MIP models and exact methods for the Discrete Lot-sizing and Scheduling Problem with sequence-dependent changeover costs and times

Céline Gicquel

ÉCOLE CENTRALE DES ARTS ET MANUFACTURES « ÉCOLE CENTRALE PARIS »

THÈSE

présentée par Céline GICQUEL

pour l'obtention du

GRADE DE DOCTEUR

Spécialité : Génie industriel

Laboratoire d'accueil : Laboratoire de Génie Industriel

SUJET

MIP models and exact methods for the Discrete Lot-sizing and Scheduling Problem with sequence-dependent changeover costs and times

soutenue le 27 octobre 2008

devant un jury composé de :

Pr M. Gourgand, ISIMA	**Président**
Pr S. Dauzère-Péres, Ecole des Mines de Saint-Etienne	**Rapporteur**
Pr L. A. Wolsey, Université Catholique de Louvain	**Rapporteur**
Pr Y. Dallery, Ecole Centrale Paris	**Directeur de thèse**
Pr M. Minoux, Université de Paris 6	**Directeur de thèse**

2008-42

ÉCOLE CENTRALE DES ARTS ET MANUFACTURES « ÉCOLE CENTRALE PARIS »

THÈSE

présentée par Céline GICQUEL

pour l'obtention du

GRADE DE DOCTEUR

Spécialité : Génie industriel

Laboratoire d'accueil : Laboratoire de Génie Industriel

SUJET

MIP models and exact methods for the Discrete Lot-sizing and Scheduling Problem with sequence-dependent changeover costs and times

soutenue le 27 octobre 2008

devant un jury composé de :

Pr M. Gourgand, ISIMA	**Président**
Pr S. Dauzère-Péres, Ecole des Mines de Saint-Etienne	**Rapporteur**
Pr L. A. Wolsey, Université Catholique de Louvain	**Rapporteur**
Pr Y. Dallery, Ecole Centrale Paris	**Directeur de thèse**
Pr M. Minoux, Université de Paris 6	**Directeur de thèse**

2008-42

Remerciements

Pour commencer, je tiens à remercier sincèrement Michel Gourgand, Stéphane Dauzère-Pérès et Laurence Wolsey d'avoir accepté de faire partie de mon jury de thèse. Je suis très honorée de l'intérêt que vous avez porté à ce travail.

Je remercie également très chaleureusement mes deux directeurs de thèse, Yves Dallery et Michel Minoux. Cette thèse n'aurait pas existé sans le soutien et les conseils que vous m'avez prodigués au cours de ces trois années.

Je remercie tous les membres du Laboratoire de Génie Industriel de l'Ecole Centrale Paris qui m'ont accueillie et m'ont permis de travailler quotidiennement dans la convivialité et la bonne humeur. J'ai une pensée particulière pour Anne, Asma, Aude, Bill, Corinne, Oualid, Sylvie et Zied.

Enfin, je n'aurais jamais pu surmonter toutes les difficultés rencontrées sans le soutien inconditionnel et les encouragements de ma famille et de mes amis. Merci à mes parents sans qui tout cela ne serait pas arrivé. Merci en particulier à Erwann qui m'a encouragée alors que cette thèse n'était encore qu'un vague idée et à Mari et Khaled qui en ont suivi au jour le jour les hauts et les bas.

Je pense aussi à Cindy, Jabert... et à tous les jeunes d'A Bras Ouverts. Les week-ends passés à vos côtés m'ont permis de ne pas me perdre dans mes équations et de conserver bien en vue ce qui fait l'essentiel de la vie.

Céline

Abstract

Lot-sizing is one of the many issues arising in the context of production planning. Its main objective is to determine the timing and level of production so as to reach the best possible trade-off between minimizing setup and inventory holding costs and satisfying customer demand. When a limited production capacity and a deterministic time-varying demand rate are assumed, lot-sizing leads to the formulation of large-sized mixed-integer programs, most of which are hard to solve.

In the present work, we deal with one of the many capacitated dynamic lot-sizing models, the Discrete Lot-sizing and Scheduling Problem or DLSP, and study several variants of this problem where changeover costs and/or times are sequence-dependent. We propose various extensions of an existing exact solution approach for the single-level, single-resource DLSP with sequence-dependent changeover costs.

Our contributions concerns both problem modelling and efficient implementation of solution algorithms. In terms of problem modelling, we investigate the integration of various additional relevant industrial concerns into the basic model. More precisely, we consider the following operational aspects: the presence of a multi-attribute product structure which can be exploited to reduce the size of the optimization problem, the integration of positive changeover times to better model the production loss caused by a changeover and the presence of identical parallel resources that need to be planned simultaneously. In terms of algorithmic developments, we present for each of these extensions a solution procedure aiming at providing exact optimal solutions: a tight MIP formulation for the corresponding problem variant is derived and the resulting mixed-integer program is solved thanks to a commercial MIP solver. Moreover, results of extensive computational experiments carried out to evaluate the proposed solution approaches are provided. In general, they show the practical usefulness of the proposed algorithms at solving medium to large-sized instances with a reasonable computational effort.

Keywords: Production planning, Lot-sizing, Sequence-dependent changeover costs and times, Mixed-integer linear programming, Valid inequalities

Résumé

Le dimensionnement des lots de production est une des nombreuses activités survenant dans le cadre de la planification de production. Il a pour objet de déterminer quand et combien produire de façon à réaliser le meilleur compromis possible entre la minimisation des coûts liés à la production (coûts fixes de reconfiguration des ressources, coûts de stockage...) et la satisfaction de la demande des clients. Dans ce travail, nous supposons la capacité de production limitée et la demande des clients connue et variable dans le temps. Dans ce cas, le problème d'optimisation de la taille des lots de production conduit à la formulation de programmes linéaires mixtes en nombres entiers, dont la plupart sont difficiles à résoudre.

Nous nous intéressons ici en particulier à un problème de planification de production par lots connu sous le nom de "Discrete Lot-sizing and Scheduling Problem" ou "DLSP". Plus précisément, nous étudions plusieurs variantes de ce problème dans lesquelles les coûts et/ou les temps de changement de produits sur la ressource sont dépendent de la séquence et nous proposons diverses extensions d'une méthode disponible dans la littérature pour la résolution exacte du problème mono-niveau, mono-resource.

Nos contributions portent à la fois sur la modélisation du problème et sur l'implémentation de méthodes efficaces de résolution.

En ce qui concerne la modélisation, nous étudions l'intégration de diverses aspects opérationnels dans le modèle de base afin d'en améliorer la pertinence industrielle. Ainsi nous considérons les extensions suivantes : la prise en compte d'une structure de produits "multi-attribut" qui permet de diminuer la taille du problème d'optimisation à résoudre, l'intégration de temps de changement positifs afin de mieux modéliser la perte de production causée par une reconfiguration de la ressource et la présence de plusieurs ressources parallèles dont la production doit être planifiée simultanément.

En ce qui concerne la résolution du problème, nous présentons pour chacune des extensions du modèle de base une approche de résolution visant à fournir des solutions optimales exactes. Nous proposons une formulation forte du programme linéaire mixte en nombres entiers correspondant au problème et utilisons un solveur commercial pour le résoudre. De plus, nous fournissons les

résultats de nombreux tests numériques permettant d'évaluer les algorithmes de résolution proposés. En général, ces résultats montrent l'utilité pratique de ces algorithmes pour la résolution d'instances de moyenne et grande taille en des temps de calcul compatibles avec une application industrielle.

Mots-clés : Planification de production, Dimensionnement des lots de production, Coûts et temps de reconfiguration dépendant de la séquence, Ressources parallèles, Programmation linéaire mixte, Inégalités valides

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Nowadays, industrial companies increasingly find that they must rely on effective supply chains to successfully compete in the global market and networked economy. In [86], supply chain management is defined as "the task of integrating organizational units along a supply chain and coordinating materials, information and financial flows in order to fulfil (ultimate) customer demands with the aim of improving competitiveness of the supply chain as a whole". Supply chain management spans all movement and storage of raw materials, work-in-process inventory, and finished goods from point-of-origin to point-of-consumption. The author of [86] describes three building blocks as playing a major role in the efficient coordination of flows throughout a supply chain:

- the use of information and communication technology to instantaneously exchange information such as sales data, forecasts, orders, shipments... between different partners of a supply chain,
- a process orientation within the organization in order to improve cooperation between the business functions and focus the organization on creating value for the customer,
- the use of advanced planning systems (APS), i.e. of optimization softwares able to extend the capabilities of the widely used Enterprise Resource Planning (ERP) and to help solving the various planning problems arising in supply chain management.

The models and methods presented in the present thesis work deals with production planning and are (ultimately) meant to be embedded in optimization softwares such as APS. Therefore we briefly provide in the sequel a short description of advanced planning systems.

According to [38], the role of planning in a supply chain is to support "decision-making by identifying alternatives of future activities and selecting good ones or even the best one". Even if ERP systems are efficient at gathering data and keeping them in synchronization across an organization, they fail at providing good feasible plans for supply chain activities. This is mainly explained by the fact that they rely on traditional planning and scheduling systems (such as Manufacturing Resource Planning) which utilize a stepwise procedure to allocate material and production capacity, plan materials and capacity separately and do not consider limited material availability or capacity constraints. Advanced Planning Systems such as SAP APO, Manugistics, i2... are meant to solve these problems. In contrast to ERP which are mainly transactional systems, APS can be described as analytical systems which use the data stored in ERP systems to provide good feasible plans (see chapter 2 of [78]). These optimization softwares

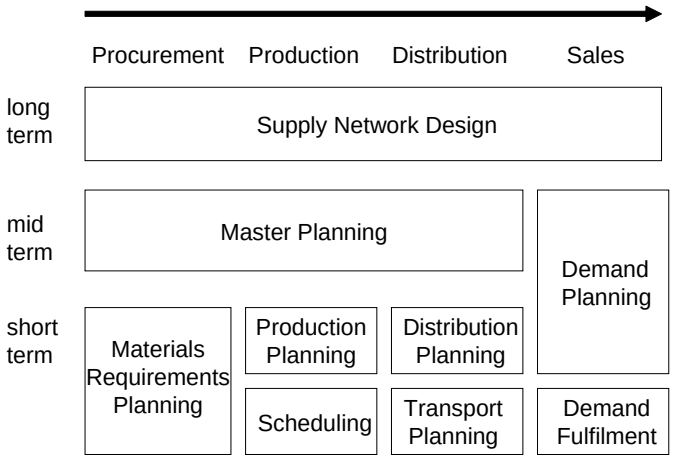

Figure 1.1: Architecture of Advanced Planning Systems: Supply Chain Matrix

are able to reach a good trade-off between financial and customer satisfaction objectives and can thus be useful decision support tools.

There is a huge variety of decisions that need to be made to efficiently plan the supply chain activities. APS rely on a two-dimension classification of these decisions and use the so-called "supply chain matrix" (see figure 1.1) to structure the planning tasks along two axes:

- the level of the managerial decision making involved and the time during which the decision will have an impact on the future development of the supply chain. Following the principles of hierarchical production planning, three planning levels are used: "long-term/strategic", "mid-term/tactical" and "short-term/operationnal".

- the supply chain process involved. Four different processes are identified: procurement, production, distribution and sales.

Among the various modules of the supply chain matrix, the production planning module aims at deciding about the products to be made, the timing and level of the production as well as the resources to be used. It is most often run within each production site and uses a rather high level of detail: individual items are usually considered, time is divided into short periods such as days or shifts, planning is performed for each machine group or flow line that may become a bottleneck. Production planning aims at satisfying the demand assigned by master planning to the considered production site as well as at minimizing production costs and times. Among other decisions, short-term production planning comprises the determination of lot-sizes and the detailed scheduling of the production resources. In case the loading of the resources is strongly affected by the sequence of jobs, both lot-sizing and detailed scheduling should be performed simultaneously.

The purpose of the present work is to discuss such a situation where lot-sizing and scheduling decisions are linked by the presence of sequence-dependent changeover costs and to investigate models and algorithms for solving the resulting discrete optimization problem.

Lot-sizing

Lot-sizing or batching is defined in [64] as "the clustering of items for transportation or manufacturing processing at the same time". Lot-sizing problems arise in production whenever setup times or setup costs are required in order to prepare a resource for the processing of a new product. Setup actions can involve many different operations such as cleaning, preheating, machine adjustments, calibration, inspection, test runs or change in tooling... *Setup costs* or changeover costs account for instance for the additional workforce needed to set up the equipment, for the production loss during the resource downtime and for the raw materials consumed during the setup operations. To minimize setup costs and times and obtain a more efficient use of production resources, production should be run with large batches. However, batching generates cycle inventory as the production cannot be synchronized with the actual demand pattern. Items must be held in stock between the time they are produced and the time they are actually used to satisfy the demand. This generates *inventory holding costs* because of tied up capital, inventory value depreciation and of the cost of storing goods (warehousing, handling...).

Hence the objective of lot-sizing is to reach a good (or even optimal) trade-off between setup costs and inventory holding costs while taking customer satisfaction into account. Making the right decisions in lot-sizing will affect directly the production system performance and its productivity and therefore has a strong impact on the ability of a manufacturing company to compete in the market. Furthermore lot-sizing often leads to difficult optimization problems. This probably explains the existence of a vast amount of academic research dealing with lot-sizing.

In what follows, we give a brief overview of lot-sizing models, relying on the typology proposed by [64] and restricting ourselves to deterministic models. The proposed classification (see table 1.1) is based on two characteristics:

- resource constraints. Models are uncapacitated if capacity constraints on resources are not restrictive or capacitated if capacity constraints are explicitly stated.
- demand rate. Demand can be considered constant or dynamic.

An early attempt of modelling the trade-off between setup and inventory holding costs is the Economic Order Quantity (EOQ) model developed by [48] which assumes a single item with a constant demand rate and an infinite production capacity. The optimal solution of the EOQ

	Infinite capacity	Finite capacity
Constant demand	EOQ	ELSP
Dynamic demand	WW	CLSP, DLSP...

Table 1.1: Typology of lot-sizing models

model is easy to derive, but because of its rather strong assumptions, its practical relevance may be questioned.

A first extension of the EOQ model is the Economic Lot Scheduling Problem (ELSP) where multiple items with a constant demand rate share the same production resource with a limited capacity. In the ELSP, the objective is to find a production schedule which minimizes long-run average cost (see [33] and [105] for literature reviews on the ELSP). Solving the ELSP optimally is NP-hard, thus most solution procedures are heuristic. An important class of policies used in the resolution of the ELSP is based on cyclic production patterns where a production schedule involving all items is designed and repeated periodically.

Another extension of the EOQ is the Wagner-Whitin (WW) problem where the assumption of steady-state demand rate is dropped. In the WW problem, a single item with a dynamic demand has to be produced on a facility with an unlimited capacity. The planning horizon is subdivided into several discrete periods. Demand is given per period and may vary over time. An exact solution procedure based on dynamic programming is presented in [96]. Literature reviews on various extensions of the WW problem were provided by Wolsey (see [97]) and Brahimi, Dauzère-Peres, Najid and Nordly (see [14]).

Finally, capacitated dynamic lot-sizing models combine both complicating features and consider multiple items with a dynamic demand sharing a production resource with a limited capacity. These models result in the formulation of large-sized mixed-integer programs, most of which are hard to solve. Because of this, various solution techniques from the Operations Research field have been proposed to solve them. Surveys on capacitated dynamic lot-sizing models can be found in [64], [30], [54] and [55]. In the present work, we focus on one of the problems belonging to this general class (the Discrete Lot-sizing and Scheduling Problem or DLSP) and propose models and algorithms to solve various extensions of this problem.

1.2 Context

The present work follows a previous study by Nicolas Miègeville for Saint-Gobain Glass (see [72]). Among others, the author of [72] considers the problem of planning the so-called "float lines" used for flat glass production. The most important components of a float line are the furnace where glass is made by the fusion of silica and other components, the floating tin bath and the annealing zone where the continuous ribbon of molten glass is gradually cooled under strictly monitored conditions to give it its physical characteristics, and the cut area where the glass is cut into sheets of various sizes. Floating lines can be described as capacitated, expensive and inflexible resources. For instance, the cost of a new float line can be estimated at around 80 millions of euros. Moreover changeovers on a float line between different types of glass can last as long as a few days, the associated costs being dependent on the production sequence. Decisions about lot-sizing and scheduling are therefore particularly important in this industrial context. In [72], the author proposes an original model of the production system to be planned, aiming mainly at reducing the size of the resulting optimization problem and relying on a standard commercial solver to solve it. In the present work, we build on this previous study and use some of the original modelling ideas presented in it to develop extensions of the specific variant of lot-sizing problems discussed here.

However, in order to be able to solve lot-sizing problems by feeding a mixed-integer programming (MIP) formulation into a standard solver, we have to focus about providing the best possible formulation. Namely, the efficiency of the Branch & Bound procedure embedded in MIP solvers such as CPLEX or XPRESS-MP is highly impacted by the quality of the lower bounds used to evaluate the nodes of the research tree. In the standard Branch & Bound procedure, these lower bounds are provided by the optimal solution of the linear relaxation of the problem, the value of which strongly depends on the way the problem is formulated as a mixed-integer program. Thus, as pointed out by [5], "in spite of the remarkable improvements in the quality of general purpose mixed-integer programming software, the effective solution of a variety of lot-sizing problems depends crucially on the development of tight formulations for the special problem features occurring in practice." Research aiming at improving the MIP formulation of lot-sizing problems through extended reformulations and valid inequalities was initiated in the early 1980s and there now exists a good knowledge about the "right" way to formulate many simple production planning problems. This knowledge, combined with the progress of commercial solvers, enables us to solve problems that were considered out of reach some ten years ago. An introduction on the relevant literature is provided by L.A. Wolsey in [98]. The present work belongs to this line of research. Indeed in our solution approach, we use existing knowledge about

the polyhedral structure of the problem under study and propose several extensions of known tight reformulations and families of valid inequalities in order to take additional operational aspects into account.

1.3 Description and main contributions

In the present manuscript, we discuss a variant of lot-sizing problems known as the Discrete Lot-sizing and Scheduling Problem (DLSP). A detailed description of the DLSP will be given in the sequel but we briefly recall the main features of this model.

The DLSP is a small bucket model: the planning horizon is divided into a rather large number of short periods and during a planning period, at most one type of item can be produced on the resource. Moreover a discrete production policy is assumed, implying that an item, if assigned to a planning period, must be produced at full capacity.

We consider here a complicating operational aspect: the sequence-dependency of changeover costs and times. Changeover costs and times are said to be sequence-dependent when their value depends not only on the item which will be produce after the changeover but also on the item which was produced before the changeover. Sequence-dependent changeover costs and times create additional linkages between items in the MIP formulation and thus make it more difficult to use decomposition methods such as Lagrangian relaxation or column generation to solve the problem. Moreover, additional binary variables and constraints have to be introduced in the MIP formulation, resulting in an increased MIP size.

In [98], Wolsey proposes to solve the DLSP with sequence-dependent changeover costs using a tight MIP formulation and a standard commercial solver. Our main contributions relate to extensions of this work to cases where additional relevant industrial concerns are incorporated in the model:

- For the single-resource DLSP without changeover times, we present a new way of modelling the production system to be planned by properly exploiting *a multi-attribute product structure*. When such a structure is present in the industrial context under study, we suggest to exploit it using an adaptation of the formulation given in [98]. Computational experiments show that thanks to the proposed model, we are able to speed up the resolution of the problem by CPLEX solver for a large class of instances.

- We also consider the integration of *positive changeover times* in the model and develop an extension of the formulation proposed in [98] to take this into account. We provide computational results which indicate that the proposed MIP modelling and reformulation

approach seems to be particularly efficient for the instances with a medium number of items and a low capacity utilization.

- Finally, we discuss the case of *identical parallel resources* and derive a family of strong valid inequalities to deal with this variant. Some insights about the impact of a partial specialization of the resources on both the algorithmic performance and the total production cost are also provided.

1.4 Outline of the thesis manuscript

We now present the general structure of the manuscript. We briefly describe the content of each chapter and (if relevant) give the corresponding paper.

In chapter 2, we review the literature on capacitated lot-sizing models, focusing particularly on recent developments. A classification of the main contributions based on the number of resources and product levels considered in the models is proposed. The corresponding working paper can be found in [42].

In chapter 3, we study the DLSP with a single production resource and zero changeover times and discuss two new ways of modelling the production system to be planned by properly exploiting structures frequently encountered in industrial applications. One of these modelling ideas proves useful at improving the efficiency of a commercial solver to solve medium to large-sized instances. A paper version (see [40]) of this chapter is under review for publication.

In chapter 4, we present a new tight formulation for the DLSP with sequence-dependent changeover costs and changeover times. Our proposal is based on the extension of the tight MIP formulation available for the case without changeover times (see [98]) to the case of positive changeover times. The obtained formulation is then further strengthened thanks to the use of a family of valid inequalities. The results of our computational experiments indicate that the proposed approach is efficient at solving medium to large-sized instances. A paper version (see [43]) of this chapter has been accepted for publication in *Operations Research Letters*.

In chapter 5, we consider the DLSP with identical parallel resources and sequence-dependent changeover costs and present an initial MIP formulation for this specific variant. The presence of parallel resources makes this extension of the DLSP particularly difficult to solve. This is why we discuss a heuristic procedure aiming at providing good approximate solutions for this problem. Although the computational behavior of the proposed algorithm is not really satisfactory, this preliminary study enables us to identify several interesting directions for future research.

In chapter 6, we focus on improving the initial MIP formulation described in chapter 5

and derive a family of valid inequalities for the single-item variant denoted DLS-CC-SC with several machines in [78]. The results of our computational experiments show that thanks to the proposed enhanced formulation, the efficiency of the Branch & Bound procedure embedded in CPLEX solver can be significantly improved. Moreover, our computational experiments provide some insights about the impact of a partial specialization of the resources on both the algorithmic performance and the total production cost.

Chapter 7 provides general concluding remarks and highlights several interesting directions for future research.

Finally, we provide in appendix A the revised version of a paper (see [41]) in which an optimization problem arising in the context of glass production is investigated. This work was carried out as part of our thesis project. However it does not relate to production planning but rather to production line design. We therefore do not include it in the main part of the present manuscript but in the appendix section. This paper is under review for publication.

Chapter 2

A literature review on capacitated lot-sizing models

We discuss one of the many processes arising in the context of supply chain management, namely production planning. We focus on one type of production planning models called capacitated lot-sizing models. These models appear to be well suited for the case where the available production resources are rather inflexible. We review the literature on single-level single-resource lot-sizing models as well as their extensions to multi-level and/or multi-resource problems.

2.1 Introduction

Production planning is the process of determining a tentative plan for how much production will occur in the next time periods, during an interval of time called planning horizon. It is an important challenge for industrial companies because it has a strong impact on their performance in terms of customer service quality and operating costs. However, production planning often proves itself to be a very complex task, mainly for the following reasons:

- Most often a production resource is not fully dedicated to the production of a single product but is rather used to produce different types of product. In many industries, the production resources available are not flexible and can produce only one type of product at a time with a given production rate. Thus a production planner is faced with a competition between products sharing the same production facility and has to decide which products should be produced, when and in which quantities, while taking into account all constraints arising from the production system. In some cases, these constraints can be so tight that even finding a feasible production plan can be very difficult.

- A production plan has to meet several conflicting objectives, namely guaranteeing an excellent customer service level and minimizing production and inventory costs. Thus basic policies like not satisfying the demand exceeding the production capacity or keeping high levels of inventory to be able to meet any demand are usually not commercially acceptable or much too expensive. A good production plan is therefore the result of a trade-off between conflicting objectives.

- A production plan is never fixed for ever. Its validity is restricted to a predefined planning horizon so that at the latest, when reaching the end of the planning horizon, a new plan has to be designed that reflects the current status of the production system. Moreover reality will nearly always deviate from the plan and if the discrepancy between the plan and the actual situation is too large, the plan has to be revised before the end of the planning horizon.

Production planning is thus a difficult and recurring problem for industrial companies and there is a strong need for decision support systems. The development of such decision support systems has been the focus of a large body of the Operations Research literature for the last fifty years and there is now a wide variety of models available for production planning and inventory management.

In the present chapter, we focus on one type of production planning models: capacitated dynamic lot-sizing models. Capacitated lot-sizing models are based on the following assumptions:

- Production resources have a limited capacity and can produce only one type of product at a time. They are rather inflexible, meaning that a significant amount of setup is required to change production from one type of product to another.
- Demand for all products is deterministic and time varying. It has to be satisfied without backlogging, i.e. the production plan should be built so that a perfect customer service level is achieved.
- There are two types of cost to be taken into account:
  - setup costs. Setup costs are the costs incurred when changing the resource configuration from one type of product to another one. A reconfiguration may involve operations such as line clearance, color purging, tool or die changes... Setup costs account for the loss of potential production during the duration of the setup, the additional workforce needed, the additional raw material consumed during the setup...
  - inventory holding costs. Inventory holding costs account for the opportunity costs of capital as well as for the direct costs of storing goods (warehousing, handling...).

To minimize setups costs, production should be run with large batches but at the expense of high inventory costs. On the contrary, inventory levels can be kept low if production of a product is run in frequent and small batches, but at the expense of high setup costs. Thus capacitated lot-sizing models aim at finding a production schedule achieving an optimal trade-off between setup and inventory holding costs, while complying with given capacity constraints and insuring that demand for all products is satisfied without backlogging. Recent overviews on the lot-sizing literature can be found among others in [30], [54], [55] and [98].

The practical relevance of capacitated lot-sizing is supported by the numerous examples of their application in various industries: tile manufacturing [22], tire industry [53], plastic injection molding [21], textile industry ([31], [82]), paper production [45], metallic alloy moulding [29], packaging lines in process industries ([84], [69])... Moreover, as pointed out by [78], multi-level multi-resource lot-sizing models are promising candidates to replace the traditional MRPII logic which provides only suboptimal production schedules.

The purpose of this chapter is to present a general survey on capacitated lot-sizing models. We will review the main contributions to this long standing but active research field, focusing particularly on recent developments.

The complexity of lot-sizing models depends on the features taken into account in the model. As a first step for classification, we use the following characteristics because they strongly impact the complexity of lot-sizing decisions:

- *number of resources.* The products can be made on one single machine (single-resource models) or on multiple machines (multi-resource models). The use of parallel machines complicates the problem as we not only have to determine the timing and level of production, but we also have to assign production lots to machines.

- *number of levels.* Production systems may be single-level or multi-level. In single-level systems, the final products are obtained directly from raw materials after processing by a single operation with no intermediate subassembly. Demand on products is assessed directly from customer orders or market forecasts. In multi-level systems, there is a parent-component relationship between items. Raw materials after processing through several operations change to end products. The output of an operation (level) is an input for another operation. Therefore the demand at one level depends on the lot-sizing decisions made at the parents' level. As a consequence, multi-level problems are more difficult to solve than single-level problems.

- *planning horizon discretization.* Lot-sizing problems can be either big bucket or small bucket problems. Big bucket problems are those where the time period is long enough to produce multiple types of items while for small bucket problems the time period is so short that only one type of items can be produced in each time period.

The chapter is organized as follows. Section 2.2 provides a general review on established single-level single-resource models. In section 2.3, we then discuss single-level multi-resource models. Finally, section 2.4 deals with multi-level extensions of lot-sizing models.

2.2 Single-level single-resource models

In this section, we deal with single-level single-resource models: all products to be made are end items and make use of the same resource with a limited production capacity.

2.2.1 Big bucket models

The capacitated lot-sizing problem (CLSP)

The capacitated lot-sizing problem (**CLSP**) is a typical example of a big bucket problem, where many different items can be produced on the same resource in one time period. The classical CLSP consists in determining the amount and timing of the production of products in the planning horizon: the outcome is a production plan giving for each planning period the quantity (lot size) of each item that should be produced. However detailed scheduling decisions are not

integrated in the CLSP. The usual approach is therefore to solve the CLSP first and to solve a scheduling problem for each period separately afterwards.

In the CLSP, it is required that the resource is setup for a given item in each period where it is produced. The resulting setup costs and times may vary for each item and each period but, as the exact sequence of production within each time period is not defined, they should be *sequence-independent*, i.e. they should not depend on the exact sequence followed to make the products on the resource.

Before going on with the literature review, we briefly present the mixed-integer programming (MIP) formulation for the basic CLSP with zero setup times.

We wish to optimize the production schedule for a set of N items over an horizon featuring T planning periods. A period is indexed by $t = 1, ..., T$, an item by $i = 1, .., N$.

We use the following notation for the parameters:

- d_{it}: deterministic demand (in units) for item i in period t,
- P_t: available production capacity (in time units) on the resource in period t,
- v_{it}: capacity needed (in time units) to produce one unit of i in period t,
- h_i: holding costs per unit and period for item i,
- c_{it}: setup costs for item i in period t.

In the CLSP, the items to be produced can have different production rates on the resource. This is why the production capacity is not expressed as the number of items that can be produced in a planning period, but rather as an available amount of time (P_t) that will be consumed by the produced items with an item-specific production rate (v_{it}).

Decision variables are defined as follows:

- I_{it}: inventory level corresponding to item i at the end of period t,
- x_{it}: production quantity for item i in period t,
- y_{it}: binary setup variables. $y_{it} = 1$ if the resource is setup for item i in period t, and 0 otherwise.

Using this notation, the CLSP can be formulated as a MIP model:
(CLSP)

$$min \sum_{i=1}^{N} \sum_{t=1}^{T} (h_{it} I_{it} + c_{it} y_{it}) \tag{2.1}$$

$$\forall i, \forall t, I_{it} = I_{i,t-1} + x_{it} - d_{it} \tag{2.2}$$

$$\forall i, \forall t, v_{it}x_{it} \leq P_t y_{it} \tag{2.3}$$

$$\forall t, \sum_{i=1}^{N} v_{it}x_{it} \leq P_t \tag{2.4}$$

$$\forall i, \forall t, I_{it} \geq 0 \tag{2.5}$$

$$\forall i, \forall t, x_{it} \geq 0 \tag{2.6}$$

$$\forall i, \forall t, y_{it} \in \{0, 1\} \tag{2.7}$$

The objective, to minimize the sum of inventory holding costs and setup costs, is expressed by (2.1). Constraints (2.2) express the inventory balance. Due to restrictions (2.3), production of an item can only take place if the resource is setup for that particular item. Constraints (2.4) are the capacity constraints. The set of constraints (2.2) and (2.5) ensure that demand for each item is fulfilled without backlogging. Inequalities (2.6) are the non negativity conditions on the production quantities. The binary character of the setup variables is expressed by (2.7).

A recent review on the literature about the CLSP can be found in [59]. The authors classify solution methods into three main categories: exact methods, common-sense or specialized heuristics and mathematical programming-based heuristics.

The use of exact methods to solve the CLSP is described among others in [2], [4], [5], [34] and [98]. The goal of this line of research is to improve the MIP formulation of the problem using reformulations and valid inequalities so that commercial solvers like CPLEX or XPRESS-MP are able to solve practical instances using a standard Branch & Bound type procedure.

Common-sense or specialized heuristics can be found for instance in [27] and [63]. In [27], a first production plan is built using a greedy period-by-period heuristic based on the single-item Silver-Meal approach ([83]). In a second step, this initial plan is modified so that feasibility is guaranteed and costs are reduced. [63] develop a heuristic algorithm using an iterative item-by-item strategy for generating solutions to the problem. In each iteration, a subset of items from those not already scheduled is selected and production schedules over the planning horizon for this set of items are determined. To ensure feasibility of the overall problem, each item is scheduled by solving a bounded single item lot-sizing problem where production capacity is restricted to take into account the production of already scheduled items.

A general drawback of common-sense heuristics is that they can be rather difficult to adapt for different variants or extensions of the problem because in most cases we have to alter the heuristic completely. On the contrary, mathematical programming-based heuristics which use an

optimum seeking mathematical programming procedure to generate a solution are more general and allow for extensions to different problems. Another advantage is that many of these heuristics provide lower bounds on the optimal solution cost, thus providing guidance for the assessment of the quality of the obtained solution. However they usually require much more computational effort for real-world problems and due to their technical concepts cannot be implemented easily by practitioners. Many mathematical programming-based procedures used to solve the CLSP rely upon a Lagrangian relaxation of the capacity constraints. By dualizing capacity constraints into the objective function, the problem decomposes into a series of single item uncapacitated problems, each of which can be solved using an efficient single-item algorithm. This approach is applied among others by [26], [90] and [93]. Some other heuristic solution approaches based on different methods like column generation or metaheuristics can also be found in the literature. The reader is referred to [59] for more details.

Extensions of the CLSP

As mentioned above, in the CLSP, the decision variables are the production quantities of every item in every period, which can be considered as production orders to be released and submitted to the shop floor. This type of model does not involve the lot sequence within a period: this decision has to be determined by an additional scheduling step. However the need for simultaneous lot-sizing and scheduling arises in the case of sequence-dependent setup costs which is frequently encountered in process industries. Therefore recent research has focused on extending the CLSP to incorporate scheduling decisions and deal with sequence-dependent setup costs. This problem is called General Lot-sizing and Scheduling Problem (GLSP) in some papers.

The integration of scheduling decisions in the CLSP formulation can be done in several ways. In [45] and [46], the production sequence within a period is defined through the use of setup state variables giving the resource configuration at the beginning of each period and a series of setup transition variables linked by flow conservation constraints. In both papers, the resulting problem is solved thanks to a specialized heuristic. [37] and [70] use a different approach where each period of the planning horizon is divided into a fixed number of micro-periods with variable length. The production sequence within each period is obtained by assigning an item to each micro-period. Their solution method is based on the use of a local search algorithm called threshold accepting. Finally, in [47], the authors build a predetermined sets of efficient production sequences. In this case, the production planning problem consists in selecting for each planning period a production sequence among those already identified as efficient and in determining the corresponding lot sizes. A tailored enumeration method of the Branch & Bound

type is used to optimally solve medium-sized instances of the problem.

2.2.2 Small bucket models

In small bucket models, the assumption is made that during each time period, at most one type of item can be produced on the resource. Thanks to this assumption, lot-sizing and scheduling decisions can be made simultaneously: namely a unique item is assigned to each planning period and the resulting sequence of item-period assignments defines the production schedule. Note that in small bucket models, the production of a lot may last several periods and setup costs should be incurred in a period only if the production of a new lot begins. To model this, new decision variables often called start-up variables or changeover variables are introduced. In the sequel, we use the binary variable z_{it} to indicate whether the production of a new lot of item i is beginning in period t ($z_{it} = 1$) or not ($z_{it} = 0$).

The Continuous Setup Lot-sizing Problem (CSLP)

A first small bucket model is the so-called Continuous Setup Lot-sizing Problem (**CSLP**). In the CSLP, only one item can be produced by period and the quantity produced can be any value between 0 and the resource capacity.

Using the same notation as in subsection 2.2.1, a MIP model of the CSLP can be stated as follows:

(CSLP)

$$min \sum_{i=1}^{N} \sum_{t=1}^{T} (h_{it} I_{it} + c_{it} z_{it}) \tag{2.8}$$

$$\forall i, \forall t, I_{it} = I_{i,t-1} + x_{it} - d_{it} \tag{2.9}$$

$$\forall i, \forall t, v_{it} x_{it} \leq P_t y_{it} \tag{2.10}$$

$$\forall t, \sum_{i=1}^{N} y_{it} \leq 1 \tag{2.11}$$

$$\forall i, \forall t, z_{it} \geq y_{it} - y_{i,t-1} \tag{2.12}$$

$$\forall i, \forall t, I_{it} \geq 0 \tag{2.13}$$

$$\forall i, \forall t, x_{it} \geq 0 \tag{2.14}$$

$$\forall i, \forall t, y_{it} \in \{0, 1\} \tag{2.15}$$

$$\forall i, \forall t, z_{it} \in \{0, 1\} \tag{2.16}$$

The objective, to minimize the sum of inventory holding costs and startup costs, is expressed by

(2.8). Constraints (2.9) express the inventory balance. (2.10) guarantee that production of an item can only take place if the resource is setup for that particular item and that capacity limits are respected. Constraints (2.11) ensure that only one item may be produced per period. The beginning of a new lot is defined by means of inequalities (2.12). The set of constraints (2.9) and (2.13) ensure that demand for each item is fulfilled without backlogging. Inequalities (2.14) are the non negativity conditions on the production quantities. The binary character of the setup and startup variables is represented by (2.15) and (2.16).

[60] try to solve the CSLP using Lagrangian relaxation applied to the capacity constraints. More recently, [20] presents a cutting-plane approach based on several families of valid inequalities derived for the single-item version of the problem. [95] develops an integer programming column generation algorithm to solve the same problem and uses the cutting-planes proposed by [20] to tighten the formulation of the master linear program at each node of the Branch & Bound tree.

The Discrete Lot-sizing and Scheduling problem (DLSP)

The Discrete Lot-sizing and Scheduling problem (**DLSP**) is another small bucket model. The difference with the CSLP is that a discrete production policy is assumed, implying that an item, if assigned to a planning period, must be produced at full capacity. This "all-or-nothing" assumption is enforced by replacing in the formulation CSLP the inequalities (2.10) by the equalities:

$$\forall i, \forall t, v_{it} x_{it} = P_t y_{it} \tag{2.17}$$

The first contributions on the DLSP use sequence-independent setup costs. [35] solve medium-sized instances using a Branch & Bound procedure where the lower bounds are determined by means of Lagrangian relaxation. [17] describe a heuristic for the DLSP with positive setup times based on dual ascent and column generation techniques. [15] also address the DLSP with sequence-independent setup times and take into account additional operational constraints on batch availability. They develop a two-phase simulated-annealing heuristic to solve their problem and are able to find good solutions for instances involving at most 10 items and 100 periods.

