to find iteratively better solutions in the neighborhood of the incumbent solution. Naturally, both their effectiveness and efficiency are closely related to the neighborhood structures used. Several frameworks have been developed to improve the performance of local search heuristics, avoiding the entrapment in local optima through different search schemes that cross barriers in the solution space typology. Variable Neighborhood Search (VNS) is a recent local search based approach that makes use of systematic changes of the neighborhood structure during the search (Hansen and Mladenovic [2001]). The reader is referred to Jans and Degraeve [2007a] for an up-to-date overview of the existing algorithms for solving dynamic lot-sizing, focusing specially on meta-heuristics.

Throughout the search, neighborhood sizes decrease significantly, thus suggesting the use of a scheme in which efficiency is placed over effectiveness in a first step, and the opposite in a second step. We make use of a new metric to measure the distance between two solutions that allows us to order different neighborhoods. This new VNS scheme combines features of other two, to obtain a compromise between efficiency and effectiveness.

In Section 5.2 we propose a suitable model to formulate the production planning and scheduling problem that arises at a tactical level. In Section 5.3 we explain the solution approach to tackle this problem. Numerical experiments are given in Section 5.4. The chapter ends with a short summary and outlook.

5.2 Mathematical formulation

Let us recall that the production planning is only constrained by the hot area (Figure 3.3) of this semi-continuous manufacturing process. We are dealing with a multi-facility production system, where each facility (plant) has a set of furnaces where the glass paste is produced in order to meet the demand. The glass paste is continuously distributed to a set of parallel molding machines that shape the finished product. Only one color of glass can be produced at any time in each furnace and machines served by the same furnace produce only one color of glass containers at a time. Additionally, there are high sequence dependent setup times involved in color changeovers. Machine idleness is not allowed and, consequently, machines fed by the same furnace must process during the same amount of time. Each machine can only run one product at a time.

At the upper level, the general management objectives are to maximize met demand and facilities' throughput, and to minimize inventory levels. We will disregard the transportation costs, however we can always prevent a certain geographic area from receiving items produced on a plant at a cost of adding extra constraints in the model. We will tackle this multi-objective optimization problem by combining various objectives into a single-one, by means of a weighting function a priori defined by the decision maker.

This level's output is a 12-month rolling horizon plan that assigns colors to furnaces, schedules color campaigns within each furnace, and assigns products to machines monthly (at this level, monthly product demand forecasts are provided). A partial plan for the first month of a furnace was presented in Figure 3.7. Hence, at this level we integrate planning decisions (the number and dimensions of product lots) and scheduling decisions (color campaigns programming). Remember that machine balancing constraints and technology differences between machines (even those of the same furnace) do not allow us to aggregate machines into furnaces.

In order to state the long-term production planning model, the following notation is used.

Indices:

i, j	product: $i \in [N], j \in [N]$
l, u	product color: $l \in [L], u \in [L]$
t	period: $t \in [T]$
y	furnaces: $y \in [Y]$
k	machines: $k \in \left[\sum_{y} \lvert K_y \rvert\right]$

Parameters:

p_{ik}	processing time of product i on machine k (tonnes per day)
η_k	efficiency of machine k
K_y	set of machines fed by furnace y
$\lvert K_y \rvert$	number of machines fed by furnace y
F_l	set of products with color l
D	set of pairs (i, k), such that product i cannot be assigned to machine k
st_{luy}	wasted melted glass to set up furnace y from color l to color u (tonnes)
d_{it}	demand for product i at the end of period t (tonnes)
C_{yt}	capacity of furnace y in period t (tonnes)
M	big number

Decision variables:

X_{ikt}	number of days in which product i is produced on machine k in period t
I^+_{it}	stock of product i at the end of period t
I^-_{it}	backlog of product i at the end of period t

α_{lyt} $\begin{cases} 1, & \text{if the furnace } y \text{ is setup for color } l \text{ at the beginning of period } t \\ 0, & \text{otherwise} \end{cases}$

Y_{it} $\begin{cases} 1, & \text{if a stockout of product } i \text{ occurs at the end of period } t \\ 0, & \text{otherwise} \end{cases}$

T_{luyt} $\begin{cases} 1, & \text{if a setup occurs on furnace } y \text{ from color } l \\ & \text{to color } u(\neq l) \text{ in period } t \\ 0, & \text{otherwise} \end{cases}$

V_{lyt} auxiliary variables that assign color l on furnace y in period t

Z_{lyt} number of days that glass of color l is melted on furnace y in period t

Using the above notation, a mixed linear programming can be formulated as follows.

$$\min \lambda_1 \cdot \sum_l \sum_u \sum_y \sum_t st_{luy} \cdot T_{luyt} + \lambda_2 \cdot \sum_i \sum_t I^+_{it}/T + \lambda_3 \cdot \sum_i \sum_t Y_{it} \tag{5.1}$$

$$I^+_{it} - I^-_{it} = I^+_{i(t-1)} - I^-_{i(t-1)} + \sum_k X_{ikt} \cdot p_{ik} \cdot \eta_k - d_{it} \qquad i \in [N], t \in [T] \tag{5.2}$$

$$I^-_{it} \leq M \cdot Y_{it} \qquad i \in [N], t \in [T] \tag{5.3}$$

$$Y_{it} + Y_{i(t+1)} \leq 1 \qquad i \in [N], t \in [T] \tag{5.4}$$

$$\sum_{k \in K_y} \sum_i X_{ikt} \cdot p_{ik} + \sum_l \sum_u st_{luy} \cdot T_{luyt} \leq C_{yt} \qquad y \in [Y], t \in [T] \tag{5.5}$$

$$\sum_{i \in F_l} X_{ikt} = Z_{lyt} \qquad k \in K_y, l \in [L], y \in [Y], t \in [T] \tag{5.6}$$

$$X_{ikt} \leq M \cdot \left(\sum_u T_{ulyt} + \alpha_{lyt} \right) \qquad i \in F_l, k \in K_y, l \in [L], y \in [Y], t \in [T] \tag{5.7}$$

$$\sum_l \alpha_{lyt} = 1 \qquad y \in [Y], t \in [T] \tag{5.8}$$

$$\alpha_{lyt} + \sum_u T_{ulyt} = \alpha_{ly(t+1)} + \sum_u T_{luyt} \qquad l \in [L], y \in [Y], t \in [T] \tag{5.9}$$

$$V_{uyt} \geq V_{lyt} + N \cdot T_{luyt} - (N-1) - N \cdot \alpha_{lyt} \qquad l \in [L], u \in [L] \setminus \{u\}, y \in [Y], t \in [T] \tag{5.10}$$

$$\sum_{t} X_{ikt} = 0 \qquad (i,k) \in D \tag{5.11}$$

$$(X_{ikt}, I^{+}_{it}, I^{-}_{it}, \alpha_{lyt}, Z_{lyt}) \geq 0, (Y_{it}, T_{luyt}) \in \{0,1\}. \tag{5.12}$$

This model is an extension to Single-Machine Capacitated Lotsizing Problem (CLSP) with sequence dependent setups and setup carryovers presented in Chapter 4 – see also Almada-Lobo et al. [2007a]. The objective function (5.1) is to minimize the weighted sum of sequence dependent setup times, average inventory levels and number of stockouts. Setup times involved in color changeovers are measured in wasted melted glass tonnage. We note that this sub-function tackles the objective of maximizing the facilities throughput. Inventory levels are also expressed in tonnes. As referred before, we are dealing with a multiple objective combinatorial optimization problem. Preferences for objectives are *a priori* declared to form a weighted linear scalarizing function, used to aggregate several objectives into a single one. Each function can be transformed as follows: $F^t_i(x) = \frac{F_i(x) - F^0_i}{F^0_i}$. When F^0_i represents the function minimum, this transformation is referred to a lower-bound approach (Marler and Arora [2005]) and it yields non-dimensional objective function values. Note that the weights λ's define the search directions and are normalized, i.e. $\sum_{j=1}^{3} \lambda_j = 1$.

Constraints (5.2) represent the inventory balances, accounting for backorders. Constraints (5.3) keep track of incurred stockouts. Demand not met in period t must be fulfilled in the following period by constraints (5.4), i.e. backorders last one period at most. We note that stockouts are allowed not only to face scarce capacity situations, but also to maximize production efficiency. In order to prioritize production intentions by customers' importance (as said in Section 3.3.1), backorders may not be allowed for A-type customers.

Constraints (5.5) ensure that the total glass production on each furnace and in each period does not exceed the available melting capacity. Constraints (5.6) force machines fed by the same furnace to process each color for the same amount of time. These constraints along with constraints (5.5) limit the amount of production that can be allocated to a machine. Let T_{ulyt} denote the input setup for color l on furnace y in period t and T_{luyt} the output setup for color l on furnace y in period t. Constraints (5.7)-(5.10) determine the sequence of color runs on each furnace in each period and keep track of the furnace configuration state by determining the color that a furnace is ready to process (color setup carryover information is thereby tracked). Production of glass color l can occur on furnace y in period t if that furnace is set up for l at the beginning of t or if at least one input setup is performed for color l on furnace y. These conditions are guaranteed by constraints (5.7). Constrains (5.8) ensure that α is one for exactly one color in a given period, i.e. ensure a furnace to be set up for a color run at the beginning of each period.

Constraints (5.9) maintain and carry the setup configuration state of the furnace into the next period and balance the network flow. Constraints (5.10) avoid disconnected sub-tours, whilst connected sub-tours are avoided at the optimal solution by the assumption of triangle inequality of color changeover times. Technological constraints (5.11) prevent the assignment of products to some machines. Finally, constraints (5.12) represent the integrality and non-negativity constraints. Note that the integrality constraint on α_{lyt} is not necessary if we assume that at the beginning of the planning horizon the furnace is set up for a color.

The model described earlier has $NT + L^2YT$ binary variables, $NKT + 2NT + 2LYT$ continuous variables and, at least, $L^2YT + NT(3 + K) + T(LK + 2Y)$ constraints, where $K = \sum_y |K_y|$ denotes the total number of machines.

5.3 A solution approach

5.3.1 Variable Neighborhood Search

It is well known that CLSP is a NP-hard problem. Even single-item CLSP has been shown by Bitran et al. [1982] to be NP-hard. Trigeiro et al. [1989] and Maes et al. [1991] have shown that even finding a feasible solution for CLSP with setups is NP-complete. As we will see, the size of an instance from our case study does not allow the model presented in the previous section to be solved until optimality, thus motivating the development of heuristics to solve the original problem.

As said before, VNS is a metaheuristic that systemizes the idea of neighborhood changes to avoid entrapment at a local optimum (Hansen and Mladenovic [2001]). Several VNS variants have been proposed and successively applied to a broad set of combinatorial optimization problems. Variable Neighborhood Descent (VND) is a deterministic best improvement descent method. On the other hand, the basic VNS combines a stochastic (to avoid cycling) shaking phase with a deterministic local descent phase. Its local search step can be replaced by a VND search, obtaining the General VNS (GVNS). The Reduced VNS (RVNS) is a pure stochastic method, in which the descent step of VND is replaced by solutions randomly generated in increasingly far neighborhoods. RVNS aims to increase efficiency at a cost of likely reducing the effectiveness of the search.

The basic VNS is not a trajectory following method, but conducts a systematic search through increasingly distant neighborhoods of the incumbent solution. To efficiently cross barriers or "humps" in the solution space typology it is mandatory to study which set of neighborhood structures must be used in the shaking phase and which set in the local search phase, and how to order those neighborhoods. Typically, neighborhoods are induced on the solution space by some metric (or quasi-metric) function, that defines the distance between any two solutions (by comparing attributes). For instance, s insert-moves may be independently compounded to construct a neighborhood structure s. These compounded neighborhoods are commonly used in permutation problems (e.g. scheduling problems) in which a solution Π' obtained by a single-move sm_1 to incumbent solution Π is reversible, i.e., there exists at least another move sm_2 that when applied to Π' falls back into Π:

$$\Pi' := sm_1(\Pi) \Rightarrow \exists sm_2 : \Pi'' = \Pi := sm_2(\Pi').$$

We call move sm_2 the inverse move of sm_1, i.e. $sm_2 = (sm_1)^{-1}$. The distance between two solutions is easily assessed in permutation problems, typically based on the Hamming distance (the number of elementary changes in 0-1 vectors to turn one solution into another). Most VNS approaches induce their neighborhoods from that distance, in which solutions of the same neighborhood have the same distance from the incumbent solution. When the type of move applied to an incumbent solution has an inverse one, the successive neighborhoods (N_s) obtained by iterating it s times are nested: $N_1(x) \subset N_2(x) \subset \cdots \subset N_{s_{max}}(x)$. To avoid the entrapment in a local optimum, Hansen and Mladenovic [2003b] consider the following desirable condition: $X \subset N_{s_{max}}$, where X is the set of feasible solutions. Nested neighborhoods have increasing sizes, therefore neighborhoods are ranked from the smallest to the largest.

Unfortunately, even the simplest moves applied to the production planning and scheduling problem hereby studied may not have inverse moves and, consequently, the above desirable condition does not hold, i.e. it is not possible to explore X completely with a high-order compounded neighborhood. Here, there is no neighborhood with a pre-determined distance from the incumbent solution, thus increasing the importance of the choice of neighborhoods and the way they are ordered.

5.3.2 Solution representation

As noted in the mathematical formulation, since only major setups (or single color family joint setups) are considered at this level and demand forecasts are discretisized and concentrated at the end of each month, there is no need to detail scheduling, i.e. to sequence products on machines, but only to determine the amounts produced within each color campaign.

Nevertheless, this solution representation would limit the type of moves to be applied to a solution, especially those that affect partial color campaigns because it would increase the complexity in choosing the product lots to be shifted. Thus, we will make use of a representation that sequences product lots on a machine. We note however that the products sequence in each month within each color campaign does not influence the objective function value and that the solution to be presented to the decision maker should only contain monthly product lot sizes per campaign.

5.3.3 Initial Solution

A constructive heuristic to obtain a starting solution is presented in this subsection. This phase deals with one element at a time, which represents a lot of product i to be produced on machine k (as early as possible). A period-by-period heuristic working from period 1 to T will be used.

In each step, we use a priority index that sequences pairs (i, k) on a candidate list to choose product i to be processed on machine k. This index is composed of five factors, namely the machine flexibility, the product flexibility, the machine workload, the color changeover and the production starting date.

Some of those factors depend on the variable $NPotS(i, k)$, that stands for the potential number of days required to produce the gross requirements of the next period of product i on machine k.

The machine flexibility factor ϕm_k of machine k is defined as the number of days to which all items' gross requirements may be assigned on machine k, and is given by $\phi m_k = \sum_i NPotS(i,k)$. The item flexibility factor ϕp_i of product i denotes the total number of days on all machines to which product i may be assigned. Therefore, $\phi p_i = \sum_k NPotS(i,k)$. The current workload (expressed in number of days) of machine k is represented by NS_k. The color changeover factor R_{ik} equals 1 if the assignment of product i on machine k induces a new color changeover, and 0 otherwise. Finally, the earliest starting date ES_{ik} to process product i on machine k cope simultaneously with machine and furnace balancing, and stockouts. Obviously it tends to produce more stocks.

Whenever one pair (i,k) is assigned, the following priority index is computed for each of the items i with (positive) gross requirements on machine k: $w(i,k) = \beta_1 \cdot \phi m_k + \beta_2 \cdot \phi p_i + \beta_3 \cdot NS_k + \beta_4 \cdot R_{ik} + \beta_5 \cdot ES_{ik}$. Notice that this composite dispatching rule is dynamic, i.e., is time dependent. After assigning product i to machine k, the machine's factors are modified. For instance, a pair (i,k) may present, at a time, a higher priority in relation to a pair (i',k') and, afterwards, this rank may be upturned, even if neither is assigned during that interval[1].

This constructive heuristic is given in Algorithm 2.

5.3.4 Neighborhood structures

Given a solution x, the elements of the neighborhood $N(x)$ of x are those solutions that can be obtained by applying an elementary modification (a move) to x. We will present 6 different single moves that are illustrated in Figure 5.2.

Given a set of color campaigns $c = 1,...,C$ and a set of furnaces $y = 1,...,Y$, let $y(c)$

[1]We observed in Chapter 3 that inventory levels cover, on average, over 2 months of demand, triggered by long production runs. Thus, instead of computing gross requirements period-by-period, one may rely on a period aggregation factor Δ to calculate them. On each iteration Ω, each product gross requirements from $t = (\Omega - 1) \cdot \Delta + 1$ to $t = \Omega \cdot \Delta$ would be determined. Note that a higher aggregation factor Δ reduces the number of iterations (and running times), increases lots dimension, tends to reduce the number of color campaigns at a cost of increasing stocks level and, finally, hardens the generation of feasible solutions.

```
input instance()
set $t'$ to 1
repeat
    calculate each product gross requirements for $t = t'$
    calculate $NPotS(i,k)$ for $t = t'$
    while $\exists(i,k) : NPotS(i,k) > 0$
        for $i = 1$ to $N$
            for $k = 1$ to $K$
                if $NPotS(i,k) > 0$ and $(i,k) \notin D$ then
                    calculate priority index $w(i,k)$
                end if
            end for
        end for
        rank candidate pairs (i,k) according to increasing values of $w(i,k)$
        select candidate pair $(i',k')$ ranked first
        assign product $i'$ to machine $k'$ in the earliest possible day
        update product i' gross requirements for $t = t'$
        update $NPotS(i',k)\forall k$ for $t = t'$
    end while
until $t' = T$
```

Algorithm 2: Pseudo-code to generate an initial solution

denote the furnace on which color campaign c is currently assigned. Insert moves and pairwise interchanges (called swaps) are two of the most frequently used move operators in permutation problems. Typically, an insert move ($Insertion[c,d]$) picks up color campaign c from furnace $y(c)$ and inserts it into a new position that immediately precedes the location of color campaign d in furnace $y(d)$ (this color campaign and following ones on this furnace have to be postponed). In addition, $Insertion[c, y(\cdot)]$ removes c from furnace $y(c)$ and inserts it at the end of furnace y. After a move, lots are anticipated as much as possible to cope with machine balancing constraints. If, after a move, two campaigns of the same color are adjacent in a furnace, both are merged. Naturally, this procedure will reduce the number of campaigns on a solution. In fact, with an insertion move, the total number of color campaigns C will either remain constant or be reduced to at most two color campaigns. Figure 5.1a) plots the size of the insertion neighborhood as a function of the number of color campaigns C of an incumbent solution for $Y = 8$ (the number of furnaces of our case study). Note that the insertion neighborhood size (denoted as $|NS_{Insertion}|$) is given by $C^2 + (Y - 3) \cdot C + Y$, $\forall Y \geq 1, C \geq 1$.

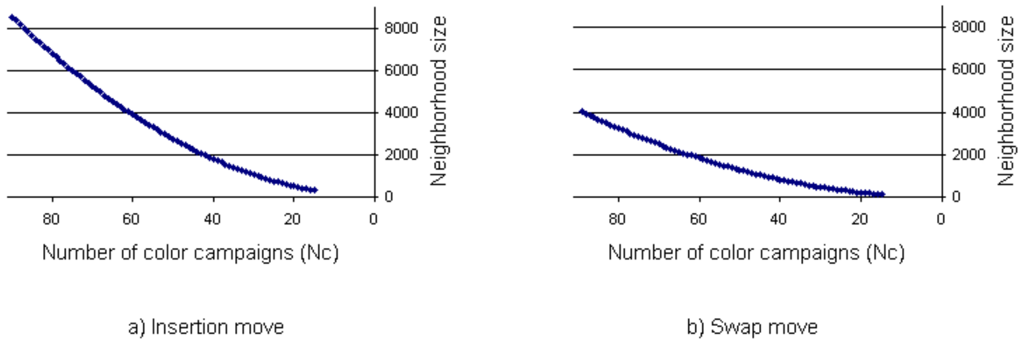


Figure 5.1: Insertion and Swap Neighborhood sizes with 8 furnaces

When a color campaign is shifted from a furnace into another, its lots have to be assigned to new machines. Lots are picked up by increasing starting dates, and are scheduled in the first compatible available machine. If a lot cannot be assigned to any of the target furnace's machines, then the move doesn't produce any feasible solution. Insertion moves may induce a reparation mechanism whenever production on the target furnace exceeds its aggregated capacity. The production surplus is shifted back to the end of the source furnace planning horizon. This mechanism may create by itself a new color campaign. Therefore, the simple insert move does not have, in this problem, an inverse move.

A swap move $Swap[c, d]$, on the other hand, exchanges two color campaigns in different positions, and can be considered as a combined two insert moves. The merger mechanism may reduce C by at most four color campaigns. The size of the swap neighborhood is $C \cdot (C - 1)/2$. A swap move may also require a reparation mechanism, however not so often as an insert move.

