Simultaneous lotsizing and scheduling – extensions and solution approaches

genehmigte Dissertation

zur Erlangung des akademischen Grades

eines Doctor oeconomiae (Dr. oec.)

der Universität Hohenheim

Fakultät Wirtschafts- und Sozialwissenschaften

Institut für Interorganizational Management & Performance

Fachgebiet Betriebswirtschaftslehre,

insbesondere Supply Chain Management

vorgelegt von:

Dipl.-Wirtsch.-Ing. Martin Wörbelauer

Stuttgart 2018

Betreuer:	Prof. Dr. Herbert Meyr
Zweitgutachterin:	Prof. Dr. Katja Schimmelpfeng
Dekan:	Prof. Dr. Karsten Hadwich
Tag der Einreichung:	10.08.2017
Tag der Disputation:	06.12.2017

Acknowledgments

The present PhD thesis was written during my time as a research assistant at the department of Supply Chain Management at the University of Hohenheim in Stuttgart. Many people have supported, motivated and accompanied me throughout this process. I am very grateful to each of them.

First of all, I would like to communicate my deep thanks to Prof. Dr. Herbert Meyr for being my PhD advisor. You always made time to discuss arising problems or to talk about the progress and the next steps of my work. These conversations were always very helpful due to your broad knowledge base, your experience and your analytical capability. Moreover, you always maintained a very pleasant and open communication atmosphere. Thank you very much for this and for your continuous support.

I am also very grateful to Prof. Dr. Katja Schimmelpfeng for acting as co-referee of my thesis and to Prof. Dr. Robert Jung for being the chairman of my PhD committee.

Special thanks go to my co-authors Dr. Karina Copil, Prof. Dr. Horst Tempelmeier and Prof. Dr. Bernardo Almada-Lobo for the friendly and productive collaboration.

I would like to express my gratitude to my current and former colleagues at the department of Supply Chain Management. I am especially grateful to Dr. Stephanie Eppler, Mirko Kiel, Stephan Fichtner and Dr. Jaime Cano-Belman for their support, for the fruitful discussions about research projects, for the pleasant non-scientific conversations and for the great atmosphere within our department.

There are many friends outside of the university from whom I also received support, sometimes by having discussions about research topics, but mostly by helping me to find some occasional distance from my work. I am sincerely grateful to all of you. Especially, I would like to thank Christine Rau for backing me and for doing a lot of proofreading in the final phase of my PhD thesis. Finally, I am very grateful to my family for always motivating me and for giving me confidence. Thank you very much for being there.

Martin Wörbelauer

Contents

1 Introduction

The present thesis examines *simultaneous lotsizing and scheduling* models. These models are used to determine cost-minimal production plans to fulfill a dynamic demand of several products (see, e.g., Drexl and Kimms 1997). The subsequent section describes the detailed planning problem of these models and explains in general why planning is worthwhile for a company and in which cases simultaneous lotsizing and scheduling models are appropriate. Additionally, the main motivations of this thesis will be described. These main motivations are to overcome shortcomings concerning the mapping of practical applications by simultaneous lotsizing and scheduling models and to overcome shortcomings of the solution methods applied to solve such models. Section 1.2 formulates the research goals of the thesis and explains in which way these goals will be reached. Finally, Section 1.3 describes the detailed organization of the thesis.

1.1 Motivation

Planning is very important for every organization wanting to generate profits by selling physical products or services. For example, in the case of a manufacturer who disregards planning, it might happen that necessary production materials are missing because they have not been ordered early enough. As another example, consider a manufacturer having committed a customer order but being unable to meet the desired quantities on time since not enough production capacities have been reserved. In detail, planning is the process of decision preparation. I.e., alternatives must be identified and an appropriate or even optimal option should be chosen afterwards. (see Fleischmann et al. 2015, p. 71)

Lotsizing is a common planning problem of a manufacturer. A lotsize is defined as the number of products produced on a machine at once. A basic lotsizing model is the *economic order quantity model* of Harris (1913) which considers a single product having a constant demand rate and being produced on a single machine. The goal of the model is finding a lotsize which minimizes the total costs. Normally, a setup operation is necessary to prepare the machine and enable the start of the production of a lot. Each setup operation causes setup costs. Therefore, large lotsizes would be preferable to avoid a high number of setups. In contrast, large lotsizes will lead to high holding costs for storing the products until they are used to fulfill the demand. I.e., the total costs, which should be minimized, consist of the sum of setup and holding costs.

Often, the above described model does not map all characteristics which may arise in a lotsizing scenario of a practical application. One category of models counteracting this shortcoming is named *simultaneous lotsizing and scheduling models*. Such models consider a limited number of production stages. Each stage may consist of several heterogeneous parallel (production) lines[1]. A limited horizon is planned and divided into several periods, each having a fixed production capacity. A deterministic given demand for multiple products may differ from period to period and must be fulfilled without backlogging. Like in the economic order quantity model, holding costs and setup costs are considered. Moreover, the setup costs might be sequence-dependent, i.e., they might be different depending on the previously produced product. Therefore, the sequence of the different lots must be planned as well. Most of the models incorporate sequence-dependent setup times which define the consumption of production capacities to prepare a line for the production of a lot. Since there is a strong interdependency between the sequence of the products and the remaining production capacity, lotsizing and scheduling must be planned simultaneously. Quite often, the models incorporate line-dependent and product-dependent production costs and production speeds. Additionally, some models respect minimum lotsizes. Typically, simultaneous lotsizing and scheduling models are formulated as *mixed integer programs* consisting of an objective function and multiple constraints. (see, e.g., Meyr 1999, Chapter 3)

The following paragraphs classify simultaneous lotsizing and scheduling in a framework of planning tasks arising in an organization. However, an organization cannot be considered as a stand-alone entity. Typically, it is connected to other organizations forming a *supply chain*. A supply chain defines a

> "*...network of organizations that are involved, through upstream and downstream linkages, in the different processes and activities that produce value in the form of products and services in the hands of the ultimate consumer*" (Christopher 2005, p. 17).

For example, a supply chain may consist of several suppliers which deliver pre-products to a manufacturer who then produces final products. Afterwards, a third-party logistics provider transports the products to the final customers. If the supply chain members are legally separated, this constellation is called *inter-organizational* supply chain. In a large company, the different supply chain members may merely be different departments of the same company. In this case, it is named *intra-organizational* supply chain. (see Stadtler 2015, pp. 3-4)

The task of organizing and planning all processes in a supply chain is called *supply chain planning*. Figure 1.1 provides a general overview of the planning tasks arising in a supply chain. (see Fleischmann et al. 2015, pp. 76-82)

As one can see in Figure 1.1, planning tasks of a supply chain are structured by the main supply chain processes *procurement*, *production*, *distribution* and *sales*. Additionally, the planning tasks are classified by the length of the considered planning horizon. A typical *long-term* planning horizon comprises several years. Thus, planning on an aggregated level is essential in order to get a manageable problem size. For example, it is a common approach to define a time grid based on years or months instead of weeks or days. Due to the long-term impact of decisions of this planning level, they are called *strategic decisions*. An example of a long-term planning task is the definition of plant locations.

[1]A line may consist of one or several machines. In the case of several machines, the sequence of the machines must be identical for all products (flow line system). (see, e.g., Günther and Tempelmeier 2016, pp. 13-14)

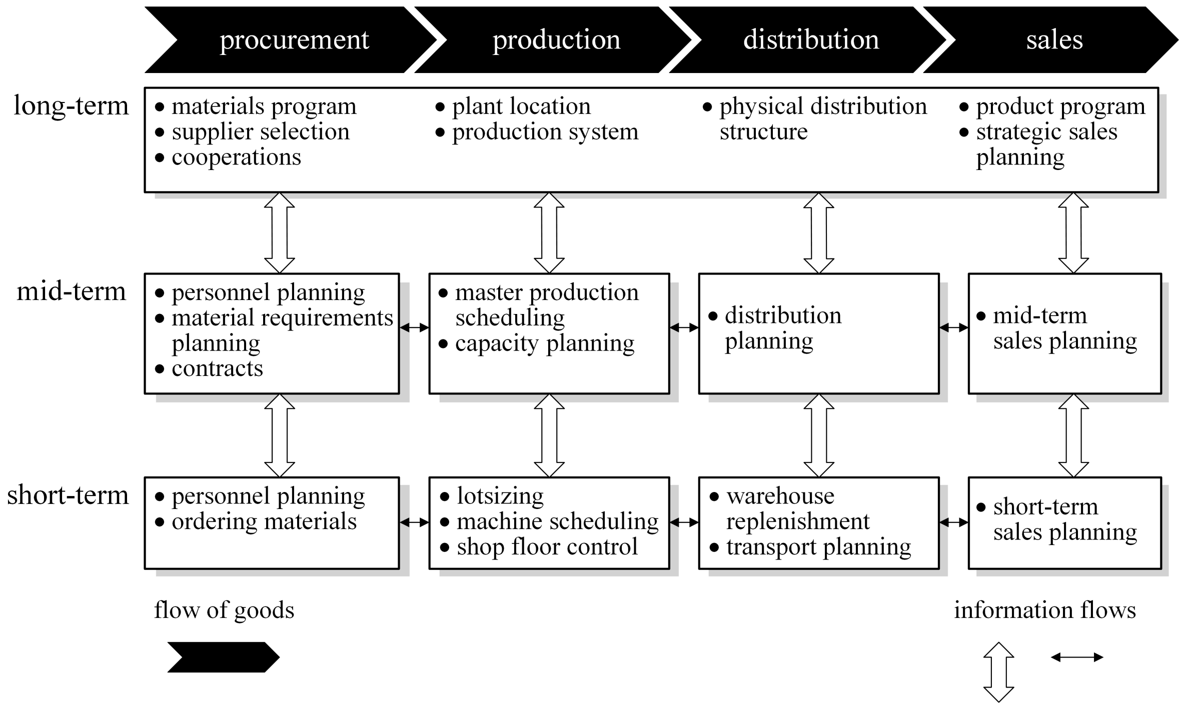


Figure 1.1: Supply chain planning matrix (see Rohde et al. 2000 or for the presented English version Fleischmann et al. 2015, p. 77)

Mid-term planning is more detailed than long-term planning and comprises a planning horizon of between half a year and two years. Often, this level of planning is named *tactical planning*. One main intention of mid-term planning is to define a rough plan of the material flows. This is very useful, e.g., in the case of long purchasing lead times of pre-products making it necessary to initiate orders several months in advance. Sometimes, simultaneous lotsizing and scheduling is applied to define aggregated production plans for product groups on the mid-term planning level (see Meyr and Mann 2013, p. 729).

Finally, the planning horizon of *short-term* planning ranges from a few days up to three months. It is the most detailed planning level and comprises, e.g., the short-term transportation planning. Moreover, simultaneous lotsizing and scheduling is classified in the section of short-term production planning. Short-term planning is also called *operational planning*.

As one can see in the supply chain planning matrix, horizontal information flows occur between the planning tasks of the different supply chain processes and vertical information flows occur between the different planning levels. Software, named *Advanced Planning Systems*, is used to support the supply chain planning. One aspect of Advanced Planning Systems is to find an optimal or at least very good plan instead of solely finding a feasible plan. Often, the quality of a plan is defined by its costs arising during execution. This is also the case for simultaneous lotsizing and scheduling (see, e.g., Meyr 1999, Chapter 3).

Typically, Advanced Planning Systems use the concept of *hierarchical production planning* (see Hax and Meal 1975). Hierarchical production planning supports the decision making, especially in the case of high interdependencies between the different planning decisions. On the one hand, hierarchical production planning is necessary since successive planning cannot incorporate all dependencies between the different planning decisions. On the other hand, it is impossible to simultaneously perform all planning tasks in one model due to too long computation times (additional reasons can, e.g., be found in Meal 1984). Like in a successive planning, hierarchical planning decomposes the overall planning task into several modules. Decisions on a higher planning level, i.e., concerning a longer planning horizon, define the framework of decisions on lower planning levels which concern shorter planning horizons. Moreover, hierarchical planning systems regularly comprise feedback systems from lower planning levels to higher planning levels (for different organizations of hierarchical planning systems see Schneeweiß 2003). All in all, one can see that the concept of hierarchical planning is directly incorporated in the supply chain planning matrix. (see Fleischmann et al. 2015, Chapter 4, which also provides a deeper explanation of the supply chain planning matrix)

After explaining the relevance of planning and classifying simultaneous lotsizing and scheduling using the supply chain planning matrix in the paragraphs before, the motivation to the content of this thesis will be described in more detail. Since the last review especially focusing on simultaneous lotsizing and scheduling has been published many years ago (Drexl and Kimms 1997), it seems reasonable to do a comprehensive literature research and analyze the results. Using a detailed classification scheme known from Meyr (1999), a structured comparison of the different model formulations should be enabled. Additionally, the classification scheme helps to describe the development of this research field and to identify current research gaps and trends. One result of this review is that some models do not only consider production lines as limited *(primary) resources*, but also consider so-called *secondary resources*. However, this topic is not treated to the full extend so far, as will be explained in the following.

One example of a secondary resource is a setup operator (see, e.g., Tempelmeier and Buschkühl 2008) who is responsible for the setups on multiple lines. Typically, a setup operator can only set up one line at the same time. Thus, if the setup operator is neglected during planning, it could happen that the resulting plan schedules two setups at the same time and therefore cannot be implemented. Raw material defines another secondary resource which is necessary for production (see, e.g., Göthe-Lundgren et al. 2002). If the raw material is neglected during planning, it might happen that a plan cannot be realized since the material is missing. Obviously, the incorporation of secondary resources is very important. However, the existing models are very specialized and in most cases only consider one kind of secondary resources (e.g., setup operators or raw materials). Hence, it will be very difficult for a company to find an existing model which fits exactly their production scenario. Thus, there is a need for a general model which provides a wide applicability.

Another aspect is the performance of solution methods used to solve simultaneous lotsizing and scheduling models. A company will only use such models if production plans can be built in a reasonable amount of time. Additionally, a created plan must have a high solution quality, i.e., the resulting total costs (sum of holding, setup and production costs) must be low. However, practical applications often comprise several lines and many products have to be planned simultaneously due to high interdependencies. Current solution approaches still show high computation times for large-scaled problems (see, e.g., Meyr and Mann 2013). Thus, a further aspect of research in this thesis is to develop a solution method which performs better.

In the following section, the detailed research goals of this thesis and the basic methodologies to reach these goals will be described.

1.2 Goals and methodology

As already addressed in Section 1.1, different types of secondary resources exist and most models focus only on one type of secondary resources. Therefore, the goal is to formulate a general model which is capable of mapping all types of secondary resources. Moreover, it should be possible to “deactivate” unnecessary variables and constraints to decrease the problem complexity. The literature review mentioned in Section 1.1 identifies all simultaneous lotsizing and scheduling models which consider secondary resources. However, a more detailed analysis of the considered secondary resources is necessary to identify which types of secondary resources exist and how they can be differentiated. Additionally, it is worthwhile to think about further types or characteristics of secondary resources which should be included in the model. The final step is to choose a basic simultaneous lotsizing and scheduling model and extend it in a way that it becomes capable of mapping all types of secondary resources.

Especially in the case of large-scaled problems which simultaneously consider multiple lines, current solution methods still have optimization potential (see Section 1.1). Thus, the goal is to develop a solution approach which performs significantly better. This will enable a wider applicability in practical applications. One promising approach performing better compared to former heuristics is presented in Meyr and Mann (2013). The idea is to decompose the multi-line problem into independent single-line problems. Since the resulting problems are less complex and well-performing heuristics for single-line problems exist, they can be solved in very short time. Afterwards, the solution of the original problem can be formed. Meyr and Mann (2013) decompose the multi-line problem as follows: the time grid of the original model is aggregated, i.e., the number of periods is reduced. Afterwards, the aggregated problem is solved and the resulting schedule is used to define line-dependent demands for all products. Since this heuristic performs well but raises long computation times for problems with many products, it seems worthwhile to try the following adaption: instead of doing a time aggregation, products are aggregated to so-called setup families. The approach of considering a problem on different aggregation levels and using the solution of a higher aggregation level as framework for the problem description on a lower level is also the basis of hierarchical production planning. In the solution heuristic of this thesis, a so-called *formal constructional hierarchy* (see Schneeweiß 2003, Chapter 3) is used. I.e., the hierarchy is only introduced to support the mathematical solving process and symmetric information exists (for more details see Schneeweiß 2003).

1.3 Outline of the thesis

The detailed agenda of the present thesis is organized as follows. Chapter 2 provides a structured literature review which has already been published in the journal OR Spectrum (Copil et al. 2017). After an introduction, a generic model formulation of the general lotsizing and scheduling problem (GLSP) is explained. This model, firstly presented by Meyr (1999, Chapter 4), generalizes other models for simultaneous lotsizing and scheduling. These other models are the discrete lotsizing and scheduling problem (DLSP) by Fleischmann (1990), the continuous setup lotsizing problem (CSLP) by Karmarkar and Schrage (1985), the proportional lotsizing and scheduling problem (PLSP) by Drexl and Haase (1995) and the capacitated lotsizing problem with sequence-dependent setups (CLSD) by Haase (1996). Additionally, a classification scheme (see Meyr 1999) to analyze simultaneous lotsizing and scheduling models is explained. Section 2.3 is separated into six subsections. Each subsection is devoted to one of the basic model formulations GLSP, CLSD, PLSP, CSLP and DLSP. The last subsection comprises model formulations which cannot be explicitly assigned to one of the former basic models. In each subsection, the main features of the identified models of the literature are described. An overview table analyzing all described models using the classification scheme known from Section 2.2 closes each subsection. Additionally, a short discussion of each table is added. Section 2.4 analyzes and summarizes the results of the literature review. It is structured into four subsections, each focusing on a special aspect. The first subsection discusses the occurrence of different model characteristics (e.g., whether or not setup times are considered). The next subsection focuses on the occurrence of further model extensions like secondary resources or perishability of products. Subsection 2.4.3 discusses for which practical applications the models have been developed. Finally, the last subsection identifies trends in the developed solution approaches. Section 2.5 summarizes the results of the literature review and provides an outlook on future research topics.

Chapter 3 comprises a working paper titled “Simultaneous lotsizing and scheduling considering secondary resources”. This paper has been written by Martin Wörbelauer, Herbert Meyr[2] and Bernardo Almada-Lobo[3]. The paper has been submitted to OR Spectrum and the review is still in process. A short introduction describes the importance of incorporating secondary resources. Additionally, it points out the necessity of a general formulation which assures a wide applicability. The following section is devoted to a literature review specialized in simultaneous lotsizing and scheduling models considering secondary resources. Compared to the literature review of Chapter 2 it explains in detail which secondary resources are considered and how they are modeled. Therefore, it is structured by three identified types of secondary resources. Finally, a newly developed classification scheme is presented and used to classify the different models in a more detailed way. The result is used to formulate the requirements of a general model considering secondary resources. Section 3.3 presents the basic model formulation which is an adaption of the general lotsizing and scheduling model for parallel production lines (GLSPPL) of Meyr (2002) and Meyr and Mann (2013). The following section introduces

[2]Department of Supply Chain Management, University of Hohenheim, Stuttgart, Germany.

[3]INESC-TEC, Faculdade de Engenharia, Universidade do Porto, Portugal.

all model extensions necessary to represent all types of secondary resources. Section 3.5 adds additional features to the model. For example, the setup operation is divided into a dismounting, cleaning and mounting operation, allowing to represent the resource use in a more detailed way. Numerical examples demonstrate the wide applicability and show the functionalities of the general model (Section 3.6). Finally, Section 3.7 provides a summary and an outlook.

Chapter 4 proposes a new solution heuristic for large-scaled simultaneous lotsizing and scheduling problems. Section 4.1 provides an introduction explaining the basic concept of the heuristic. Additionally, it briefly motivates why the GLSPPL is chosen as the model to be solved. The subsequent section comprises a short literature review which is focused on models using setup families and focused on GLSP formulations and associated solution approaches. Section 4.3 describes the model formulation of the GLSPPL. Afterwards, Section 4.4 presents the solution approach in detail. It is structured into several subsections. The first subsection describes the detailed framework of the heuristic. Subsequently, it is described how parameters (e.g., setup times) of setup families can be determined. Subsection 4.4.3 explains the way of assigning products to setup families. For this purpose, two new algorithms are presented. The first one defines setup families mainly based on setup characteristics, the second one mainly assigns products to setup families randomly. Subsection 4.4.4 describes how the solution of the aggregated problem can be disaggregated to form independent single-line problems. Finally, Section 4.4.5 presents an iterative process which should be applied if an iteration of the heuristic does not find a solution which completely fulfills the given demand. The results of numerical tests are presented in Section 4.5. Finally, Section 4.6 summarizes the results and provides a short outlook on further research fields.

Chapter 5 is divided into two subsections. The first one summarizes the content of the thesis, the second one provides an outlook on further research topics.

Bibliography

Christopher, M., 2005. Logistics and supply chain management, creating value-adding networks, 3rd Edition. Financial Times Prentice Hall, Harlow.

Copil, K., Wörbelauer, M., Meyr, H., Tempelmeier, H., 2017. Simultaneous lotsizing and scheduling problems: a classification and review of models. OR Spectrum 39 (1), 1–64.

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

Drexl, A., Kimms, A., 1997. Lot sizing and scheduling – survey and extensions. European Journal of Operational Research 99 (2), 221–235.

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

Fleischmann, B., Meyr, H., Wagner, M., 2015. Advanced planning. In: Stadtler, H., Kilger, C., Meyr, H. (Eds.), Supply chain management and advanced planning: concepts, models, software and case studies, 5th Edition. Springer Berlin, Ch. 4, pp. 71–95.

Göthe-Lundgren, M., Lundgren, J. T., Persson, J. A., 2002. An optimization model for refinery production scheduling. International Journal of Production Economics 78 (3), 255–270.

Günther, H.-O., Tempelmeier, H., 2016. Produktion und Logistik, 12th Edition. Books on Demand, Norderstedt.

Haase, K., 1996. Capacitated lot-sizing with sequence dependent setup costs. OR Spectrum 18 (1), 51–59.

Harris, F. W., 1913. How many parts to make at once. Factory, The Magazine of Management 10 (2), 135–136, 152.

Hax, A. C., Meal, H. C., 1975. Hierarchical integration of production planning and scheduling. In: Geisler, M. (Ed.), Logistics. TIMS Studies in the Management Sciences, Vol. 1. Elsevier Amsterdam, pp. 53–69.

Karmarkar, U. S., Schrage, L., 1985. The deterministic dynamic product cycling problem. Operations Research 33 (2), 326–345.

Meal, H. C., 1984. Putting production decisions where they belong. Harvard Business Review 2 (Mar.–Apr.), 102–111.

Meyr, H., 1999. Simultane Losgrößen- und Reihenfolgeplanung für kontinuierliche Produktionslinien. Deutscher Universitätsverlag, Wiesbaden.

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

Meyr, H., Mann, M., 2013. A decomposition approach for the general lotsizing and scheduling problem for parallel production lines. European Journal of Operational Research 229 (3), 718–731.

Rohde, J., Meyr, H., Wagner, M., 2000. Die Supply Chain Planning Matrix. PPS Management 5 (1), 10–15.

Schneeweiß, C., 2003. Distributed Decision Making, 2nd Edition. Springer, Berlin, Heidelberg.

Stadtler, H., 2015. Supply chain management — an overview. In: Stadtler, H., Kilger, C., Meyr, H. (Eds.), Supply chain management and advanced planning: concepts, models, software and case studies, 5th Edition. Springer Berlin, Ch. 1, pp. 3–28.

Tempelmeier, H., Buschkühl, L., 2008. Dynamic multi-machine lotsizing and sequencing with simultaneous scheduling of a common setup resource. International Journal of Production Economics 113 (1), 401–412.

2 Simultaneous lotsizing and scheduling problems: a classification and review of models

Abstract The current paper[4] presents a structured overview over the literature on dynamic simultaneous lotsizing and scheduling problems. We introduce a classification scheme, review the historical development of research in this area and identify recent developments.

The main contribution of the present review is the discussion of the historical development of the body of knowledge in the field of simultaneous lotsizing and scheduling and the identification of recent trends. This helps to reveal research opportunities, but it can also be helpful in the selection of appropriate models for industrial applications.

Keywords Dynamic lotsizing, Scheduling, Review

2.1 Introduction

In recent years, there has been an increasing interest in simultaneous lotsizing and scheduling problems, not only in academia, but also in many companies. Many publications are motivated by observations made in the process industry. Here, the production system often comprises a limited number of stages. Each of these stages may consist of several parallel production lines with finite capacities. Demands for individual products are usually associated with a time period (due date) and vary over time as a result of a forecasting procedure or due to known customer order arrivals. Holding costs occur for inventory in stock at the end of each period. A changeover from one product to another causes setup costs as well as setup times which are often sequence-dependent. Besides this lotsizing problem, there exists a scheduling problem, which comprises the sequencing of the products. This problem is of special importance if setup costs or setup times are sequence-dependent as the sequence influences the total costs and capacity consumption. Thus, instead of a successive planning, the lotsizing and scheduling problems should be solved simultaneously.

While a large number of reviews on production planning under consideration of setups have been published in the last years, many of these reviews focus on the lotsizing part of the problem and do not consider scheduling aspects at all (Buschkühl et al. 2010), or they contain only brief sections devoted to the simultaneous treatment of lotsizing and scheduling (Zhu and Wilhelm 2006; Jans and Degraeve 2008; Quadt and Kuhn 2008). Solely Drexl and Kimms (1997) focus almost entirely on simultaneous

[4]This paper has already been published in OR Spectrum (see Copil et al. 2017).

lotsizing and scheduling, thereby considering dynamic, time-varying demands. However, more than 17 years have passed since then.

In the current review, we present a structured overview over the latest literature on simultaneous lotsizing and scheduling. We confine our discussion to dynamic lotsizing and scheduling models in discrete time with period-specific demands for individual items. Thereby, it is assumed that the objective is to fulfill the actual orders or rather forecasted demands as late as possible (just in time) under consideration of holding and setup costs and with respect to finite capacities. Producing as late as possible reduces the average amount of stock on hand. A standard argument used in the definition of a lotsizing model is that low stock on hand induces low capital costs. Hence, holding costs should be included in the objective function. Günther (2014) has pointed out, that in a short-time planning horizon, holding costs as out-of-pocket capital costs are usually quite small. As a consequence, he proposes a block planning approach to produce as early as possible using the makespan criterion which is often used in scheduling. In this approach, first the production events are scheduled, followed by the determination of the production quantities (lotsizes). However, although it may be worthwhile to tackle the lotsizing problem in this way, we observed in many practical cases that the amount of stock on hand as the result of a lotsizing decision is a key performance indicator which is monitored by the logistic management as well as by financial analysts. Therefore, we focus on standard lotsizing models which try to find the optimal trade-off between holding and setup costs. Holding costs may then serve as a parameter to select the products which should be produced in advance in case of scarce capacity.

Considering the number of more than 160 publications included in our review, it is not warranted to describe each model and solution approach in detail. We rather introduce a classification scheme and discuss the development of the different models over time in terms of the incorporation of additional constraints, the application of solution techniques and their potential application in industry. To the best of our knowledge, there is no up-to-date review which focuses on these aspects. Thus, it is worthwhile to examine the progress of research that has been achieved particularly in the last two decades.

With respect to the maximum number of setups per period, models based on "microperiods" with at most one setup per period and models based on "macroperiods" with any number of setups can be distinguished. For both model classes several basic model formulations have been proposed. Models based on microperiods are the discrete lotsizing and scheduling problem (DLSP) by Fleischmann (1990), the continuous setup lotsizing problem (CSLP) by Karmarkar and Schrage (1985) and the proportional lotsizing and scheduling problem (PLSP) by Drexl and Haase (1995). Macroperiods are used in the formulations of the general lotsizing and scheduling problem (GLSP) by Fleischmann and Meyr (1997) and of the capacitated lotsizing problem with sequence-dependent setups (CLSD) by Haase (1996). All these models support lotsizing as well as scheduling decisions with the objective to minimize the sum of setup and holding costs under consideration of due dates and finite capacities. In the following, we use these five basic models to structure our review. We suggest a classification scheme which, on the one hand, allows a strong differentiation between the models and, on the other hand, allows to draw conclusions concerning practical applications.

For the sake of brevity, we do not repeat the model formulations of the DLSP, CSLP, PLSP and the CLSD. Instead, we present a generic formulation of the GLSP in Sect. 2.2 which allows us to derive the other models just by variation of the input data. Section 2.3 is divided into five subsections in which extensions of the basic models are classified and discussed, respectively. Models which do not directly fit into this structure are described in an extra Sect. 2.3.6. An overview table is appended to each subsection helping to derive current trends with respect to extensions of the basic models, practical applications and solution approaches. A detailed analysis is given in Sect. 2.4. Section 2.5 gives a brief summary of the results of our analysis and identifies opportunities for future research.