The DLSP with sequence-dependent setup costs is addressed in [36] and [80] who both reformulate the problem as as Travelling Salesman Problem with Time Windows. Studying the same variant, [56] show the equivalence between the DLSP with a single resource and a scheduling problem named Batch Sequencing Problem (BSP) and present a specific Branch & Bound type algorithm to solve the resulting BSP.

	Big bucket models	Small bucket models
Sequence-independent changeover costs	[2], [4], [5], [26], [27], [34], [59], [63], [90], [93]	[15], [17], [20], [35], [60], [67], [66], [73], [94], [95]
Sequence-dependent changeover costs	[37], [45], [46], [47], [70]	[36], [56], [80]

Table 2.1: Literature review: single-level single-resource models

There is also a rather large amount of polyhedral results for the DLSP. Strong valid inequalities for the single-item variant can be found in [67], [66], [73] and [94]. These valid inequalities can be used to tighten the formulation of multi-item instances, thus improving the efficiency of the standard Branch & Bound procedure embedded in commercial solvers. Excellent literature reviews on polyhedral results for the DLSP can be found in [78] and [98].

Table 2.1 summarizes our literature review on single-level single-resource models.

2.3 Single-level multi-resource models

The lot-sizing models presented in the previous section assume that the products are processed on a single resource. However in many cases a manufacturer has access to multiple machines or production lines, which can be used in parallel. In this section, we focus on the single level, parallel resources problem. A recent review on lot-sizing problems involving parallel resources can be found in [52]. As mentioned above, parallel resources further complicate the production planning problem. Namely, as an item can be produced on several machines, there is an additional decision to be made: the assignments of production lots to resources. As for the single-resource models, a distinction can be made between big bucket and small bucket models.

2.3.1 Big bucket models

We first consider extensions of the classical CLSP described in section 2.2.1 to the case of parallel resources. [102] consider a capacitated lot-sizing problem with parallel machines. They assume that a lot cannot be split among several machines so that in a given period, an item can be produced on one machine at most. They develop hybrid heuristics combining local search techniques such as tabu search and a genetic algorithm to deal with the resulting problem. [92] address the same problem and propose a heuristic based on the Lagrangian relaxation of the capacity constraints and subgradient optimization: at each iteration, a series of single-item multi-resource problems are solved using a dynamic programming algorithm. Finally, [52] focuses on the CLSP with parallel identical machines: all the resources have the same available capacity and the setup and production costs are identical on each of the resources. In this case, there exists a large

number of equivalent solutions with the same total cost that differ only by the numbering of the machines. As this degeneracy will slow down the Branch & Bound algorithm, he proposes to add symmetry breaking constraints to the mixed-integer programming formulation in order to obviate this problem.

There are also some papers extending the big bucket models with sequence-dependent setup costs presented in section 2.2.1 to the case of several parallel resources. Among them, [58] propose an original model where the sequence of products produced on a machine in a period is modelled as a collection of subsequences. Each subsequence is made of at most 5 items and by enumeration, an optimal ordering for these items can be found. The lot-sizing problem is then formulated as the problem of assigning a subsequence chosen among those predetermined to a position in the global production schedule on each resource. The authors propose a column-generation approach combined with a Branch & Bound procedure to solve the resulting problem. [19] use a model similar to the one presented in [46] to solve a variant of the CLSP with heterogenous parallel resources and sequence-dependent setup times. The problem is solved heuristically using a rolling-horizon method. While planning production on a rolling horizon basis, only the lot-sizing and sequencing decisions regarding the first periods of the horizon will be actually implemented in the production system. Namely after a few periods, the horizon is rolled forward and the model is applied once more with updated demand, inventory and capacity information. [19] propose to determine precisely the lot-sizing and sequencing decisions only for the first planning periods. The other production decisions for the end of the planning horizon (which will not be actually implemented) are only approximately evaluated, without considering explicitly setup costs and times. This enables them to reduce the size of the mixed-integer program to be solved and thus to save a significant amount of computing time while avoiding some drawbacks arising from a purely myopic approach. In a recent paper, [71] extends his GLSP model to the case of parallel production lines and uses a solution procedure combining local search strategies with dual reoptimization to solve real problems gathered from the consumer goods industry.

2.3.2 Small bucket models

We now present extensions of the small bucket models to the case of multiple parallel resources.

In [84], a first extension of the CSLP is used to plan production on several packaging lines in a process industry. In their model, the authors consider that an item is a combination of a package size and a product to be filled into the packages. They assume that items can be grouped into families: a family can be either a package size or a product according to the industrial application. A major setup will occur if a transition between items belonging to different families has to be

	Big bucket models	Small bucket models
Sequence-independent changeover costs	[52], [102], [92]	[22], [53], [69], [84]
Sequence-dependent changeover costs	[19],[58], [71]	[21], [23], [82]

Table 2.2: Literature review: single-level multi-resource models

carried out whereas the transitions between two items belonging to the same family will lead only to a minor setup. In their model, they impose that only one family can be produced per planning period and focus on defining the exact sequence of family-period assignment. They use a standard Branch & Bound procedure to solve small instances involving 4 products, 5 periods and a single production line. Industrial applications of the CSLP with multiple parallel resources can be found in [21] for the planning of injection molding operations and in [69] for the planning of a yoghurt-packaging facility. In both papers, specialized heuristics are used to solve industrial instances of the problem.

[22] and [23] consider production planning for the curing stage in a tile manufacturing facility. The problem is formulated as a DLSP with heterogenous parallel resources (the curing kilns). They apply Lagrangian relaxation to the inventory balance constraints to decompose the problem into a series of single-resource independent subproblems. Combining this with a subgradient optimization method, they are able to obtain strong lower bounds on the optimal cost. Feasible production schedules are generated from every Lagrangian solution using a so-called product-line assignment heuristic. Another industrial extension of the DLSP can be found in [53] who propose a production planning model for an international tire manufacturer. The problem involves multiple capacitated resources of different types: the molds and the heaters needed to build and cure the tires. It is solved by a column-generation-based algorithm combined with Lagrangian relaxation to reduce the degeneracy of the master problem. [82] solve a DLSP with multiple parallel machines arising in a company producing acrylic fibres using a problem-specific heuristic.

Table 2.2 summarizes our literature review on single-level multi-resource models.

2.4 Multi-level multi-resource big bucket models

In a multi-level lot-sizing problem, the production planning is considered not only for the final level (i.e. the end products), but also for the components and subassemblies needed to make the end products. Because of the parent-component relationship between items, production at one level leads to demand for components at a lower level (dependent demand). At the highest level,

production is triggered by market demand (independent demand).

The parent-component relationship between items, also known as the bill of materials, is usually represented by an acyclic directed network where every node in the network is an item, an arc represents the assembly or distribution relation between items and the weight of an arc is the quantity relation (also called the "gozinto factor") between the two terminal nodes of the arc. Different kinds of product structures can be distinguished:

- serial product structure: each item has a single predecessor and a single successor in the network.
- assembly product structure: each item can be made from several predecessors (i.e. components) but has a single successor (i.e. parent).
- general product structure: each item can be made from several predecessors and can have several successors. Thus there may be several end products that have some components in common: this situation is sometimes referred to as component commonality.

Most contributions on multi-level lot-sizing problem use big bucket models and a general product structure. They can thus be seen as extensions of the classical CLSP described in section 2.2 to the multi-level multi-resource case. This is why we chose to classify the literature with respect to the type of solution approach used rather than with respect to the planning horizon discretization.

We classify solution methods into four main categories: exact methods, specialized heuristics, mathematical programming-based heuristics and metaheuristics.

2.4.1 Exact methods

Most single-level capacitated lot-sizing problems are NP-hard. The multi-level extension makes them even harder because of the interdependency between levels created by the parent-component relationship between items. The demand at lower levels is namely the result of the lot-sizing decisions made at highest levels. Practical instances are often too difficult to be optimally solved with a commercial integer optimization software. Therefore most existing solution approaches are based on heuristic techniques and the literature on exact solution methods to solve multi-level capacitated lot-sizing problems is rather sparse.

Noticeable exceptions can be found in [4], [5], [77], [78] and [98]. These papers are based on the concept of echelon stock. The echelon stock of an item in a given period can be defined as the total stock of this item within the system, whether held directly as stock or as the stock of other items containing one ore more units of this item. The problem can be reformulated using

echelon stock variables and the obtained reformulation can be seen as a series of single-item lot-sizing subproblems linked by capacity constraints. Thanks to this, valid inequalities available for single-item problem can be used. Some computational experiments based on a branch-and-cut procedure using these valid inequalities can be found in [4] and [5] but they are limited to instances involving a single resource.

2.4.2 Specialized heuristics

Several dedicated heuristics have been proposed for solving multi-level extensions of the CLSP. They mainly aim at building a good feasible solution for the problem, but without assessing the quality of the found solution with respect to some lower bounds on the optimal cost.

Most of them follow a level-by-level approach but modify the setup and inventory costs at highest levels to model the interdependencies. [8] study a multi-level CLSP with a serial product structure. They solve the problem by applying sequentially a multi-item single-level specialized heuristic to each level of the problem, beginning with the end products and proceeding through the raw materials. To compensate with this level-by-level myopic approach, before solving the lot-sizing problem at a given level, they modify the setup and inventory costs following the procedure described in [10]. This cost-adjustment approach enables them to (approximately) model the impact of the lot-sizing decisions made at the given level on the lowest levels. A similar approach is used in [89] for general product structures.

Another type of special-purpose heuristic can be found in [18]. The authors study a multi-level CLSP with multiple resources and a general product structure. The starting point for their heuristic is a feasible production plan for the uncapacitated problem. It is obtained by applying sequentially the optimal Wagner-Whitin algorithm to solve single-item single-level problems, beginning with the end items and proceeding to items at the lowest levels. Afterwards, they try to achieve a feasible production plan for the capacitated problem by moving production backwards in time from overloaded periods to earlier underloaded ones while maintaining the feasibility of the plan with respect to demand satisfaction and component availability. With their heuristic, they were able to find good solutions for instances involving 40 items, 2 resources and 12 planning periods.

2.4.3 Mathematical programming-based heuristics

As already mentioned for the single-level single-resource CLSP, mathematical programming based heuristics make use of an optimum seeking mathematical programming methodology and adapt it to generate good feasible solutions for practical instances.

A first example of such an approach can be found in [88]. They propose to solve the multi-level multi-resource CLSP with a general product structure by Lagrangian relaxation applied to both the multi-level inventory balance constraints and the resource capacity constraints. Thanks to this relaxation, the overall problem is decomposed into single-level single-item lot-sizing sub-problems. They use subgradient optimization to update the Lagrangian multipliers and obtain good lower bounds on the cost of an optimal production plan. This procedure is combined with a sophisticated forward and backward scheduling heuristic to transform the obtained unfeasible solutions into good feasible solutions for the initial problem.

A second family of mathematical programming-based approaches involves various Linear Programming relaxations of a MIP formulation of the multi-level multi-resource CLSP. [65] solve a multi-level CLSP with an assembly product structure and several resources, each of them being dedicated to a specific product level. They reformulate the problem using extended production variables and solve the linear programming relaxation of the obtained tightened formulation. They try to build a feasible solution for the initial problem by applying a number of rounding heuristics on the linear programming solution. Their heuristic was tested on instances involving only serial product structures with up to 3 levels. [49] and [61] describe a coefficient-modification heuristic where small LP restrictions of the original problem are repeatedly solved. At each iteration, capacity constraints and objective function coefficients are modified in the linear program to account for the capacity consumed and the costs incurred by the setups on the resource.

[85] proposes to reduce the complexity of the overall MIP model by using a time-oriented decomposition approach leading to the resolution of a series of reduced-sized mixed-integer programs. In this approach, lot-sizing decisions are not made altogether for the entire planning horizon but sequentially, each time for a limited time interval called the lot-sizing window. In each step, setup decisions are made only for the periods within the lot-sizing windows while setup decisions already made for previous periods are taken into account and setup decisions for periods following the lot-sizing window are only approximated through continuous variables. The resulting sub-model whose size is drastically reduced is solved by a commercial solver. Lot-sizing windows are then deployed in internally rolling schedules up to the end of the planning horizon given by the initial decision problem so that a production schedule for the entire horizon is obtained. Their computational experiments show that the proposed heuristic provides a better solution quality than the heuristic found in [88].

	Big bucket models	Small bucket models
Sequence-independent changeover costs	[4], [5], [6], [7], [8], [18], [49], [50], [61], [65], [100], [101], [78], [85], [88], [89], [98], [99]	[4], [5], [77], [78], [98]
Sequence-dependent changeover costs		

Table 2.3: Literature review: multi-level multi-resource models

2.4.4 Metaheuristics

In the past decade, meta-heuristics such as tabu search, simulated annealing and genetic algorithms have become more and more popular for solving complex combinatorial problems. One of the main reasons for their success is their flexibility and ability to handle large and complex problems. Thus these methods seem especially adapted for multi-level extensions of the standard lot-sizing problems. But a major disadvantage is the fact that they do not provide a lower bound to assess the solution quality: it has to be calculated separately. Moreover, although their basic principles are easy to understand, this type of algorithms are in fact fairly complex because of all the special adaptations that are needed to make them work better.

Applications of metaheuristics to solve multi-level lot-sizing problems can be found among others in [6], [7], [50], [62], [101] [100] and [99]. A detailed review on this subject can be found in [54].

Table 2.3 summarizes our literature review on multi-level multi-resource models.

2.5 Conclusion

In the present chapter, we reviewed the literature on single-level single-resource lot-sizing models as well as their extensions to multi-level and/or multi-resource problems. Although research on capacitated lot-sizing started some fifty years ago, lot-sizing problems are still challenging because many extensions are very difficult to solve. This research field thus remains very active.

As mentioned by [55], the research on lot-sizing is currently evolving towards two directions:

- Whereas the early models were usually more compact and captured only the main trade-off, there is now an increased attention to model into more detail specific characteristics of the production system such as sequence-dependent costs, multiple resources, backlogging... The objective is to better represent real life production planning problems and to provide more valuable decision support to managers. The present work belongs to this line of research. Indeed, we investigate various extensions of the single-level, single-resource DLSP

in order to integrate into the basic model additional relevant industrial aspects such as sequence-dependent changeover costs, positive setup times or parallel resources.

- Another new interesting research area deals with the integration of lot-sizing models into more global models in order to better coordinate production and distribution decisions. Examples of integrated production-distribution planning models can be found in [24], [25], [32], [39], [75], [81], [91] and [103].

Finally, solution approaches for such difficult extensions of the lot-sizing problems should be based on previous research. Hybrid optimization procedures combining the strength of different methodologies like MIP formulation strengthening and metaheuristics seem to be a promising research direction.

Chapter 3

The single-resource DLSP without changeover times

We consider the Discrete Lot-sizing and Scheduling Problem. More precisely, we study the variant with a single production resource, sequence-dependent changeover costs and no changeover times. We propose two new ways of modelling the production system to be planned by properly exploiting structures frequently encountered in industrial applications. Using these new modelling ideas, we are able to reduce the size of the mixed-integer program to be solved. Computational results show that, thanks to the formulation exploiting the modelling idea referred to as a "multi-attribute product structure", exact optimal solutions can in general be obtained more efficiently as compared with previously described approaches.

3.1 Introduction

In the present chapter, the Discrete Lot-sizing and Scheduling Problem (DLSP) with a single resource and zero changeover times is considered. As defined by [35], the single-resource DLSP is based on several key assumptions:

- There is one resource (a machine, a production line, ...) with a limited production capacity.
- All items to be produced are end items.
- Demand for products is deterministically known and time-varying.
- The production plan is established for a finite time horizon subdivided into several discrete periods.
- At most one item can be produced per period ("small bucket" model) and the facility processes either one product at full capacity or is completely idle ("all-or-nothing" assumption).
- Costs to be minimized are the inventory holding costs and the changeover costs.

In the DLSP, the changeover costs to be incurred when the production of a new lot begins can depend either on the next item only (sequence-independent case) or on both the previous and the next items (sequence-dependent case). We consider here the (more difficult) case of sequence-dependent changeover costs. In this chapter, we assume zero changeover times. However the integration of positive changeover times is an important extension of this type of models and will be considered in chapter 4.

The DLSP with sequence-dependent changeover costs was studied by [36] and [80]. They both reformulate the problem as a Travelling Salesman Problem with Time Windows and use either Lagrangian relaxation or a dynamic programming-based algorithm to solve it. [56] show the equivalence between the DLSP with sequence-dependent changeover costs and the Batch Sequencing Problem (BSP) and use a specific Branch & Bound type algorithm for solving the BSP to optimality. In these papers, the number of items considered in the computational experiments is relatively small (no more than 10 items) whereas the horizon length can be up to 100 periods. More recently, [98] proposes to strengthen an initial MIP (mixed-integer programming) formulation of the DLSP with sequence-dependent changeover costs using both a reformulation of the changeover variables and valid inequalities. Thanks to this strengthened formulation, the lower bounds provided by the linear relaxation of the problem are significantly better, enabling a Branch & Bound type procedure to solve the problem more efficiently. However, as pointed

out by [5], the large number of variables needed in the reformulation to handle changeovers is an important drawback of this approach.

The purpose of chapter 3 is to study two new ways of modelling the production system to be planned by properly exploiting structures frequently encountered in industrial applications. The proposed models basically aim at reducing the size of the resulting mixed-integer program to be solved by a commercial solver, mainly by eliminating an important fraction of the variables and constraints needed to handle changeovers. This can be achieved by exploiting one of the following ideas:

- The first idea originates from the observation that changeover matrices used in industrial applications often involve few different elements as compared to their size. For instance, a matrix that could involve a hundred different values may involve only ten. This may be explained by the fact that these matrices are evaluated by production experts who tend to first range changeovers by type and to evaluate the cost of each changeover type afterwards. In the model to be presented in section 3.3, this observation is exploited to reduce the size of the problem to be solved by using a *factorization* of the changeover matrices. This can be achieved by aggregating ("factorizing") individual changeovers between pairs of items into a small number of changeover types, each type being defined by a common changeover cost value.

- The second idea originates from the observation that products can usually be described in terms of a set of physical attributes such as color, dimension, quality level... If this is possible, each item to be produced will be identified, not only by a unique index as it is usually done, but also by a M-tuple, each component of which indicates the value of the corresponding attribute for the given item. When such a *multi-attribute product structure* can be exhibited in the industrial context under study, it can also be exploited to reduce the size of the problem to be solved. This can be achieved by looking at changeovers at an aggregate level using the relevant physical attributes instead of considering each individual changeover between items.

Both ideas were first presented and exploited in [72] to solve a production planning problem arising in the float glass manufacturing industry. But these ideas were directly combined together to solve the industrial problem under study and were not evaluated by comparison with other existing approaches. Here we propose to close this gap by :

- considering each idea separately and proposing an improved MIP formulation exploiting it,

- comparing the obtained formulations to a reference formulation found in the literature.

The chapter is organized as follows. In section 3.2, we first introduce a strengthened reformulation proposed by [98] for the DLSP with sequence-dependent changeover costs. In our computational experiments, we use it as a reference for comparison with our models. Our first proposal to model the production system using a factorization of changeover matrices is described in section 3.3. In section 3.4, we present our second proposal based on the use of a multi-attribute product attributes. In both sections, we first recall the modelling idea we exploited and introduce the derived formulations for the DLSP with sequence-dependent changeover costs. Results of computational experiments carried out on a large number of randomly generated instances to evaluate our approaches are then reported. Section 3.5 presents the concluding remarks and discussions for future investigations.

3.2 A strong formulation for the DLSP with sequence-dependent changeover costs

In this section, we first recall a strong formulation for the DLSP with sequence-dependent changeover costs. This formulation was first presented by [60] for the variant of the DLSP referred to as CSLP (Continuous Setup Lot-sizing Problem), where the all-or-nothing assumption is relaxed. More recently, [5] and [98] proposed to use it to solve the DLSP with sequence-dependent changeover costs. We next discuss a further strengthening of this formulation obtained by exploiting valid inequalities proposed by [94]. The use of such a strengthened formulation to solve a pigment sequencing problem involving 10 items and 100 periods is reported in chapter 14 of [78].

3.2.1 Initial formulation

We wish to optimize the production schedule for a set of N items over an horizon featuring T planning periods. A period is indexed by $t = 1, ..., T$, an item by $i = 0, .., N$. We agree to use item $i = 0$ to represent idle periods.

We use the following notation for the parameters:

- d_{it}: demand (in units) for item i in period t,
- P_{it}: production capacity (in units per period) for item i in period t,
- h_i: holding costs per unit and period for item i,
- c_{ij}: changeover costs from item i to item j.

Decision variables are defined as follows:

- I_{it}: inventory level corresponding to item i at the end of period t.
- y_{it}: setup variables. $y_{it} = 1$ if the resource is setup for item i in period t, and 0 otherwise.
- w_{ijt}: changeover variables. $w_{ijt} = 1$ if the resource is switched from item i to item j at the beginning of period t, and 0 otherwise.

With this notation, [5] and [98] propose to formulate the DLSP with sequence-dependent changeover costs as follows:

(DLSP0)

$$min \sum_{i=1}^{N} \sum_{t=1}^{T} h_i I_{it} + \sum_{i=0}^{N} \sum_{j=0}^{N} \sum_{t=1}^{T} c_{ij} w_{ijt} \tag{3.1}$$

$$\forall i, \forall t, I_{it} = I_{i,t-1} + P_{it} y_{it} - d_{it} \tag{3.2}$$

$$\sum_{i=0}^{N} y_{i0} = 1 \tag{3.3}$$

$$\forall i, \forall t, y_{i,t-1} = \sum_{j=0}^{N} w_{ijt} \tag{3.4}$$

$$\forall j, \forall t, y_{jt} = \sum_{i=0}^{N} w_{ijt} \tag{3.5}$$

$$\forall i, \forall j, \forall t, w_{ijt} \geq 0 \tag{3.6}$$

$$\forall i, \forall t, I_{it} \geq 0 \tag{3.7}$$

$$\forall i, \forall t, y_{it} \in \{0, 1\} \tag{3.8}$$

The objective, minimizing the sum of inventory holding costs and changeover costs, is expressed by (3.1). Changeover costs c_{ij} are incurred between two successive production batches of item i and item j, in the first period of production of item j.

Constraints (3.2) express the inventory balance. The "all-or-nothing" assumption is enforced by the term $P_{it} y_{it}$ in the equality: if the resource is setup for i in period t, then all the available capacity is used and the production quantity of item i must be equal to P_{it}. (3.3) is also linked to the "all-or-nothing" assumption: together with constraints (3.4)-(3.5), they ensure that in each period, the resource either produces a single product at full capacity, or is idle (i.e $y_{0t} = 1$).

Equalities (3.4) and (3.5) link the setup variables with the changeover variables. (3.4) guarantee that item i can be produced in period $t-1$ if and only if a changeover from i to another item j (possibly $i = j$) takes place at the beginning of period t. Similarly, (3.5) guarantee that

item j can be produced in period t if and only if a changeover from another item i (possibly $i = j$) to item j takes place at the beginning of period t.

(3.6) state the non-negativity of the changeover variables: observe, as pointed out by [5], that thanks to constraints (3.3)-(3.5) and (3.8), there is no need to define variables w_{ijt} as binary variables. The set of constraints (3.2) and (3.7) ensure that demand for each item is fulfilled without backlogging. The binary character of the setup variables is represented by (3.8).

3.2.2 Strengthening the formulation with valid inequalities

As suggested by [98], the formulation DLSP0 can be further strengthened using a family of strong valid inequalities developed by [94] for the single-item DLSP with Wagner-Whitin costs, constant capacity and no backlogging.

In lot-sizing problems, the expression "Wagner-Whitin costs" refers to the case where the unit production cost u_{it} and the storage cost h_{it} satisfy: $\forall i, \forall t, h_{it} + u_{it} \geq u_{i,t+1}$. This means that if changeover costs are not taken into account, it is more expensive to produce a given item in period t and keep it in stock till the end of period $t+1$ than to produce it in period $t+1$. This condition is often referred to as the absence of speculative motive for early production. Thus, in the presence of Wagner-Whitin costs, the only reasons why a demand is produced before the period where it occurs, are the limited resource capacity and the production fixed costs. In the various models studied in the present work, we do not consider the unit production cost (either because it can be neglected by comparison to other costs or because it is constant throughout the horizon and therefore not subject to optimization). The Wagner-Whitin condition is thus satisfied by all studied instances as we assume positive inventory holding costs (i.e. $\forall i, \forall t, h_{it} \geq 0$). Moreover, when the resource capacity is constant throughout the planning horizon, demand can be measured in terms of how many units can be produced during one production period, i.e. the production capacity and demand quantity can be normalized to one unit per period without loss of generality: $d_{it} \in \{0, 1\}$ and $P_{it} = 1$

We first introduce some additional notation:

- $D_{i,t,\tau}$: cumulated demand for item i in the interval $\{t, ..., \tau\}$. Thanks to the normalization, demand on item i is binary so that $D_{i,t,\tau}$ is equal to the number of positive demand periods for i in $\{t, ..., \tau\}$.

- $S_{i,q}$: q^{th} positive demand period for item i. Note that $S_{i,D_{i,1,t}+q}$ denotes the q^{th} period with positive demand after period t.

We also introduce the start-up variables z^r_{it} defined as:

$$z^r_{it} = \begin{cases} 1 & \text{if the production of a new lot of item } i \text{ begins at period } t, \text{ i.e. if} \\ & \text{a start-up for item } i \text{ takes place at the beginning of period } t, \\ 0 & \text{otherwise.} \end{cases}$$

The start-up variables are linked to the changeover variables by the equations:

$$\forall i, \forall t, z_{it} = \sum_{j:j\neq i} w_{jit} \tag{3.9}$$

With this notation, the following inequations (3.10) are valid inequalities for the DLSP with sequence-dependent changeover costs:

$$\forall t, \forall i, \forall p \in \{0...D_{i,t+1,T}\}, I_{it} \geq \sum_{q=1}^{p} \left(1 - y_{i,t+q} - \sum_{\tau=t+q+1}^{S_{i,D_{i,1,t}+q}} z_{i\tau}\right) \tag{3.10}$$

We briefly explain the underlying idea. First note that $y_{i,t+q} + \sum_{\tau=t+q+1}^{S_{i,D_{i,1,t}+q}} z_{i\tau} = 0$ if and only if the resource is not setup for item i in period $t+q$ and no startup for i takes place between the period $t+q+1$ and the period where the q^{th} demand after period t occurs, i.e. if and only if no production of item i is possible in the interval $\{t+q, ..., S_{i,D_{i,1,t}+q}\}$. In this case, the quantity needed to satisfy the q^{th} demand after period t should be in stock at the end of period t. Thus we see that constraints (3.10) force an increase of the stock of item i at the end of period t by one for each index q for which no production occurs in the interval $\{t+q, ..., S_{i,D_{i,1,t}+q}\}$. The reader is referred to [94] for a detailed proof of the validity of (3.10).

In the computational experiments to be presented in subsections 3.3.4 and 3.4.6, the following cutting-plane generation strategy has been implemented to strengthen the formulation DLSP0:

1. We solve the linear relaxation of the problem using the formulation DLSP0.
2. We check whether each valid inequality of type (3.10) is satisfied. If it is violated by the current continuous solution, we add it to the formulation.
3. If at least one violated inequality is found in step 2, we go back to step 1 and repeat until no more violated valid inequalities can be generated.

The resulting strengthened formulation is denoted DLSP0*.

As pointed out by [5], an important drawback of the formulation DLSP0 is that the number of variables needed in the formulation to handle changeovers, $(N+1)^2T$, grows very rapidly with the problem size. In the sequel, we present two ways to avoid this issue in certain situations, namely when changeover matrices can be factorized or when products can be described as combinations

of a number of physical attributes.

3.3 The DLSP with factorized changeover cost matrices

In this section, we aim at evaluating the idea we refer to as the "factorization" of changeover cost matrices. In most papers dealing with the DLSP with sequence-dependent changeover costs (see [36], [56] and [80]), changeovers between pairs of items are considered individually and modelled one by one in the formulation. This is explained by the fact that it is assumed that the corresponding changeover costs can be precisely evaluated one by one. However, in many cases, changeover matrices are the result of a human evaluation by production experts. Due to the complexity of this task, these experts tend to first range changeovers by type and to evaluate the cost of each changeover type afterwards. As a consequence, the obtained changeover matrices contain only a small number of different values as compared to their size.

This observation was made for the industrial case described in [72]. Another possible application of the "factorization" can be found in the presence of a structure sometimes referred to as "major/minor setup cost structure". In this case, there are only two different values in the changeover costs matrix: a large value corresponding to a major setup and a smaller value corresponding to a minor setup. This situation is described for instance in [36] for the case where there is a natural order of the products (e.g. from light to dark colors) or for the case where there are product families. Similar situations are described in [9], [28] and [84].

In the sequel, this observation is exploited to reduce the size of the problem to be solved by using a *factorization* of the changeover matrices. This can be achieved by aggregating ("factorizing") individual changeovers between pairs of items into a small number of changeover types, each type being defined by a common changeover cost value.

3.3.1 Initial formulation

We now present a formulation for the DLSP with factorized changeover costs. We use the same notation as in section 3.2 for the following parameters:

- d_{it}: demand for item i in t,
- P_{it}: production capacity for item i in t,
- h_i: holding costs per unit and period for i,
- c_{ij}: changeover costs from item i to item j.

We assume that there is a limited number A of changeover types. A is equal to the number of different values found in the changeover cost matrix. Each changeover type $\alpha = 1...A$ corresponds to a strictly positive changeover cost C_α.

We introduce the following notation:

- $S_C(\alpha) = \{(i,j) \text{ s.t. } c_{ij} = C_\alpha\}$ is the subset of pairs of items (i,j) such that the changeover from i to j is a changeover of type α.
- $S_C^1(\alpha) = \{i \in [0...N] \text{ s.t. } \exists j \in [0...N] \text{ s.t. } c_{ij} = C_\alpha\}$ is the subset of items having at least one changeover of type α toward another item.
- $S_C^2(\alpha) = \{j \in [0...N] \text{ s.t. } \exists j \in [0...N] \text{ s.t. } c_{ij} = C_\alpha\}$ is the subset of items having at least one changeover of type α from another item.

We use the following decision variables:

- I_{it}: inventory level corresponding to item i at the end of period t.
- y_{it}: setup variables. $y_{it} = 1$ if the resource is setup for item i in period t, and 0 otherwise.
- $w_{\alpha t}$: changeover variables. $w_{\alpha t} = 1$ if a changeover of type α takes place at the beginning of t, 0 otherwise.

(DLSP1)

$$min \sum_{i=1}^{N} \sum_{t=1}^{T} h_i I_{it} + \sum_{\alpha=1}^{A} \sum_{t=1}^{T} C_\alpha w_{\alpha t} \tag{3.11}$$

$$\forall i, \forall t, I_{it} = I_{i,t-1} + P_{it} y_{it} - d_{it} \tag{3.12}$$

$$\forall t, \sum_{i=0}^{N} y_{it} = 1 \tag{3.13}$$

$$\forall t, \forall \alpha, \forall i \in S_C^1(\alpha), w_{\alpha t} \geq y_{i,t-1} + \sum_{j \text{ st } C_{ij}=C_\alpha} y_{jt} - 1 \tag{3.14}$$

$$\forall t, \forall \alpha, \forall j \in S_C^2(\alpha), w_{\alpha t} \geq \sum_{i \text{ st } C_{ij}=C_\alpha} y_{i,t-1} + y_{jt} - 1 \tag{3.15}$$

$$\forall \alpha, \forall t, w_{\alpha t} \leq \sum_{i \in S_C^1(\alpha)} y_{i,t-1} \tag{3.16}$$

$$\forall \alpha, \forall t, w_{\alpha t} \leq \sum_{j \in S_C^2(\alpha)} y_{jt} \tag{3.17}$$

$$\forall i, \forall t, I_{it} \geq 0 \tag{3.18}$$

$$\forall i, \forall t, y_{it} \in \{0,1\} \tag{3.19}$$

$$\forall \alpha, \forall t, w_{\alpha t} \in \{0,1\} \tag{3.20}$$

The objective, minimizing the sum of inventory holding costs and changeover costs, is expressed by (3.11). Note that changeover costs are not computed for each pair of items as in the formulation DLSP0 but for each changeover type α.

Constraints (3.12) express the inventory balance. Equalities (3.13) ensure that in each period, the resource either produces a single product, or is idle (i.e $y_{0t} = 1$).

Constraints (3.14)-(3.17) link the setup variables with the changeover variables. (3.14) guarantee that there is a changeover of type α at the beginning of period t if an item $i \in S_C^1(\alpha)$ is produced in period $t-1$ and one of the items j such that $C_{ij} = C_\alpha$ is produced in period t. Similarly, (3.15) guarantee that there is a changeover of type α at the beginning of period t if an item $j \in S_C^2(\alpha)$ is produced in period t and one of the items i such that $C_{ij} = C_\alpha$ is produced in period $t-1$. (3.16) and (3.17) ensure that there is a changeover of type α at the beginning of period t only if an item $i \in S_C^1(\alpha)$ is produced in $t-1$ and an item $j \in S_C^2(\alpha)$ is produced in period t.

The set of constraints (3.12) and (3.18) ensure that demand for each item is fulfilled without backlogging. The binary character of the setup and changeover variables is represented by (3.19) and (3.20).

Let us now compare the number of changeover variables in the formulations DLSP0 and DLSP1. For the sake of simplicity, we do not take into account the item $i = 0$ in the comparison. As shown in section 3.2, in this case, the formulation DLSP0 includes N^2T changeover variables, one for each possible pair of items and for each period. When a factorization of the changeover matrices is possible, the proposed formulation DLSP1 includes AT changeover variables. This leads to a significant reduction in the number of variables as A is seen to be much smaller than N^2 in many practical applications.

However, due to the aggregate representation of changeovers used in the formulation DLSP1, it is not possible to express the link between setup and changeover variables using equalities similar to the tight equalities (3.4)-(3.5) used in formulation DLSP0. This is why we had to adapt the formulation proposed in [35] and to link setup and changeover variables using a large number of weaker inequalities. As will be discussed in subsection 3.3.4, this leads to an important loss of efficiency in the solution procedure.

3.3.2 Strengthening the formulation with valid inequalities

As for the formulation DLSP0, the formulation DLSP1 can be further strengthened under the assumption of Wagner-Whitin costs, constant capacity and no backlogging. This can be achieved by extending the inequalities (3.10) to the case of factorized changeover cost matrices.

In order to do this, we introduce the same notation as in section 3.2 for parameters $D_{i,t,\tau}$ and $S_{i,q}$. We also introduce startup variables z_{it} defined as in section 3.2.

The start-up variables are linked to the setup variables by the following set of equations:

$$\begin{cases} \forall i, \forall t, z_{it} \geq y_{it} - y_{i,t-1} \\ \forall i, \forall t, z_{it} \leq y_{it} \\ \forall i, \forall t, z_{it} \leq 1 - y_{i,t-1} \end{cases}$$

Moreover variables z_{it} satisfy the following inequations:

$$\forall i, \forall t, z_{it} \leq \sum_{\alpha \text{ st } i \in S_C^2(\alpha)} w_{\alpha t} \tag{3.21}$$

Namely, if t is the first period of production of a lot of item i, a changeover of type α such that $i \in S_C^2(\alpha)$ must take place at the beginning of period t.

With this notation, we have:

Proposition 3.1 *All feasible solutions of DLSP1 satisfy:*

$$\forall t, \forall i, \forall p \in \{0...D_{i,t+1,T}\}, I_{it} \geq \sum_{q=1}^{p} \left(1 - y_{i,t+q} - \sum_{\tau=t+q+1}^{S_{i,D_{i,1,t}+q}} z_{i\tau}\right) \tag{3.22}$$

Proof 3.1 *The underlying idea is the same as the one used to derive the valid inequalities (3.10) for formulation DLSP0. As constraints (3.10), (3.22) force an increase of the stock of item i at the end of period t by one for each index q for which no production occurs in the interval $\{t+q, ..., S_{i,D_{i,1,t}+q}\}$.*

The reader is referred to [94] for a detailed proof of the validity of (3.10). □

We also derived a second family of valid inequalities in order to obtain a better evaluation of changeover costs in fractional solutions and thus to strengthen the formulation DLSP1.