A transpose move $Transpose[c]$ interchanges two adjacent color campaigns. Therefore, the transpose neighborhood is a subset of the insert and swap neighborhoods, with size

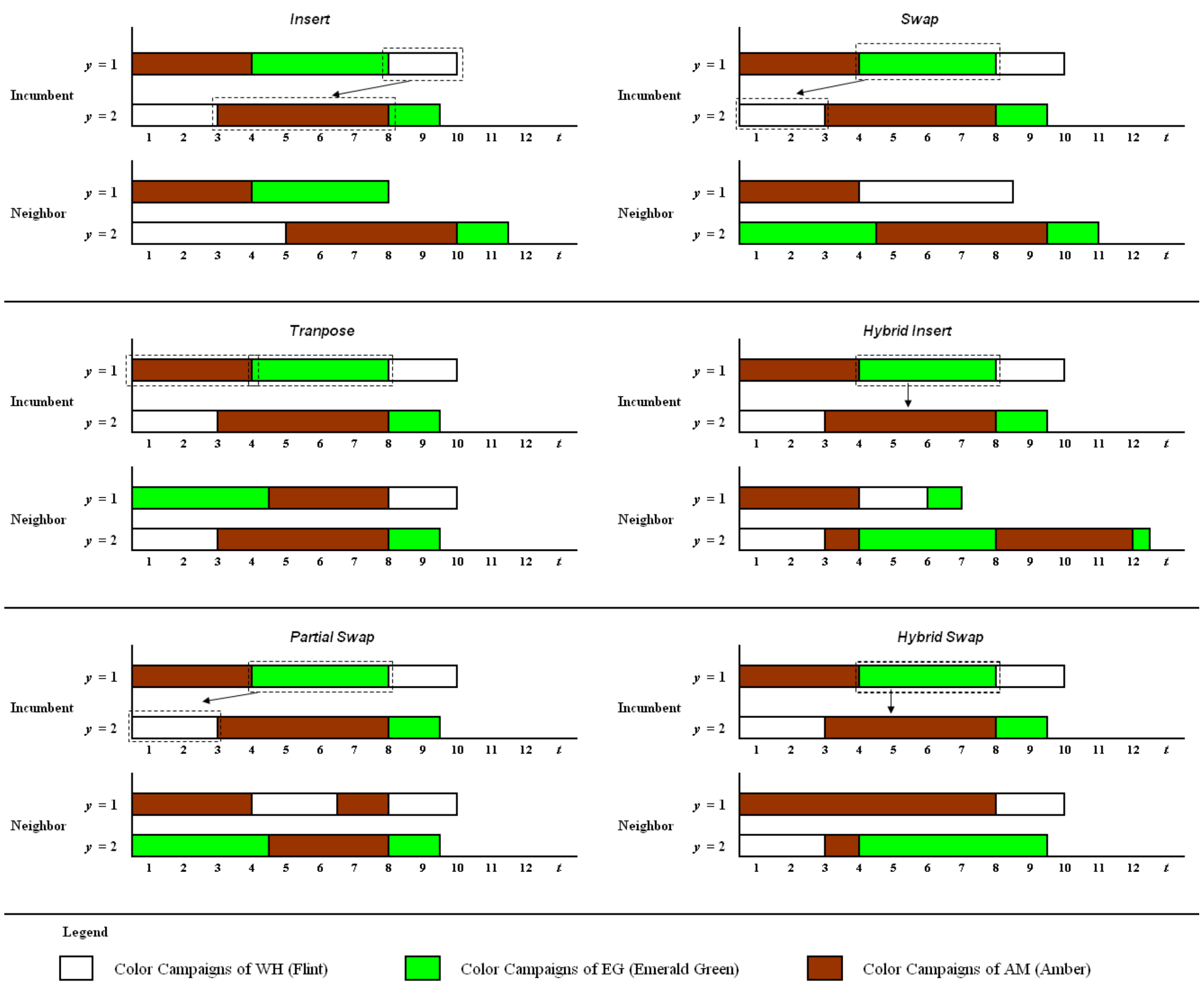

Insert
Swap
Tranpose
Hybrid Insert
Partial Swap
Hybrid Swap
Incumbent
Neighbor
y = 1
y = 2
1 2 3 4 5 6 7 8 9 10 11 12 t
Legend
Color Campaigns of WH (Flint)
Color Campaigns of EG (Emerald Green)
Color Campaigns of AM (Amber)

Figure 5.2: Illustrating different neighborhood structures

$N - 1$.

One of the weaknesses of the aforementioned transpose, insert and swap moves arises from the fact that they don't have inverse moves (that undo the previous ones). For instance, after an insert or swap move being accepted two color campaigns may be indefinitely merged, thus reducing the future search flexibility. Furthermore, without the reparation mechanism, they do not allow a color campaign split, not allowing strong perturbations to an incumbent solution. When the incumbent solution is already good (typically entails a plan with few and large color campaigns due to the high sequence dependent setup times involved in a color changeover), pure insert and swap neighborhoods are unlikely to obtain an improved solution, mainly because of demand requirements. For instance, a three-period length color campaign may not have enough freedom to be shifted to other location (i.e., trapped in a position).

New neighborhood structures can be induced by compounding a multiple s-insertion or s-swap independent moves, as conducted by Congram et al. [2002]. Again, these kind of moves may work in pure permutation problems. Despite (in theory) changing a high number of solution attributes, they may not allow to jump out from good local optima without the help of reparation mechanisms.

Therefore, we propose other possible moves that enable diversification strategies to reach valleys that are not so far away from the incumbent solution, which can be explored afterwards by an intensification strategy.

The following moves, namely hybrid insertion, partial swap and hybrid swap, define small-size neighborhoods, that are very effective considering the goal they were developed for, i.e. to perturb an incumbent solution. Contrarily to the previous ones, these moves break color campaigns, (temporarily) increasing the total number of color campaigns. The hybrid insertion move $HybInsertion[c, y]$ removes color campaign c from furnace $y(c)$ and inserts into another one ($\neq y(c)$), preserving its starting date. The overlapped target furnace lots from one or more color campaigns are postponed. $|NS_{HybInsertion}|$ equals to $C \cdot (Y - 1)$. Even without the reparation mechanism, this move splits one color campaign of the target furnace (in the most likely case that there isn't any target furnace's color campaign with the same starting date). Note that the example given in Figure 5.2 for the $HybInsertion[c, y]$ induced the reparation mechanism. After moving a campaign from $y = 1$ to $y = 2$, the surplus of the latter (being part of another Emerald Green color campaign) is shifted back to furnace $y = 1$.

The partial swap move $PartialSwap[c, d]$ picks up color campaign c and inserts it into a new position that immediately precedes the location of color campaign d from a different furnace. The overlapped lots are shifted to the source furnace into the location of the removed color campaign c. The reparation mechanism is not required since the workload of both furnaces remains constant. Note that one or more entire color campaigns and/or at most one partial color campaign are transferred into the source furnace. This neighborhood structure has $C^2 - \sum_{k=1}^{Y} C_k^2$ neighbors, where C_k denotes the current number of color campaigns of furnace k. Hence, $|NS_{PartialSwap}| \leq \frac{C^2 \cdot (Y-1)}{Y}$. Unlike the swap move, *PartialSwap* is not symmetric, i.e., given an incumbent solution Π and two neighbors Π' and Π'':

$$\Pi' := PartialSwap[c,d](\Pi) \neq \Pi'' := PartialSwap[d,c](\Pi)$$

Partial swap perturbation to a solution modifies a small set of components and avoids cycling. Even with other moves it is very unlikely to fall back in the incumbent solution: therefore this move can be considered irreversible.

Finally, $HybridSwap[c,y]$ moves color campaign c from furnace $y(c)$ to a different one, preserving its starting date. However, contrarily to the $HybInsertion[c,y]$, in this case the overlapped target furnace lots from one or more color campaigns are shifted to the source furnace. The *Hybrid Swap* size, $|NS_{HybSwap}|$, equals $C \cdot (Y-1)$.

After each of these moves, a smoothing procedure is applied at the end of the modified color campaigns, to reduce machine's idleness within each color campaign. If possible, lots or partial lots are shifted from machines with higher loads to machines with lower loads.

Since the neighborhoods induced by the above moves are not nested, the size of the neighborhoods does not determine the sequence in which they should be used. We believe that under this condition, one has to analyze the effects of each move type to order them. This is why, unsurprisingly, applying insertion before swap neighborhoods may be better than the opposite, despite the fact that the size of the former is larger than the latter's, as reported in Besten and Stutzle [2001] for various single machine scheduling problems.

Figure 5.3 summarizes the main effects of each move type. Missing entries indicate that the corresponding effect is not achieved by the move. An entry with the symbol "?" means that the corresponding effect may be obtained by the move, but not always. It is clear that *Transpose* and *Insertion* are suitable for intensification of the search and *HybSwap* for diversification.

In order to complete the sequence, we next define a way of measuring the distance between any two solutions. Figure 5.4 illustrates a neighbor obtained from an $HybridSwap[EG_1, y = 2]$ move applied to a starting solution.

MAIN EFFECTS	Transpose	Insertion	Swap	HybInsertion	PartialSwap	HybSwap
Breaks a color campaign on one of the furnaces				✓	✓	✓
Changes the number and the length of color campaigns	?	?	?	✓	✓	✓
Demands a reparation mechanism		?	?	?		
Interaction between furnaces			✓		✓	✓
Number of furnaces in which the assignment color campaign/furnace changes	1	1	2	1	2	2
Preserves the original starting date of color campaigns				✓		✓

Figure 5.3: Effects of different types of moves

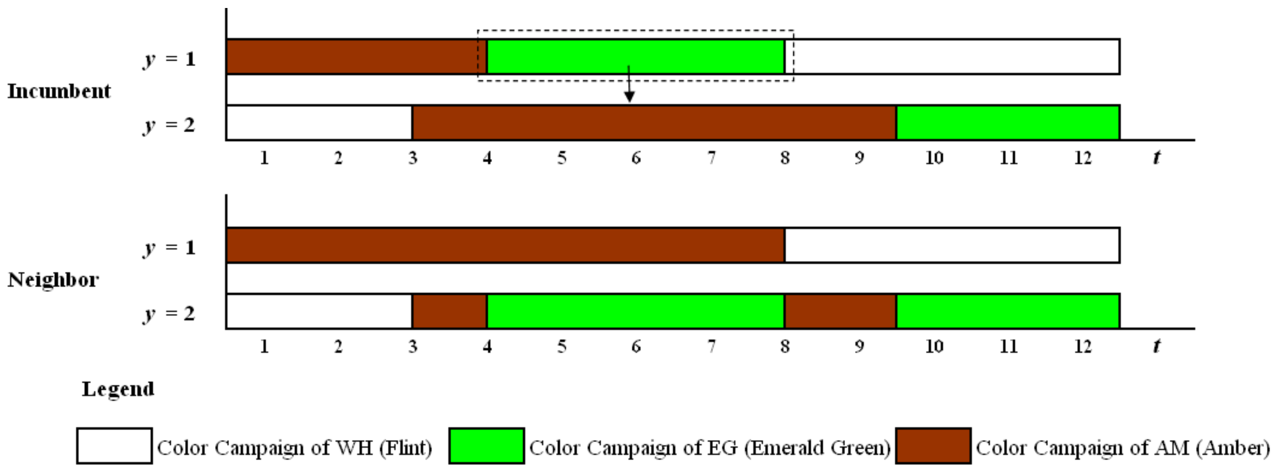

Incumbent
y = 1
y = 2
1 2 3 4 5 6 7 8 9 10 11 12 t
Neighbor
y = 1
y = 2
1 2 3 4 5 6 7 8 9 10 11 12 t
Legend
Color Campaign of WH (Flint)
Color Campaign of EG (Emerald Green)
Color Campaign of AM (Amber)

Figure 5.4: Example of an *Hybrid Swap* move

We measure the distance between two solutions by matching color campaigns furnace by furnace. When both solutions are compared for furnace $y = 1$ (see Figure 5.5), one may intuitively match together both color campaigns AM (amber) and both color campaigns WH (flint). Therefore, only color campaign EG (emerald green) of the incumbent solution remains unmatched. We value each unmatched campaign by one unit, and each matched pair of campaigns with different lengths by half a unit.

For instance, the distance between the two solutions of Figure 5.5 is four units (1.5 for

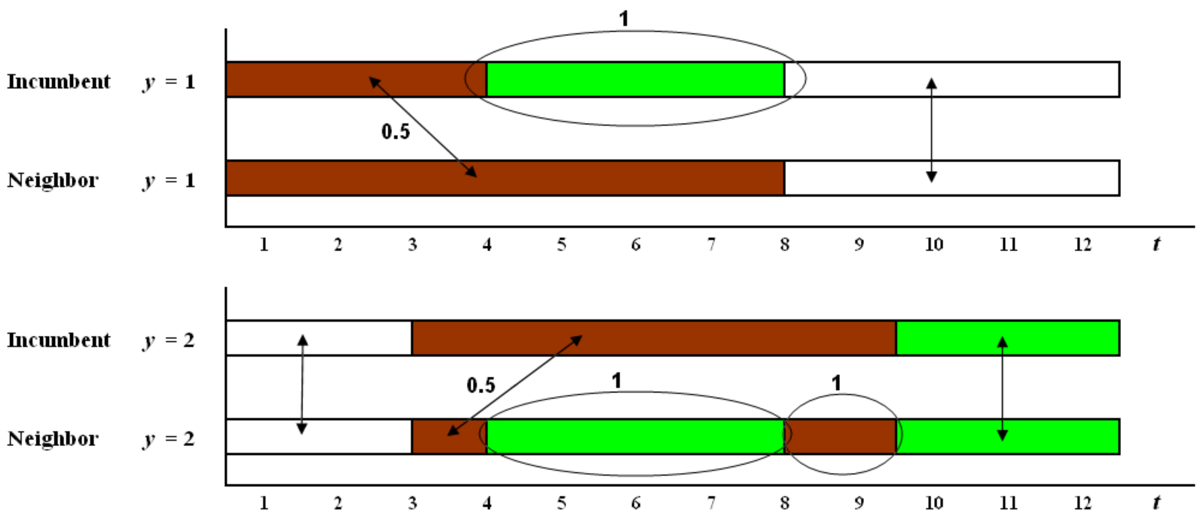


Figure 5.5: Distance between two solutions

$y = 1$ and 2.5 for $y = 2$).

We generated a high number of neighbors by applying the moves presented earlier to different incumbent solutions of our case study. Table 5.1 presents the mean and coefficient of variation of the distance produced by each move.

Table 5.1: Average distance from the incumbent solution of neighbors obtained through different moves

	Moves	$\overline{x}$	*c.v.*
T	*Transpose*	1.8	0.15
I	*Insertion*	2.2	0.25
PS	*Partial Swap*	3.2	0.45
S	*Swap*	3.3	0.60
HI	*Hybrid Insertion*	3.4	0.33
HS	*Hybrid Swap*	3.5	0.55

c.v. = coefficient of variance

We note that if technological constraints had been relaxed (if products could be assigned to any of the furnaces), both *Partial Swap* and *Hybrid Swap* moves would have scored higher.

5.3.5 Objective function

The objective function has already been presented in Section 2.2. However, the hard machine balancing constraints (4.8) are relaxed in order to increase search flexibility. Moreover, the output of this level should allow the decision maker to supervise the adequacy of the company set of equipments (number and type of machines, and furnaces) to face the forecasted demand. Such adequacy can be measured by the number of idle days in a production plan. In practice, such imbalances are solved by an iterative negotiation with the commercial department to 'seek' more demand for products that can be produced in the emptier machines, or, more unlikely, to reduce demand forecast figures for products that are produced in the most heavily loaded machines. The production planner may also load the idle days with high rotation products to be stocked.

Hence, constraints (4.8) are relaxed and the number of idle days is penalized in the objective function. However, instead of also aggregating this component in the objective, a two-level hierarchical objective function will be used. The first level aims at minimizing the maximum idle time per color campaign. The second level attempts at minimizing the aggregate function described earlier. Despite appearing to be counterintuitive, the assessment of machine idleness in a first place works extremely well and provides better results than other scenarios.

5.3.6 VNS design

In this section we present a new variant of Variable Neighborhood Search, that results from combining RVNS with the standard VNS.

As mentioned before, the basic VNS combines stochastic and deterministic neighborhood changes. Despite providing very good results for a large set of combinatorial optimization problems, it becomes less attractive for larger instances (Hansen and Mladenovic [2003a]).

The RVNS variant is suitable to solve problems with large instances, for which the time consuming local search routine is prohibitive (or, at least, not desirable). Neighbors are randomly selected from neighborhoods increasingly far away from the incumbent solution. It is likely that the efficiency gain of RVNS comes together with an effectiveness loss. Nevertheless, good results for RVNS with moderate running times when compared to the basic VNS are reported in Hansen and Mladenovic [2001].

A combination of RVNS and VNS may provide efficient and good results for the problem presented. In fact, since the initial solution is constructed with a myopic dispatching rule (in a sense that it tackles gross requirements period by period), it contains a high number of color campaigns. Thus, at the beginning of the search, local search procedures are very costly due to the neighborhood sizes. Therefore, starting the search with RVNS is likely to reduce very quickly the neighborhood dimensions. After some iterations, RVNS has difficulty in obtaining better neighboring solutions. At this point, the reduction of the number of color campaigns enables us to apply schemes that explore the whole neighborhood structure. We will make use of the standard VNS in this second phase.

In order to maintain the simplicity of the VNS principles, both RVNS and the shaking step of VNS make use of the same set of neighborhood structures. The local search step of VNS is applied with the Insertion neighborhood. RVNS/VNS steps are presented in Algorithm 3.

```
initialization.
select the set of neighborhood structures to be used in the shaking phases of RVNS and VNS;
construct an initial solution;
choose two stopping conditions;
(1) while (stopping condition 1 not met) RVNS;
(2) while (stopping condition 2 not met) VNS;
```

Algorithm 3: Steps of the variant RVNS/VNS

This new scheme has only one more parameter than the basic VNS, namely the second stopping condition.

In the next section we benchmark the RVNS/VNS variant against the basic VNS and RVNS.

5.4 Results

In this section we present results for a real glass container manufacturer instance. The company has four plants and a total of eight furnaces. Furnaces can feed from two to five machines, to a total of 26 forming machines (with different industrial characteristics). Sales turn on 515 finished products, of nine different colors. Real data of demand, machines, furnaces, initial inventory levels, color changeover times, products and technological constraints are available in Instance [2005]. We note that for this instance, the optimization model described earlier would have 13956 0/1 variables, 174768 continuous variables and 189996 constraints.

Three VNS variants were tested: the basics VNS and RVNS, and the new variant RVNS/VNS. The *Insertion* neighborhood was used for the local search sub-routine of both VNS and RVNS/VNS. Different sets and sequences of neighborhood structures were used for the shaking phases. In all cases, we use the maximum number of iterations between two improvements as stopping criterion: 10 to basic VNS, to VNS sub-routine of RVNS/VNS and to RVNS sub-routine of RVNS/VNS; 30 to basic RVNS.

The weighted linear scalarizing function (f) was used with the following parameters: ($\lambda_1 = 0.8; F_1^0 = 4000$), ($\lambda_2 = 0.1; F_2^0 = 104000$) and ($\lambda_3 = 0.1; F_3^0 = 40$), for setups, inventory levels and stockouts, respectively. This set of values was based on the company's knowhow and feeling.

The VNS variants were run for three very different types of initial solutions. Table 5.2 illustrates the weights of the factors of the composite dispatching rule described in Section 3.3, that allow us to compute the starting solutions SS_1, SS_2 and SS_3. The quality of each starting solution is also represented in the same table, namely the maximum number of idle days (NId), the objective function value (f) and the number of color campaigns (C). It is clear that SS_1 outperforms SS_2, whilst SS_2 is better than SS_3. Figure 5.6 illustrates part of production plan of the starting solution SS_1. We note that the maximum number of idle days ($NId = 20$) occurs on the first color campaign of the furnace MGA due to a production imbalance on the second machine. Part of the starting solution SS_3 is provided in Figure C.1 in the Appendix.

Table 5.2: Characteristics of three different starting solutions

	β_1	β_2	β_3	β_4	β_5	NId	f	C
SS_1	0	0	1	$NS_k + ES_{ik}$	1	20	3.68	58
SS_2	0	0	1	NS_k	1	41	5.16	67
SS_3	0	0	0	14	2	49	5.90	76

We implemented the algorithm in C++ and run the experiments on a IBM Machine with a 3.2 GHz processor and 1Gb of memory. Tables 5.3-5.5 show the results of the computational experiments for starting solutions SS_1, SS_2 and SS_3, respectively. Six sets of neighborhoods were tested: in the first two, neighborhood structures are ordered by the distance function mentioned earlier (both differ only on the transpose neighborhood). The third and fourth sets repeat the same neighborhoods of the first two, but ordered by their cardinality. Finally, the last two contain the traditional $2-Insertion$ and $3-Insertion$ sets of neighborhoods, respectively.

Ten runs were executed for each configuration: VNS variant, set of neighborhoods and initial solution. The average computation time (CPU seconds), denoted as t_{avg}, the average maximum number of idle days (NId_{avg}) and the average objective function value (f_{avg}) are presented. Figure 5.7 depicts an example of a solution obtained with the configuration: RVNS/VNS variant, set of neighborhoods {I;PS;S;HI;HS} and initial solution SS_1. This solution has 20 color campaigns, 49 stockouts, 6798 tonnes of setups and 107800 tonnes of average inventory levels, with $NId = 3$ and $f = 0.59$.

The performance of the algorithms depends on the initial solution: the best results were obtained for the best initial solution SS_1 (Table 5.3), however, results for the worst starting solution SS_3 (Table 5.5) outperform in many cases the ones obtained for SS_2 (Table 5.4). It seems that the value of f is positively correlated with the value of Nid for the same VNS variant and initial solution. As expected, the computation times increase for problems with worse starting solutions (enhanced by the fact that neighborhood sizes decrease throughout the search).