2.2 Classification scheme for simultaneous lotsizing and scheduling models

In the following, we describe the generic version of the GLSP. Furthermore, we use this model to discuss the other models (DLSP, CSLP, PLSP and CLSD). In the second part of this chapter, we explain our classification approach in detail.

2.2.1 Generic model

As mentioned above, one can differentiate between models based on microperiods (small-bucket models) and models which use macroperiods (large-bucket models). The basic formulations of the DLSP, CSLP and PLSP are small-bucket models for simultaneous lotsizing and scheduling. The capacitated lotsizing problem (CLSP) is an early large-bucket model which determines the lotsizes but not the sequence of the lots. This feature is introduced by the GLSP and the CLSD. Since these models combine macroperiods and an approach to sequence the lots, they are also called hybrid models (see Suerie 2005). To some extent, the GLSP generalizes the other models because each of these basic models can be represented as a special case of the GLSP for a single capacitated production resource. This property will be used to illustrate the main differences between the models. For this purpose, a “generic” version of the GLSP, in the following denoted as gGLSP, will be presented, which has been proposed by Meyr (1999, Chap. 4).

This model considers several physical products k $(k = 1,2,\dots,K)$ plus a fictitious dummy product $k = 0$ which is used to indicate a neutral setup state of the production resource. For each physical product $k > 0$ and each macroperiod t $(t = 1,2,\dots,T)$ a demand d_{kt} has to be fulfilled without backlogging. The production time per unit of product k is given by the production coefficient a_k. Changeovers between physical products $i > 0$ and $k > 0$ cause sequence-dependent setup costs sc_{ik}. A shutdown of the production resource is modeled using the neutral state and causes shutdown costs sc_{i0}. If no production takes place but the resource is set up for a physical product, standby costs pc_k occur for preserving the setup state for product $k > 0$ on the production resource. Standby costs pc_0 for staying in the neutral state are typically zero. A startup from the neutral state causes startup costs sc_{0k}. The aim is to schedule the products on the production resource in a way that the total costs are minimized. The costs comprise sequence-dependent setup costs, standby costs and holding costs hc_k, $k > 0$, on the inventory at the end of each macroperiod.

Table 2.1: Symbols used in model gGLSP

Indices and sets:	
i, k	Product index, $i, k = 0, 1, \dots, K$, whereat 0 is the neutral state
s	Index of microperiods, $s = 1, 2, \dots, S$
t	Index of macroperiods, $t = 1, 2, \dots, T$
$\mathcal{S}_t$	Set of microperiods s within macroperiod t
Data:	
sc_{ik}	Setup costs for a changeover from product i to product k
hc_k	Holding costs for product $k > 0$ (per unit and per macroperiod)
pc_k	Standby costs for preserving the setup state of product k on the production resource (per time unit)
a_k	Production time per unit of product k ($a_0 = 1$)
st_{ik}	Setup time for a changeover from product i to product k
C_t	Capacity of the production resource in macroperiod t (time)
I_{k0}	Initial inventory of product $k > 0$ at the beginning of planning (units)
d_{kt}	Demand of product k in macroperiod t (units)
ω_{k0}	$\omega_{k0} = 1$ indicates that the production resource is set up for product k at the beginning of planning ($\omega_{k0} = 0$, otherwise)
q_k^{min}	Minimal production quantity of product $k > 0$ (units); minimal time for neutral state $k = 0$
Variables:	
$q_{ks} \geq 0$	Production quantity of physical product $k > 0$ (units) in microperiod s; time spent in neutral state if $k = 0$, respectively
$\overline{q}_{ks} \geq 0$	Duration (time) for which the setup state of product k is preserved on the production resource in microperiod s ($\overline{q}_{0s} = 0$ w.l.o.g.)
$I_{kt} \geq 0$	Inventory (units) of product $k > 0$ at the end of macroperiod t
$\omega_{ks} \in \{0, 1\}$	Setup state variable; $\omega_{ks} = 1$ indicates that the production resource is set up for product k in microperiod s (0 otherwise)
$z_{iks} \in \{0, 1\}$	Changeover variable; $z_{iks} = 1$ indicates a changeover from product i to product k in microperiod s (0 otherwise)

Microperiods s ($s = 1, 2, \dots, S$) are used to model the sequence of the products within the macroperiods. In a microperiod a single physical product is produced or the setup state for a physical product is conserved without production or the resource is in the neutral state. Each macroperiod t consists of a predefined sequence of microperiods. $\mathcal{S}_t$ denotes the set of microperiods s within macroperiod t and

$|\mathcal{S}_t|$ is the total number of microperiods within macroperiod t. The length of each macroperiod is given by the capacity C_t of the production resource. While the capacities are input to the model, the lengths of the microperiods are decision variables. They result from multiplying the production quantities q_{ks} by the production coefficients a_k plus setup times or from the time $\overline{q}_{ks}$ in which a setup state is preserved. The setup state is defined by a binary variable ω_{ks} which is 1 if the production resource is set up for product k in microperiod s and 0, otherwise. A changeover from setup state i to setup state k in microperiod s is indicated by the decision variable $z_{iks} \in \{0,1\}$. A setup causes a sequence-dependent setup time st_{ik}. Finally, the variables $I_{kt} \geq 0$ denote the inventory of product k at the end of macroperiod t. All parameters and variables used in the model are summarized in Tab. 2.1. The model formulation of the gGLSP is stated below.

gGLSP:

Objective function:

$$\text{Min} \sum_{s=1}^{S}\sum_{i=0}^{K}\sum_{k=0}^{K} sc_{ik} \cdot z_{iks} + \sum_{k=1}^{K}\sum_{t=1}^{T} hc_k \cdot I_{kt} + \sum_{k=0}^{K}\sum_{s=1}^{S} pc_k \cdot \overline{q}_{ks} \tag{2.1}$$

Subject to:

$$\sum_{k=0}^{K}\sum_{s\in\mathcal{S}_t}(a_k \cdot q_{ks} + \overline{q}_{ks}) + \sum_{i=0}^{K}\sum_{k=0}^{K}\sum_{s\in\mathcal{S}_t} st_{ik} \cdot z_{iks} = C_t \qquad \forall t \tag{2.2}$$

$$I_{kt} = I_{k,t-1} + \sum_{s\in\mathcal{S}_t} q_{ks} - d_{kt} \qquad \forall t, k > 0 \tag{2.3}$$

$$\sum_{k=0}^{K} \omega_{ks} = 1 \qquad \forall s \tag{2.4}$$

$$a_k \cdot q_{ks} + \overline{q}_{ks} \leq C_t \cdot \omega_{ks} \qquad \forall k, t, s \in S_t \tag{2.5}$$

$$z_{iks} \geq \omega_{i,s-1} + \omega_{ks} - 1 \qquad \forall i, k, s \tag{2.6}$$

$$q_{ks} \geq q_k^{min}(\omega_{ks} - \omega_{k,s-1}) \qquad \forall k, s \tag{2.7}$$

The objective function (2.1) describes the total costs consisting of setup costs, holding costs and costs for preserving setup states. Equations (2.2) guarantee that the production does not exceed the capacity in any macroperiod. More precisely, the total time used for production, for preserving setup states of the production resource and for changeovers is equal to the given capacity per macroperiod. Equations (2.3) assure that the inventory of the physical product $k > 0$ at the end of macroperiod t is equal to the inventory at the end of the previous macroperiod plus the production quantity of period t minus the demand of that period. Equations (2.4) ensure that in each microperiod the production resource is set up for exactly one product k $(k = 0, 1, \ldots, K)$. The linking-constraints (2.5) guarantee that if one of the continuous variables q_{ks} or $\overline{q}_{ks}$, $k > 0$, is greater than zero, the binary variable ω_{ks} is set to one. Otherwise, the resource is in the neutral state ($k = 0$).

Changeovers are indicated by Equations (2.6). It should be noted that the binary changeover variables z_{iks} can be relaxed. In an optimal solution, these decision variables only take the values zero or one. This is caused by the objective function (a small value for z_{iks} is desired) and Equations (2.6) in combination with ω_{ks} (defined as binary variable). Equations (2.7) are used to realize minimum lotsizes q_k^{min}, which may be necessary due to technical limitations of the production process. Minimum lotsizes are also important if the triangle inequalities ($sc_{ik} + sc_{kj} \geq sc_{ij}$) are violated. That means it is cheaper (or consumes less setup time) to switch from product i to product k and then to product j than to switch directly from product i to j. For example, the triangle inequality is violated if product k has a cleansing function. Note that it may be optimal to set up a certain product k more than once per macroperiod if the triangle inequalities are violated.

Fleischmann and Meyr (1997) actually distinguish between the GLSP with loss of setup state (GLSPLS) and the GLSP with conservation of setup state (GLSPCS). In the *GLSPLS* the setup state is not conserved during “idle” periods, i.e., periods in which no production of physical products takes place. For example, if there has been production of product $k > 0$ in microperiod $s-2$, but no production at all ($\sum_{k>0} q_{k,s-1} = 0$) in microperiod $s-1$, then a setup is necessary although the same product k is produced in microperiod s again. This is different in the *GLSPCS*. Here, no additional setup is necessary after such an idle period if the same product is produced again. The GLSPLS can be represented as a special case of the gGLSP by setting the standby costs to infinity ($pc_k = \infty, k > 0$). In this case, the resource changes into the neutral state $k = 0$ if it is not completely utilized. On the other hand, the gGLSP can be specialized to the GLSPCS if the standby costs are set to zero ($pc_k = 0$) and the neutral state is forbidden, e.g., by setting the setup costs for changeovers into the neutral state to infinity ($sc_{k0} = \infty$). Deeper insights into GLSP-based models will be given in Sect. 2.3.1.

In the following, we will show that all the abovementioned basic model formulations can also be represented as specializations of the gGLSP (see Fig. 2.1). Since the other basic models do not consider minimal production quantities and setup times, we define $q_k^{min} = 0$ and $st_{ik} = 0$.

The basic formulation of the *CLSD* allows the conservation of the setup state, but limits the number of lots per macroperiod to K. Hence, in the gGLSP, besides requesting $pc_k = 0$ and $sc_{k0} = \infty$, the number of microperiods per macroperiod is equal to the number of products ($|\mathcal{S}_t| = K$). Furthermore, in the CLSD each product can be set up at most once per macroperiod. Inequalities (2.8) enforce this behavior for the gGLSP (if the triangle inequalities are violated). Section 2.3.2 will present extensions of the basic CLSD which mitigate these additional constraints.

$$\sum_{s \in \mathcal{S}_t} \sum_{\substack{i=0 \\ i \neq k}}^{K} z_{iks} \leq 1 \qquad \forall t, k > 0 \tag{2.8}$$

In contrast to the CLSD, the basic *PLSP* allows at most one single changeover per period while the setup state can be conserved. In total, at most two different products can be produced in a single period. This can be modeled for the gGLSP by setting $|\mathcal{S}_t| = 2$ and restricting the total number of changeovers per macroperiod according to inequalities (2.9). In Sect. 2.3.3 we will discuss various extensions.

$$\sum_{s \in \mathcal{S}_t} \sum_{k=0}^{K} \sum_{\substack{i=0 \\ i \neq k}}^{K} z_{iks} \leq 1 \qquad \forall t \tag{2.9}$$

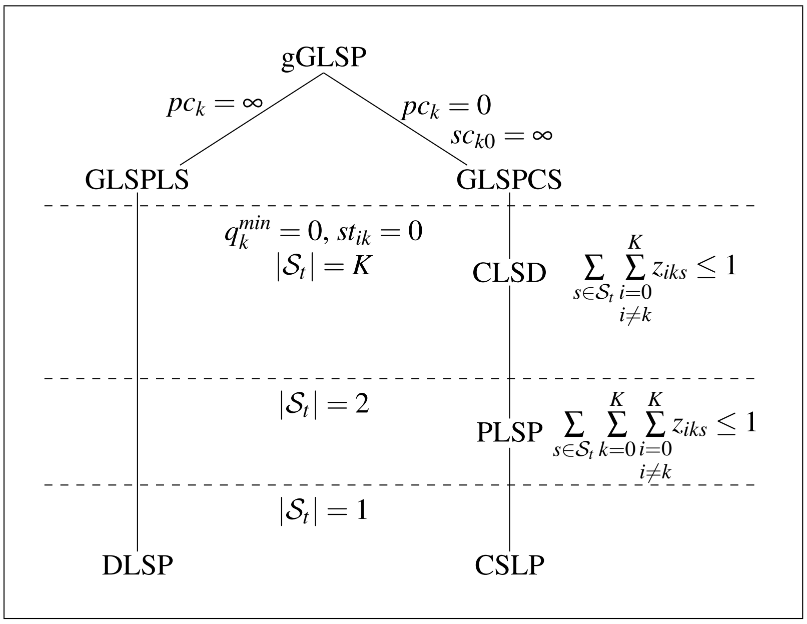

gGLSP
$pc_k = \infty$
$pc_k = 0$
$sc_{k0} = \infty$
GLSPLS
GLSPCS
$q_k^{min} = 0$, $st_{ik} = 0$
$|\mathcal{S}_t| = K$
CLSD
$\sum_{s \in \mathcal{S}_t} \sum_{\substack{i=0 \\ i \neq k}}^{K} z_{iks} \leq 1$
$|\mathcal{S}_t| = 2$
PLSP
$\sum_{s \in \mathcal{S}_t} \sum_{k=0}^{K} \sum_{\substack{i=0 \\ i \neq k}}^{K} z_{iks} \leq 1$
$|\mathcal{S}_t| = 1$
DLSP
CSLP

Figure 2.1: Relationship between models (see Meyr, 1999, p. 84)

Like the PLSP, the *CSLP* allows at most one single changeover per period and the conservation of the setup state ($pc_k = 0$, $sc_{k0} = \infty$). However, compared to the PLSP, at most one single product can be produced per period. To cover this constraint in the gGLSP, the number of microperiods per macroperiod is set to one ($|\mathcal{S}_t| = 1$). Variations of the basic CSLP are described in Sect. 2.3.4.

The main characteristic of the *DLSP* is its all-or-nothing assumption. The length of a period is equal to the capacity of the considered resource. In each period, there is only the choice to produce during the complete period or not to produce at all. Since this assumption is very restrictive, usually the lengths of the periods are very short. In this regard, the DLSP is a small-bucket model. In periods without production the setup state is lost. Thus, the DLSP can be derived from the GLSPLS (and with $pc_k = \infty$ from the gGLSP as well; see Fig. 2.1) by fixing the number of microperiods per macroperiod to one ($|\mathcal{S}_t| = 1$). Then the length of a microperiod equals its corresponding macroperiod's length. Since no setup times exist ($st_{ik} = 0$) and setup states are not conserved ($\overline{q}_{ks} = 0$), Equations (2.2) either allow the production of a physical product $k > 0$ for the whole period length or staying in the neutral state $k = 0$. This puts the all-or-nothing assumption into practice. Extensions of the DLSP are discussed in Sect. 2.3.5.

Note that in Figure 2.1 on the paths gGLSP - GLSPCS - CLSD - PLSP - CSLP and gGLSP - GLSPLS - DLSP, the models become increasingly restricted and specialized with respect to the assumptions. This means that less real world applications can be found which match their assumptions. However, they may provide some bases for more dedicated solution techniques. So far, we have presented a first basic differentiation between the basic model types available. In the following, we present a classification scheme that serves as a basis for a more detailed discussion of the models presented in the literature.

2.2.2 Classification scheme

In the last decades, numerous modifications and extensions of the lotsizing and scheduling models presented in Sect. 2.2.1 have been presented. To structure the discussion of these models, we use the classification scheme presented by Meyr (1999). We first discuss all attributes and their possible values. Table 2.2 summarizes the classification scheme and defines acronyms. Attributes in curly brackets are mandatory, while attributes in squared brackets are optional. At the end of Sect. 2.2.2, a classification of the above basic models will serve as an example.

Table 2.2: Classification scheme

Description	Attribute	Potential value	Acronym
BOM	**Bill-of-materials structure**	single-level	1
		serial	s
		assembly	a
		divergent	d
		general, acyclic	g
Ps	**Production stage**	single-stage	1[:Tl]
		multi-stage	{Nb}:{Sq}[:Tl]
{Nb}	Number	given, limited number	♯
		free	fr
{Sq}	Sequence	serial	s
		cross-linked	cl
[:Tl]	Transfer of lots	before completion (open)	o
		only after completion (closed)	c
M	**Machines per stage**	one machine	1
		parallel, identical	pi
		parallel, non-identical	pn
Sc	**Setup costs**		{Sdp}[:Tie]
{Sdp}	Sequence dependence	sequence-independent	si
		sequence-dependent	sd
[:Tie]	Triangle inequality	must be kept	Δk
	(from cost perspective)	violable	Δv
Css	**Conservation of setup state**	continuous setup	cs
	(from cost perspective)	lost setup	ls
		both possible in model	cl
St	**Setup time**		{Sdp}:{Lg}[:Per]
{Lg}	Length of setups	discrete	d
		upper bound	max
		free	fr
[:Per]	Number of considered periods	one (macro)period	p
		limited number of (micro)periods	♯
		free number of (micro)periods	fr
exoT	**Exogenous time structure**	free scheduling of exogenous state changes	fr
		discrete external time grid	d:{LotX}
{LotX}	Maximum number of lots	limited number	♯
	(per period in external time grid)	number of products	K
		free	fr

Description	Attribute	Potential value	Acronym
endoS	**Endogenous state changes**	fixed on external time grid	fi
		free towards exogenous time structure	fr
		additional discrete internal time grid	d:{LotN}:{Time}
{LotN}	Maximum number of lots (per period in internal time grid)	limited number	$\sharp$
{Time}	Time of change	fixed on internal time grid	fi
		free towards internal time grid	fr
Ls	**Lotsize**	discrete, multiple	dm
		continuous, ≥ 0	c
		continuous, $\geq$ minimum value	min
		continuous, $\leq$ maximum value	max
		continuous, between minimum and maximum value	mima

Bill-of-materials structure: The product structure is documented with the bill-of-materials (BOM). If only one BOM-level is considered, the product structure is called *single-level*. Otherwise, it is called *multi-level*. In the latter case, the product structure is *serial* if each product has at most one predecessor and one successor. A product structure is of the *assembly* (converging) type if each product has at most one successor but can have multiple predecessors. If a product has several successors but at most one predecessor, a *diverging* structure is given. A *general* product structure comprises converging and diverging portions.

Production stages: The production system may consist of only one *(single-stage)* or of multiple stages of production (*multi-stage*). In the multi-stage case, some models consider a *given, limited number* of production stages. If the number is not restricted, this will be denoted as *"free"*. Furthermore, production systems can be distinguished by the sequence of their material flow. For example, the material flow can be *cross-linked* in case of a job-shop production, whereas a flow line system always shows a *serial* sequence. Furthermore, the transfer of production lots between the stages may differ. In some cases, a lot is transferred to the next stage as a whole only *after its completion* ("closed production"). In other cases, processed parts are already moved to the next stage *before* the whole lot has been *completed* ("open production").

Machines per stage: One or several *parallel* machines, showing the same functionality, may be available to execute a certain production task. Parallel machines are called "homogeneous" if they show *identical* costs and production speeds. In case they are *not identical*, they are called "heterogeneous". Note that "lotsplitting" may occur if parallel machines exist.

Setup costs: Setup operations are often associated with costs, e.g., when a machine must be cleaned before a product is produced. In some industries, these costs are the same no matter which product has been produced last on this resource (*"sequence-independent"*). In other industries, however, the setup costs differ with respect to those predecessors (*"sequence-dependent"*[5]). For example, when colored products are produced, a change from a light to a dark color usually is

[5]Unfortunately, "sequence-independent" is a typing error in Copil et al. (2017).

less laborious and cheaper than a change from a dark to a light color. The *triangle inequality is kept* if the costs for a changeover from product i to k are lower than the costs of a detour from i to j to k, i.e., if $sc_{ij} + sc_{jk} \geq sc_{ik}$ for any j. There are also industries (such as the chemical industry) in which triangle inequalities may be *violated*.

Conservation of setup state: As explained in Sect. 2.2.1, the setup state may be conserved (*continuous setup*) or get *lost* after idle periods. As the gGLSP has shown, *both* situations may be covered by the same model.

Setup time: Regardless whether setup states are conserved or not, the duration of a setup may be bound by some *discrete* time pattern, be limited by some predefined *upper bound* or not be restricted in any way (*"free"*). The latter case is rather common for pure scheduling models, but not so much for lotsizing and scheduling models. These usually depend on some exogenous time structure (see below) to represent capacities, inventories and demand. With regard to this time structure, setup times are often limited by the period length, i.e., setup operations do not exceed the end of a period and therefore last for at most *one period*. In contrast, period-overlapping setup operations do ensure more flexibility. These may be *limited* by a predefined *number of periods* or not be restricted at all (*"free"*). Like setup costs, setup times may be sequence-dependent.

Exogenous time structure: The exogenous time structure represents the points in time at which externally given events, that are defined by the data of the model, are considered. In lotsizing and scheduling models typically the demands, the capacity consumption and the inventory development are concerned. State changes or events relating to these data usually do not occur *"freely"* at any point in time, but are bound to a predefined, *discrete external time grid*. Whether the time structure represents macroperiods (as in the CLSD, for example) or microperiods (as, e.g., in the CSLP), depends on the context. As Sect. 2.2.1 has shown, the number of lots per period may also differ – depending on the model formulation. It may be *limited* in advance to some maximum *number* $|\mathcal{S}_t|$. For example, $|\mathcal{S}_t|$ equals 2 for the PLSP or the *number of products K* for the CLSD. If an integer number $|\mathcal{S}_t|$ can arbitrarily be set, the maximum number of lots per period is denoted as *"free"*.

Endogenous state changes: The endogenous time structure represents the points in time at which internal events are captured by decision variables. These events are, for example, the starting and ending times of production lots or, in general, changes of the production system's state like changeovers. They might occur at any time (*"free"*), be bound on the external time grid (*"fixed"*) or depend on an *additional discrete internal time grid*. If the latter one exists, state changes can also be *free* with respect to this internal time grid or *fixed* on it. Moreover, the number of lots can then also be *limited* for each period of the *internal* time grid.

Lotsize: The size of a lot can either be *discrete* or *continuous*. Discrete means that only integer *multiples* of some predefined batch size are allowed. The batch size may be a modeling constraint (as in the DLSP) and/or a technical requirement (e.g., in chemical industries when reactors or

Table 2.3: Classification of basic models

References	BOM	Ps	M	Sc	Css	St	exoT	endoS	Ls	Industry	Heuristics/comments
Meyr (1999)	1	1	pn	sd:Δv	cl	sd:max:p	d:fr	fr	min	–	Not solved (gGLSP)
Fleischmann and Meyr (1997)	1	1	1	sd:Δv	ls	–	d:fr	fr	min	–	Not solved (GLSPLS)
Fleischmann and Meyr (1997)	1	1	1	sd:Δv	cs	–	d:fr	fr	min	FI	TA+BA (GLSPCS)
Haase (1996)	1	1	1	sd:Δk	cs	–	d:K	fr	c	–	BA (CLSD)
Haase (1994)	1	1	1	sd:Δk	cs	–	d:2	fr	c	–	RR (PLSP)
Karmarkar and Schrage (1985)	1	1	1	si	cs	–	d:1	fr	c	–	B&B+LR (CSLP)
Fleischmann (1990)	1	1	1	si	ls	–	d:K	d:1:fi	dm	–	B&B+LR (DLSP)

Table 2.4: Industrial settings acronyms

Acronym	Industrial setting
AFI	Animal food industry
AI	Automobile industry
BI	Beverage industry
CGI	Consumer goods industry
CHES	DuPont, BASF, James River, Champion international[7]
CI	Chemical industry
EI	Electronics industry
FI	Food industry
PhI	Pharmaceutical industry
PI	Process industry
SI	Semiconductor industry

ovens must be filled to a certain grade). The production quantity that comprises multiple batches of the same product consecutively produced is sometimes called a "campaign". Continuous lots may also be limited in their size. However, here the *minimum* or *maximum bounds* do not equal each other and thus allow a higher degree of freedom for planning.

In the following, the gGLSP proposed by Meyr (1999) and the basic models of Sect. 2.2.1 are categorized according to this classification. An overview is given in Table 2.3. A dash means that the attribute is not considered in a model, e.g., if the model does not include setup times.[6] An additional column shows (if available) the underlying industry of the described model formulation. The meaning of the associated acronyms can be found in Table 2.4. In the last column of Table 2.3, solution approaches and further interesting characteristics are summarized. Table 2.5 shows the associated acronyms of the different solution approaches.

The gGLSP is formulated for a single-level product structure and a single production stage. In contrast to the simplified version (2.1) – (2.7), the original formulation of Meyr (1999) includes multiple parallel, non-identical machines. The setup costs are sequence-dependent and may violate the triangle-inequalities. Depending on the input data, the setup state can be conserved or get lost after idle periods. Setup times are sequence-dependent and limited to a maximum duration which is equal to the length of a (macro)period. The discrete external time grid $t = 1, \ldots, T$ defines the exogenous time structure. The

[6]In the tables shown below, a field will be left empty if it is not possible to identify the value of the attribute.
[7]Practical problems, not dedicated to an industry, c.f., Baker and Muckstadt Jr. (1989)

dynamic demands as well as the capacities are related to this time grid. The maximum number of lots within a macroperiod is free since an integer number of microperiods per macroperiod can arbitrarily be chosen. Endogenous state changes like changeovers are not bound to the exogenous time structure. Furthermore, lotsizes are continuous, but can be restricted to a minimum size. Meyr (1999) does not report experiences about the implementation or solution of this model.

Table 2.5: Solution approaches acronyms

Acronym	Solution approaches
ATSP	Asymmetric traveling salesman problem
BA	Backward-oriented heuristic
BACKADD	Backward-oriented, regret-based, biased random sampling method
B&B	Branch&bound
B&C	Branch&cut
CG	Column generation
DP	Dynamic programming
DS	Demand shuffle
F&O	Fix&optimize
F&R	Fix&relax
GA	Genetic algorithm
GRASP	Greedy randomized adaptive search procedure
HLSA	Hybrid Lagrangian-simulated annealing-based heuristic
HOPS	Hamming-oriented partition search
INSRF	Iterative variable neighborhood search with a relax-and-fix construction heuristic
LD	Lagrangean decomposition
LP	Linear programming
LR	Lagrangean relaxation
LS	Local search
MA	Memetic algorithm
MIP	Mixed integer programming
PSO	Particle swarm optimization
RH	Rolling horizon
RM	Randomized measures
RR	Randomized regrets
SA	Simulated annealing
SPL	Simple plant location
STN	State-task-network
TA	Threshold accepting
TS	Tabu search
TSP	Traveling salesman problem
TSPTW	Traveling salesman problem with time windows
VNDS	Variable neighborhood decomposition search
VNS	Variable neighborhood search

Both the GLSPLS and GLSPCS introduced by Fleischmann and Meyr (1997) are restricted to a single machine and do not take setup times into account. They differ with respect to their consideration of the setup state. While the GLSPLS is not solved, different solution heuristics for the GLSPCS are presented. These apply the local search (LS) meta-heuristic threshold accepting (TA) to generate setup sequences and different backward oriented greedy heuristics (BA) to calculate the lotsizes. In a backward-oriented heuristic, the solution is created starting in the last period of the planning horizon and going step by step backwards until the first period is reached. Thereby, predefined operations are performed in each considered period. The heuristics are tested for problem instances originating from the food industry (FI).

The CLSD of Haase (1996) considers a single-level product structure and one machine. Setup costs can be sequence-dependent. Nevertheless, the triangle inequality must hold since each product can be set up at most once per macroperiod. The setup state is conserved after idle periods, but setup times are not considered. A discrete time grid is used as exogenous time structure. In each period of this time grid, the number of lots is limited to the number of products. The endogenous state changes are free and not bound to the exogenous time structure. Continuous lotsizes are possible. The model is solved with a backward oriented heuristic.

The PLSP proposed by Haase (1994) is similar to the CLSD. However, the number of lots per period of the exogenous time grid is limited to two. Haase proposed a randomized regret (RR)-based heuristic to solve the problem. This heuristic determines which product to schedule in a period using so-called randomized regrets. These regrets define the lost potential savings if a product is not produced in a certain period. The product to be scheduled is randomly selected with a probability which is proportional to the regrets.

The CSLP of Karmarkar and Schrage (1985) covers only sequence-independent setup costs. Besides, its main difference to the PLSP is that at most one product can be produced per period. The model is solved by a combination of branch&bound (B&B) and Lagrangean relaxation (LR). The LR generates lower bounds for minimization problems by allowing the violation of a complicating constraint and introducing penalty costs with the help of Lagrangean multipliers instead. The Langrangean decomposition (LD) is a special case which decomposes the overall problem into subproblems. The lower bound derived by solving the relaxed problem and an upper bound resulting from a (heuristically generated) feasible solution are updated in an iterative procedure. An adaptation of the multipliers supports convergence of the lower and upper bound.[8]

Fleischmann (1990) proposes a model allowing only one setup per period. The production for a product either runs at full capacity or not at all (all-or-nothing assumption). For comparison reasons and in order to use data for macroperiod models from the literature, Fleischmann (1990) also describes a transformation from the CLSP to the DLSP which in general allows to model demand, inventory holding costs and capacities on basis of macroperiods while the all-or-nothing assumption still holds for significantly shorter microperiods (see Table 2.3). As his numerical tests show, Fleischmann uses a discrete external time grid, which limits the number of lots per macroperiod to the number of products, to model the macroperiods. An additional discrete internal time grid models the microperiods. In a microperiod of the internal time grid, the production of at most one product is allowed and state changes are fixed to this internal time grid. This way, the all-or-nothing property is put into practice.