We first define an additional changeover type $\alpha = 0$ to represent changeovers between distinct items with a zero associated cost (i.e. changeovers between items (i, j) s.t. $i \neq j$ and $C_{ij} = C_0 = 0$) and introduce additional binary variables w_{0t} defined as:

$$w_{0t} = \begin{cases} 1 & \text{if a changeover of type } \alpha = 0 \text{ takes place between two items } i \neq j \\ & \text{at the beginning of period } t, \\ 0 & \text{otherwise.} \end{cases}$$

With this notation, we have:

Proposition 3.2 *All feasible solutions of DLSP1 satisfy:*

$$\forall t = 1...T, \sum_{i=0}^{N} |y_{it} - y_{i,t-1}| = 2 \sum_{\alpha=0}^{A} w_{\alpha t} \tag{3.23}$$

Proof 3.2 *We consider an arbitrary integral feasible solution of DLSP1, say (I, y, w) and an arbitrary time period $t \geq 1$ and we show that the chosen feasible solution satisfies the corresponding valid equality.*

We denote i_0 and i_1 the items for which the resource is setup in period $t-1$ and t respectively. There are two possibilities:

- *if $i_0 = i_1$, the resource is setup for the same item in period $t-1$ and t so that $\forall i, |y_{it} - y_{i,t-1}| = 0$ and $\sum_{i=0}^{N} |y_{it} - y_{i,t-1}| = 0$. There is no changeover at the beginning of period t. Thus $\forall \alpha = 0, ..., A, w_{\alpha t} = 0$ and we have $2 \sum_{\alpha=0}^{A} w_{\alpha t} = 0$.*

- *if $i_0 \neq i_1$, we have: $|y_{i_0 t} - y_{i_0,t-1}| = 1$, $|y_{i_1 t} - y_{i_1,t-1}| = 1$ and $\forall i \notin \{i_0, i_1\}, |y_{it} - y_{i,t-1}| = 0$. As a consequence, $\sum_{i=0}^{N} |y_{it} - y_{i,t-1}| = 2$. There is a changeover from i_0 to i_1 at the beginning of period t so that exactly one of the variables $w_{\alpha t}$ $(\alpha = 0, ..., A)$ equals 1 and $2 \sum_{\alpha=0}^{A} w_{\alpha t} = 2$*

As $\sum_{i=0}^{N} |y_{it} - y_{i,t-1}| = 2 \sum_{\alpha=0}^{A} w_{\alpha t}$ in both cases, this establishes the validity of (3.23). □

Due to the presence of absolute values in their expression, valid equalities (3.23) cannot be used directly to strengthen the formulation DSLP1 but they can be exploited to derive weaker valid inequalities, the expression of which is linear.

Proposition 3.3 *All feasible solutions of DLSP1 satisfy:*

$$\forall t = 1...T, \forall \epsilon \in \{-1; 1\}^{N+1}, \sum_{i=0}^{N+1} \epsilon_i (y_{it} - y_{i,t-1}) \leq 2 \sum_{\alpha=0}^{A} w_{\alpha t} \tag{3.24}$$

Proof 3.3 *We have: $\forall \epsilon_i \in \{-1; 1\}, \epsilon_i (y_{it} - y_{i,t-1}) \leq |y_{it} - y_{i,t-1}|$.*

Thus,

$$\forall \epsilon \in \{-1;1\}^{N+1}, \sum_{i=0}^{N+1} \epsilon_i (y_{it} - y_{i,t-1}) \leq \sum_{i=0}^{N} |y_{it} - y_{i,t-1}|$$
$$\leq 2 \sum_{\alpha=0}^{A} w_{\alpha t}$$

□

Due to their large number, all valid inequalities of type (3.24) cannot be added *a priori* to the formulation. They can however be generated as needed according to a cutting-plane generation strategy. This can be done by using the following separation algorithm:

(SEP) Given (I^*, y^*, w^*) the optimal solution of the linear relaxation of (3.11)-(3.20), for $t = 1...T$:

1. For $i = 0...N$, compute $dif_{it} = y^*_{it} - y^*_{i,t-1}$.
2. Compute e_i using the following rules:
   - if $dif_{it} \geq 0$, $e_i = 1$.
   - if $dif_{it} < 0$, $e_i = -1$.
3. Compute $Dif_t = \sum_{i=0}^{N+1} e_i dif_{it}$.
   - if $Dif_t > 2 \sum_{\alpha=0}^{A} w^*_{\alpha t}$, the valid inequality corresponding to $\epsilon = (e_1, e_2, ..., e_N)$ is violated.
   - else all valid inequalities corresponding to period t are satisfied by the current continuous solution.

If for each time period $t = 1...T$, $Dif_t \leq 2 \sum_{\alpha=0}^{A} w^*_{\alpha t}$, then (I^*, y^*, w^*) satisfies all valid inequalities (3.24), otherwise at least one valid inequalities has been found.

In the computational experiments to be presented in subsection 3.3.4, the following cutting-plane generation strategy has been implemented to strengthen formulation DLSP1:

1. We solve the linear relaxation of the problem using formulation DLSP1.
2. We check whether each valid inequality of type (3.22) is satisfied. If it is violated by the current continuous solution, we add it to the formulation.
3. When no more valid inequality of type (3.22) can be found, we look for violated valid inequalities of type (3.24) using separation algorithm (SEP).
4. If at least one violated inequality is found in steps 2 or 3, we go back to step 1 and repeat

item	0	1	2	3	4	5
inventory holding costs	0	7	8	7	5	5

Table 3.1: Simple example using factorization: inventory holding costs

	0	1	2	3	4	5
0	0	200	200	200	200	40
1	40	0	40	200	200	200
2	40	40	0	200	200	200
3	40	200	40	0	200	200
4	200	40	200	40	0	200
5	200	40	200	200	200	0

Table 3.2: Simple example using factorization: changeover costs matrix

period	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
item 1	1	0	1	0	0	0	0	0	0	0	1	0	0	1	0
item 2	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
item 3	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0
item 4	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1
item 5	0	0	0	0	0	0	0	0	1	1	0	1	0	0	1

Table 3.3: Simple example using factorization: demand on products

until no more violated valid inequalities can be generated.

The resulting strengthened formulation is denoted DLSP1*.

3.3.3 A small illustrative example

We use a very simple example to illustrate the proposed model and to show an application of the formulation DLSP1. We consider a production planning problem involving $N = 5$ items and $T = 15$ periods. We agree to use the item $i = 0$ to describe an idle period. We assume that there are $A = 2$ changeover types. The first changeover type $\alpha = 1$ has an associated cost of $C_1 = 40$ and the second changeover type $\alpha = 2$ corresponds to the changeover cost $C_2 = 200$. Table 3.1 gives the inventory holding costs for each item. The changeover cost matrix can be found in table 3.2. Note that this matrix contains only $A = 2$ distinct values whereas there could be $6 * 5 = 30$ distinct positive values for the matrix coefficients. Table 3.3 provides the demand for each item.

Figure 3.1 shows the optimal production plan obtained while using the formulation DLSP1, the cost of which is $Z^* = 918$. The first line gives the lot schedule and the second line indicates the type of the changeover occurring at the beginning of the corresponding time period.

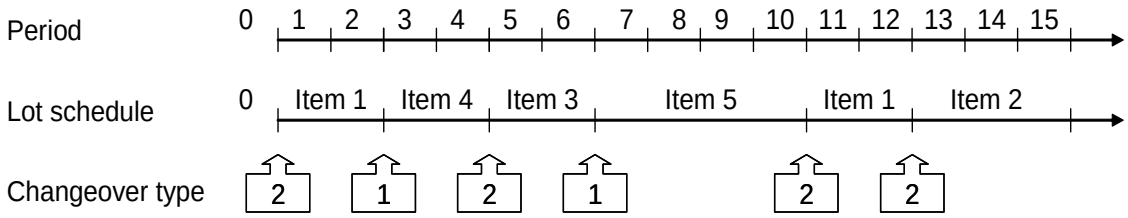

Figure 3.1: DLSP with factorized changeover matrices: optimal production plan for the simple example

3.3.4 Computational results

In this subsection, we discuss the results of some computational experiments carried out to compare the two formulations presented in sections 3.2 and 3.3. We adapted the procedure described in [80] to generate instances of the DLSP with factorized changeover costs matrices and created 3 sets of randomly generated instances

The instances differ with respect to the following characteristics:

- *Problem dimension*: The problem dimension is represented by the number of products N, the number of periods T and the number of changeover types A. We use three different combinations, leading to 3 sets of problem instances:
  - set A: $N = 10$, $T = 60$ and $A = 2$;
  - set B: $N = 25$, $T = 50$ and $A = 2$;
  - set C: $N = 30$, $T = 100$ and $A = 2$.
- *Inventory holding costs*: For each item, inventory holding costs have been generated randomly from a discrete uniform $DU(5, 10)$ distribution.
- *Production capacity utilization*: Production capacity utilization ρ is defined as the ratio of the total cumulated demand on the total cumulated available capacity. We experimented different values for ρ: 0.5, 0.7 and 0.9.
- *Demand pattern*: Binary demand for each product has been randomly generated according to the procedure described in [80].
- *Changeover costs*: Changeover costs C_{ij} have been randomly generated from a uniform distribution on the discrete set $\{C_1, ..., C_A\}$. We tested several possibilities: the values of the various changeover costs can either be chosen to be close to one another or be defined so that there is one changeover type having a corresponding cost significantly higher than the other ones. In our study, we limit our experiments to instances with $A = 2$ and we

measure the relative difference between the two possible values for changeover costs, C_1 and C_2, by the ratio r: $r = C_2/C_1$. We tested several values for r: 1.1, 2, 5, 10 and 30. In all instances, the changeover costs between two items belong to the interval [0,200].

For each possible combination of problem dimension, production capacity utilization and changeover cost ratio, 5 instances were generated, resulting in a total of $3 \times 3 \times 5 \times 5 = 225$ instances. All tests were run on a Pentium 4 (2.8 Ghz) with 505 Mb of RAM, running under Windows XP. We used a standard MIP software (CPLEX 8.1.0) with the solver default settings to solve the obtained mixed-integer programs, using either formulation DLSP0* or formulation DLSP1*.

Tables 3.4-3.6 show the computational results obtained with both formulations, for sets A, B and C respectively. As the value of the ratio r appears to have an impact on the quality of the results, we grouped the instances with respect to the value of r so that each line corresponds to the average value for 15 randomly generated instances (5 instances for each possible value of production capacity utilization). For both series of results, we provide:

- *Variables* and *Constraints*: the number of variables and constraints in the formulation.
- $\#VI$: the average number of valid inequalities of type (3.10) or (3.22) added to the formulation by the cutting-plane generation procedure.
- $\#Opt$: for set A and B instances, the number of instances out of the corresponding 15 instances that could be solved to optimality within 30 minutes of computation.
- $\#Feas$: for set C instances, the number of instances out of the corresponding 15 instances for which a feasible solution could be found within 30 minutes of computation.
- *Gap*: for the instances that could not be solved to optimality, the average gap obtained after 30 minutes of computation between the best integer solution (if one could be found) and the best lower bound obtained.

We now compare the results obtained with formulations DLSP0* and DLSP1*. Results from tables 3.4-3.6 show that:

- For small and medium instances (sets A and B), the results obtained with formulation DLSP0* are much better. Namely, using this formulation, all instances could be solved to optimality within 30 minutes of computation whereas using formulation DLSP1*, only 8% of the instances could be solved to optimality. Moreover, the obtained residual gap for the other instances is large (39% on average).

	Formulation DLSP0*		Formulation DLSP1*	
Variables	8520		2040	
Constraints	1921		6061	
#VI	1211		644	
ratio r	#Opt	Gap	#Opt	Gap
r=1	15	0%	0	53%
r=2	15	0%	0	58%
r=5	15	0%	0	41%
r=10	15	0%	1	28%
r=30	15	0%	5	6%

Table 3.4: Factorized changeover cost matrix: results for set A instances

	Formulation DLSP0*		Formulation DLSP1*	
Variables	36350		3900	
Constraints	3851		11801	
#VI	880		471	
ratio r	#Opt	Gap	#Opt	Gap
r=1	15	0%	0	59%
r=2	15	0%	0	63%
r=5	15	0%	0	48%
r=10	15	0%	0	31%
r=30	15	0%	7	5%

Table 3.5: Factorized changeover cost matrix: results for set B instances

	Formulation DLSP0*		Formulation DLSP1*	
Variables	102200		9300	
Constraints	9201		28101	
#VI	3414		1032	
ratio r	#Feas	Gap	#Feas	Gap
r=1	15	5%	11	66%
r=2	9	13%	11	71%
r=5	14	6%	13	72%
r=10	14	3%	14	59%
r=30	15	5%	15	40%

Table 3.6: Factorized changeover cost matrix: results for set C instances

- For larger instances (set C), the number of instances for which at least one feasible solution could be found within 30 minutes of computation is approximately the same for both formulations. However, the residual gap is much smaller with formulation DLSP0* (6% as compared to 62% with formulation DLSP1*).

These results are mainly explained by the fact that in formulation DLSP1*, weak inequalities (3.14)-(3.15) are used to link setup and changeover variables rather that tight equalities similar to (3.4)-(3.5). As a consequence, in fractional solutions obtained while solving the linear relaxation of formulation DLSP1*, the changeover costs between items are poorly approximated and the quality of the resulting lower bounds is weak. Thus, even if the number of variables introduced in the formulation is drastically reduced while using factorized changeover matrices, it does not compensate for the poor quality of the lower bounds. This leads to a severe loss in the efficiency of the Branch & Bound procedure imbedded in the commercial solver. This study shows that reducing the size of the mixed-integer program to be solved is not enough to make it easier to solve. A good (tight) formulation is critical if we are to obtain good results.

In the sequel, we study another modelling idea: the description of products as combinations of physical attributes. Using this idea, we are able to reduce the size of the obtained mixed-integer program while maintaining the quality of the lower bounds provided by the linear relaxation. Thanks to the combination of these two advantages, it will be shown that the proposed formulation is able to outperform formulation DLSP0* on many instances.

3.4 The DLSP with products described as combinations of physical attributes

In most papers dealing with the DLSP, each individual item to be produced is described with a single index (i in the formulation presented above) and is considered independently of the other items. However, in many industrial situations, the items to be produced are described in terms of a set of physical characteristics or attributes (e.g. color, diameter, size, shape, mixture composition, quality level...). Moreover it is frequently the case that many items share a common value for some attribute so that we can define a (small) finite number of possible values for each attribute. In what follows, we propose to exploit this fact to derive a new formulation for the DLSP which is likely to be solved more efficiently using standard MIP software.

This can be achieved by using an adaptation of the strong DLSP0 formulation described in section 3.2. In the proposed formulation, we look at changeovers at an aggregate level using the relevant physical attributes instead of considering each individual changeover between items. By

doing so, we are able to significantly reduce the number of changeover variables and associated constraints in the formulation, while maintaining the quality of the bounds provided by the linear relaxation of the problem. We extend the approach used by [98] to derive valid inequalities for the resulting mixed-integer linear program. Computational results show that exact optimal solutions can in general be obtained more efficiently with the new model as compared with previously described approaches.

3.4.1 Main assumptions

In order to use the proposed product description as a combination of physical attributes, we need to make several assumptions on the production system.

1. We first suppose that each item to be produced can be described by a set of M physical attributes, each of them takes a finite number of discrete values. We also suppose that each item is uniquely identified thanks to a M-tuple, each component of which gives the value of the corresponding attribute for the given item.

2. Second, we assume that the setup state of the resource can also be described using product attributes. Thus, we will not describe the setup state of the resource by indicating the item that the resource is able to produce, but by indicating, for each attribute, for which value of this attribute the resource is setup. The resource setup state will therefore also be described by a M-tuple, each component of which gives the value of the corresponding attribute for the present state of the resource. To ensure consistency, it should be understood that a given item can be produced on the resource if and only if the resource is setup with the correct value for every attribute.

3. Third, we assume that we are able to evaluate the changeover costs on the resource *for each attribute separately*. This means that given an attribute and two possible values for this attribute, we are able to evaluate the cost of a changeover from one value to the other and that this cost does not depend on the setup state of the resource with respect to the other attributes.

4. Finally, we need to specify how the costs relative to different attributes will combine, i.e. how we will compute changeover costs when changeovers for different attributes happen simultaneously on the resource. We consider here the case where the global changeover costs is the *sum* of all individual changeover costs for the different attributes. Another possible assumption is that global changeover costs equal the *maximum* of the individual costs.

Thanks to these assumptions, we will be able to decide about the production plan on the resource using the product attributes. In this case, the production plan will consist of a set of parallel sequences, one for each attribute. Each of these sequences indicates, for every planning period, for which value of the corresponding attribute the resource is setup. Thus, in each planning period, combining the values for the different attributes, we will be able to deduce the item for which the resource is setup. A detailed mathematical programming formulation is proposed in subsection 3.4.3, but in order to illustrate the usefulness of the new model, we first discuss some industrial situations where it appears to be well suited.

3.4.2 Possible industrial applications

In order to illustrate the practical relevance of the proposed model, we provide examples of industrial situations found in the literature where using physical attributes to describe the products is appropriate.

- In [28] and [84], a production planning problem for a packaging line is considered. For this type of production line, two physical attributes of the products have to be taken into account: the size or shape of the package and the product used to fill it. Hence each item can be described by means of two attributes: the package size/shape and the product to be used. Each individual item would be described by a pair (k_1, k_2) where k_1 is the index of the corresponding package size/shape and k_2 the index of the corresponding product.

- [29] discuss a lot-sizing problem they found in an automated foundry. Each item to be produced can be described by two attributes: the type of metal alloy it is made of and the shape it takes from the used mould. Here we could use as well a pair (k_1, k_2) where k_1 would give the index of the alloy type and k_2 the index of the mould shape.

- [82] study a production planning problem arising in the textile industry in a company producing acrylic fibers. The authors report that two physical characteristics of the products have an impact on the scheduling of the plant spinning unit, namely the fiber composition and their diameter. Thus, we could use a pair (k_1, k_2) to describe each item: k_1 would refer to the fiber composition and k_2 to its diameter.

- [72] considers the production planning problem for a float glass production line. Here each item (a glass sheet) can be described using several physical characteristics: glass color and quality, dimensions of the sheet (thickness, width and length). An item could thus be described using a 5-tuple, with components corresponding to the color, quality and dimensions of the corresponding glass sheet. As mentioned in the introduction of this

chapter, the production system model studied was first suggested in this particular context of application.

Though far from being exhaustive, the above list is a good indication of the wide applicability of the model proposed here.

3.4.3 Initial formulation

We now present a formulation for the DLSP with product attributes and sequence-dependent changeover costs. This formulation can be used to solve the DLSP when a product description using physical attributes is possible and when the assumptions discussed in subsection 3.4.1 hold.

We use the same notation as in section 3.2 for the parameters relative to items:

- d_{it}: demand for item i in t,
- P_{it}: production capacity for item i in t,
- h_i: holding costs per unit and period for i.

We assume that each item can be described using M physical characteristics or attributes. Correspondence between items and attributes is given by a matrix $\mathcal{A}$ of dimensions $M \times (N+1)$. $\mathcal{A}_{mi}$ represents the value of the attribute m for item i and the i^{th} column of $\mathcal{A}$ gives the M-tuple describing item i in terms of product attributes. For each attribute m, we have:

- a set of possible values: $k \in [0, V^m]$
- a changeover cost matrix: $\mathcal{C}^m$. C^m_{kl} is the cost of a transition from the value $k \in [0, V^m]$ to the value $l \in [0, V^m]$ of attribute m.

We agree to use the M-tuple (0,0,....,0) to describe the item $i = 0$: i.e $\forall m, \mathcal{A}_{m0} = 0$.

We use the following decision variables:

- I_{it}: inventory level corresponding to item i at the end of period t.
- y_{it}: setup variables at the item level. $y_{it} = 1$ if the resource is setup for item i in period t, and 0 otherwise.
- w^m_{klt}: changeover variables at the attribute level. $w^m_{klt} = 1$ if a switch from the value k to the value l of attribute m takes place at the beginning of period t, and 0 otherwise.

Under the assumption that changeover costs related to different attributes are added whenever two transitions occur simultaneously (see assumption 4 in subsection 3.4.1), the DLSP can be formulated as follows:

(DLSP2)

$$min \sum_{i=1}^{N} \sum_{t=1}^{T} h_i I_{it} + \sum_{m=1}^{M} \sum_{k=0}^{V^m} \sum_{l=0}^{V^m} \sum_{t=1}^{T} C_{kl}^m w_{klt}^m \tag{3.25}$$

$$\forall i, \forall t, I_{it} = I_{i,t-1} + P_{it} y_{it} - d_{it} \tag{3.26}$$

$$\sum_{i=0}^{N} y_{i0} = 1 \tag{3.27}$$

$$\forall m, \forall k \in [0, V^m], \forall t, \sum_{i \text{ st } \mathcal{A}_{mi}=k} y_{i,t-1} = \sum_{l=0}^{V^m} w_{klt}^m \tag{3.28}$$

$$\forall m, \forall l \in [0, V^m], \forall t, \sum_{i \text{ st } \mathcal{A}_{mi}=l} y_{it} = \sum_{k=0}^{V^m} w_{klt}^m \tag{3.29}$$

$$\forall m, \forall (k,l) \in [0, V^m] \times [0, V^m], \forall t, w_{klt}^m \geq 0 \tag{3.30}$$

$$\forall i, \forall t, I_{it} \geq 0 \tag{3.31}$$

$$\forall i, \forall t, y_{it} \in \{0, 1\} \tag{3.32}$$

The objective, minimizing the sum of changeover costs and inventory holding costs, is expressed by (3.25). Note that inventory holding costs are computed item by item whereas changeover costs are computed attribute by attribute. Constraints (3.26) express the inventory balance. Combined with the non negativity constraints (3.31), they prevent any backlogging. (3.27), together with constraints (3.28)-(3.29), guarantee that in each period the resource either produces a single item or is idle.

Equalities (3.28) and (3.29) link the setup variables with the changeover variables. First note that the term $\sum_{i \text{ st } \mathcal{A}_{mi}=k} y_{it}$ equals 1 if and only if an item i requiring the resource to be setup for the value k of the attribute m is produced in period t, i.e. if and only if the resource is setup for the value k of attribute m in period t. Thus (3.28) guarantee that the resource is setup for the value k of attribute m in period $t-1$ if and only if a changeover from value k to another possible value l of attribute m (possibly $l = k$) takes place at the beginning of period t. Similarly, (3.29) guarantee that the resource is setup for the value l of attribute m in period t if and only if a changeover from another possible value k of attribute m (possibly $k = l$) to value l takes place at the beginning of period t. The non negativity of the changeover variables is stated by (3.30) and the binary character of the setup variables is expressed by (3.32).

The formulation DLSP2 can be easily modified to consider the other possible assumption about the combination of costs relative to different attributes, i.e. the assumption that global changeover costs equal the maximum of the individual costs. This can be done by defining addi-

tional continuous variables C_t to evaluate the changeover costs to be incurred at the beginning of each period t. In this case, the DLSP can be formulated as follows:

(DLSP2 MAX)

$$min \sum_{i=1}^{N} \sum_{t=1}^{T} h_i I_{it} + \sum_{t=1}^{T} C_t$$

$$\text{s.t. } \forall m, \forall t, C_t \geq \sum_{k=0}^{V^m} \sum_{l=0}^{V^m} C_{kl}^m w_{klt}^m$$

$$\text{and } (3.26) - (3.32)$$

In the sequel, we assume that global changeover costs equal the sum of the individual costs and thus use formulation DLSP2. However similar results could be obtained with the other assumption.

Let us now compare the number of changeover variables in the formulations DLSP0 and DLSP2. For the sake of simplicity, we do not take into account the item $i = 0$ in the comparison. As shown in section 3.2, in this case, DLSP0 includes N^2T changeover variables, one for each possible pair of items and for each period. Note that when the product description using attributes is possible, we can compute the number of products as the number of possible combinations obtained by choosing for each attribute m one value out of V^m. Thus we have: $N^2 = \left(\prod_{m=1}^{M} V^m\right)^2$. Now, as can be seen above, in formulation DLSP2, there are $\sum_{m=1}^{M} \left(V^{m2}\right)T$ changeover variables, one for each pair of possible values of each attribute and for each period. In most cases where the product description using attributes will be implemented, we will have: $\sum_{m=1}^{M} \left(V^{m2}\right) \ll \left(\prod_{m=1}^{M} V^m\right)^2$, *thus leading to a significant reduction in the number of changeover variables needed in the formulation.*

In words, in the proposed model, we do not consider each individual changeover between items, but rather look at changeovers at a more aggregate level using product attributes. By doing so, we are able to significantly reduce the size of the mixed-integer linear programm to be solved (e.g. using a Branch & Bound procedure).

3.4.4 Strengthening the formulation with valid inequalities

As for the formulation DLSP0, the formulation DLSP2 can be further strengthened under the assumption of Wagner-Whitin costs, constant capacity and no backlogging. This can be achieved by extending the inequalities (3.10) to the formulation DLSP2. In order to do this, we first define

two new sets of variables:

$$Y_{kt}^m = \sum_{i \text{ st } \mathcal{A}_{mi}=k} y_{it} = \begin{cases} 1 & \text{if the resource is setup for the value } k \text{ of} \\ & \text{attribut } m \text{ in period } t, \\ 0 & \text{otherwise.} \end{cases}$$

$$Z_{kt}^m = \sum_{l \in [0,V^m] \text{ st } l \neq k} w_{lkt}^m = \begin{cases} 1 & \text{if a startup for the value } k \text{ of attribut } m \\ & \text{takes place at the beginning of period } t, \\ 0 & \text{otherwise.} \end{cases}$$

With this notation, we have:

Proposition 3.4 *All feasible solutions of DLSP2 satisfy:*

$$\forall t, \forall i, \forall p \in \{0...D_{i,t+1,T}\}, \forall m = 1...M, I_{it} \geq \sum_{q=1}^{p} \left(1 - Y_{\mathcal{A}_{mi},t+q}^m - \sum_{\tau=t+q+1}^{S_{i,D_{i,1,t+q}}} Z_{\mathcal{A}_{mi},\tau}^m\right) \tag{3.33}$$

Proof 3.4 *Before the proof, which extends the one given in [94] for the formulation DLSP0, we briefly explain the idea underlying (3.33).* $Y_{\mathcal{A}_{mi},t+q}^m + \sum_{\tau=t+q+1}^{S_{i,D_{i,1,t+q}}} Z_{\mathcal{A}_{mi},\tau}^m = 0$ *if and only if the resource is not setup in period* $t+q$ *for the value* $\mathcal{A}_{mi}$ *of attribute* m *needed to produce item* i *and no startup for this value occurs between the period* $t+q+1$ *and the period where the* q^{th} *demand after period* t *occurs, i.e. if and only if no production of item* i *is possible in the interval* $\{t+q, ..., S_{i,D_{i,1,t+q}}\}$. *In that case, the quantity needed to satisfy the* q^{th} *demand on item* i *after period* t *should be in stock at the end of period* t.

Now consider an arbitrary integral feasible solution of DLSP2, say (I, y, w, Y, Z). *We arbitrarily choose an item* i, *a period* t, *a demand occurrence* $p \in \{0...D_{i,t+1,T}\}$ *and an attribute* m *and we show that the chosen feasible solution satisfies the corresponding valid inequality. In the sequel, for the sake of simplicity, we drop the item index* i *and we denote* $k = \mathcal{A}_{mi}$ *the value of attribute* m *for item* i.

We denote R_q *the* q^{th} *production period for this item in the feasible solution considered. By definition, we have* $R_1 < R_2 < ... < R_q < ... < R_{D_{1,T}}$. *Moreover, because backlogging is not allowed, the* q^{th} *production period must occur before the* q^{th} *demand period:* $\forall q, R_q \leq S_q$.

Let q_0 *be the highest index such that* $R_{D_{1,t+q}} < t+q$. *Then we have:*

- $\forall q \leq q_0, R_{D_{1,t+q}} < t+q$. *The* q^{th} *demand after period* t *is produced before period* $t+q$.
- $\forall q > q_0, t+q \leq R_{D_{1,t+q}} \leq S_{D_{1,t+q}}$. *The* q^{th} *demand after period* t *is produced between* $t+q$ *and the period* $S_{D_{1,t+q}}$ *where it occurs. In this case, the resource must be setup for*

the value k of attribute m at least once in the interval $\{t+q, ..., S_{D_{1,t+q}}\}$. Thus we have:

$$\forall q > q_0, Y^m_{ik,t+q} + \sum_{\tau=t+q+1}^{S_{D_{1,t+q}}} Z^m_{k,\tau} \geq 1 \tag{3.34}$$

Hence,

$$\begin{aligned}
&\sum_{t=1}^{t} y_t + \sum_{q=1}^{p} \left(Y^m_{k,t+q} + \sum_{\tau=t+q+1}^{S_{D_{1,t+q}}} Z^m_{k,\tau} \right) \\
&\qquad \geq \sum_{t=1}^{t} y_t + \sum_{q=1}^{q_0} Y^m_{k,t+q} + \sum_{q=q_0+1}^{p} \left(Y^m_{k,t+q} + \sum_{\tau=t+q+1}^{S_{D_{1,t+q}}} Z^m_{k,\tau} \right) && (3.35) \\
&\qquad \geq \sum_{t=1}^{t} y_t + \sum_{q=1}^{q_0} y_{t+q} + p - q_0 && (3.36) \\
&\qquad \geq D_{1,t} + q_0 + p - q_0 && (3.37) \\
&\qquad \geq D_{1,t} + p
\end{aligned}$$

(3.35) comes from the fact that $\sum_{q=1}^{q_0} \sum_{\tau=t+q+1}^{S_{D_{1,t+q}}} Z^m_{k,\tau} \geq 0$. To obtain (3.36), we use the fact that $y_{t+q} \leq Y^m_{k,t+q}$ as well as the inequalities (3.34). Finally, (3.37) is true because, by definition of q_0, the cumulated demand $D_{1,t} + q_0$ is satisfied by the cumulated production before $t + q_0$, $\sum_{t=1}^{t+q_0} y_t$, so that $\sum_{t=1}^{t+q_0} y_t \geq D_{1,t} + q_0$.

As $\sum_{t=1}^{t} y_t - D_{1,t}$ is the inventory level of item i at the end of period t, this establishes the validity of (3.33). □

The number of valid inequalities (3.33) grows quite fast with the problem size and the production capacity utilization: e.g. for the instances involving 30 products and 100 periods with a capacity utilization of 90% (see subsection 3.4.6), there are more than 21000 valid inequalities (3.33) for formulation DLSP2. Hence it is not possible to include directly all valid inequalities in formulation DLSP2. In the computational experiments to be presented in subsection 3.4.6, the following cutting-plane generation strategy has been implemented to strengthen formulation DLSP2:

1. We solve the linear relaxation of the problem using formulation DLSP2.

2. We check whether each valid inequality of type (3.33) is satisfied. If it is violated by the current continuous solution, we add it to the formulation.

3. If at least one violated inequality is found in step 2, we go back to step 1 and repeat until no more violated valid inequalities can be generated.

The resulting strengthened formulation is denoted DLSP2*.

item	0	1	2	3	4
attribute 1: bottle size	0	1	1	2	2
attribute 2: liquid composition	0	1	2	1	2
product description	(0,0)	(1,1)	(1,2)	(2,1)	(2,2)

Table 3.7: Simple example of multi-attribute product structure: product description

0	1	2
0	100	200
1	0	200
2	100	0

Attribute 1

0	1	2
0	10	10
1	0	20
2	10	0

Attribute 2

Table 3.8: Simple example of multi-attribute product structure: changeover costs

period	1	2	3	4	5	6	7	8	9	10
item 1	0	1	0	0	1	0	0	1	0	0
item 2	0	0	0	0	0	0	0	0	0	1
item 3	0	0	0	0	1	1	0	1	0	1
item 4	0	0	0	1	0	0	0	0	0	0

Table 3.9: Simple example of multi-attribute product structure: demand on products

3.4.5 A small illustrative example

We use a very simple example to illustrate the proposed model and to show an application of the formulation DLSP2. We consider a bottle filling line where 4 items can be produced. An item is described by the corresponding bottle size (attribute 1 with two possible values) and the composition of the liquid to be used (attribute 2 with two possible values). Table 3.7 shows how each of the 4 items can be described using the two attributes. We agree to use the item $i = 0$ described by the pair (0,0) for the idle period. Table 3.8 gives the changeover costs for each attribute and table 3.9 provides the demand for each product.

Figure 3.2 shows the optimal production plan obtained while using formulation DLSP2*. The first two lines give the sequence of setup states for each attribute. In each planning period, we can deduce from these sequences the item for which the resource is setup. The changeover costs to be incurred between each lot are shown below. We used the assumption that changeover costs relative to different attributes are added whenever changeovers for different attributes occur simultaneously. This is the case here at the beginning of periods 1, 4, 9 and 10 where both the bottle size and the liquid composition are changed.

Before going on with the computational results, we briefly explain with this simple example how, for each attribute m, the equalities (3.28)-(3.29) can be seen as flow conservation constraints in a network. Namely, as pointed out by [5], the definition of a production plan can be seen

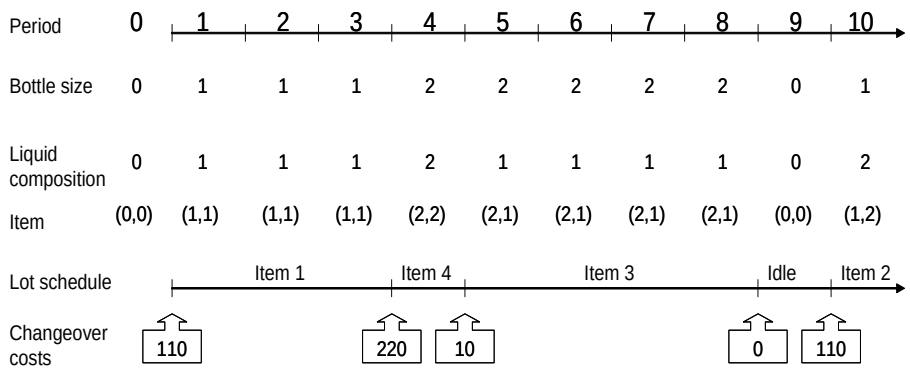

Figure 3.2: DLSP with a multi-attribute product structure: optimal production plan for the simple example

as a problem of defining a single unit flow in a network under additional constraints. With this interpretation, equalities (3.4)-(3.5) of formulation DLSP1 can be seen as flow conservation constraints in the corresponding network.

Here we consider, for a arbitrarily chosen attribute m, the graph $\mathcal{G}^m = (\mathcal{V}^m, \mathcal{E}^m)$. A node $v \in \mathcal{V}^m$ corresponds to a pair (k,t) where $k \in [0, V^m]$ is a possible value for attribute m and $t \in [0,T]$ is a time period. There is an oriented arc $a \in \mathcal{E}^m$ from node v_1 to node v_2 if and only if $v_1 = (k,t)$ et $v_2 = (l, t+1)$. The setup variable at the attribute level Y^m_{kt} corresponds to the flow through node (k,t) and the changeover variables $w^m_{kl,t+1}$ corresponds to the flow between node (k,t) and node $(l,t+1)$. With this interpretation, a production sequence on the resource for attribute m corresponds to a flow of a single unit through graph $\mathcal{G}^m$, starting from a node $(k,0)$ (initial setup state of the resource with respect to attribute m) and arriving in a node (l,T) (final setup state of the resource with respect to attribute m). Thus equalities (3.28) can be seen as flow conservation constraints, stating that the flow trough node $(k,t-1)$ is equal to the sum of the flows on the arcs directed away from this node. Similarly, equalities (3.29) can be seen as flow conservation constraints, stating that the flow through node (l,t) is equal to the sum of the flows on the arcs directed toward this node.

Figure 3.3 shows, for both attributes, the interpretation of the optimal production plan for the illustrative example as a single unit flow in the corresponding networks. For the sake of simplicity, only the arcs with a positive flow are shown.

3.4.6 Computational results

In this subsection, we discuss the results of computational experiments carried out to compare the two formulations presented in sections 3.2 and 3.4. We created 5 sets of randomly generated instances. The instances differ with respect to the following characteristics:

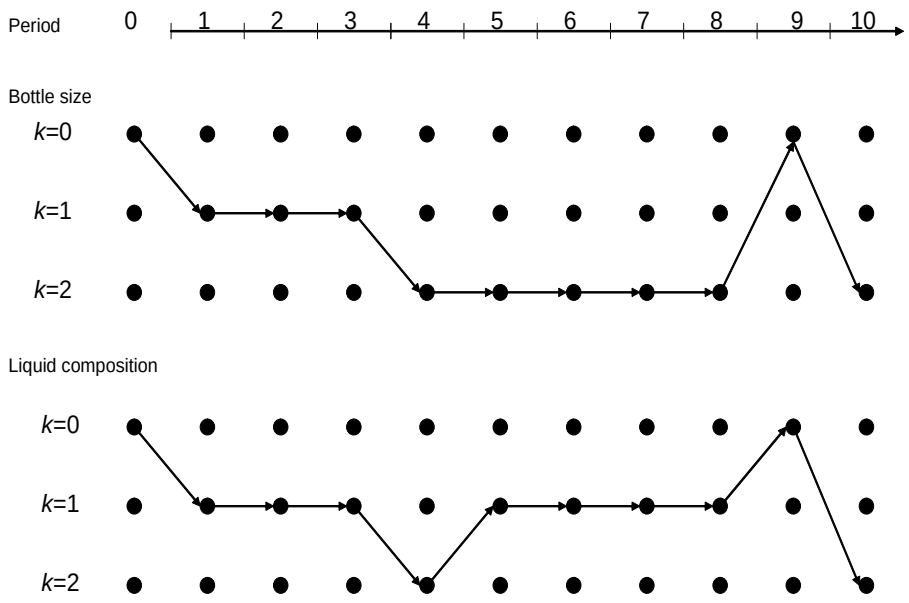

Figure 3.3: DLSP with a multi-attribute product structure: interpretation of the production plan for the simple example as a single unit flow in networks

	N	T	M	V^m
set A	10	60	2	$V^1 = 2, V^2 = 5$
set B	25	50	2	$V^1 = 5, V^2 = 5$
set C	25	50	3	$V^1 = 3, V^2 = 3, V^3 = 3$
set D	30	100	3	$V^1 = 2, V^2 = 3, V^3 = 5$
set E	30	100	5	$V^1 = 2, V^2 = 2, V^3 = 2, V^4 = 2, V^5 = 2$

Table 3.10: Multi-attribute product structure: characteristics of generated instances

- *Problem dimension*: The problem dimension is represented by the number of products N and the number of periods T. We use three different combinations:

  $(N, T) \in \{(10, 60), (25, 50), (30, 100)\}$.