Regarding the neighborhood sets, sequencing neighborhood structures by the new distance function (first two rows of each table) achieves improved solution quality when com-

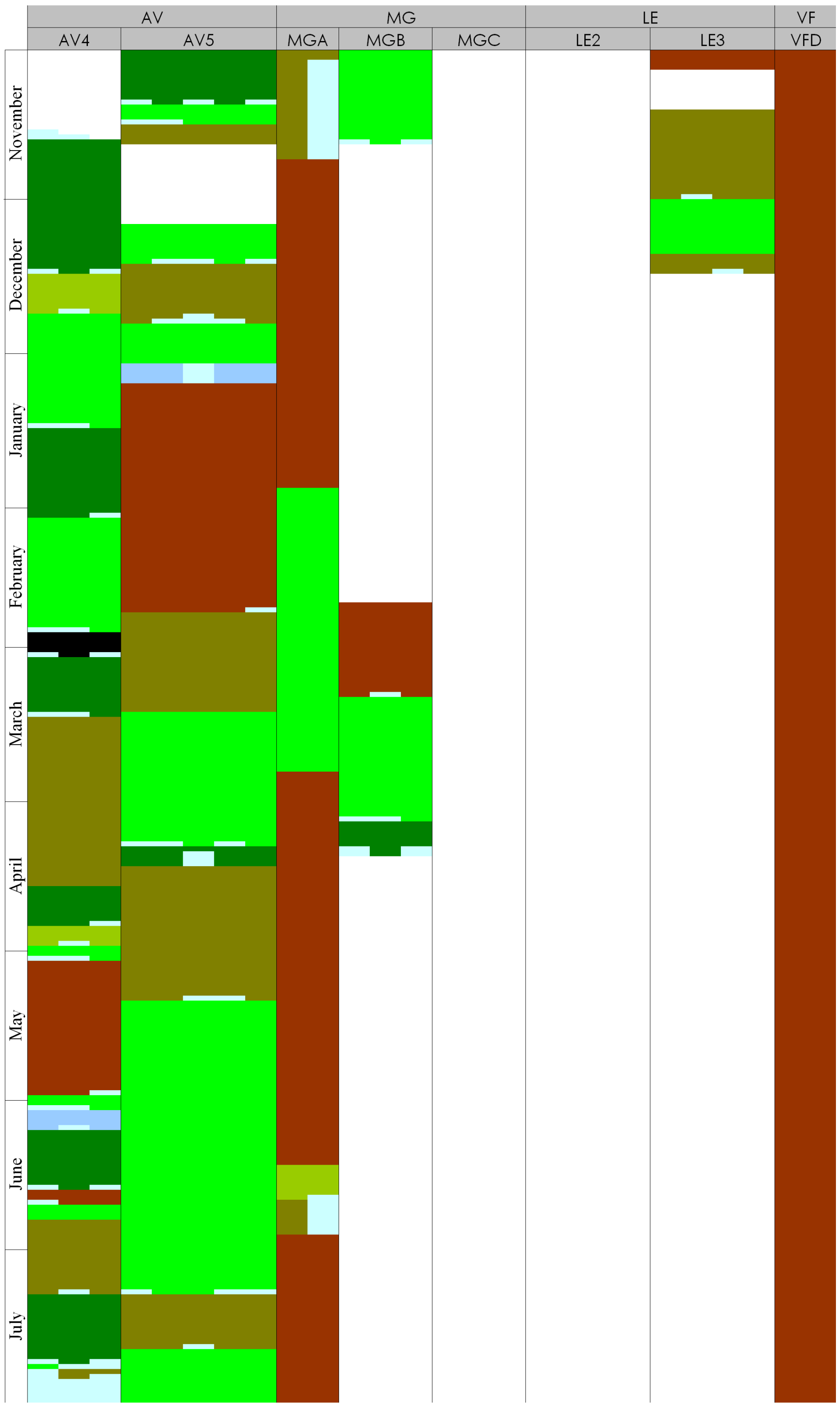


Figure 5.6: Part (9 first months) of the Initial Solution SS_1

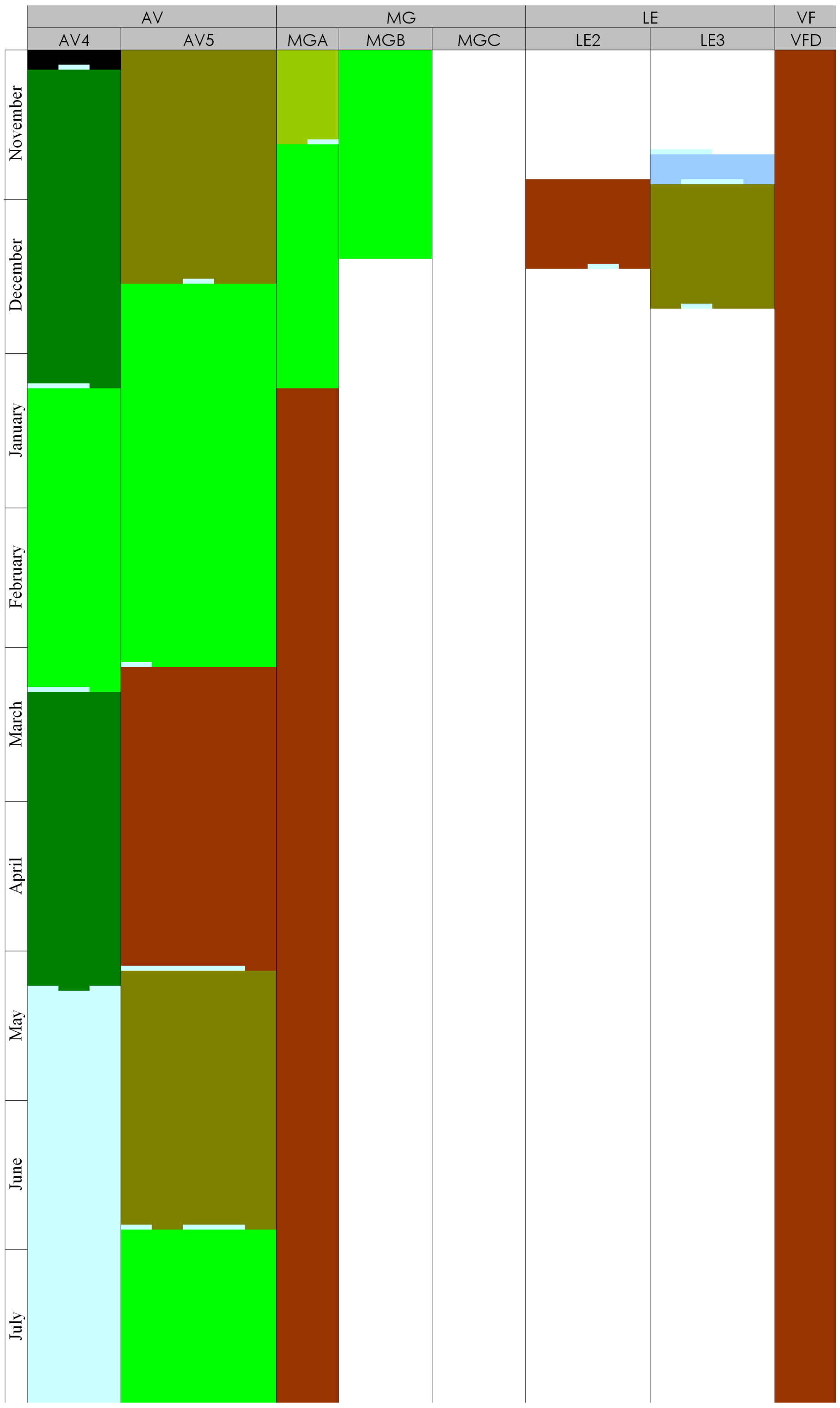


Figure 5.7: Part (9 first months) of the Final Solution SS_1

Table 5.3: Average results for starting solution SS_1

		RVNS			VNS			RVNS/VNS		
	NS	NId_{avg}	f_{avg}	t_{avg}	NId_{avg}	f_{avg}	t_{avg}	NId_{avg}	f_{avg}	t_{avg}
1	{I;PS;S;HI;HS}	3.8	0.90	206	3.7	0.48	882	3.4	0.83	712
2	{T;I;PS;S;HI;HS}	4.0	0.90	226	3.8	0.51	850	3.3	0.84	750
3	{HI;HS;PS;S;I}	3.9	1.01	265	3.9	0.68	704	3.4	0.89	808
4	{T;HI;HS;PS;S;I}	3.4	1.03	259	3.5	0.55	724	3.3	0.82	832
5	{I;2I}	4.0	1.06	267	3.9	0.75	405	3.8	1.00	393
6	{I;2I;3I}	3.7	0.97	1078	4.0	0.74	538	3.5	0.96	811

Table 5.4: Average results for starting solution SS_2

		RVNS			VNS			RVNS/VNS		
	NS	NId_{avg}	f_{avg}	t_{avg}	NId_{avg}	f_{avg}	t_{avg}	NId_{avg}	f_{avg}	t_{avg}
1	{I;PS;S;HI;HS}	22.8	1.26	282	5.4	1.10	836	22.0	1.23	757
2	{T;I;PS;S;HI;HS}	19.9	1.39	269	3.6	1.10	750	23.0	1.21	1017
3	{HI;HS;PS;S;I}	19.3	1.41	343	6.5	1.11	1066	19.6	1.24	776
4	{T;HI;HS;PS;S;I}	19.8	1.37	274	4.0	1.12	1036	23.8	1.22	991
5	{I;2I}	27.8	1.41	255	4.7	1.18	514	25.7	1.38	457
6	{I;2I;3I}	25.8	1.36	1210	5.5	1.18	864	25.1	1.32	971

Table 5.5: Average results for starting solution SS_3

		RVNS			VNS			RVNS/VNS		
	NS	NId_{avg}	f_{avg}	t_{avg}	NId_{avg}	f_{avg}	t_{avg}	NId_{avg}	f_{avg}	t_{avg}
1	{I;PS;S;HI;HS}	26.0	1.28	308	14.0	1.04	1020	24.8	1.23	1017
2	{T;I;PS;S;HI;HS}	23.8	1.24	270	5.2	1.03	1561	22.5	1.05	1182
3	{HI;HS;PS;S;I}	16.3	1.68	335	4.9	1.06	1243	10.6	1.05	1135
4	{T;HI;HS;PS;S;I}	19.9	1.42	331	10.6	1.05	1259	18.1	1.01	1093
5	{I;2I}	37.8	1.59	474	10.7	1.20	708	33.7	1.35	655
6	{I;2I;3I}	32.8	1.48	1283	13.1	1.23	1094	28.3	1.39	1405

pared to the standard procedure of sequencing neighborhoods by their sizes (third and fourth rows of each table). The use of the *Transpose* neighborhood does not influence the performance of the algorithms. We note that this neighborhood is often successfully used in scheduling problems, but within the local search procedure in order to reduce computation times (since it is the smallest neighborhood). As expected due to the specificities of this environment, the $s-Insertion$ sets do not perform well (fifth and sixth rows of each table), and it is not clear whether here the $2-Insertion$ set is better than the $3-Insertion$ set, as generally acknowledged for several problems.

Finally, the basic VNS is always superior to RVNS, and almost always to RVNS/VNS. RVNS/VNS seems to be a compromise between VNS and RVNS.

The best solution found has 17 color campaigns, 57 stockouts, 5262 tonnes of setups and 107000 tonnes of average inventory levels, with $NId = 3$ and $f = 0.30$. Unfortunately, we were not able to benchmark our results against the ones obtained by our case study. At the time the real plan was built up, the number of stockouts and average inventory levels were not estimated by the company for the entire 12-month horizon. One of the main motivations of the company to run this research project was exactly to include the number of stockouts and the average inventory levels among the criteria used to build and evaluate the production plans.

The fact that we are deriving long term plans might reduce the importance of the computation times. Typically, the planner receives every week both updated forecasts from the commercial department and potential demands for some (new) products that he must simulate to assess their operational impact. However, in order to cope with the management rules of the different production sites, the planner has to simulate many scenarios, each one based on different assumptions and, therefore, making the computation times play an important role. We then perform a comparison between the basic VNS and the variant RVNS/VNS for the same computation times, namely 400 and 500 seconds, using the neighborhood sequence {I;PS;S;HI;HS}. Table 5.6 reports the average of the solution quality obtained in ten runs for each configuration: initial solution, computation time and VNS variant.

It is clear that as the quality of the initial solution worsens and/or the available time decreases, the variant RVNS/VNS becomes more attractive and interesting than the basic

Table 5.6: Average results for different computation times

		VNS		RVNS/VNS	
SS	t_{avg}	NId_{avg}	f_{avg}	NId_{avg}	f_{avg}
SS_1	400	3.6	0.79	3.4	0.89
SS_1	500	3.8	0.68	3.6	0.92
SS_2	400	14.6	1.34	20.3	1.26
SS_2	500	3.8	1.17	24.1	1.24
SS_3	400	23.6	1.65	19.6	1.21
SS_3	500	9.7	1.29	26.0	1.27

VNS.

5.5 Validation and integration of production plans

The objective of this section is twofold. First, we explain how the results of the aforementioned algorithm were validated by the company, illustrating it with several examples. Then we provide an overview on the main blocks of SAP R/3 production planning module (SAP-PP) and highlight the integration of SAP-PP and the system where the VNS algorithm will be allocated (hereafter referred as APS–advanced planning system).

Validation

The production plans suggested by APS were scrutinized and compared with real-life plans (RLPlan) by PPD members of the company. Over a dozen of real-world instances, from February 2005 to April 2007, were analyzed. In order to validate the outputs, several (soft) rules at the time used by PPD to make the act of planning easier were considered, such as the prohibition of assigning some colors to a set of furnaces, or to force the allocation of a set of containers (as those tailored to bottle olive oil) to a machine. Both the number and extension of color campaigns of APS plan and RLPlan were benchmarked and the reasonability of the assignment of products on machines was taken into account. We now go through four of those instances to show the reader additional APS fringe benefits that can be reaped by the company. Figure C.2 (Appendix) presents RLPlan (on the left) and APS plan (on the right) developed in April 2007. Both production plans have the same number of color campaigns. However, APS plan allows one to question the workload imbalance on machine 3 of furnace AV5 and suggests a flint campaign between georgia green and cobalt blue campaigns on furnace LE2[2]. For another instance, analyzed in June 2006, APS plan of Figure C.3 discloses a georgia green campaign on furnace LE2, disregarded at the time by PPD. It is worth of note the importance of freezing the first two months while performing a rolling horizon procedure (the first color campaign on AV4 differs from RLPlan to APS plan). The last case stresses the role of APS in balancing sales with operations throughout the sales budgeting process and to evaluate scenarios for which the planner cannot rely on past data. Figure C.4 illustrates the APS plan of October 2006. The planner had to face different challenging production settings: a new furnace (AV2) would start working on January 2007 and furnaces MGA e MGB were planned to be rebuilt at the end of 2007. In addition, the hypothesis (raised by the executive board) of dedicating machine 3 of AV5 to produce containers of a major client, resulted in considering (for the first time) an emerald green campaign in furnace MGB. APS also alerted for the impossibility of accomplishing the initial budget (triggered by the assignment of an unexpected flint color campaign on furnace MGB - see Figure C.4), and supported PPD to reduce the flint and amber aggregate forecasted sales in 7500 and 3700 tonnes, respectively, avoiding unnecessary future stockouts (see Figure C.5).

Finally, in order to balance as much as possible the machine workload, APS supports each time the planner with new possibilities of product assignments on machines, as we testified in two years of testing.

Integration

The ERP system implemented in the company contained insufficient support for improvement of production planning tasks. The project aimed to develop algorithms to be embedded in APS interfaced with the planning module of SAP. It is clear from Figure 5.9 that APS

[2]This fact is currently being analyzed by the company since the data we were given implies a violation of changeovers triangular inequality, which is denied by elements of the Fusion department.

works almost as a “black box”, using data from the company global information system database and feeding back the results to SAP-PP. Implementing APS with ERP system results in a number of required interfaces. The process of creating a functional architecture and the respective interfaces between the two systems are not at stake here.

SAP-PP main blocks are depicted in Figure 5.8, in which dotted lines blocks are influenced by the results of APS. Information flows through text files.

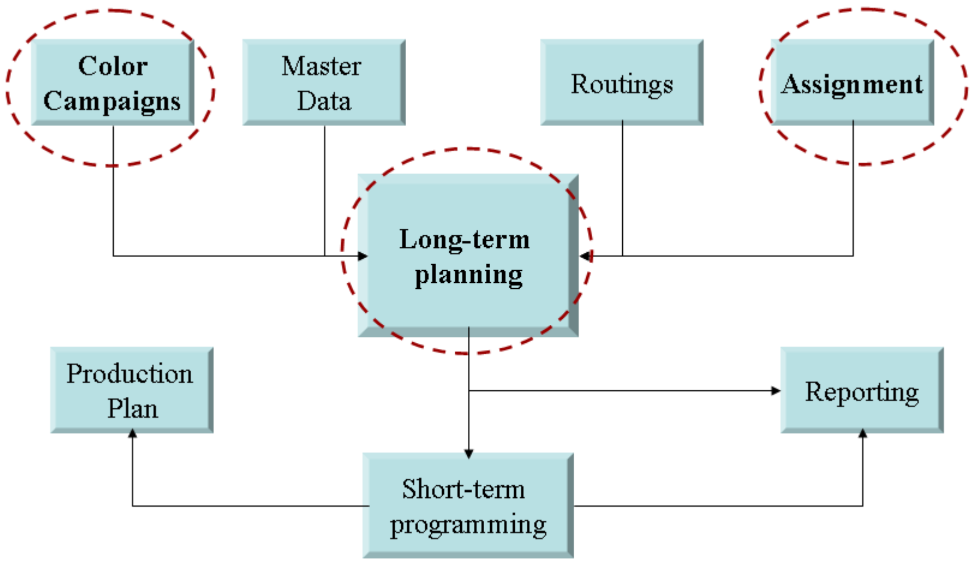


Figure 5.8: SAP R/3 production planning module information flow

APS is currently being implemented. We are aware that many scheduling systems, of the most diverse manufacturing environments, remain in use for a limited amount of time right after their implementation. It is our priority to guarantee that, despite one obstacle or another that may appear, APS is not ignored and is used on a regular basis.

5.6 Conclusions

Our work is motivated by the production planning and scheduling problem faced by a large glass container company at the long-term planning level. It is a complex problem, in which the scheduling of color campaigns on furnaces and the loading of products on machines are done simultaneously.

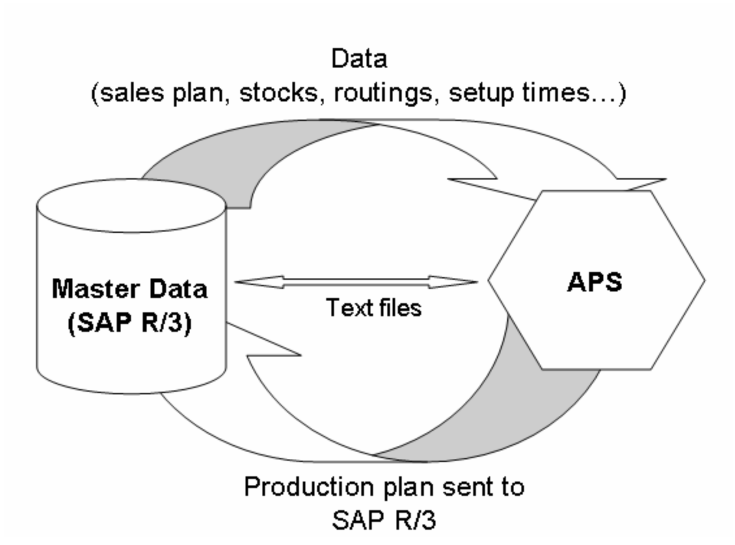


Figure 5.9: Integration of SAP R/3 production planning module and our system

Since color campaigns may be merged (to tackle the machine balancing constraints), neighborhood sizes decrease significantly throughout the search. Therefore, it seems reasonable to speed up the search whilst it is possible to achieve better solutions without a big effort, and to search thoroughly afterwards. The approach proposed in this chapter turns out to be a compromise between RVNS efficiency and VNS effectiveness through computational tests performed on a real-life instance.

The sequencing of neighborhoods through a new distance function proved to be an effective approach, improving the performance of VNS variants. More work is desirable to assess this distance function in other production planning and scheduling environments and the new VNS variant in other type of problems.

We conclude the chapter with some considerations on the validation process of the production plans by the company, and on the integration of an advanced planning system (that allocates the algorithm) with a standard ERP.

Chapter 6

Short-term production planning and scheduling

Summary. We address the short-term production planning and scheduling problem coming from the glass container industry. A furnace melts the glass that is distributed to a set of parallel molding machines. Both furnace and machine idleness are not allowed. The resulting multi-machine multi-item continuous setup lotsizing problem with a common resource has sequence-dependent setup times and costs. Production losses are penalized in the objective function since we deal with a capital intensive industry. We present two mixed integer programming formulations for this problem, which are reduced to a network flow type problem. The two formulations are improved by adding valid inequalities that lead to good lower bounds. We rely on a Lagrangian decomposition based heuristic for generating good feasible solutions. We report computational experiments for randomly generated instances and for real-life data.