[8]See Buschkühl et al. (2010), pp. 243-244.

A lotsize has either to be zero or an integer multiple of a complete microperiod's total production quantity. The model is also solved using a combination of branch&bound and Lagrangean relaxation.

As demonstrated in Table 2.3 for the paper of Fleischmann (1990), in the literature review of Sect. 2.3 we classify only the most advanced/functional model of a publication if several models are proposed within the same paper.

2.3 Literature review

In the following we discuss more than 160 publications which are relevant for simultaneous lotsizing and scheduling according to the criteria explained above. The major focus is set on papers that appeared in the last two decades. Some earlier papers are included in order to illustrate the genesis of the field.

Sections 2.3.1 – 2.3.5 present the extensions of the basic formulations of the GLSP, CLSD, PLSP, CSLP and DLSP in the sequence implied by Figure 2.1 and Table 2.3. In Sect. 2.3.6 we discuss research which we have not been able to uniquely assign to one of these model types. At the end of each section, a table summarizes the various papers according to the classification of Sect. 2.2.2. Within each table, the papers are sorted in sequence of their date of publication in order to point out the historical development of each modeling approach over time.

2.3.1 General lotsizing and scheduling problem (GLSP)

The GLSP has first been presented by Fleischmann and Meyr (1997). Its main characteristics have already been discussed in the previous sections. Koçlar and Süral (2005) notice that the GLSPCS is limited by the fact that the minimum lotsize must be fulfilled completely in the first microperiod of a lot. This restriction is based on modeling reasons and it is relevant at macroperiod boundaries. They propose a modification of the minimum lotsize constraint which overcomes this restriction.

The gGLSP has first been presented by Meyr (1999) including sequence-dependent setup times and non-identical parallel machines. As shown in Sect. 2.2.1, it is possible to choose between the conservation and the loss of a setup state in idle periods. Further variations that can be derived from the general formulation (c.f., Meyr 1999) are the GLSP with setup times (GLSPST) presented in Meyr (2000) and the GLSP for parallel lines (GLSPPL) published in Meyr (2002). Both problems are solved with improved versions of the heuristic of Fleischmann and Meyr (1997). For a fixed setup pattern, these determine the lotsizes optimally, instead of heuristically, by means of "dual reoptimization". The single-machine heuristic performs very well because a fast network flow algorithm can be applied. The multi-machine heuristic, however, is less satisfying. Thus, Meyr and Mann (2013) propose a better scalable alternative. They aggregate the original multi-machine problem and solve the reduced problem heuristically in order to decompose the original GLSPPL into a set of isolated single-machine problems. These are solved by the above mentioned GLSPST heuristic of Meyr (2000). If some solution is infeasible with respect to the original problem, the production capacities of the aggregate multi-machine problem are modified for a new iteration. In this way, different types of real world problems (production of incontinence pads and acrylic glass, printing of consumer goods' labels) are successfully solved.

Meyr (2004) extends the single-stage GLSP to multiple production stages (GLSPMS). He solves small test instances using CPLEX. Seeanner and Meyr (2013) improve the model with respect to the synchronization of the machines. Different model reformulations and MIP-based solution heuristics such as fix&relax (F&R) are compared. The F&R heuristic divides the binary variables of a model into distinct subsets and considers these subsets sequentially instead of simultaneously. Thus three groups of variables result: within the currently considered subset, all variables are binary and have to be optimized. Variables of already earlier considered subsets are fixed and the remaining variables are relaxed. The subsets are processed in a predefined sequence until all binary variables have been optimized. The time structure of the multi-stage GLSP is similar to the single-stage formulation. However, the starting times of microperiods are identical on each machine. Using this common time structure, the synchronization of predecessor and successor relations is facilitated. The GLSPMS is modified by Seeanner et al. (2013) who allow for changeovers in the first microperiod. Furthermore, the synchronization between production lines is again formulated more generally and compactly. The authors propose a combination of variable neighborhood decomposition search (VNDS) and fix&optimize (F&O). In contrast to the F&R heuristic, the F&O heuristic needs to start with an initial solution. Some subsets of binary variables of this solution are released and re-optimized. The other binary variables remain fixed to their current values. This procedure is repeated by varying the subset of variables to be released. If the initial solution was feasible, newly generated solutions will be feasible, too, and not worse than already found solutions. An extension of the aforementioned formulation is presented by Seeanner (2013, p. 143). He considers limited raw materials and setup operators as secondary scarce resources. A further extension allows microperiod-overlapping setup times and minimum lotsizes (Seeanner 2013, p. 148).

Another multi-stage GLSP formulation is proposed by Fandel and Stammen-Hegener (2006). In order to calculate the holding costs more accurately, three types of microperiods are distinguished: one type for production, one for setups and one for idle time. Unfortunately, the resulting model formulation is non-linear.

Günther et al. (2006) introduce the concept of block planning. In contrast to the recent paper of Günther (2014), in which the usefulness of holding costs is called into question, here holding costs associated with macroperiods are explicitly considered. Block planning groups production orders into blocks. The sequence of orders and simultaneously the sequence of products within a block are predefined in advance. However, the sizes of the orders and their corresponding processing times are decision variables. Changeovers within a block cause minor sequence-independent setup costs and times. Changeovers from one block to another cause major (constant) setup times. Because blocks can be interpreted as macroperiods and the processing times of the orders can be interpreted as microperiods of variable lengths, the model is closely related to the GLSP. However, since the sequence of products per block is predefined, it can much easier be solved.

The synchronized and integrated two-level lotsizing and scheduling problem (SITLSP) is proposed by Toledo et al. (2006). It tackles a problem from the beverage industry. Raw materials are stored in several tanks which are connected to various bottling lines. Planning concerns the filling of the tanks with different flavors, the assignment of the tanks to the bottling lines and the lotsizing and scheduling of the final products on the bottling lines. Additionally to the GLSP time grid, they use a CSLP-like time structure to synchronize the two production stages. Only small test instances are solved using a standard solver. The model is explained in more detail by Toledo et al. (2015). Furthermore, they propose test instances with benchmark results. Toledo et al. (2008b) heuristically solve the SITLSP by means of different genetic algorithm (GA) approaches. Further solution heuristics are proposed by Toledo et al. (2008a). They apply tabu search (TS), TA, GA and a combination of GA with TS and with TA, respectively. Additionally, Toledo et al. (2009) also propose a GA and test it using data from a soft drink company. Toledo et al. (2010) concentrate on parallel GA approaches taking advantage of multi-core processors.

Marinelli et al. (2007) consider a two-stage production system from the food industry. In the first stage, tanks are supplied with yogurt. In the second stage, the yogurt is filled into pots. Sequence-independent setup times and costs arise when a tank is refilled and when the package sizes are changed on the filling stage. A heuristic that decomposes the problem into a scheduling and a lotsizing problem is proposed. This approach leads to near-optimal solutions in very short computation times using data of the company. Although the authors classify their model as a “hybrid continuous setup and capacitated lotsizing problem” (CSLP-CLSP), we prefer to discuss it within the current section because of its GLSP-typical time structure.

A problem similar to the SITLSP is considered by Ferreira et al. (2009). However, their assumptions are more restrictive, e.g., that each filling line can only be connected to a single tank at the same point in time. The tanks and the filling lines are synchronized based on waiting time calculations. The formulation is called two-stage, multi-machine lot-scheduling model (P2SMM). Assuming that the tanks are no bottlenecks, the single-stage, multi-machine lot-scheduling model (P1SMM) is formulated. After this model has been solved, the setup variables of the P2SMM are determined (relaxation approach). Furthermore, F&R strategies are used to solve the P2SMM directly or combined with the relaxation approach.

A model similar to the P1SMM is proposed by Ferreira et al. (2010). They consider small-scale soft drink plants with just one filling line. Several tanks are available. Here, the filling line is the only bottleneck. As a consequence only minimum and maximum filling levels of the tanks must be considered besides the lotsizing and scheduling of the filling line. Several combinations of CPLEX strategies (e.g., presolve on/off) and F&R strategies are used to solve the model.

Ferreira et al. (2012) propose different alternative single-stage formulations of the P2SMM. The variables associated with the tank are omitted because taking the maximum of the setup times of the tank and the filling line realizes the synchronization. This way it is ensured that the tank is always ready when filling starts. Numerical tests are carried out with the original P2SMM as a benchmark. Another solution approach for the P2SMM of Ferreira et al. (2009) is proposed by Toledo et al. (2014). They apply a GA to determine lot sequences and use an LP model to calculate the lotsizes. Baldo et al. (2014) modify the P2SMM for an application in the brewery industry. They divide the planning horizon into two parts. In the first part planning is very detailed, but in the second part only the lotsizing is considered. F&R combined with F&O solves the problem heuristically.

Two models for the production planning of animal food compounds are proposed by Toso et al. (2009). The models differ in the consideration of setup times at macroperiod boundaries. Production stages are serially arranged, but only one stage is a bottleneck. Thus a single-stage model is sufficient. The lotsize is based on batches, due to technical and economical reasons. The authors solve the models using CPLEX and different F&R strategies.

Lang (2010) embeds a State-task-network (STN) approach into a multi-stage GLSP formulation. For each task, it is possible to define the quantities of the input and the output products. Sequence-dependent setup costs and times occur if a changeover from one task to another occurs. Using this formulation, it is possible to consider product substitution.

Based on Fandel and Stammen-Hegener (2006), Mohammadi (2010) proposes a model for a flexible flow-shop. He also uses three types of microperiods (production, setup and idle type), but defines the number of microperiods per macroperiod in advance. Thus, the model is easier to solve. This is done heuristically by rolling horizon (RH) approaches and F&R. Rolling horizon is a specific form of F&R in which the planning horizon is divided into three subsections. The first subsection uses fixed variables (frozen zone), the second subsection represents the problem in detail and the third subsection represents the model in a simplified way, e.g. without setup times. The problem is solved several times. In each iteration, the detailed results of the previous solution are added to the frozen zone, i.e. the detailed zone is iteratively shifted until the end of the planning horizon is reached.

Almeder and Almada-Lobo (2011) consider tools as an additional, capacitated resource in a single-stage, multi-machine problem. It is possible to produce multiple products with the same tool. Thus, tool changeovers instead of product changeovers are modeled. Since a tool cannot simultaneously be used on two or more machines, tool synchronization is required. This is accomplished with the help of continuous variables for the starting and ending times of tool use. The model is compared to a CLSD-based formulation which is also presented in this publication.

Inspired by a practical case from the process industry, Transchel et al. (2011) formulate a model for a single machine producing multiple end products from multiple pre-products. In order to account for the limited availability of pre-products, it is possible to restrict the corresponding production capacities. The authors propose two reformulations as transportation problems and compare all three models by numerical tests.

Deterioration and perishability is considered by Pahl et al. (2011). They propose a model including end products which deteriorate after a maximum lifetime. The model is solved using Xpress.

Camargo et al. (2012) formulate a model for the following planning problem: multiple products are produced on parallel, non-identical machines. These machines are fed by a single, upstream machine, which produces different pre-products. Each end product requires exactly one pre-product. Nevertheless, a pre-product can be used to produce several end products belonging to the same product family. In each microperiod, only one single pre-product can be processed. All downstream machines are fed with this pre-product. That means, that it is possible to produce different end products on different machines in one microperiod, if all end products require the same pre-product. The model formulation is compared to a CLSD-based and a continuous formulation in numerical tests using CPLEX. Camargo et al. (2014) adapt this model for usage in the yarn production. Here, on the first production stage different types of fibers are blended on a single machine. In the second stage different yarn types are processed using the fibers from the first stage. Unlike Camargo et al. (2012), holding costs are only indirectly considered by using time-varying production costs. A Hamming-oriented partition search (HOPS) is proposed to solve the problem. This approach uses B&B in combination with F&O. A partition attractiveness measure, which is calculated using Hamming distances, helps deciding about which variables to fix.

Santos and Almada-Lobo (2012) propose a model for a pulp and paper mill planning problem. Here, wood chips are processed in a digester resulting in black liquor and virgin pulp. The black liquor is used to produce energy and the virgin pulp is combined with recycled pulp to produce different types of paper. Many specific properties of the production process are taken into account. For example, the digester has a flexible rotation speed, which affects the output and can only be changed within a limited range per microperiod. Converting black liquor into energy is also restricted with respect to different aspects, e.g., by the capacity of the evaporator or concerning the potential steam output. The model is solved by means of a stochastic F&O approach (see James and Almada-Lobo 2011). A modification of the above model with the objective to maximize the steam output used to generate electrical energy is presented by Figueira et al. (2013). These authors propose a new solution approach in which a variable neighborhood search (VNS) determines the setup pattern. While the rotation speed of the digester is selected heuristically, the continuous variables of the remaining model (after fixing the setup pattern) are calculated by an exact method. The solution approach is tested for scenarios based on real world data. A GA-based solution approach for the problem of Santos and Almada-Lobo (2012) is presented by Furlan et al. (2013). Furlan et al. (2015) extend this model to the case of parallel non-identical paper machines. Different GA approaches are used to solve this model.

A planning problem from the wood floor industry is treated by Tiacci and Saetta (2012). These authors formulate a single-machine model without consideration of setup times. Inspired by de Araujo et al. (2007), they simplify their model using a rolling horizon approach which accounts for microperiods and setup costs in the first macroperiod only. Since omitting setup costs after the first macroperiod has a strong influence on the solution and could lead to poor production plans, the authors adapt their model and approximate setup costs in macroperiods $t \geq 2$.

Mohammadi and Poursabzi (2014) present a GLSP formulation for multiple production stages. Two rolling horizon approaches are proposed.

A multi-stage GLSP formulation for a flexible job-shop problem is presented by Rohaninejad et al. (2015). Different production stages are synchronized using additional variables to track the starting and ending times of microperiods. A combination of a GA and a particle swarm optimization (PSO) algorithm is used to solve the problem.

The presented GLSP models are summarized in Table 2.6. The first publications up to Meyr (2002) consider only single-level bill-of-materials (BOM = 1) and one production stage (Ps = 1). The first multi-stage formulation was published by Meyr (2004). Afterwards, most of the models consider multi-stage bill-of-materials and multiple production stages (e.g. Toledo et al. (2006) consider divergent bill-of-materials and two serially arranged production stages). There is also a focus on parallel heterogeneous machines per production stage (M = pn). Only a small number of models does not consider sequence-dependent setup costs and times (*Sc* and *St* = *si*). Examples are Günther et al. (2006) and Marinelli et al. (2007). The time structure of all models is typical for GLSP formulations: a discrete time grid consisting of macroperiods is used as time structure for externally given events (*exoT* = *d:K* or *d:fr*). Since microperiods with flexible lengths are used, the endogenous state changes are not fixed to a time grid (*endoS* = *fr*). Usually, the number of lots per macroperiod is free (*exoT* = *d:fr*) since the number of microperiods per macroperiod can be arbitrarily chosen. In some cases, the number of lots per macroperiod is limited to the number of products (*exoT* = *d:K*). This is sometimes directly represented in the model formulation (see e.g. Rohaninejad et al. 2015). In other cases, allowing a setup for more than *K* products per macroperiod would not be appropriate (e.g. if the triangle inequality is kept *Sc* = *sd:*Δ*k*). An example is Camargo et al. (2012). Most of the models permit a continuous lotsize (*Ls* = *c*), often restricted by a lower bound (*Ls* = *min*) as it can be found starting with Fleischmann and Meyr (1997) up to Furlan et al. (2015). Minimum lotsizes are important to appropriately model the production technology or to correctly model the setup costs if the triangle inequality is violated. Most of the models are based on real world scenarios which are taken from various industries (see column *industry*). To solve the problems, meta-heuristics based on local search or evolutionary principles are mainly used as it is shown in column *heuristics/comments*.

Table 2.6: Literature overview GLSP

References	BOM	Ps	M	Sc	Css	St	exoT	endoS	Ls	Industry	Heuristics/comments
Fleischmann and Meyr (1997)	1	1	1	sd:Δv	cs	–	d:fr	fr	min	FI	TA+BA; GLSPLS not solved
Meyr (1999)	1	1	pn	sd:Δv	cl	sd:max:p	d:fr	fr	min	–	Not solved
Meyr (2000)	1	1	1	sd:Δv	cs	sd:max:p	d:fr	fr	min	FI	TA/SA+dual reoptimization
Meyr (2002)	1	1	pn	sd:Δv	cs	sd:max:p	d:fr	fr	min	CHES	TA/SA+dual reoptimization
Meyr (2004)	g	fr:cl	pn	sd:Δv	cl	sd:max:2	d:fr	fr	min	–	MIP-solver
Koçlar and Süral (2005)	1	1	1	sd:Δv	cs	–	d:fr	fr	min	–	Not solved
Fandel and Stammen-Hegener (2006)	g	fr:cl	pn	sd:Δk	cl	sd:max:p	d:K	fr	c	–	Not solved; non-linear formulation
Günther et al. (2006)	1	1	1	si	ls	si:max	d:K	fr	c	CGI	MIP-solver; block planning
Toledo et al. (2006)	d	2:s	pn	sd:Δk	cs	sd:max:2	d:K	fr	c	BI	MIP-solver; add. time structure for synchron.
Marinelli et al. (2007)	s	2:s	pn	si	ls	si:max:1	d:K	fr	dm	FI	Decomposition-heuristic
Toledo et al. (2008a)	d	2:s	pn	sd:Δk	cs	sd:max:2	d:K	fr	c	BI	TS, TA, GA; add. time structure for synchron.
Toledo et al. (2008b)	d	2:s	pn	sd:Δk	cs	sd:max:2	d:K	fr	c	BI	GA; add. time structure for synchron.
Ferreira et al. (2009)	d	2:s	pn	sd:Δv	cs	sd:max:p	d:fr	fr	min	BI	Relaxation to 1 stage + secondary resource, F&R
Toledo et al. (2009)	d	2:s	pn	sd:Δk	cs	sd:max:2	d:K	fr	c	BI	GA; add. time structure for synchron.
Toso et al. (2009)	1	1	1	–	cs	sd:max:p	d:K	fr	dm	AFI	MIP-solver, F&R
Ferreira et al. (2010)	d	1	1	sd:Δv	cs	sd:max:p	d:fr	fr	min	BI	MIP-solver, F&R; secondary resources
Lang (2010)	g	fr:cl	pn	sd:Δv	cl	sd:max:2	d:fr	fr	min	–	Not solved; STN, product substitution
Mohammadi (2010)	s	fr:s:c	pi	sd:Δk	cs	sd:max:p	d:K	fr	c	–	RH+F&R

References	BOM	Ps	M	Sc	Css	St	exoT	endoS	Ls	Industry	Heuristics/comments
Toledo et al. (2010)	d	2:s	pn	sd:Δk	cs	sd:max:2	d:K	fr	c	BI	GA; add. time structure for synchron.
Almeder and Almada-Lobo (2011)	1	1	pn	sd:Δk	cs	sd:max:p	d:K	fr	c	SI	MIP-solver; secondary resources
Pahl et al. (2011)	1	1	1	sd:Δv	cs	sd:max:p	d:fr	fr	min	CGI	MIP-solver; perishability
Transchel et al. (2011)	d	1	1	sd:Δv	cs	sd:max:p	d:fr	fr	min	PI	MIP-solver
Camargo et al. (2012)	d	1	pn	sd:Δk	cs	sd:max:p	d:K	fr	c	PI	MIP-solver; secondary resources
Ferreira et al. (2012)	d	1	pn	sd:Δv	ls	sd:max:p	d:fr	fr	mima	BI	MIP-solver; secondary resources
Santos and Almada-Lobo (2012)	g	2:s	1	sd:Δv	cs	sd:max:p	d:fr	fr	min	PI	Stochastic F&O; secondary resources
Tiacci and Saetta (2012)	1	1	1	sd:Δk	cs	–	d:K	fr	c	CGI	RH
Figueira et al. (2013)	g	2:s	1	sd:Δv	cs	sd:max:p	d:fr	fr	min	PI	VNS; secondary resources
Furlan et al. (2013)	g	2:s	1	sd:Δv	cs	sd:max:p	d:fr	fr	min	PI	GA; secondary resources
Meyr and Mann (2013)	1	1	pn	sd:Δv	cs	sd:max:p	d:fr	fr	min	CGI	Decomposition
Seeanner (2013, p. 143)	g	fr:cl	pn	sd:Δv	cl	sd:max:2	d:fr	fr	min	–	MIP-solver; secondary resources
Seeanner (2013, p. 148)	g	fr:cl	pn	sd:Δv	cl	sd:fr	d:fr	fr	min	–	Not solved
Seeanner et al. (2013)	g	fr:cl	pn	sd:Δv	cl	sd:max:2	d:fr	fr	min	CGI	VNDS+F&O
Seeanner and Meyr (2013)	g	fr:cl	pn	sd:Δv	cl	sd:max:2	d:fr	fr	min	CGI	Reformulations, F&R, LP&Fix
Baldo et al. (2014)	d	2:s	pn	si	cs	sd:max:p	d:K	fr	c	BI	F&R+F&O
Camargo et al. (2014)	d	1	pn	sd:Δk	cs	sd:max:p	d:K	fr	c	CGI	HOPS; secondary resources
Mohammadi and Poursabzi (2014)	g	fr:cl	pn	sd:Δk	cl	sd:max:p	d:K	fr	c	–	RH
Toledo et al. (2014)	d	2:s	pn	sd:Δv	cs	sd:max:p	d:fr	fr	min	BI	GA
Furlan et al. (2015)	g	2:s	pn	sd:Δv	cs	sd:max:p	d:fr	fr	min	PI	GA; secondary resources
Rohaninejad et al. (2015)	g	fr:cl:c	pn	si	ls	sd:max:p	d:K	fr	c	–	GA+PSO
Toledo et al. (2015)	d	2:s	pn	sd:Δk	cs	sd:max:2	d:K	fr	c	BI	MIP-solver; add. time structure for synchron.

2.3.2 Capacitated lotsizing problem with sequence-dependent setups (CLSD)

Another large-bucket model formulation for simultaneous lotsizing and scheduling is the capacitated lotsizing problem with linked lotsizes and sequence-dependent setups (CLSD), first formulated by Haase (1996). Compared to the GLSP, the sequence of the lots within a macroperiod is not modeled by a predefined order of microperiods. Instead, it is described by a numbering of the products produced within the period – as it is done in a tour of a traveling salesman problem (TSP). However, in contrast to a TSP, the first and the last product of a macroperiod (tour) do not have to be equal. Furthermore, it has to be decided whether a product is produced at all within a macroperiod (i.e., whether it is part of the TSP tour). In the basic version of Haase (1996), subtour elimination constraints avoid a setup sequence from a product k back to itself within the same macroperiod. Thus, a product can be produced at most once per macroperiod. Later in Sect. 2.3.2, extensions of the CLSD will be presented, which allow to produce the same product several times within a macroperiod (and thus allow a violation of the triangle inequalities if technically necessary minimum lotsizes are respected). Haase (1996) uses a backward-oriented heuristic to solve the problem. The heuristic starts in the last macroperiod

T, creates a production plan for the current period and jumps to the previous period $T-1$ until the first period is reached. For the determination of the products and the associated lotsizes in a period various priority rules are used. Shim et al. (2011) develop an alternative heuristic for the model of Haase (1996). After generating an initial solution by first sequencing products and then determining the corresponding lotsizes, the solution is improved with a backward scheduling method. Afterwards, the solution is further enhanced with the help of movements from earlier to later periods in a forward scheduling method.

A model with overtime, sequence-dependent setup times but only product-dependent setup costs can be found in Laguna (1999). For the solution, lotsizing and scheduling is decomposed and period-overlapping setups are allowed. A TS heuristic changes the schedule in order to improve the solution.

In the single-stage, parallel-machine model of Clark and Clark (2000), multiple products can be produced per macroperiod, but a predefined number of N setups must take place per period and machine. For this reason, a binary variable is introduced which takes the value 0 if the n^{th} setup on machine m in period t occurs from product i to k. If less than N "real" setups are required, phantom setups are introduced, i.e., a phantom setup from product k to itself on machine m at the n^{th} position in period t with zero setup time is scheduled. A rolling horizon (RH) approach is applied to speed up the solution process. Clark (2003) extends the model for multiple production stages and multi-level product structures, whereby the products are uniquely assigned to a given number of work centers. Almeder et al. (2015) present two alternative model formulations for a problem similar to the one of Clark (2003). However, their models also consider sequence-dependent setup costs. Each machine can produce a predefined set of products. A product can be produced only on one machine. In order to allow "zero" lead times, the concrete starting time of the production is recorded per product and (macro)period. "Zero lead times" means that a successor product can be processed within the same period as its predecessor. The starting times are coordinated for all products within the same period. The first formulation ensures that a production may not start until the batch of the preceding product is finished. The second formulation allows lot-streaming, i.e., the production of the successor product may start while the predecessor's lot is still produced.

Another multi-level model formulation of the CLSD with a unique assignment of products to machines is proposed by Grünert (1998). This model also includes positive lead times which means that a product cannot be produced in the same macroperiod as its predecessor. Since macroperiods are rather long time buckets, this assumption appears to be unrealistic if the bill-of-materials comprises many levels. Grünert (1998) also presents a reformulation of the model based on echelon inventory. Lagrangean decomposition and linear programming are used to generate a feasible solution which is later improved by a TS heuristic.

Quadt and Kuhn (2005) consider a practical problem observed in the semiconductor industry. The production system comprises multiple stages, each with several parallel machines. However, there is a bottleneck stage which is modeled by a single-stage lotsizing and scheduling model. The overall problem is solved in three steps. In a first step, the lotsizing and scheduling model is solved heuristically for aggregate products. Instead of introducing a binary setup variable for each product-machine combination, the authors use integer setup variables denoting the number of machines which are set up for a given product. In the following steps, detailed schedules for the end products are constructed and propagated to the other non-bottleneck stages. Quadt and Kuhn (2009) propose a model for the same problem with sequence-independent setups. In a first step, the lotsizing problem is solved period by period with the help of a reduced model formulation including overtime. In a second step, the actual schedule is determined.

Gupta and Magnusson (2005) develop a single-machine model for a company producing sandpaper. They introduce new variables which, for example, define whether a product is scheduled as the first or the last product in a period, or whether at least one or exactly one product is produced in a period. Their heuristic solution approach consists of three steps: generating an initial solution, shifting quantities by means of certain rules and executing a greedy scheduling heuristic based on ascending setup costs. Finally, the production plan is refined, e.g., by combining production quantities of the same product from different periods.

Pochet and Wolsey (2006, p. 381) propose a model which sequences products in one macroperiod. The model can easily be adapted to consider multiple macroperiods. Reformulations are proposed.

Almada-Lobo et al. (2007) present an alternative model formulation of the CLSD which is motivated by a practical case of the glass container industry. Here, cycles without disconnected subtours are allowed. That means that the first product of a period can be produced a second time within this period (to efficiently use capacity if the setup already took place in the preceding period). Nevertheless, other products still can only be produced once per period and disconnected subtours are still forbidden. They show that their model needs less binary variables than the models of Clark and Clark (2000) and Gupta and Magnusson (2005). Furthermore, a tighter model formulation and valid inequalities are presented. A heuristic solution procedure is proposed that consists of multiple steps: first, a lot-for-lot strategy is applied which may lead to overtime. Secondly, products are scheduled and overtime is eliminated. Finally, the solution quality is improved by shifting production quantities backward/forward or by rescheduling to reduce setup or inventory holding costs.

Almada-Lobo and James (2010) develop two meta-heuristics for this problem. They present a variable neighborhood search with variable neighborhood descent heuristic and a TS heuristic. For this purpose, the problem is represented as a sequence of jobs, each carrying a predefined demand and a due date. The initial solution is generated by the heuristic of Almada-Lobo et al. (2007). James and Almada-Lobo (2011) extend the model for parallel machines and propose a combination of an iterative variable neighborhood search (INS) and a MIP-based approach (F&R) termed INSRF. Motivated by a practical case of a steel mill, Kwak and Jeong (2011) add a special setup time structure to the model of Almada-Lobo et al. (2007), whereby setup times depend on two adjacent products' differences in size. Kwak and Jeong (2011) decompose the overall problem. First, the lotsizing problem is solved with estimated setup times and costs and neglected sequence dependencies. Next, based on the resulting production quantities a schedule is generated. Menezes et al. (2011) also adapt the CLSD of Almada-Lobo et al. (2007). Again, cycles without disconnected subtours are allowed. However, since minimum lotsizes are considered, the triangle inequality may also be violated. Additionally, period-overlapping setups can occur, i.e., setup times (and minimum lotsizes) may be split across adjacent periods. This is realized by introducing a variable defining the amount of time which is still necessary to finish a setup and a new binary variable indicating if such a setup continues to the next period. A method is presented for identifying and removing disconnected subtours.