- *Multi-attribute product structure*: The product structure is described by the number of attributes M and the number of possible values V^m for each attribute m. We use five different combinations, leading to 5 sets of instances. Table 3.10 gives the characteristics of the generated instances for each set.

  For set C and E instances, we have $\prod_{m=1}^{M} V^m > N$. Therefore we used the following procedure to generate matrix $\mathcal{A}$:

  1. We generated a matrix $\mathcal{A}'$ with $\prod_{m=1}^{M} V^m$ columns. $\mathcal{A}'$ describes all possible combinations of the attribute values.

  2. For each column i of $\mathcal{A}'$, we randomly generated a weight w_i from a discrete uniform $DU(1, \prod_{m=1}^{M} V^m)$ distribution.

3. The matrix $\mathcal{A}$ is generated by selecting the N columns of $\mathcal{A}'$ corresponding to the N smallest weight w_i.

- *Inventory holding costs*: For each item, inventory holding costs have been generated randomly from a discrete uniform $DU(5, 10)$ distribution.

- *Production capacity utilization*: Production capacity utilization ρ is defined as the ratio of the total cumulated demand on the total cumulated available capacity. We experimented different values for ρ: 0.5, 0.7 and 0.9.

- *Demand pattern*: Binary demands for each item have been randomly generated according to the procedure described in [80].

- *Changeover costs*: For each attribute m, changeover costs C^m_{lk} have been randomly generated from a discrete uniform $DU(C^m_{min}, C^m_{max})$ distribution. We tested several possibilities: the changeover costs for all attributes can either be taken from the same interval or the changeover costs for the first attribute are greater than for the other(s). In our study, we define the ratio r as: $r = \frac{C^1_{mean}}{C^m_{mean}}$ where C^m_{mean} denotes the mean of interval $[C^m_{min}, C^m_{max}]$. We tested several values for r: 1, 2, 5, 10 and 30. In all instances, the resulting changeover costs between two items belong to the interval [0,200].

For each possible combination of multi-attribute product structure, production capacity utilization and changeover costs ratio, 5 problems were generated, resulting in $5 \times 3 \times 5 \times 5 = 375$ instances. All tests were run on a Pentium 4 (2.8 Ghz) with 505 Mb of RAM, running under Windows XP. We used a standard MIP software (CPLEX 8.1.0) with the solver default settings, using either formulation DLSP0 or formulation DLSP2.

Tables 3.11-3.15 show the computational results obtained with formulations DLSP0* and DLSP2*, for each set of instances. As the value of the ratio r appears to have an impact on the results quality, we grouped the instances with respect to the value of r so that each line corresponds to the average value for 15 randomly generated instances (5 instances for each value of production capacity utilization). For both series of results, we provide:

- *#Opt*: for set A, B and C instances, the number of instances out of the corresponding 15 instances that could be solved to optimality within 30 minutes of computation.

- *#Feas*: for set D and E instances, the number of instances out of the corresponding 15 instances for which a feasible solution could be found within 30 minutes of computation.

	Formulation DLSP0*		Formulation DLSP2*	
Variables	8520		3960	
Constraints	1921		1681	
#VI	1191		2108	
ratio r	#Opt	Gap	#Opt	Gap
r=1	13	3%	5	9%
r=2	10	6%	5	5%
r=5	2	12%	6	4%
r=10	0	21%	11	3%
r=30	1	14%	15	0%

Table 3.11: Multi-attribute product structure: results for set A instances

	Formulation DLSP0*		Formulation DLSP2*	
Variables	36350		6150	
Constraints	3851		2451	
#VI	838		1453	
ratio r	#Opt	Gap	#Opt	Gap
r=1	3	7%	0	17%
r=2	0	16%	0	16%
r=5	0	26%	4	11%
r=10	0	31%	7	4%
r=30	0	36%	11	5%

Table 3.12: Multi-attribute product structure: results for set B instances

- *Gap*: for the instances that could not be solved to optimality, the average relative gap value obtained after 30 minutes of computation between the best integer solution (if one could be found) and the best lower bound found.

We now compare the results obtained with formulations DLSP0* and DLSP2*. The results from tables 3.11-3.15 show that:

- for high values of ratio r ($r \geq 5$), i.e. when one attribute has corresponding changeover costs clearly higher than the other(s) attribute(s), the results obtained with formulation DLSP2* are better. This can be seen as:
  - a feasible solution could be obtained for all instances,
  - more instances could be solved to optimality within 30 minutes of computation,
  - when a guaranteed optimal solution could not be found within 30 minutes of computation, the residual gap is smaller.
- for small values of the ratio r ($r \leq 2$), formulation DLSP0* provides better results for medium-sized instances (sets A, B and C). However, this is not the case for the larger instances in sets D and E. Namely, for these instances,

	Formulation DLSP0*		Formulation DLSP2*	
Variables	36350		4950	
Constraints	3851		2451	
#VI	840		1653	
ratio r	#Opt	Gap	#Opt	Gap
r=1	5	11%	0	20%
r=2	3	13%	0	21%
r=5	0	24%	0	14%
r=10	0	24%	9	10%
r=30	0	39%	8	7%

Table 3.13: Multi-attribute product structure: results for set C instances

	Formulation DLSP0*		Formulation DLSP2*	
Variables	102200		12200	
Constraints	9201		5601	
#VI	2792		2865	
ratio r	#Feas	Gap	#Feas	Gap
r=1	10	38%	15	40%
r=2	7	42%	15	42%
r=5	10	48%	15	35%
r=10	10	57%	15	29%
r=30	10	62%	15	24%

Table 3.14: Multi-attribute product structure: results for set D instances

	Formulation DLSP0*		Formulation DLSP2*	
Variables	102200		10600	
Constraints	9201		6001	
#VI	2792		2911	
ratio r	#Feas	Gap	#Feas	Gap
r=1	12	29%	15	31%
r=2	10	34%	15	36%
r=5	9	47%	15	33%
r=10	10	54%	15	29%
r=30	12	59%	15	26%

Table 3.15: Multi-attribute product structure: results for set E instances

- a feasible solution could not always be found with formulation DLSP0* whereas at least one feasible solution could be found for each instance with formulation DLSP2*.
- the residual gap is significantly smaller on some instances with formulation DLSP2*.

Comparison between the results obtained with the two formulations thus shows that using formulation DLSP2*, we are able to improve the efficiency of the Branch & Bound procedure, especially for the high values of ratio r and for the largest instances. This can be explained by two main factors:

- Using formulation DLSP2*, the problem size (i.e. the number of variables and constraints) is significantly reduced. As a consequence, the time spent at each node of the Branch & Bound tree to solve the linear relaxation is shorter and more nodes can be explored within 30 minutes of computation.
- The formulation enhancement obtained thanks to the valid inequalities adapted for formulation DLSP2 gives better results when ratio r has a high value. More precisely, for high values of r, the lower bounds provided by formulation DLSP2* are higher than the ones provided by formulation DLSP0*. On the contrary, for small values of r, the lower bounds provided by the formulation DLSP0* are higher than the ones provided by formulation DLSP2*.

Thus the combined advantages of a reduced problem size and of tighter lower bounds enable formulation DLSP2* to outperform the formulation DLSP0* on many instances.

3.5 Conclusion and perspectives

We presented here two new formulations for the DLSP with sequence-dependent setup costs. These formulations are derived using two original modelling ideas:

- The first idea is based on the so-called *factorization of changeover matrices*. This leads to a significant reduction in the number of changeover variables to be introduced in the formulation. However this idea cannot be combined with a tight formulation providing good lower bounds. As a consequence, as shown by the computational results, the proposed formulation is not able to outperform an existing tight formulation found in the literature.
- The second idea is to use a possible *description of the products as combinations of a number of physical attributes*. When such a structure is present in the industrial context under study, we show how to exploit it to reduce the size of the mixed-integer linear program

to be solved while maintaining the quality of the lower bounds provided by the linear relaxation. Thanks to these combined advantages, we are able to improve the efficiency of the solution process. Computational experiments show that the proposed formulation DLSP2* performs better than the tight formulation DLSP0* we chose as a reference for comparison, especially in cases where one of the physical attributes has corresponding changeover costs higher than the other(s) attribute(s).

To conclude, several interesting subjects for future research are worth mentioning:

- To strengthen the various formulations studied in the present chapter, we used and adapted the valid inequalities proposed for the single-item single-resource DLSP by [94]. However, it might be useful to investigate the use of other existing reformulations (valid inequalities, extended formulations,...) such as those proposed in [78] for lot-sizing problems.

- It would also be interesting to investigate possible extensions of formulation DLSP2* based on the multi-attribute product structure to problems with positive changeover times or problems involving multiple resources.

- In our model based on the multi-attribute product structure, we considered each attribute with the same level of detail. However, in industrial applications such as the one presented in [72], it may be possible to rank each attribute according to the relative importance of the corresponding changeover costs and thus to establish a hierarchy between attributes. In this case, it may be reasonable to consider that the changeover costs for the less important attributes are sequence-independent and to allow these attributes to take several values within a time period. This would lead to the formulation of a hybrid big/small bucket model similar to the ones presented in [29], [31], [69] and [72].

Chapter 4

The single-resource DLSP with positive changeover times

We consider the Discrete Lot-sizing and Scheduling Problem with sequence-dependent changeover costs and times. We propose to solve this problem as a mixed-integer program using a commercial solver. This is achieved thanks to the extension of an existing tight formulation for the case without changeover times to the case with positive changeover times. The results of our computational experiments show that using the proposed tight MIP formulation, instances of medium size can be optimally solved with a reasonable computational effort.

4.1 Introduction

The models presented in chapter 3 assume that a changeover does not incur any delay in the production plan. However in many practical applications, changeover operations such as cleaning, preheating, machine adjustments, calibration, inspection, test runs, change in tooling... require a significant amount of time that must be accounted for in the model. This can be done by using positive changeover times to represent the capacity loss caused by a changeover.

In the present chapter, the Discrete Lot-sizing and Scheduling Problem (DLSP) with sequence-dependent changeover costs and times is considered. We briefly recall the basic assumptions on which the DLSP relies:

- Demand for products is deterministic and time-varying.
- The production plan is established for a finite time horizon subdivided in several discrete periods.
- At most one item can be produced per period ("small bucket" model) and the facility processes either one product at full capacity or is completely idle ("all-or-nothing assumption").
- Costs to be minimized are the inventory holding costs and the changeover costs.

Here the single level single machine variant of this problem is studied: all items to be produced are end items and share the same constrained resource. In the DLSP, it is assumed that there is a changeover between two production runs for different items, resulting in a changeover cost and/or a changeover time. Changeover costs and times can depend either on the next item only (sequence-independent case) or on the sequence of items (sequence-dependent case). Significant changeover times which consume scarce production capacity tend to further complicate the problem. We consider here the most difficult variant: the DLSP with sequence-dependent changeover costs and times (denoted DLSPSD in the sequel). Moreover, there are two ways to represent changeover times in a small bucket model: changeover times can be assumed to be equal either to an integer number of planning periods or to a fraction of a planning period. In the sequel, we assume that changeover times are equal to an integral multiple of the time bucket.

The DLSP has received much attention in the literature. However only a few papers deal with the variant studied here. [80] reformulate the DLSPSD as a Travelling Salesman Problem with Time Windows and use a dynamic programming-based algorithm to solve it. [56] show the equivalence between the DLSPSD and the Batch Sequencing Problem (BSP) and use a specific

Branch & Bound type algorithm for solving the BSP to optimality. In both papers, the mixed-integer programming formulation proposed for the problem is weak and does not provide lower bounds good enough to solve the problem using a commercial solver (see results in section 3.2). However, as pointed out by [78], there is now a good knowledge about the "right" way to formulate many simple production planning submodels as mixed integer programs and, thanks to it, many practical production planning problems can be (approximately) solved using commercial solvers. To the best of our knowledge, these results have not yet been exploited to solve the DLSPSD. In the present chapter, we attempt to close this gap by proposing a new tight formulation for this specific variant of the problem.

The purpose of this chapter is thus to introduce a strengthened formulation for the DLSP with sequence-dependent changeover costs and times. This formulation is an extension of the formulation proposed by [98] for the DLSP with sequence-dependent changeover costs and zero changeover times. Thanks to this strengthened formulation, the lower bounds provided by the linear relaxation of the problem are significantly better, enabling a Branch & Bound type procedure to solve the problem more efficiently.

The chapter is organized as follows. In section 4.2, we first recall the formulation proposed by [80] and [56] for the DLSP with sequence-dependent changeover costs and times. In section 4.3, we present the proposed tight formulation for the DLSPSD. Some computational results obtained with this formulation are given in section 4.4 and section 4.5 provides the concluding remarks.

4.2 A first formulation for the DLSP with sequence-dependent changeover times

In this section, we first recall the formulation proposed by [80] for the DLSPSD.

We wish to optimize the production schedule for a set of N items over an horizon featuring T planning periods. A period is indexed by $t = 1, ..., T$, an item by $i = 0, .., N$. We agree to use item $i = 0$ to represent idle periods.

We use the following notation:

- d_{it}: demand (in units) for item i in period t.
- P_{it}: production capacity (in units per period) for item i in period t.
- h_i: holding costs per unit and period for item i.
- c_{ij}: changeover costs from item i to item j.

- T_{ij}: changeover time from item i to item j. T_{ij} is assumed to be an integer number of planning periods.

Decision variables are defined as follows:

- I_{it}: inventory level corresponding to item i at the end of period t.
- y_{it}: setup variables. y_{it} equals 1 if the resource is setup for item i in period t, and 0 otherwise.
- $\forall(i,j)$ st $i \neq j, w_{ijt}$: changeover cost variables. If $T_{ij} > 0$, w_{ijt} equals 1 during the first period of a changeover from item i to item j, and 0 otherwise. If $T_{ij} = 0$, w_{ijt} equals 1 in the first period of production of j, and 0 otherwise.
- $\forall(i,j)$ st $T_{ij} \neq 0, v_{ijt}$: changeover time variables. v_{ijt} equals 1 during each period of a changeover from item i to item j, and 0 otherwise.

(DLSPSD1)

$$min \sum_{i=1}^{N} \sum_{t=1}^{T} h_i I_{it} + \sum_{i=0}^{N} \sum_{j=0, j\neq i}^{N} \sum_{t=1}^{T} c_{ij} w_{ijt} \tag{4.1}$$

$$\forall i, \forall t, I_{it} = I_{i,t-1} + P_{it} y_{it} - d_{it} \tag{4.2}$$

$$\forall(i,j) \text{ st } T_{ij} > 0, \forall t = 1 \ldots T, \forall \tau = t - T_{ij} \ldots t-1, \text{ if } \tau \geq 0, y_{jt} + y_{i\tau} \leq 1 \tag{4.3}$$

$$\forall(i,j) \text{ st } T_{ij} > 0, \forall t = T_{ij} \ldots T, \forall \tau = t - T_{ij} \ldots t-1, v_{ij\tau} \geq y_{i,t-T_{ij}-1} + y_{jt} - 1 \tag{4.4}$$

$$\forall(i,j) \text{ st } T_{ij} > 0, \forall t, w_{ijt} \geq v_{ijt} - v_{ij,t-1} \tag{4.5}$$

$$\forall(i,j) \text{ st } i \neq j \text{ and } T_{ij} = 0, \forall t, w_{ijt} \geq y_{i,t-1} + y_{jt} - 1 \tag{4.6}$$

$$\forall(i,j) \text{ st } i \neq j, \forall t, w_{ijt} \leq y_{i,t-1} \tag{4.7}$$

$$\forall(i,j) \text{ st } i \neq j, \forall t, w_{ijt} \leq y_{j,t+T_{ij}} \tag{4.8}$$

$$\forall t, \sum_{i=0}^{N} y_{it} + \sum_{(i,j) \text{ st } T_{ij}>0} v_{ijt} = 1 \tag{4.9}$$

$$\forall i, \forall t, I_{it} \geq 0 \tag{4.10}$$

$$\forall i, \forall t, y_{it} \in \{0,1\} \tag{4.11}$$

$$\forall (i,j) \text{ st } j \neq i, \forall t, w_{ijt} \in \{0,1\} \tag{4.12}$$

$$\forall (i,j) \text{ st } T_{ij} > 0, \forall t, v_{ijt} \in \{0,1\} \tag{4.13}$$

The objective, to minimize the sum of inventory holding costs and changeover costs, is expressed by (4.1). Changeover costs c_{ij} are incurred between two successive production runs of item i and item j, in the first period of production of item j if $T_{ij} = 0$ or in the first period of the transition from i to j if $T_{ij} > 0$.

Constraints (4.2) express the inventory balance. The "all-or-nothing" assumption is enforced by the term $P_{it}y_{it}$ in the equality: if the resource is setup for i in period t, then all the available capacity is used and the production quantity of item i must be equal to P_{it}. Together with constraints (4.10), they also ensure that demand for each item is fulfilled without backlogging.

Constraints (4.3) ensure that if item j is produced in period t, no other item i with changeover times $T_{ij} > 0$ can be produced in periods $[t - T_{ij}, t - 1]$, since these periods need to be reserved either for item j or for a transition from item i.

For pair of items (i,j) such that $T_{ij} > 0$, constraints (4.4) force that if production takes place for item i in period $t - T_{ij} - 1$ and for item j in period t, then periods $[t - T_{ij}, t - 1]$ are reserved for the transition from i to j.

For the case of positive changeover time ($T_{ij} > 0$), constraints (4.5) force $w_{ijt} = 1$ if period t is the first period of a transition from item i to item j. Similarly for the case of zero changeover time ($T_{ij} = 0$), constraints (4.6) force $w_{ijt} = 1$ if period t is the first period of production of item j after a changeover from item i.

If $w_{ijt} = 1$, constraints (4.7) ensure that item i is produced in period $t-1$, whereas constraints (4.8) ensure that item j is produced in period $t + T_{ij}$.

(4.9) ensure that in each period, the resource either produces a single item at full capacity, or is idle (i.e $y_{0t} = 1$), or is in transition between two items.

The binary character of the setup and changeover variables is represented by constraints (4.11)-(4.13).

4.3 A tight formulation for the DLSPSD

We now present a tight formulation for the DLSP with sequence-dependent changeover costs and times. This formulation is an extension of the formulation proposed by [5] and [98] for the DLSP with sequence-dependent changeover costs and zero changeover times.

4.3.1 Initial formulation

We use the same notation as in section 4.2 for the problem parameters.

Decision variables are defined as follows:

- I_{it}: inventory level corresponding to item i at the end of period t.
- y_{it}: setup variables. y_{it} equals 1 if the resource is setup for item i in period t, and 0 otherwise.
- w_{ijt}: changeover cost variables. If $T_{ij} > 0$, w_{ijt} equals 1 during the first period of a transition from item i to item j, and 0 otherwise. If $T_{ij} = 0$, w_{ijt} equals 1 in the first period of production of j, and 0 otherwise.
- v_t: changeover time variables. v_t equals 1 during each period of a changeover between two items, and 0 otherwise.

With this notation, we propose to formulate the DLSPSD as follows:
(DLSPSD2)

$$min \sum_{i=1}^{N} \sum_{t=1}^{T} h_i I_{it} + \sum_{i=0}^{N} \sum_{j=0}^{N} \sum_{t=1}^{T} c_{ij} w_{ijt} \tag{4.14}$$

$$\forall i, \forall t, I_{it} = I_{i,t-1} + P_{it} y_{it} - d_{it} \tag{4.15}$$

$$\forall i, \forall t, y_{i,t-1} = \sum_{j=0}^{N} w_{ijt} \tag{4.16}$$

$$\forall j, \forall t, y_{jt} = \sum_{i=0 \ldots N \text{ st } t-T_{ij}>0} w_{ij,t-T_{ij}} \tag{4.17}$$

$$\forall t, \sum_{i=0}^{N} y_{it} + v_t = 1 \tag{4.18}$$

$$\forall i, \forall t, I_{it} \geq 0 \tag{4.19}$$

$$\forall i, \forall t, y_{it} \in \{0, 1\} \tag{4.20}$$

$$\forall i, \forall j, \forall t, w_{ijt} \in [0, 1] \tag{4.21}$$

$$\forall t, v_t \in [0, 1] \tag{4.22}$$

The objective, minimizing the sum of inventory holding costs and changeover costs, is expressed by (4.14). Note that, in the formulation DLSPSD2, variables w_{iit} are introduced: $w_{iit} = 1$ means that the resource is setup for item i both in period $t-1$ and in period t, i.e. that a production run for item i takes place over periods $t-1$ and t.

Constraints (4.15) express the inventory balance. Together with constraints (4.19), they ensure that demand for each item is fulfilled without backlogging.

Equalities (4.16) and (4.17) link the setup variables with the changeover cost variables. (4.16) guarantee that item i can be produced in period $t-1$ if and only if a changeover from i to another item j (possibly $j = i$) takes place at the beginning of period t. Similarly, (4.17) guarantee that item j can be produced in period t if and only if a changeover from another item i (possibly $i = j$) to item j begins early enough (i.e. in period $t - T_{ij}$) to be finished at the beginning of period t.

(4.18) ensure that in each period, the resource either produces a single product at full capacity, or is idle (i.e $y_{0t} = 1$), or is in transition between two items (i.e. $v_t = 1$).

The binary character of the setup variables is represented by (4.20). (4.21) and (4.22) state the non-negativity of the changeover variables: observe, as pointed out by [5], that thanks to constraints (4.16)-(4.18) and (4.20), there is no need to define variables w_{ijt} and v_t as binary variables.

We note that thanks to this reformulation, there is no need to introduce explicit changeover time variables v_{ijt} in the formulation to ensure that positive changeover times between production runs for different items are respected. Thus the entire set of inequalities (4.3)-(4.8) of formulation DLSPSD1 is replaced by the (much smaller) set of equalities (4.16)-(4.17).

4.3.2 Strengthening the formulation with valid inequalities

As shown in [98] for the case without changeover times, the formulation DLSPSD2 can be further strengthened through a family of valid inequalities adapted from the ones developed by [94]. We investigate here an extension of this idea to the case of positive changeover times and propose a family of valid inequalities for the problem (4.14)-(4.22).

This can be done using the assumption of Wagner-Whitin costs, constant capacity and no backlogging. In this case, demands and production capacity can be normalized without loss of generality: $d_{it} \in \{0, 1\}$ and $P_{it} = 1$. We first introduce some additional notation:
- $D_{i,t,\tau}$: cumulated demand for item i in the interval $\{t, ..., \tau\}$. Demand on item i is binary so that $D_{i,t,\tau}$ is equal to the number of positive demand periods for i in $\{t, ..., \tau\}$.
- $S_{i,q}$: q^{th} positive demand period for item i. Note that $S_{i,D_{i,1,t}+q}$ denotes the q^{th} period with positive demand for item i after period t.

We also introduce the start-up variables z_{it} defined as follows: z_{it} equals 1 if the production of a new lot of item i starts at the beginning of period t, 0 otherwise. These start-up variables are linked to the changeover variables by the equations:

$$\forall j, \forall t, z_{jt} = \sum_{i:i \neq j} w_{ij,t-T_{ij}} \tag{4.23}$$

Equalities (4.23) state that the production of a new lot of item j begins in period t if and only if a changeover from another item $i \neq j$ starts "early enough" (i.e. in period $t - T_{ij}$) to be finished at the beginning of period t.

With this notation, we have:

Proposition 4.1 *The following inequations (4.24) are valid inequalities for the DLSP with sequence-dependent changeover costs and times:*

$$\forall t, \forall i, \forall p \in \{0...D_{i,t+1,T}\}, I_{it} \geq \sum_{q=1}^{p} \left(1 - y_{i,t+q} - \sum_{\tau=t+q+1}^{S_{i,D_{i,1,t}+q}} z_{i\tau}\right) \tag{4.24}$$

Proof 4.1 *A sketch of proof is as follows. First note that* $y_{i,t+q} + \sum_{\tau=t+q+1}^{S_{i,D_{i,1,t}+q}} z_{i\tau} = 0$ *if and only if the resource is not setup for item* i *in period* $t+q$ *and no startup for* i *takes place between the period* $t+q+1$ *and the period where the* q^{th} *demand after period* t *occurs, i.e. if and only if no production of item* i *is possible in the interval* $\{t+q, ..., S_{i,D_{i,1,t}+q}\}$. *In this case, the quantity needed to satisfy the* q^{th} *demand after period* t *should be in stock at the end of period* t. *Thus we see that constraints (4.24) force an increase of the stock of item* i *at the end of period* t *by one for each index* q *for which no production occurs in the interval* $\{t+q, ..., S_{i,D_{i,1,t}+q}\}$. *A detailed proof of the validity of (4.24) can easily be derived from the above (see also [94]).* □

In the computational experiments to be presented in section 4.4, the following cutting-plane generation strategy has been implemented to strengthen the DSLPSD2 formulation by adding violated valid inequalities (4.24):

1. We solve the linear relaxation of the problem using the formulation DLSPSD2.

2. We check whether each valid inequality of type (4.24) is satisfied. If it is violated by the current continuous solution, we add it to the formulation.

3. If at least one violated inequality is found in step 2, we go back to step 1 and repeat until no more violated valid inequalities can be generated.

The resulting strengthened formulation is denoted DLSPSD2*.

item i	0	1	2	3	4
inventory holding costs h_i	0	7	5	6	7

Table 4.1: Simple example with positive changeover times: inventory holding costs

Changeover costs

	0	1	2	3	4
0	0	100	120	180	105
1	110	0	176	115	198
2	103	164	0	128	140
3	156	135	122	0	137
4	196	188	142	154	0

Changeover times

	0	1	2	3	4
0	0	0	2	0	2
1	2	0	1	0	1
2	0	0	0	1	0
3	2	1	0	0	1
4	0	0	0	0	0

Table 4.2: Simple example with positive changeover times: changeover costs and times between items

4.3.3 A small illustrative example

We use a small example to show an application of the formulation DLSPSD2 and to illustrate the interpretation of equalities (4.16)-(4.17) as flow conservation constraints in a network.

We consider a problem involving $N = 4$ items and $T = 15$ periods. We agree to use item $i = 0$ to denote idle periods. Table 4.1 gives the inventory holding costs for each item and table 4.2 provides the changeover costs and times between pairs of items. Demand over the planning horizon for each item is provided in table 4.3.

The upper part of figure 4.1 shows the optimal production plan obtained while using the formulation DLSPSD2 , the cost of which is $Z^* = 903$. The symbol "Tr" denotes a period where the resource is in transition between two production runs.

Before going on with the computational results, we briefly illustrate on this small example the interpretation of equalities (4.16)-(4.17) as flow conservation constraints. Namely, as pointed out by [5] for the variant with zero changeover times, the definition of a production plan can be seen as a problem of defining a single unit flow in a network under additional constraints.

More precisely, we consider a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. A node $v \in \mathcal{V}$ corresponds to a item-period pair (i, t). There is an oriented arc $a \in \mathcal{E}$ from node v_1 to node v_2 if and only if $v_1 = (i, t)$ et $v_2 = (j, t + T_{ij} + 1)$. The setup variable y_{it} corresponds to the flow through node (i, t) and the

period	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
item 1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0
item 2	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
item 3	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1
item 4	0	0	0	0	1	0	0	0	0	0	0	1	0	0	1

Table 4.3: Simple example with positive changeover times: demand on items

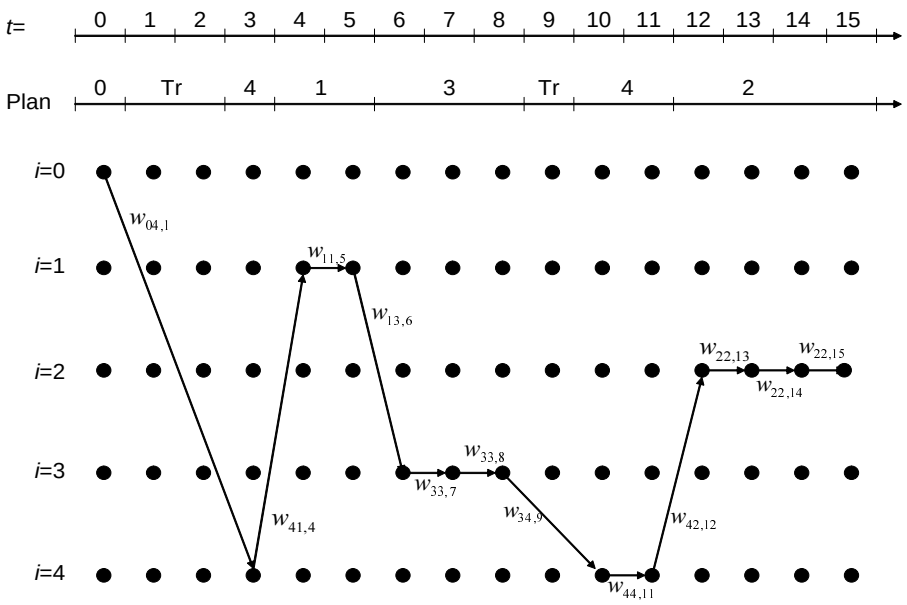

Figure 4.1: Positive changeover times: optimal production plan for the small example

changeover variables $w_{ij,t+1}$ corresponds to the flow between node (i,t) and node $(j,t+T_{ij}+1)$. With this interpretation, a production sequence on the resource corresponds to a flow of a single unit through the network, starting from a node $(i,0)$ (initial setup state of the resource) and arriving in a node (j,T) (final setup state of the resource). Thus equalities (4.16) can be seen as flow conservation constraints, stating that the flow through node $(i,t-1)$ is equal to the sum of the flows on the arcs directed away from this node. Similarly, equalities (4.17) can be seen as flow conservation constraints, stating that the flow through node (j,t) is equal to the sum of the flows on the arcs directed toward this node.

Due to the presence of positive changeover times, the structure of graph $\mathcal{G}$ used here is seen to be different from the one used in [5]. Namely, when all changeover times are equal to zero, the arcs in graph $\mathcal{G}$ link pairs of nodes related to successive planning periods, which is not the case anymore with positive changeover times.

The lower part of figure 4.1 shows the interpretation of the optimal production plan for the illustrative example as a single unit flow in the corresponding network. For the sake of simplicity, only the arcs with a positive flow are shown.

4.4 Computational results

In this section, we discuss the results of some computational experiments carried out to evaluate the formulation DLSPSD2* proposed in section 4.3.

4.4.1 Problem instances generation

We created several sets of randomly generated instances following the procedure described in [80] and [56]. The reader is referred to these references for more details. The generated instances differ with respect to the following characteristics:

- *Problem dimension*: The problem dimension is represented by the number of products N and the number of periods T. We use 7 different item-period combinations, namely (N,T) = {(5,20), (10,40), (5,60), (10,60), (15,60), (10,90), (15,90)}, leading to 7 instance sets denoted sets A to G.

- *Production capacity utilization*: Production capacity utilization ρ is defined as the ratio between the total cumulated demand and the total cumulated available capacity. Because changeover times are nonzero, we experimented different medium values for ρ: ρ was varied between 0.5 and 0.75, in steps of 0.05.

For each possible combination of problem dimension and production capacity utilization, 5 problems were generated, resulting in $7 \times 6 \times 5 = 210$ instances. All tests were run on a Pentium 4 (2.8 Ghz) with 505 Mb of RAM, running under Windows XP. We used a standard MIP software (CPLEX 8.1.0) with the solver default settings to solve the problem, using either formulation DLSPSD1 presented in section 4.2 or formulation DLSPSD2* presented in section 4.3.

4.4.2 Comparison of formulations DSLPSD1, DLSPSD2 and DLSPSD2*

We first carried out some computational experiments in order to evaluate the reformulation proposed in section 2.1 and the family of valid inequalities derived in section 2.2. The comparison has been limited to the smallest instances (sets A and B) since computation time limits are exceeded for set B instances with the DLSPSD1 formulation.

Table 4.4 shows the results obtained with the DSLPSD1, DLSPSD2 and DLSPSD2* formulations for the sets A and B instances. For each formulation, we provide:
- *Variables* and *Constraints*: the average number of variables and constraints.
- $\#VI$: for the DLSPSD2* formulation, the average number of valid inequalities of type (4.24) added by the cutting-plane generation procedure.
- $\#Opt$: the number of instances out of the corresponding 30 instances that could be solved to optimality within 20 minutes of computation.
- Gap_0: the integrality gap, i.e. the relative difference between the lower bound provided by the linear relaxation of the problem and the value of an optimal solution. For the DLSPSD2*

	set A			set B		
MIP formulation	DLSPSD1	DLSPSD2	DLSPSD2*	DLSPSD1	DLSPSD2	DLSPSD2*
Variables	1167	960	960	7880	5720	5720
Constraints	2392	361	361	19946	1321	1321
#VI	_	_	124	_	_	479
#Opt	27	30	30	0	0	30
Gap_0(%)	84	59	4	92	68	5
	[65;95]	[47;69]	[0;12]	[80;96]	[57;73]	[0;13]
#Nodes	49362	738	3	2497	35497	100
	[16;191523]	[11;2780]	[0;19]	[391;4969]	[24705;52968]	[2;319]
CPU_{IP}(s)	343	3	1	1200	1200	30
	[3;1200]	[0;8]	[0;3]	[1200;1200]	[1200;1200]	[3;100]
Gap (%)	2	0	0	81	33	0
	[0;29]	[0;0]	[0;0]	[60;91]	[11;44]	[0;0]

Table 4.4: Results for set A and B instances

formulation, we consider the lower bound obtained after the cutting-plane generation procedure has stopped.

- *#Nodes*: the number of nodes of the search tree explored before a guaranteed optimal solution is found or the computation time limit of 20 minutes is reached.

- CPU_{IP}: the computation time in seconds required to find a guaranteed optimal solution. If one could not be found, we use the computation time limit of 1200 seconds.

- *Gap*: the gap obtained after 20 minutes of computation between the best integer solution and the best lower bound found.

For performance measures Gap_0, *Nodes*, CPU_{IP} and *Gap*, we provide the average value (on the first line) and the minimum and maximum values (in brackets on the second line) for the considered set of randomly generated instances.

Table 4.4 shows that the results obtained with the DLSPSD2* formulation are much better than the ones obtained with the DLSPSD1 formulation. Namely, computation times are significantly reduced and more instances can be solved to optimality within the time limit while using the DLSPSD2* formulation.

This can be explained by the combination of two advantages:

1. The lower bounds provided by the linear relaxation of the DLSPSD2* formulation are much better than the ones obtained with the DLSPSD1 formulation. This formulation improvement is achieved to a large extent thanks to the use of a small number of valid inequalities (4.24). This can be seen e.g. for set A instances for which the integrality gap is reduced in average from 84% with the DLSPSD1 formulation to 59% with the basic

DLSPSD2 formulation and 4% with the strengthened DLSPSD2* formulation.

2. The MIP size (number of variables and constraints) is significantly reduced with the DLSPSD2* formulation. As a consequence, the time spent at each node of the branch and bound tree to solve the linear relaxation is shorter. This size reduction is explained by the fact that using the DLSPSD2* formulation, there is no need to introduce explicit changeover time variables v_{ijt} for each possible transition between pairs of items (i, j). Moreover, the set of equalities (4.16)-(4.17) are sufficient to ensure that positive changeover times between production runs for different items are respected. As a consequence, the numerous inequalities needed in the DLSPSD1 formulation to link changeover time variables to setup and changeover cost variables can be eliminated from the formulation.

Thus, thanks to tighter lower bounds and a reduced MIP size, the efficiency of the branch and bound procedure embedded in CPLEX solver is significantly improved while using the DLSPSD2* formulation.

4.4.3 Results with the MIP formulation DLSPSD2*

In order to further validate our approach, we carried out additional computational experiments. More precisely, we considered instances similar to the ones studied in [56] and [80], i.e. instances for which $(N, T) = \{(5, 60), (10, 60)\}$ (sets C-D). We also used 3 additional sets of larger instances for which $(N, T) = \{(15, 60), (10, 90), (15, 90)\}$ (sets E-G).

Table 4.5 displays the detailed results obtained with the DLSPSD2* formulation. We observe that:
- For medium size instances (sets C-D), 98% of the generated instances could be solved to optimality within 20 minutes of computation.
- For large size instances (sets E-G), 42% of the generated instances could be solved to optimality within the computation time limits. Moreover the average remaining gap obtained after 20 minutes of computation between the best integer solution and the best lower bound found is small (3.6% on average).