6.1 Introduction

Process industries are capital intensive leading to a strong focus on improving efficiencies and reducing costs to remain competitive. It is imperative that demand is satisfied in the most cost-effective manner. The main operational driver is to maximize the facilities throughput by means of a specialization of processes to decrease downtimes. We deal with the short term production planning problem faced by a glass container manufacturing company. Glass containers are intermediary in nature, and can be considered as almost a commodity. It is a semi-continuous manufacturing process, where a common resource (furnace) produces the glass to be distributed to a set of parallel machines that will form the containers. Significant machine setup times and costs are incurred for switchovers from one product to another. The problem is to find production orders that maximize the "good tonnage" produced, while meeting a deterministic demand without backlogging. Thus, one has to minimize the production losses due to machine switchovers and furnace under capacity utilization, as well as holding costs. Additional complicate requirements are taken into account, such as minimum lot-sizes, machine balancing and furnace idleness. The resulting lotsizing and scheduling problem is an extension of the standard continuous setup lotsizing problem (CSLP).

In Chapter 2, we have presented various small time bucket models, in which at most one setup may be executed per period (DLSP, CSLP and PLSP). Thus, they are tailored for developing short-term production schedules. As far as CSLP is concerned, the majority of the mentioned papers address the single-machine variant. Contrarily to DLSP or PLSP, we are only aware of two attempts to solve the difficult multi-machine CSLP, and only one incorporates sequence dependent setup times and costs (Dastidar and Nagi [2005]). It is well known that solving CSLP is at least as hard as solving the associated DSLP. Vanderbeck [1998] questions whether the decomposition approach is practical for the generalization of CSLP to the case of multiple machines. In our problem, the total amount of the renewable, continuous resource (molten glass) available at any time is limited. Since the production rate of a product on a machine depends on the amount of the continuous resource allotted to it at a time, machines may have to produce below their own capacity. Thus, the production environment at stake does not allow an extension of DSLP and, consequently, we focus on the difficult CSLP. Productivity losses from making too many small batches are usual in lotsizing models. To the best of our knowledge, this is the first work to address the production losses of not using all of a resource, which is critical in some process industries.

This chapter solves a mixed integer programming formulation of an extension of CSLP that appears in short-term glass container production planning and scheduling (Almada-Lobo et al. [2007b]). We employ a Lagrangian decomposition approach to decouple the problem into more manageable pieces. The Lagrangian relaxation problem is modeled as a network flow type problem. We use the solution of the decomposition to develop a model-based Lagrangian heuristic by means of an efficient subgradient optimization procedure for solving the Lagrangian dual and a simple primal heuristic for yielding feasible solutions. On top of this, we implement valid inequalities that enable us to considerably improve the quality of lower bounds.

The main contributions of our work are as follows. To the best of our knowledge, this is the first work on multi-machine CSLP with sequence dependent setup times and costs and production loss costs. We solve a relevant industrial problem of a major competitive capital intensive industry. A novel Lagrangian relaxation of a proposed formulation is designed in such a way that it results in an easily solvable subproblem. In order to achieve this we relax the original formulation. We stress that a straightforward application of Lagrangian does not produce satisfactory results (these experiments are not shown here). Another major contribution of our work is a set of valid inequalities to improve the quality of the lower bounds. An excellent feature of these inequalities is the fact that their impact increases as the number of products and periods increases. Finally, we validate our approach with both real-life data and random instances.

The reminder of the chapter is organized as follows. In Section 6.2 we present mathematical models of the production planning problem arising at a short-term level in glass container industry: the exact formulation and a simplified one, which is an extension of CSLP. The following section is dedicated to a reduction of these models into network-flow problems by means of Lagrangian relaxations of the problem. The overall algorithm underlying the heuristic based on the Lagrangian approach is presented in Section 6.4. Computational results are given in Section 6.5, and in the concluding section we summarize our work.

6.2 The comprehensive formulation

Let us recall the most important requirements of this operational problem:

- each product can be carried over to the next time period,
- at most one product can be produced on a machine in any time period,

- active mold cavities are reconfigurable at the end of each time period, but the number must be within a certain range,
- the furnace can be idle,
- a product changeover at a machine uses the capacity of the furnace and therefore there is a corresponding cost, and
- a machine can only be idle at the tail of the planning horizon (they cannot be restarted during the horizon).

Here, the planner has to decide on the number of active mold cavities of each machine in each period, to ensure that machines' throughput does not exceed furnace capacity.

As before, t denotes time periods, which range from 1 to T, i and j index products, which are labeled from 1 to N, and k denotes machines, which range from 1 to K. In general, we denote by $[M]$ the set $\{1, 2, \ldots, M\}$, and by $\nu(\cdot)$ optimal values of underlying optimization problems.

We are given the following data:

d_{it}	demand for product i at the end of period t (expressed in tons)
$\overline{n}_{ik}$	the maximum number of mold cavities of machine k in which product i can be produced
$\underline{n}_{ik}$	the minimum number of mold cavities of machine k in which product i can be produced
p_{ik}	quantity of product i produced per mold cavity of machine k in a period (tons)
s_{ijk}	setup time of a changeover from product i to product j, $j \neq i$ on machine k (tons)
c_{ijk}	cost incurred to set up machine k from product i to product j, $j \neq i$
h_i	holding cost of carrying one ton of product i from one period to the next
C	melting capacity of the furnace in a period (tons).

As discussed earlier, during changeover, gobs, which are measured in tons, are returned back to the furnace. We call this the setup time even though it is measured in tons. We denote by ω the conversion factor between the idle time of the furnace and the unit of cost (usually the monetary unit).

We use the following decision variables:

Y_{it}^k	$\begin{cases} 1, & \text{if product } i \text{ is assigned to machine } k \text{ in period } t \\ 0, & \text{otherwise} \end{cases}$
Q_t	$\begin{cases} 1, & \text{if the furnace is active in period } t \\ 0, & \text{otherwise} \end{cases}$
Z_{ijt}^k	$\begin{cases} 1, & \text{if product } j \text{ is scheduled in period } t \text{ and product } i \text{ in period } (t-1), \\ & \text{both on machine } k \\ 0, & \text{otherwise} \end{cases}$
N_{it}^k	number of active mold cavities of machine k dedicated to product i in period t
I_{it}	stock of product i at the end of period t (tons)
$\overline{Id}_t$	idle capacity of the furnace in period t (tons).

We assume that I_{i0} denotes the initial inventory of product i. The short term lotsizing and scheduling problem is modeled as the following MILP formulation, denoted by F_1.

$$\nu(F_1) = \min \sum_{i,j,k,t} c_{ijk} \cdot Z_{ijt}^k + \omega \cdot \sum_t \overline{Id}_t + \sum_{i,t} h_i \cdot I_{it} \tag{6.1}$$

$$I_{it} + d_{it} - I_{i(t-1)} = \sum_k \left(p_{ik} \cdot N_{it}^k - \sum_j s_{jik} \cdot Z_{jit}^k \right) \quad i \in [N], t \in [T] \tag{6.2}$$

$$\sum_{i,k} p_{ik} \cdot N_{it}^k + \overline{Id}_t = C \cdot Q_t \quad t \in [T] \tag{6.3}$$

$$N_{it}^k \leq \overline{n}_{ik} \cdot Y_{it}^k \quad i \in [N], k \in [K], t \in [T] \tag{6.4}$$

$$N_{it}^k \geq \underline{n}_{ik} \cdot Y_{it}^k \quad i \in [N], k \in [K], t \in [T] \tag{6.5}$$

$$\sum_i Y_{it}^k \leq 1 \quad k \in [K], t \in [T] \tag{6.6}$$

$$\sum_i Y_{it}^k \geq \sum_i Y_{i(t+1)}^k \quad k \in [K], t \in [T-1] \tag{6.7}$$

$$Q_t = \sum_i Y_{it}^k \quad k \in [K], t \in [T] \tag{6.8}$$

$$Y_{jt}^k + Y_{i(t-1)}^k \leq Z_{ijt}^k + 1 \quad i \in [N], j \in [N] \setminus \{i\}, k \in [K], t \in [T] \tag{6.9}$$

$$(I_{it}, \overline{Id}_t, Q_t) \geq 0, N_{it}^k \text{ integer}, (Y_{it}^k, Z_{ijt}^k) \in \{0,1\}.$$

The objective function (6.1) aims at minimizing the sum of sequence dependent changeover, and holding and furnace idleness costs. Idleness is an opportunity cost for not pulling the maximum out of the furnace. Constraints (6.2) balance the inventory flow for two consecutive periods and together with $I_{it} \geq 0$ ensure that demand is met without backlogging. Note that the parameter p_{ik} ("good" tonnage of product i produced per mold cavity of machine k in a day) is given by

$$p_{ik} = CR_{ik} \cdot w_i \cdot 24 \cdot 60 \cdot \eta_k,$$

where CR_{ik} denotes the cavity rate of product i on machine k, w_i the weight of product i and η_k the efficiency of machine k .

Constraints (6.3) restrict the furnace melted tonnage per period to its capacity and define its idleness ($\overline{Id}_t$). In constraints (6.4), Y_{it}^k is forced to be one if a production occurs for product i on machine k in period t and the number of active mold sections (N_{it}^k) is limited by the respective pair machine/product capacity. The technological constraints, such as product i not able to be processed on machine k, are reflected in the parameter $\overline{n}_{ik}$ that would equal to zero in such circumstances. In case of a production, constraints (6.5) activate a minimum number of mold cavities on a machine. Nonzero $\underline{n}_{ik}$ implies a manager's decision based on an intrinsic restriction from the underlying production process. Constraints (6.6) prevent a machine from processing simultaneously more than one product. Intermittent machine idleness is not allowed by constraints (6.7), forcing idle periods to be placed at the end of the planning horizon (after an idle period, the machine remains idle until the end of the planning horizon). Machines fed by the same furnace must be active in the same periods of time, which is ensured by (6.8). Constraints (6.9) guarantee the coherency between variables Y_{it}^k and Z_{ijt}^k. Finally, (6.10) represent the integrality and non-negativity constraints. Note that the integrality condition of Q_t is not necessary.

In addition, the short-term planning process must also respect management rules of the different production sites like, for instance, the changing of a lot on a machine being possible only on working days and on some predefined shifts (since it is undertaken by teams of highly skilled workers), or the number of changes per week limited per facility. The number of available mold equipments may also limit the number of machines on which a product can be allocated simultaneously. Therefore, job splitting may not be allowed. All such restrictions are easy to incorporate in the model by using the existing variables.

6.3 The solution methodology

Clearly, formulation F_1 is very hard to solve. This model is simplified by relaxing the integrality of N_{it}^k and introducing continuous variables X_{it}^k to capture the approximate quantity (expressed in tons) of product i produced on machine k in period t, and by assuming null initial inventory for every product.

Let $M_{ik} = \overline{n}_{ik} \cdot p_{ik}$ be an upper bound on the quantity of product i to be produced on machine k per time period and let $m_{ik} = \underline{n}_{ik} \cdot p_{ik}$ be a lower bound on the same quantity. The model F_2 reads

$$\nu(F_2) = \min \omega \cdot C \cdot \sum_t Q_t - \omega \cdot \sum_{i,k,t} X_{it}^k + \sum_{i,j,k,t} (c_{ijk} - \omega \cdot s_{ijk}) \cdot Z_{ijt}^k + \sum_{i,t} h_i \cdot \left(\sum_k \sum_{s=1}^{t} X_{is}^k - \sum_{s=1}^{t} d_{is} \right) \tag{6.10}$$

$$\sum_k \sum_{s=1}^{t} X_{is}^k - \sum_{s=1}^{t} d_{is} \geq 0 \qquad i \in [N], t \in [T] \tag{6.11}$$

$$\sum_{i,k} X_{it}^k + \sum_{i,j,k} s_{ijk} \cdot Z_{ijt}^k \leq C \cdot Q_t \qquad t \in [T] \tag{6.12}$$

$$X_{it}^k + \sum_j s_{jik} \cdot Z_{jit}^k \leq M_{ik} \cdot Y_{it}^k \qquad i \in [N], k \in [K], t \in [T] \tag{6.13}$$

$$X_{it}^k + \sum_j s_{jik} \cdot Z_{jit}^k \geq m_{ik} \cdot Y_{it}^k \qquad i \in [N], k \in [K], t \in [T] \tag{6.14}$$

constraints (6.6) – (6.9)

$$(X_{it}^k, Q_t) \geq 0, (Y_{it}^k, Z_{ijt}^k) \in \{0, 1\}. \tag{6.15}$$

This model is an extension of the standard CSLP, which is computationally NP-hard. Clearly, there is no known polynomial algorithm to check feasibility a priori. We first argue that F_2 is a relaxation of F_1.

Proposition 1. We have $\nu(F_1) \geq \nu(F_2)$.

Proof. Let us define X_{it}^k as

$$X_{it}^k = p_{ik} \cdot N_{it}^k - \sum_j s_{jik} \cdot Z_{jit}^k \qquad i \in [N], k \in [K], t \in [T]. \tag{6.16}$$

We can remove inventory variables from model F_1 assuming, without loss of generality, null initial inventory level for every product (i.e., $I_{i0} = 0$ for every i). This fact, together with (6.16), allows us to replace (6.2) by (6.11). These constraints state that the cumulative production for item i is at least equal to the cumulative demand up to each period t. In addition, incorporating (6.16) into (6.4) and (6.5) yields constraints (6.13) and (6.14). We can argue that constraints (6.12) hold as follows:

$$C \cdot Q_t = \sum_{i,k} p_{ik} \cdot N_{it}^k + \overline{Id}_t \geq \sum_{i,k} p_{ik} \cdot N_{it}^k = \sum_{i,k} X_{it}^k + \sum_{i,j,k} s_{ijk} \cdot Z_{ijt}^k.$$

We note that both furnace capacity and setup times are expressed in tons. Thus, variable $\overline{Id}_t$ in F_1 represents the unused tonnage of the furnace in an active period t. By using (6.3) and considering (6.16), we derive the objective function (6.10). Clearly, from (6.16), X's only take integer values in F_1 since N and Z are integer and binary variables, respectively. On the other hand, X's are continuous variables in F_2. If S and R are the feasible regions of polytopes F_1 and F_2, respectively, then we have just established that $S \subseteq R$. This clearly shows that $\nu(F_2) \leq \nu(F_1)$. □

6.3.1 A network formulation A

We now reduce F_2 to a network-flow type problem. We first observe that the following constraints are an alternative formulation to constraints (6.7) and (6.9):

$$\sum_j Z_{jit}^k \geq \sum_j Z_{ij(t+1)}^k \qquad i \in [N], k \in [K], t \in [T-1]. \tag{6.17}$$

Contrarily to formulation F_2, here it is mandatory that Z_{iit}^k equals to one when machine k is set up for product i from period $t-1$ to period t (a phantom setup) and $s_{iik}=0$. Thus we define $s_{iik}=c_{iik}=0$ and we also use Z_{iit}^k. Constraints (6.17) ensure a balanced network flow of each machine configuration state and carry the setup state of the machine into the next period (as done by constraints (6.9)). They impose an output setup performed in period $t+1$ for product i to be preceded by an input setup in period t for the same product. Moreover, these constraints force idle periods to be placed at the end of the planning horizon. In case production stops in period $t-1$, period t contains no setups (idle period), i.e. $\sum_{i,j} Z_{ijt}^k = 0$ and, by constraints (6.17), $\sum_{i,j} Z_{ijs}^k = 0$ for every $s > t$. As a result, constraints (6.17) also replace constraints (6.7). We also observe that

$$Y_{it}^k = \sum_j Z_{jit}^k, \tag{6.18}$$

i.e., we conclude that product i is only assigned to machine k in period t if an input setup is performed for product i. Thus, variables Y_{it}^k can be eliminated from model F_2. After dividing the objective function (6.10) by ω, we reduce model F_2 to the following network formulation P_A:

$$\nu(P_A) = \min C \cdot \sum_t Q_t - \sum_{i,k,t} X_{it}^k + \sum_{i,j,k,t} \left(\frac{c_{ijk}}{\omega} - s_{ijk}\right) \cdot Z_{ijt}^k + \sum_{i,t} \frac{h_i}{w} \cdot \left(\sum_k \sum_{s=1}^t X_{is}^k - \sum_{s=1}^t d_{is}\right)$$

constraints (6.11) and (6.12)

$$X_{it}^k + \sum_j s_{jik} \cdot Z_{jit}^k \le M_{ik} \cdot \sum_j Z_{jit}^k \qquad i \in [N], t \in [T], k \in [K] \tag{6.19}$$

$$X_{it}^k + \sum_j s_{jik} \cdot Z_{jit}^k \ge m_{ik} \cdot \sum_j Z_{jit}^k \qquad i \in [N], t \in [T], k \in [K] \tag{6.20}$$

$$\sum_{i,j} Z_{ijt}^k \le 1 \qquad t \in [T], k \in [K] \tag{6.21}$$

$$\sum_j Z_{jit}^k \ge \sum_j Z_{ij(t+1)}^k \qquad i \in [N], t \in [T-1], k \in [K] \tag{6.22}$$

$$\sum_{i,j,k} Z_{ijt}^k = K \cdot Q_t \qquad t \in [T] \tag{6.23}$$

$$X_{it}^k \ge 0, (Z_{ijt}^k, Q_t) \in \{0,1\}. \tag{6.24}$$

By the aforementioned arguments this is an equivalent formulation to F_2. The only constraints that link the parallel machines together are (6.11) and (6.12). If we dualize these constraints by multiplying them by non-negative vectors of dual multipliers λ_{it} and π_t, respectively, the Lagrangian problem PLD_A is stated as

$$\nu(PLD_A) = \min_{X,Q,Z} \; C \cdot \sum_t Q_t - \sum_{i,k,t} X_{it}^k + \sum_{i,j,k,t} \left(\frac{c_{ijk}}{\omega} - s_{ijk}\right) \cdot Z_{ijt}^k + \sum_{i,t} \frac{h_i}{w} \cdot \left(\sum_k \sum_{s=1}^{t} X_{is}^k - \sum_{s=1}^{t} d_{is} \right)$$
$$+ \sum_{i,t} \lambda_{it} \cdot \left(\sum_{s=1}^{t} d_{is} - \sum_k \sum_{s=1}^{t} X_{is}^k \right) + \sum_t \pi_t \left(\sum_{i,k} X_{it}^k + \sum_{i,j,k} s_{ijk} \cdot Z_{ijt}^k - C \cdot Q_t \right)$$

$$\text{subject to} \qquad (6.19) - (6.24).$$

Reorganizing the terms of the objective function yields

$$\nu(PLD_A) = \min \sum_{i,k,t} \left(\pi_t - 1 - \sum_{s=t}^{T} (\lambda_{is} - \frac{h_i}{w}) \right) \cdot X_{it}^k + C \cdot \sum_t (1 - \pi_t) \cdot Q_t$$
$$+ \sum_{i,j,k,t} \left(s_{ijk} \cdot (\pi_t - 1) + \frac{c_{ijk}}{\omega} \right) \cdot Z_{ijt}^k + \sum_{i,t} d_{it} \cdot \sum_{s=t}^{T} (\lambda_{is} - \frac{h_i}{w}).$$

Taking (6.23) to replace Q_t in the previous expression, this problem decouples by machine into a set of single machine models PLD_A^k that can be written as:

$$\nu(PLD_A^k) = \min \sum_{i,t} \left(\pi_t - 1 - \sum_{s=t}^{T} (\lambda_{is} - \frac{h_i}{w}) \right) \cdot X_{it}^k + C \cdot \sum_t (1 - \pi_t) \cdot \frac{\sum_{i,j} Z_{ijt}^k}{K}$$
$$+ \sum_{i,j,t} \left(s_{ijk} \cdot (\pi_t - 1) + \frac{c_{ijk}}{\omega} \right) \cdot Z_{ijt}^k$$

$$\text{subject to} \qquad (6.19) - (6.22)$$
$$X_{it}^k \geq 0, Z_{ijt}^k \in \{0, 1\}.$$

The overall Lagrangian dual problem PLD_A can be formulated as

$$\nu(PLD_A) = \max_{\lambda,\pi} P_A = \sum_k \nu(PLD_A^k) + \sum_{i,t} d_{it} \cdot \sum_{s=t}^{T} (\lambda_{is} - \frac{h_i}{w}) \tag{6.25}$$

$$\text{subject to } \lambda_{it}, \pi_{it} \geq 0.$$

If we consider the production of product i on machine k in period t, then $\sum_j Z^k_{jit} = 1$, $\sum_j Z^k_{ji't} = 0$ for $i' \in [N] \setminus \{i\}$ and $Q_t = 1$. Given this condition and considering that this product has followed product j on the same machine ($Z^k_{jit} = 1$), we can calculate X^k_{it} by solving the following problem:

$$\begin{aligned} \theta(i,j,k,t,\lambda,\pi) = \max_X X^k_{it} \cdot \left(1 - \pi_t + \sum_{s=t}^{T}(\lambda_{is} - \frac{h_i}{w})\right) \\ \text{subject to } X^k_{it} \le M_{ik} - s_{jik} \\ X^k_{it} \ge m_{ik} - s_{jik} \\ X^k_{it} \ge 0. \end{aligned}$$

Note that if $1 - \pi_t + \sum_{s=t}^{T}(\lambda_{is} - h_i/w) \ge 0$, then $X^k_{it} = [M_{ik} - s_{jik}]^+$, otherwise $X^k_{it} = [m_{ik} - s_{jik}]^+$, where $[\cdot]^+$ represents $\max\{0; \cdot\}$. Clearly, the amount of product i to be produced in period t results from a tradeoff between multipliers λ_{it} and π_t, i.e., a tradeoff between an eventual stockout of product i in period t (constraint (6.11) is violated) and an excess of furnace production in period t (violation of constraint (6.12)). Given multipliers λ's and π's, in each Lagrangian iteration we solve N^2TK problems of the form $\theta(i,j,k,t,\lambda,\pi)$ to determine a priori the production amounts of each assigned product.