An extended formulation of the problem of Almada-Lobo et al. (2007) from the glass container industry is presented in Almada-Lobo et al. (2008). The production system consists of several furnaces which transfer melted glass to a set of parallel machines producing the glass containers. For technical reasons, these machines must be supplied continuously with melted glass. Standard VNS with stochastic and deterministic neighborhood changes and a reduced VNS (random generation of solutions) are used to solve the problem heuristically.

Clark et al. (2010) propose several models for the problem of an animal feed company described by Toso et al. (2009) (see Sect. 2.3.1). One model allows setup carry-overs and another one plans initial setups between periods with the help of dummy products. As some products have cleansing properties, the triangle inequality may be violated. A minimum production quantity constraint avoids unnecessary setups. An asymmetric traveling salesman problem (ATSP) solution method is applied. For this reason, an iterative subtour elimination is incorporated.

Based on the CLSD, Almeder and Almada-Lobo (2011) present an alternative formulation for the GLSP with common tools (see Sect. 2.3.1). They introduce additional variables, representing the starting times of the tool attachment and ending times of a tool exchange on a machine. A further binary variable defines the sequences of tool usages on different machines. Setup state and changeover variables are tool-dependent in contrast to the usual product-dependent decision variables. A numerical analysis shows that this type of CLSD is superior to the corresponding GLSP-variant in terms of solution time and quality.

Lang and Shen (2011) describe a problem concerning windshield interlayer production. In the considered company, some products can be substituted. For example, foils with large width can be used instead of foils with shorter width. For the end of the planning horizon, final inventory targets are given. An F&R and an F&O heuristic are used to solve large problem instances. In the F&O heuristic, different subproblems are created with time-, product- and substitute-based decomposition strategies. Pahl et al. (2011) consider the CLSD with perishability constraints.

The problem treated by Camargo et al. (2012) consists of two production stages. On the first stage, a common resource produces a pre-product which is supplied to the second stage with multiple parallel machines. Since only one pre-product can be processed at the same time, different new variables are introduced, e.g., defining the starting and ending times of batches on the common resource. In addition to the GLSP-based formulation, Camargo et al. (2012) propose a CLSD formulation and an alternative model which neglects the period-based time grid. For a soft drink production system, Ferreira et al. (2012) present an alternative large-bucket model formulation based on the ATSP with subtour elimination constraints. This formulation uses the valid inequalities of Almada-Lobo et al. (2007).

Amorim et al. (2013) compare different single-stage, parallel-machine formulations based on the models of Amorim et al. (2011) and of Erdirik-Dogan and Grossmann (2008) as well as on the large-bucket models of Almada-Lobo et al. (2007) and of Kopanos et al. (2011) (see Sect. 2.3.6). They propose a new formulation with major and minor setups. Guimarães et al. (2013) assume a predefined set of setup sequences to be given. Their model chooses between these sequences and simultaneously computes appropriate lotsizes. Non-triangular setups are possible. Minimum and maximum production quantities are considered. The authors propose a heuristic to construct the sequences. Another RH- and F&R-based heuristic generates solutions. Finally, the solution is improved by an F&O heuristic.

In Xiao et al. (2013), an extended formulation for the CLSD is presented for a problem in the semiconductor manufacturing industry. A single production stage comprising parallel heterogeneous machines is considered as the main bottleneck. Upstream production stages are indirectly included by introducing time windows which restrict the starting times of setups. Products can be produced on a predefined subset of machines. A preference for the underlying machine-product combination exists, but may be violated. F&O heuristics are developed for solving practical problems. An F&R heuristic is used for initialization. Xiao et al. (2015) present an alternative solution approach which combines a Lagrangean relaxation algorithm with a simulated annealing approach to the hybrid Lagrangean-simulated annealing-based heuristic (HLSA). The problem is decomposed into a lot-sizing and a parallel-machine scheduling problem within the LR method. The SA improves the solution of the scheduling problem and provides an upper bound for the sub-gradient optimization of the LR algorithm.

Clark et al. (2014) formulate a model which allows non-triangular setup times. Multiple lots of the same product can be produced per period. In order to avoid unnecessary setups and inappropriate cleaning operations, constraints on the minimum lotsize are used. New constraints are introduced to coordinate sequences, avoid subtours and facilitate backlogs. Guimarães et al. (2014) also allow multiple lots of a given product per period, but use a single-commodity-flow formulation for subtour elimination.

Maldonado et al. (2014) present three different model formulations for the soft drink industry problem of Ferreira et al. (2012). Only one production line is considered. The models differ w. r. t. the subtour elimination constraints included. The authors present, amongst others, an ATSP-based formulation as well as a new multi-commodity-flow formulation.

Tempelmeier and Copil (2016) consider a scarce setup resource in a single-stage, multi-machine production system. Since there is only one setup resource, setups and hence production plans must be coordinated across all machines. In a basic model, a unique assignment of products to machines exists. The authors also present extensions for parallel machines, batch production with cleaning processes between batches and shelf life restrictions, which have been observed in the food industry. Considering the limited capacity of the setup resource, an additional model formulation allows multiple setups per product and period. An F&R and an F&O heuristic are proposed to solve large instances of practical size.

An overview of the CLSD-based model formulations is given in Table 2.7. Almost all model formulations correspond to the typical CLSD-structure: each product can be produced once per period (*exoT* = *d:K*) and endogenous events are not limited by an internal grid (*endoS* = *fr*). All models consider either sequence-dependent setup times (*St* = *sd*) and/or sequence-dependent setup costs (*Sc* = *sd*). Also, the triangle inequality usually has to be kept in the basic model formulation (*Sc* = *sd*:Δk). However, there are a few models allowing a violation of the triangle inequality (*Sc* = *sd*:Δv) and the production of multiple lots of a product per period (*exoT* = *d:fr*). These comprise for instance Menezes et al. (2011), Guimarães et al. (2013) and Guimarães et al. (2014). In these cases, a minimum lotsize has to be produced (*Ls* = *min*). Tempelmeier and Copil (2016) allow multiple lots per period with respect to the scarce setup resource. Since the triangle inequality is kept, lotsizes can be continuous (*Ls* = *c*).

Table 2.7: Literature overview CLSD

References	BOM	Ps	M	Sc	Css	St	exoT	endoS	Ls	Industry	Heuristics/comments
Haase (1996)	1	1	1	sd:Δk	cs	–	d:K	fr	c	–	BA
Grünert (1998)	g	fr:cl	1	sd:Δk	cs	sd:max:p	d:K	fr	max	–	LD, TS
Laguna (1999)	1	1	1	si	cs	sd:max:p	d:K	fr	c	CGI	Decomposition, TS
Clark and Clark (2000)	1	1	pn	sd:Δk	cs	sd:max:p	d:K	fr	c	–	RH, F&R
Clark (2003)	g	fr:s	1	–	cs	sd:max:p	d:K	fr	c	–	RH, F&R
Gupta and Magnusson (2005)	1	1	1	sd:Δk	cs	sd:max:p	d:K	fr	c	CGI	Initialization, sequencing, improvement
Quadt and Kuhn (2005)	1	1	pi	sd:Δk	cs	sd:max:p	d:K	fr	c	SI	Decomposition
Pochet and Wolsey (2006, p. 381)	1	1	1	sd:Δk	ls	–	d:fr	fr	c	–	Reformulation
Almada-Lobo et al. (2007)	1	1	1	sd:Δk	cs	sd:max:p	d:K	fr	c	PI	Five-step-heuristic; MIP-solver
Almada-Lobo et al. (2008)	d	1	pn	sd:Δk	cs	sd:max:p	d:K	fr	c	PI	VNS; secondary resources
Quadt and Kuhn (2009)	1	1	pi	si	cs	si	d:K	fr	c	SI	Decomposition
Almada-Lobo and James (2010)	1	1	1	sd:Δk	cs	sd:max:p	d:K	fr	c	–	VNS and TS
Clark et al. (2010)	1	1	1	–	cs	sd:max:p	d:K	fr	min	AFI	Sequences with ATSP, then lotsizing
Almeder and Almada-Lobo (2011)	1	1	pn	sd:Δk	cs	sd:max:p	d:K	fr	c	SI	MIP-solver; secondary resources
James and Almada-Lobo (2011)	1	1	pn	sd:Δk	cs	sd:max:p	d:K	fr	max	–	INSRF
Kwak and Jeong (2011)	1	1	1	sd:Δk	cs	sd:max:p	d:K	fr	c	PI	Hierarchical integration of lotsizing and sequencing
Lang and Shen (2011)	1	1	1	sd:Δk	cs	sd:max:p	d:K	fr	c	CGI	F&R and F&O; product substitution
Menezes et al. (2011)	1	1	1	sd:Δv	cs	sd:fr	d:fr	fr	min	–	MIP-solver
Pahl et al. (2011)	1	1	1	sd:Δk	cs	sd:max:p	d:K	fr	c	CGI	MIP-solver; perishability
Shim et al. (2011)	1	1	1	sd:Δk	cs	–	d:K	fr	c	PI	Generation and improvement
Camargo et al. (2012)	d	1	pn	sd:Δk	cs	sd:max:p	d:K	fr	c	PI	MIP-solver; secondary resources
Ferreira et al. (2012)	d	1	pn	sd:Δk	cs	sd:max:p	d:K	fr	mima	BI	MIP-solver; secondary resources
Amorim et al. (2013)	1	1	pn	sd:Δk	cs	sd:max:p	d:K	fr	c	CGI	MIP-solver
Guimarães et al. (2013)	1	1	1	sd:Δv	cs	sd:max:p	d:fr	fr	mima	–	Decomp., F&R/F&O
Xiao et al. (2013)	1	1	pn	sd:Δk	cs	sd:max:p	d:K	fr	c	SI	F&R, F&O
Clark et al. (2014)	1	1	1	si	cs	sd:max:p	d:fr	fr	min	–	MIP-solver
Guimarães et al. (2014)	1	1	1	sd:Δv	cs	sd:max:p	d:fr	fr	min	–	MIP-solver
Maldonado et al. (2014)	1	1	1	sd:Δk	cs	sd:max:p	d:K	fr	mima	BI	MIP-solver; secondary resources
Tempelmeier and Copil (2016)	1	1	pi	sd:Δk	cs	sd:max:p	d:fr	fr	c	FI	F&R, F&O; secondary resources, perishability
Almeder et al. (2015)	g	fr:s:o	1	sd:Δk	cs	sd:max:p	d:K	fr	c	–	MIP-solver
Xiao et al. (2015)	1	1	pn	sd:Δk	cs	sd:max:p	d:K	fr	c	SI	HLSA

Numerous publications are based on practical cases (from the AFI, CGI, FI, PI or SI). These publications show that the consideration of secondary resources or bottleneck machines becomes more frequent, see for example Almada-Lobo et al. (2008), Almeder and Almada-Lobo (2011), Camargo et al. (2012), Ferreira et al. (2012), Maldonado et al. (2014) and Tempelmeier and Copil (2016). Substitution (Lang and Shen 2011) or perishability (Pahl et al. 2011 or Tempelmeier and Copil 2016) are included into some model formulations. Besides that, starting with Clark and Clark (2000), numerous models consider parallel machines ($M = pn$ or pi) both for theoretical or practical problems. For the CLSD, mainly exact or MIP-based solution approaches are used, see column *heuristics/comments*.

2.3.3 Proportional lotsizing and scheduling problem (PLSP)

The PLSP was introduced by Haase (1994) (see also Drexl and Haase 1995). In comparison to the GLSP and CLSD, the production of at most two different products per period is possible by allowing one setup at any time during a period. Thus, in terms of practical application, the length of such a period will usually be rather short. This marks the PLSP as a small-bucket model. Although Haase (1994) already included sequence-dependent setup costs in a model formulation, the basic model of Drexl and Haase (1995) does not consider setup times. It is solved using a backward-oriented, regret-based, biased random sampling method (BACKADD). The method solves the problem starting in the last period and moving back one by one period until the first period is reached. Randomized regrets are used to determine which product should be scheduled in a period. Thereby, the products are randomly selected with a probability proportional to the regrets depending on potential savings. The "randomized regret"-based heuristic is extended by Drexl and Haase (1996). They partition the parameter search space into subspaces via sequential analysis based on hypothesis testing. More advanced model variations of the PLSP including setup times or parallel machines, respectively, are also presented by Drexl and Haase (1995). Here, the modeling approach allows period-overlapping setups. The authors propose randomized measures to adapt BACKADD accordingly.

Drexl et al. (1995) and Kimms (1996b) extend the PLSP for a general, multi-level product structure and propose an RR-based solution heuristic. However, still only a single machine is considered. Kimms (1996a) compares a modification of these RR-based methods with a TS approach. In Kimms (1997a), an iterative two-phase procedure is proposed. A feasible production plan is constructed using a backward-oriented approach. Afterwards, demands are shifted, a new production plan is generated and compared to the currently best solution. The second phase is called demand shuffle (DS). Kimms (1997b, p. 31) extends the model to multiple production stages. He presents DS-based heuristics and numerical tests for the special case in which products are uniquely assigned to machines. In addition, Kimms (1997b, p. 60) proposes a multi-level PLSP model for parallel machines and a corresponding DS-based heuristic. Kimms and Drexl (1998) summarize the extensions of the PLSP and propose a backward-oriented RR-heuristic to solve the multi-level, parallel-machine problem. Finally, Kimms (1999) considers the model with a unique assignment of products to machines and shows that a new GA is superior to his earlier TS approach with respect to computation time and solution quality.

Chang et al. (2004) extend the model of Kimms (1999) to consider product life cycles. They assume that the capacity consumption depends on the phase of the life cycle. Phase-dependent weights are introduced to allocate the available capacity. In addition, setup learning effects are integrated by changing the setup costs over time. Cash flows are considered by discounting the setup and inventory holding costs with a rate of return per period and product. A GA with a backward-oriented procedure is used to solve the lotsizing and scheduling problem.

Wolsey (1997) proposes a general model formulation which includes product-dependent start-up and sequence-dependent setup costs and fulfills the characteristics of the CSLP (see Sect. 2.3.4) by allowing the setup for and production of one product per period. Different extensions considering changeovers are presented. Thereby, one extension allows the production of two products per period what corresponds to the PLSP.

Suerie (2005) concentrates on selected problem settings found in the chemical industry. These include, for example, minimum and maximum production quantities to initiate a chemical reaction or to ensure cleaning operations. Also, production quantities may consist of multiple fillings, so-called batches. Thus, the whole contiguous production run of a product, that may continue over several consecutive periods ("campaign"), must be recorded. To ensure this, the author introduces a product-specific campaign variable, which contains the cumulated production quantity of the product, starting after the last campaign, summing up to the current period. Suerie (2006) picks up this idea and presents two new model formulations. These enable the modeling of "period-overlapping" setups. Contrary to traditional formulations of the PLSP, a setup is not confined to a single period, but may run over multiple consecutive periods. In the first model formulation (POST1), a continuous variable similar to the campaign variable of Suerie (2005) is introduced. It cumulates the setup time from the first to the last period of a setup. A binary variable defines whether a setup has already been finished or still continues at the end of a period. The second model (POST2) introduces two new variables. They record the relative share of the overall setup time at the end of a period, on the one hand, or at the beginning of/within a period, on the other hand.

Pochet and Wolsey (2006, p. 378) present a PLSP model and propose a reformulation to strengthen the initial model. A model specialized for the process industry is presented by Pochet and Wolsey (2006, p. 470). Shutdown costs and cleaning times which occur after a lot is finished are considered instead of setup times and costs. The model formulation is further tightened.

Tempelmeier and Buschkühl (2008) present a model formulation for the PLSP with multiple machines inspired by a practical problem observed in the automobile industry. Each product is uniquely assigned to a predefined machine. However, setups are carried out by a single, common setup operator. Similar to Tempelmeier and Copil (2016), isolated lotsizing for each machine is not possible anymore as the setups must be coordinated across all machines to avoid overlapping of setups. A new continuous variable records the beginning of a setup on a machine in a microperiod. Another variable stores the visiting sequence of the setup operator for the machines. A simple plant location reformulation is proposed. A third model formulation differentiates between minor and major setups.

In order to avoid symmetry, Kaczmarczyk (2011) reformulates the single-stage, parallel-machine PLSP with respect to a problem of the electronics industry. He introduces integer variables instead of binaries. Different new constraints help to coordinate flow variables for different machines. Several model formulations are presented with setup variables or start-up and switch-off variables. Pahl and

Voß (2010) add perishability to the PLSP similar to the approaches for the GLSP and CLSD.

Stadtler (2011) proposes a formulation for the PLSP with a single machine but a multi-level, general product structure that allows zero lead times in order to tackle some selected problems of a pharmaceutical company. The author draws upon ideas from Suerie (2005) in order to coordinate the production sequence of products directly related in the BOM structure. Stadtler (2011) also extends the model for period-overlapping setup operations and uses modeling techniques of Suerie (2006). Finally, he adds batch-flow restrictions to the model. Stadtler and Sahling (2013) define two model formulations for the PLSP with general product structures, multiple serial machines and a unique assignment of a product to a machine. Similar to Stadtler (2011), period-overlapping setups are modeled with the help of two continuous variables defining the planned setup time at the end or at the beginning/within a period. A fixed lead time of one period is used in the first model, but no longer needed in the second formulation. Two – with respect to the bill-of-materials – consecutive products can be produced within the same period. F&R is used to create an initial solution, which is improved by F&O.

Table 2.8: Literature overview PLSP

References	BOM	Ps	M	Sc	Css	St	exoT	endoS	Ls	Industry	Heuristics/comments
Haase (1994)	1	1	1	sd:Δk	cs	–	d:2	fr	c	–	RR
Drexl and Haase (1995, p. 75)	1	1	1	si	cs	–	d:2	fr	c	–	RR
Drexl and Haase (1995, p. 81)	1	1	1	si	cs	si:fr:fr	d:2	fr	c	–	RM
Drexl and Haase (1995, p. 82)	1	1	pn	si	cs	–	d:2	fr	c	–	RM
Drexl et al. (1995)	g	1	1	si	cs	–	d:2	fr	c	–	RR
Drexl and Haase (1996)	1	1	1	si	cs	–	d:2	fr	c	–	RR, sequential analysis
Kimms (1996a)	g	1	1	si	cs	–	d:2	fr	c	–	RR, TS
Kimms (1996b)	g	1	1	si	cs	–	d:2	fr	c	–	RR
Kimms (1997a)	g	1	1	si	cs	–	d:2	fr	c	–	DS
Kimms (1997b, p. 31)	g	fr:cl	1	si	cs	–	d:2	fr	c	–	DS; intensive test
Kimms (1997b, p. 60)	g	fr:cl	pn	si	cs	–	d:2	fr	c	–	DS; rudimental tests
Wolsey (1997)	1	1	1	sd:Δk	cs	si:max:1	d:2	fr	c	–	Unit flow formulation
Kimms and Drexl (1998)	g	fr:cl	pn	si	cs	–	d:2	fr	c	–	RR
Kimms (1999)	g	fr:cl	1	si	cs	–	d:2	fr	c	–	GA
Chang et al. (2004)	g	fr:cl	1	si	cs	–	d:2	fr	c	–	GA
Suerie (2005)	1	1	1	si	cs	si:max:1	d:2	fr	dm	CI	MIP-solver
Pochet and Wolsey (2006, p. 378)	1	1	1	sd:Δk	ls	–	d:2	fr	c	–	Reformulation
Pochet and Wolsey (2006, p. 470)	g	2:s	pn	si	ls	si:max:1	d:2	fr	min	PI	Reformulation
Suerie (2006)	1	1	1	si	cs	si:fr:fr	d:2	fr	c	CI	MIP-solver
Tempelmeier and Buschkühl (2008)	1	1	1	si	cs	si:max:1	d:2	fr	c	AI	MIP-solver; secondary resources
Pahl and Voß (2010)	1	1	1	si	ls	si:max:1	d:2	fr	c	CGI	MIP-solver; perishability
Stadtler (2011)	g	1	1	si	cs	si:max:fr	d:2	fr	dm	PhI	MIP-solver
Kaczmarczyk (2011)	1	1	pi	si	cs	si	d:2	fr	c	EI	MIP-solver
Stadtler and Sahling (2013)	g	fr:s:o	1	si	cs	si:fr:fr	d:2	fr	c	–	F&R as initialization then F&O

Table 2.8 summarizes the presented formulations for the model PLSP. The number of models is smaller compared to the number of big bucket formulations. As the column *exoT* shows, a production of at most two products is allowed in all models (*d*:2), which corresponds to the assumptions of the

PLSP. Endogenous events can be scheduled freely (*endoS* = *fr*). Table 2.8 shows that setup costs are mainly sequence-independent (*Sc* = *si*). Positive (also sequence-independent) setup times (*St* = *si*) are predominantly considered in more recent publications, starting with Suerie (2005). Almost all models considering setup times are based on practical cases (AI, CGI, CI, EI, PhI, PI) which proves the relevance of these types of constraints. Drexl and Haase (1995, p. 81), Suerie (2006), Stadtler (2011) and Stadtler and Sahling (2013) present models which additionally allow period overlapping setups (*St* = *si:fr:fr* or *St* = *si:max:fr*). Furthermore, Table 2.8 shows that many PLSP-based model formulations consider a general and multi-level bill-of-materials structure (*BOM* = *g*). Lotsizes of Suerie (2006) and Stadtler (2011) additionally comprise multiple batches (*Ls* = *dm*). The first publications for PLSP-formulations use RR- and DS-based solution methods. However, more recent papers rather rely on standard MIP-solvers.

2.3.4 Continuous setup lotsizing problem (CSLP)

The CSLP is more restrictive than the PLSP. It allows to produce at most a single product within a period. Since a period's capacity does not need to be completely exhausted, "continuous" lotsizes extending over multiple periods are possible. The CSLP is qualified as a small-bucket model since this kind of models have only a small number of different products within each period. Karmarkar and Schrage (1985) propose the first multi-item CSLP formulation. They consider a single machine and incorporate sequence-independent setup costs and time-dependent production costs. The model is solved by a B&B approach, in which Lagrangean relaxation is used to determine lower bounds and to generate subproblems. The subproblems are solved by dynamic programming (DP). Dynamic programming means that a problem is divided into subproblems. The optimal solutions of the subproblems are recursively combined to get the optimal solution of the overall problem. The authors come to the conclusion that the results are not very encouraging. Pochet and Wolsey (1991) add cuts to the model which lead to slightly improved results.

A model formulation with non-identical, parallel lines and sequence-dependent setup costs is proposed by de Matta and Guignard (1995). They solve the model using Lagrangean relaxation. As Table 2.9 shows, the time structure differs from the original one of Karmarkar and Schrage (1985). The reason is that demand and inventory holding costs are modeled on a macroperiod basis as it has already been done by Fleischmann (1990) for the DLSP (see Sect. 2.2.2). Since the microperiods used for production have fixed lengths, the model is not classified as a GLSP. Compared to the DLSP of Fleischmann (1990), the endogenous state changes are different. Because of the all-or-nothing assumption, the DLSP fixes them to the internal time grid (see Table 2.3), which is not the case in the CSLP formulation of de Matta and Guignard (1995).

Wolsey (1997) proposes a formulation with sequence-dependent setup times. The objective function minimizes the sum of the following time-dependent costs: production costs, holding costs, sequence-dependent and -independent setup costs and costs which occur if the machine is in the state to produce a certain product. The latter costs do not only occur if the machine is idle, like the standby costs in the GLSP (Meyr 1999, Chap. 4). They also occur while production takes place. A reformulation using a “flow conservation of the setup state” is introduced and further tightened. Numerical tests are not presented.

Vanderbeck (1998) considers a CSLP formulation with sequence-independent setup times. Column generation (CG) and a cutting plane algorithm are combined to solve the problem. In general, CG approaches define a master problem with a reduced number of variables and one or multiple subproblems. The subproblems decide which variables improve the solution of the master problem and are added to this problem. The single-item subproblems of Vanderbeck (1998) which arise because initial stocks are treated as decision variables are solved by dynamic programming. Another CSLP formulation with sequence-independent setup costs and further costs which occur while the machine is prepared for production is presented by Constantino (2000). The author incorporates real-life aspects such as minimum lotsizes and backlogging. Several valid inequalities are derived. Finally, a branch&cut (B&C) algorithm is applied to solve the problem. Compared to B&B, the computation times are reduced significantly.

An injection molding process is considered by Dastidar and Nagi (2005). Production can take place on parallel, heterogeneous workcenters, but requires additional resources such as conveyors or grinders. A binary parameter indicates if such a resource is necessary to produce a certain product on a certain workcenter or not. A decomposition algorithm groups the workcenters and generates subproblems (two-phase workcenter-based decomposition strategy). This leads to a much better solution performance than a monolithic approach.

Pochet and Wolsey (2006, p. 173) present a CSLP model for a machine which bottles cleaning liquids. They propose valid inequalities and solve the model using F&R and F&O.

A practical case of a foundry is considered by de Araujo et al. (2007). Multiple raw materials are processed consecutively on a single machine. Changeovers from one raw material to another cause sequence-dependent setup costs and times. The processing of each raw material simultaneously leads to several end products. The authors state that the model formulation was inspired by the GLSP. However, the length of the microperiods is predetermined. Thus, similar to de Matta and Guignard (1995), we prefer to classify it as a CSLP. A rolling horizon approach combined with an F&R heuristic solves the problem heuristically. De Araujo et al. (2008) adapt the formulation for sequence-independent setup costs and times. De Araujo and Clark (2013) convert the model from de Araujo et al. (2007) into a simple plant location (SPL) formulation to apply the same solution approaches.

A quite general State-task-network model for the process industry is proposed by Gaglioppa et al. (2008). In a basic version, a single processing unit exists which can carry out different tasks of arbitrary lengths. Each task may handle multiple input materials and can lead to multiple end products. The model is extended to cover multi-stage parallel processing units. The authors formulate valid inequalities and execute some numerical tests.

A problem from the glass container production is considered by Almada-Lobo et al. (2010). The production system consists of a furnace supplying multiple parallel molding machines which process the melted glass. This system is represented as a single-stage model considering the melted glass as an additional scarce resource which limits the production quantities of the molding machines. For each molding machine and period, it is possible to determine the output rate by deciding about the number of active mold cavities. First, the authors present a formulation in which this number of cavities has to be integer, then they relax the model by allowing continuous lotsizes. The furnace can be inactive, but only at the end of the planning horizon. If the furnace is active, the complete capacity of the furnace should be used. Otherwise, penalty costs are incurred. The model is converted into a formulation with "flow conservation of the setup state" and decomposed into subproblems by Lagrangean relaxation. Toledo et al. (2013) consider the same problem, but only allow discrete lotsizes in form of complete batches. However, these are not fixed to microperiod boundaries as it would be the case in a DLSP formulation. The authors propose a GA combined with a heuristic to determine the optimal number of mold cavities.

Pahl and Voß (2010) formulate a DLSP model considering perishability (see Sect. 2.3.5). This model is relaxed into a CSLP formulation by allowing continuous lotsizes. However, the CSLP is only an intermediate model and it is directly converted into the PLSP formulation mentioned in Sect. 2.3.3.

Table 2.9 gives an overview of the CSLP based models. Multi-stage bill-of-materials (*BOM* = *d* or *g*) are only considered by de Araujo et al. (2007), de Araujo et al. (2008), Gaglioppa et al. (2008) and de Araujo and Clark (2013). Column *M* shows that most of the models consider only a single machine. Exceptions are de Matta and Guignard (1995), Dastidar and Nagi (2005), Gaglioppa et al. (2008), Almada-Lobo et al. (2010) and Toledo et al. (2013). There exist models which consider sequence-dependent setup costs (*Sc* = *sd*) presented for example by de Matta and Guignard (1995) and Wolsey (1997), while other models proposed e.g. by Karmarkar and Schrage (1985) and Pochet and Wolsey (1991) consider sequence-independent setup costs. Starting from 1997, nearly all models respect setup times (see column *St*).