Thus, even if the proposed approach was implemented on a computer with more computing power, these results suggest the potential of the MIP modelling approach to solve instances larger than the ones considered in [56] and [80].

	set C	set D	set E	set F	set G
Variables	2880	8580	17280	12870	25830
Constraints	1080	1981	2881	2971	4321
#VI	1084	1131	1093	2534	2517
#Opt	30	29	27	8	3
Gap_0(%)	5	5	4	5	6
	[1;13]	[1;11]	[2;6]	[3;10]	[2;17]
#Nodes	150	382	440	604	299
	[0;832]	[74;1385]	[13;1217]	[48;1085]	[30;707]
CPU_{IP} (s)	46	248	440	1089	1150
	[9;168]	[60;1200]	[53;1200]	[82;1200]	[144;1200]
Gap(%)	0	0.01	2	3	5
	[0;0]	[0;0.13]	[0;2]	[0;7]	[0;15]

Table 4.5: Results for set C-G instances obtained with the DLSPSD2* formulation

4.5 Conclusion

We presented a new tight formulation for the DLSP with sequence-dependent changeover costs and times. Our proposal is based on the extension of a tight MIP formulation available for the case without changeover times to take into account positive changeover times. The obtained formulation is then further strengthened thanks to the use of a family of valid inequalities.

The computational experiments carried out show that using the proposed formulation, we are able to significantly improve the efficiency of the Branch & Bound procedure imbedded in CPLEX solver thanks to tighter lower bounds and a reduced MIP size and to solve medium-sized instances with a reasonable computational effort.

To conclude, several interesting directions for future research are worth mentioning:

- We assumed here that there is a single capacitated resource in the production planning problem to be solved. However many industrial applications involve several resources. Hence, it might be useful to investigate possible extensions of formulation DLSPSD2* to problems involving multiple parallel resources.

- In chapter 3, a new formulation (denoted DLSP2*) for the DLSP with sequence-dependent changeover costs exploiting a possible description of the products as combinations of a number of physical attributes is presented. To derive this formulation, we assumed zero changeover times. It could be interesting to investigate its extension to the case of positive changeover times.

Chapter 5

The multi-resource DLSP: MIP formulation and heuristic solution approach

We present an initial MIP formulation for the DLSP with sequence-dependent changeover costs and multiple (parallel) resources. The presence of parallel resources makes this extension of the DLSP particularly difficult to solve as can be seen in our first computational results obtained while using a commercial solver. This is why we propose a heuristic procedure aiming at providing good approximate solutions for this problem. We develop a solution approach based on the representation of a solution by the demand assigned to each resource. Although the computational behavior of the proposed algorithm is not really satisfactory, this preliminary study enables us to identify several interesting directions for furture research.

5.1 Introduction

The models presented in the previous chapters assume that there is a single resource available to produce all items. However in many cases, a manufacturer has access to multiple machines or production lines, which can be used in parallel. Single-resource models may still be useful in these situations. Namely, when the set of items to be produced can be partitioned into disjoint subsets, each of which being assigned to one of the available resources, the multi-resource production planning problem is seen to decompose into a series of single-resource problems.

However, in most industrial applications involving multiple production resources, the machines or production lines have some flexibility. A given item can thus be produced on several resources and the problem cannot be decomposed into a series of single-resource problems. As a result, we have to plan production for all the resources simultaneously. The presence of parallel resources complicates the problem mainly because there is an additional decision to be made: we have to determine not only the timing and level of production, but also the assignment of production lots to machines.

In the present chapter, we consider the extension of the Discrete Lot-sizing and Scheduling Problem with sequence-dependent changeover costs to the case where there are several identical parallel resources. This problem is based on the following key assumptions:

- There are several identical parallel resources: the production capacity and the changeover cost matrices are the same for each resource.
- All items to be produced are end items.
- Demand for items is deterministically known and time-varying.
- The production plan is established for a finite time horizon subdivided into several discrete periods.
- For each resource, at most one type of product can be produced per period ("small bucket" model) and the facility processes either one type of product at full capacity or is completely idle ("all-or-nothing" assumption).
- Costs to be minimized are the inventory holding costs and the sequence-dependent changeover costs.
- Changeover delays between two production lots are assumed to be zero.

The DLSP has received much attention in the literature. However only a few papers deal with the variant studied here. The authors of [23] consider production planning for the curing

stage in a tile manufacturing facility and formulate their problem as a DLSP with heterogenous parallel resources (the curing kilns). Their model involves sequence-dependent changeover costs and times. Their solution procedure uses a Lagrangian relaxation of the inventory balance constraints to decompose the problem into a series of single-resource independent subproblems. They were able to provide approximate solutions for instances involving at most 5 items, 5 partially specialized processors and 52 time periods. More recently, [82] heuristically solve a multi-resource DLSP with sequence-dependent changeover costs arising in a company producing acrylic fibres. They use a special-purposed algorithm adapted from a heuristic found in the literature. The related CSLP with multiple heterogenous resources was studied by [21] for the planning of an injection molding plant in the health care industry. In their problem, changeovers are sequence-dependent and incur both changeover costs and delays. The authors develop a two-phased resource-based heuristic aiming at decomposing the initial large problem into smaller subproblems. In their numerical experiments, they consider instances involving a maximum of 51 items, 45 resources and 30 periods.

Moreover, the literature on exact solution approaches to solve single-level multi-resource lot-sizing problems is rather sparse. A noticeable exception can be found in [78] (chapter 14) where the use of MIP modeling and reformulation approach to solve real life production planning problems is presented. Among others, the authors of [78] study an industrial case arising in a plant producing insulating boards by extrusion. Their problem shares some common features with the problem discussed here. Indeed, it involves several heterogenous parallel resources with varying capacities and sequence-dependent changeover times. A big bucket model similar to the ones studied in [58] and [71] is used: the planning horizon is divided into long planning periods and lot-sizing and scheduling decisions are made simultaneously to decide about the exact production sequence within each planning period. In [78], a unit flow formulation using equalities similar to constraints (3.4)-(3.5) of the DLSP0 formulation to link setup and changeover variables is proposed. It is further strengthened thanks to several families of valid inequalities for single-resource single-item subproblems. In the sequel, we use a similar approach but we study a small bucket model and assume identical resources with a constant capacity.

The purpose of the present chapter is thus to propose a first MIP formulation for the DLSP with identical parallel resources and sequence-dependent changeover costs and to present a simple heuristic solution procedure aiming at providing good solutions for medium to large-sized instances.

This chapter is organized as follows. In section 5.2, we first introduce an initial MIP formulation for the DLSP with identical parallel resources and sequence-dependent changeovers. In

section 5.3, we present a small industrial example we found in a French company. Results of computational experiments obtained with the proposed MIP formulation are then reported in section 5.4. However, even for the small instances, this initial formulation does not seem strong enough to provide good solutions. This is why we investigate a heuristic solution procedure based on the representation of a solution by the demand assigned to each resource. An early version of the proposed algorithm is presented in section 5.5 whereas some preliminary results are discussed in section 5.6. Section 5.7 provides the concluding remarks.

5.2 Initial formulation

In this section, we derive a MIP formulation for the DLSP with parallel resources and sequence-dependent changeover costs. This formulation is an extension of the tight formulation proposed in [5] and [98] for the DLSP with a single resource and sequence-dependent changeover costs.

We wish to optimize the production schedule for a set of N items to be produced on R parallel resources over an horizon featuring T planning periods. A period is indexed by $t = 1, ..., T$, an item by $i = 0, .., N$ and a resource by $r = 1, ..., R$. We agree to use item $i = 0$ to represent idle periods on the various resources.

We use the following notation for the parameters:

- d_{it}: demand (in units) for item i in period t,
- P^r_{it}: production capacity of resource r (in units per period) for item i in period t,
- h_i: inventory holding costs per unit and period for item i,
- c^r_{ij}: cost of a changeover from item i to item j on resource r.

Note that, when the available resources are assumed to be identical, the values of the parameters P^r_{it} and c^r_{ij} are the same for every resource $r = 1, ..., R$.

Decision variables are defined as follows:

- I_{it}: inventory level corresponding to item i at the end of period t.
- y^r_{it}: setup variables. $y^r_{it} = 1$ if the resource r is setup for item i in period t, and 0 otherwise. Note that $y^r_{0t} = 1$ corresponds to the case where resource r is idle on time period t.
- w^r_{ijt}: changeover variables. $w^r_{ijt} = 1$ if the resource r is switched from item i to item j at the beginning of period t, and 0 otherwise.

With this notation, we propose to formulate the DLSP with parallel resources and sequence-dependent changeover costs as follows:

(DLSPPR)

$$min \sum_{i=1}^{N} \sum_{t=1}^{T} h_i I_{it} + \sum_{r=1}^{R} \sum_{i=0}^{N} \sum_{j=0}^{N} \sum_{t=1}^{T} c_{ij}^r w_{ijt}^r \tag{5.1}$$

$$\forall i, \forall t, I_{it} = I_{i,t-1} + \sum_{r=1}^{R} P_{it}^r y_{it}^r - d_{it} \tag{5.2}$$

$$\forall r, \sum_{i=0}^{N} y_{i0}^r = 1 \tag{5.3}$$

$$\forall r, \forall i, \forall t, y_{i,t-1}^r = \sum_{j=0}^{N} w_{ijt}^r \tag{5.4}$$

$$\forall r, \forall j, \forall t, y_{jt}^r = \sum_{i=0}^{N} w_{ijt}^r \tag{5.5}$$

$$\forall r, \forall i, \forall j, \forall t, w_{ijt}^r \geq 0 \tag{5.6}$$

$$\forall r, \forall i, \forall t, I_{it} \geq 0 \tag{5.7}$$

$$\forall r, \forall i, \forall t, y_{it}^r \in \{0, 1\} \tag{5.8}$$

The objective, minimizing the sum of inventory holding costs and changeover costs, is expressed by (5.1). Changeover costs c_{ij}^r are incurred between two successive production batches of item i and item j on resource r, in the first period of production of item j.

Constraints (5.2) express the global inventory balance and, together with (5.3), guarantee that the "all-or-nothing" assumption is fulfilled. Note that the total production quantity of item i in period t is equal to the sum of the quantities produced on the available resources: $\sum_{r=1}^{R} P_{it}^r y_{it}^r$.

(5.3), together with constraints (5.4)-(5.5), ensure that in each period, each resource r either produces a single product at full capacity, or is idle (i.e $y_{0t}^r = 1$).

For each resource r, equalities (5.4) and (5.5) link the setup variables with the changeover variables. (5.4) guarantee that item i can be produced in period $t-1$ on resource r if and only if a changeover from i to another item j (possibly $i = j$) takes place on this resource at the beginning of period t. Similarly, (5.5) guarantee that item j can be produced in period t on resource r if and only if a changeover from another item i (possibly $i = j$) to item j takes place on this resource at the beginning of period t.

(5.6) state the non-negativity of the changeover variables: observe, as pointed out by [5], that thanks to constraints (5.3)-(5.5) and (5.8), there is no need to define variables w_{ijt}^r as binary variables. The set of constraints (5.2) and (5.7) ensure that demand for each item is fulfilled

without backlogging. The binary character of the setup variables is enforced by (5.8).

When there is a single resource with Wagner-Whitin costs and a constant capacity, the initial formulation for the DLSP can be further strengthened thanks to the strong valid inequalities developed by [94]. This approach was proposed in [98] for the basic model and we investigated in chapters 3 and 4 extensions of these valid inequalities to various single-resource variants of the DLSP. However, these valid inequalities are based on the key assumptions that demand can be normalized without loss of generality (i.e. $d_{it} \in \{0, 1\}$) and that at most one unit of demand can be produced in each time period.

In the present case, all resources have the same constant capacity throughout the planning horizon so that we can still define a unit of item i as the maximum quantity of this item that can be produced per period on one resource. Thus production capacity can be normalized ($P_{it}^r = 1$) and demands can be expressed without loss of generality as integer multiples of the production capacity on a resource (i.e. $\forall i, \forall t, d_{it} \in \{0, 1, ..., R\}$).

But the assumption that at most one unit of demand can be produced in each time period does not hold anymore as several units of a given item can be produced simultaneously on various resources. This hinders the direct extension of the valid inequalities developed by [94] to the general multi-resource problem. As a consequence, we did not use them in our first computational results to be presented in section 5.4.

5.3 A small industrial example

Before going on with a discussion on our computational results, we present a small industrial example in order to show the practical relevance of the proposed model. This example was found in a French company producing energetic components for gas generators. These gas generators are used in automotive safety applications such as airbag deployment and safety belt tensioning. More details about this industrial case can be found in [11]. In the sequel, numerical data relative to costs have been modified for confidentiality reasons.

This example deals with the production of $N = 12$ semi-finite products which can be described as propellant tubes. As the further transformation of these intermediate products into end products does not involve capacity-constrained resources, it is not considered here. The plan is established for 12 weeks, each week being divided into 2 periods, so that the planning horizon is made of $T = 24$ periods. There are $R = 2$ extruding machines which can be considered as identical parallel resources.

The set of products is partitioned into 2 families, each family being defined by a common

Product (family 1)	1	2	3	4	5	6
Inventory holding costs	11148	8730	8830	8973	7388	14301

Product (family 2)	7	8	9	10	11	12
Inventory holding costs	6752	7451	5581	4239	6885	7063

Table 5.1: Simple example with 2 parallel resources: inventory holding costs

chemical composition. Products $i = 1..6$ belong to family 1, products $i = 7...12$ to family 2. Products of the same family differ with respect to their shape and dimensions. The cost of a (minor) changeover between two products belonging to the same family is evaluated at 2360 euros. As the cleaning operations required to change the chemical composition are dangerous and time consuming, the cost of a (major) changeover between products belonging to different families is evaluated at 1.5 times the cost of a minor changeover (i.e. at 3540 euros). Demand on family 1 products represents around 60% of the total demand, demand on family 2 products around 40%.

The production capacity per period is constant throughout the planning horizon so that we can define a unit of product as the quantity of this product which can be produced on one resource during a planning period. Thus demand can be normalized: $d_{it} \in \{0, 1, 2\}$. Table 5.1 provides the inventory holding costs expressed in euros per unit per period for each product.

The model and MIP formulation presented in section 5.2 can be used to plan production in this industrial application. We created a set of instances (set I) based on this example and used it in our computational experiments to be presented in subsection 5.4. Moreover, current practice in the company is to prohibit production of family 2 products on resource 1 in order to avoid major changeovers. We also carried out some limited computational experiments (see subsection 6.4) to evaluate the additional cost resulting from this policy.

5.4 Computational results obtained with the initial formulation

In this subsection, we discuss the results of some computational experiments carried out to evaluate the formulation presented in section 5.2. We used 4 sets of randomly generated instances. The instances differ with respect to the following characteristics:

- *Problem dimension*: The problem dimension is represented by the number of products N, the number of periods T and the number of resources R. We use 4 different combinations, leading to 4 sets of problems:
  - set A: $N = 5$, $T = 30$ and $R = 2$;

  - set B: $N = 10$, $T = 40$ and $R = 2$;
  - set C: $N = 5$, $T = 30$ and $R = 3$;
  - set D: $N = 10$, $T = 40$ and $R = 3$.

- *Inventory holding costs*: For each item, inventory holding costs have been generated randomly from a discrete uniform $DU(5, 10)$ distribution.

- *Production capacity utilization*: Production capacity utilization ρ is defined as the ratio between the total cumulated demand ($\sum_{i=1}^{N} \sum_{t=1}^{T} d_{it}$) and the total cumulated available capacity ($R \times T$). ρ was varied between 0.75 and 0.95, in steps of 0.05.

- *Demand pattern*: Integer demands $d_{it} \in \{1, ...R\}$ for each product have been randomly generated according to a procedure similar to the one used in [80].

- *Changeover costs*: Changeover costs c_{ij}^{r} have been randomly generated from a discrete uniform $DU(100, 200)$ distribution. As the resources are assumed to be identical, there is a common changeover cost matrix for all resources.

We also created an additional set of instances, denoted set I, based on the numerical data of the industrial example presented in subsection 5.3. As no data about real demands was available, we randomly generated demand matrices by adapting the procedure described in [80] in order to respect the allocation of production volume among the two product families. The production capacity utilization in the industrial application was close to 100%. This is why we experimented several high values for the production capacity utilization ρ: 0.9, 0.95 and 1.

For each possible combination of problem dimension and production capacity utilization, 10 instances were generated, resulting in a total of $4 \times 5 \times 10 + 3 \times 10 = 230$ instances. All tests were run on a Pentium 4 (2.8 Ghz) with 505 Mb of RAM, running under Windows XP. We used a standard MIP software (CPLEX 8.1.0) with the solver default settings to solve the problems using formulation DLSPPR.

Tables 5.2 to 5.6 display for each set of instances the computational results obtained with the formulation DLSPPR. We grouped the instances with respect to the value of ρ so that each line corresponds to the average value for 10 randomly generated instances. For each set of instances, we provide:

- *Variables* and *Constraints*: the number of variables and constraints.

- *#Opt*: the number of instances out of the corresponding 10 instances that could be solved to optimality within 30 minutes of computation.

	Formulation DLSPPR		
Variables	2670		
Constraints	872		
	#Opt	CPU_{IP}	Gap
$\rho = 0.75$	3	1604s	15%
$\rho = 0.80$	2	1617s	19%
$\rho = 0.85$	0	1800s	13%
$\rho = 0.90$	0	1800s	14%
$\rho = 0.95$	2	1626s	15%

Table 5.2: Initial formulation for the DLSP with parallel resources: results for set A instances

	Formulation DLSPPR		
Variables	10960		
Constraints	2162		
	#Opt	CPU_{IP}	Gap
$\rho = 0.75$	0	1800s	48%
$\rho = 0.80$	0	1800s	44%
$\rho = 0.85$	0	1800s	54%
$\rho = 0.90$	0	1800s	47%
$\rho = 0.95$	0	1800s	41%

Table 5.3: Initial formulation for the DLSP with parallel resources: results for set B instances

- CPU_{IP}: the average computation time in seconds required to find a guaranteed optimal solution. If one could not be found, we use the computation time limit of 1800 seconds.
- *Gap*: for the instances that could not be solved to optimality, the average relative gap value obtained after 30 minutes of computation between the best integer solution and the best lower bound found.

Results from table 5.6 show that thanks to the initial formulation DLSPPR, we are able to obtain guaranteed optimal solutions for 66% of set I instances. Moreover, the average remaining gap between the best integer solution and the best lower bound obtained after 30 minutes of

	Formulation DLSPPR		
Variables	3930		
Constraints	1233		
	#Opt	CPU_{IP}	Gap
$\rho = 0.75$	0	1800s	17%
$\rho = 0.80$	2	1610s	18%
$\rho = 0.85$	0	1800s	19%
$\rho = 0.90$	0	1800s	15%
$\rho = 0.95$	0	1800s	15%

Table 5.4: Initial formulation for the DLSP with parallel resources: results for set C instances

	Formulation DLSPPR		
Variables	16240		
Constraints	3043		
	#Opt	CPU_{IP}	Gap
$\rho = 0.75$	0	1800s	36%
$\rho = 0.80$	0	1800s	42%
$\rho = 0.85$	0	1800s	41%
$\rho = 0.90$	0	1800s	40%
$\rho = 0.95$	0	1800s	41%

Table 5.5: Initial formulation for the DLSP with parallel resources: results for set D instances

	Formulation DLSPPR		
Variables	9024		
Constraints	1538		
	#Opt	CPU_{IP}	Gap
$\rho = 0.9$	10	761s	0.25%
$\rho = 0.95$	4	1190s	0.16%
$\rho = 1$	6	1139s	0.22%

Table 5.6: Initial formulation for the DLSP with parallel resources: results for set I instances

computational is small (below 1%). This indicates the practical usefulness of the proposed formulation at solving small-sized instances arising in industrial applications such as the one described in section 5.3.

However, as can be seen from the results displayed in tables 5.2 to 5.5, the formulation DLSPPR does not seem efficient at providing good solutions for medium to large-sized instances of the problem. Namely, for set A instances, which involve only 5 items, 2 resources and 30 planning periods, the average remaining gap between the best integer solution and the best lower bound obtained after 30 minutes of computation is 15%. For set D instances, the average gap after 30 minutes of computation is above 40%.

These prohibitively long computation times provided us the motivation to develop an approximate solution approach for the problem under study. Moreover, as pointed out by [73], "the (single resource) constant capacity, multi-item discrete lot-sizing problem with start-up costs is NP-hard". Here we deal with the extension of this problem to the case of identical parallel resources. Hence, even if the corresponding feasibility problem can be polynomially solved (see [79]), the optimization problem studied in the present chapter is NP-hard.

5.5 A heuristic solution procedure for the multi-resource DLSP

We describe here the proposed two-phased resource-based decomposition heuristic to solve the DLSP with identical parallel resources and constant capacity. In the sequel, we use the same notation as in section 5.2 for the problem parameters and variables.

5.5.1 General description of the heuristic procedure

The main idea of the proposed heuristic procedure is to decompose the multi-resource DLSP into a series of single-resource problems, each of which being easier to solve than the original multi-resource problem. In order to do this, we allocate each unit demand of matrix D to one of the resources. The demand matrix D is thus decomposed into R matrices D^r where D^r is the demand matrix assigned to resource r. Once the demand has been allocated to the various resources, we solve each resulting single-resource DLSP thanks to the tight MIP formulation DLSP0* described in chapter 3.

The main issue here is therefore to find a good allocation of the demands to the resources, i.e. a good decomposition of the demand matrix D into R matrices D^r. In the sequel, we try to achieve this thanks to a two-phased heuristic:

1. We build an initial solution by finding a feasible decomposition of the demand matrix D.
2. We try to improve it using a local search type procedure.

Before going on with a detailed presentation of each phase, we briefly explain how the improvement phase works. In our approach, a solution for the multi-resource DLSP is represented by R matrices D^r of dimension $N \times T$ such that D^r is a feasible demand matrix assigned to resource r and $D = \sum_{r=1}^{R} D^r$. A neighbor of a solution is defined as another solution for which the demand allocation to the various resources has been slightly modified. These changes in the demand matrices may result from shifting some demand from one resource to another or from exchanging demands between two resources. Starting from an initial (current) solution and using one of these neighborhood operations, we build several neighbored solutions or candidates. The cost of each candidate solution could be evaluated by solving a series of R single-resource DLSP. However, this would lead to prohibitive computation times. This is why we choose to evaluate each candidate solution using a lower bound easier to compute. This lower bound is provided by the linear relaxation of the production planning problem involving R independent parallel resources (see section 5.5.3 for more detail). As this linear relaxation is computed using a tight MIP formulation, the obtained lower bound is of good quality and can be used to evaluate each candidate. At each iteration in the improvement phase, we look in the neighborhood for

the candidate solution with the lowest cost. If this candidate has a cost lower than the cost of the current best solution, this candidate is accepted as the new current solution and a new neighborhood is built. If no better candidate solution can be found, the procedure stops.

A general outline of the algorithm is as follows:

Phase 1: Initialization
Find an initial feasible allocation of demands to resources using algorithm INI presented in subsection 5.5.2 and compute an initial feasible solution.

Phase 2: Improvement
Step 1. Create candidate solutions using one of the neighborhood operations.
Step 2. Evaluate each candidate solution by solving the linear relaxation of the corresponding multi-resource production planning problem involving R parallel independent resources (using the formulation DLSPPRi presented in section 5.5.3).
Step 3. Check whether a candidate solution has an estimated cost lower than the value of the linear relaxation of the current best known solution.
Step 4. If a candidate solution is identified in step 3, go to step 5. Else stop the procedure.
Step 5. Compute the exact cost of the identified candidate solution by solving R single-resource DLSP (using the formulation DLSP0* presented in section 3.2). If the obtained cost is lower than the cost of the current best known solution, replace the current solution by the candidate solution and go to step 1. Else stop the procedure.

5.5.2 Phase I: Building an initial feasible solution

We identified several rationales which could be used to assign demands to resources and build a (good) initial feasible solution:

1. Consecutive demands for a given item should be assigned to the same resource as they are likely to be produced during the same production run.

2. Specializing each resource on a subset of items should help reducing total costs as some changeovers could be avoided.

3. Items with a low volume of demand should be produced on a single resource to avoid doing numerous changeovers for items which represent only a small portion of the total demand.

4. A resource should be dedicated to items with a very high volume of demand.

5. Pairs or subsets of items for which the corresponding changeover costs are lower than the average should be assigned to the same resource.

6. The production capacity utilization should be approximately equal for all the resources, i.e. the workload assigned to each of the resource should be approximately balanced.

In our implementation, we used the first two rationales to build the following initialization algorithm **(INI)**.

> Step 1. Analyze demand matrix D to identify, for each item, sequences of positive demand periods.
>
> Step 2. For $t = 1...T$, for $i = 1...N$,
> 1. Consider the sequence (if any) of positive demand periods for item i beginning in period t.
> 2. Try to assign the whole sequence to the resource which has the largest volume of demand for item i already assigned to it.
> 3. If the assignment tested in step 2.2 is not feasible, try to assign the whole sequence to another resource.
> 4. If no feasible assignment could be found in step 2.3, assign each unit demand in the considered sequence to the first resource for which this assignment is feasible.

Algorithm INI provides a feasible solution in which, as much as possible, sequences of consecutive demands are assigned to the same resource and resources are specialized on a subset of items. However, it would be worth investigating the use of some of the ideas presented above (either individually or combined together) to build initial feasible solutions. Namely, as it is shown in our preliminary computational results, the solutions provided by algorithm INI have a rather high cost.

5.5.3 Phase II: Improving a feasible solution

In the second phase of our solution procedure, we try to improve the initial solution by a local search type procedure.

Representation of a solution

A solution of the multi-resource DLSP is characterized by an assignment of each unit demand in matrix D to a resource $r = 1...R$. It is thus represented by R matrices D^r of dimension $N \times T$,

with D^r giving the demands assigned to resource r and $D = \sum_{r=1}^{R} D^r$. The feasibility of an assignment can be easily checked by the following set of inequalities:

$$\forall r, \forall t, \sum_{i=1}^{N} \sum_{\tau=1}^{t} D_{i,\tau}^{r} \leq t \tag{5.9}$$

Namely, for each resource r, the available capacity up to period t should be sufficient to accommodate the total demand assigned to this resource up to period t.

Definition of the neighborhood

A neighbor of a solution is defined as another solution for which the demand allocation to the various resources has been slightly modified. We use two types of neighborhood operations:

- MOVE: a demand sequence for a given item is shifted from resource r to resource r'.
- SWAP: two demand sequences (possibly for different items) are exchanged between resources r and r'.

We first define a *demand sequence* as a sequence of consecutive periods with a positive demand. But, in order to get some additional flexibility, we allow in some cases a demand sequence to include periods without demand and define a limit for the maximum number of these zero demand periods.

In our computational experiments, we use successively 3 types of neighborhoods:

1. Demand sequences can include as most T periods with zero demand and candidate solutions are obtained by a SWAP operation. Here we try to find good candidates by exchanging a whole line of demand matrix D^r with a whole line of demand matrix $D^{r'}$.
2. Demand sequences cannot include any period with zero demand and candidate solutions are obtained by a MOVE operation.
3. Demand sequences can include as most 1 period with zero demand and candidate solutions are obtained by a SWAP operation.

For each type of neighborhood, we consider all the neighbors that could possibly be obtained with the corresponding operation. As a consequence, we have to evaluate at each iteration a rather large number of candidate solutions.

Evaluation of candidate solutions

Each candidate solution should be evaluated by the cost of the corresponding optimal production plan. This could be done by solving for each neighbor a series of R single-resource DLSP

(using tight formulation DLSP0* described in chapter 3) or by solving the following planning problem involving R independent parallel resources (see formulation DLSPPRi described in the sequel).

This would lead to prohibitive computation times because for each candidate solution, one (or R) mixed-integer program(s) would have to be solved. This is why we chose to evaluate each neighbor by a lower bound easier to compute. This approximate evaluation is provided by the linear relaxation of the production planning problem involving R independent parallel resources. This problem is described below by the formulation DLSPPRi where the same notation as in chapter 5 is used. The formulation DLSPPRi is similar to the formulation DLSPPR of chapter 5. The main difference is that resources are not coupled anymore by inventory balance equations. Namely, variables I_{it}, which represent a common inventory supplied by all the resources and used to meet all demands d_{it}, are replaced by variables I^r_{it} which represent the inventory specific to resource r, supplied only by this resource and used to meet demands d^r_{it} exclusively. The related problem could thus be decomposed into R single-resource subproblems but, as we only solve its linear relaxation, we found it more efficient to solve it as such rather than to decompose it.

(DLSPHeur)

$$min \sum_{r=1}^{R}\sum_{i=1}^{N}\sum_{t=1}^{T} h_i I^r_{it} + \sum_{r=1}^{R}\sum_{i=0}^{N}\sum_{j=0}^{N}\sum_{t=1}^{T} c^r_{ij} w^r_{ijt} \tag{5.10}$$

$$\forall i, \forall t, I^r_{it} = I^r_{i,t-1} + \sum_{r=1}^{R} P^r_{it} y^r_{it} - d^r_{it} \tag{5.11}$$

$$\forall r, \sum_{i=0}^{N} y^r_{i0} = 1 \tag{5.12}$$

$$\forall r, \forall i, \forall t, y^r_{i,t-1} = \sum_{j=0}^{N} w^r_{ijt} \tag{5.13}$$

$$\forall r, \forall j, \forall t, y^r_{jt} = \sum_{i=0}^{N} w^r_{ijt} \tag{5.14}$$

$$\forall i, \forall t, z^r_{it} = \sum_{j:j\neq i} w^r_{jit} \tag{5.15}$$

$$\forall t, \forall i, \forall p \in \{0...D^r_{i,t+1,T}\}, I^r_{it} \geq \sum_{q=1}^{p} \left(1 - y^r_{i,t+q} - \sum_{\tau=t+q+1}^{S_{i,D^r_{i,1,t}+q}} z^r_{i\tau}\right) \tag{5.16}$$

$$\forall r, \forall i, \forall j, \forall t, w^r_{ijt} \geq 0 \tag{5.17}$$

$$\forall r, \forall i, \forall t, I_{it} \geq 0 \tag{5.18}$$

$$\forall r, \forall i, \forall t, y^r_{it} \in \{0, 1\} \tag{5.19}$$

$$\forall r, \forall i, \forall t, z^r_{it} \in [0, 1] \tag{5.20}$$

Hence, at each iteration, we need to solve a rather large number of linear programs. In order to speed up this evaluation step, we used the following ideas:

- Infeasible candidate solutions can be identified without solving a linear program thanks to the feasibility check based on inequalities (5.9).
- For each feasible candidate to be tested, a lot of information is available in advance since it differs from the current solution only by slight changes in the problem data. Therefore, rather than solving each linear program individually "starting from scratch", we use the "advanced basis" option of CPLEX solver in order to reduce the number of simplex iterations.
- We introduce an additional constraint in formulation DLSPPRi enforcing that the total cost of the candidate should be lower than the cost of the linear relaxation of the current best known solution. Thus too expensive candidates are refused as soon as the lower bound provided by the dual simplex algorithm exceeds this limit. This early rejection of expensive candidates aims at saving some simplex iterations and should help decreasing computation times.

Once the cost of every candidate solution has been approximately estimated, we check whether a candidate has a cost lower than the linear relaxation of the current best known solution. If such a candidate can be identified, its exact cost is evaluated by solving to optimality R single-resource problems. If the exact cost of the candidate solution is lower that the cost of the current best known solution, the candidate is accepted as the new current best known solution.

5.6 Preliminary computational results obtained with the heuristic solution procedure

We discuss here some preliminary computational experiments carried out to evaluate the performance of the proposed heuristic solution procedure for the DLSP with parallel resources.

We use the instances of sets A and C described in subsection 5.4 and compare the results obtained using the exact solution approach described in section 5.2 with the results obtained using the approximate solution approach described in section 5.5.

Tables 5.7 and 5.8 display the computational results obtained with both solution approaches. We grouped the instances with respect to the value of ρ so that each line corresponds to the average value for 10 randomly generated instances.

	Formulation DLSPPR		Heuristic		
	CPU_{IP}	Gap_{IP}	Gap_{ini}	CPU_{heur}	Gap_{fin}
$\rho = 0.75$	1604s	15%	37%	118s	18%
$\rho = 0.80$	1617s	19%	40%	139s	23%
$\rho = 0.85$	1800s	13%	38%	220s	22%
$\rho = 0.90$	1800s	14%	48%	230s	22%
$\rho = 0.95$	1626s	15%	35%	245s	20%

Table 5.7: Heuristic solution approach: results for set A instances

	Formulation DLSPPR		Heuristic		
	CPU_{IP}	Gap	Gap_{ini}	CPU_{heur}	Gap_{fin}
$\rho = 0.75$	1800s	17%	55%	257s	34%
$\rho = 0.80$	1610s	18%	50%	256s	32%
$\rho = 0.85$	1800s	19%	56%	280s	41%
$\rho = 0.90$	1800s	15%	53%	384s	36%
$\rho = 0.95$	1800s	15%	54%	347s	41%

Table 5.8: Heuristic solution approach: results for set C instances

For the exact MIP solution approach, we provide:

- CPU_{IP}: the average computation time in seconds required to find a guaranteed optimal solution. If one could not be found, we use the computation time limit of 1800 seconds.
- Gap_{IP}: the average relative gap value obtained after 30 minutes of computation between the best integer solution and the best lower bound. If a guaranteed optimal solution could be found within the time limit, we use a gap of 0%.

For the heuristic solution approach, we provide:

- Gap_{ini}: the average relative gap value between the initial integer solution provided by algorithm INI and the best lower bound obtained with the exact solution approach after at most 30 minutes of computation.
- CPU_{heur}: the average running time of the heuristic.
- Gap_{fin}: the average relative gap between the best solution found by the heuristic procedure and the best lower bound found by the exact solution approach.

Results from tables 5.7 and 5.8 show that the proposed algorithm is rather efficient at improving the initial feasible solution. Namely, the gap between the best known solution provided by the heuristic and the best known lower bound provided by the exact solution approach is reduced from an average of 46% before the improvement phase begins to an average of 29%

after this phase stops. Thus, the lower bound based on the linear relaxation of the formulation DLSPPRi seems to be a good estimation of a candidate solution and can be used efficiently to identify interesting neighbors of the current solution. Furthermore, for set A instances, the average remaining gap (Gap_{fin}) obtained with the heuristic solution procedure is only slightly larger than the gap (Gap_{IP}) obtained with the MIP approach (21% vs 15%) whereas the average computation time is significantly decreased (1700s vs 190s).

However, the computational behavior of the proposed heuristic is not really satisfactory. Indeed, its running times are quite high and the quality of the obtained solutions is poor. Moreover, we were not able to find solutions for sets B and D instances because of exceeded computational memory limits. This may be explained by the following difficulties:

- The cost of the initial solutions provided by algorithm INI is high (46% above the best known lower bound on average).
- A large amount of computational effort is needed to evaluate the candidate solutions at each iteration. This may be due to the combination of several reasons:
  - At each iteration, we generate all possible candidates obtained with the chosen neighborhood operation. As a consequence, the number of candidate solutions to be evaluated (i.e. the number of linear programs to be solved) is rather large.
  - The linear programs to be solved are rather large because of the introduction of the numerous (single-resource) valid inequalities (5.16).
  - The reoptimization carried out by CPLEX solver using "an advanced basis" is not very efficient and the number of simplex iterations carried out before optimality is reached remains high. This is probably accounted for by the fact that a change in the demand matrix coefficients affects both the right-hand side of the inventory balance constraints (5.11) and the left-hand side (variable coefficients) of the valid inequalities (5.16). Thus the linear programs to be sequentially solved at each iteration of the heuristic differ significantly from each other.
- The local search carried out to improve the current solution is rather simple. We use a deterministic sequence of three neighborhoods and the procedure has no way to avoid becoming stuck in a local optimum if it finds one.

To obviate these difficulties, we may try to embed our solution approach in a local search procedure such as simulated annealing or threshold accepting. Namely in these methods, neighbors

are generated one by one by applying a randomly chosen neighborhood operation and local optima can be partially avoided thanks to the fact that a candidate worse than the current solution may be accepted as the new current solution under certain conditions. This may help improving the quality of the obtained solutions. However, the application of both methods will still be hindered by the fact that evaluating a candidate solution will require a rather large amount of computational effort.

Another possible option would be to represent solutions by the resource production schedules rather than by the demand assigned to each resource. The main advantage of this option is that evaluating the cost of a solution will be much easier than in the representation used here. Namely, in the DLSP, thanks to the "all-or-nothing" assumption, fixing the setup pattern for each resource enables one to compute directly the cost of the corresponding production schedule. Therefore, using this solution representation, there is no need anymore to solve a linear program to evaluate each candidate. An approach based on this type of representation can be found in [15]. In this paper, the authors address the single-resource DLSP with sequence-independent setup costs and times and take into account additional operational constraints on batch availability. They developed a two-phased simulated-annealing heuristic to solve their problem and were able to find good solutions for instances involving at most 10 items and 100 periods. In order to solve the problem studied here, it may be worth extending their work to the case of identical parallel resources and sequence-dependent changeover cots. Another interesting direction could be to adapt the approach proposed by [71] in which an extension of the CLSP to the case of parallel resources and sequence-dependent setup costs and times is considered. The author also uses the resource setup patterns to characterize each solution but as he does not assume a discrete production policy, he has to solve a generalized network flow problem to compute optimal production quantities for each resource and evaluate the exact cost of each solution. His solution approach is based on the combined use of a local search metastrategy such as Threshold Accepting and dual reoptimization. It was successful at solving practical problems gathered from the consumer goods industry but running times remained rather high. A simplified version of this procedure, in which a direct evaluation of each candidate solution would be used, could also prove useful at solving the multi-resource DLSP considered here.