We have established that PLD^k_A reduces to

$$\nu(PLD^k_A) = \min_Z \sum_{i,j,t} \left(-\theta(i,j,k,t,\lambda,\pi) + \frac{C}{K} \cdot (1 - \pi_t) + s_{jik} \cdot (\pi_t - 1) + \frac{c_{jik}}{\omega}\right) \cdot Z^k_{jit}$$

$$\begin{aligned} \text{subject to} \quad & (6.21), (6.22) \\ & Z^k_{ijt} \in \{0,1\}. \end{aligned}$$

Consider machine k and given multipliers λ's and π's let

$$w^k_{ijt} = \begin{cases} -\theta(i,j,k,t,\lambda,\pi) + \frac{C}{K} \cdot (1 - \pi_t) + s_{jik} \cdot (\pi_t - 1) + \frac{c_{jik}}{\omega}, & \text{if } i \in [N] \\ \infty, & \text{otherwise.} \end{cases}$$

Next we show how to efficiently solve PLD^k_A. Let us define an acyclic (it contains no directed cycle) graph G with $V = [N] \times [T], A = [N] \times [N] \times [T]$ for each machine k, where each node (i, t) represents the product i to be produced in period t on machine k, and each arc $a : (i, t) \to (j, t+1)$ corresponds to the setup from product i to product j at the beginning of period $t + 1$ on machine k. Each of these arcs has weight w^k_{ijt}. Next we define a new network $G^0 = (V^0, A^0)$ by adding source node $(s, 0)$ and arcs $(s, 0) \to (j, 1)$ for every j with weight w^k_{sj1}, where s is the product produced in period 0, and a target node $(v, T + 1)$ and arcs $(i, T) \to (v, T + 1)$ for every i with zero weight. We also consider T additional nodes (i^*, t), where i^* represents a fictitious product to model the idleness state of a machine, and flow in arcs $(i, t - 1) \to (i^*, t)$ for every i, t and flow out arcs $(i^*, t) \to (v, T + 1)$ for every t both with zero weights. We note that there is not any arc $(i^*, t - 1) \to (j, t)$ for every $j \neq v$, since after an idle period the machine remains idle until the end of the planning horizon (intermittent machine idleness is not allowed). This network is illustrated in Figure 6.1.

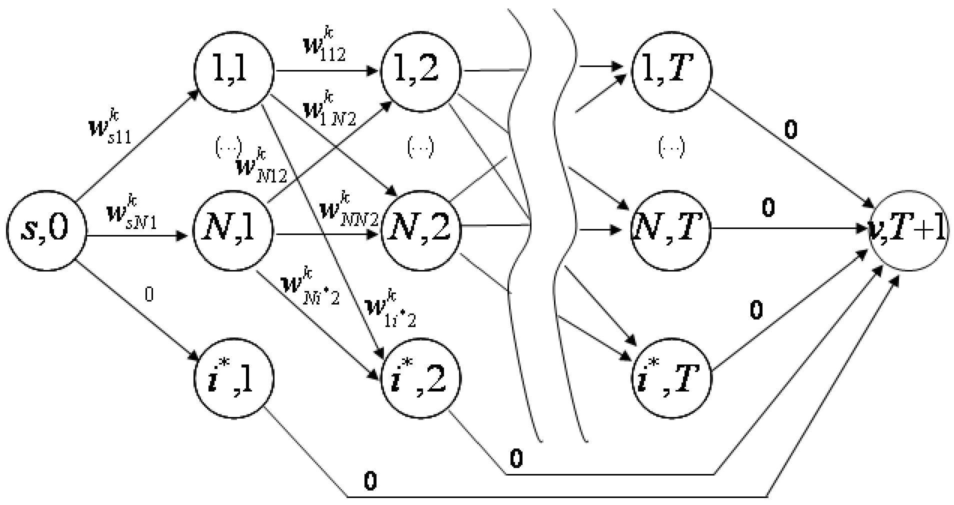


Figure 6.1: Network representation of problem PLD^k_A.

We solve PLD^k_A by finding a shortest path on an acyclic graph from node $(s, 0)$ to node $(v, T + 1)$. Note that the weight of an arc can be positive, negative or zero depending on the values of λ and π. We refer the reader to the $O(m)$ reaching algorithm described in Ahuja et al. [1993] for solving the shortest path problem in acyclic networks. Here m is the number of arcs in the network, which in our case is $O(TN^2)$. The shortest path problem is a special version of the minimum cost flow problem, with zero lower flow bound on each unit capacity arc, which aims to send 1 unit of flow from node s to node v along the path with the minimum cost (length). It is well known that in a feasible and bounded minimum cost flow problem with node supplies and arc flow bounds that are integer, there exists an optimal integral flow vector (see, e.g. Bertsekas [1998]).

Problem PLD_A^k is not directly a minimum cost flow problem, but nevertheless integrality of Z is automatic, as shown in the following theorem.

Theorem 2. *Problem PLD_A^k exhibits the integrality property, i.e. its LP relaxation exhibits an optimal integral solution.*

Proof. Assume that PLD_A^k is feasible. We prove the statement by showing that an optimal path of the network depicted in Figure 6.1 contains an optimal solution to PLD_A^k. First, we formulate the network in Figure 6.1. Let $f[(i,t_1) \to (j,t_2)]$ and $w[(i,t_1) \to (j,t_2)]$ denote, respectively, the value of flow and the weight on arc $a : (i,t_1) \to (j,t_2)$, $a \in A^0$. The formulation is as follows:

$$\min \sum_{a \in A^0} w[a] \cdot f[a] \tag{6.26}$$

$$\sum_{(j,t_2) \in V^0} f[(i,t_1) \to (j,t_2)] - \sum_{(k,t_3) \in V^0} f[(k,t_3) \to (i,t_1)] = \begin{cases} 1, & (i,t_1) = (s,0) \\ 0, & (i,t_1) \in V^0 \setminus \{(s,0),(v,T+1)\} \\ -1, & (i,t_1) = (v,T+1) \end{cases} \tag{6.27}$$

$$0 \le f[a] \le 1 \quad a \in A^0. \tag{6.28}$$

Mass balance and capacity constraints are given by (6.27) and (6.28). Recall that $G^0 \supset G$.

Now let $Z_{jit}^\star = f^\star[(j,t-1) \to (i,t)]$, for every $(j,t-1) \to (i,t) \in A$ for an optimal $f^\star$. As already argued, there exists an optimal flow vector that is integer and, therefore, $Z_{jit}^\star$ only takes binary values. It is clear that $\sum_{i,j} f^\star[(i,t_1) \to (j,t_2)] \le 1$, for every $t_2 \in [T]$, otherwise the inflow into the sink node would be greater than 1 (not allowed by constraints (6.27)) and, therefore, constraints (6.21) hold. In addition, excluding the sink node, constraints (6.27) state that the inflow of each node equals or exceeds its outflow. Then, any feasible solution satisfies (6.22) and, consequently, we conclude that $Z^\star$ is binary. $\square$

Let PLP_A be the linear programming relaxation of P_A. Since the extreme points of the feasible set of solutions of the relaxation problem PLD_A are integral, then $\nu(P_A) > \nu(PLD_A) = \nu(PLP_A)$ (Geoffrion [1974]).

6.3.2 A network formulation B

We implemented an algorithm based on PLD_A, but the results were not satisfactory and are not presented. Note that combining constraints (6.18) with (6.8), we derive the equivalent requirement:

$$Q_t = \sum_{i,j} Z_{ijt}^k \qquad \forall t \in [T], k \in [K]. \tag{6.29}$$

The second network formulation is derived by replacing the set of constraints (6.23) by the set of requirements (6.29). This problem (P_B) is formulated as

$$\nu(P_B) = \min C \cdot \sum_t Q_t - \sum_{i,k,t} X_{it}^k + \sum_{i,j,k,t} \left(\frac{c_{ijk}}{\omega} - s_{ijk}\right) \cdot Z_{ijt}^k + \sum_{i,t} \frac{h_i}{w} \cdot \left(\sum_k \sum_{s=1}^t X_{is}^k - \sum_{s=1}^t d_{is}\right)$$

$$\text{subject to} \qquad (6.11), (6.12), (6.19) - (6.22), (6.29)$$

$$X_{it}^k \geq 0, (Z_{ijt}^k, Q_t) \in \{0, 1\}.$$

It is easy to prove that this variant is at least as strong as the network formulation P_A, i.e., $\nu(P_B) \geq \nu(P_A)$.

Again, if we dualize the constraints (6.11)-(6.12) that link the parallel machines together with multipliers λ_{it} and π_t, respectively, we obtain the following Lagrangian problem PLD_B:

$$\nu(PLD_B) = \min \sum_{i,k,t} \left(\pi_t - 1 - \sum_{s=t}^T (\lambda_{is} - \frac{h_i}{w})\right) \cdot X_{it}^k + C \cdot \sum_t (1 - \pi_t) \cdot Q_t$$
$$+ \sum_{i,j,k,t} \left(s_{ijk} \cdot (\pi_t - 1) + \frac{c_{ijk}}{\omega}\right) \cdot Z_{ijt}^k + \sum_{i,t} d_{it} \cdot \sum_{s=t}^T (\lambda_{is} - \frac{h_i}{w})$$

$$\text{subject to} \qquad (6.19) - (6.22), (6.29)$$

$$X_{it}^k \geq 0, (Z_{ijt}^k, Q_t) \in \{0,1\}.$$

Due to machine balancing constraints, any feasible solution satisfies $Q_1 = \ldots = Q_l = 1$ and $Q_{l+1} = \ldots = Q_T = 0$, for a particular l. Given a fixed l, $\nu(PLD_B)$ decouples by machine into a set of single machine models $PLD_{B_l}^k$ as follows:

$$\nu(PLD_B) = \min_l \left[C \cdot \sum_{t=1}^{l} (1-\pi_t) + \sum_k \nu(PLD_{B_l}^k) \right] + \sum_{i,t} d_{it} \cdot \sum_{s=t}^{T} (\lambda_{is} - \frac{h_i}{w}),$$

where

$$\nu(PLD_{B_l}^k) = \min_{X,Z} \sum_{i,t} \left(\pi_t - 1 - \sum_{s=t}^{T} (\lambda_{is} - \frac{h_i}{w}) \right) \cdot X_{it}^k + \sum_{i,j,k,t} \left(s_{ijk} \cdot (\pi_t - 1) + \frac{c_{ijk}}{\omega} \right) \cdot Z_{ijt}^k,$$

subject to (6.19), (6.20), (6.22)

$$\sum_{i,j} Z_{ijt}^k = \begin{cases} 1, & t \in [l], k \in [K] \\ 0, & t \in [T] \setminus [l], k \in [K] \end{cases} \tag{6.30}$$

$$X_{it}^k \geq 0, Z_{ijt}^k \in \{0,1\}.$$

Similarly to PLD_A^k, $PLD_{B_l}^k$ can be reduced to

$$\nu(PLD_{B_l}^k) = \min_Z \sum_{i,j,t} \left(-\theta(i,j,k,t,\lambda,\pi) + s_{jik} \cdot (\pi_t - 1) + \frac{c_{jik}}{\omega} \right) \cdot Z_{jit}^k$$

subject to (6.22), (6.30)

$$Z_{ijt}^k \in \{0,1\}.$$

Note that (6.30) is dependent on l.

Clearly, $PLD_{B_l}^k$ also exhibits the integrality property (see Almada-Lobo et al. [2007b]). However, we note that the LP relaxation of (6.19)-(6.23) has integrality gap, therefore we can not apply Geoffrion's theorem.

6.3.3 Valid Inequalities

In this section we present four classes of valid inequalities to tighten the network formulation P_B.

From (6.11), (6.18) and $X_{it}^k \leq M_{ik} \cdot Y_{it}^k$, which follows from (6.13), we obtain that

$$\sum_{j,k} \sum_{s=1}^{t} M_{ik} \cdot Z_{jis}^k \geq \sum_{s=1}^{t} d_{is} \qquad i \in [N], t \in [T] \tag{6.31}$$

are valid inequalities. Clearly then every valid inequality for this knapsack type problem is valid for P_B. There are many known valid inequalities.

The second class of inequalities exploit the fact that once a furnace is idle, it remains inactive until the end of the time horizon.

Proposition 2. The following set of inequalities

$$\sum_{j} Z_{jit}^k \leq \sum_{j} Z_{ij(t+1)}^k + Q_t - Q_{t+1} \qquad i \in [N], t \in [T-1], k \in [K] \tag{6.32}$$

are valid for P_B.

Proof. Let Q, Z be a feasible solution to P_B and let us fix i, t, k. Clearly, $0 \leq \sum_{\bar{t}}(Q_{\bar{t}-1} - Q_{\bar{t}}) \leq 1$ since $Q_{\bar{t}-1} \geq Q_{\bar{t}}$ for every $\bar{t}$.

In case $Q_{s-1} - Q_s = 1$ for an s, then $Q_1 = Q_2 = \ldots = Q_{s-1} = 1$ and $Q_s = \ldots = Q_T = 0$. It follows that $Q_{s-1} - Q_s = 1$ and $Q_{\bar{t}-1} - Q_{\bar{t}} = 0$ for every $\bar{t} \neq s$. If $\sum_j Z_{jit}^k = 0$, then product i is not produced in period t and (6.32) is valid since the right-hand side is nonnegative. If $\sum_j Z_{jit}^k = 1$ (product i produced in period t on machine k), then clearly $Q_t = 1$. In this case we distinguish two further cases: if $Q_{t+1} = 0$ (i.e., the production stops in period t), then it follows from (6.23) that $\sum_{i,j} Z_{ij(t+1)}^k = 0$ and therefore (6.32) holds; if $Q_{t+1} = 1$, then constraints (6.22) and (6.23) imply $\sum_j Z_{ij(t+1)}^k = 1$, validating (6.32).

If such an s does not exist, then $Q_1 = Q_2 = \ldots = Q_T = 0$ and hence $\sum_j Z_{jit}^k = \sum_j Z_{ijt}^k = 0$ for every i and t. We conclude that (6.32) clearly holds. □

We note that if $Q_t = Q_{t+1}$, then (6.32) together with (6.22) impose $\sum_j Z_{jit}^k = \sum_j Z_{ij(t+1)}^k$ and, therefore, there is balanced flow through each node.

The third set of inequalities is based on those presented in Pochet and Wolsey [2006].

Proposition 3. The inequalities

$$\sum_j Z^k_{ji(t-1)} + \sum_{j:j\neq i} Z^k_{jit} \leq 1 - \sum_{j:j\neq i} Z^k_{jjt} \quad i \in [N], t \in [T] \setminus \{1\}, k \in [K]. \tag{6.33}$$

are valid for P_B.

Proof. Let us consider a Z feasible to P_B and we fix i, t, k. For ease of notation we introduce $W^k_{jt} = \sum_{u:u\neq j} Z^k_{ujt}$, which equal to 1 if start-up occurs for product j on machine k in period t, and 0 otherwise. We can now rewrite (6.33) as

$$\sum_j Z^k_{ji(t-1)} + W^k_{it} \leq 1 - \sum_{h:h\neq i} \left(\sum_j Z^k_{jht} - W^k_{ht} \right) \quad i \in [N], t \in [T] \setminus \{1\}, k \in [K]. \tag{6.34}$$

To show (6.34), we consider three cases.

1) Let us first consider $\sum_j Z^k_{ji(t-1)} + W^k_{it} = 0$. Then $\sum_{h:h\neq i} \left(\sum_j Z^k_{jht} - W^k_{ht} \right) = \sum_{h:h\neq i} Z^k_{hht} \leq 1$, where we used (6.21). This establishes (6.34).

2) Let now $W^k_{it} = 1$. It implies that product i is not produced in period $t-1$ and $\sum_j Z^k_{ji(t-1)} = 0$. Hence the left-hand side of (6.34) equals 1. Clearly then $W^k_{ht} = 0$ for every $h, h \neq i$. We also have $\sum_j Z^k_{jht} = 0$ for every $h, h \neq i$. We conclude that $\sum_{h:h\neq i} \left(\sum_j Z^k_{jht} - W^k_{ht} \right) = 0$ and thus the right-hand side in (6.34) equals 1.

3) Let us now assume that $\sum_j Z^k_{ji(t-1)} = 1$. Then product i is produced in period $t-1$ and hence no start-up for product i occurs in period t. It means that $W^k_{it} = 0$ and the left-hand side of (6.34) is thus 1. If product i is produced also in period t, then clearly $W^k_{ht} = \sum_j Z^k_{jht} = 0$ for every $h, h \neq i$. If product i is not produced in period t, then any setup for product $h \neq i$ in period t must be accompanied by a start-up, i.e., $\sum_{j,h:h\neq i} Z^k_{jht} = \sum_{h:h\neq i} W^k_{ht}$. We conclude that the right-hand side of (6.34) is 1.

From the three cases it follows that $\sum_j Z^k_{ji(t-1)} + W^k_{it} \leq 1$. Thus case 1 covers the case $\sum_j Z^k_{ji(t-1)} + W^k_{it} = 0$, while the remaining two cases cover $\sum_j Z^k_{ji(t-1)} + W^k_{it} = 1$. This argument shows that the three cases cover all possibilities. □

For the remaining class of inequalities, let $M^*_i = \max_k M_{ik}$ and we define

$$\delta_t = \left(N - \left\lfloor \frac{(t-1)\cdot K}{\min_i \lceil \frac{\sum_s d_{is}}{M_i^*} \rceil} \right\rfloor \right)^+ .$$

Proposition 4. The inequalities

$$\delta_t \le \sum_{\substack{i,j,k \\ j \neq i}} \sum_{s=t}^{T} Z_{jis}^k \quad t \in [T] \tag{6.35}$$

are valid for P_B.

Proof. In a feasible solution to P_B any product has a minimum number of production time slots given by $\min_i \lceil \frac{\sum_s d_{is}}{M_i^*} \rceil$. At the end of period $t-1$, we might have faced the entire production requirements of at most $\left\lfloor \frac{(t-1)\cdot K}{\min_i \lceil \frac{\sum_s d_{is}}{M_i^*} \rceil} \right\rfloor$ products. Thus δ_t is a lower bound on the number of start-ups that must be performed in periods $t, t+1, \ldots, T$. It is easy to see that in period t the minimum number of start-ups for the remaining planning horizon is given by δ_t. Thus (6.35) are valid for P_B. □

6.4 A Lagrangian Heuristic

In this section we exploit the problem structure and build a heuristic method to obtain feasible solutions based on Lagrangian relaxation.

The success of any Lagrangian approach depends upon three features: the tightness of the lower bound provided by the sub-problem, the ability to produce good primal feasible solutions, and the efficiency in solving the Lagrangian dual. A successful technique to solve the Lagrangian dual is the well-known subgradient optimization algorithm (see, e.g., Held et al. [1974]). Let $PLD_B(\lambda^m, \pi^m)$ denote the dual function at iteration m. In order to compose a search direction to update the multipliers, let us define two subgradients of

$PLD_B(\lambda^m, \pi^m)$ based on

$$\zeta_t^m = \sum_{i,k} (\overline{X}_{it}^k)^m + \sum_{i,j,k} s_{ijk} \cdot (\overline{Z}_{ijt}^k)^m - C \cdot (\overline{Q}_t)^m \quad t \in [T] \text{ and} \tag{6.36}$$

$$\Omega_{it}^m = \sum_{s=1}^{t} d_{is} - \sum_k \sum_{s=1}^{t} (\overline{X}_{is}^k)^m \quad i \in [N], t \in [T], \tag{6.37}$$

where $\overline{X}, \overline{Z}, \overline{Q}$ denote an optimal solution to $PLD_B(\lambda^m, \pi^m)$. Lagrangian multipliers are updated according to the recursions

$$\pi_t^{m+1} = [\pi_t^m + \varphi^m \cdot \zeta_t^m]^+ \text{ and } \lambda_{it}^{m+1} = [\lambda_{it}^m + \tau^m \cdot \Omega_{it}^m]^+,$$

where φ^m and τ^m are the step sizes in iteration m and $[\cdot]^+$ ensures their projection onto the nonnegative orthant. A suitable step size is crucial for fast convergence of the subgradient method. Let μ^m be a parameter satisfying $0 < \mu^m \leq 2$, UB an upper value on the dual function PLD_B and $\| \cdot \|$ the Euclidean norm. We use the following stepsizes:

$$\varphi^m = \mu^m \cdot \frac{UB - \nu\,(PLD_B(\lambda^m, \pi^m))}{\| \zeta^m \|^2} \text{ and } \tau^m = \mu^m \cdot \frac{UB - \nu\,(PLD_B(\lambda^m, \pi^m))}{\| \Omega^m \|^2}.$$

The main advantage of this relaxation is that it yields a simple sub-problem, since solving each $PLD_{B_l}^k$ is equivalent to finding a shortest path on an acyclic graph for each l. This property enables a large number of subgradient iterations in order to solve the Lagrangian dual.

In our implementation, we chose $\mu^1 = 1.1$ as an initial value, and if no improvement of the lower bound is obtained in 10 successive iterations, we set $\mu^m = 0.5 \cdot \mu^{m-1}$ and reset μ^m back to 1.1 whenever we get an improved solution. This algorithm is stopped when the gap between the upper bound and the Lagrangian bound is less than 0.15%, or after 30 iterations without a lower bound improvement.