Table 2.9: Literature overview CSLP

References	BOM	Ps	M	Sc	Css	St	exoT	endoS	Ls	Industry	Heuristics/comments
Karmarkar and Schrage (1985)	1	1	1	si	cs	–	d:1	fr	c	–	B&B+LR
Pochet and Wolsey (1991)	1	1	1	si	cs	–	d:1	fr	c	–	Cutting planes
de Matta and Guignard (1995)	1	1	pn	sd:Δk	cs	–	d:K	d:1:fr	c	PhI	LR
Wolsey (1997)	1	1	1	sd:Δk	cl	sd:max:1	d:1	fr	c	–	Reformulation; not solved
Vanderbeck (1998)	1	1	1	si	cs	si:max:1	d:1	fr	c	–	CG+ cutting planes
Constantino (2000)	1	1	1	si	cl	si:max:1	d:1	fr	min	–	B&C
Dastidar and Nagi (2005)	1	1	pn	sd:Δk	cs	sd:max:1	d:1	fr	c	CGI	Decomposition; secondary resources
Pochet and Wolsey (2006, p. 173)	1	1	1	si	ls	–	d:1	fr	mima	CGI	Valid inequalities, F&R, F&0
de Araujo et al. (2007)	d	1	1	sd:Δv	cs	sd:max:1	d:fr	d:1:fr	dm	–	RH+F&R+LS
de Araujo et al. (2008)	d	1	1	si	cs	si:max:1	d:K	d:1:fr	dm	PI	RH+F&R+LS
Gaglioppa et al. (2008)	g	fr:cl:c	pn	si	cs	–	d:1	fr	c	PI	Valid inequalities, STN
Almada-Lobo et al. (2010)	1	1	pn	sd:Δv	cs	sd:max:1	d:1	fr	mima	PI	LR; secondary resources
Pahl and Voß (2010)	1	1	1	si	ls	si:max:1	d:1	fr	c	CGI	Not solved; perishability
de Araujo and Clark (2013)	d	1	1	sd:Δv	cs	sd:max:1	d:fr	d:1:fr	dm	–	RH+F&R+LS, SPL formulation
Toledo et al. (2013)	1	1	pn	sd:Δv	cs	sd:max:1	d:1	fr	dm	PI	GA; secondary resources

In most cases the time structure is CSLP-typical: exogenous events are considered using a discrete time grid of microperiods (*exoT* = *d*). Endogenous events are not bound to a time grid (*endoS* = *fr*). The number of lots per microperiod is limited to at most one (*exoT* = *d*:1). Some of the models (de Matta and Guignard 1995; de Araujo et al. 2007; de Araujo et al. 2008; de Araujo and Clark 2013) consider demand on basis of macroperiods which consist of a predefined number of microperiods. In fact, this results in two time grids: one macroperiod time grid with the possibility of producing *K* or an arbitrary number of lots per macroperiod (*exoT = d:K* or *d:fr*), and one microperiod time grid which allows one lot per microperiod but does not further limit the starting and ending times of lots (*endoS* = *d*:1:*fr*). These models are not classified as GLSP since the lengths of microperiods are defined in advance. As it is shown in column *Ls*, lotsizes are continuous (e.g. Karmarkar and Schrage 1985) or discrete (e.g. de Araujo et al. 2007) and only seldom limited to upper or lower bounds (e.g. Pochet and Wolsey 2006, p. 173). Some models are directly based on industrial scenarios. For instance, Dastidar and Nagi (2005) consider a case from the consumer goods industry. Solution approaches are e.g. LR (e.g. de Matta and Guignard 1995), cutting planes (e.g. Vanderbeck 1998) or RH heuristics (e.g. de Araujo et al. 2007).

2.3.5 Discrete lotsizing and scheduling problem (DLSP)

The term DLSP was first used by Fleischmann (1990). It emphasizes that the resulting lotsizes are always multiple integers of the period capacity. This fact is founded in the all-or-nothing assumption. Lasdon and Terjung (1971) present a model formulation for a tire manufacturer based on this assumption. They consider multiple products with dynamic demand which are produced on parallel, identical machines. Setup times are neglected. A special restriction is that on each machine a die is needed for production. These dies are only available in a limited number. Furthermore, the model is extended to respect limited personnel capacity. The personnel is necessary for mounting and dismounting the dies. The basic model is solved using column generation. Eppen and Martin (1987) use variable redefinition to reformulate this problem. Furthermore, they apply LP relaxation to determine better bounds for a B&B procedure.

Schrage (1982) presents a model formulation for a single machine with sequence-dependent setup costs. He also proposes an LP relaxation as a solution approach. However, numerical experiments are not presented.

Liberatore and Miller (1985) consider a problem from a tile company. They propose a hierarchical planning model which includes simultaneous lotsizing and scheduling for parallel, heterogeneous production lines. In addition, they present a reformulation without inventory variables. The model is solved using LP relaxation. De Matta and Guignard (1994a) present a solution approach based on Lagrangean relaxation for the same problem. De Matta and Guignard (1994b) adapt the model formulation with respect to sequence-dependent setup times. However, these setup times are indirectly modeled as production losses, measured in units of the products produced. They also apply LR and use real data from a tile manufacturer.

As already mentioned in Sect. 2.2.2, Fleischmann (1990) presents a DLSP formulation considering a single machine. He eliminates the inventory variables and expresses the cumulated demand by the corresponding number of production periods. This results in a much simpler formulation which facilitates to model demand and inventory costs associated with macroperiods. Nevertheless, there still exists a more detailed endogenous time structure on the basis of microperiods to represent the schedule. The advantage of this mix of periods is that reasonable plans can be constructed despite of the restricting all-or-nothing assumption. Fleischmann (1990) solves the DLSP using a B&B procedure combined with LR. A linear description of Fleischmann's basic DLSP is given by van Hoesel and Kolen (1994). Furthermore, the authors propose a DP approach. Both approaches are derived using a shortest path formulation of the DLSP.

A model with conservation of the setup state in idle periods is presented by Magnanti and Vachani (1990). For this purpose they differentiate between binary variables which indicate setup states, changeovers and production. It is possible to realize a loss of a setup state by omitting the setup state variables. The authors also propose an extension to include sequence-independent setup times. These can last an integer multiple of the period length. Campbell (1992) presents a model for identical, parallel machines with sequence-independent setup times and costs. Here, setup times are limited to at most one microperiod. LR is proposed as a solution approach.

Cattrysse et al. (1993) use a model formulation which considers sequence-independent setup times. They present a heuristic based on dual ascent and column generation methods. For this purpose, they reformulate the model as a set partitioning problem. Brüggemann and Jahnke (1994) introduce a model for a two-stage production process. A lot can only be transferred to the second production stage after its completion on the first production stage. A simulated annealing (SA) algorithm solves the model heuristically. Later Brüggemann and Jahnke (2000a)[9] adapt the single-stage model of Cattrysse et al. (1993) for a similar transfer strategy: only after finishing the complete lot, the quantity produced gets available on stock. Therefore, a new variable is introduced, which indicates the last production period of a lot. Again, the model is solved using SA.

De Matta (1994) compares Lagrangean decomposition with Lagrangean relaxation as solution approaches for a single-machine problem with sequence-independent setup costs. A model with sequence-dependent setup costs is presented by Fleischmann (1994). His solution method is based on a reformulation as a traveling salesman problem with time windows (TSPTW) and on LR. Salomon et al. (1997) extend this DLSP for sequence-dependent setup times. Inspired by Fleischmann (1994), they reformulate their model as a TSPTW and solve it with the help of the dynamic programming approach of Dumas et al. (1995).

Göthe-Lundgren et al. (2002) formulate a quite general, multi-stage model and apply it to an oil refinery. The production system consists of a distillation unit and two different hydro-treatment units. It is possible to run the three production units in different modes. A "run-mode" of a production unit defines the input materials consumed and the output products generated. Changeovers between run-modes can be interpreted as setups because they disturb the production process. Extensions are also considered. Examples are variable yield levels for run modes, which can be chosen by additional variables, or common resources, which restrict the choice of run-modes. An exact and a heuristic approach are proposed to solve the problem. Persson et al. (2004) modify the model to additionally consider sequence-dependent changeover costs when switching the run-mode. Here, special variants

[9]Note the erratum Brüggemann and Jahnke (2000b).

of TS are applied to solve the model.

Wolsey (2002) reviews reformulations, valid inequalities and algorithms for single-item lotsizing problems. As we have seen above, many solution approaches for the DLSP (like, e.g., LR) use decomposition of the multi-item model into simpler single-item models. Thus, Wolsey's findings are relevant for the DLSP, too. Miller and Wolsey (2003) present different DLSP extensions which consider backlogging, initial inventory variables or both of these characteristics. Furthermore, they incorporate safety stocks and piecewise-linear holding costs. They reformulate their models to get tight formulations and test them using data taken from practice. A so-called "minimal model" (c.f., Brüggemann et al. 2003a) of the DLSP is constructed by Brüggemann et al. (2003b). They introduce one parameter which denotes periods with positive demand and a second parameter which stores the related demand quantities. Thus it is possible to construct a model which gets along without time-indexed variables. The authors use this conceptual model for complexity discussions, but they do not solve it.

The multi-stage model presented by Muramatsu et al. (2003) takes general product structures and parallel, non-identical machines into account. Integer lead times for moving the products from one machine to another are considered. LR and item-based decomposition are used to create simpler subproblems. These subproblems are solved by DP. Jans and Degraeve (2004) present another single-stage, parallel-machine DLSP formulation for a tire manufacturer. Tires are cured in heaters. For each curing process an additional mold is used. Both heaters and molds are scarce resources. The main part of a setup operation is spent to warm up the molds. Because of the personnel's availability, this can only happen at the beginning of a day's first shift. To prevent the molds from cooling, production must always run at full capacity (all-or-nothing). The model is solved with a column generation procedure.

Pochet and Wolsey (2006, p. 374) formulate several DLSP models with different characteristics such as backlogging or sequence-dependent setup costs. They propose different reformulations to get tighter formulations. Another DLSP model is presented by Pochet and Wolsey (2006, p. 466). They propose tighter formulations and discuss different accountings of setup costs during idle times if product i is produced before the idle time and product k afterwards.

Gicquel (2008, p. 36) considers a single-machine DLSP and notes that planners often group changeovers based on technical considerations and then assign product combinations to these changeover types with type-specific changeover costs. Grouping product changeovers in this manner reduces the size of the DLSP model. Gicquel (2008) uses cutting plane generation strategies to solve the aggregated model based on changeover types and the standard model formulation without changeover grouping. Thereby, the standard model performs better because of the tighter formulation. Gicquel et al. (2009a) apply a similar principle. However, they exploit the setup characteristics in a more realistic way. If each product has different physical attributes, like color or dimension, then sequence-dependent changeover costs can be considered on this attribute level. If several attributes are changed at the same time, then the corresponding changeover costs sum up. Because of this aggregation the number of changeover variables is reduced. Gicquel et al. (2009a) benchmark their approach with a product-dependent formulation. They add valid inequalities to both models. Here, in many cases, the new formulation performs better. A tight formulation for the DLSP with additional sequence-dependent changeover times is proposed by Gicquel et al. (2009b). They adapt the approach of van Eijl and van Hoesel (1997), referred to by Wolsey (2002), and introduce valid inequalities.

Gicquel et al. (2011) present a DLSP formulation for parallel, identical machines and adapt the valid inequalities of van Eijl and van Hoesel (1997) for this problem. A cutting plane generation procedure is proposed, which incorporates these inequalities. Gicquel et al. (2012) also consider identical, parallel machines. Integer instead of binary variables help to identify the setup states of the machines and the changeovers. This aggregate planning of the machines improves the solution performance. The aggregated formulation is further strengthened by various approaches. The resulting models are compared with each other with the help of numerical experiments. A quadratically constrained, quadratic binary programming formulation of the DLSP is given by Gicquel et al. (2014). This formulation is semidefinitely relaxed what leads to stronger lower bounds than linear relaxations.

As mentioned before, Pahl and Voß (2010) also propose a DLSP formulation considering deterioration and perishability. Computational tests show that the PLSP formulation is superior to the DLSP with perishability constraints. They come to the conclusion that the DLSP is less useful in case of deterioration and perishability because the all-or-nothing assumption leads to too high stock levels.

Supithak et al. (2010) assume different demands for different products with different, probably[10] discrete due dates which have to be fulfilled by a single machine. Furthermore, the model considers sequence-independent setup costs, holding costs and costs for tardiness. An order can be satisfied by several lots. Different heuristics are proposed to solve small and large problem instances.

Gicquel and Minoux (2015) consider a single-machine DLSP formulation with "flow conservation of the setup state". The authors propose multi-product valid inequalities. The separation problem is solved using exact and heuristic algorithms.

The aforementioned models are summarized in Table 2.10. Most of the models consider a single machine ($M = 1$). Until 2004, only a few models considered sequence-dependent setup costs ($Sc = sd$). Examples are Schrage (1982) and Fleischmann (1994). The other models merely respect sequence-independent setup costs ($Sc = si$). Some models consider setup times (see column *St*). See for example Magnanti and Vachani (1990), Campbell (1992) and Gicquel et al. (2009b). The time structure of most of the models is DLSP-typical: exogenous events are fixed to a microperiod time grid. At most one lot is allowed per microperiod ($exoT = d{:}1$). Endogenous state changes are also fixed to the external time grid ($endoS = fi$). However, in some models (e.g. Fleischmann 1990 and Brüggemann and Jahnke 1994), microperiods are aggregated to macroperiods to create an external time grid with longer periods. In these cases, K lots are allowed per macroperiod ($exoT = d{:}K$). Endogenous events are fixed to the microperiod structure ($endoS = d{:}1{:}fi$). Based on the all-or-nothing assumption, all lotsizes are integer multiples of a predefined batch size ($Ls = dm$). Some models are based on industrial applications. Examples are Lasdon and Terjung (1971), Liberatore and Miller (1985), Jans and Degraeve (2004) and Persson et al. (2004). Solution approaches mainly rely on LP relaxation (e.g. Eppen and Martin 1987), LR (e.g. de Matta 1994), DP (Campbell 1992) and LD (Muramatsu et al. 2003).

[10]The model formulation is incomplete, but an example is mentioned.

Table 2.10: Literature overview DLSP

References	BOM	Ps	M	Sc	Css	St	exoT	endoS	Ls	Industry	Heuristics/comments
Lasdon and Terjung (1971)	1	1	pi	si	ls	–	d:1	fi	dm	PI	CG; secondary resources
Schrage (1982)	1	1	1	sd:Δv	cs	–	d:1	fi	dm	–	LP relaxation
Liberatore and Miller (1985)	1	1	pn	si	ls	–	d:1	fi	dm	PI	LP-relaxation
Eppen and Martin (1987)	1	1	pi	si	ls	–	d:1	fi	dm	PI	Variable redefinition, LP-relaxation; secondary resources
Fleischmann (1990)	1	1	1	si	ls	–	d:K	d:1:fi	dm	–	B&B+LR
Magnanti and Vachani (1990)	1	1	1	si	cs	–	d:1	fi	dm	–	B&B+cutting planes
Magnanti and Vachani (1990)	1	1	1	si	ls	si:d:fr	d:1	fi	dm	–	Not solved
Campbell (1992)	1	1	pi	si	cs	si:max:1	d:1	fi	dm	–	LR+DP
Cattrysse et al. (1993)	1	1	1	si	ls	si:d:fr	d:1	fi	dm	–	Set partitioning problem+dual ascent+CG
Brüggemann and Jahnke (1994)	g	2:s:c	1	si	ls	si:d:fr	d:K	d:1:fi	dm	–	SA; non-linear formulation
de Matta (1994)	1	1	1	si	ls	–	d:1	fi	dm	–	LR, LD
de Matta and Guignard (1994a)	1	1	pn	si	ls	–	d:1	fi	dm	PI	LR
de Matta and Guignard (1994b)	1	1	pn	sd:Δv	ls	sd:max:1	d:1	fi	dm	PI	LR
Fleischmann (1994)	1	1	1	sd:Δv	ls	–	d:fr	d:1:fi	dm	–	TSPTW reformulation+LR
van Hoesel and Kolen (1994)	1	1	1	si	ls	–	d:1	fi	dm	–	Linear reformulation, DP
Salomon et al. (1997)	1	1	1	sd:Δv	ls	sd:d:fr	d:1	fi	dm	–	TSPTW reformulation+DP
Brüggemann and Jahnke (2000a)	1	1:c	1	si	ls	si:d:fr	d:K	d:1:fi	dm	–	SA
Göthe-Lundgren et al. (2002)	g	fr:cl	pn	si	ls	–	d:1	fi	dm	PI	Valid inequalities, TS; secondary resources
Brüggemann et al. (2003b)	1	1	1	si	cs	–	d:K	d:1:fi	dm	–	Not solved; conceptual model for complexity discussion
Miller and Wolsey (2003)	1	1	1	si	ls	–	d:1	fi	dm	–	Tight reformulations
Muramatsu et al. (2003)	g	fr:cl	pn	si	ls	si:d:fr	d:1	fi	dm	–	LD, DP
Jans and Degraeve (2004)	1	1	pn	si	ls	si:max:1	d:1	fi	dm	PI	CG; secondary resources
Persson et al. (2004)	g	fr:cl	pn	sd:Δv	ls	–	d:1	fi	dm	PI	TS; secondary resources
Pochet and Wolsey (2006, p. 374)	1	1	1	sd:Δv	ls	–	d:1	fi	dm	–	Reformulation
Pochet and Wolsey (2006, p. 466)	1	1	1	sd:Δv	cs	–	d:1	fi	dm	–	Reformulation
Gicquel (2008, p. 36)	1	1	1	sd:Δv	cs	–	d:1	fi	dm	–	Cutting planes; changeover types
Gicquel et al. (2009a)	1	1	1	sd:Δv	ls	–	d:1	fi	dm	–	Cutting planes; products described by attributes
Gicquel et al. (2009b)	1	1	1	sd:Δv	ls	sd:d:fr	d:1	fi	dm	–	Valid inequalities
Pahl and Voß (2010)	1	1	1	si	ls	si:max:1	d:1	fi	dm	CGI	MIP-solver; perishability
Supithak et al. (2010)	1	1	1	si	cs	–	d:1	fi	dm	–	BA, GA
Gicquel et al. (2011)	1	1	pi	si	ls	–	d:1	fi	dm	–	Cutting planes
Gicquel et al. (2012)	1	1	pi	si	cs	–	d:1	fi	dm	–	Aggregation+cutting planes
Gicquel et al. (2014)	1	1	1	sd:Δv	ls	sd:d:fr	d:1	fi	dm	–	Semidefinite relaxation+valid inequalities; quadratic formulation
Gicquel and Minoux (2015)	1	1	1	sd:Δv	ls	–	d:1	fi	dm	–	Valid inequalities

2.3.6 Other models

Different other models can be found in the literature which we have not been able to clearly relate to one of the model types of Sects. 2.3.1 - 2.3.5. Many of these publications are closer to the scheduling field. Thus, the focus often rather lies on scheduling operations on continuous time scales and coordinating their starting and ending times than on solving a lotsizing problem. Nevertheless, the papers consider, formulate or solve a lotsizing and scheduling model according to the definition of Sect. 2.1 (more or less) simultaneously. Table 2.11 provides a summary of such problems.

Aras and Swanson (1982) propose a model for a single machine. Holding costs are calculated in a very detailed manner, even differing within a period. The consequence is that the lot sequence of a period has an impact on the total costs although setup costs are not considered. A backward-oriented heuristic is used to solve problem instances taken from a bearing company.

A model considering a packing line in the food processing industry is proposed by Smith-Daniels and Smith-Daniels (1986). They distinguish between major setups, which occur if a changeover from one setup family to another setup family takes place, and minor setups, which occur for a changeover from one product to another product of the same family. In each macroperiod, products of just one setup family are allowed. Smith-Daniels and Ritzman (1988) propose a model for serially arranged production stages and serial bills-of-materials.

A model for parallel, non-identical machines and sequence-dependent setup costs and times is formulated by Baker and Muckstadt Jr. (1989). They present the CHES problems, a set of five practical problems which have been collected by Chesapeake Decision Sciences. A part of the setup costs violate the triangle inequalities. Kang et al. (1999) propose a "sequence splitting" model and a B&B heuristic to solve these problems. Although possible in principle, setup times are not taken into account. However, their idea to split a setup sequence within a macroperiod into sub-sequences, which contain each product at most once, allows to handle violated triangle inequalities in a quite flexible manner. The goal is to maximize the contribution margin. Meyr (2002) uses the CHES instances to compare the performance of his GLSPPL heuristics with the results of Kang et al. (1999).

Heuts et al. (1992) consider a case of the process industry in which different products are manufactured in a reactor before they are stored in tanks. For successful production, the reactor must be completely filled (batch production). Furthermore, the tank capacities are limited. The authors propose two heuristics to solve this problem.

Dauzère-Pérès and Lasserre (1994) present a combination of a lotsizing model and a job-shop model. For solving the problem, the authors iterate between a lotsizing model with a fixed production sequence and a scheduling model with given lotsizes. Different solution approaches are presented for the scheduling problem including a shifting bottleneck procedure and a priority-rule-based dispatching heuristic. In Dauzère-Pérès and Lasserre (2002), a multi-level capacitated lotsizing model without setup times is presented which includes scheduling issues. Urrutia et al. (2014) proceed in a similar way to solve an integrated lotsizing and job-shop scheduling problem. A Lagrangean heuristic with a fixed sequence and relaxed capacities is used for solving the lotsizing problem. A TS heuristic improves the sequences. Wolosewicz et al. (2015) pick up the underlying problem with setup times, too.

They propose an additional model formulation generating a production plan based on a given sequence of operations on a machine. A Lagrangean relaxation approach is used to solve the problem.

Sikora et al. (1996) consider a problem in which different jobs with individual due dates are processed on serially arranged machines. Unfortunately, a complete model formulation is missing. A modified silver/meal heuristic in combination with a sequencing algorithm is proposed to solve the problem. Sikora (1996) uses a GA to tackle the same planning problem.

A model based on a continuous time scale is introduced by Jordan and Drexl (1998). Different jobs must be scheduled on a single machine. It is possible to consolidate jobs to save sequence-dependent setup costs and times. A B&B approach is proposed.

Haase and Kimms (2000) present a lotsizing model which considers predefined sequences calculated with a TSP algorithm. The model represents a problem of a German company manufacturing printing machines. It is solved by simultaneously deciding which sequence is the most efficient one and which production quantity should be produced. During that process the Wagner-Whitin property is kept. Branch&bound is proposed as a solution method. Kovács et al. (2009) reformulate the model and improve the pre-processing of sequence generation with DP.

Erdirik-Dogan and Grossmann (2008) propose a single-stage scheduling model with parallel machines. It comprises an inventory balance equation, includes a profit-oriented objective function and divides periods into slots. Products are assigned to slots. Binary variables coordinate transitions between adjacent slots. Furthermore, starting and ending times of slots are recorded and coordinated over the machines. The problem is decomposed and iteratively solved with multiple steps: first, the model is relaxed. Here, the focus lies on assigning and sequencing the products. An upper bound is provided for the profit. In a second step, the original problem is solved ignoring unassigned products. The new solution is compared to the upper bound.

Mateus et al. (2010) extend the CLSP for backordering and combine it with the model of Rocha et al. (2008) which treats the parallel-machine scheduling problem. They interpret lots as jobs and coordinate the starting times and sequences with additional variables, which associate products to periods and machines. Since the model is hard to solve, they use decomposition. First, a lotsizing problem is solved to optimality. Next, the lots are decomposed into jobs, which are then sequenced by a greedy randomized adaptive search procedure (GRASP).

Mohammadi et al. (2010b) and Mohammadi et al. (2010a) present a lotsizing and scheduling model for a flow-shop. A lot can be transferred from one machine to the next only after it has been completely finished. Exactly K (number of products) setups must be performed per period and on each machine. Setups from a product to itself are allowed and do not consume time. Even though the sequence is coordinated period-overlapping (the first setup in a period is from the last product of the previous period), a setup state variable does not exist and therefore a production quantity is only positive if a setup is performed. Mohammadi et al. (2010b) present several heuristics. The first one uses a period-based decomposition similar to an F&R heuristic. In another heuristic, the sequence is kept equal for each machine, whereat an F&R approach is used. The remaining heuristics focus on the position of the setups and solve the problem step by step. Hereby, demand is either fulfilled by production during the same period (third heuristic) or by producing in earlier periods (fourth heuristic). Mohammadi et al. (2010c) describe a more restricted model with similar sequences, but constant lotsizes over all stages. The model is solved with RH approaches. By contrast, Mohammadi et al. (2011) develop a GA for this model. Mohammadi and Ghomi (2011) combine the GA with an RH approach. Here, the GA determines the binary decisions within the predefined time window. Babaei et al. (2014) include backlogging into the model. Filho et al. (2012) also use this problem and the model of Mohammadi et al. (2010a) as a basis. However, they present a new variant of the "Asynchronous Team approach" proposed by de Souza and Talukdar (1993) as a solution procedure. So-called "agents" help to construct solutions which are saved in shared memory to be improved or deleted again.

Kopanos et al. (2011) consider a single-stage, parallel-machine problem occurring in a bottling facility. Products are grouped to setup families. Backlogging is possible. The production speed is limited by a minimum and maximum production rate. The setup state is conserved. Setups are sequence-dependent across families, but sequence-independent within families, and may be period-overlapping. In addition, maintenance can be considered.

Karimi-Nasab and Seyedhoseini (2013) present a model formulation for a multi-level lotsizing problem with flexible machines which includes a job-shop problem because different processes are necessary for each component. Each machine can work with different production modes/speed levels. The production mode of a machine has to be set before the production starts, i.e., various decisions depend on the production speed. Karimi-Nasab et al. (2013) include "compressible" process times which may vary within a defined interval. According to a given bill-of-materials, a sufficient quantity of components has to be produced before the end product can be assembled. The production of a component is associated with setups (e.g., change of tools, machine inspections) and additional cleaning operations. Each item can be produced only on a predefined set of machines and backlogging is allowed in order to generate feasible production plans. A memetic algorithm (MA) solves the problem heuristically. Nearly the same problem is considered by Karimi-Nasab et al. (2015). They consider a consumption factor of 1 in their bill-of-materials. A PSO algorithm solves the problem. Karimi-Nasab and Modarres (2015) consider the single-level formulation of this problem and add some valid inequalities within a B&C approach. A numerical study is carried out with CPLEX.

Ramezanian et al. (2013a) consider a multi-stage capacitated lotsizing and scheduling problem. Maintenance is also included because the time required for a maintenance task is treated as an additional continuous decision variable. An RH approach is applied to solve the problem heuristically. Ramezanian et al. (2013b) present an extended formulation including setup state conservation. The model is compared to the formulation of Mohammadi et al. (2010c). Again, RH approaches with different freezing strategies are used to solve large problem instances.

A specialized model for wine bottling which considers personnel restrictions is presented by Mac Cawley (2014). The products are grouped into product families. A family changeover causes sequence-independent setup times. Different product sequences for each family are defined in advance. The setup times for a product changeover are sequence-dependent but can be calculated in advance for each given sequence. The task is to assign the families to production lines and macroperiods to fulfill an aggregated demand at the end of the planning horizon for each product. Within each macroperiod, one of the given sequences corresponding to the assigned product family has to be chosen. Furthermore, the lotsizes have to be determined. All of these tasks are presented in a single model formulation. As it is obvious, the model can be decomposed. This is done by aggregating the demand of each product family, solving the resulting problem and determining the sequences and lotsizes in a second step. A similar approach has already been proposed by Meyr and Mann (2013).

Table 2.11 summarizes these models. They mainly consider sequence-dependent setups but vary in their ability to conserve setup states (*Css* = *ls* if setup states are lost and *Css* = *cs* if setup states are conserved). Many of the presented models consider a multi-level production process ($Ps \neq 1$). Among these, a serial (*Ps* = *fr:s*) and a cross-linked (*Ps* = *fr:cl*) flow of materials can be found. For both, a closed transfer of lots is explicitly assumed in recent years (*Ps* = *fr:s:c* and *Ps* = *fr:cl:c*). This can be explained by the high number of models basing on job-shop or flow-shop scheduling problems. Some frequently used modeling approaches can be identified: lots are often considered as jobs in the presented models (see for instance Dauzère-Pérès and Lasserre 2002), or multiple sequences are predefined (e.g. Haase and Kimms 2000) or formulations from lot-sizing and job-shop scheduling problems are combined (for example Dauzère-Pérès and Lasserre 1994). Almost all kinds of solution approaches can be found (see column *heuristics/comments*).