5.7 Conclusion

We presented here an initial MIP formulation for the DLSP with sequence-dependent changeover costs and parallel resources. Although the proposed formulation proves useful at solving small-sized instances arising in industrial applications such as the one described in section 5.3, it does

not seem strong enough to provide solutions for larger instances.

This is why we first tried to develop a heuristic procedure aiming at providing good approximate solutions for medium and large-sized instances of the problem. Although the computational behavior of the proposed algorithm is not really satisfactory, this study enabled us to identify several interesting directions for further research, among which is the use of a local search method based on the representation of a solution by the resource setup patterns.

In the next chapter, we focus on improving the initial MIP formulation proposed here and derive a family of strong valid inequalities for the single-item DLSP with a constant capacity, startup costs and parallel resources (i.e. the variant denoted DLS-CC-SC with parallel resources in [78]).

Chapter 6

The multi-resource DLSP: valid inequalities and exact solution approach

We focus on improving the initial MIP formulation introduced in the previous chapter for the DLSP with parallel resources and sequence-dependent changeover costs. We derive a family of strong valid inequalities for the case where the available resources are identical and the production capacity is constant throughout the planning horizon. The results of our computational experiments show that thanks to the proposed enhanced formulation, the efficiency of the Branch & Bound procedure embedded in CPLEX solver can be significantly improved. Moreover, our computational experiments provide some insights about the impact of a partial specialization of the resources on both the algorithmic performance and the total production cost.

6.1 Introduction

In the previous chapter, we discussed the DLSP with sequence-dependent changeover costs and identical parallel resources. We presented a first MIP formulation for this problem. However, even if the proposed formulation was an extension of an existing tight formulation for the single-resource variant of the problem, it does not seem strong enough to be able to solve medium to large-sized instances. Namely, as can be seen from tables 5.2 to 5.5 in section 5.4, the average remaining gap after 30 minutes of computation is between 15% for the smallest studied instances and 47% for the largest studied instances. These results are mainly explained by the fact that the lower bounds provided by the linear relaxation of the initial formulation introduced in section 5.2 only provides a poor approximation to the exact optimal integer solution values.

In order to address this issue, we first tried to develop a heuristic solution procedure where the multi-resource problem is decomposed into a series of single-resource problem. However, the preliminary results obtained with an early version of the algorithm were not satisfactory. A possibility would have been to investigate one of the directions for future research identified in section 5.7, among which was the use of a local search method based on the representation of a solution by the resource setup patterns. But we decided to focus on improving the initial formulation and on extending the valid inequalities available for the single-resource variant of the problem to the multi-resource variants.

The purpose of the present chapter is thus to derive strong valid inequalities for the single-item DLSP with constant capacity, startup cost and several machines (i.e. the variant denoted DLS-CC-SC with several machines in [78]) and to provide some insights about the impact of a partial specialization of the resources on both the algorithmic performance and the total production cost.

This chapter is organized as follows. In section 6.2, we first describe how the initial formulation proposed in chapter 5 can be strengthened by valid inequalities and symmetry-breaking constraints. Computational experiments were carried out to evaluate the impact of the proposed formulation enhancements. In our computational study, we considered two cases:

- the resources are totally versatile: each resource is able to produce the complete set of items (see section 6.3).
- the resources are partially specialized: each resource is able to produce only a predetermined subset of items. These subsets may have some items in common so that the planning problem cannot be decomposed into a series of single-resource problems. (see section 6.4)

Section 6.5 provides the concluding remarks.

6.2 Strengthening the formulation with valid inequalities

We study in this section several ways of strengthening the formulation DLSPPR presented in 5.2. The main idea here is to extend the strong valid inequalities developed by [94] for the single-resource single-item DLSP to the case of parallel resources. However, these valid inequalities are based on the key assumption that the resource capacity is constant throughout the planning horizon. As a result, demand can be normalized without loss of generality and at most one unit of demand can be produced in each time period (i.e. $d_{it} \in \{0,1\}$ and $P_{it} = 1$). The extension of the valid inequalities developed by [94] to the general multi-resource problem is hindered by the fact that with several production resources available, for a given item, several units of demand can be produced in each time period. In what follows, we first provide valid inequalities for the specific case where an item can be produced on at most one resource in a given period. We then propose an extension of the valid inequalities developed by [94] for the general case. In both cases, we assume that there are R identical resources with a constant production capacity, Wagner-Whitin costs and no backlogging. These assumptions are necessary in order to normalize the demand and production capacity as shown e.g. in [35] and [94].

6.2.1 Valid inequalities for a specific case

We first investigate a specific case based on the assumption that an item can be produced on at most one resource in a given period. This assumption is based on the idea that in most optimal production plans, the available resources will not be producing the same type of items in the same period. This additional assumption on the production system may sometimes lead to an increase in the optimal cost but it seems to be rather reasonable for many industrial applications. For instance, the authors of [102], who studied the CLSP with parallel resources, used a similar "no lot splitting" assumption: in their model, the production of an item within a given time period cannot be split among different facilities, it has to be processed on a single facility.

To guarantee that a given item can be produced on at most one resource per period, the following additional constraints are introduced in the formulation DLSPPR:

$$\forall i, \forall t, \sum_{r=1}^{R} y_{it}^{r} \leq 1 \tag{6.1}$$

Thanks to this assumption, demands and production capacity can be normalized without loss of generality ($d_{it} \in \{0,1\}$ and $P_{it}^{r} = 1$) and the property that at most one unit of demand can be produced in each time period holds.

We first introduce notation similar to those used in chapters 3 and 4 :

- $D_{i,t,\tau}$: cumulated demand for item i in the interval $\{t, ..., \tau\}$.
- $S_{i,q}$: q^{th} positive demand period for item i. Note that $S_{i,D_{i,1,t}+q}$ denotes the q^{th} period with positive demand after period t.

We also introduce the start-up variables z_{it}^r defined as:

$$z_{it}^r = \begin{cases} 1 & \text{if the production of a new lot of item } i \text{ begins in period } t \text{ on resource } r, \text{ i.e. if} \\ & \text{a start-up for item } i \text{ takes place at the beginning of period } t \text{ on resource } r, \\ 0 & \text{otherwise.} \end{cases}$$

The start-up variables are linked to the changeover variables through the equations:

$$\forall r, \forall i, \forall t, z_{it}^r = \sum_{j:j\neq i} w_{jit}^r \tag{6.2}$$

With this notation, a whole family of valid inequalities can be deduced from the following result:

Proposition 6.1 *Under the assumption that an item can be produced on at most one resource in a given period, all feasible solutions of DLSPPR satisfy:*

$$\forall t, \forall i, \forall p \in \{1...D_{i,t+1,T}\}, I_{it} \geq \sum_{q=1}^{p} \left(1 - \sum_{r=1}^{R} \left(y_{i,t+q}^r + \sum_{\tau=t+q+1}^{S_{i,D_{i,1,t}+q}} z_{i\tau}^r\right)\right) \tag{6.3}$$

Proof 6.1 *Before providing the proof, we briefly explain the idea underlying (6.3).*

$\sum_{r=1}^{R} \left(y_{i,t+q}^r + \sum_{\tau=t+q+1}^{S_{i,D_{i,1,t}+q}} z_{i\tau}^r\right) = 0$ *if and only if no resource is setup for item i in period $t+q$ and no startup for this item occurs between the period $t+q+1$ and the period where the q^{th} demand after period t occurs, i.e. if and only if no production of item i is possible in the interval $[t+q, S_{i,D_{i,1,t}+q}]$. In this situation, as at most $q-1$ units of demand can be produced in the interval $[t+1, t+q]$ (one unit produced per time period), the quantity needed to satisfy the q^{th} demand on item i after period t should be in stock at the end of period t.*

Now consider an arbitrary integral feasible solution of DLSPPR, say (I, y, w, z). We arbitrarily choose an item i, a period t and a demand occurrence $p \in \{1...D_{i,t+1,T}\}$ and we show that the chosen feasible solution satisfies the corresponding valid inequality. In the sequel, for the sake of simplicity, we drop the item index i.

We denote T_q the q^{th} production period for this item in the feasible solution considered. As we assume that at most one unit of demand can be produced per period, we have $T_1 < T_2 < ... < T_q < ... < T_{D_{1,T}}$. Moreover, because backlogging is not allowed, the q^{th} production period must

occur before the q^{th} demand period: $\forall q, T_q \leq S_q$.

Let q_0 be the highest index such that $T_{D_{1,t}+q} < t+q$. Then we have:

- *$\forall q \leq q_0, T_{D_{1,t}+q} < t+q$. For $q \leq q_0$, the q^{th} demand after period t is produced before period $t+q_0$.*

- *$\forall q > q_0, t+q \leq T_{D_{1,t}+q} \leq S_{D_{1,t}+q}$. For $q > q_0$, the q^{th} demand after period t is produced between $t+q$ and the period $S_{D_{1,t}+q}$ where it occurs. In this case, one of the resources must be setup for item i at least once in the interval $[t+q, S_{D_{1,t}+q}]$. Thus we have:*

$$\forall q > q_0, \sum_{r=1}^{R} \left(y_{i,t+q}^r + \sum_{\tau=t+q+1}^{S_{i,D_{i,1,t}+q}} z_{i\tau}^r\right) \geq 1 \tag{6.4}$$

Hence,

$$\begin{aligned}
&\sum_{\tau=1}^{t}\sum_{r=1}^{R} y_\tau^r + \sum_{q=1}^{p}\Big(\sum_{r=1}^{R}\big(y_{t+q}^r + \sum_{\tau=t+q+1}^{S_{D_{1,t}+q}} z_\tau^r\big)\Big) \\
&\qquad \geq \sum_{\tau=1}^{t}\sum_{r=1}^{R} y_\tau^r + \sum_{q=1}^{q_0}\sum_{r=1}^{R} y_{t+q}^r + \sum_{q=q_0+1}^{p}\sum_{r=1}^{R}\big(y_{t+q}^r + \sum_{\tau=t+q+1}^{S_{D_{1,t}+q}} z_\tau^r\big) && (6.5)\\
&\qquad \geq \sum_{\tau=1}^{t}\sum_{r=1}^{R} y_\tau^r + \sum_{q=1}^{q_0}\sum_{r=1}^{R} y_{t+q}^r + p - q_0 && (6.6)\\
&\qquad \geq D_{1,t} + q_0 + p - q_0 && (6.7)\\
&\qquad \geq D_{1,t} + p && (6.8)
\end{aligned}$$

(6.5) comes from the fact that $\sum_{q=1}^{q_0}\sum_{r=1}^{R}\sum_{\tau=t+q+1}^{S_{D_{1,t}+q}} z_\tau^r \geq 0$. To obtain (6.6), we use the inequalities (6.4). Finally, (6.7) is true because, by definition of q_0, the cumulated demand $D_{1,t}+q_0$ is satisfied by the cumulated production before $t+q_0$, $\sum_{\tau=1}^{t+q_0}\sum_{r=1}^{R} y_\tau^r$, so that $\sum_{\tau=1}^{t+q_0}\sum_{r=1}^{R} y_\tau^r \geq D_{1,t}+q_0$.

As $\sum_{\tau=1}^{t}\sum_{r=1}^{R} y_\tau^r - D_{1,t}$ is the inventory level at the end of period t, this establishes the validity of (6.3). □

6.2.2 Valid inequalities for the general case

When there are R identical parallel resources, the formulation DLSPPR can also be strengthened using a family of strong valid inequalities similar to those developed by [94]. However, the extension is not as easy as in the previous case because, for a given item, the property that at most one unit of demand can be produced in each time period does not hold any more.

When all resources have the same constant capacity throughout the planning horizon, we can define a unit of item i as the maximum quantity of this item that can be produced per period on one resource. Thus production capacity can be normalized ($P^r_{it} = 1$) and demands can be expressed without loss of generality as integer multiples of the production capacity on a resource (i.e. $\forall i, \forall t, d_{it} \in \{0, 1, ..., R\}$).

We introduce the following notation:

- $D_{i,t,\tau}$: cumulated demand for item i in the interval $\{t, ..., \tau\}$.
- $S_{i,q}$: period where the q positive *unit demand* occurs. Note that we may have $S_{i,q} = S_{i,q-1}$ or $S_{i,q} = S_{i,q+1}$ if demand for item i is strictly superior to 1 in a given period. $S_{i,D_{i,1,t}+q}$ denotes the period where the q^{th} positive *unit demand* occurs after period t.

We also use the start-up variables z^r_{it} as defined above.

With this notation, we have:

Proposition 6.2 *If the production planning problem involves R identical parallel resources, all feasible solutions of DLSPPR satisfy:*

$$\forall t, \forall i, \forall p \in \{1...D_{i,t+1,T}\}, I_{it} \geq \sum_{q=1}^{p} \Big(1 - NProd(i,t,q)\Big) \tag{6.9}$$

where $NProd(i,t,q)$ is defined as follows:

- *Let $a \in \mathbb{N}$ and $b \in [0, R-1]$ be defined as $q = a \times R + b$.*
- *Let $(\mathcal{R}_1, \mathcal{R}_2)$ be a partition of $\mathcal{R} = \{1, ...R\}$ such that $|\mathcal{R}_1| = b - 1$.*
- *$NProd(i,t,q)$ is expressed as:*

$$NProd(i,t,q) = \sum_{r \in \mathcal{R}_1} y^r_{i,t+a+2} + \sum_{r \in \mathcal{R}_2} (y^r_{i,t+a+1} + z^r_{i,t+a+2}) + \sum_{r \in \mathcal{R}} \sum_{\tau=t+a+3}^{S_{i,D_{i,1,t}+q}} z^r_{i,\tau} \tag{6.10}$$

Proof 6.2 *Before proceeding to the proof, let us observe that the basic idea underlying (6.9) is the same as the one underlying the valid inequalities (3.10) for formulation DLSP0. The main difference comes from the fact that with R identical parallel resources, it is possible to produce in a given period at most R units of each item instead of at most 1 unit of each item.*

Constraints (6.9) ensure that if the quantity needed to satisfy the q^{th} unit demand on item i after period t cannot be produced between $t+1$ and the period $S_{i,D_{i,1,t}+q}$ where it occurs, it will be in stock at the end of period t. Indeed, the term $NProd(i,t,q)$ is defined so that if $NProd(i,t,q) = 0$, at most $q-1$ units of item i can be produced in the interval $[t+1, S_{i,D_{i,1,t}+q}]$.

Namely, suppose $NProd(i,t,q) = 0$ *for an arbitrary choice of* i, t *and* q. *Let* a *and* b *be the quotient and remainder of the euclidian division of* q *by* R *(i.e.* $q = a \times R + b$*):*

- *In periods* $\tau \in [t+1; t+a]$, *all resources may produce item* i *so that at most* $a \times R$ *units of item* i *can be produced.*
- *In period* $t+a+1$, *we have:* $\forall r \in \mathcal{R}_2, y^r_{i,t+a+1} = 0$ *with* $|\mathcal{R}_2| = R - b + 1$ *so that at most* $b-1$ *units of item* i *can be produced on the resources in set* $\mathcal{R}_1$.
- *In periods* $\tau \in [t+a+2; S_{i,D_{i,1,t}+q}]$, *no production of item* i *is possible. Namely for each* $r \in \mathcal{R}_1$, *the resource cannot be setup for item* i *in period* $t+a+2$ *and no startup for this item can occur between period* $t+a+3$ *and period* $S_{i,D_{i,1,t}+q}$. *Similarly, for each* $r \in \mathcal{R}_2$, *the resource cannot be setup for item* i *in period* $t+a+1$ *and no startup for this item can occur between period* $t+a+2$ *and period* $S_{i,D_{i,1,t}+q}$.

As a consequence, if $NProd(i,t,q) = 0$, *at most* $a \times R + b - 1 = q - 1$ *units of item* i *can be produced in the interval* $[t+1, S_{i,D_{i,1,t}+q}]$ *and the inventory level of item* i *at the end of period* t *should be increased by 1. In proposition 6.2,* $\mathcal{R}_1$ *can be seen as the subset of resources that may produce item* i *only in the interval* $[t+1; t+a+1]$ *and* $\mathcal{R}_2$ *as the subset of resources that may produce item* i *only in the interval* $[t+1; t+a]$.

We now provide the proof for proposition 6.2. Consider an arbitrary integral feasible solution of DLSPPR, say (I, y, w, z). *We arbitrarily choose an item* i, *a period* t *and a demand occurrence* $p \in \{1...D_{i,t+1,T}\}$ *and we show that the chosen feasible solution satisfies all corresponding valid inequalities of type (6.9). In the sequel, for the sake of simplicity, we drop the item index* i.

We denote T_q *the period where the* q^{th} *unit of item* i *is produced in the feasible solution considered. Note that we may have* $T_q = T_{q-1}$ *or* $T_q = T_{q+1}$ *if several resources produce item* i *in a given period. Because backlogging is not allowed, the* q^{th} *production period must occur before the* q^{th} *demand period:* $\forall q, T_q \leq S_q$.

Let $q_0 = a_0 \times R + b_0$ *be the smallest index such that:*

- $b_0 = 1$.
- $T_{q_0} = a_0 + 1$
- $\forall q < q_0, T_q \leq a_0$

We have: $\forall q = a \times R + b \geq q_0, T_q \geq a + 1$. *This means that for* $q \in [q_0; p]$, *the* q^{th} *unit demand after period* t *must be produced between the period* $t+a+1$ *and the period where it occurs. As a consequence:* $\forall q \in [q_0; p], Nprod(t,q) \geq 1$.

Hence,

$$\sum_{\tau=1}^{t}\sum_{r=1}^{R} y_\tau^r + \sum_{q=1}^{p} Nprod(t,q)$$

$$\geq \sum_{\tau=1}^{t}\sum_{r=1}^{R} y_\tau^r + \sum_{q=1}^{q_0-1} Nprod(t,q) + \sum_{q=q_0}^{p} Nprod(t,q) \tag{6.11}$$

$$\geq \sum_{\tau=1}^{t}\sum_{r=1}^{R} y_\tau^r + \sum_{q=1\ldots q_0-1 \ st\ q=1 \ mod\ R} Nprod(t,q) + \sum_{q=q_0}^{p} Nprod(t,q) \tag{6.12}$$

$$\geq \sum_{\tau=1}^{t}\sum_{r=1}^{R} y_\tau^r + \sum_{q=1\ldots q_0-1 \ st\ q=1 \ mod\ R} \sum_{r=1}^{R} y_{t+\lfloor \frac{q}{R} \rfloor}^r + \sum_{q=q_0}^{p} Nprod(t,q) \tag{6.13}$$

$$\geq \sum_{\tau=1}^{t}\sum_{r=1}^{R} y_\tau^r + \sum_{a=1}^{a_0}\sum_{r=1}^{R} y_{t+a}^r + \sum_{q=q_0}^{p} Nprod(t,q) \tag{6.14}$$

$$\geq \sum_{\tau=1}^{t+a_0}\sum_{r=1}^{R} y_\tau^r + p - q_0 + 1 \tag{6.15}$$

$$\geq D_{1,t} + q_0 - 1 + p - q_0 + 1 \tag{6.16}$$

$$\geq D_{1,t} + p \tag{6.17}$$

(6.12) and (6.13) come from the fact that:

$$\sum_{q=1}^{q_0-1} Nprod(t,q) \geq \sum_{q=1\ldots q_0-1 \ st\ q=1 \ mod\ R} Nprod(t,q) \geq \sum_{q=1\ldots q_0-1 \ st\ q=1 \ mod\ R} \sum_{r=1}^{R} y_{t+\lfloor \frac{q}{R} \rfloor}^r$$

(6.14) is obtained using $\forall q \in [q_0; p], Nprod(t,q) \geq 1$. *(6.15) is true because, by definition of* q_0, *the cumulated demand* $D_{1,t} + q_0 - 1$ *is satisfied by the cumulated production before* $t + a_0$, *so that* $\sum_{\tau=1}^{t+a_0}\sum_{r=1}^{R} y_\tau^r \geq D_{1,t} + q_0 - 1$.

As $\sum_{\tau=1}^{t}\sum_{r=1}^{R} y_\tau^r - D_{1,t}$ *is the inventory level at the end of period* t, *this establishes the validity of (6.9).* □

The number of valid inequalities (6.9) is much larger than the number of valid inequalities (6.3). Namely, for each item i, each period t and each demand $p \in \{1\ldots D_{i,t+1,T}\}$, there are $\prod_{q=1}^{p} C_R^{b-1}$ valid inequalities because for each index $q = a \times R + b$, we have C_R^{b-1} possibilities to partition $\mathcal{R}$. This is why in the implementation of the solution procedure, we used the following separation algorithm (SEP).

Given (I^*, y^*, w^*, z^*) the optimal solution of the linear relaxation of DLSPPR:

For $i = 1...N$, for $t = 1...T$, for $p = 1...D_{i,t+1,T}$:

1. For $q = 1...p$:
- consider all partitions $(\mathcal{R}_1, \mathcal{R}_2)$ of $\mathcal{R} = \{1, ...R\}$ and select the partition $(\mathcal{R}_1^q, \mathcal{R}_2^q)$ so as to minimize $\sum_{r\in\mathcal{R}_1} y_{i,t+a+2}^r + \sum_{r\in\mathcal{R}_2}(y_{i,t+a+1}^r + z_{i,t+a+2}^r)$.
- compute $v_q^* = NProd(i,t,q)$ using partition $(\mathcal{R}_1^q, \mathcal{R}_2^q)$.

2. Compute $V = I_{it}^* + \sum_{q=1}^{p} v_q^*$.
- If $V < p$, the valid inequality of type (6.9) obtained by using partition $(\mathcal{R}_1^q, \mathcal{R}_2^q)$ defined in step 1 for each index q in the sum is violated.
- If $V \geq p$, all valid inequalities of type (6.9) corresponding to item i, period t and demand occurrence p are satisfied.

6.2.3 Symmetry-breaking constraints

As mentioned by [52] who discusses the CLSP with parallel machines, an additional difficulty arises when the available resources are identical. Namely, given a solution of the CLSP, a different solution with the same total cost can be created just by renumbering in each time period the machines. As as result, there is a large number of equivalent optimal solutions. It is known that this symmetry is likely to deteriorate the efficiency of the Branch & Bound algorithm, due to unnecessary node duplication. To obviate this problem, [52] proposes several types of symmetry-breaking constraints to be added to the formulation.

In small bucket models, as planning periods are linked by setup and changeover variables, the machines cannot be renumbered independently in each period. Thus these models do not suffer from symmetry problems to the same extent as big bucket models. However, given a solution of the DLSP, i.e. a production sequence for each resource, a distinct solution with the same total cost can be obtained by modifying the assignment of production sequences to resources, i.e. by renumbering the resources globally. In order to avoid any difficulty caused by symmetry in the Branch & Bound procedure and to exclude alternative equivalent solutions, we add symmetry-breaking constraints to formulation DLSPPR. We use constraints similar to the constraints (SBC6) proposed by [52]: we break symmetry by ordering the resources according to decreasing total changeover costs per resource.

$$\forall r = 2...R, \sum_{i=0}^{N}\sum_{j=0}^{N}\sum_{t=1}^{T} c_{ij}^{r-1} w_{ijt}^{r-1} \leq \sum_{i=0}^{N}\sum_{j=0}^{N}\sum_{t=1}^{T} c_{ij}^{r} w_{ijt}^{r} \tag{6.18}$$

6.2.4 Cutting-plane generation procedure

In the computational experiments to be presented in subsections 6.3 and 6.4, the following cutting-plane generation strategy has been implemented to strengthen the formulation DLSPPR:
1. We add symmetry-breaking constraints (6.18) and solve the linear relaxation of the problem using formulation DLSPPR.
2. For each item $i = 1...N$, each time period $t = 1...T$ and each demand occurrence $p = 1...D_{i,t+1,T}$, we look for the most violated valid inequality of type (6.9) using algorithm SEP. If we find one, we add it to the formulation.
3. If at least one violated inequality is found in step 2, we go back to step 1 and repeat until no more violated valid inequalities can be generated.
The resulting strengthened formulation is denoted DLSPPR*.

6.3 Computational results: the case of versatile resources

In this section, we discuss the results of some computational experiments carried out to evaluate the impact of the formulation enhancements presented in section 6.2. We used the same 5 sets of instances as in chapter 5. All tests were run on a Pentium 4 (2.8 Ghz) with 505 Mb of RAM, running under Windows XP. We used a standard MIP software (CPLEX 8.1.0) with the solver default settings to solve the problems, using either formulation DLSPPR or formulation DLSPPR*.

Tables 6.1 to 6.5 show the computational results obtained with the formulations DLSPPR and DLSPPR*, for each set of instances. We grouped the instances with respect to the value of ρ so that each line corresponds to the average value for 10 randomly generated instances. For both series of results, we provide:

- *Variables* and *Constraints*: the number of variables and constraints.
- $\#VI$: the average number of valid inequalities of type (6.9) added to the formulation DLSPPR by the cutting-plane generation procedure.
- $\#Opt$: the number of instances out of the corresponding 10 instances that could be solved to optimality within 30 minutes of computation.
- CPU_{IP}: the average computation time in seconds required to find a guaranteed optimal solution. If one could not be found, we use the computation time limit of 1800 seconds.
- *Gap*: for the instances that could not be solved to optimality, the average relative gap

	Formulation DLSPPR			Formulation DLSPPR*		
Variables	2670			2670		
Constraints	872			873		
#VI	0			580		
	#Opt	CPU_{IP}	Gap	#Opt	CPU_{IP}	Gap
$\rho = 0.75$	3	1604s	15%	10	209s	0%
$\rho = 0.80$	2	1617s	19%	9	624s	2%
$\rho = 0.85$	0	1800s	13%	9	410s	1%
$\rho = 0.90$	0	1800s	14%	10	708s	0%
$\rho = 0.95$	2	1626s	15%	9	733s	4%

Table 6.1: Versatile parallel resources: results for set A instances

	Formulation DLSPPR			Formulation DLSPPR*		
Variables	10960			10960		
Constraints	2162			2163		
#VI	0			1204		
	#Opt	CPU_{IP}	Gap	#Opt	CPU_{IP}	Gap
$\rho = 0.75$	0	1800s	48%	1	1783s	10%
$\rho = 0.80$	0	1800s	44%	0	1800s	9%
$\rho = 0.85$	0	1800s	54%	0	1800s	13%
$\rho = 0.90$	0	1800s	47%	0	1800s	11%
$\rho = 0.95$	0	1800s	41%	0	1800s	12%

Table 6.2: Versatile parallel resources: results for set B instances

value obtained after 30 minutes of computation between the best integer solution and the best lower bound found.

Results from table 6.1 to 6.5 show that thanks to the formulation enhancements proposed in section 6.2, the efficiency of the Branch & Bound procedure is significantly improved. This can be seen as :

- using the formulation DLSPPR*, 35% of the instances could be solved to optimality within 30 minutes of computation as compared to 12% using the formulation DLSPPR.
- the remaining gap after 30 minutes of computation is significantly reduced (e.g. from 15% with the basic formulation to 1.4% with the enhanced formulation for set A instances).
- the number of nodes explored before a guaranteed optimal solution is found or the computation time limit is reached is on average 10 times smaller while using the formulation DLSPPR* than while using the formulation DLSPPR.

These results can be explained mainly by the fact that thanks to the valid inequalities (6.9), the lower bounds provided by the linear relaxation of the problem are significantly improved.

	Formulation DLSPPR			Formulation DLSPPR*		
Variables	3930			3930		
Constraints	1233			1235		
#VI	0			597		
	#Opt	CPU_{IP}	Gap	#Opt	CPU_{IP}	Gap
$\rho = 0.75$	0	1800s	17%	2	1591s	10%
$\rho = 0.80$	2	1610s	18%	2	1622s	11%
$\rho = 0.85$	0	1800s	19%	0	1800s	10%
$\rho = 0.90$	0	1800s	15%	1	1723s	10%
$\rho = 0.95$	0	1800s	15%	1	1800s	9%

Table 6.3: Versatile parallel resources: results for set C instances

	Formulation DLSPPR			Formulation DLSPPR*		
Variables	16240			16240		
Constraints	3043			3045		
#VI	0			1498		
	#Opt	CPU_{IP}	Gap	#Opt	CPU_{IP}	Gap
$\rho = 0.75$	0	1800s	36%	0	1800s	18%
$\rho = 0.80$	0	1800s	42%	0	1800s	23%
$\rho = 0.85$	0	1800s	41%	0	1800s	26%
$\rho = 0.90$	0	1800s	40%	0	1800s	25%
$\rho = 0.95$	0	1800s	41%	0	1800s	24%

Table 6.4: Versatile parallel resources: results for set D instances

	Formulation DLSPPR			Formulation DLSPPR*		
Variables	9024			9024		
Constraints	1538			1539		
#VI	0			114		
	#Opt	CPU_{IP}	Gap	#Opt	CPU_{IP}	Gap
$\rho = 0.9$	10	761s	0.25%	10	338s	0%
$\rho = 0.95$	4	1190s	0.16%	10	358s	0%
$\rho = 1$	6	1139s	0.22%	8	741s	0.12%

Table 6.5: Versatile parallel resources: results for set I instances

This can be seen for example on set A instances for which the use of valid inequalities of type (6.9) results in a reduction of the integrality gap from 42% to 10%. Moreover this formulation improvement is achieved thanks to the generation of a relatively small number of cutting-planes (580 on average for set A instances).

However, for set B to D instances, the remaining gap after 30 minutes of computation remains rather large (above 10%), even with the enhanced formulation DLSPPR*. In order to obviate this problem, we propose in the sequel to use partially specialized resources, i.e. to prohibit the production of some items on some resources. These additional constraints on the production system leads to a reduction of the solution space and thus might help solving the production planning problem more easily. Besides, it corresponds to a rather common industrial practice. In what follows, some computational experiments exploiting this idea are presented.

6.4 Computational results: the case of partially specialized resources

We now discuss the results of some computational experiments carried out to evaluate the impact of a partial specialization of the resources on both the algorithmic performance and the total production cost.

We used the same sets of instances as in the previous subsection but we modified the MIP formulation DLSPPR in order to prohibit the production of certain types of items on some resources.

We first define a "specialization ratio" μ as the ratio between the number of prohibited item-resource combinations and the total number of possible item-resource combinations. Thus $\mu = \frac{0}{NR} = 0$ for versatile resources and $\mu = \frac{NR-N}{NR} = \frac{R-1}{R}$ for totally specialized resources. We experimented two values for μ:

- $\mu = 0.2$ corresponding to the case of a rather low specialization of the resources,
- $\mu = 0.4$ corresponding to the case of a rather high specialization of the resources.

For set A to D instances, we used the following rules to assign types of items to resources:
1. We sort the items in the increasing order of their total demand $D_{i,1T}$ on the planning horizon. We denote i_k the item in the k^{th} position in the resulting sequence.
2. We prohibit production on some resources for the items with the lowest total demands. Table 6.6 provides detailed data about the prohibited item-resource combinations for each set of instances and each value of μ.

Instances	$\mu = 0.2$	$\mu = 0.4$
set A	$(i_1, 2)$	$(i_1, 2)$; $(i_4, 2)$
	$(i_2, 1)$	$(i_2, 1)$; $(i_3, 1)$
set B	$(i_1, 2)$; $(i_4, 2)$	$(i_1, 2)$; $(i_2, 2)$; $(i_7, 2)$; $(i_8, 2)$
	$(i_2, 1)$; $(i_3, 1)$	$(i_3, 1)$; $(i_4, 1)$; $(i_5, 1)$; $(i_6, 1)$
set C	$(i_1, 3)$	$(i_1, 3)$; $(i_2, 3)$
	$(i_3, 2)$	$(i_1, 2)$; $(i_3, 2)$
	$(i_2, 1)$	$(i_2, 1)$; $(i_3, 1)$
set D	$(i_1, 3)$; $(i_6, 3)$	$(i_2, 3)$; $(i_3, 3)$; $(i_4, 3)$; $(i_5, 3)$
	$(i_2, 2)$; $(i_5, 2)$	$(i_1, 2)$; $(i_3, 2)$; $(i_4, 2)$; $(i_6, 2)$
	$(i_3, 1)$; $(i_4, 1)$	$(i_1, 1)$; $(i_2, 1)$; $(i_5, 1)$; $(i_6, 1)$

Table 6.6: Partially specialized parallel resources: prohibited item-resource (i_k, r) combinations

Our choice for selecting the prohibited item-resource combinations is based on the following rationale. We first reduce the number of possible producing resources for the items with a low volume of demand, i.e. for the items in the first positions in our sorting. The underlying idea is to avoid doing numerous changeovers for items which represent only a small portion of the global demand. In the resulting planning problem, the link between resources is thus created by the items with the highest volume of demand, the production of which has to be allocated among the various resources during the production planning process. Second, we attempt to obtain an approximately balanced volume of preassigned demands for each resource. For example, in set A instances and $\mu = 0.4$, we assigned to resource 1 the item with the lowest volume i_1 and the item with a rather high volume i_4, and to resource 2 the items i_2 and i_3 with a medium volume. Thus, the fraction of the production capacity of a resource that will be devoted to items which can be produced only on this specific resource is approximately the same for all resources.

For set I instances, we followed the policy used by the plant production managers, i.e. we prohibited production of family 2 products only on resource 1.

We used the formulation DLSPPR described in subsection 5.2 and eliminated variables corresponding to prohibited setup states or changeovers. In the presence of partially specialized resources, the model does not suffer from symmetry problems so that there is no need to use constraints of type (6.18) to exclude alternative equivalent solutions. The obtained basic formulation is then strengthened according to a cutting-plane generation procedure similar to the one presented in subsection 6.2. We used valid inequalities of type (6.9) for the items which can be produced on several resources and valid inequalities of type (3.10) for items which can be produced on a single resource. The resulting strengthened formulation is denoted DLSPPRs*.

Table 6.7 to 6.11 display the results obtained using formulation DLSPPRs* for the case of partially specialized resources ($\mu = 0.2$ or $\mu = 0.4$). *Variables*, *Constraints*, $\#VI$, $\#Opt$, CPU_{IP},

	$\mu = 0.2$					$\mu = 0.4$				
Variables	1950					1200				
Constraints	812					511				
#VI	572					574				
	#Inf	#Opt	CPU_{IP}	Gap	%AC	#Inf	#Opt	CPU_{IP}	Gap	%AC
$\rho = 0.75$	0	10	41s	0%	3%	2	8	13s	0%	11%
$\rho = 0.80$	2	8	93s	0%	2%	4	6	20s	0%	7%
$\rho = 0.85$	2	8	105s	0%	4%	2	8	17s	0%	11%
$\rho = 0.90$	0	10	92s	0%	5%	2	8	13s	0%	15%
$\rho = 0.95$	0	10	111s	0%	4%	4	6	10s	0%	14%

Table 6.7: Partially specialized parallel resources: results for set A instances

	$\mu = 0.2$				$\mu = 0.4$			
Variables	6160				4800			
Constraints	1760				1521			
#VI	1192				1176			
	#Inf	#Opt	CPU_{IP}	Gap	#Inf	#Opt	CPU_{IP}	Gap
$\rho = 0.75$	0	4	1310s	6%	2	8	477s	0%
$\rho = 0.80$	0	3	1510s	4%	0	9	559s	1%
$\rho = 0.85$	1	0	1800s	6%	3	5	1114s	0%
$\rho = 0.90$	1	2	1486s	6%	4	6	512s	0%
$\rho = 0.95$	0	2	1564s	4%	5	3	922s	2%

Table 6.8: Partially specialized parallel resources: results for set B instances

Gap are defined as in subsection 6.3. We also provide *#Inf*, the number of instances out of the corresponding 10 instances that are infeasible while using the assignment rules described in table 6.6 and $\%AC$, the mean increase in the optimal cost caused by the partial resource specialization. $\%AC$ is defined as the relative difference between the optimal cost of the production plan obtained with partially specialized resources and the optimal cost of the production plan obtained with versatile resources. We computed $\%AC$ only for set A instances because we did not have enough instances solved to optimality in sets B to D.