An important component of the Lagrangian solution is deriving feasible solutions to F_2. A solution to F_2 is characterized by the setup pattern Z_{ijt}^k and the production quantities X_{it}^k that are assigned according to this setup pattern. We fix the setup variables of F_2 with the values from the underlying PLD_B solution, and solve the remaining linear program to optimality to obtain the production amounts.

6.5 Computational Results

Computational experiments were performed on an ASUS personal computer with 3.0 GHz CPU and 2GB of random access memory. CPLEX 10.1 from ILOG was used as the mixed integer programming solver and the Lagrangian approach was coded in OPL version 5.1 also from ILOG.

Test problems were generated using the following generator. The number of machines K equals to three, the number of products N were 5, 10, and 15 and the number of periods T were 30 and 45. External demand occurs at the end of each fifth period for all products (time between orders equals 5 time periods) and it is drawn from uniform distribution $U(400, 1000)$. Let Cut denote the (approximate) furnace capacity utilization. We consider medium capacitated problems ($Cut = 0.6$) and high capacitated problems ($Cut = 0.8$). Furnace daily capacity C is given by $\sum_{i,t} d_{it}/(T \cdot Cut)$. Setup times ($s_{ijk}$) were generated based on $U[0.10 \cdot \frac{C}{K}, 0.20 \cdot \frac{C}{K}]$ for every $i \neq j$, and are zero for $i = j$. If we assume that the cost c_{ijk} captures the wasted tonnage, then they are measured in tons and $c_{ijk} = s_{ijk}$ and the weighting factor ω equals to 1. Since the glass containers are almost considered as a commodity, holding costs (h_i) are the same for all products and they are 0.20.

Regarding the upper and lower bounds, two different settings are analyzed. The first setting S_1 considers upper bounds on the production of product i on machine k (M_{ik}) obtained from a normal distribution with an expected value of C/K and a coefficient of variation of 0.1, while the respective lower bound (m_{ik}) is derived based on the expression $U[0.4, 0.8] \cdot M_{ik}$. This setting reflects the unique properties of the glass container industry. The second setting S_2 considers $M_{ik} = m_{ik} = C/K$ and reflects the characteristics of DLSP instances (as the discrete production policy takes place), and is used for comparison purposes. We note that the parameters of this generator were based on our case study data. For instance, the coefficients of the uniform distribution that randomly generates m_{ik} followed from the fact that some machines can stop at most 20% of their sections, while other can stop almost 60% of them.

For each quadruplet N, T, Cut and S (with $K = 3$), ten different instances were generated. In addition, for $K = 4$, $Cut = 0.8$ and S_1, ten instances were generated for each pair (N, T). Hereafter we present for each instance type the average of the values obtained across the 10 instances. Figure 6.2 illustrates a feasible solution for an instance of type $K = 3/S_1/Cut = 0.8/N = 10/T = 30$. Here, the furnace is active throughout the entire planning horizon, and it is heavily loaded in the last third of the plan. Capacity is clearly tight, therefore the furnace (and the associated machines) could not be stopped beforehand.

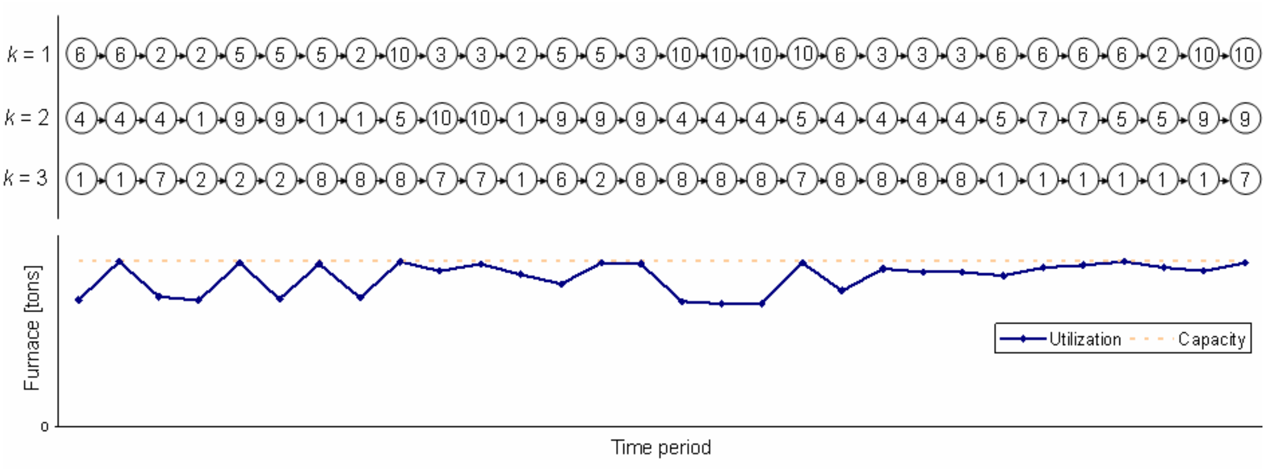


Figure 6.2: Solution example for the instance $K = 3/S_1/Cut = 0.8/N = 10/T = 30$

An instance of type $K = 3, S_1, N = 15, T = 45$ produces an IP in formulation F_2 with 6,330 rows, 32,418 columns and 194,895 nonzeros, while $K = 4, S_1, N = 15, T = 30$ contains 5,372 rows, 26,562 columns and 156,619 nonzeros. These are fairly large IPs that are very hard to solve to optimality in reasonable time.

We first compare the LP relaxation of models F_2 and F_2 strengthened by the four sets of valid inequalities described in Section 6.3.3. The optimal value of the LP relaxation of F_2 is denoted by $\nu(F_{2LP})$ and after adding the valid inequalities by $\nu(F^{\star}_{2LP})$. Tables 6.1 and 6.2 present the gaps $\frac{\nu(F^{\star}_{2LP})-\nu(F_{2LP})}{\nu(F_{2LP})}$ for settings S_1 and S_2, respectively. The impact of the valid inequalities is larger for instances with medium capacity utilization than for instances with high capacity utilization. Moreover, it is clear that this impact is more stressed for setting S_2 than for S_1, and tends to increase as the number of products and periods increase.

Tables 6.3 and 6.4 display the minimum, average, and the maximum gap of the heuristic solution from the lower bound for settings S_1 and S_2, respectively. The heuristic finds

Table 6.1: Comparison (%) of $F^{\star}_{2LP}$ and F_{2LP} for setting S_1, $K = 3$

	$Cut = 0.6$		$Cut = 0.8$	
N	$T = 30$	$T = 45$	$T = 30$	$T = 45$
5	2.2%	2.6%	0.1%	0.0%
10	2.9%	3.4%	0.4%	0.2%
15	2.5%	3.6%	1.0%	0.3%

Table 6.2: Comparison (%) of $F^{\star}_{2LP}$ and F_{2LP} for setting S_2, $K = 3$

	$Cut = 0.6$		$Cut = 0.8$	
N	$T = 30$	$T = 45$	$T = 30$	$T = 45$
5	6.3%	8.8%	2.3%	3.0%
10	7.1%	9.4%	2.4%	2.9%
15	8.3%	11.1%	2.5%	2.9%

a feasible solution for all problem instances, excepting the two most tightly capacitated instance: $S_1, Cut = 0.8/N = 15, T = 30$ and $S_1, Cut = 0.8, N = 15, T = 45$. The results indicate that for both S_1 and S_2 the heuristic performance deteriorates as the number of products increases. This situation is emphasized when the discrete production policy is relaxed and the furnace may run up to capacity (setting S_1). Regarding the number of periods, it seems that the performance of the heuristic behaves differently from S_1 to S_2. For setting S_1 its performance worsens as the number of products increases, whereas for S_2 the gap between the lower and the upper bound either tends to decrease as T increases ($Cut = 0.6$) or it is almost not influenced by T ($Cut = 0.8$). It is clear that the algorithm performs very well on S_2 since the largest gap is less than 10%. The performance on S_1 is not that encouraging. The total computational times and the number of iterations in each run are given in Table 6.5 for $S_1, Cut = 0.8$. We note that for all instances the total running time never exceeded 1 hour.

Table 6.6 presents the average number of branch-and-bound nodes and the optimality gap (%) obtained by CPLEX 10.1 for the same instances as those presented in Table 6.3 within a one hour time limit. An empty field means that CPLEX 10.1 was not able to

Table 6.3: Gap (%) between the lower and upper bounds for setting S_1, $K = 3$

	$Cut = 0.6$		$Cut = 0.8$	
N	$T = 30$	$T = 45$	$T = 30$	$T = 45$
5	4.0/5.7/7.4	5.9/8.0/12.3	8.9/12.5/17.5	19.4/22.8/26.9
10	7.3/11.9/19.0	11.8/23.9/30.8	20.7/24.8/28.7	30.8/47.6/53.1
15	13.4/22.0/29.9	29.1/37.4/47.3	29.2/38.5/49.2	

minimum / average / maximum gap (%)

Table 6.4: Gap (%) between the lower and upper bounds for setting S_2, $K = 3$

	$Cut = 0.6$		$Cut = 0.8$	
N	$T = 30$	$T = 45$	$T = 30$	$T = 45$
5	1.3/2.5/3.7	0.6/1.0/1.4	0.9/1.2/1.9	0.6/1.3/4.0
10	2.8/5.6/9.9	2.4/3.4/5.2	1.1/1.7/2.7	0.7/2.4/6.2
15	4.5/6.8/8.5	3.3/5.2/6.9	1.3/3.0/5.0	

minimum / average / maximum gap (%)

Table 6.5: Average running times for $Cut = 0.8$ and setting S_1, $K = 3$

	$T = 30$		$T = 45$	
N	# Iterations	CPU time (secs)	# Iterations	CPU time (secs)
5	32	63	37	134
10	55	655	59	$1,174$
15	76	$1,864$	81	$2,976$

generate any feasible solutions within the time limit. There was even an instance for $T = 45, N = 15, Cut = 0.6$ where CPLEX 10.1 was not able to find a solution. This instance was discarded and it was not included in the reported average. As the size of the instance gets bigger our results clearly outperform those obtained by CPLEX. Only for $N = 5, 10$ and $T = 30, 45$ with $Cut = 0.6$ and $N = 5$, $T = 30, 45$ with $Cut = 0.8$ CPLEX 10.1 outperforms our Lagrangian approach. In all other cases we produce substantially better gaps (and also lower running times). It is also clear from the substantially lower number of branch-and-bound nodes as N increases that LP relaxations become much more difficult. This is another advantage of our Lagrangian approach since we do not solve LP relaxations and thus our algorithm is more scalable.

Table 6.6: CPLEX 10.1 optimality gap (%) and nodes within the one hour time limit

	$Cut = 0.6$				$Cut = 0.8$			
	$T = 30$		$T = 45$		$T = 30$		$T = 45$	
N	# Iter.	Gap	# Iter.	Gap	# Iter.	Gap	# Iter.	Gap
5	$6,020,622$	0.7%	$3,129,722$	1.7%	$4,442,038$	1.4%	$2,314,822$	4.5%
10	$1,842,742$	8.3%	$928,530$	11.1%	$1,747,835$		$892,739$	
15	$819,888$	28.1%	$534,417$	58.7%	$695,177$		$457,354$	

Table 6.7 displays the same statistics as Table 6.3 for larger instances, made up of four machines for $Cut = 0.8$. Comparing the results in Tables 6.3 and 6.7 it is clear that the gap improves as the number of machines increases. Here again, the algorithm failed to find feasible solutions for hardiest instance ($K = 4, S_1, N = 15, T = 45$). We note that CPLEX 10.1 runs out of memory for instances $N = 15, T = 45, Cut = 0.8$ without finding any upper bound.

Table 6.7: Gap (%) between the lower and upper bounds for setting S_1, $K = 4$

	$Cut = 0.8$	
N	$T = 30$	$T = 45$
5	4.8/5.7/6.8	7.4/10.9/15.2
10	12.9/19.5/25.9	32.2/44.8/53.0
15	18.5/26.9/38.2	

minimum / average / maximum gap (%)

Figure 6.3 shows the trend of the gap between the lower and upper bound in the Lagrangian approach for an instance of the type $K = 3, S_1, Cut = 0.8, N = 5, T = 30$. After approximately 25 iterations with a lower bound improvement, the gap appears to stabilize.

Finally, Table 6.8 gives the solution gap (%) and the main characteristics of different real-world instances of our problem. The results for these instances outperform considerably

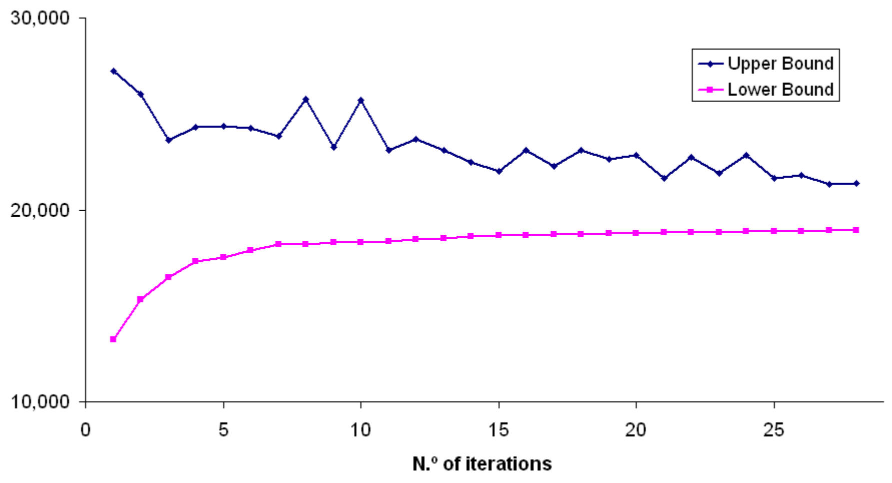


Figure 6.3: Solution example for the instance $K = 3, S_1, Cut = 0.8, N = 10, T = 30$

those obtained previously for randomly generated instances. Two main reasons that make the problem slightly easier for real-world instances are as follows. First, in real-world instances technology constraints do not allow products to be assigned to some machines, which reduces the number of variables. Second, the demand is not observed every fifth period for every product but it is more sparse (e.g., two orders of the same product may lag more than 10 time periods). It can be seen, despite the small sample dimension, that the gap increases with the number of time periods, and as the instances become more highly capacitated.

6.6 Conclusions

In this chapter, we address the short-term production planning and scheduling problem faced by a glass container company, where a limited renewable, continuous resource is distributed to a set of parallel molding machines. After developing an exact formulation, we simplify it into an extension of the standard continuous lotsizing problem (CSLP). Computationally, the problem corresponding to this model is NP-hard. We then reduce it to a network flow type model, which is decoupled by machine through a Lagrangian relaxation scheme. Since the subproblems are easily solvable, we are able to run a large number of iterations in a short period of time. Feasible solutions are generated with a model-based Lagrangian heuristic. We carry out a set of computational experiments on relatively large real-world and randomly

Table 6.8: Gap (%) between the lower and upper bounds for various real-life instances

N	T	K	Cut	m_{ik}/M_{ik}	Gap(%)
7	35	3	0.83	0.64	10.5
4	24	3	0.71	0.74	1.4
7	13	3	0.77	0.67	4.1
6	32	3	0.81	0.63	1.8
14	19	3	0.84	0.65	5.9
14	16	5	0.63	0.69	4.9
8	13	4	0.50	0.74	3.7
7	23	4	0.80	0.74	3.9
11	16	4	0.60	0.75	5.2
9	15	3	0.80	0.64	4.9
8	18	5	0.80	0.68	4.6
12	21	3	0.71	0.67	5.1

generated instances.

The contributions of this research are fourfold. First, we solve a relevant industrial problem within a very competitive industry. Second, we are not aware of any research tackling CSLP with multiple-machines with the presence of production losses (due to setups and capacity surplus). Due to its inherent complexity, the research community has overlooked multi-machine CSLP with sequence dependent setups. Third, we have employed an efficient Lagrangian based heuristic for this problem. Finally, we have implemented valid inequalities that enable us to reduce the integrality gap.

CSLP has not been widely tackled by researchers due to its computational challenges. This study further suggests that this problem (clearly useful in practice) is a challenging area for future research. Additionally, opportunity costs for not pulling the most out of a resource are critical in capital intensive industries. Though our computational results are very encouraging, there is room for improvements. We need to study under what conditions the presented valid inequalities are strong (define facets), since it seems that their impact on the improvement of the lower bound is dependent upon the instance type. Another important research question is to find valid inequalities to be added to (6.19)-(6.23) in order that its LP relaxation has no integrality gap (that provide a complete description of the respective polyhedron). Additionally, it is clearly important to find strategies to combine even stronger valid inequalities based on the polyhedral structure of this problem with tighter reformulations. These inequalities should take into account the furnace capacity constraints and should cut across multiple machines. Lagrangian relaxation appears to be very suitable for determining feasible solutions. An improvement heuristic may be developed to even further close the gap between the lower and upper bound.

Chapter 7

Conclusions and research agenda

This thesis investigates the production planning and scheduling problem that arises in the glass container industry. We note that some of the challenges posed by this problem are common to other manufacturing environments, thus we believe that the models, data and solution approaches described here have applications to the solution of other problems. Chapter 2 gives an overview of various deterministic single-level dynamic lotsizing models. In Chapter 3 we present the main features and challenges of glass container industry and devise a hierarchical framework to tackle the production planning. Chapter 4 looks into capacitated lotsizing problem (CLSP) with sequence dependent setups, and in chapters 5 and 6 we address extensions of CLSP and continuous setup lotsizing models (CSLP) to tackle the upper and lower levels, respectively, of this hierarchical system. Here we briefly summarize our four general contributions and discuss possible research directions.

Firstly, we study in depth the glass container industry. We detect a set of production planning improvement opportunities in our case study (owing to the large number of potential production combinations, almost no attempt at optimizing the schedules is made). We claim that the complexity of the whole production planning system is not suitable to be tackled by a monolithic model (even though, such model is discussed in Appendix D). We address the design of the hierarchical system for glass container industry and propose a new two-level hierarchical structure to overcome the conceptual inadequacies of the traditional three-level approach.

Secondly, as far as CLSP is concerned, we first analyze the model proposed by Gupta and

Magnusson [2005], prove that is actually not a completely accurate formulation and add a new set of constraints into their model to provide an exact formulation for this problem. We then present two novel exact formulations for modeling setup carryover in the challenging CLSP problem with sequence dependent setup times and costs. The models are simpler than others available in the literature. We formally show that one of the formulations is stronger than the other and we implement valid inequalities that enable us to reduce the integrality gap of the LP relaxation with a desirable property: the integrality gap decreases as the number of products increases (it is independent on the number of periods). To introduce the most violated inequalities, we solve a separation problem. Moreover, we develop a five-step heuristic that is effective both in finding a feasible solution (even for tightly capacitated instances) and in producing good solutions to these problems. We are only aware of one paper in the literature to publish results for this problem, that are clearly beaten by ours.

Thirdly, a new variant of the Variable Neighborhood Search (VNS) is introduced to tackle the lotsizing and scheduling problem that arises at the long-term planning level of the glass container industry. Throughout the search, neighborhood sizes decrease significantly. Therefore, we try to speed up the search whilst it is possible to achieve better solutions without a big effort, and to search thoroughly afterwards. The approach proposed here turns out to be a compromise between the efficiency and effectiveness of two other VNS variants, through computational tests performed on a real-life instance. It is clear that as the quality of the initial solution worsens and/or the available computational time decreases, this new variant becomes more attractive and interesting than the other two. In addition, the sequencing of neighborhoods through a new distance measure proves to be an effective approach, improving the performance of VNS variants.

Fourthly, we address the short-term lotsizing and scheduling problem of the glass container industry. Two mathematical formulations are presented: an exact formulation and a simplified one, which is an extension of CSLP that is reduced to a network flow type problem. A novel Lagrangian relaxation of this problem is designed in such a way that it results in an easily solved subproblem (which is formally proved). Another major contribution of our work is the set of valid inequalities to improve the quality of the lower bounds: their impact increases as the number of products and periods increases. Part of the Lagrangian solution is used to generate upper bounds. The heuristic finds a feasible solution for almost all problem instances and it is clear that the quality of the feasible solution improves as the number of machines increases. This approach is validated with both real-life data and randomly generated instances. Remarkably, the results for real-world instances outperform considerably those obtained for randomly generated instances. It is worth noting that we are not aware of any paper tackling CSLP with multiple-machines at the presence of production losses (due to its inherent complexity, the research community has overlooked multi-machine CSLP with sequence dependent setups).

The ongoing research on CLSP explores the simplicity of the five-step heuristic by means of its implementation as a building block of more complex heuristics and metaheuristics. It is worth pointing out that it may effectively generate good initial solutions and repair infeasibilities due to the capacity requirements of solutions generated by any local search procedure. Regarding VNS, more work is desirable to assess the distance measure in other production planning and scheduling environments and the new VNS variant in other type of problems. As far as CSLP is concerned, we need to study under which conditions the suggested valid inequalities are strong (define facets), since it seems that their impact on the improvement of the lower bound is dependent upon the instance type. An improvement heuristic is to be developed to sharpen the gap between lower and upper bounds.