Table 2.11: Literature overview for other models

References	BOM	Ps	M	Sc	Css	St	exoT	endoS	Ls	Industry	Heuristics/comments
Aras and Swanson (1982)	1	1	1	–	cs	si:max:p	d:K	fr	c	CGI	BA
Smith-Daniels and Smith-Daniels (1986)	1	1	1	si	cs	sd:max:p	d:K	fr	c	FI	MIP-solver
Smith-Daniels and Ritzman (1988)	s	fr:s	1	–	cs	sd:max:p	d:K	fr	c	FI	MIP-solver
Baker and Muckstadt Jr. (1989)	1	1	pn	sd:Δv	cs	sd:max:p	d:fr	fr	mima	CHES	not named
Heuts et al. (1992)	1	1	1	sd:Δk	cs	sd:max:p	d:K	fr	dm	PI	Heuristic, incomplete formulation
Dauzère-Pérès and Lasserre (1994)	1	fr:cl	1	si	ls	si	d:K	fr	c	–	Decomposition
Sikora (1996)	1	fr:s	1	sd		sd	d:K	fr		EI	GA
Sikora et al. (1996)	1	fr:s	1	sd		sd:max:p	d:K	fr	max	EI	Heuristic
Jordan and Drexl (1998)	1	1	1	sd:Δk	ls	sd:fr	fr	fr	d	–	B&B
Kang et al. (1999)	1	1	pn	sd:Δv	cs	–	d:fr	fr	mima	CHES	CG+B&B+heuristics
Haase and Kimms (2000)	1	1	1	sd:Δk	cs	sd:max:p	d:K	fr	c	CGI	B&B
Dauzère-Pérès and Lasserre (2002)	g	fr:cl	1	si	ls	–	d:K	fr	c	–	Decomposition
Erdirik-Dogan and Grossmann (2008)	1	1	pn	sd:Δv	cs	sd:max:p	d:fr	fr	min	–	Decomposition
Kovács et al. (2009)	1	1	1	sd:Δk	cs	sd:max:p	d:K	fr	c	–	MIP-solver
Mateus et al. (2010)	1	1	pn	si	ls	sd:max:p	d:K	fr	c	–	Decomposition of heuristics for lotsizing scheduling (GRASP)
Mohammadi et al. (2010a), Mohammadi et al. (2010b), Mohammadi et al. (2010c)	1	fr:s:c	1	sd:Δk	ls	sd:max:p	d:K	fr	c	–	RH, F&R
Kopanos et al. (2011)	1	1	pn	sd:Δk	cs	sd:max:2	d:K	fr	mima	PI	MIP-solver
Mohammadi and Ghomi (2011)	1	fr:s:c	1	sd:Δk	ls	sd:max:p	d:K	fr	c	–	GA with RH approach
Mohammadi et al. (2011)	1	fr:s:c	1	sd:Δk	ls	sd:max:p	d:K	fr	c	–	GA

References	BOM	Ps	M	Sc	Css	St	exoT	endoS	Ls	Industry	Heuristics/comments
Filho et al. (2012)	1	fr:s:c	1	sd:Δk	ls	sd:max:p	d:K	fr	c	–	Asynchronous Team approach
Babaei et al. (2014)	1	fr:s:c	1	sd:Δk	cs	sd:max:p	d:K	fr	c	–	GA
Karimi-Nasab and Seyedhoseini (2013)	g	fr:cl:c	1	sd:Δk	ls	sd:max:p	d:K	fr	c	–	Cutting planes, MIP-solver
Karimi-Nasab et al. (2013)	g	fr:cl:c	1	sd:Δk	ls	sd:max:p	d:K	fr	c	–	MA
Ramezanian et al. (2013a)	1	fr:s:c	1	sd:Δk	ls	sd:max:p	d:K	fr	c	–	RH
Ramezanian et al. (2013b)	1	fr:s:c	1	sd:Δk	cs	sd:max:p	d:K	fr	c	–	RH
Mac Cawley (2014)	1	1	pn	sd:Δv	ls	sd:max:p	d:fr	fr	mima	BI	Decomposition; given product sequences, secondary resources
Urrutia et al. (2014)	1	fr:cl	1	si	ls	si	d:K	fr	c	–	Decomposition, LR, TS
Karimi-Nasab and Modarres (2015)	1	fr:cl:c	1	sd:Δk	ls	sd:max:p	d:K	fr	c	–	Cutting planes/B&C, MIP-solver
Karimi-Nasab et al. (2015)	g	fr:cl:c	1	sd:Δk	ls	sd:max:p	d:K	fr	c	–	PSO, MIP-solver
Wolosewicz et al. (2015)	1	fr:cl	1	si	ls	si	d:K	fr	c	–	LR

2.4 Conclusions

The overview of Sect. 2.3 and especially the Tables 2.6 – 2.11 now allow to identify trends as well as gaps in the field of simultaneous lotsizing and scheduling. Section 2.4.1 provides an analysis of attributes and characteristics using the classification scheme of Table 2.2 as a basis. In Sect. 2.4.2, further extensions are discussed, which have been considered rather seldom in the literature and thus have only been marked by brief keywords in the "comments" columns of Tables 2.6 – 2.11. Section 2.4.3 summarizes findings on practical applications, which have been shown in the "industry" columns (see Table 2.4 to decode the acronyms). Finally, trends regarding solution approaches (see columns "heuristics" and Table 2.5 for the acronyms) are in the focus of Sect. 2.4.4.

2.4.1 Attributes and characteristics

The number of models for a multi-level **bill-of-materials structure** and multiple **production stages** has strongly increased during the recent years. These are often tailored to specific industrial applications such as soft drink production systems or pulp and paper mills. Among the multi-level bills-of-materials, divergent and general structures are pre-dominant. Serial bills-of-materials mainly occur in the context of flow-shops. Many of the multi-stage models are limited to only two production stages which is typical for make-and-pack environments. Sometimes it is sufficient to schedule just one of these stages explicitly. Afterwards, the other stage can be modeled in a quantity-based, aggregate manner only, for example, as an additional scarce resource ("secondary resource") like the pre-product in a divergent bill-of-materials or as a tank restriction.

Otherwise, the schedules of predecessor and successor stages must be synchronized. This means that a successor product can only be processed on a successor machine if a single piece ("open" transfer of lots), a certain batch size or a complete lot ("closed" transfer of lots) of its corresponding predecessor product has been finished on the predecessor machine. This can easily be ensured by positive lead times of at least one period if all machines consider the same time grid as a common basis. However, this way of modeling becomes the more unrealistic the longer the periods of this time grid are. Thus, microperiod-based formulations allow to model multi-stage problems in a simple and realistic way, but suffer from a high number of periods involved and some difficulties in properly representing setup times (see below). Among the microperiod models, the PLSP seems to be most appropriate because it shows the highest degree of freedom. On the contrary, macroperiod-based formulations only appear realistic if pre-products can further be processed within the same macroperiod, i.e., if they allow "zero" lead times. To achieve this, CLSD-based models introduce further variables measuring and restricting the starting and ending times of lots. In the GLSP, however, a common time grid for microperiods (see, e.g., Seeanner and Meyr 2013) or an additional CSLP-like time grid (see, e.g., Toledo et al. 2006 and Toledo et al. 2015) can be introduced, too. Models for both an open and a closed transfer of lots exist.

Despite the recent, increased interest in multi-stage formulations, the majority of models and solution methods does only consider final items (BOM = 1) of a single-stage production system (Ps = 1). Many of these papers are motivated by industrial applications, too. The authors argue that – even though the real world production system actually does comprise several stages – only one of them is a (stationary) bottleneck and needs to be modeled. Often multiple parallel machines can be found on this bottleneck stage. Sometimes, these are identical ($M = pi$). Usually they are heterogeneous ($M = pn$). The reason probably is that not all of them have been acquired at the same point in time. Therefore, they may show different degrees of technological maturity in terms of flexibility, speed, capacities, setup times, setup costs and variable production costs. The GLSP seems to be the first choice for modeling problems with heterogeneous, parallel machines.

Almost all macroperiod models which are based on the GLSP and CLSD and also most of the other models presented in Sect. 2.3.6 (see Table 2.11) consider sequence-dependent **setup times** ($St = sd$). In this case, **setup costs** sometimes do not have to be taken into account or might be considered as sequence-independent ($Sc = si$). On the contrary, all microperiod-models consider setup costs, but only some of them also consider setup times. To be more specific: while setup times were neglected in early PLSP formulations, almost all models respect setup times beginning with Suerie (2005). The same trend can be identified for the CSLP, beginning with Wolsey (1997). If setup times are taken into account, they usually are sequence-independent. Mostly, the same holds true for the setup costs.

Macroperiod-based formulations seem to be first choice when setup times must be modeled. The reason probably is that "simple" standard formulations assume a complete setup to be executed within a single period. This is realistic if rather long macroperiods are considered. However, for short microperiods this assumption would often be too unrealistic. Then, more complex formulations such as the ones of Drexl and Haase (1995) and Suerie (2006) would be required to ensure that a setup can span over multiple microperiods ($St = si{:}fr{:}fr$). As Seeanner (2013, p. 148) has shown, period-overlapping setup times can also be applied within the GLSP. However, for this macroperiod model they only appear to be required if multi-stage production must be synchronized on a common time grid.

Not all of the models are able to properly deal with problem instances whose changeover characteristics violate the triangle inequalities ($Sc = sd{:}\Delta v$). Such instances can easily be handled by the GLSP if minimum lotsizes are postulated (see Sect. 2.2.1 and constraints (2.7)). Some minor disadvantage[11] might be that the whole minimum lotsize has to fit into a single macroperiod. However, since macroperiods are rather long and minimum lotsizes do only represent technological requirements, this assumption should not be crucial. The same holds true for the CLSD. Nevertheless, the CLSD shows an additional, more serious disadvantage: the subtour elimination constraints of the standard CLSD formulation allow a certain product to be produced at most once per macroperiod. If triangle inequalities are violated, such a restriction is too limiting and must be relaxed. Thus, some authors (e.g., Menezes et al. 2011; Guimarães et al. 2013) propose more sophisticated, but also more complex formulations or sequence generation methods that permit connected, but forbid disconnected subtours. The sequence splitting model of Kang et al. (1999) offers an alternative way how macroperiod-based models can deal with violated triangle inequalities. Microperiod-based models generally seem less appropriate because of their higher probability that a technological necessary minimum lotsize exceeds a single period's length.

Most of the CLSD and PLSP models allow the **conservation of a setup state** (*Css* = *cs*). In contrast, most DLSP models assume that the setup state is lost (*Css* = *ls*). In general, both cases may occur, but usually not within a single model formulation. Only a few CSLP- and GLSP-models have been found that are able to handle both cases by a mere variation of input data (*Css* = *cl*) as shown in Sect. 2.2.1.

The **exogenous time structure** (*exoT*) and the **endogenous state changes** (*endoS*) of the models of Sects. 2.3.1 – 2.3.5 by definition naturally coincide with the ones of their corresponding basic formulations presented in Table 2.3. There are only a few exceptions which can easily be explained. The GLSP and the CLSD show *exoT* = *d:fr* if a product can be set up several times per macroperiod what makes sense if the triangle inequalities are violated (*Sc* = *sd:*Δ*v*) or if secondary resources are considered. They show *exoT* = *d:K* if a product can be set up at most once per macroperiod. This is reasonable if the triangle inequalities are kept (*Sc* = *sd:*Δ*k*) or the setup costs are sequence-independent (*Sc* = *si*). The CSLP can similarly be transformed to a macroperiod model as Fleischmann (1990) has done for the DLSP (see Sect. 2.2.2). If this was the case, *exoT* = *d:K* and *endoS* = *d*:1:*fr* do occur in Table 2.9 instead of *exoT* = *d*:1 and *endoS* = *fr*. The other way round, *exoT* = *d*:1 and *endoS* = *fi* instead of *exoT* = *d:K* and *endoS* = *d:*1:*fi* are shown in Table 2.10 if Fleischmann's transformation has not been applied for the DLSP. The only exception is Fleischmann (1994) with *exoT* = *d:fr* because his sequence-dependent, macroperiod-based formulation additionally allows a violation of the triangle inequalities.

Most of the other models of Sect. 2.3.6 and Table 2.11 either show *exoT* = *d:K* or *exoT* = *d:fr*. All of them show *endoS* = *fr*. This means that – similar to the GLSP and CLSD – these are also hybrid models with an exogenous macroperiod structure and free endogenous state changes. However, the modeling of these state changes does not allow a clear categorization either as GLSP (predefined sequence of microperiods within a macroperiod) or CLSD (TSP-like definition of the sequence of products within a macroperiod). The batch sequencing problem (BSP) of Jordan and Drexl (1998) is the only exception that shows *exoT* = *fr*. This model was actually intended for scheduling. Therefore, demand is represented via due dates for customer orders ("jobs"), which are defined on a continuous

[11]Which can be mitigated as, for example, Seeanner (2013, p. 148) has shown.

time scale. Since the authors have proven that the DLSP can be transformed into a BSP, this scheduling model has also been included into our review.

All in all, it can be summarized that large-bucket models dominate in the meantime. While most of the models reviewed were based on small-buckets until and during the nineties, the CLSD by Haase (1996) and the GLSP by Fleischmann and Meyr (1997) obviously revealed new modeling approaches. Still some research has been done on the small-bucket models DLSP, CSLP and PLSP during the past five to ten years. However, this has clearly been outnumbered by the macroperiod-based research presented in Tables 2.6, 2.7 and 2.11.

Concerning the attribute "**lotsize**" the whole range of potential values presented in Table 2.2 can be found. Multiple repetitions of a discrete lotsize ($Ls = dm$) are obligatory for the DLSP because of its all-or-nothing assumption. Nevertheless, this sometimes also occurs for the other model types. Then, usually an industrial application with a batch production system is taken into account. If violated triangle inequalities can be considered, positive minimum lotsizes need to be respected (see Sect. 2.2.1), i.e., either $Ls = min$, $Ls = mima$ or $Ls = dm$ are valid. Maximum lotsizes ($Ls = max$, $Ls = dm$ or $Ls = mima$) do only occur seldom. If they do, they are usually also motivated by industrial applications. Technical constraints requiring a maximum lotsize can, for example, exist in the case of batch production in reactors or tanks, which can continuously be filled up to a certain level, or filters that contaminate during production and thus at the latest have to be cleaned after a maximum quantity produced.

2.4.2 Further extensions

Some models consider **a second type of scarce resource** as a possible extension to the basic formulations presented in Figure 2.1 and Table 2.3. These resources are required in addition to the primary production resources (e.g., C_t in inequalities (2.2)). If there are multiple machines, this resource may be required by all or a subset of the machines but cannot be utilized by all of them at the same point in time. Examples are setup operators, dies or tanks that are additionally needed for a setup or for carrying out the production. Early works of Lasdon and Terjung (1971) and Eppen and Martin (1987) concerning the DLSP already considered this problem characteristic. Interestingly, this problem was not picked up by research for another 15 years. Starting with Göthe-Lundgren et al. (2002), Persson et al. (2004) and Jans and Degraeve (2004), there seems to be an increased interest in this type of problem during the recent years. Most of the work is inspired by industrial applications. Many models concentrate on a single production stage and parallel machines. To coordinate the usage of the scarce resource over all parallel machines is – because of its all-or-nothing assumption – rather easy in the DLSP. This explains the DLSP's early usage for this kind of additional complexity. More general models track the starting and ending times of setups or production activities and coordinate them across multiple machines. This is also less difficult for microperiod-based models like the PLSP (see, e.g., Tempelmeier and Buschkühl 2008) than for macroperiod-based ones like the GLSP (see, e.g., Almeder and Almada-Lobo 2011) or the CLSD (see, e.g., Tempelmeier and Copil 2016). However, if sequence-dependent setup times are important, the additional effort of the macroperiod models is obviously well spent.

Pahl and Voß (2010), Pahl et al. (2011) and Tempelmeier and Copil (2016) consider **perishability** as an additional extension of the basic formulations, which is of particular interest for the agrifood business. The idea is to prevent spoilage, decay or general obsolescence of stored products by limiting the total amount of inventory. The research of Amorim et al. (2011) and Amorim et al. (2012), which is not further discussed here because it does not specifically consider inventory holding costs (see Sect. 2.1), confirms the impression that perishability issues get an increased attention in the recent past.

Lang (2010) and Lang and Shen (2011) are the only authors who consider **product substitution**. A reason might be that simultaneous lotsizing and scheduling is mainly relevant for make-to-stock production (see Sect. 2.4.3). There, customers expect that their orders are fulfilled from stock on hand. Overall, product substitution is a planning task which is mainly relevant for demand fulfillment (see, e.g., Kilger and Meyr 2015). Nevertheless, Lang and Shen (2011) think one step ahead to save production costs by substitution: their idea is that the option of substitution affects the optimal production sequence and vice versa. Thus it might, for example, be beneficial to save setup times by substituting products by others instead of producing them at all.

2.4.3 Practical applications

By reviewing Sect. 2.3, it can be observed that many formulations are based on practical cases. Most frequently, problems from the process industry are used. Examples are Heuts et al. (1992), de Matta and Guignard (1994a), de Matta and Guignard (1994b), de Araujo et al. (2008), Kopanos et al. (2011) or Transchel et al. (2011). Santos and Almada-Lobo (2012) and Figueira et al. (2013) consider problems from a pulp and paper mill. Almada-Lobo et al. (2007), Almada-Lobo et al. (2008), Almada-Lobo et al. (2010), Kopanos et al. (2011) or Toledo et al. (2013) treat problems observed in, e.g., glass container production and bottling facilities. Other problems from the process industry can be found in oil refineries or tire manufacturing. Problems in the consumer goods industry are considered frequently, too. Laguna (1999), Haase and Kimms (2000), Gupta and Magnusson (2005), Lang and Shen (2011), Tiacci and Saetta (2012), Seeanner et al. (2013), Seeanner and Meyr (2013) or Camargo et al. (2014) formulate models for associated problems. Numerous models are based on problems from the beverage industry, e.g., by Toledo et al. (2006), Ferreira et al. (2009), Ferreira et al. (2012) or Baldo et al. (2014). The (animal) food industry provides problems for Smith-Daniels and Smith-Daniels (1986), Smith-Daniels and Ritzman (1988), Toso et al. (2009), Clark et al. (2010) or Tempelmeier and Copil (2016). Other industries produce automobiles, electronics, semiconductors, or pharmaceutical as well as chemical products in a broader sense.

All in all, a surprisingly large number of papers is motivated by industrial applications, indicating that this field of research shows a high practical relevance. Considering the fact that – starting with Harris (1913) and the economic order quantity model – lotsizing in general has been studied for around a 100 years and that dynamic lotsizing is also more than half a century old (see Wagner and Whitin 1958), the past 15 years' boom of simultaneous lotsizing and scheduling research appears somewhat surprising. It can best be explained by this obvious industrial relevance and a still unsatisfactory support for decision making.

Most of the industrial applications consider production systems of the flow-shop type in which continuous products or a large number of discrete pieces are produced. Almost all of these applications include setup times, which in most cases are sequence-dependent. Exceptions can mainly be found for some DLSP- and PLSP-based papers of Sects. 2.3.3 and 2.3.5. Models representing these practical applications do often only consider a single production stage ($Ps = 1$), even though the underlying real world problem may consist of multiple stages. In this case, one stage has been identified as the main, stationary bottleneck. On this bottleneck stage, parallel production lines may be present – sometimes homogenous, but mostly heterogeneous. Usually, only the products processed by the bottleneck stage are simultaneously treated in a lotsizing and scheduling model ($BOM = 1$). These products do not need to be end products, but rather may be defined as product families. Even applications in the semiconductors industry, actually comprising a large number of stages, can be found which are modeled this way (see Quadt and Kuhn 2005; Quadt and Kuhn 2009; Xiao et al. 2013). As mentioned above, sometimes it is sufficient to model a second stage only in an aggregate way by means of an additional scarce resource (see, e.g., Ferreira et al. 2009).

Within the multi-stage industrial applications, the two-stage, serial models ($Ps = 2{:}s$) do prevail. They often come along with a divergent bill-of-materials ($BOM = d$). This is, for example, the case in common make-and-pack situations in which a basic product is produced on the first stage and then packed or filled in packages of different sizes, brands or country-specific languages on the second stage. Since the bottleneck may shift depending on the mix of the various products' demand, all stages must be considered in detail within an integrated model.

2.4.4 Solution approaches

By comparing the models, it becomes obvious that the GLSP can easier be formulated by means of standard modeling languages than the CLSD because of its straightforward, sequential ordering of microperiods and because it does not need any subtour elimination constraints. On the contrary, tightness of bounds and thus solvability with standard branch&bound[12] or branch&cut MIP-solvers rather favor the CLSD (see, e.g., Almeder and Almada-Lobo 2011). Here, the enormous knowledge about reformulations, valid inequalities etc. concerning the traveling salesman problem can be exploited to strengthen model formulations. Nevertheless, as long as sequence-dependent setup costs and times are involved, lower bounds remain disappointing. Thus, optimality can merely be proven for small problem instances. Heuristics are the only choice to solve problems of industrial size. As Sect. 2.3 shows, traditional sophisticated solution approaches such as dynamic programming, Lagrangean relaxation, Lagrangean decomposition or column generation only have successfully been applied to the DLSP and CSLP. Such methods are able to take advantage of the very restrictive assumptions made by these microperiod models (like the all-or-nothing assumption; see also Fig. 2.1). The more general a model formulation is the less this is possible, for example, because a model does not decompose into easily solvable subproblems any more as it would be needed for an LR.

[12]Unfortunately, "branch & cut" is a typing error in Copil et al. (2017).

Thus, the rise of more general models like the PLSP, CLSD and GLSP during the late nineties came along with the rise of deterministic and randomized meta-heuristics basing on local search (like TA, SA, TS, RR, RM, DS; see Table 2.5) or evolutionary principles (like GA). Basically, setup sequences are varied by means of local search or the GA. The corresponding integer variables are fixed and the solutions are evaluated by solving the resulting continuous lotsizing problems. For the latter, either rule-based heuristics or LP methods can be applied. The meta-heuristic controls the selection and acceptance of neighbored candidate sequences or whole populations of sequences. Obviously, such methods are the more demanding the more setups are involved and the more complex the embedded lotsizing problems are. However, the solution quality of meta-heuristics often remains unclear as neither the optimal solutions nor lower bounds (in case of minimization) are available. By contrast, an LR heuristic always provides a lower bound resulting from the relaxed problem and often also an upper bound derived by utilizing the Lagrangean multipliers to detect feasible solutions. The difference between these bounds allows to assess the solution quality.

As a next generation of solution methods for the models of Sect. 2.3, MIP-based approaches like F&R, F&O and RH can be identified. Similar to LR or LD, they also decompose the original problem into less complex subproblems that are successively solved. However, these subproblems usually still have the original MIP formulation. They just show a reduced number of integer variables involved by either fixing or relaxing the remaining ones. Thus, these approaches strongly rely on the power of modern MIP-solvers. Apparently, strengthening the original model formulation does also improve the performance of any MIP-based heuristic applying this formulation. This explains why recent research often concentrates on tight formulations for small problem instances which are solved either heuristically (until a certain time limit or optimality gap is reached) or to optimality (zero gap) by a standard MIP-solver only. The advantage is that even complex model formulations can easily be reused as part of a solution heuristic, once they have been tested and validated for the first time.

2.5 Outlook

Obviously, the interest in small-bucket models is not that strong anymore. The researchers' main focus on industrial applications might be a reason for this. Small-bucket models are indeed suitable for industrial use, too. However, this is apparently less often the case than for the large-bucket models GLSP and CLSD. Better hardware and more powerful standard solvers now allow to solve more complex applications. Thus, some characteristics or constraints which have been considered as less important in former times do nowadays receive increased attention.

Among these, *shifting bottlenecks* are probably the most prominent ones. To consider bottlenecks that may vary depending on a dynamically changing mix of customer demand requires models that are able to respect several stages of production simultaneously. On the one hand, these models should not build on unrealistically long lead times, which have only been introduced to ease modeling but do not exist in industrial practice (see the above discussion on zero lead times and line synchronization). Thus, microperiod-based models would better suit than macroperiod-based ones. On the other hand, sufficiently long setup times need to be modeled, too. This is easier accomplished with macroperiod-based formulations. Here, further research is needed, for example, to compare the different ideas that have been presented by Seeanner and Meyr (2013) or Seeanner et al. (2013) and Stadtler (2011) or Stadtler and Sahling (2013) among each other.

Furthermore, *secondary resources* have attracted more and more attention in recent times. Such secondary resources can, for example, be expensive dies or tools, but also specialized personnel like setup operators, that support setup, production or maintenance processes. However, as the preceding sections have shown, in an aggregate way, even a second stage of production may have been modeled as such a scarce secondary resource. Secondary resources, which are required simultaneously with the production systems' primary resources (which are typically the machines) introduce additional complexity into the planning problem as now the production schedule must be coordinated across multiple machines. This prevents the decomposition of a multi-machine problem into multiple single-machine problems. In light of their growing degree of attention, it seems worthwhile to more deeply analyze the underlying industrial applications to develop a classification and a general modeling framework for these kinds of secondary resources.

When considering the GLSP and CLSD from a *modeling* and *methodological* point of view, the CLSD shows advantages if MIP-based solution techniques are applied, which require tight model formulations. Nevertheless, as Almeder and Almada-Lobo (2011) state, it is still necessary to further strengthen such formulations. On the contrary, some industrial characteristics like changeovers violating the triangle inequalities are easier to model when using a GLSP-type time structure. "Easier" means in this context not only less error-prone, but also better extendable for further industrial needs. Here, local search or evolutionary methods might be more appropriate because they do not rely on the tightness of the MIP formulation. However, math-programming techniques can be applied for the GLSP, too, at least if smaller MIP-subproblems are considered. Seeanner et al. (2013) proposed a compromise that seems to be an interesting prospect for future research: by combining variable neighborhood decomposition search and F&O, they use a *hybrid* of both types of solution approaches.

Traditional job-shop planning separates the lotsizing and the scheduling tasks from each other and executes them one after the other. It seems that the gap between such a successive planning and the simultaneous lotsizing and scheduling models of this review is reducing. Some approaches presented in Sect. 2.3.6 already bridge this gap by solving integrated lotsizing and job-shop scheduling models in a hierarchical and repetitive manner. From a methodological point of view, these again can be considered as iterative decomposition approaches, complementing the ones mentioned in Sect. 2.4.4. They rather focus on exploiting the bill-of-materials than on taking advantage of time-related aspects as, for example, rolling horizon decompositions do. For the multitude of industrial flow-shop applications, which Table 2.4 and Sect. 2.3 report from, this has not yet been necessary. The products manufactured there usually show a quite simple structure. However, future research on this type of decompositions may indeed create the necessary experiences to solve complex multi-stage problems in a more integrated manner.

Summing up, simultaneous lotsizing and scheduling is a prosperous research area motivated by many industrial applications. These applications do often comprise a one- or two-stage production system with one or several parallel, mostly heterogeneous machines (production lines) per stage. Substantial, sequence-dependent changeover times do prevail. Many of the applications originate from process or consumer goods industries including the production of food and beverages. There, products are demanded in high quantities and usually made to stock. Because of the high effort for the changeovers, lots of sufficient size have to be determined on the one hand. On the other hand, a clever choice of their sequence is necessary. The strong interdependency between both types of decisions calls for an integrated planning approach.

The substantial effort for changeovers does also forbid an extremely high product variety as it is, for example, well known from premium automobiles' production. Thus, lean management and a one-piece flow are not yet an issue in these kinds of industries. Product variety often arises from a few, similar types of products produced (major setups), which are then packaged in various different formats (minor setups). Nevertheless, also for these industries there is evidence (see Günther 2014) that production technology will improve so that setups become less costly and time-consuming, that product variety will grow, that decreased delivery times will play a more prominent role and that there will be a shift from a pure make-to-stock (MTS) to a hybrid MTS/make-to-order production. Then, a more integrated production and distribution planning or block planning approaches as propagated by Günther (2014) will become more and more important. It would be worthwhile to analyze in more detail whether at all, when and how a change of paradigm is to be expected and to extend this paper's review for such new planning approaches.

Nevertheless, the acquisition of the currently applied production technology has required notable investments. Such a technology can usually not completely be replaced in a single big bang. The transfer to a more modern production technology will rather go step by step, for example, by replacing one production line after the other. This may span over dozens of years. Thus, the need for a simultaneous lotsizing and scheduling will for sure remain for a long time.

Acknowledgments We thank the anonymous reviewers and area editor for their helpful comments.

Bibliography

Almada-Lobo, B., Carravilla, M. A., Oliveira, J. F., 2008. Production planning and scheduling in the glass container industry: a VNS approach. International Journal of Production Economics 114 (1), 363–375.

Almada-Lobo, B., James, R. J. W., 2010. Neighbourhood search meta-heuristics for capacitated lot-sizing with sequence-dependent setups. International Journal of Production Research 48 (3), 861–878.

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

Almada-Lobo, B., Klabjan, D., Carravilla, M. A., Oliveira, J. F., 2010. Multiple machine continuous setup lotsizing with sequence-dependent setups. Computational Optimization and Applications 47 (3), 529–552.

Almeder, C., Almada-Lobo, B., 2011. Synchronisation of scarce resources for a parallel machine lot-sizing problem. International Journal of Production Research 49 (24), 7315–7335.

Almeder, C., Klabjan, D., Traxler, R., Almada-Lobo, B., 2015. Lead time considerations for the multi-level capacitated lot-sizing problem. European Journal of Operational Research 241 (3), 727–738.

Amorim, P., Antunes, C. H., Almada-Lobo, B., 2011. Multi-objective lot-sizing and scheduling dealing with perishability issues. Industrial & Engineering Chemistry Research 50 (6), 3371–3381.

Amorim, P., Günther, H.-O., Almada-Lobo, B., 2012. Multi-objective integrated production and distribution planning of perishable products. International Journal of Production Economics 138 (1), 89–101.