We now discuss the results from tables 6.1-6.4 ($\mu = 0$) and tables 6.7-6.10 ($\mu = 0.2$ or 0.4). Comparison of the computational results obtained for the different values of μ shows that the partial specialization of the production resources significantly improves the algorithmic performance. This can be seen for example on set A instances for which the mean computation time is reduced from 536s for $\mu = 0$ to 88s for $\mu = 0.2$ and 15s for $\mu = 0.4$. Similarly, for set B instances, the mean remaining gap after 30 minutes of computation is decreased from 11% for $\mu = 0$ to 5% for $\mu = 0.2$ and to less than 1% for $\mu = 0.4$. This improvement in the efficiency of the Branch & Bound procedure can be explained mainly by the reduction of the MIP size (number of variables and constraints) thanks to prohibited item-resource combinations and by

	$\mu = 0.2$				$\mu = 0.4$			
Variables	2850				2130			
Constraints	1053				873			
#VI	575				598			
	#Inf	#Opt	CPU_{IP}	Gap	#Inf	#Opt	CPU_{IP}	Gap
$\rho = 0.75$	0	6	1004s	2%	1	9	81s	0%
$\rho = 0.80$	2	3	1272s	2%	5	5	160s	0%
$\rho = 0.85$	1	3	1491s	3%	5	5	81s	0%
$\rho = 0.90$	2	5	1058s	2%	5	5	52s	0%
$\rho = 0.95$	2	4	1118s	3%	6	4	106s	0%

Table 6.9: Partially specialized parallel resources: results for set C instances

	$\mu = 0.2$				$\mu = 0.4$			
Variables	11200				7120			
Constraints	2560				2083			
#VI	1409				1451			
	#Inf	#Opt	CPU_{IP}	Gap	#Inf	#Opt	CPU_{IP}	Gap
$\rho = 0.75$	0	0	1800s	13%	4	0	1800s	4%
$\rho = 0.80$	0	0	1800s	16 %	3	0	1800s	9%
$\rho = 0.85$	0	0	1800s	20 %	1	0	1800s	10%
$\rho = 0.90$	0	0	1800s	17 %	1	0	1800s	10%
$\rho = 0.95$	0	0	1800s	19 %	3	0	1800s	11%

Table 6.10: Partially specialized parallel resources: results for set D instances

	$\mu = 0.25$				
Variables	6000				
Constraints	1296				
#VI	137				
	#Infeas	#Opt	CPU_{IP}	Gap	%AddCost
$\rho = 0.9$	0	10	21s	0%	1.6%
$\rho = 0.95$	0	10	15s	0%	1.4%
$\rho = 1$	0	10	9s	0%	2.0%

Table 6.11: Partially specialized parallel resources: results for set I instances

the substitution of the multi-resource valid inequalities of type (6.9) by stronger valid inequalities of type (3.10) for the items which can be produced on a single resource.

However the cost of a high specialization of the production resources is rather high. Namely, there is a large proportion of instances that are feasible while using versatile resources but become infeasible while using specialized resources (26% of the instances when $\mu = 0.4$). Moreover, for set A instances, the increase in the optimal cost is significant (11.6% on average when $\mu = 0.4$). As a consequence, a reasonable approach could be to use a low specialization of the production resources. As shown by our computational experiments, this would lead to a good compromise between an improved algorithmic performance and a deteriorated optimal cost. We note here that the use of randomly generated instances makes it more difficult to find general rules to prohibit item-resource combinations. In industrial applications where items can be grouped into families or where the total volume of demand is unequally allocated amongst the items, it may be easier to identify a natural way of specializing the production resources.

This is the case for the small industrial example presented in section 5.3. Comparison of results from tables 6.5 and 6.11 show that the instances are much easier to solve while using the partial specialization of the resources defined by the plant management. This can be seen as the mean computation time is divided by a factor of 30 (from 479s to 15s). Moreover, this improvement can be achieved at the expense of a rather small increase (1.7% on average) in the optimal cost of the obtained production plans. Hence these results, although preliminary, seem to validate the policy followed by the production managers to plan the two extruding resources.

6.5 Conclusion

We focused on strengthening the basic formulation proposed in chapter 5 for the DLSP with parallel resources and sequence-dependent changeover costs. We derived a family of strong valid inequalities for the case where the available resources are identical and the production capacity is constant throughout the planning horizon. These valid inequalities were obtained by extending the valid inequalities proposed by [94].

The results of our computational experiments show that thanks to the proposed enhanced formulation, the efficiency of the Branch & Bound procedure embedded in CPLEX solver can be significantly improved. Moreover, our computational study provides some insights about the impact of a partial specialization of the resources on both the algorithmic performance and the total production cost. Namely, our results seem to indicate that a reasonable approach in an industrial context could be to use a low specialization of the production resources in order to reach a good compromise between an improved algorithmic performance and a deteriorated

optimal cost.

To conclude, several interesting directions for future research are worth mentioning:

- In chapter 5, we identified several interesting directions for the development of a heuristic solution procedure for the problem. It may be worth investigating one of these directions as it should enable us to solve larger instances of the problem. The obtained algorithm could be used either alone or as part of a hybrid optimization procedure. We could for instance speed up the Branch & Bound procedure embedded in a commercial solver by providing it good feasible solutions obtained with a heuristic algorithm.

- We assumed in the present study that there is no changeover times between production runs of different items. For instance, in the small industrial example presented in section 5.3, we did not model explicitly changeover times although they represent a significant amount of production loss. For the sake of simplicity, we chose to reduce the available capacity by 20% in each time period but this may lead to infeasible or inadequate production plans. Considering positive changeover times would thus be an important further step towards a better representation of real life planning problems.

- In the small industrial example, there is another operational aspect that we did not take into account: the extruding machines use some tools with a limited lifetime to produce propellant tubes. When one of this tool is used up, it must be changed so that there are two types of changeovers on the machines: changeover due to a transition between two production runs of different items and changeover within a production lot due to tool wear. A similar situation is described in [82] and in chapter 14 of [78] but, to the best of our knowledge, a MIP formulation for the DLSP taking this operational aspect into account has not been proposed yet. This would thus deserve further analysis.

- Finally, we only considered single-level problems. However in many industrial applications, we have to deal with multi-level product structures. It would thus be worth addressing the multi-level extensions of the models studied here.

Chapter 7

Conclusion and future research

7.1 Conclusion

Lot-sizing is one of the many issues arising in the context of production planning. Its main objective is to determine the timing and level of production so as to reach the best possible trade-off between minimizing setup and inventory holding costs and satisfying customer demand. When a limited production capacity and a deterministic time-varying demand rate are assumed, lot-sizing leads to the formulation of large-sized mixed-integer programs, most of which are hard to solve.

In the present work, we dealt with one of the many capacitated dynamic lot-sizing models, the Discrete Lot-sizing and Scheduling Problem or DLSP, and studied several variants of this problem where changeover costs and/or times are sequence-dependent. In [98], the author proposes to solve the DLSP with sequence-dependent changeover costs using a tight MIP formulation and a standard commercial solver. Our contributions relate to extensions of this work to cases where additional relevant industrial concerns are incorporated in the model.

In terms of problem modelling, we investigated the integration of various operational aspects into the model:

- the presence of a multi-attribute product structure which can be exploited to reduce the size of the optimization problem,
- the integration of positive changeover times to better model the production loss caused by a changeover,
- the presence of identical parallel resources that need to be planned simultaneously.

For each of these extensions, we proposed a solution procedure aiming at providing exact optimal solutions: a tight MIP formulation for the corresponding problem variant is derived and the resulting mixed-integer program is solved thanks to a commercial MIP solver. Moreover, we carried out computational experiments to evaluate these solution procedures. In general, our results show the practical usefulness of the proposed algorithms at solving medium to large-sized instances with a reasonable computational effort.

7.2 Future research

There are several challenging options for future research.

First, in order to derive tight MIP formulations, we made several assumptions on the production system to be planned. Further analysis would be required in order to better model

industrial applications by dropping some of these assumptions and integrating additional relevant aspects such as a multi-level product structure or a time-varying production capacity. Namely, as shown by our literature review in chapter 2, most multi-level capacitated lot-sizing models assume sequence-independent changeover costs. It may thus be worth investigating the multi-level extension of the DLSP with sequence-dependent changeover costs. Besides, we assume in the present work a constant production capacity throughout the planning horizon in order to be able to use or derive strong valid inequalities for the single-item subproblems embedded in our models. However it is rather common in an industrial context that capacity varies over time. Thus, even if single-item subproblems are NP-hard when capacity is time-varying, integrating a time-varying capacity in the proposed models could be an interesting subject for future research.

Second, the lot-sizing models discussed here focus on the production stage of the supply chain. However, separate optimization of the supply chain activities may result in a suboptimal global solution. It may thus be worth investigating the integration of lot-sizing into more global models. In the case where products have to be manufactured and shipped to different distribution centers, retailers or end customers, it makes sense to consider production and distribution simultaneously at an operational level. In such a situation we should consider fixed and variables costs for both production and transportation and coordinate lot-sizing and routing decisions. Examples of integrated production-distribution planning models can be found in [24], [25], [32], [39], [103], [75], [81] and [91].

Finally, one of the major limitations of the lot-sizing models discussed in the present work is the assumption of deterministic demand and processing times. Production planning is mostly based on data about future demands which are estimated by forecasting models. But there will always be a more or less important forecast error. Moreover the production process may be affected by uncertainties such as stochastic operation yields, quality problems or machine failures. These uncertainties, both in demand and in processing times, may lead to a reduced product availability and may thus deteriorate the customer service offered by the company. In a deterministic planning environment, these problems are usually tackled by using safety stocks or by planning production on a rolling horizon basis. But even if coupled with one of these procedures, deterministic lot-sizing models fail at capturing the complexity of a stochastic environment. Moreover, as pointed out by [55], deterministic and stochastic lot-sizing models may suggest qualitatively different optimal solutions. The integration of uncertainties into lot-sizing models thus opens an interesting area for further research.

Appendix A

Optimizing glass coating lines: MIP model and valid inequalities

We provide the revised version of a paper in which we studied an optimization problem arising in the context of glass production. This work was carried out as part of our thesis project. However it does not relate directly to production planning but rather to production line design. We therefore include it as an appendix to our main manuscript. This paper is currently under review for *European Journal of Operational Research*.

A.1 Introduction

This work is motivated by an industrial problem arising in the glass industry in connection with a specific transformation of flat glass called *glass coating*. Glass coating consists of depositing in vacuum thin layers of metal on the surface of glass sheets. As a general rule, the process involves several layers of distinct metals. This aims at giving the glass additional properties such as a better thermal insulation: see e.g. [1] for an overview on the applications of coated glass. According to the sequence and thickness of the layers, the property obtained is different: hence production managers have to cope with some product diversity.

Glass coating can be done on specific production lines called "soft-coating lines" using a process called "cathodic sputtering" ([87]). Basically, these lines are made of a number of metallic cathodes, each being used to spray or "sputter" a specific metal on the glass sheets. Each sheet can go only once through the production line: during this single passage, all the metal layers to be deposited on the sheet must be sputtered following the sequence imposed by the product specifications.

The cathodes are ordered along the line: a configuration of the line corresponds to a sequence of cathodes. A cathode contains a finite volume of a single metal. Once the metal of a cathode has been used up, the cathode must be changed. But, due to technical reasons, this requires a line shutdown during several days. Because of these time-consuming changeovers, soft-coating lines are operated according to the following organization. All cathodes on the line are changed together during a line shutdown. After this, production takes place continuously with this configuration during the next production run, the duration of which is typically about one month. When the run is over, all cathodes are changed and a new configuration is set up.

The problem addressed in the present paper concerns the determination of the optimal configuration to be set up between two line shutdowns. This decision can be based on reliable future demand forecasts: the requested products and the anticipated surface to be coated are assumed to be perfectly known. The configuration set up at the beginning of a production run should be able to process all needed products in the quantity requested until the next production shutdown. In this context, determining the configuration to be set up consists of selecting among a set of available cathodes the ones to be placed on the line, ordering them along the production line and deciding how to use them to process the requested products. Because of its limited capacity, a cathode may not be sufficient to sputter the entire volume needed to process a given layer. Thus we have to consider the situation where a layer is sputtered by several cathodes placed at different positions on the line. The objective is to minimize the number of cathodes to be placed on the line. Indeed, the larger the number of cathodes to be placed on the line during a setup, the

greater the changeover operations will be and the more time will be lost for useful production. Investigation of models and algorithms for solving the resulting discrete optimization problem is the subject addressed in the present paper.

The problem under study shares some common features with a string processing problem called the Shortest Common Supersequence problem (see e.g. [68]). It is however significantly different due to various extra constraints which must be taken into account, one of the most significant being the limitations imposed on cathode capacity, which frequently result in the use of a significant number of additional positions.

The problem of optimizing glass coating lines can also be related to the "Assembly Line Design Problem" (ALDP). In the ALDP, a production line is described as a series of workstations, each being responsible for performing a specific set of assembly tasks. The problem consists of selecting a piece of equipment for each workstation and deciding which tasks should be performed by which workstation. Recent overviews on the literature on the ALDP can be found in [3] and [13]. Nevertheless, the glass coating line problem is different from the ALDP studied in most papers (see e.g. [76] and [16]). The main reason is that in the ALDP, each assembly task is performed exactly once, i.e is assigned to a single workstation on the line, whereas on a glass-coating line, a layer can be sputtered by several cathodes placed at different positions on the line. The problem of optimizing glass coating lines can thus be seen as an extension of the ALDP to the case where a task can be assigned to more than one workstation ("parallelized" in the terminology of [13]). To the best of our knowledge, the only solution approach already available to deal with this particular extension of the "Assembly Line Design Problem" can be found in [12] who propose a flexible heuristic search procedure that can be modified to solve various extensions of the "Assembly Line Balancing Problem". In their paper, the authors assume that the processing time of a parallelized task is equally allocated to the chosen workstations. On the contrary, on a glass coating line, the volume of a layer sputtered by several cathodes can be unequally divided among the various cathodes so that we have to decide about the allocation of the metal volume to be sputtered among the chosen cathodes. Moreover, their solution approach is purely heuristic whereas ours being based on a mixed integer linear programming (MIP) model is intended to provide exact optimal solutions.

The paper is organized as follows. In section A.2, we introduce an initial mathematical formulation of this problem as a mixed integer linear program. In section A.3, we consider several ways to strengthen this initial formulation by adding valid inequalities of various types. In section A.4, we discuss the results of some computational experiments showing the practical usefulness of the proposed valid inequalities at improving the efficiency of a Branch & Bound

type procedure. Conclusions and perspectives for future work are presented in section A.5.

A.2 Problem formulation

We wish to determine the optimal configuration of a glass coating line to be set up between two production shutdowns. In this section, we introduce an initial formulation for this optimization problem as a mixed integer linear program. To describe the problem precisely we introduce the following notation.

The set of anticipated requirements is supposed to involve M metals and P distinct final products. Each metal type is indexed by m: $m = 1, 2, ..., M$. Each product, indexed $p = 1, 2, ..., P$, is made of a glass sheet on which O_p layers are to be sputtered. For a given product p, a layer $o = 1, 2, ..., O_p$ is made of a specific metal denoted m_{po} and its thickness is given by e_{po}. The anticipated surface of product p to be processed during the production run, S_p, being known, the volume of metal m_{po} needed to sputter the o^{th} layer of product p can be deduced as: $V_{po} = e_{po} * S_p$.

Possible positions of cathodes on the production line are indexed by $i = 1, 2, ..., N$. These positions are ordered according to the orientation of production flow. The cathode $i+1$ is located immediately after the cathode i along the line.

Available cathodes correspond to C types of cathodes. For each type of cathode $c = 1, 2..., C$, we assume that we know m_c, the corresponding metal, V_c, the volume of available metal in each cathode and ν_c, the number of cathodes belonging to this type. We agree to use an additional type $c = 0$ (the empty cathode) to represent free positions on the line. For each metal m, we denote $C(m)$ the subset of cathode types c such that $m_c = m$. The complementary subset is denoted $\overline{C}(m) = \{c = 1, 2, ..., C \text{ st } m_c \neq m\}$.

A.2.1 First formulation of the problem as a MIP

Here we first provide a mathematical statement of the problem involving the following decision variables:

- $z_c^i = 1$ if a cathode of type c is placed in line position i, $z_c^i = 0$ otherwise.
- $y_{po}^i = 1$ if the cathode placed in position i is used to sputter the o^{th} layer of product p, $y_{po}^i = 0$ otherwise.
- x_{po}^i gives the proportion of V_{po} sputtered by the cathode placed in the i^{th} position. Thus all the x_{po}^i are continuous variables in $[0; 1]$.

Considering the criterion of minimizing the total number of positions used, the formulation

proposed is:

$$min \sum_{i=1}^{N} \sum_{c=1}^{C} z_c^i \tag{A.1}$$

$$\forall i, \sum_{c=0}^{C} z_c^i = 1 \tag{A.2}$$

$$\forall c, \sum_{i=1}^{N} z_c^i \leq \nu_c \tag{A.3}$$

$$\forall i, \forall p, \forall o, y_{po}^i + \sum_{c \in \overline{C}(m_{po})} z_c^i \leq 1 \tag{A.4}$$

$$\forall p, \forall o, \sum_{i=1}^{N} x_{po}^i = 1 \tag{A.5}$$

$$\forall i, \forall p, \forall o, x_{po}^i \leq y_{po}^i \tag{A.6}$$

$$\forall p, \forall (o, o') \text{ st } o > o', \forall (i, i') \text{ st } i < i', y_{po'}^{i'} + y_{po}^i \leq 1 \tag{A.7}$$

$$\forall i, \forall m, \sum_{c \in C(m)} V_c z_c^i - \sum_{(p,o) \text{ st } m_{po}=m} x_{po}^i V_{po} \geq 0 \tag{A.8}$$

$$\forall i \in [1; N-1], z_0^i \leq z_0^{i+1} \tag{A.9}$$

$$\forall i, \forall p, \forall o, y_{po}^i \in [0; 1], x_{po}^i \in [0; 1] \text{ and } \forall i, \forall c, z_c^i \in [0; 1] \tag{A.10}$$

$$\forall i, \forall p, \forall o, y_{po}^i \in \{0; 1\} \text{ and } \forall i, \forall c, z_c^i \in \{0; 1\} \tag{A.11}$$

The objective expressed by (A.1) is to minimize the total number of cathodes placed on the line. Constraints (A.2) ensure that at most one cathode is placed in position i. $z_0^i = 1$ means that the i^{th} position on the line is free. Constraints (A.3) ensure that no more than the number of available cathodes of type c, ν_c, are placed on the line. Constraints (A.4) guarantee the compatibility between the metal m_{po} for layer o of product p and the metal of the cathode placed in position i: we cannot open the connection $y_{po}^i = 1$ if the cathode in position i contains a metal other than m_{po}. Equalities (A.5) ensure that the demand is perfectly met: all the volume of each layer should be sputtered. Constraints (A.6) link the continuous variables x_{po}^i with the binary variables y_{po}^i: some volume of the o^{th} layer of product p can be sputtered in position i only if the connection is open. Precedence constraints (A.7) force compliance with the order according to which the layers of a product should be sputtered: for a given product p, the layer o which is above the layer o' should not be processed with a cathode i placed before the cathode i' if the latter is used to sputter o'. Inequalities (A.8) guarantee that the limited capacity of the cathodes is not exceeded: for each type of metal, the volume remaining at the end of the

Table A.1: Optimizing glass coating lines: data for problem P0

Product $p=1$	o	1	2	3	4
	Metal m_{po}	Ag	Au	Ti	Ag
	Volume V_{po}	210	400	100	100
Product $p=2$	o	1	2	3	
	Metal m_{po}	Ag	Ti	Au	
	Volume V_{po}	410	800	200	
Product $p=3$	o	1	2	3	
	Metal m_{po}	Au	Pt	Ti	
	Volume V_{po}	4000	1000	1000	

Cathode c	1	2	3	4
Number ν_c	5	5	5	5
Metal m_c	Ag	Ti	Au	Pt
Volume V_c	1000	2000	3000	2000

Table A.2: Optimizing glass coating lines: optimal solution for problem P0

Line	Volume sputtered for the layer (p,o)		
i Cathode	$p=1$	$p=2$	$p=3$
1 Ag, 1000	(1,1) 210	(2,1) 410	
2 Au, 3000	(1,2) 400		(3,1) 1200
3 Ti, 2000	(1,3) 100	(2,2) 800	
4 Au, 3000		(2,3) 200	(3,1) 2800
5 Pt, 2000			(3,2) 1000
6 Ti, 2000			(3,3) 1000
7 Ag, 1000	(1,4) 100		

production run in the cathode placed in position i should be non-negative. Constraints (A.9) are used to enforce consecutive empty positions at the end of the line in case all positions are not used.

A.2.2 A small illustrative example

Problem P0 is a small instance we use in order to illustrate the problem and its resolution. P0 involves $M=4$ metals, $P=3$ products made of 3 or 4 layers and $N=12$ positions on the line. Table A.1 gives the numerical data relative to this example. The optimal configuration in this case is a sequence of $Z^*=7$ cathodes. Table A.2 gives this sequence as well as the optimal use of cathodes to process the 3 products. We may notice that the first layer of product $p=3$ is sputtered by two cathodes made of gold (placed at positions 2 and 4). This is due to the fact that the volume of metal needed to sputter this layer exceeds the capacity of a single cathode made of gold. In the sequel, P0 is used to illustrate various features of the proposed resolution method.

A.3 Valid inequalities

The formulation introduced in section 2 enables us to solve exactly only small instances: computation times for industrial problems of larger size using one of the best currently available commercial MIP solver are prohibitively long as can be seen from table A.4. A possible explanation for this lies in the observation that the linear relaxation of the problem (A.1)-(A.11) only provides a poor approximation to the exact optimal integer solution values. In order to address this issue, we investigate below several ways of strengthening the initial formulation (i.e. of reducing the integrality gap). The enhancements discussed here focus on various aspects of the problem under study, namely:

- available cathodes have a limited capacity,
- only one metal can be assigned to each position on the line,
- precedence constraints between layers of a given product must be respected.

In section 4, computational experiments will be reported showing that, thanks to these enhancements, the linear relaxation is tightened and instances of significantly larger size can be solved exactly with standard integer linear programming tools.

A.3.1 Valid inequalities from limited capacity of available cathodes

For each metal, we can compute a lower bound on the number of cathodes containing this metal to be placed on the line. This gives M valid inequalities (A.12) that can be added to the formulation.

$$\forall m, \sum_{i=1}^{N} \sum_{c \in C(m)} z_c^i \geq \left\lceil \frac{\sum_{p=1}^{P} \sum_{o=1...O_p \text{ st } m_{po}=m} V_{po}}{Max\{V_c, c \in C(m)\}} \right\rceil \tag{A.12}$$

Namely, for each metal m, dividing the global volume of metal needed to process all final products by the volume contained in the maximum capacity cathode containing metal m and rounding up gives the minimal number of cathodes of type $c \in C(m)$ to be placed on the production line.

A.3.2 Valid inequalities from metal compatibility constraints

In this subsection, we discuss another family of valid inequalities to further strengthen the formulation. In a first step, we derive a series of binary exclusion constraints. These constraints are logical consequences of the formulation (A.1)-(A.11). In a second step, we exploit the special structure of these constraints to derive stronger valid inequalities which correspond to *maximal clique constraints* in the underlying graph. (See e.g. [74]). We note here that similar approaches have been used on other optimization problems such as assembly line design ([76]), harvest scheduling ([44]), cellular telecommunications networks design ([57]) or air line crew scheduling

([104]). We observe however that in all the above-mentioned references the structures of the underlying constraint graphs were significantly different from those studied in the present paper, leading to clearly distinct separation algorithms. In particular, in [76], the separation of clique constraints is carried out using either complete enumeration or a greedy heuristic whereas our separation algorithms are exact and polynomial.

We first state various families of binary exclusion constraints. These constraints are implied by the constraints (A.2)-(A.11) of the initial formulation but their explicit statement turns out to be useful with respect to strengthening. They link pairs of binary variables related to the same position i on the production line, but to different products, layers or types of cathodes:

$$\forall i, \forall c, \forall p, \forall o \text{ st } m_c \neq m_{po}, z_c^i + y_{po}^i \leq 1 \tag{A.13}$$

$$\forall i, \forall p, \forall o, \forall p', \forall o' \text{ st } m_{po} \neq m_{p'o'}, y_{po}^i + y_{p'o'}^i \leq 1 \tag{A.14}$$

$$\forall i, \forall p, \forall (o, o') \text{ st } o \neq o', y_{po}^i + y_{po'}^i \leq 1 \tag{A.15}$$

$$\forall i, \forall c, \forall c' \text{ st } c \neq c', z_c^i + z_{c'}^i \leq 1 \tag{A.16}$$

Constraints (A.13) state that for a given position, there is an incompatibility between a cathode and a given layer if the corresponding metals are different. Similarly, constraints (A.14) state that two layers made of distinct metals cannot be sputtered at the same position. Constraints (A.15) are a consequence of the precedence constraints: they guarantee that two layers belonging to a given product will not be sputtered at the same position on the production line. Constraints (A.16) ensure that two distinct cathodes will not be placed at the same position on the production line.

We next investigate a strengthened formulation for the constraints (A.13)-(A.16) based on the analysis of the associated constraint graph and the use of valid inequalities deduced from maximal cliques.

In the constraint graph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$, a node $v \in \mathcal{V}$ represents either a type of cathode placed in a given position (i.e a variable z_c^i) or a metal layer sputtered in a given position (i.e a variable y_{po}^i). There is an edge $a \in \mathcal{A}$ between two pairs of nodes if the corresponding variables are linked by one of the binary exclusion constraints (A.13)-(A.16). Constraints (A.13)-(A.16) all deal with variables related to a same position i on the production line. Therefore, there is no edge in graph $\mathcal{G}$ between two nodes corresponding to different positions on the line. In addition, for a given position, the binary variables involved as well as the exclusion relations linking them are identical. $\mathcal{G}$ is thus seen to decompose into N independent subgraphs with identical structure: $\mathcal{G}^i = (\mathcal{V}^i, \mathcal{A}^i)$ with $\mathcal{V}^i$ the subset of nodes related to position i and $\mathcal{A}^i$ the subset of edges linking

these nodes. In the remainder of this subsection, we will study one of these graphs $\mathcal{G}^i$ for an arbitrary choice of i.

A set $C \subset \mathcal{V}^i$ is called a *clique* if each pair of nodes in C is connected by an edge. A *maximal clique* is a clique which is not properly contained in another clique. Each maximal clique in $\mathcal{G}^i$ thus gives rise to a valid inequality called a *maximal clique constraint* stating that the sum of corresponding binary variables should be less than or equal to 1. In the following, those are referred to as *type I valid inequalities*. We observe that constraints (A.2) and (A.4) of the initial formulation are among the clique constraints we can obtain thanks to the study of one of the subgraphs $\mathcal{G}^i$. But not all of them are *maximal* clique constraints. In the sequel, we show how to exploit the special structure of the graphs $\mathcal{G}^i$ to strengthen these constraints and find other maximal clique constraints, in particular constraints linking variables related to various distinct products.

The structure of a constraint graph $\mathcal{G}^i$ is close to that of a complete multipartite graph, i.e a graph with node set partitioned into clusters such that any two nodes belonging to different clusters have an edge connecting them and that there is no connection between nodes within a single cluster. Here, the set of nodes $\mathcal{V}^i$ can be divided into $M+1$ subsets $\mathcal{V}^i_m$ which will be referred to as clusters. The cluster $\mathcal{V}^i_m \subset \mathcal{V}^i$ is the subset of nodes in $\mathcal{G}^i$ related to metal m. There is an additional cluster, denoted $\mathcal{V}^i_0$, made of a single node, namely the node related to the variable z^i_0. Thanks to constraints (A.13), (A.14) and (A.15), there is an edge between any two nodes belonging to different clusters. But, because of the constraints (A.15) and (A.16), there are some additional edges between nodes belonging to the same cluster, namely in the cases where the corresponding variables are related to two layers of the same product or to two distinct types of cathodes.

The clusters $\mathcal{V}^i_m$ can be seen to have a special structure. Namely let $\pi(m) \subset \{1,..P\}$ be the set of indices of products p using metal m in at least one layer. The cluster $\mathcal{V}^i_m$ is made of $1+|\pi(m)|$ disjoint cliques (possibly containing a single node):
- K^0_m, the clique containing the nodes related to cathode types $c \in C(m)$,
- for each $p \in \pi(m)$, K^p_m, the clique containing the nodes related to the layers of product p made of metal m.

Proposition A.1 is a direct consequence of this particular graph structure.

Proposition A.1 *A maximal clique in $\mathcal{G}^i$ is the union of $M+1$ cliques, each clique belonging to a different cluster $\mathcal{V}^i_m$ of the graph.*

Figure A.1 illustrates the structure of a graph $\mathcal{G}^i$ for problem P0 introduced in section 2.2. We show a maximal clique containing 6 nodes and yielding the valid inequality: $z^i_0 + z^i_2 + y^i_{1,1} +$

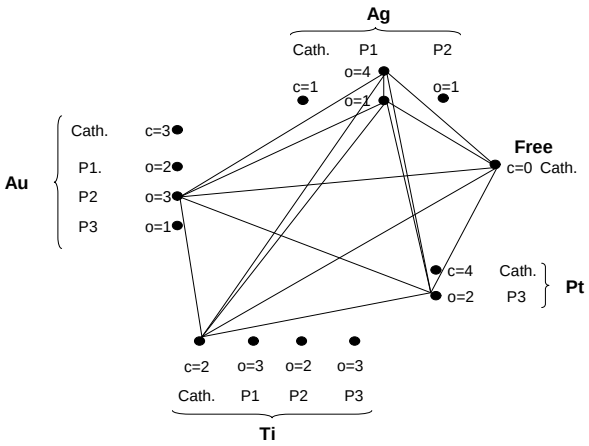

Figure A.1: Optimizing glass coating lines: constraint graph $\mathcal{G}^i$ for problem P0

$y^i_{1,4} + y^i_{2,3} + y^i_{3,2} \leq 1$. For the sake of simplicity, only the edges linking the nodes of this maximal clique are displayed.

Using proposition A.1, we can compute the number of maximal cliques in a graph $\mathcal{G}^i$ as the number of possible combinations obtained by choosing in each cluster $\mathcal{V}^i_m$ one clique out of $(1 + |\pi(m)|)$ cliques:

Proposition A.2 *The number of maximal cliques in a graph $\mathcal{G}^i$ is given by:*
$\prod_{m=1..M}(1 + |\pi(m)|)$.

This number can be quite high: e.g for the industrial problem P20 (see appendix), $\mathcal{G}^i$ has 2880 maximal cliques so that $2880 * 30 = 86400$ type I valid inequalities should be added to the formulation. Due to this large number, all maximal cliques constraints cannot be added *a priori* to the model. They can, however, be generated as needed according to a cutting-plane strategy. In order to do this, we need to address the so-called *separation problem*.

The separation problem here can be stated as follows: "given (z^*, y^*, x^*) the optimal solution of the linear relaxation of the problem, find a violated type I valid inequality or decide that (z^*, y^*, x^*) satisfies all type I valid inequalities ". To solve this problem, we use the following separation algorithm:

(SEP1) Given (z^*, y^*, x^*) the optimal solution of (A.1)-(A.10), for $i = 1...N$,

1. assign to each node in $\mathcal{G}^i$ a weight equal to the value of the corresponding variable,
2. for $m = 0...M$,

- compute the weight of each clique in the cluster $\mathcal{V}^i_m$: this weight is defined as the sum of the weight of all clique nodes.

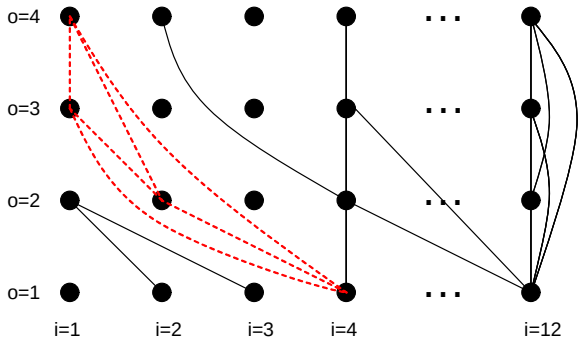

Figure A.2: Optimizing glass coating lines: constraint graph $\mathcal{G}_1$ for product $p = 1$ of problem P0

- select the clique K_m^{max} of maximal weight w_m^{max}.

3. compute $W = \sum_{m=0...M} w_m^{max}$.

- if $W > 1$, the valid inequality given by the maximal clique $C = \bigcup_{m=0...M}(K_m^{max})$ is violated,
- else all valid inequalities corresponding to position i are satisfied.

If, for each position $i = 1...N$, $W \leq 1$, then (z^*, y^*, x^*) satisfies all type I valid inequalities, otherwise at least one violated valid inequality has been found. In the sequel, algorithm (SEP1) is used to generate type I violated inequalities in order to strengthen the initial formulation.

A.3.3 Valid inequalities from precedence constraints between layers

We now focus on another subset of constraints in our problem: the precedence constraints between layers of a given product. As in subsection 3.2, we exploit the special structure of these binary exclusion constraints to derive a family of stronger valid inequalities corresponding to maximal clique constraints and to further strengthen the formulation.

We first explain how this family of stronger valid inequalities is derived. We have two families of binary exclusion constraints related to a single product p: constraints (A.7) of the original model stated in section 2 and valid inequalities (A.15) stated in the previous subsection. We define the corresponding constraint graph $\mathcal{G}_p = (\mathcal{V}_p, \mathcal{A}_p)$. A node $v \in \mathcal{V}_p$ refers to a binary variable y_{po}^i and can thus be indexed by (i, o). There is an edge $a \in \mathcal{A}_p$ between two nodes of $\mathcal{V}_p$ if there is a binary exclusion constraint (A.7) or (A.15) linking the corresponding variables. Figure A.2 shows the graph $\mathcal{G}_1$ obtained for product $p = 1$ of problem P0. Only a fraction of edges is presented, the edges drawn as dotted lines connect the nodes belonging to a maximal clique.

Maximal cliques in graphs $\mathcal{G}_p$ have special features that can be exploited as shown by the following result:

Proposition A.3 *A maximal clique in $\mathcal{G}_p$ consists of exactly O_p nodes, each node related to a different layer of product* p.

Proof A.1 *No two nodes in $\mathcal{V}_p$ related to the same layer are connected so that the cardinality of a clique in $\mathcal{G}_p$ cannot be greater than O_p, the number of layers of product* p.

In addition, a clique in $\mathcal{G}_p$ cannot be maximal if it does not include a node related to each layer $1...O_p$. Namely, suppose K *is a clique containing $O_p - 1$ nodes. All nodes relate to a different layer so that all layers $1, 2, ...O_p$ except layer* o *are present.* K *is thus the subset of nodes $K = \{(i_1, 1), (i_2, 2), ..., (i_{o-1}, o-1), (i_{o+1}, o+1), ..., (i_{O_p}, O_p)\}$ with $i_1 \geq i_2 \geq ... \geq i_{o-1} \geq i_{o+1} \geq ... \geq i_{O_p}$.*

We now show that K *cannot be a maximal clique. Consider a node (i_o, o) such that $i_{o-1} \geq i_o \geq i_{o+1}$. This node is connected to each node in* K. *We have namely:*
- $\forall \omega = 1...o-1, i_\omega \geq i_o$ and $\omega < o$. Hence there is a precedence constraint linking $y_{p\omega}^{i_\omega}$ and $y_{po}^{i_o}$ and (i_o, o) is connected to (i_ω, ω).
- similarly, $\forall \omega = o+1...O_p, i_\omega \leq i_o$ and $\omega > o$. Hence there is a precedence constraint linking $y_{p\omega}^{i_\omega}$ and $y_{po}^{i_o}$ and (i_o, o) is connected to (i_ω, ω).
So $K \bigcup \{(i_o, o)\}$ is a clique containing K*:* K *is not a maximal clique of $\mathcal{G}_p$.*

Each maximal clique in $\mathcal{G}_p$ provides a valid inequality for our problem. These valid inequalities will be referred to as *type II valid inequalities.*

We can compute the number of maximal cliques in a graph $\mathcal{G}_p$ by induction, as stated below:

Proposition A.4 *Let $\mathcal{G}_p(N, L)$ be the graph for a product* p *made of* L *layers and a production line with* N *positions. We denote by $\mu(N, L)$ the number of maximal cliques in $\mathcal{G}_p(N, L)$. We have:*
(i) $\forall N, \mu(N, 1) = N$
(ii) $\forall N, \forall L \geq 2, \mu(N, L) = \sum_{i=1}^{N} \mu(N - i + 1, L - 1)$
(The proof is left to the reader).

The number of maximal cliques in $\mathcal{G}_p$ grows very fast with the problem size, in particular with the number N of positions and the number L of layers. With the recurrence given above, the reader can easily check that e.g. for the product $p = 5$ in problem P20 ($N = 30$, $L = 8$), there are more than 38 billion type II valid inequalities. Hence it is not possible to include directly all type II valid inequalities in the formulation. This is why we propose two ways of using them to strengthen the formulation.

First, we remove constraints (A.7) from the formulation and replace them by a much smaller number of type II valid inequalities. This involves finding a subset of type II valid inequalities

such that every binary exclusion constraint of type (A.7) is implied by at least one valid inequality belonging to this subset, i.e. to find a subset of maximal cliques in $\mathcal{G}_p$ such that each edge $a \in \mathcal{A}_p$ is covered by at least one maximal clique belonging to this subset. The following heuristic procedure **(REP)** was devised in order to identify such a subset while keeping the number of clique constraints as small as possible. It is based on the idea that the edges of graph $\mathcal{G}_p$ can be covered in a systematic way by relying on the angle they make with the horizontal axis. More precisely, for each node (i, O_p) in $\mathcal{V}_p$, we generate the maximal cliques made up by the edges forming an angle α with the horizontal axis such that $tan(\alpha) = \frac{b}{a}$ where $a = 1...N - i$ and $b = 1...O_p - 1$.