Further research into lotsizing and scheduling problems poses interesting challenges. The majority of the literature relies on instance random generators to carry out computational experiments, overlooking the effect of dealing with real life instances. There is a need for real life data, as well as for more insights into how to trade off the complexity of reality (if the model is unrealistic leads to poor decision-making) with mathematical tractability (if the model is unsolvable, it is useless). Extensions of this research can be accomplished in several ways for application to different realistic problem settings. Naturally, the inclusion of industrial features turns mathematical models larger and more complex. To deal with additional complexities, the combination of existing algorithms, tighter models and stronger valid inequalities based on the polyhedral structure of these problems come into play. More powerful hybrid algorithms are needed to reduce the integrality gap. It is our goal to obtain insights into the way different solution approaches must be combined to develop efficient tools for solving these hard problems. Given theirs flexibility to deal with complex manufacturing environments, metaheuristics will lie at the very heart of our hybrid approaches. While combining metaheuristics with mixed integer programming (MIP) techniques, we believe the key will be to isolate the aspects that a MIP can solve quickly and try to predict MIP performance.

There exists very few reformulation results concerning multi-item, multi-machine production planning problems. Apart from the challenging theoretical point of view, the practical relevance of this problem is supported by examples of its application on various industries. Belvaux and Wolsey [2001] and Wolsey [2002] claim that many practical lotsizing problems can be solved using commercial optimization software if tight formulations are provided.

Note that the majority of lotsizing and scheduling models published so far only perform a setup whenever there is sufficient time available to do it entirely. In the case of very large setup times (as the ones observed in the glass container industry that may reach 120 hours), setup operations may start at the end of one period and finish at the beginning of the following period. This new feature has not yet been incorporated in CLSP models. Recently, Suerie [2006] proposes two formulations to model proportional setup lotsizing problems with period overlapping setup times. Derivation of special purpose heuristics and the improvement of the integrality gap of these formulations are needed.

The extension of CLSP with sequence dependent setups and setup carryovers to the case of multiple machines is an interesting area for future research. We note that new valid inequalities should cut across multiple machines and take into account the capacity constraints, the potential cycles and paths of products sequence in each time period. Jans [2006] looks at how to incorporate parallel machines in a MIP model using commercial optimization software, but the author considers sequence independent setups and no setup carryover. Another extension is to problems with sequence dependent setups that do not satisfy the triangular inequality. There remains room for research on the recent work of Clark [2006].

Hopefully, our work will provide some motivation for further research on these topics. The important questions may not have been asked, so let's keep on working.

Bibliography

R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice-Hall, Englewood Cliffs, NJ, 1993.

F. Al-Khayyal, P. Griffin, and N. Smith. Solution of a large-scale tewo-stage decision and scheduling problem using decomposition. *European Journal of Operational Research*, 132: 453–465, 2001.

B. Almada-Lobo. *Concepção e elaboração de um modelo para gestão da produção*. Relatório de estágio, Faculdade de Engenharia da Universidade do Porto, 2002.

B. Almada-Lobo. Planeamento e escalonamento de produção: Visão global e caso de estudo. Technical report, Faculdade de Engenharia da Universidade do Porto, January 2005.

B. Almada-Lobo, D. Klabjan, M. A. Carravilla, and J. F. Oliveira. Single machine multi-product capacitated lotsizing with sequence-dependent setups. *International Journal of Production Research*, 45(20):4873–4894, 2007a.

B. Almada-Lobo, D. Klabjan, M. A. Carravilla, and J. F. Oliveira. Multi-machine continuous setup lotsizing with sequence-dependent setups. *Submitted for publication in the Computational Optimization and Applications*, 2007b.

B. Almada-Lobo, J. F. Oliveira, and M. A. Carravilla. Production planning and scheduling in the glass container industry: A VNS approach. *Submitted for publication in the International Journal of Production Economics*, 2007c.

B. Almada-Lobo, J. F. Oliveira, and M. A. Carravilla. A note on "The capacitated lot-sizing and scheduling problem with sequence-dependent setup costs and setup times". *Computers and Operations Research*, 35(4):1374–1376, 2008.

R. Alvarez-Valdes, A. Fuertes, J. Tamarit, G. Giménez, and R. Ramos. A heuristic to schedule flexible job-shop in a glass factory. *European Journal of Operational Research*, 165:525–534, 2005.

R. N. Anthony. Planning and control systems: A framework for analysis. Technical report, Harvard University Graduate School of Business, 1965.

S. Araujo, N. Arenales, and A. R. Clark. Joint rolling-horizon scheduling of materials processing and lot-sizing with sequence-dependent setups. *Journal of Heuristics*, 13(4):337–358, 2007.

I. Barany, T. J. Vanroy, and L. A. Wolsey. Strong formulations for multi-item capacitated lot sizing. *Management Science*, 30(10):1255–1261, 1984.

G. Belvaux and L. A. Wolsey. Modelling practical lot-sizing problems as mixed-integer programs. *Management Science*, 47(7):993–1007, 2001.

D. Bertsekas. *Network optimization: continuous and discrete models.* Athena Scientific, Belmont, MA, 1998.

M. Besten and T. Stutzle. Neighborhoods revisited: An experimental investigation into the effectiveness of variable neighborhood descent for scheduling. In *MIC'2001 - 4th Metaheuristics International Conference*, Porto, Portugal, 2001.

G. R. Bitran and A. Hax. On the design of hierarchical production planning systems. *Decision Science*, 8:28–55, 1977.

G. R. Bitran, E. A. Haas, and A. C. Hax. Hierarchical production planning - a single stage system. *Operations Research*, 29(4):717–743, 1981.

G. R. Bitran, E. A. Haas, and A. C. Hax. Hierarchical production planning - a 2-stage system. *Operations Research*, 30(2):232–251, 1982.

K. Boskma. Aggregation and the design of models for medium-term planning of production. *European Journal of Operational Research*, 10(3):244–249, 1982.

N. Brahimi, S. Dauzere-Peres, N. M. Najid, and A. Nordli. Single item lot sizing problems. *European Journal of Operational Research*, 168(1):1–16, 2006.

W. Bruggemann and H. Jahnke. Remarks on: "Some extensions of the discrete lotsizing and scheduling problem". *Management Science*, 43(1):122–122, 1997.

W. Bruggemann and H. Jahnke. The discrete lot-sizing and scheduling problem: Complexity and modification for batch availability. *European Journal of Operational Research*, 124(3): 511–528, 2000.

P. Bukh. A bibliography of hierarchical production planning tecniques, methodology, and apllications - version 2. Technical report, Institute of Management, University of Aarhus, 1994.

M. Carravilla and J. P. Sousa. A hierarchical production planning in a make-to-order company: A case study. *European Journal of Operational Research*, 86:43–56, 1995.

D. Cattrysse, M. Salomon, R. Kuik, and L. N. Vanwassenhove. A dual ascent and column generation heuristic for the discrete lotsizing and scheduling problem with setup times. *Management Science*, 39(4):477–486, 1993.

J. Chevalier, J. Barrier, and P. Richard. Production planning in the glass industry. In *Workshop on Production Planning and Control*, pages 282–285, Mons, 1996.

A. Clark. Multi-period production setup-sequencing and lot-sizing through ATSP subtour elimination and patching. In *Proceedings of the 25th Workshop of the UK Planning and Scheduling Special Interest Group*, pages 80–87, University of Nottingham, 2006.

A. R. Clark and S. J. Clark. Rolling-horizon lot-sizing when set-up times are sequence-dependent. *International Journal of Production Research*, 38(10):2287–2307, 2000.

R. K. Congram, C. N. Potts, and S. L. van de Velde. An iterated dynasearch algorithm for the single-machine total weighted tardiness scheduling problem. *Informs Journal on Computing*, 14(1):52–67, 2002.

M. Constantino. A polyhedral approach to a production planning problem. *Annals of Operations Research*, 96:75–95, 2000.

L. Daheur and E. Jacquet-Lagreze. A glass bottle production scheduling system using a mixed integer programming formulation. In *Workshop on Production Planning and Control*, pages 161–164, Mons, 1996.

S. G. Dastidar and R. Nagi. Scheduling injection molding operations with multiple resource constraints and sequence dependent setup times and costs. *Computers and Operations Research*, 32(11):2987–3005, 2005.

S. Dauzere-Peres and J. Lasserre. On the importance of sequencing decisions in production planning and scheduling. *International Transactions in Operational Research*, 9(6):779–793, 2002.

R. Dematta and M. Guignard. Dynamic production scheduling for a process industry. *Operations Research*, 42(3):492–503, 1994.

R. Dematta and M. Guignard. The performance of rolling production schedules in a process industry. *IIE Transactions*, 27(5):564–573, 1995.

A. Drexl and K. Haase. Proportional lotsizing and scheduling. *International Journal of Production Economics*, 40(1):73–87, 1995.

G. D. Eppen and R. K. Martin. Solving multi-item capacitated lot-sizing problems using variable redefinition. *Operations Research*, 35(6):832–848, 1987.

J. Erschler, G. Fontan, and C. Merce. Consistency of the disaggregation process in hierarchical planning. *Operations Research*, 34(3):464–469, 1986.

EuropeanComission. Best available techniques in the glass manufacturing industry. Technical report, 2001.

B. Fleischmann. The discrete lot-sizing and scheduling problem. *European Journal of Operational Research*, 44(3):337–348, 1990.

B. Fleischmann. The discrete lot-sizing and scheduling problem with sequence-dependent setup costs. *European Journal of Operational Research*, 75(2):395–404, 1994.

B. Fleischmann and H. Meyr. The general lotsizing and scheduling problem. *Or Spektrum*, 19(1):11–21, 1997.

G. Fontan, C. Merce, J. Hennet, and J. LASSERRE. Hierarchical scheduling for decision suport. Technical report, Laboratoire d'Analyse et d'Architecture des Système, 2002.

A. Geoffrion. Lagrangian relaxation for integer programming. *Mathematical Programming Study*, 2:82–114, 1974.

M. Gopalakrishnan. A modified framework for modelling set-up carryover in the capacitated lotsizing problem. *International Journal of Production Research*, 38(14):3421–3424, 2000.

M. Gopalakrishnan, D. M. Miller, and C. P. Schmidt. A framework for modeling setup carryover in the capacitated lot-sizing problem. *International Journal of Production Research*, 33(7):1973–1988, 1995.

M. Gopalakrishnan, K. Ding, J. M. Bourjolly, and S. Mohan. A tabu-search heuristic for the capacitated lot-sizing problem with set-up carryover. *Management Science*, 47(6):851–863, 2001.

S. C. Graves. Using Lagrangian techniques to solve hierarchical production planning problems. *Management Science*, 28:260–275, 1982.

D. Gupta and T. Magnusson. The capacitated lot-sizing and scheduling problem with sequence-dependent setup costs and setup times. *Computers and Operations Research*, 32(4):727–747, 2005.

K. Haase. *Lot sizing and scheduling for Production Planning.* Lecture Notes in Economics and Mathematical Systems. Springer-Verlag, Berlin, 1994.

K. Haase and A. Kimms. Lot sizing and scheduling with sequence-dependent setup costs and times and efficient rescheduling opportunities. *International Journal of Production Economics*, 66(2):159–169, 2000.

P. Hansen and N. Mladenovic. Variable neighborhood search: Principles and applications. *European Journal of Operational Research*, 130:449–467, 2001.

P. Hansen and N. Mladenovic. Variable neighborhood search. In F. Glover and G. Kochenberger, editors, *Handbook of Metaheuristics*, pages 145–184. Kluwer, 2003a.

P. Hansen and N. Mladenovic. A tutorial on variable neighborhood search. Technical report, GERAD, 2003b.

A. Hax and H. C. Meal. *Hierarchical Integration of Production Planning and Scheduling*, volume 1 of *Logistics*. M. A. Geisler Ed, Amsterdam:North Holland, 1975.

D. W. He, A. Kusiak, and A. Artiba. A scheduling problem in glass manufacturing. *IIE Transactions*, 28(2):129–139, 1996.

M. Held, P. Wolfe, and H. Crowder. Validation of subgradient optimization. *Mathematical Programming*, 6:62–88, 1974.

K. S. Hindi. Solving the single-item, capacitated dynamic lot-sizing problem with startup and reservation costs by tabu search. *Computers and Industrial Engineering*, 28(4):701–707, 1995.

K. S. Hindi, K. Fleszar, and C. Charalambous. An effective heuristic for the CLSP with set-up times. *Journal of the Operational Research Society*, 54(5):490–498, 2003.

Instance. Glass container production planning and scheduling instance, 2005. URL `https://www.fe.up.pt/si/conteudos_geral.conteudos_ver?pct_parametros=p_codigo=343486&pct_pag_id=5564`.

R. Jans. Solving lotsizing problems on parallel identical machines using symmetry breaking constraints. Technical report, Erasmus Research Institute of Management, 2006.

R. Jans and Z. Degraeve. An industrial extension of the discrete lot-sizing and scheduling problem. *IIE Transactions*, 36(1):47–58, 2004.

R. Jans and Z. Degraeve. Meta-heuristics for dynamic lot sizing: A review and comparison of solution approaches. *European Journal of Operational Research*, 177(3):1855–1875, 2007a.

R. Jans and Z. Degraeve. Modeling industrial lot sizing problems: a review. *International Journal of Production Research*, doi:10.1080/00207540600902262, 2007b.

R. Jans and Z. Degraeve. New Dantzig-Wolfe remormulation and branch-and-price algorithm for the capacitated lot sizing. *Accepted for publication in Operations Research*, 2007c.

C. Jordan. *Batching and scheduling: models and methods for several problem classes.* PhD, Christian-Albrechts-Universitat, 1996.

C. Jordan and A. Drexl. Discrete lotsizing and scheduling by batch sequencing. *Management Science*, 44(5):698–713, 1998.

S. Kang, K. Malik, and L. J. Thomas. Lotsizing and scheduling on parallel machines with sequence-dependent setup costs. *Management Science*, 45(2):273–289, 1999.

B. Karimi, S. M. T. Fatemi Ghomi, and J. M. Wilson. The capacitated lot sizing problem: a review of models and algorithms. *Omega*, 31(5):365–378, 2003. TY - JOUR.

U. Karmarkar and L. Schrage. The deterministic dynamic product cycling problem. *Operations Research*, 33(2):326–45, 1985.

U. S. Karmarkar, S. Kekre, and S. Kekre. The dynamic lot-sizing problem with startup and reservation costs. *Operations Research*, 35(3):389–398, 1987.

A. Kimms. Competitive methods for multi-level lot sizing and scheduling: Tabu search and randomized regrets. *International Journal of Production Research*, 34(8):2279–2298, 1996.

A. Kimms. A genetic algorithm for multi-level, multi-machine lot sizing and scheduling. *Computers and Operations Research*, 26(8):829–848, 1999. TY - JOUR.

A. Kimms and A. Drexl. Proportional lot sizing and scheduling: Some extensions. *Networks*, 32(2):85–101, 1998a.

A. Kimms and A. Drexl. Some insights into proportional lot sizing and scheduling. *Journal of the Operational Research Society*, 49(11):1196–1205, 1998b.

A. Koclar and H. Sural. A note on “The general lot sizing and scheduling problem”. *Or Spectrum*, 27(1):145–146, 2005.

L. S. Lasdon and R. C. Terjung. Efficient algorithm for multi-item scheduling. *Operations Research*, 19(4):946–969, 1971.

M. J. Liberatore and T. Miller. A hierarchical production planning system. *Interfaces*, 15 (4):1–11, 1985.

J. Maes, J. O. McClain, and L. N. Vanwassenhove. Multilevel capacitated lotsizing complexity and lp-based heuristics. *European Journal of Operational Research*, 53(2):131–148, 1991.

T. L. Magnanti and R. Vachani. A strong cutting plane algorithm for production scheduling with changeover costs. *Operations Research*, 38(3):456–473, 1990.

F. Marinelli, M. E. Nenni, and A. Sforza. Capacitated lot sizing and scheduling with parallel machines and shared buffers: A case study in a packaging company. *Annals of Operations Research*, 150(1):177–192, 2007.

R. Marler and J. Arora. Function-transformation methods for multi-objective optimization. *Engineering Optimization*, 37(6):551–570, 2005.

K. N. McKay and V. C. S. Wiers. Integrated decision support for planning, scheduling, and dispatching tasks in a focused factory. *Computers in Industry*, 50(1):5–14, 2003.

K. N. McKay, F. R. Safayeni, and J. A. Buzacott. A review of hierarchical production planning and its applicability for modern manufacturing. *Production Planning and Control*, 6 (5):384–394, 1995.

C. Merce, G. Hetreux, and G. Fontan. A hierarchical production planning based on an aggregation of time. Technical report, Laboratoire d'Analyse et d'Architecture des Système, 1997.

H. Meyr. Simultaneous lotsizing and scheduling by combining local search with dual reoptimization. *European Journal of Operational Research*, 120(2):311–326, 2000.

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

A. J. Miller and L. A. Wolsey. Tight MIP formulations for multi-item discrete lot-sizing problems. *Operations Research*, 51(4):557–565, 2003.

G. L. Nemhauser and L. A. Wolsey. *Integer and Combinatorial Optimization.* Wiley, New York, 1988.

B. D. Neureuther, G. G. Polak, and N. R. Sanders. A hierarchical production plan for a make-to-order steel fabrication plant. *Production Planning and Control*, 15(3):324–335, 2004.

L. Ozdamar, M. A. Bozyel, and S. I. Birbil. A hierarchical decision support system for production planning (with case study). *European Journal of Operational Research*, 104 (3):403–422, 1998.

R. J. Paul. Production scheduling problem in the glass-container industry. *Operations Research*, 27(2):290–302, 1979.

Y. Pochet and L. A. Wolsey. *Production Planning by Mixed Integer Programming.* Springer Series in Operations Research and Financial Engineering. Springer, New York, 2006.

P. Porkka, A. P. J. Vepsalainen, and M. Kuula. Multiperiod production planning carrying over set-up time. *International Journal of Production Research*, 41(6):1133–1148, 2003.

S. Randhawa and N. Rai. A scheduling decision aid in glass fiber manufacturing. *Computers and Industrial Engineering*, 29(1):255–259, 1995.

A. Respício, M. E. Captivo, and A. J. Rodrigues. A DSS for production planning and scheduling in the paper industry. In D. Age-2002, editor, *International Conference on Decision Making and Decision Support in the Internet Age*, pages 298–308, University College Cork, Cork, Ireland, 2002.

P. Richard. *Contribution des réseaux de Petri à l'étude de problèmes de recherche opérationnelle*. Docteur, Universite Francois Rabelais Tours, 1997.

P. Richard and C. Proust. Maximizing benefits in short-term planning in bottle-glass industry. *International Journal of Production Economics*, 64(1-3):11–19, 2000.

N. Rizk and A. Martel. Supply chain flow planning methods: a review of the lotsizing literature. Technical report, CENTOR, UniversitÃ© Laval, 2001.

M. Salomon, L. G. Kroon, R. Kuik, and L. N. Vanwassenhove. Some extensions of the discrete lotsizing and scheduling problem. *Management Science*, 37(7):801–812, 1991.

M. Salomon, M. M. Solomon, L. N. Van Wassenhove, Y. Dumas, and S. Dauzere-Peres. Solving the discrete lotsizing and scheduling problem with sequence dependent set-up costs and set-up times using the travelling salesman problem with time windows. *European Journal of Operational Research*, 100(3):494–513, 1997. TY - JOUR.

R. A. Sandbothe. A user interactive heuristic procedure for solving the multiple product cycling problem. *Computers and Operations Research*, 23(9):897–907, 1996.

C. Schneeweiss. Hierarchical structures in organizations – a conceptual-framework. *European Journal of Operational Research*, 86(1):4–31, 1995.

C. R. Sox and Y. B. Gao. The capacitated lot sizing problem with setup carry-over. *IIE Transactions*, 31(2):173–181, 1999.

C. Suerie. Campaign planning in time-indexed model formulations. *International Journal of Production Research*, 43(1):49–66, 2005.

C. Suerie. Modeling of period overlapping setup times. *European Journal of Operational Research*, 174(2):874–886, 2006.

C. Suerie and H. Stadtler. The capacitated lot-sizing problem with linked lot sizes. *Management Science*, 49(8):1039–1054, 2003.

V. T'Kindt, J. C. Billaut, and C. Proust. Solving a bicriteria scheduling problem on unrelated parallel machines occurring in the glass bottle industry. *European Journal of Operational Research*, 135(1):42–49, 2001.

W. Trigeiro, L. J. Thomas, and J. McClain. Capacitated lot-sizing with setup times. *Management Science*, 35(3):353–366, 1989.

F. Vanderbeck. Lot-sizing with start-up times. *Management Science*, 44(10):1409–1425, 1998.

E. Vicens, M. E. Alemany, C. Andres, and J. J. Guarch. A design and application methodology for hierarchical production planning decision support systems in an enterprise integration context. *International Journal of Production Economics*, 74(1-3):5–20, 2001.

H. Wagner and T. Whitin. Dynamic version of the economic lot size model. *Management Science*, 5:89–96, 1958.

S. Webster. Remarks on: "Some extensions of the discrete lotsizing and scheduling problem". *Management Science*, 45(5):768–769, 1999.

L. A. Wolsey. Uncapacitated lot-sizing problems with start-up costs. *Operations Research*, 37(5):741–747, 1989.

L. A. Wolsey. MIP modelling of changeovers in production planning and scheduling problems. *European Journal of Operational Research*, 99(1):154–165, 1997.

L. A. Wolsey. Solving multi-item lot-sizing problems with an mip solver using classification and reformulation. *Management Science*, 48(12):1587–1602, 2002.

Appendix A

Examples of DLSP, CSLP and PLSP

We test the standard formulations of DLSP, CSLP and PLSP on the same small data set provided in Table A.1 for a three-product, ten-period problem.