Amorim, P., Pinto-Varela, T., Almada-Lobo, B., Barbósa-Póvoa, A. P. F. D., 2013. Comparing models for lot-sizing and scheduling of single-stage continuous processes: operations research and process systems engineering approaches. Computers and Chemical Engineering 52, 177–192.

Aras, O., Swanson, L., 1982. A lot sizing and sequencing algorithm for dynamic demands upon a single facility. Journal of Operations Management 2 (3), 177–185.

Babaei, M., Mohammadi, M., Ghomi, S. M. T. F., 2014. A genetic algorithm for the simultaneous lot sizing and scheduling problem in capacitated flow shop with complex setups and backlogging. International Journal of Advanced Manufacturing Technology 70 (1-4), 125–134.

Baker, T., Muckstadt Jr., J., 1989. The CHES problems. Tech. rep., Chesapeake Decision Sciences, Inc.

Baldo, T. A., Santos, M. O., Almada-Lobo, B., Neto, R. M., 2014. An optimization approach for the lot sizing and scheduling problem in the brewery industry. Computers & Industrial Engineering 72, 58–71.

Brüggemann, W., Fischer, K., Jahnke, H., 2003a. Problems, models and complexity. Part I: Theory. Journal of Mathematical Modelling and Algorithms 2 (2), 121–151.

Brüggemann, W., Fischer, K., Jahnke, H., 2003b. Problems, models and complexity. Part II: Application to the DLSP. Journal of Mathematical Modelling and Algorithms 2 (2), 153–169.

Brüggemann, W., Jahnke, H., 1994. DLSP for two-stage multi-item batch production. International Journal of Production Research 32 (4), 755–768.

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

Brüggemann, W., Jahnke, H., 2000b. Erratum to "The discrete lot-sizing and scheduling problem: Complexity and modification for batch availability" [EJOR 124 (3) (2000) 511–528]. European Journal of Operational Research 126 (3), 688.

Buschkühl, L., Sahling, F., Helber, S., Tempelmeier, H., 2010. Dynamic capacitated lot-sizing problems: a classification and review of solution approaches. OR Spectrum 32 (2), 231–261.

Camargo, V. C. B., Toledo, F. M. B., Almada-Lobo, B., 2012. Three time-based scale formulations for the two stage lot sizing and scheduling in process industries. Journal of the Operational Research Society 63, 1613–1630.

Camargo, V. C. B., Toledo, F. M. B., Almada-Lobo, B., 2014. HOPS – hamming-oriented partition search for production planning in the spinning industry. European Journal of Operational Research 234 (1), 266–277.

Campbell, G. M., 1992. Using short-term dedication for scheduling multiple products on parallel machines. Production and Operations Management 1 (3), 295–307.

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

Chang, P., Chang, C., Chen, C., 2004. An extension of the multi-machine multi-level proportional lot sizing and scheduling model for product-life-cycle demand. International Journal of Operations Research 1 (1), 11–22.

Clark, A. R., 2003. Optimization approximations for capacity constrained material requirements planning. International Journal of Production Economics 84 (2), 115–131.

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

Clark, A. R., Mahdieh, M., Rangel, S., 2014. Production lot sizing and scheduling with non-triangular sequence-dependent setup times. International Journal of Production Research 52 (8), 2490–2503.

Clark, A. R., Morabito, R., Toso, E. A. V., 2010. Production setup-sequencing and lot-sizing at an animal nutrition plant through ATSP subtour elimination and patching. Journal of Scheduling 13 (2), 111–121.

Constantino, M., 2000. A polyhedral approach to a production planning problem. Annals of Operations Research 96 (1-4), 75–95.

Copil, K., Wörbelauer, M., Meyr, H., Tempelmeier, H., 2017. Simultaneous lotsizing and scheduling problems: a classification and review of models. OR Spectrum 39 (1), 1–64.

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

Dauzère-Pérès, S., Lasserre, J.-B., 1994. Integration of lotsizing and scheduling decisions in a job-shop. European Journal of Operational Research 28 (3), 413–426.

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

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

de Araujo, S. A., Arenales, M. N., Clark, A. R., 2008. Lot sizing and furnace scheduling in small foundries. Computers & Operations Research 35 (3), 916–932.

de Araujo, S. A., Clark, A. R., 2013. A priori reformulations for joint rolling-horizon scheduling of materials processing and lot-sizing problem. Computers & Industrial Engineering 65 (4), 577–585.

de Matta, R., 1994. A Lagrangean decomposition solution to a single line multiproduct scheduling problem. European Journal of Operational Research 79 (1), 25–37.

de Matta, R., Guignard, M., 1994a. Dynamic production scheduling for a process industry. Operations Research 42 (3), 492–503.

de Matta, R., Guignard, M., 1994b. Studying the effects of production loss due to setup in dynamic production scheduling. European Journal of Operational Research 72 (1), 62–73.

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

de Souza, P. S., Talukdar, S. N., 1993. Asynchronous organizations for multialgorithm problems. In: Proceedings of the 1993 ACM/SIGAPP symposium on applied computing: states of the art and practice. pp. 286–293.

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

Drexl, A., Haase, K., 1996. Sequential-analysis based randomized-regret-methods for lot-sizing and scheduling. Journal of the Operational Research Society 47 (2), 251–265.

Drexl, A., Haase, K., Kimms, A., 1995. Losgrößen- und Ablaufplanung in PPS-Systemen auf der Basis randomisierter Opportunitätskosten. Zeitschrift für Betriebswirtschaft 65 (3), 267–285.

Drexl, A., Kimms, A., 1997. Lot sizing and scheduling – survey and extensions. European Journal of Operational Research 99 (2), 221–235.

Dumas, Y., Desrosiers, J., Gelinas, E., Solomon, M. M., 1995. An optimal algorithm for the traveling salesman problem with time windows. Operations Research 43 (2), 367–371.

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

Erdirik-Dogan, M., Grossmann, I. E., 2008. Simultaneous planning and scheduling of single-stage multi-product continuous plants with parallel lines. Computers & Chemical Engineering 32 (11), 2664–2684.

Fandel, G., Stammen-Hegener, C., 2006. Simultaneous lot sizing and scheduling for multi-product multi-level production. International Journal of Production Economics 104 (2), 308–316.

Ferreira, D., Clark, A. R., Almada-Lobo, B., Morabito Neto, R., 2012. Single-stage formulations for synchronised two-stage lot sizing and scheduling in soft drink production. International Journal of Production Economics 136 (2), 255–265.

Ferreira, D., Morabito Neto, R., Rangel, S., 2009. Solution approaches for the soft drink integrated production lot sizing and scheduling problem. European Journal of Operational Research 196 (2), 697–706.

Ferreira, D., Morabito Neto, R., Rangel, S., 2010. Relax and fix heuristics to solve one-stage one-machine lot-scheduling models for small-scale soft drink plants. Computers & Operations Research 37 (4), 684–691.

Figueira, G., Santos, M. O., Almada-Lobo, B., 2013. A hybrid VNS approach for the short-term production planning and scheduling: a case study in the pulp and paper industry. Computers & Operations Research 40 (7), 1804–1818.

Filho, M. A. F., Santos, M. O., Meneses, C. N., 2012. Asynchronous teams for joint lot-sizing and scheduling problem in flow shops. International Journal of Production Research 50 (20), 5809–5822.

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

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

Fleischmann, B., Meyr, H., 1997. The general lotsizing and scheduling problem. OR Spectrum 19 (1), 11–21.

Furlan, M. M., Almada-Lobo, B., Santos, M. O., Morabito Neto, R., 2013. A genetic algorithm-based heuristic with unequal individuals to tackle the lot-sizing and scheduling problem. In: Proceedings of XLV SBPO - Simpösio Brasileiro de Pesquisa Operacional. pp. 3505–3516.

Furlan, M. M., Almada-Lobo, B., Santos, M. O., Morabito Neto, R., 2015. Unequal individual genetic algorithm with intelligent diversification for the lot-scheduling problem in integrated mills using multiple-paper machines. Computers & Operations Research 59, 33–50.

Gaglioppa, F., Miller, L. A., Benjaafar, S., 2008. Multitask and multistage production planning and scheduling for process industries. Operations Research 56 (4), 1010–1025.

Gicquel, C., 2008. MIP models and exact methods for the discrete lot-sizing and scheduling problem with sequence-dependent changeover costs and times. Ph.D. thesis, École Centrale Paris.

Gicquel, C., Lisser, A., Minoux, M., 2014. An evaluation of semidefinite programming based approaches for discrete lot-sizing problems. European Journal of Operational Research 237 (2), 498–507.

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

Gicquel, C., Minoux, M., 2015. Multi-product valid inequalities for the discrete lot-sizing and scheduling problem. Computers & Operations Research 54, 12–20.

Gicquel, C., Minoux, M., Dallery, Y., 2009b. On the discrete lot-sizing and scheduling problem with sequence-dependent changeover times. Operations Research Letters 37 (1), 32–36.

Gicquel, C., Minoux, M., Dallery, Y., 2011. Exact solution approaches for the discrete lot-sizing and scheduling problem with parallel resources. International Journal of Production Research 49 (9), 2587–2603.

Gicquel, C., Wolsey, L. A., Minoux, M., 2012. On discrete lot-sizing and scheduling on identical parallel machines. Optimization Letters 6 (3), 545–557.

Göthe-Lundgren, M., Lundgren, J. T., Persson, J. A., 2002. An optimization model for refinery production scheduling. International Journal of Production Economics 78 (3), 255–270.

Grünert, T., 1998. Multi-level sequence-dependent dynamic lotsizing and scheduling. Shaker Verlag, Aachen.

Guimarães, L., Klabjan, D., Almada-Lobo, B., 2013. Pricing, relaxing and fixing under lot sizing and scheduling. European Journal of Operational Research 230 (2), 399–411.

Guimarães, L., Klabjan, D., Almada-Lobo, B., 2014. Modeling lotsizing and scheduling problems with sequence dependent setups. European Journal of Operational Research, 239 (3), 644–662.

Günther, H.-O., 2014. The block planning approach for continuous time-based dynamic lot sizing and scheduling. Business Research 7 (1), 51–76.

Günther, H.-O., Grunow, M., Neuhaus, U., 2006. Realizing block planning concepts in make-and-pack production using MILP modelling and SAP APO. International Journal of Production Research 44 (18-19), 3711–3726.

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

Haase, K., 1994. Lotsizing and scheduling for production planning. Springer, Berlin.

Haase, K., 1996. Capacitated lot-sizing with sequence dependent setup costs. OR Spectrum 18 (1), 51–59.

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

Harris, F. W., 1913. How many parts to make at once. Factory, The Magazine of Management 10 (2), 135–136, 152.

Heuts, R. M. J., Seidel, H. P., Selen, W. J., 1992. A comparison of two lot sizing-sequencing heuristics for the process industry. European Journal of Operational Research 59 (3), 413–424.

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

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

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

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

Kaczmarczyk, W., 2011. Proportional lot-sizing and scheduling problem with identical parallel machines. International Journal of Production Research 49 (9), 2605–2623.

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

Karimi-Nasab, M., Modarres, M., 2015. Lot sizing and job shop scheduling with compressible process times: A cut and branch approach. Computers & Industrial Engineering 85, 196–205.

Karimi-Nasab, M., Modarres, M., Seyedhoseini, S., 2015. A self-adaptive PSO for joint lot sizing and job shop scheduling with compressible process times. Applied Soft Computing 27, 137–147.

Karimi-Nasab, M., Seyedhoseini, S. M., 2013. Multi-level lot sizing and job shop scheduling with compressible process times: A cutting plane approach. European Journal of Operational Research 231 (3), 598–616.

Karimi-Nasab, M., Seyedhoseini, S. M., Modarres, M., Heidari, M., 2013. Multi-period lot sizing and job shop scheduling with compressible process times for multilevel product structures. International Journal of Production Research 51 (20), 6229–6246.

Karmarkar, U. S., Schrage, L., 1985. The deterministic dynamic product cycling problem. Operations Research 33 (2), 326–345.

Kilger, C., Meyr, H., 2015. Demand fulfillment and ATP. In: Stadtler, H., Kilger, C., Meyr, H. (Eds.), Supply chain management and advanced planning: concepts, models, software and case studies, 5th Edition. Springer Berlin, Ch. 9, pp. 177–194.

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

Kimms, A., 1996b. Multi-level, single-machine lot sizing and scheduling (with initial inventory). European Journal of Operational Research 89 (1), 86–99.

Kimms, A., 1997a. Demand shuffle – a method for multilevel proportional lot sizing and scheduling. Naval Research Logistics 44 (4), 319–340.

Kimms, A., 1997b. Multi-level Lot Sizing and Scheduling. Physics, Heidelberg.

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

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

Koçlar, A., Süral, H., 2005. A note on "The general lot sizing and scheduling problem". OR Spectrum 27 (1), 145–146.

Kopanos, G. M., Puigjaner, L., Maravelias, C. T., 2011. Production planning and scheduling of parallel continuous processes with product families. Industrial & Engineering Chemistry Research 50 (3), 1369–1378.

Kovács, A., Brown, K. N., Tarim, S. A., 2009. An efficient MIP model for the capacitated lot-sizing and scheduling problem with sequence-dependent setups. International Journal of Production Economics 118 (1), 282–291.

Kwak, I., Jeong, I., 2011. A hierarchical approach for the capacitated lot-sizing and scheduling problem with a special structure of sequence-dependent setups. International Journal of Production Research 49 (24), 7425–7439.

Laguna, M., 1999. A heuristic for production scheduling and inventory control in the presence of sequence-dependent setup time. IIE Transactions 31 (2), 125–134.

Lang, J. C., 2010. Production and Inventory Management with Substitutions. Lecture Notes in Economics and Mathematical Systems. Springer, Heidelberg et al.

Lang, J. C., Shen, Z. M., 2011. Fix-and-optimize heuristics for capacitated lot-sizing with sequence-dependent setups and substitutions. European Journal of Operational Research 214 (3), 595–605.

Lasdon, L., Terjung, R. C., 1971. An efficient algorithm for multi-item scheduling. Operations Research 19 (4), 946–969.

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

Mac Cawley, A. F., 2014. The international wine supply chain: challenges from bottling to the glass. Ph.D. thesis, Georgia Institute of Technology.

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

Maldonado, M., Rangel, S., Ferreira, D., 2014. A study of different subsequence elimination strategies for the soft drink production planning. Journal of Applied Research and Technology 12 (4), 631–641.

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

Mateus, G. R., Ravetti, M. G., de Souza, M. C., Valeriano, T. M., 2010. Capacitated lot sizing and sequence dependent setup scheduling: an iterative approach for integration. Journal of Scheduling 13 (3), 245–259.

Menezes, A. A., Clark, A. R., Almada-Lobo, B., 2011. Capacitated lot-sizing and scheduling with sequence-dependent, period-overlapping and non-triangular setups. Journal of Scheduling 14 (2), 209–219.

Meyr, H., 1999. Simultane Losgrößen- und Reihenfolgeplanung für kontinuierliche Produktionslinien. Deutscher Universitätsverlag, Wiesbaden.

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

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

Meyr, H., 2004. Simultane Losgrößen- und Reihenfolgeplanung bei mehrstufiger kontinuierlicher Fertigung. Zeitschrift für Betriebswirtschaft 74 (6), 585–610.

Meyr, H., Mann, M., 2013. A decomposition approach for the general lotsizing and scheduling problem for parallel production lines. European Journal of Operational Research 229 (3), 718–731.

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

Mohammadi, M., 2010. Integrating lotsizing, loading, and scheduling decisions in flexible flow shops. The International Journal of Advanced Manufacturing Technology 50 (9-12), 1165–1174.

Mohammadi, M., Ghomi, S. M. T. F., 2011. Genetic algorithm-based heuristic for capacitated lotsizing problem in flow shops with sequence-dependent setups. Expert Systems with Applications 38 (6), 7201–7207.

Mohammadi, M., Ghomi, S. M. T. F., Jafari, N., 2011. A genetic algorithm for simultaneous lotsizing and sequencing of the permutation flow shops with sequence-dependent setups. International Journal of of Computer Integrated Manufacturing 24 (1), 87–93.

Mohammadi, M., Ghomi, S. M. T. F., Karimi, B., Torabi, S. A., 2010a. MIP-based heuristics for lotsizing in capacitated pure flow shop with sequence-dependent setups. International Journal of Production Research 48 (10), 2957–2973.

Mohammadi, M., Ghomi, S. M. T. F., Karimi, B., Torabi, S. A., 2010b. Rolling-horizon and fix-and-relax heuristics for the multi-product multi-level capacitated lotsizing problem with sequence-dependent setups. Journal of Intelligent Manufacturing 21 (4), 501–510.

Mohammadi, M., Karimi, B., Ghomi, S. M. T. F., Torabi, S. A., 2010c. A new algorithmic approach for capacitated lot-sizing problem in flow shops with sequence-dependent setups. International Journal of Advanced Manufacturing Technology 49 (1-4), 201–211.

Mohammadi, M., Poursabzi, O., 2014. A rolling horizon-based heuristic to solve a multi-level general lot sizing and scheduling problem with multiple machines (MLGLSP_MM) in job shop manufacturing system. Uncertain Supply Chain Management 2 (3), 167–178.

Muramatsu, K., Warman, A., Kobayashi, M., 2003. A near-optimal solution method of multi-item multi-process dynamic lot size scheduling problem. JSME International Journal Series C Mechanical Systems, Machine Elements and Manufacturing 46 (1), 46–53.

Pahl, J., Voß, S., 2010. Discrete lot-sizing and scheduling including deterioration and perishability constraints. In: Dangelmaier, W., Blecken, A., Delius, R., Klöpfer, S. (Eds.), Advanced manufacturing and sustainable logistics. Vol. 46 of Lecture notes in business information processing. Springer Berlin Heidelberg, pp. 345–357.

Pahl, J., Voß, S., Woodruff, D. L., 2011. Discrete lot-sizing and scheduling with sequence-dependent setup times and costs including deterioration and perishability constraints. In: Proceedings of the 44th Hawaii international conference on system sciences. IEEE Computer Society, Los Alamitos, pp. 1–10.

Persson, J. A., Göthe-Lundgren, M., Lundgren, J. T., Gendron, B., 2004. A tabu search heuristic for scheduling the production processes at an oil refinery. International Journal of Production Research 42 (3), 445–471.

Pochet, Y., Wolsey, L. A., 1991. Solving multi-item lot-sizing problems using strong cutting planes. Management Science 37 (1), 53–67.

Pochet, Y., Wolsey, L. A., 2006. Production Planning by Mixed Integer Programming. Springer, New York.

Quadt, D., Kuhn, H., 2005. Conceptual framework for lot-sizing and scheduling of flexible flow lines. International Journal of Production Research 43 (11), 2291–2308.

Quadt, D., Kuhn, H., 2008. Capacitated lot-sizing with extensions: a review. 4OR 6 (1), 61–83.

Quadt, D., Kuhn, H., 2009. Capacitated lot-sizing and scheduling with parallel machines, back-orders and setup carry-over. Naval Research Logistics 56 (4), 366–384.

Ramezanian, R., Saidi-Mehrabad, M., Fattahi, P., 2013a. MIP formulation and heuristics for multi-stage capacitated lot-sizing and scheduling problem with availability constraints. Journal of Manufacturing Systems 32 (2), 392–401.

Ramezanian, R., Saidi-Mehrabad, M., Teimoury, E., 2013b. A mathematical model for integrating lot-sizing and scheduling problem in capacitated flow shop environments. International Journal of Advanced Manufacturing Technology 66 (1-4), 347–361.

Rocha, L. R., Ravetti, M. G., Meteus, G. R., Pardalos, P. M., 2008. Exact algorithms for a scheduling problem with unrelated parallel machines and sequence and machine-dependent setup times. Computers & Operations Research 35 (4), 1250–1264.

Rohaninejad, M., Kheirkhah, A., Fattahi, P., 2015. Simultaneous lot-sizing and scheduling in flexible job shop problems. International Journal of Advanced Manufacturing Technology 78 (1), 1–18.

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

Santos, M. O., Almada-Lobo, B., 2012. Integrated pulp and paper mill planning and scheduling. Computers & Industrial Engineering 63 (1), 1–12.

Schrage, L., 1982. The multiproduct lot scheduling problem. In: Dempster, M., Lenstra, J., Rinnooy Kan, A. (Eds.), Deterministic and stochastic scheduling. Kluwer Academic Publishers, Dordrecht, Holland, pp. 233–244.

Seeanner, F., 2013. Multi-stage simultaneous lot-sizing and scheduling – planning of flow lines with shifting bottlenecks. Produktion und Logistik. Springer Gabler, Wiesbaden.

Seeanner, F., Almada-Lobo, B., Meyr, H., 2013. Combining the principles of variable neighborhood decomposition search and the fix&optimize heuristic to solve multi-level lot-sizing and scheduling problems. Computers & Operations Research 40 (1), 303–317.

Seeanner, F., Meyr, H., 2013. Multi-stage simultaneous lot-sizing and scheduling for flow line production. OR Spectrum 35 (1), 33–73.

Shim, I., Kim, H., Doh, H., Lee, D., 2011. A two-stage heuristic for single machine capacitated lot-sizing and scheduling with sequence-dependent setup costs. Computers & Industrial Engineering 61 (4), 920–929.

Sikora, R., Chhajed, D., Shaw, M. J., 1996. Integrating the lot-sizing and sequencing decisions for scheduling a capacitated flow line. Computers and Industrial Engineering 30 (4), 659–679.

Sikora, R. T., 1996. A genetic algorithm for integrating lot-sizing and sequencing in scheduling a capacitated flow line. Computers and Industrial Engineering 30 (4), 969–981.

Smith-Daniels, V. L., Ritzman, L. P., 1988. A model for lot sizing and sequencing in process industries. Journal of Production Research 26 (4), 647–674.

Smith-Daniels, V. L., Smith-Daniels, D. E., 1986. A mixed integer programming model for lot sizing and sequencing packaging lines in the process industries. IIE Transactions 18 (3), 278–285.

Stadtler, H., 2011. Multi-level single machine lot-sizing and scheduling with zero lead times. European Journal of Operational Research 209 (3), 241–252.

Stadtler, H., Sahling, F., 2013. A lot-sizing and scheduling model for multi-stage flow lines with zero lead times. European Journal of Operational Research 225 (3), 404–419.

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

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

Supithak, W., Liman, S. D., Montes, E. J., 2010. Lot-sizing and scheduling problem with earliness tardiness and setup penalties. Computers & Industrial Engineering 58 (3), 363–372.

Tempelmeier, H., Buschkühl, L., 2008. Dynamic multi-machine lotsizing and sequencing with simultaneous scheduling of a common setup resource. International Journal of Production Economics 113 (1), 401–412.

Tempelmeier, H., Copil, K., 2016. Capacitated lot sizing with parallel machines, sequence-dependent setups and a common setup operator. OR Spectrum 38 (4), 819–847.

Tiacci, L., Saetta, S., 2012. Demand forecasting, lot sizing and scheduling on a rolling horizon basis. International Journal of Production Economics 140 (2), 803–814.

Toledo, C. F. M., Arantes, M. d. S., de Oliveira, R. R. R., Almada-Lobo, B., 2013. Glass container production scheduling through hybrid multi-population based evolutionary algorithm. Applied Soft Computing 13 (3), 1352–1364.

Toledo, C. F. M., de Jesus Filho, J. E. F., França, P. M., 2008a. Meta-heuristic approaches for a soft drink industry problem. In: IEEE International Conference on Emerging Technologies and Factory Automation, 2008. pp. 1384–1391.

Toledo, C. F. M., de Oliveira, L., de Freitas Pereira, R., França, P. M., Morabito, R., 2014. A genetic algorithm/mathematical programming approach to solve a two-level soft drink production problem. Computers & Operations Research 48 (1), 40–52.

Toledo, C. F. M., de Oliveira, L., de Oliveira, R. R. R., Pereira, M. R., 2010. Parallel genetic algorithm approaches applied to solve a synchronized and integrated lot sizing and scheduling problem. In: Proceedings of the 2010 ACM Symposium on Applied Computing. New York, pp. 1148–1152.

Toledo, C. F. M., França, P. M., Morabito, R., Kimms, A., 2009. Multi-population genetic algorithm to solve the synchronized and integrated two-level lot sizing and scheduling problem. International Journal of Production Research 47 (11), 3097–3119.

Toledo, C. F. M., França, P. M., Rosa, K. A., 2008b. Evaluating genetic algorithms with different population structures on a lot sizing and scheduling problem. In: Proceedings of the 2008 ACM Symposium on Applied Computing (Fortaleza, Ceara, Brazil). pp. 1777–1781.

Toledo, C. F. M., Kimms, A., França, P. M., Morabito, R., 2006. A mathematical model for the synchronized and integrated two-level lot sizing and scheduling problem, University of Campinas, Brazil.

Toledo, C. F. M., Kimms, A., França, P. M., Morabito, R., 2015. The synchronized and integrated two-level lot sizing and scheduling problem: Evaluating the generalized mathematical model. Mathematical Problems in Engineering Article ID 182781.

Toso, E. A. V., Morabito, R., Clark, A. R., 2009. Lot sizing and sequencing optimisation at an animal-feed plant. Computers & Industrial Engineering 57 (3), 813–821.

Transchel, S., Minner, S., Kallrath, J., Löhndorf, N., Eberhard, U., 2011. A hybrid general lot-sizing and scheduling formulation for a production process with a two-stage product structure. International Journal of Production Research 49 (9), 2463–2480.

Urrutia, E. D. G., Aggoune, R., Dauzère-Pérères, S., 2014. Solving the integrated lot-sizing and job-shop scheduling problem. International Journal of Production Research 17 (52), 5236–5254.

van Eijl, C. A., van Hoesel, C. P. M., 1997. On the discrete lot-sizing and scheduling problem with Wagner-Whitin costs. Operations Research Letters 20 (1), 7–13.

van Hoesel, S., Kolen, A., 1994. A linear description of the discrete lot-sizing and scheduling problem. European Journal of Operational Research 75 (2), 342–353.

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

Wagner, H. M., Whitin, T. M., 1958. Dynamic version of the economic lot size model. Management Science 5 (1), 89–96.

Wolosewicz, C., Dauzère-Pérès, S., Aggoune, R., 2015. A Lagrangian heuristic for an integrated lot-sizing and fixed scheduling problem. European Journal of Operational Research 244 (1), 3–12.

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

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

Xiao, J., Yang, H., Zhang, C., Zheng, L., Gupta, J. N. D., 2015. A hybrid lagrangian-simulated annealing-based heuristic for the parallel-machine capacitated lot-sizing and scheduling problem with sequence-dependent setup times. Computer & Operations Research 63, 72–82.

Xiao, J., Zhang, C., Zheng, L., Gupta, J. N. D., 2013. MIP-based fix-and-optimise algorithms for the parallel machine capacitated lot-sizing and scheduling problem. International Journal of Production Research 51 (16), 5011–5028.

Zhu, X., Wilhelm, W. E., 2006. Scheduling and lot sizing with sequence-dependent setup: a literature review. IIE Transactions 38 (11), 987–1007.

3 Simultaneous lotsizing and scheduling considering secondary resources

Abstract[13] Typical simultaneous lotsizing and scheduling models consider the limited capacity of the production system by respecting a maximum time the respective machines or production lines can be available. Further types of resources necessary for production — like setup tools, setup operators or raw materials — may become bottlenecks and thus cannot be neglected in optimization models. These are called “secondary resources”. This paper provides a structured overview of the literature on simultaneous lotsizing and scheduling involving secondary resources. The proposed classification yields for the first time a unified view of scarce production factors. The insights about different types of secondary resources help to develop a new model formulation generalizing and extending the currently used approaches that are specific for some settings. Some illustrative examples demonstrate the functional principle and flexibility of this new formulation which can thus be used for a wide range of applications.

Keywords Scheduling, Dynamic lotsizing, Secondary resources, Mixed integer programming

3.1 Introduction

As it has been shown by the literature review of Copil et al. (2017) there has been a great research interest in simultaneous lotsizing and scheduling over the last decades. The formulated models typically consider one or just a few production stages. Each production stage may consist of one or more parallel machines (often aggregated to production flow lines if the sequence of machines is fixed and identical for all products) with scarce capacities. Furthermore, setup times and costs, which may be sequence-dependent, occur due to changeovers from one product to another. Product-specific demand is given per period and varies dynamically over time. If large lotsizes are built, the products have to be stored, what causes inventory holding costs. Many of the current model formulations are directly motivated by practical applications. Due to improvements in modeling knowledge, solution techniques and computing power it is now possible to represent industrial challenges in a more detailed manner.