(REP)

1. For $p = 1...P$, for $i = 1...N$, for $a = 1...N - i$, for $b = 1...O_p - 1$,
   generate the following type II valid inequality:

$$\sum_{k=1}^{\lceil \frac{O_p}{b} \rceil} \sum_{o=O_p-b(k-1)}^{O(O_p,k,b)} y_{p,o}^{I(i,k,a)} \leq 1$$

   with $I(i,k,a) = max(i + a(k-1); N)$ and $O(O_p, k, b) = min(O_p - bk + 1; 1)$.
2. Check wether each binary exclusion constraint of type (A.7) is covered by at least one generated type II valid inequality. If an uncovered binary exclusion constraint is found corresponding to layers o and $o' < o$ and positions i and $i' > i$, generate the following type II valid inequality:
   $\sum_{\omega=1}^{o'} y_{p,\omega}^{i'} + \sum_{\omega=o'+1}^{O_p} y_{p,\omega}^{i} \leq 1$
3. For each generated type II valid inequality, check wether all binary exclusion constraints it replaces are covered by more than one type II valid inequalities. If so, eliminate the corresponding type II valid inequality.

As shown by the computational experiments to be presented in section 4, the use of procedure (REP) results in a substantial reduction on the total number of constraints in the model as well as in an enhancement of the formulation.

Second, in order to further strengthen the formulation, we generate additional type II valid inequalities according to a cutting-plane strategy. This involves solving the following separation problem for type II valid inequalities: "given (z^*, y^*, x^*) the optimal solution of the linear relaxation of the problem, find a type II violated valid inequality or decide that (z^*, y^*, x^*) satisfies all type II valid inequalities". In order to solve it, we will make use of proposition A.5 below. We first build the oriented graph $\tilde{\mathcal{G}_p} = (\tilde{\mathcal{V}_p}, \tilde{\mathcal{A}_p})$ in which nodes in $\tilde{\mathcal{V}_p}$ correspond to binary variables

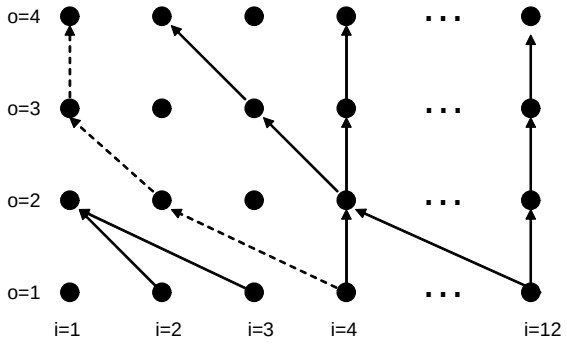

Figure A.3: Optimizing glass coating lines: oriented graph $\mathcal{G}_1$ for product $p = 1$ of problem P0

y^i_{po} and are indexed by (i, o). There is an oriented arc from node (i', o') to node (i, o) if $i \leq i'$ and $o = o' + 1$. We define a path as a sequence of nodes linked by arcs directed from a node to the following one. A maximal path is a path which is not contained in another path. Figure A.3 shows the graph $\tilde{\mathcal{G}}_1$ obtained for product 1 in problem P0. Only a fraction of all arcs is presented.

Proposition A.5 *There is an 1-1 correspondence between the maximal cliques of $\mathcal{G}_p$ and the maximal paths of the associated oriented acyclic graph $\tilde{\mathcal{G}}_p$.*

Proof A.2 *The graph $\tilde{\mathcal{G}}_p$ is an oriented acyclic graph. Due its special structure, a maximal path P in $\tilde{\mathcal{G}}_p$ contains O_p nodes, each one corresponding to a different layer. P is a subset of nodes in $\tilde{\mathcal{V}}_p$: $P = \{(i_1, 1)..., (i_o, o)..., (i_{O_p}, O_p)\}$ with $i_1 \geq ... \geq i_o \geq ... \geq i_{O_p}$. Thanks to proposition A.3, we know that the corresponding subset of nodes in $\mathcal{V}_p$ is a maximal clique of $\mathcal{G}_p$. Thus a maximal path of $\tilde{\mathcal{G}}_p$ corresponds to a maximal clique in $\mathcal{G}_p$. The converse is straightforward.*

Thanks to proposition A.5, solving the separation problem for type II valid inequalities reduces to the solution of a number of longest path problems in an acyclic graph, leading to the following separation algorithm: **(SEP2)** Given (z^*, y^*, x^*) the optimal solution of (A.1)-(A.10), for $p = 1...P$:

1. Assign to each node (i, o) in $\tilde{\mathcal{G}}_p$ a weight equal to the value of the corresponding variable y^i_{po}.
2. Find the maximal weight path $\tilde{P}^{max}$ in the acyclic oriented graph $\tilde{\mathcal{G}}^p$ with a standard longest path algorithm.
3. Let W^{max} be the weight of $\tilde{P}^{max}$.

- if $W^{max} > 1$, the valid inequality given by the maximal clique K^{max} corresponding to the path $\tilde{P}^{max}$ is violated,

- else all type II valid inequalities are satisfied for product p.

If $W^{max} \leq 1$ for all products $p = 1...P$, then all type II valid inequalities are satisfied, otherwise we have found at least one violated inequality. In the sequel, algorithm (SEP2) is used to generate type II violated inequalities in order to strengthen the initial formulation.

A.4 Computational results

In this section, we discuss the results of the computational experiments carried out to evaluate the impact of the formulation enhancements presented in section 3. We also present an empirical study carried out to show the influence of some problem parameters on the algorithmic performance.

A.4.1 Comparison of the initial and enhanced formulations

In order to evaluate the impact of the proposed formulation enhancements, we solved the problem with a standard MIP software (CPLEX 8.1.0) using either the initial formulation described in section 2 or the enhanced formulation. More precisely, the strengthened formulation is obtained thanks to the following procedure:

1. We use procedure (REP) to replace precedence constraints of type (A.7) by a subset of type II valid inequalities and we add the M valid inequalities (A.12) to the formulation. We solve the linear relaxation of the problem.
3. We use the separation algorithm (SEP1) to add type I violated valid inequalities.
4. When no more type I violated valid inequalities can be found, we look for type II violated valid inequalities using the separation algorithm (SEP2).
5. When no more type II violated valid inequalities can be found, we go back to step 3 and repeat until no more violated valid inequalities (whether of type I or type II) can be generated.

All the tests were run on a Pentium 4 (2.8 GHz) with 504 Mb of RAM, running under Windows XP. We used the default settings of CPLEX solver. This means that some cutting planes, among which are clique cuts, cover cuts and Gomory fractional cuts, are added automatically to the model (see [51] for more details).

We used an industrial data set available in [72] to build 20 test problems. P20 is the industrial problem presented in [72] and described in table A.7 (see Appendix). P0 is the simple example introduced in section 2, P1 to P19 are simpler versions of P20. These instances were obtained by using one or several of the following simplifications: removal of possible positions along the production line; removal of some products; removal of some layers; removal of a metal; removal of some cathode types. Table A.3 displays the following information for each problem tested: the

number N of possible positions along the line; the total number $\sum_p O_p$ of layers to be sputtered; the number C of cathode types; the number *Var* of binary variables and the number *Const* of constraints in the initial formulation.

Table A.3: Optimizing glass coating lines: test problems

	N	$\sum_p O_p$	C	*Var*	*Const*	Remarks
P0	12	10	4	180	1117	small example in section 2.2
P1	20	17	10	560	8826	P20 without products 4 and 5
P2	20	19	10	600	11378	P20 without products 2 and 3
P3	20	20	10	620	11799	P20 without products 1 and 5
P4	20	22	10	660	14351	P20 without products 1 and 3
P5	20	21	10	640	13170	P20 without products 1 and 2
P6	25	24	5	750	20333	P20 without product 5
P7	25	25	5	775	22484	P20 without product 4
P8	25	26	5	800	24335	P20 without product 3
P9	25	25	5	775	22484	P20 without product 2
P10	25	28	5	850	27137	P20 without product 1
P11	25	26	8	875	20108	P20 without metal Ag
P12	25	22	8	775	13004	P20 without metal Ti
P13	25	25	8	750	17957	P20 without metal Au
P14	25	28	8	925	22310	P20 without metal Pt
P15	25	25	8	850	17032	P20 without metal Steel
P16	25	15	5	525	5449	P20: at most 3 layers per product
P17	25	20	5	650	10204	P20: at most 4 layers per product
P18	25	24	5	750	15208	P20: at most 5 layers per product
P19	25	28	5	850	21437	P20: at most 6 layers per product
P20	30	32	26	1770	41772	see Appendix

The computational results obtained with the initial and enhanced formulations are displayed in table A.4. For both series of results, we provide:

- *Const*: the number of constraints in the formulation. For the enhanced formulation, this is the value obtained after applying the procedure (REP).

- Gap_0: the initial gap, i.e. the relative difference between the lower bound provided by the linear relaxation of the problem and the best integer solution found after at most 8 hours of computation. For the enhanced formulation, we use the value obtained after the strengthening procedure has stopped.

- *Nodes*: the number of nodes of the search tree explored before the optimal solution is found or the computation time limit of 8 hours is reached.

- CPU_{IP}: the time in seconds required to find the optimal integer solution when it has been found.

- *Gap*: the gap obtained after at most 8 hours of computation between the best integer solution found and the best lower bound found.

For the enhanced formulation, we also provide:
- *CutsI* and *CutsII*: the number of type I and type II cuts added to the formulation during the strengthening procedure,
- CPU_{str}: the time in seconds spent to strengthen the initial formulation, i.e. to carry out procedure REP and to generate violated type I and type II valid inequalities.

As can been seen from table A.4 (columns 2-6), using the initial formulation, only 7 out of the 21 problems can be solved exactly within the computational limits. Despite long computation times (8 hours), non-optimal integer solutions are found for 13 problems and in these cases, the remaining gaps obtained remain quite large (16% on average). In addition, no feasible integer solution can be found for problem P20.

We compare these results with the ones obtained while using the enhanced formulation to solve the problem. The results from table A.4 (columns 7-14) show that computation times for small instances are decreased and that more instances (11 out of 21 problems) are solved exactly. In addition, using the enhanced formulation, a feasible integer solution is found for all test problems and, in case the optimal integer solution could not be found after 8 hours of computation, the remaining gap is significantly smaller (9.6 % on average).

Comparison between the results obtained with the two formulations thus shows that the enhanced formulation improves the efficiency of the Branch & Bound procedure. The main explanatory factor for this is that the lower bounds provided by the linear relaxation of the enhanced formulation (table A.4 column 11) appear to be stronger than the ones provided by the linear relaxation of the initial formulation (table A.4 column 3). Indeed, the integrality gap (i.e the relative difference between Z_{LP} and Z_{IP}) is reduced on average from around 22% with the initial formulation to about 7.1% with the enhanced formulation. Moreover, it is worth pointing out here that the results provided in table A.4 strongly suggest that the automatic cutting-plane generation procedures embedded in the CPLEX software do not seem able to identify the type I and type II valid inequalities exhibited and discussed in section 3.

A.4.2 Influence of cathode capacity

We carried out some additional numerical tests to evaluate the influence of cathode capacity on the algorithmic performance. We considered problems P1 to P5 described in table A.3 and we modified the data relative to the cathodes. More precisely, we considered only one type of cathodes per metal and we built instances with various cathode capacity values:
- infinite capacity,
- large capacity: for each metal, the available cathode is the cathode with the largest volume

Table A.4: Optimizing glass coating lines: results with the initial and enhanced formulations

	Initial formulation				Enhanced formulation								
	$Const$	Gap_0	$Nodes$	CPU_{IP} (s)	Gap	$Const$	$CutsI$	$CutsIC$	CPU_{str} (s)	Gap_0	$Nodes$	CPU_{IP} (s)	Gap
P0	1117	35.2	351	1.8	0	777	12	4	0.3	14.5	46	1.2	0
P1	8826	23.9	10547	688	0	3100	129	604	51	0	933	393	0
P2	11378	30.3	143329	10501	0	3465	114	174	45	13.3	53131	8334	0
P3	11799	31.6	163415	13706	0	3634	116	461	109	7.1	42141	10027	0
P4	14351	30.8	240214	#	14.5	3999	157	292	100	6.7	112190	25344	0
P5	13170	21.5	133718	17456	0	3846	131	59	54	0	3942	1500	0
P6	20333	25.6	119944	#	22.2	6342	187	1110	701	5.9	39543	#	5.9
P7	22484	22.9	89052	#	21.5	6660	226	700	795	5.9	38352	#	5.9
P8	24335	23.3	92031	#	21.2	6884	184	632	488	16.6	30136	#	16.6
P9	22484	22.2	98344	#	19.4	6660	189	74	188	5.3	26011	#	5.3
P10	27137	26.7	73411	#	25.6	7454	214	264	580	10.5	24531	#	9.6
P11	20108	13.8	105293	#	13.8	6817	227	299	192	0	12355	5604	0
P12	13004	11.1	266188	#	5.88	5700	193	364	88	5.5	50360	#	5.5
P13	17957	11.2	123220	#	11.2	6526	246	469	213	5.5	27850	#	5.5
P14	22310	11.1	82000	#	11.1	7340	268	250	342	5.5	39710	#	5.5
P15	17032	16.5	51626	12104	0	6566	190	403	215	0	20915	9208	0
P16	5449	21.9	5316	279	0	3739	13	23	2.4	10.0	3537	259	0
P17	10204	21.8	247543	#	7.1	5184	1	109	6.2	0	393	82	0
P18	15208	19.8	143136	#	19.6	6260	91	426	78	0	16880	7916	0
P19	21437	15.2	87485	#	15.2	7489	177	183	906	11.1	56047	#	11.1
P20	41772	_	51762	_	_	11754	279	70	2663	25.0	3254	#	25.0

The symbol " #" indicates that a guaranteed optimal solution could not be found within 8 hours of computation.
The symbol " _" indicates that no feasible solution could be found within 8 hours of computation.

Table A.5: Optimizing glass coating lines: influence of cathodes capacity

Cathode capacity	Gap_0	*Opt*	*Gap*	CPU_{IP} (s)	*#Nodes*
infinite	11.4	5	0	2617	8781
large	5.4	5	0	10930	30983
medium	11.9	1	8.8	> 28800	105480
small	2.6	1	2.6	> 28800	39088

among those described in table A.7,

- medium capacity: for each metal, the available cathode is the cathode with the second largest volume among those described in table A.7,

- small capacity: for each metal, the available cathode is the cathode with the third largest volume among those described in table A.7.

We used the enhanced formulation to solve these instances. Table A.5 displays the computational results. We provide Gap_0, *Gap*, CPU_{IP} and *Nodes* as defined in subsection A.4.1 and *Opt* the number of instances that could be solved to optimality within the computation limit. These results suggest that instances with medium or small capacity cathodes are more difficult to solve than instances with infinite or large capacity cathodes. Namely, all instances using infinite or large capacity could be solved to optimality within 2 hours of computation whereas only 2 out of the 10 instances using medium or small capacity cathodes could be solved to optimality within 8 hours of computation. Moreover no feasible solution could be found for 2 out of the 5 instances using small capacity cathodes.

A.4.3 Influence of product composition

We finally discuss the results of some experiments carried out to evaluate the influence of product composition, i.e. of the sequence of metal layers to be deposited on the glass sheets. We built 15 instances involving $M = 5$ metals, $P = 5$ products made of 6 layers, $N = 20$ positions on the line, $C = 5$ infinite capacity cathodes. They differ only with respect to the sequence of metal layers:

- In E1 to E5, there is a basic sequence of metal defined by product 1. Products 2 to 5 are obtained by a simple modification of this sequence (switch between two consecutive layers or modification of the metal for one layer).

- In R1 to R5, the sequences of metal layers are randomly generated from a discrete uniform $DU(1, 5)$ distribution. If two consecutive layers are made of the same metal, we repeat the random generation until a product is obtained without any identical consecutive layers.

- In H1 to H5, the sequence of layers for each product are chosen in order to obtain supposedly

difficult instances (products made of reverse sequences of metal, products made of sequences with no common metal...).

In order to compare the generated instances, we introduce a measure aiming at evaluating the difference between the products of a given instance with respect to the sequence of metal layers. This difference denoted d is defined as: $d = \sum_{p_1=1}^{P} \sum_{p_2=p_1+1}^{P} d(p_1, p_2)$ where $d(p_1, p_2) = SCS(p_1, p_2) - LCS(p_1, p_2)$. $SCS(p_1, p_2)$ is defined as the minimum number of cathodes needed to sputter products p_1 and p_2 and $LCS(p_1, p_2)$ is the maximum number of cathodes that can be used to sputter layers from both p_1 and p_2. $SCS(p_1, p_2)$ and $LCS(p_1, p_2)$ can be computed by a dynamic programming algorithm as respectively the Shortest Common Supersequence containing p_1 and p_2 and the Longest Common Subsequence contained in p_1 and p_2.

We used the enhanced formulation to solve these instances. The computational results are displayed in table A.6. These results suggest that instances with a large value of d are more difficult to solve than instances with a small value of d. Namely, all instances E1-E5 could be solved to optimality within one hour of computation whereas the mean computation time for the instances R1-R5 and H1-H5 is above 4.5 hours. Moreover no feasible solution could be found for 2 out of the 5 instances H1-H5. It is worth pointing out that for the instance P20 presented in Appendix $d = 5.8$. This seems to indicate that in a industrial situation, the products to be made on the glass coating line are quite different with respect to the sequence of metal layers to be deposited, leading to an additional difficulty to solve the problem.

Table A.6: Optimizing glass coating lines: influence of product composition

Instances	d	Gap_0	$\#Opt$	Gap	CPU_{IP} (s)	$\#Nodes$
E1-E5	2.8	22.4	5	0	2184	7405
R1-R5	5.5	13.1	4	6.5	14642	32325
H1-H5	6.4	11.1	3	0	19075	29792

A.5 Conclusion and perspectives

In this paper, we studied an optimization problem arising in the context of the glass industry in connection with a specific transformation of flat glass called glass coating. In order to improve an initial MIP formulation, three families of valid inequalities have been discussed: valid inequalities from limited capacity constraints; valid inequalities from metal compatibility constraints (type I valid inequalities); valid inequalities from precedence constraints between layers of a given product (type II valid inequalities). The results of our computational experiments confirm the positive impact of the proposed enhancements on the computation times and solution quality.

Among the possible research directions suggested by the present work, it might be worth exploring other optimization criteria such as minimizing the volume of unused metal remaining in the cathodes at the end of the production run. Indeed, partially consumed cathodes at the end of a production run represent a cost, either as a direct loss because of the unused metal or as additional constraints for the forthcoming production run because they will impose the use of a set of initial reduced capacity cathodes. Looking for other families of valid inequalities in order to further improve the formulation might also be an interesting research direction.

Appendix

Table A.7: Optimizing glass coating lines: data for problem P20

$p=1$	o	1	2	3	4				
	m_{po}	Ag	Au	Ti	Ag				
	V_{po}	2200	2400	2000	2000				
$p=2$	o	1	2	3	4	5	6	7	
	m_{po}	Ag	Au	Ti	St	Au	Ti	Ag	
	V_{po}	2100	1300	1000	1000	700	2000	4000	
$p=3$	o	1	2	3	4	5	6		
	m_{po}	Au	Pt	Ti	St	Au	Pt		
	V_{po}	2000	1000	1000	2400	1000	2000		
$p=4$	o	1	2	3	4	5	6	7	
	m_{po}	St	Ti	St	Au	Pt	Ti	St	
	V_{po}	1500	1000	2400	1000	750	500	1000	
$p=5$	o	1	2	3	4	5	6	7	8
	m_{po}	Au	Pt	Ti	St	Ag	St	Ti	St
	V_{po}	1000	750	500	1000	2000	1500	1000	2400

c	1	2	3	4	5	6	7	8	9	10
ν_c	10	10	10	10	10	10	10	10	10	10
m_c	Ag	Ag	Ag	Ag	Ag	Ag	Ti	Ti	Ti	Ti
V_c	300	500	1000	2000	3000	4000	4000	3000	2500	1000
c	11	12	13	14	15	16	17	18	19	20
ν_c	10	10	10	10	10	10	10	10	10	10
m_c	Ti	Au	Au	Au	Au	Au	St	St	St	St
V_c	400	100	500	1000	2500	3500	500	750	1000	1500
c	21	22	23	24	25	26				
ν_c	10	10	10	10	10	10				
m_c	St	Pt	Pt	Pt	Pt	Pt				
V_c	2000	500	750	1000	1500	2000				

Bibliography

[1] A. Arnaud. Industrial production of coated glass: future trends for expanding needs. *Journal of Non-crystalline Solids*, 218:12–18, 1997.

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

[3] C. Becker and A. Scholl. A survey on problems and methods in generalized assembly line balancing. *European Journal of Operational Research*, 168:694–715, 2006.

[4] G. Belvaux and L. Wolsey. *bc-prod*: A specialized branch-and-cut system for lot-sizing problems. *Management Science*, 45(5):724–738, 2000.

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

[6] R. Beretta, P.M. Franca, and V.A. Armantano. Metaheuristic approaches for the multilevel resource-constrained lot-sizing problem with setup and lead times. *Asia-Pacific Journal of Operational Research*, 22(2):261–286, 2005.

[7] R. Beretta and L.F. Rodrigues. A memetic algorithm for a multistage capacitated lot-sizing problem. *International Journal of Production Economics*, 87:67–81, 2004.

[8] PB. Billington, J. Blackburn, J. Maes, R. Millen, and L.N. Van Wassenhove. Multi-item lotsizing in capacitated multi-stage serial systems. *IIE Transactions*, 26(2):12–18, 1994.

[9] G. Bitran and S. Gilbert. Sequencing production on parallel machines with two magnitudes of sequence-dependant setup cost. *Journal of Manufacturing and Operations Management*, 3:24–52, 1990.

[10] J. Blackburn and R. Millen. Improved heuristics for multi-stage requirements planning systems. *Management Science*, 28(1):44–56, 1982.

[11] J.M.. Blondeau. *Etude et mise en place d'un outil d'aide à la planification de production.* Master thesis, Ecole Centrale Paris, Paris, France, 2007.

[12] N. Boysen and M. Fliedner. A versatile algorithm for assembly line balancing. *European Journal of Operational Research*, 184:39–56, 2008.

[13] N. Boysen, M. Fliedner, and A. Scholl. A classification of assembly line balancing problems. *European Journal of Operational Research*, 183:674–693, 2007.

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

[15] W. Brüggemann and H. Jahnke. The discrete lot-sizing and schedulting problem: Complexity and modification for batch availability. *European Journal of Operational Research*, 124:511–529, 2000.

[16] J. Bukchin and M. Tzur. Design of flexible assembly line to minimize equipment cost. *IIE Transactions*, 32:585–598, 2000.

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

[18] A.R. Clark and V.A. Armentano. A heuristic for a resource-oriented multi-stage lot-sizing problem with lead times. *Journal of the Operational Research Society*, 46:1208–1222, 1995.

[19] 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.

[20] M. Constantino. A cutting plane approach to capacitated lot-sizing with start-up costs. *Mathematical Programming*, 75:353–376, 1996.

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

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

[23] R. de Matta and M. Guignard. Studying the effects of production loss due to setup in dynamic production scheduling. *European Journal of Operational Research*, 72:62–73, 1994.

[24] R. de Matta and T. Miller. Production and inter-facility transportation scheduling for a process industry. *European Journal of Operational Research*, 158:72–88, 2004.

[25] C. Dhaenens-Flipo and G. Finke. An integrated model for an industrial production-distribution problem. *IIE Transactions*, 33:705–715, 2001.

[26] M. Diaby, H.C. Bahl, M.H Karwan, and S. Zionts. Capacitated lot-sizing and scheduling by lagrangean relaxation. *European Journal of Operational Research*, 59:444–458, 1992.

[27] P.S. Dixon and E.A. Silver. A heuristic solution procedure for the multi-item, single-level, limited capacity, lot-sizing problem. *Journal of Operations Management*, 2(1):23–39, 1981.

[28] G. Dobson. The cyclic lot scheduling problem with sequence-dependent setups. *Operations Research*, 40(4):736–749, 1992.

[29] E. dos Santos-Meza, M. dos Santos, and M. N. Arenales. A lot-sizing problem in an automated foundry. *European Journal of Operational Research*, 139:490–500, 2002.

[30] A. Drexl and A. Kimms. Lot sizing and scheduling - survey and extensions. *European Journal of Operational Research*, 99:221–235, 1997.

[31] A. Dumoulin and C. Vercellis. Tactical models for hierarchical capacitated lot-sizing problems with set-ups and changeovers. *International Journal of Production Research*, 38(6):51–67, 2000.

[32] S.D. Eksioglu, B. Eksioglu, and H.E. Romeijn. A lagrangean heuristic for integrated production and transportation planning problems in a dynamic, multi-item, two-layer supply chain. *IIE Transactions*, 39:191–201, 2007.

[33] S. Elmaghraby. The economic lot scheduling problem (ELSP): review and extensions. *Management Science*, 24(6):587–598, 1978.

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

[35] B. Fleischmann. The discrete lot sizing and scheduling problem. *European Journal of Operational Research*, 44:337–348, 1990.

[36] B. Fleischmann. The discrete lot sizing and scheduling problem with sequence-dependent set-up costs. *European Journal of Operational Research*, 75:395–404, 1994.

[37] B. Fleischmann and H. Meyr. The general lotsizing and scheduling problem. *OR Spektrum*, 19:11–21, 1997.

[38] B. Fleischmann, H. Meyr, and M. Wagner. *Advanced planning*, chapter 4, pages 71–95. Supply chain management and advanced planning. Springer, Berlin, second edition, 2002.

[39] F. Fumero and C. Vercellis. Synchronized development of production, inventory, and distribution schedules. *Transportation Science*, 33(3):330–340, 1999.

[40] C. Gicquel, N. Miègeville, M. Minoux, and Y. Dallery. Discrete lot-sizing and scheduling using product decomposition into attributes. Under review for *Computers & Operations Research*, 2008.

[41] C. Gicquel, N. Miègeville, M. Minoux, and Y. Dallery. Optimizing glass coating lines: MIP model and valid inequalities. Under review for *European Journal of Operational Research*, 2008.

[42] C. Gicquel, M. Minoux, and Y. Dallery. Capacitated lot-sizing models: a review. Working paper available on-line: http://hal.archives-ouvertes.fr/hal-00255830/fr/, 2008.

[43] C. Gicquel, M. Minoux, and Y. Dallery. A tight MIP formulation for the discrete lot-sizing and scheduling problem with sequence-dependent changeover costs and times. Submitted to *Operations Research Letters*, 2008.

[44] M. Goycoolea, A.T. Murray, F. Barohona, R. Epstein, and A. Weintraub. Harvest scheduling subject to maximum area restrictions: exploring exact approaches. *Operations Research*, 53(3):490–500, 2005.

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

[46] K. Haase. Capacitated lot-sizing with sequence dependent setup costs. *OR Spektrum*, 18:51–59, 1996.

[47] 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*, 66:159–169, 2000.

[48] F.W. Harris. How many parts to make at once. *Factory, The Magazine of Management*, 10(2):135–136, 1913.

[49] T.P. Harrison and H.S. Lewis. Lot sizing in serial assembly systems with multiple constrained resources. *Management Science*, 42(1):19–36, 1996.

[50] Y.F. Hung and K.L. Chien. A multi-class multi-level capacitated lot sizing model. *Journal of the Operational Research Society*, 51:1309–1318, 2000.

[51] ILOG. *ILOG CPLEX 8.0 : User's manual.* ILOG, Gentilly, France, 2002.

[52] R. Jans. Solving lotsizing problems on parallel identical machines using symmetry breaking constraints, 2006. Accepted for publication in *INFORMS Journal on Computing.*

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

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

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

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

[57] J. Kalvenes, J. Kennington, and E. Olinick. Hierarchical cellular network design with channel allocation. *European Journal of Operational Research*, 160:3–18, 2005.

[58] 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.

[59] B. Karimi, S.M.T. Fatemi Ghomi, and J.M. Wilson. The capacitated lot sizing problem: a review of models and algorithms. *Omega*, 31:365–378, 2003.

[60] U. Karmarkar and L. Schrage. The deterministic dynamic product cycling problem. *Operations Research*, 33(2):326–345, 1985.

[61] E. Katok, H.S. Lewis, and T.P. Harrison. Lot sizing in general assembly systems with setup costs, setup times, and multiple constrained resources. *Management Science*, 44(6):859–877, 1998.

[62] A. Kimms. A genetic algorithm for multi-level, multi-machine lot sizing and scheduling. *Computers & Operations Research*, 26:829–848, 1999.

[63] O. Kirka and M. Kökten. A new heuristic approach for the multi-item dynamic lot sizing problem. *European Journal of Operational Research*, 75:332–341, 1994.

[64] R. Kuik, M. Salomon, and L. N. van Wassenhove. Batching decisions: structure and models. *European Journal of Operational Research*, 75:243–263, 1994.

[65] J. Maes, J.O. McClain, and L.N. van Wassenhove. Multilevel capacitated lotsizing complexity and LP-based heuristics. *European Journal of Operational Research*, 53:131–148, 1991.

[66] T.L. Magnanti and T. Sastry. Facets and reformulations for solving production planning with changeover costs. *Operations Research*, 50(4):708–719, 2002.

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

[68] D. Maier. The complexity of some problems on subsequences and supersequences. *Journal of the Association for Computing Machinery*, 25(2):322–336, 1978.

[69] 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*, 150:177–192, 2007.

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

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

[72] N. Miegeville. *Supply Chain Optimization in the process industry. Methods and Case Study of the Glass Industry.* PhD thesis, Ecole Centrale Paris, Paris, France, 2005.

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

[74] G.L. Nemhauser and L.A. Wolsey. *Integer and Combinatorial Optimization.* John Wiley and Sons, USA, 1988.

[75] Y.B. Park. An integrated approach for production and distribution planning in supply chain management. *International Journal of Production Research*, 73(6):1205–1224, 2005.

[76] A. Pinnoi and W.E. Wilhelm. Assembly system design: a branch and cut approach. *Management Science*, 44(1):103–118, 1998.

[77] Y. Pochet and L.A. Wolsey. A strong cutting plane algorithm for production scheduling with changeover costs. *Management Science*, 37(1):53–67, 1991.

[78] Y. Pochet and L.A. Wolsey. *Production Planning by Mixed Integer Programming*. Springer Science, New York, USA, 2006.

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

[80] M. Salomon, M. Solomon, L. van Wassenhove, Y. Dumas, and S. Dauzère-Pérès. 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*, 100:494–513, 1997.

[81] M. Sambavisan and S. Yahya. A lagrangean-based heuristic for multi-plant, multi-item, multi-period capacitated lot-sizing problems with inter-plant transfers. *Computers & Operations Research*, 32:537–555, 2005.

[82] C. Silva and J.M. Magalhaes. Heuristic lot size scheduling on unrelated parallel machines with applications in the textile industry. *Computers & Industrial Engineering*, 50:76–89, 2006.

[83] E.A. Silver and H.C. Mear. A heuristic for selecting lot size quantities for the case of a deterministic time varying demand rate and discrete opportunities for replenishment. *Production & Inventory Management*, 14(2):64–74, 1973.

[84] V.L. Smith-Daniels and D.E. Smith-Daniels. A mixed integer programming model for lot sizing and sequencing packaging lines in the process industries. *IIE Transactions*, 18:278–285, 1986.

[85] H. Stadtler. Multilevel lot sizing with setup times and multiple constrained resources: internally rolling schedules with lot-sizing windows. *Operations Research*, 51(3):487–502, 2003.

[86] H. Stadtler. Supply chain management and advanced planning-basics, overview and challenges. *European Journal of Operational Research*, 163:4575–588, 2005.

[87] K. Suzuki. State of the art in large area vacuum coatings on glass. *Thin Solid Films*, 351:8–14, 1999.

[88] H. Tempelmeier and M. Derstroff. A lagrangean-based heuristic for dynamic multilevel multiitem constrained lotsizing with setup times. *Management Science*, 42(5):738–757, 1996.

[89] H. Tempelmeier and S. Helber. A heuristic for dynamic multi-item multi-level capacitated lotsizing for general product structures. *European Journal of Operational Research*, 75:296–311, 1994.

[90] J.M. Thizy and L.N. van Wassenhove. Lagrangean relaxation for the multi-item capacitated lot-sizing problem: a heuristic implementation. *IIE Transactions*, 17(4):308–313, 1985.

[91] C.H. Timpe and J. Kallrath. Optimal planning in large multi-site production networks. *European Journal of Operational Research*, 126, 2000.

[92] F.M.B. Toledo and V.A. Armentano. A lagrangian-based heuristic for the capacitated lot-sizing problem in parallel machines. *European Journal of Operational Research*, 175:1070–1083, 2006.

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

[94] C.A. Van Eijl and C.P.M. van Hoesel. On the discrete lot-sizing and scheduling problem with Wagner-Whitin costs. *Operations Research Letters*, 20:7–13, 1997.

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

[96] H.M. Wagner and T.M. Whitin. Dynamic version of the economic lot size model. *Management Science*, 5(1):89–96, 1958.

[97] L. Wolsey. Progress with single item lot-sizing. *European Journal of Operational Research*, 86:395–401, 1995.

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

[99] J. Xie and J. Dong. Heuristic genetic algorithms for general capacitated lot-sizing problems. *Computer & Mathematics with applications*, 44:263–276, 2002.

[100] L. Özdamar and G. Barbarosoglu. Hybrid heuristics for the multi-stage capacitated lot sizing and loading problem. *Journal of the Operational Research Society*, 50:810–825, 1999.

[101] L. Özdamar and G. Barbarosoglu. An integrated lagrangean relaxation-simulated annealing approach to the multi-level mult-item capacitated lot sizing problem. *International Journal of Production Economics*, 68:319–331, 2000.

[102] L. Özdamar and S.I. Birbil. Hybrid heuristics for the capacitated lot sizing and loading problem with setup times and overtime decisions. *European Journal of Operational Research*, 110:525–547, 1998.

[103] L. Özdamar and T. Yazgac. A hierarchical planning approach for a production-distribution system. *International Journal of Production Research*, 37(16):3759–3772, 1999.

[104] F.M. Zeghal and M. Minoux. Modeling and solving a crew assignment problem in air transportation. *European Journal of Operational Research*, 175:187–209, 2006.

[105] P.H. Zipkin. Computing optimal lot sizes in the economic lot scheduling problem. *Operations Research*, 39(1):56–63, 1991.

Résumé

Le dimensionnement des lots de production est une des nombreuses activités survenant dans le cadre de la planification de production. Il a pour objet de déterminer quand et combien produire de façon à réaliser le meilleur compromis possible entre la minimisation des coûts liés à la production (coûts fixes de reconfiguration de la ressource, coûts de stockage...) et la satisfaction de la demande des clients
Nous nous intéressons ici un problème de planification de production par lots connu sous le nom de "Discrete Lot-sizing and Scheduling Problem" ou "DLSP". Plus précisément, nous étudions plusieurs variantes de ce problème dans lesquelles les coûts et/ou les temps de changement de produits sur la ressource sont dépendant de la séquence et nous proposons diverses extensions d'une méthode disponible dans la littérature pour la résolution exacte du problème mono-niveau, mono-ressource.
Nos contributions portent à la fois sur la modélisation du problème et sur l'implémentation de méthodes efficaces de résolution. En ce qui concerne la modélisation, nous étudions l'intégration de divers aspects opérationnels dans le modèle de base afin d'en améliorer la pertinence industrielle. Ainsi nous considérons les extensions suivantes : la prise en compte d'une structure de produits "multi-attributs" qui permet de diminuer la taille du problème d'optimisation à résoudre, l'intégration de temps de changement positifs afin de mieux modéliser la perte de production causée par une reconfiguration de la ressource et la présence de plusieurs ressources parallèles dont la production doit être planifiée simultanément. En ce qui concerne la résolution du problème, nous présentons pour chacune des extensions du modèle de base une approche de résolution visant à fournir des solutions optimales exactes. En général, les résultats de nos expériences numériques montrent l'utilité pratique de ces algorithmes pour la résolution d'instances de moyenne et grande taille en des temps de calcul compatibles avec une application industrielle.

Mots clés : Planification de production par lots, coûts de reconfiguration dépendant de la séquence, programmation linéaire en nombres entiers, inégalités valides

Abstract

Lot-sizing is one of the many issues arising in the context of production planning. Its main objective is to determine the timing and level of production so as to reach the best possible trade-off between minimizing setup and inventory holding costs and satisfying customer demand. When a limited production capacity and a deterministic time-varying demand rate are assumed, lot-sizing leads to the formulation of large-sized mixed-integer programs, most of which are hard to solve.
In the present work, we deal with one of the many capacitated dynamic lot-sizing models, the Discrete Lot-sizing and Scheduling Problem or DLSP, and study several variants of this problem where changeover costs and/or times are sequence-dependent. We propose various extensions of an existing exact solution approach for the single-level, single-resource DLSP with sequence-dependent changeover costs.
Our contributions concern both problem modelling and efficient implementation of solution algorithms. In terms of problem modelling, we investigate the integration of various additional relevant industrial concerns into the basic model. More precisely, we consider the following operational aspects: the presence of a multi-attribute product structure which can be exploited to reduce the size of the optimization problem, the integration of positive changeover times to better model the production loss caused by a changeover and the presence of identical parallel resources that need to be planned simultaneously. In terms of algorithmic developments, we present for each of these extensions a solution procedure aiming at providing exact optimal solutions: a tight MIP formulation for the corresponding problem variant is derived and the resulting mixed-integer program is solved thanks to a commercial MIP solver. Moreover, results of extensive computational experiments carried out to evaluate the proposed solution approaches are provided. In general, they show the practical usefulness of the proposed algorithms at solving medium to large-sized instances with a reasonable computational effort.

Key words: Production planning, Lot-sizing, Sequence-dependent setup costs, Mixed-integer programming, Valid inequalities