Table A.1: Data for the three product, ten-period problem

	d_{it}												
	$t = 1$	$t = 2$	$t = 3$	$t = 4$	$t = 5$	$t = 6$	$t = 7$	$t = 8$	$t = 9$	$t = 10$	h_i	c_i	p_i
$i = 1$			20				50	90			3.0	200	1
$i = 2$				40			30		80		3.0	200	1
$i = 3$					40					50	3.0	200	1
	C_t												
	60	40	60	50	60	50	60	60	50	60			

We consider null initial inventory ($I_{i0} = 0$).

Tables A.2-A.4 show the optimal solution of the models DLSP, CSLP and PLSP, respectively, for this instance.

Table A.2: DLSP optimal solution: quantity of product i produced in period t (X_{it}) and its optimal value ν.

	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$	$t=6$	$t=7$	$t=8$	$t=9$	$t=10$	ν
$i=1$		40					60	60			
$i=2$				50		50			50		2810
$i=3$					60					60	

Table A.3: CSLP optimal solution: quantity of product i produced in period t (X_{it}) and its optimal value ν.

	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$	$t=6$	$t=7$	$t=8$	$t=9$	$t=10$	ν
$i=1$		20				20	60	60			
$i=2$				40	60				50		2220
$i=3$			40							50	

Table A.4: PLSP optimal solution: quantity of product i produced in period t (X_{it}) and its optimal value ν.

	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$	$t=6$	$t=7$	$t=8$	$t=9$	$t=10$	ν
$i=1$			20			50	60	30			
$i=2$				50	20			30	50		1980
$i=3$					40					50	

Appendix B

Gupta and Magnusson's model

We hereby reproduce the model formulation by Gupta and Magnusson [2005]. Let X_{it} denote the quantity of product i produced in period t, I_{it} the stock of product i at the end of period t and Y_{it} a binary variable that equals one if product i is produced in period t. In addition, the following 0/1 decision variables are defined in [Gupta and Magnusson, 2005]: ψ_{it}, β_{it}, γ_{it}, δ_t, ω_{it} and T_{ijt}. ψ_{it} (β_{it}) equals one if product i is produced first (last) in period t. γ_{it} equals one if the machine is setup for product i at the end of period t. δ_t (ω_t) is defined to be zero (one) if exactly (at least) one product is produced in period t. Finally, T_{ijt} equals one if a setup occurs from product i to product j in period t.

Gupta and Magnusson's model formulation reads:

$$\min \sum_i \sum_j \sum_t c_{ij} \cdot T_{ijt} + \sum_i \sum_t h_i \cdot I_{it} \tag{B.1}$$

$$I_{it} = I_{i(t-1)} + X_{it} - d_{it} \qquad i \in [N], t \in [T] \tag{B.2}$$

$$X_{it} - Y_{it} \leq 0 \qquad i \in [N], t \in [T] \tag{B.3}$$

$$\sum_i X_{it} + \sum_i \sum_j s_{ij} \cdot T_{ijt} \leq 1 \qquad t \in [T] \tag{B.4}$$

$$Y_{it} \leq \omega_t \qquad i \in [N], t \in [T] \tag{B.5}$$

$$\sum_i Y_{it} - 1 \leq (N-1) \cdot \delta_t \qquad t \in [T] \tag{B.6}$$

$$\omega_t \leq \sum_i \psi_{it} \leq 1 \qquad t \in [T] \tag{B.7}$$

$$\omega_t \leq \sum_i \beta_{it} \leq 1 \qquad t \in [T] \tag{B.8}$$

$$\psi_{it} \leq Y_{it} \qquad i \in [N], t \in [T] \tag{B.9}$$

$$\beta_{it} \leq Y_{it} \qquad i \in [N], t \in [T] \tag{B.10}$$

$$\psi_{it} + \beta_{it} \leq 2 - \delta_t \qquad i \in [N], t \in [T] \tag{B.11}$$

$$\sum_i \gamma_{it} = 1 \qquad t \in [T] \tag{B.12}$$

$$\sum_j T_{jit} \geq Y_{it} - \psi_{it} \qquad i \in [N], t \in [T] \tag{B.13}$$

$$\sum_j T_{ijt} \geq Y_{it} - \beta_{it} \qquad i \in [N], t \in [T] \tag{B.14}$$

$$T_{ijt} \geq \gamma_{i(t-1)} + \gamma_{jt} - \omega_t - 1 \qquad i \in [N], j \in [N] \setminus \{i\}, t \in [T] \tag{B.15}$$

$$T_{jit} \geq \psi_{it} + \gamma_{j(t-1)} - 1 \qquad i \in [N], j \in [N] \setminus \{i\}, t \in [T] \tag{B.16}$$

$$T_{ijt} \geq \beta_{it} + \gamma_{jt} - 1 \qquad i \in [N], j \in [N] \setminus \{i\}, t \in [T] \tag{B.17}$$

$$0 \leq \omega_t \leq 1 \qquad t \in [T] \tag{B.18}$$

$$(X_{it}, I_{it}, \delta_{it}) \geq 0 \tag{B.19}$$

$$(T_{ijt}, Y_{it}, \psi_{it}, \beta_{it}, \gamma_{it}) \in \{0, 1\}. \tag{B.20}$$

Appendix C

Analysis of case study instances

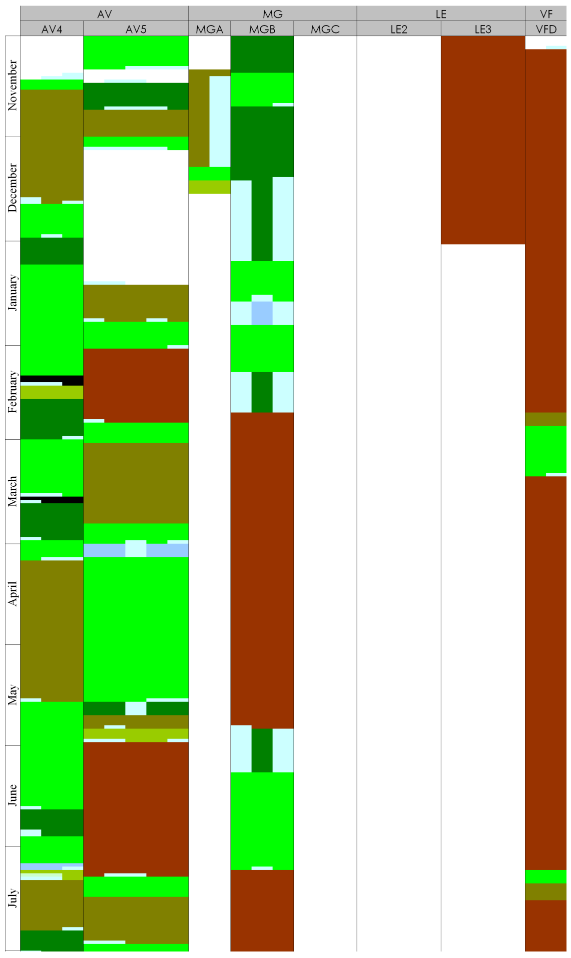


Figure C.1: Part (9 first months) of the Initial Solution SS_3

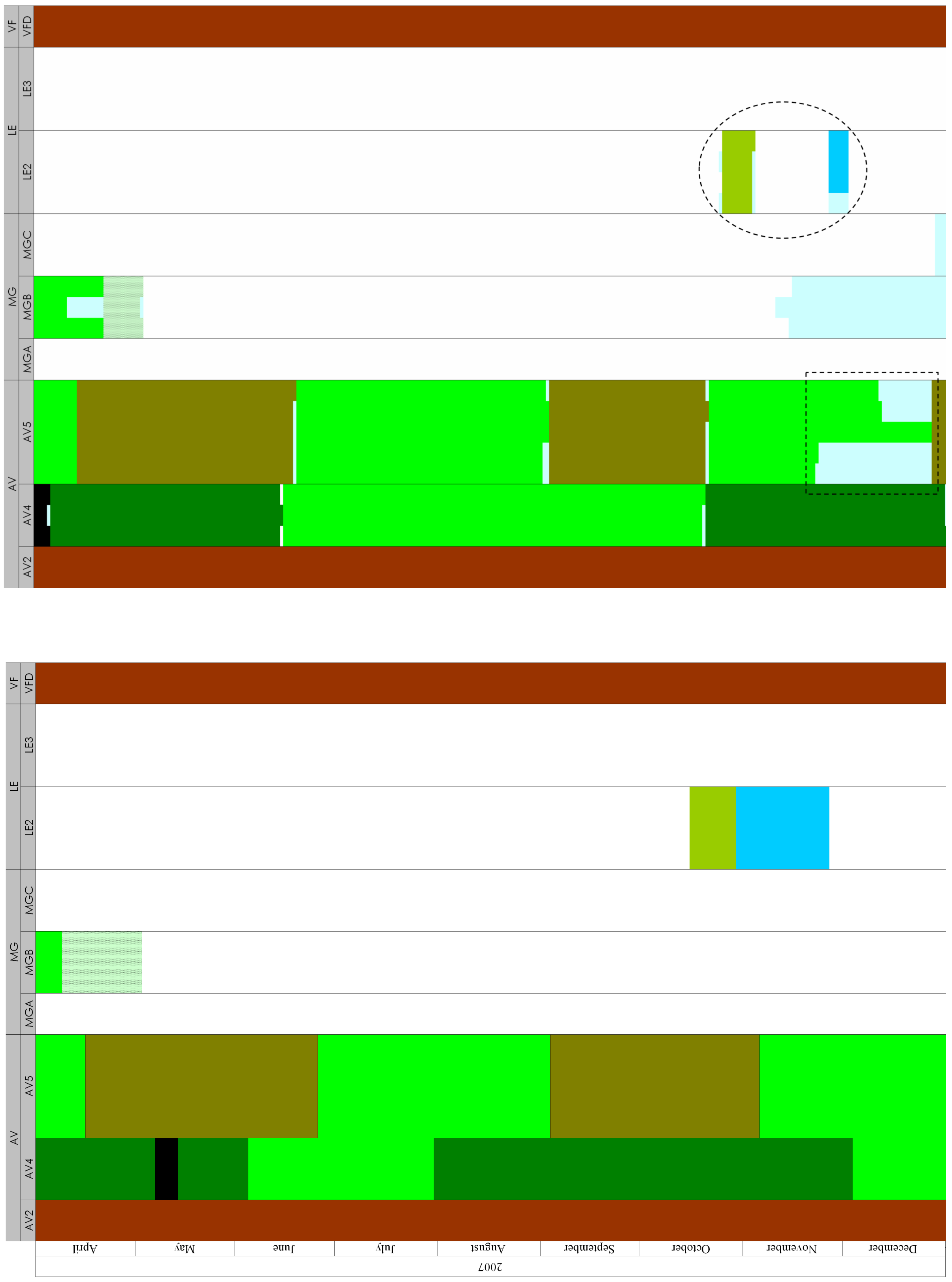


Figure C.2: Real-life (left) and APS (right) plans developed in April 2007

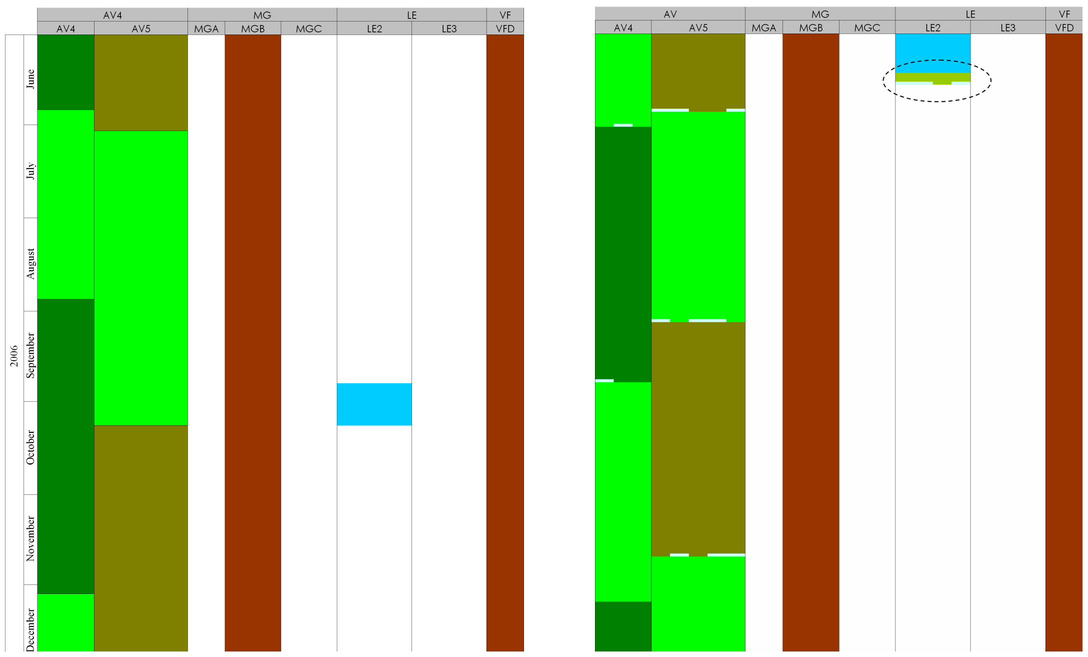


Figure C.3: Real-life (left) and APS (right) plans developed in June 2006

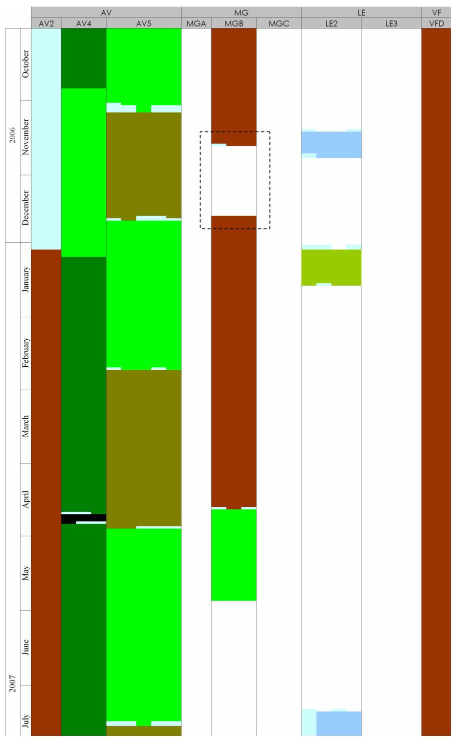


Figure C.4: APS plan for October 2006

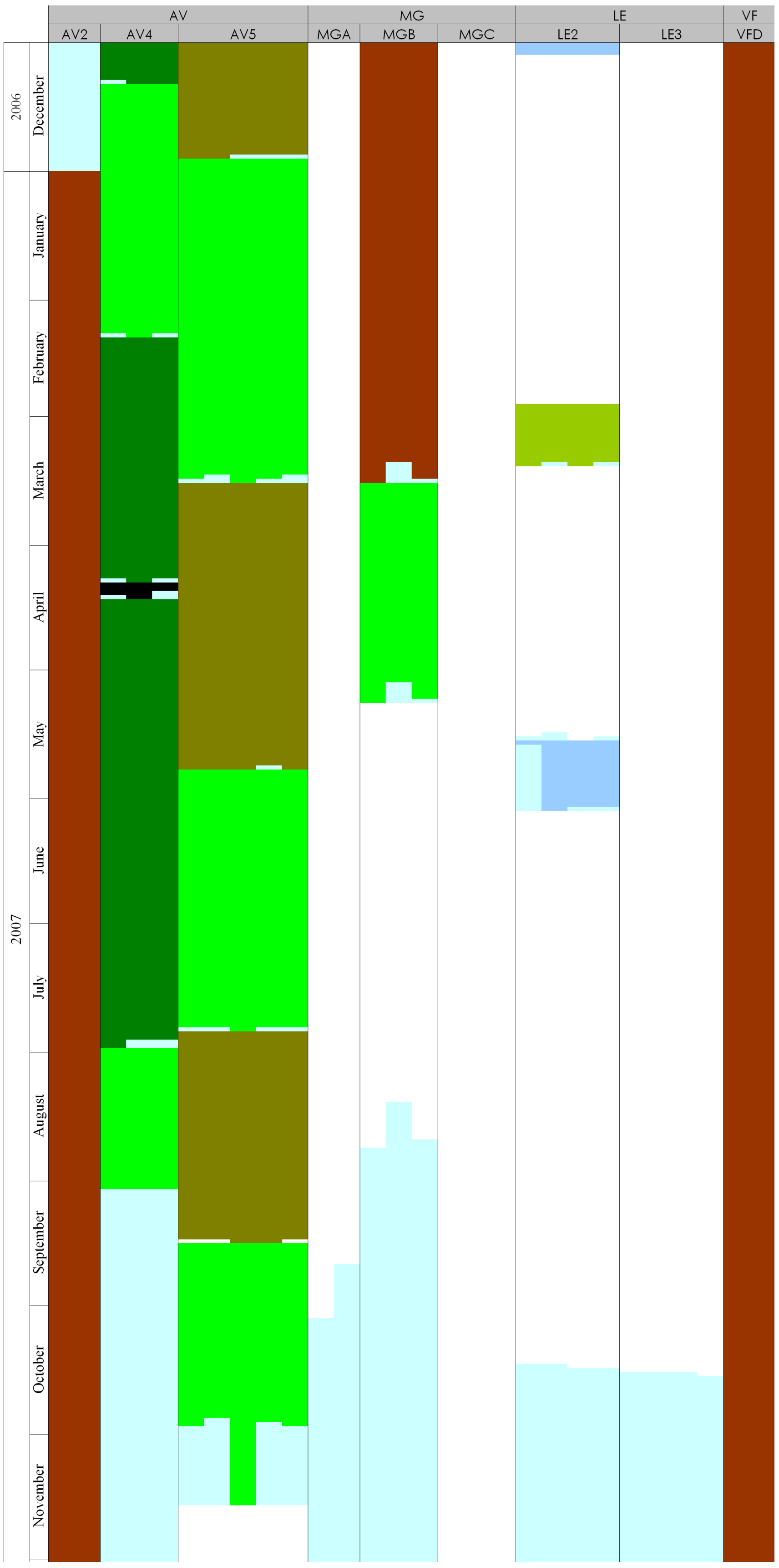


Figure C.5: APS plan for December 2006

Appendix D

Monolithic model for glass container production planning and scheduling

In order to state a monolithic model to the overall glass container production planning and scheduling problem, the planning horizon is decomposed into two sections (Figure D.1):

- the first section has two macro-periods ($t = 1, 2$), each one equivalent to one month. Each macro-period is divided into a variable number ($|S_t|$) of micro-periods (days).
- the second section includes 10 macro-periods ($t = 3, ..., 12$), where each time slot also represents one month.

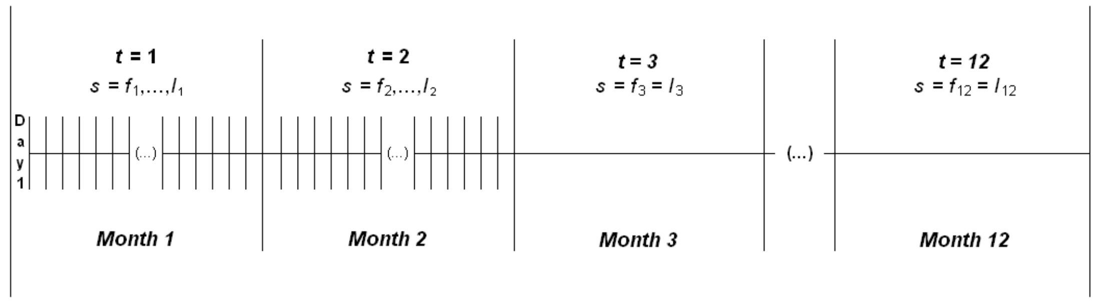


Figure D.1: Partition of the planning horizon

Remember that f_t and l_t represent the first and last micro-periods of macro-period t, respectively. This two-level time structure allows us to integrate in a single model the specific features of both levels described earlier, and does not require detailed information for the entire horizon (e.g. demand figures). We may generalize this framework considering that the planning horizon has T macro-periods, and that the last bucket of the first section corresponds to macro-period t'.

We do not formulate the model here. Such a formulation would integrate the short-term and long-term models stated in chapters 5 and 6, respectively, and would be an extension of the general lotsizing and scheduling problem (GLSP) introduced in Section 2.3. A new set of constraints would need to be added to guarantee that the color that each furnace is ready to process in the beginning of the second section (macro-period $t'+1$) is the same as that of the last product produced in the first section (macro-period t'):

$$\alpha_{uy(t'+1)} \cdot |K_y| = \sum_{i \in F_u} \sum_{k \in K_y} Y^k_{il_{t'}} \qquad u \in [L], y \in [Y]$$

where $\alpha_{uy(t'+1)}$ equals one if the furnace y is set up for color u at the beginning of macro-period $t'+1$ and zero otherwise, $Y^k_{il_{t'}}$ equals one if product i is produced in micro-period $l_{t'}$ on machine k, K_y and $|K_y|$ denote the set and number of machines, respectively, fed by furnace y and F_u denote the set of products with color u.

Naturally, the monolithic model could not be solved easily or implemented in practice due to: a) its very large dimension; b) the need to frequently recompute this MILP due to fluctuations in demand or even sudden operational constraints (e.g. molds not available as foreseen); c) the difficulty in assess different criteria to be used at different organization hierarchical levels; d) the difficulty in assess the impact of “nervousness” that generally occurs when production planning and scheduling is applied in a rolling horizon fashion.