Nevertheless, most of these simultaneous lotsizing and scheduling models consider the production capacity of these machines or production lines as the only limiting factor. Just a few also take the

[13]This paper has been written by Martin Wörbelauer, Herbert Meyr (Department of Supply Chain Management, University of Hohenheim, Stuttgart, Germany) and Bernardo Almada-Lobo (INESC-TEC, Faculdade de Engenharia, Universidade do Porto, Portugal). The paper has been submitted to OR Spectrum and the review is still in process.

capacity of one or more additional, potentially scarce production factors, like raw materials or setup operators necessary to perform changeover operations, into account. Such further production factors with limited capacity are called *"secondary resources"* (SRs; see Copil et al. 2017). If secondary resources were neglected in the planning process, resulting plans might become infeasible for real-world industrial applications. For instance, if two or more setups are scheduled in parallel on different lines, but there is only one setup operator available capable of performing at most one setup at the same time, this would result in an infeasible production plan. As can also be learned from this example, secondary resources usually affect several parallel machines or production lines simultaneously.

This paper provides a structured literature overview on simultaneous lotsizing and scheduling models which take secondary resources into account. The different types of resources that appear in different industrial settings are clustered using a unified classification. This overview will illustrate that — with respect to the secondary resources — the models of the literature are very specialized and suffer from lack of generality. For example, there are models which only consider a single setup operator or just secondary production resources. In such cases it is not possible to represent production scenarios with two setup operators or limitations in the supply of raw materials. Thus, this paper further introduces a general model formulation which is capable of handling all types of secondary resources addressed in the literature until now. Additionally, it also incorporates some functionalities which have not been represented in the literature so far, but seem reasonable for production planning as, for example, the splitting of setups into dismounting and mounting operations. Such modeling features do allow more flexible and thus more realistic production plans.

The presented model is based on the general lotsizing and scheduling problem (GLSP) of Fleischmann and Meyr (1997) and its single-stage extension for parallel production lines (GLSPPL) by Meyr (2002). It relies on a discrete time grid consisting of so-called "macroperiods". In a macroperiod multiple setups are possible. Nevertheless, for a detailed representation of the product sequence the model also uses "microperiods" with flexible length. In a microperiod at most one setup is possible. We use this microperiod structure to assure that, for example, a setup operator can be scheduled on at most a single production line at the same point in time. Since the GLSP is based on macroperiods it is called a "large-bucket" model.

The literature review will be given in the following section. It closes with a short discussion of the presented models and motivates the need for a more general and extended formulation. In Section 3.3, we modify the GLSPPL in a way that a common, but flexible microperiod time grid is simultaneously used for all parallel lines. This builds the basis to synchronize the usage of different types of secondary resources in Section 3.4. Section 3.5 shows how additional features that may become relevant in real-world applications can be incorporated into the new model. Numerical examples, which demonstrate the flexibility and broad applicability of the new model, are presented in Section 3.6. Finally, Section 3.7 provides a brief summary and identifies opportunities for future research.

3.2 Literature review

In the following, we review simultaneous lotsizing and scheduling models which incorporate SRs. We concentrate on characteristics that are important in the SR context. Additional, more general information about the presented models can be found in Copil et al. (2017). The two, probably most important characteristics are the "shareability" and "substitutability" of SRs. Shareability concerns the question whether a secondary resource can be used on only a single or on several production lines in parallel (i.e., at the same point in time) and whether it can only be used once or several times. We denote an SR as

- *"disjunctive"* if it can only serve a single production line at a single point in time and if it does not become part of the final product (e.g., a setup tool or a setup operator). A disjunctive SR can be used several times consecutively, even on different production lines.
- *"cumulative"* if it can serve several production lines simultaneously and does become part of the final product (e.g., fluid raw materials). Thus, a cumulative SR is consumed and can only be used once.

Substitutability distinguishes whether only a single type of SR could be applied to execute a certain setup or production process (denoted as *"without substitutes"*) or whether several different types of SRs do exist which could alternatively be applied (*"with substitutes"*). High- and low-skilled workers can serve as an example: a complex changeover process might only be executed properly by high-skilled operators (i.e., low-skilled operators cannot serve as substitutes), whereas simple changeovers could be executed by both types of workers alternatively.

The review is structured on the basis of these resource characteristics into the three subsections "disjunctive resources without substitutes" (Sect. 3.2.1), "disjunctive resources with substitutes" (Sect. 3.2.2) and "cumulative resources without substitutes" (Sect. 3.2.3). We are not aware of work concerning the remaining combination although industrial applications comprising cumulative resources with substitutes do certainly exist, for example, if scarce multi-purpose raw materials are involved. Finally, Table 3.2 (page 83) of Sect. 3.2.4 gives an overview. It further classifies the work on SRs in simultaneous lotsizing and scheduling by summarizing additional SR-relevant characteristics that have been identified and discussed in the preceding sections. This helps us to derive shortcomings of the current state of the art and to motivate the new model to be introduced in Sect. 3.4.

3.2.1 Disjunctive resources without substitutes

This subsection examines models which incorporate disjunctive SRs such as setup operators, which perform the changeovers, or cutting tools, which are necessary for production. These disjunctive resources can only be assigned to one production line at the same point in time. These resources are used but not consumed, i.e., they do not become part of the final product. Additionally, only resources without substitutes are considered. That means, all publications presented in this subsection assume that for each *process* (changeover from product i to j, conservation of the setup state of product j

(standby) and production of product j) of the production line it is explicitly known which resources are necessary. For example, there is no decision on which setup operator (e.g., operator A or B with different skill levels) performs a changeover from product i to j.

Lasdon and Terjung (1971) present a discrete lotsizing and scheduling problem (DLSP)[14] formulation for a tire manufacturer. The main characteristic of the DLSP is the all-or-nothing assumption, i.e., a product is produced for a complete period or there is no production at all. The problem of the tire manufacturer comprises parallel, identical machines. To produce product j it is necessary that a machine is set up for this product and that an additional die (secondary resource) is available. Since the DLSP is a small-bucket model, the synchronization of the resources (assuring that there is no overlapping use of the same SR on two or more different lines) comes true using the given time structure, i.e., the resource is assigned for the complete period to a certain line. In a large-bucket model this approach is often too restrictive since the resource's usage on another line would be blocked for quite a long time. In addition, the model is extended to consider setup operators and other equipment which is needed to perform a changeover. Eppen and Martin (1987) propose a new solution approach for the basic model (without setup resources) of Lasdon and Terjung (1971).

A proportional lotsizing and scheduling problem (PLSP, see Drexl and Haase 1995) formulation which considers secondary resources is presented by Kimms and Drexl (1998, pp. 89f). The PLSP uses continuous lotsizes and provides the possibility to produce at most two different products per microperiod on condition that one of the products has already been set up in a previous period. The presented multi-stage model formulation neglects setup times and assumes that each product is assigned to exactly one line. Nevertheless, different products can be assigned to the same production line. Each product requires multiple resources which all must be in the correct setup state. Synchronization of the resources is performed using the microperiod time grid.

Another example from a tire manufacturer is given by Jans and Degraeve (2004). The model is formulated as a DLSP with setup times. Tires are produced using different heaters. There can be multiple identical replicates of a heater type. Additionally, a mold is always necessary to produce a tire. For each possible tire-heater combination a limited number of molds is available. Due to short microperiods the molds can be assigned to active tire-heater combinations for complete periods.

A problem from the injection molding sector is considered by Dastidar and Nagi (2005). The authors use a continuous setup lotsizing problem (CSLP) formulation (Karmarkar and Schrage 1985), i.e., continuous lotsizes are possible and the number of products is limited to at most one per microperiod. Different products are produced on multiple parallel machines. For each product-machine combination a bundle of different SRs (e.g., grinders, driers) is necessary. There can be multiple replicates of the same resource. Nevertheless at most one of these replicates is required to produce a product. Again, resources are assigned to the machines for complete periods. The model respects setup times. The resources are already necessary during the setups of the products.

Tempelmeier and Buschkühl (2008) consider a common setup resource in their PLSP formulation. That is, only one setup operator is responsible for all setups. Since each product is dedicated to just one

[14]This denomination has only later been introduced by Fleischmann (1990).

line, the setup operator is the only reason for a simultaneous planning of all lines. Continuous variables are used to record the beginning of a setup on a machine in each microperiod. Binary variables document the machine-visiting sequence of the setup operator. The setup operator has a given time budget per period which can be less than or equal to the production capacity in this period. Constraints assure that the starting time of each setup is later than the ending time of the preceding setup on the previous line. Tempelmeier and Copil (2016)[15] tackle the same problem but using a CLSD formulation. The capacitated lotsizing problem with sequence-dependent setups (CLSD) was presented first by Haase (1996). It is a large-bucket model and uses a numbering of the products within a macroperiod - similar to a tour of a traveling salesman problem. Tempelmeier and Copil (2016) use variables to track both the starting and the ending times of setup operations assuring that there is at most one setup in parallel. The model is adapted to allow production of a given product on more than just a single line in parallel. Furthermore, the authors propose approaches which make it possible to set up a product several times per macroperiod.

Santos and Almada-Lobo (2012) consider a problem from the pulp and paper mill industry using a GLSP formulation. Microperiod lengths are still flexible but identical across all lines. Black liquor and virgin pulp result from processing wood chips in a digester. The black liquor has to be concentrated in an evaporator and afterwards burned in a recovery boiler to produce energy. Evaporator and boiler both show limited capacities. This limitation has influence on the regular production since the virgin pulp's output is proportional to the black liquor's output. Since the SRs just perform a transformation process, we classify them as disjunctive resources. Nevertheless, synchronization is not necessary because the SRs are just used for a single production line. The capacity limitation is quite similar to Tempelmeier and Buschkühl (2008) with the difference that the capacities of the evaporator and the boiler are given in cubic meters per hour. Figueira et al. (2013) extend the objective function to maximize the steam output. Furlan et al. (2015) tackle the same problem as Santos and Almada-Lobo (2012), extend the model for parallel paper machines and present a new solution approach. This extension does not affect the SRs.

Mac Cawley (2014) proposes a model for wine bottling. Macroperiods help to schedule product families. Changeovers from one product family to another cause sequence-independent setup times. For each product family several product sequences are defined in advance. That means, the product sequencing task is reduced to decide which given sequence should be applied in which period. Therefore it is not possible to explicitly classify this model as a GLSP or CLSD. If there is production on a line, a crew is necessary. All crews are assumed as being identical. Furthermore, there is a maximum number of possible crews in each macroperiod t. A crew is always assigned to a line for a complete macroperiod. The number of crews can be extended or reduced.

[15] See also Copil (2016).

3.2.2 Disjunctive resources with substitutes

The models presented in this subsection also incorporate *disjunctive* resources, which can be used at most on one line at the same time. However, now it is possible to choose between different *substitute* resources for a single process, i.e., the resource is not fixed a priori for a given process.

A GLSP model which considers tools as SRs is presented by Almeder and Almada-Lobo (2011). It is predefined which tools (substitute resources) could be used for a certain product-machine combination. Each product-machine combination requires just a single tool. The tool has to be available for the complete time, starting from the changeover before production and ending with the changeover after production. The first step of the synchronization process is to determine which resources are actually used. To accomplish this, the variables for setup states, changeovers and production quantities are extended by tool indices. Based on these variables, it is possible to calculate the starting times of the microperiods, whose lengths can be different on the various production lines, and the tool release times. Further constraints help to avoid overlapping of the line-specific microperiods when the same tool is used. The authors also propose a CLSD formulation which models SRs in a similar way.

Seeanner (2013, pp. 144-148) considers multiple different setup operators in a multi-stage GLSP formulation. For each setup it can be defined which setup operators are capable of performing this operation, i.e., substitutes are possible. Binary variables record which setup operator actually performs a setup. Thus, it is simple to assure that a setup operator is at most assigned to a single line per microperiod. This approach is valid because the multi-stage GLSP has an identical microperiod structure across all lines. Since it is possible to start a setup in microperiod $s-1$ and finish it in microperiod s (period-overlapping setup operations), further synchronization is necessary. Additional constraints assure that a microperiod is long enough to finish the setup that has been started in the preceding microperiod.

Copil (2016, pp. 121-137) adapts the model presented by Tempelmeier and Copil (2016) to consider multiple setup operators. Every setup operator is capable of performing each changeover. Nevertheless, setup times depend on the skill-levels of the deployed setup operators. The model is further extended to represent a practical application of a food producing company in a more detailed way.

3.2.3 Cumulative resources without substitutes

In the following, we describe models which consider *cumulative* resources, i.e., resources which can be used on more than one line at the same time. These resources are consumed. Thus they become part of the final products. Furthermore, the models are *without substitutes*, i.e., it is fixed which resources have to be used for a certain process.

Kimms and Drexl (1998, pp. 90f) propose a second extension of their PLSP model (c.f. Section 3.2.1). Here they introduce scarce SRs which can be consumed by multiple lines in parallel. Parameters define the resource consumption during production. Capacities of these resources are given per interval of periods (e.g., five periods).

Göthe-Lundgren et al. (2002) present a DLSP formulation for an oil refinery. The multi-stage model considers different production units which can be run in different modes. A “run-mode” defines the materials, which are consumed as input, and products, which are generated as output. Changeovers between run-modes can be interpreted as setups. The oil refinery consists of a distillation unit and two different hydro-treatment units. Each run-mode of a hydro-treatment unit needs a different amount of hydrogen, but the capacity of generating hydrogen is limited. Therefore, not every combination of run-modes is possible. Furthermore, during the production process unrequested sulphur is generated. The capacity to handle this undesired output product is also limited and restricts the choice of run-modes. In the general model formulation it is possible to consider R different SRs. Constraints assure that none of the given resource capacities is violated in any period. Persson et al. (2004) modify the model to consider sequence-dependent changeover costs when switching the run-mode.

Seeanner (2013, pp. 143f) extends his multi-stage GLSP formulation (c.f. Section 3.2.2) to also consider raw materials. Parameters define the consumption of each raw material during the production of one unit of a product. The overall production may not exceed a given capacity of each resource.

A group of models can be identified which — either directly in the model formulation or as part of the solution approach — represent a two-stage production process as a single-stage formulation with SRs. These models will be described in the following:

A GLSP formulation for a problem of the beverage industry is presented by Ferreira et al. (2009). The production scenario consists of multiple tanks and bottling lines. The tanks are used to prepare different flavored liquids which are packaged using the bottling lines. Only one flavor can be prepared in a tank at the same time. Due to technical reasons minimum liquid quantities have to be assured at the tank filling processes. A tank can be connected to several bottling lines at the same time. Sequence-dependent setup times and costs are considered for the changeovers of the bottling lines (e.g., setup of another bottle size) and for the changeovers of the flavors in the tanks. Only empty tanks can be refilled. First, the authors handle this problem using a two-stage model formulation. Additionally, they propose a solution approach based on a single-stage formulation with SRs. The single-stage model also uses binary variables indicating which flavor is in a tank in a certain microperiod. However, it differs from the two-stage model because these variables do not influence the objective function. After having solved the single-stage formulation, the resulting setups are used to constrain the two-stage model.

The formulation of Ferreira et al. (2010) is again based on the GLSP. However, it just takes a single bottling line into account. Although multiple tanks are connected to this line in real-world, it is sufficient to model just a single tank without setup times. The reason is that the next tank can already be prepared while another one is still supplying the bottling line. Nevertheless, minimum fill levels and maximum capacities of the tanks must be respected. Binary variables are used to track which flavor is in each tank in a certain period.

Ferreira et al. (2012) also examine the beverage problem and use a single-stage GLSP formulation with the tankfuls as SRs. The authors consider a fixed assignment of tanks to bottling lines. To put a resulting plan into practice, some kind of synchronization between tank filling and bottling is necessary. Otherwise, bottling on a line could start before its supplying tank had sufficiently been filled. This is reached by respecting artificial setup times in the single-stage model, which are taken as the maximum of the tank filling setup time and the bottling setup time. The authors also propose a CLSD formulation with the same approach for synchronization. Maldonado et al. (2014) present different CLSD formulations for the same problem. Note that in the case of cumulative SRs the need for synchronization results from simplifying a two-stage production process into a single-stage one. Whereas the synchronization of disjunctive resources in Sections 3.2.1 and 3.2.2 was necessary to avoid simultaneous usage of the same SR on several parallel lines.

Almada-Lobo et al. (2010) formulate a CSLP to handle a planning task in the glass container industry. Multiple parallel molding machines are supplied with melted glass by a furnace. The furnace's capacity is given in tons per period and the furnace can be inactive, but only at the end of the planning horizon. If the furnace is active, the complete capacity should be used. Otherwise, penalty costs incur. A single-stage formulation, which incorporates the melted glass as an SR necessary for production and setups, is used. Compared to the aforementioned problems from the beverage industry, the information which glass type to melt is known in advance from a mid-term planning. Toledo et al. (2013) restrict the lotsizes to discrete values for the same problem. Furthermore, melted glass still flows during idle times and setups and is returned into the furnace, i.e., there is no resource consumption during these times.

Camargo et al. (2012) consider a problem from the process industry. There exist one upstream machine and multiple downstream machines. The products which are produced on the downstream machines are grouped to product families. Each family needs one kind of SR. In each microperiod at most one SR can be produced on the upstream machine. To begin with the authors present a GLSP formulation. Setup times and costs of the upstream machine are omitted and a variable is introduced indicating which SR is produced on the upstream machine in a certain period. Secondary resources' maximum capacities are defined in advance per microperiod (flexible length). The authors do not discuss how such capacities of flexible periods of time could be determined for real-world applications. A capacity check assures that these capacities are not exceeded by consumption of downstream machines. Again, synchronization has to ensure that all downstream machines can only use the SR that is currently produced on the upstream machine. This synchronization is performed using a common time grid for all machines. Thus, the starting and ending times of the microperiods are tracked by variables. Additionally, the authors propose another formulation based on the CLSD. Camargo et al. (2014) adapt the problem for the yarn production using a GLSP formulation.

3.2.4 Classification scheme and discussion

Table 3.2 on page 83 further classifies and summarizes the models described. Table 3.1 gives an overview of the acronyms used. Besides "shareability" and "substitutability" additional attributes, which distinguish the various approaches found in the literature, help to characterize SRs' usage in greater detail. These are:

Basic model: This attribute specifies which basic model has been extended for the use of secondary resources. Possible values are *DLSP*, *CSLP*, *PLSP*, *CLSD*, *GLSP* or just *big bucket* if neither an explicit assignment to CLSD nor to GLSP is possible.

States concerned: Secondary resources may be necessary for *production* (p) or *setups* (s) or while setup states of production lines are *conserved* (c). Note that these tasks "production", "setup" and

Table 3.1: Classification scheme of models considering secondary resources

Description	Attribute	Potential value	Acronym
Bm	**Basic model**	discrete lotsizing and scheduling problem	DLSP
		continuous setup lotsizing problem	CSLP
		proportional lotsizing and scheduling problem	PLSP
		capacitated lotsizing problem with sequence-dependent setups	CLSD
		general lotsizing and scheduling problem	GLSP
		big bucket	big bucket
Share	**Shareability**	disjunctive	disj
		cumulative	cum
		disjunctive and cumulative	disj/cum
Sub	**Substitutability**	without	wo
		with (includes without)	w
Sc	**States concerned**	production	p
		setup	s
		conservation of setup state	c
		addressed states need the same substitutable SRs	s-p-c
		addressed states may use different substitutable SRs	s:p:c
Qr	**Quantity of different resources**	limited number	#
		unlimited	fr
Rpr	**Resource-to-process-relation**	one-to-one	1:1
		one-to-many	1:pr
		many-to-one	r:1
		many-to-many	r:pr
I	**Industry**	automobile industry	AI
		beverage industry	BI
		consumer goods industry	CGI
		food industry	FI
		process industry	PI
		semiconductor industry	SI

"conservation of the setup state" in the following will be termed as *"states"* if they are generally addressed and as *"processes"* if they are addressed in connection with a production line and a product that is produced, that is set up or whose setup state is merely conserved. This denomination has also already been used since the beginning of Section 3.2.1. If substitutes are not possible (*Sub*=*wo*), the assignment of an SR to a process is unique. Thus it suffices to list all tasks a model is able to consider. However, if substitutes do exist (*Sub*=*w*), two cases may occur:

1. Either the same SR has to be used for several subsequent processes although a substitute would exist. For example, if two alternative tools could be installed during a setup and then be used for production, either the first or the second one had to be used for *both* processes.

2. Or the substitutes can freely be exchanged. For example, if two workers had the capabilities to

execute a setup and monitor the subsequent production process, the first one could do the setup and the second one the monitoring.

In the following, case 1 will be marked by an "-" and case 2 by an ":". Thus, the first example would be abbreviated as s-p whereas the second example would be denoted as s:p.

Quantity of different resources: There exist model formulations which consider just a *limited number* of non-identical secondary resources. In most of these cases, there is only one resource like, e.g., a single setup operator who is responsible for all setups. In order to keep the classification scheme compact, our notation will not distinguish whether just a single resource or multiple identical replicates of this resource are available. However it will be marked if models provide the possibility to consider an *unlimited* number of different resources.

Resource-to-process-relation: This attribute provides information about the relation between resources *r* (only non-identical resources are considered here) and processes *pr*. There can be a *one-to-one* (1:1) assignment, i.e., each resource is uniquely assigned to a single process only. For example, a certain mold can only be used to produce a single type of tire. It is also possible that an SR can be assigned to several processes, called *one-to-many* (1:*pr*), for example, if a setup operator is able to execute two different setup operations. If multiple types of SRs are necessary for a single process this is named *many-to-one* (*r*:1). For instance, this is the case if both a tool and a setup operator are necessary to perform a certain setup operation. Furthermore, there can be a *many-to-many* (*r:pr*) assignment, for example, if a tool and a certain setup operator are needed for some setup operation and the same setup operator is also required for another setup operation. The different relationships between SRs and processes are summarized in Figure 3.1. Note that all relationships can be represented by a general formulation that is able to model the *r:pr* relation.

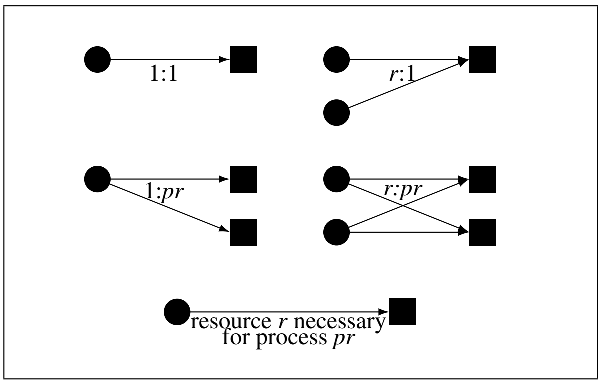


Figure 3.1: Resource-to-process-relation

Industry: Most of the models discussed are motivated by real-world applications. These stem from *automotive, beverage, consumer goods, food, process and semiconductor industries*. These industries typically rely on a flow line organization. This is the reason why we will rather use the term "production lines" than "machines" in the remainder of the paper.

When inspecting Table 3.2 and reconsidering Sections 3.2.1 – 3.2.3 the following conclusions can be drawn:

Table 3.2: Literature overview: models considering secondary resources

Author	Bm	Share	Sub	Sc	Qr	Rpr	I	Comments
Lasdon and Terjung (1971)	DLSP	disj	wo	p,s	fr	1:1	PI	for setup: Qr =1 and Rpr = 1:pr
Eppen and Martin (1987)	DLSP	disj	wo	p	fr	1:1	PI	
Kimms and Drexl (1998, pp. 89f)	PLSP	disj	wo	p,c	fr	r:pr	-	SR must be in the correct setup state; capacity of SR ≤ period capacity; multi-stage
Jans and Degraeve (2004)	DLSP	disj	wo	p	fr	1:1	PI	
Dastidar and Nagi (2005)	CSLP	disj	wo	s,p,c	fr	r:pr	CGI	
Tempelmeier and Buschkühl (2008)	PLSP	disj	wo	s	1	1:pr	AI	capacity of SR ≤ period capacity
Santos and Almada-Lobo (2012)	GLSP	disj	wo	p	2	1:pr	PI	SR restricts single line; capacity of SR ≤ period capacity; two-stage
Figueira et al. (2013)	GLSP	disj	wo	p	2	1:pr	PI	SR restricts single line; capacity of SR ≤ period capacity; two-stage
Mac Cawley (2014)	big bucket	disj	wo	s,p	1	1:pr	BI	
Tempelmeier and Copil (2016)	CLSD	disj	wo	s	1	1:pr	FI	
Furlan et al. (2015)	GLSP	disj	wo	p	2	1:pr	PI	SR restricts single line; capacity of SR ≤ period capacity; two-stage
Almeder and Almada-Lobo (2011)	GLSP & CLSD	disj	w	s-p-c	fr	1:pr	SI	
Seeanner (2013, pp. 144-148)	GLSP	disj	w	s	fr	1:pr	-	multi-stage
Copil (2016, pp. 121-137)	CLSD	disj	w	s	fr	1:pr	FI	
Kimms and Drexl (1998, pp. 90f)	PLSP	cum	wo	p	fr	r:pr	-	multi-stage
Göthe-Lundgren et al. (2002)	DLSP	cum	wo	p	fr	r:pr	PI	multi-stage
Persson et al. (2004)	DLSP	cum	wo	p	fr	r:pr	PI	multi-stage
Ferreira et al. (2009)	GLSP	cum	wo	p	fr	1:pr	BI	use of SR as solution approach for two-stage model
Almada-Lobo et al. (2010)	CSLP	cum	wo	s,p	1	1:pr	PI	
Ferreira et al. (2010)	GLSP	cum	wo	p	fr	1:pr	BI	just one production line; SR used to represent a two-stage model
Camargo et al. (2012)	GLSP & CLSD	cum	wo	p	fr	1:pr	PI	SR used to represent a two-stage model

Author	Bm	Share	Sub	Sc	Qr	Rpr	I	Comments
Ferreira et al. (2012)	GLSP & CLSD	cum	wo	p	fr	1:pr	BI	SR used to represent a two-stage model
Seeanner (2013, pp. 143f)	GLSP	cum	wo	p	fr	r:pr	-	multi-stage
Toledo et al. (2013)	CSLP	cum	wo	p,c	1	1:pr	PI	
Camargo et al. (2014)	GLSP	cum	wo	p	fr	1:pr	CGI	SR used to represent a two-stage model
Maldonado et al. (2014)	CLSD	cum	wo	p	fr	1:pr	BI	SR used to represent a two-stage model
New model formulation	GLSP	disj/cum	w	s:p:c/s-p-c	fr	r:pr	-	

- **Model for cumulative resources with substitutes is missing:** We identified models for three *types* of resources: disjunctive resources without and with substitutes and cumulative resources without substitutes. To the best of our knowledge, there does not exist any publication taking *cumulative* resources with *substitutes* into account. Nevertheless, being able to model such a situation might offer significant advantages as shown in the following example. In the spinning industry, one must determine the size and sequence of yarn production lots as well as which cotton bales (secondary resources) will provide fiber blend that ensures quality attributes (e.g., grade, color and fiber lengths) to produce the required yarns. Each blend must be set by means of different combinations of cotton bales. Nevertheless, different cotton bale combinations (substitutes) can be used to fulfill the quality attributes of a yarn.

- **Models cannot be applied to different scenarios:** The presented models are very specialized. This can easily be explained because most of these models have been tailored to a specific, practical planning problem of a certain company. Then, the advantage is that the model is not bloated by extra features which are needless for this respective company. Nevertheless, it shows the disadvantage that another company with a slightly different planning problem may not be able to also apply such a model.

- **Models might be too complex:** One very general model formulation is the one of Seeanner (2013). His formulation of disjunctive resources with substitutes can also be used for disjunctive resources without substitutes by a mere variation of the input parameters. However, in this latter case the model includes more variables than actually necessary.

- **Concerned states are limited:** Another quite general formulation is the one of Almeder and Almada-Lobo (2011). However, it also shows the shortcoming of many other models that with s-p-c *always* the same SRs have to be used for a sequence of setup, production and conservation. A more general model would be preferable that could handle an s:p:c situation, too, if this occurred in real-world. The model presented in Sections 3.3-3.5 will be capable of handling both cases.

- **Resource-to-process-relation is limited:** When looking at Table 3.2 it can be seen that models for the most general resource-to-process-relation *Rpr* = *r:pr* occur quite seldom. No model can be found at all, which is able to represent disjunctive resources with substitutes in a many-to-many relation. Nevertheless, such scenarios are easy to imagine. For example, think of a situation where a tool out of a set of alternative tools and an additional worker with a minimum skill-level out of a group of incrementally trained employees are both necessary for production.

To sum up, a general model must be able to represent disjunctive and cumulative SRs with and without substitutes. Nevertheless — to keep the model lean and to leverage solvability — it should be easy to waive unnecessary parts of the model if they are irrelevant for a certain real-world application. It should be possible to define resource usage for each state separately, but also to enforce retention of the same SR. Furthermore, a process should be allowed to require more than just a single SR. Additionally, SRs should be able to be assigned to several processes simultaneously. Such a general model will be presented in the following sections. A classification of this model can be found in the last row of Table 3.2.

3.3 Basic model formulation

The first step is to formulate a model that can be used as basis for all types of SRs. One important aspect, which must already be considered in the basic formulation, is the synchronization of disjunctive resources. To accomplish this we will build on top of the GLSP for heterogeneous parallel production lines (GLSPPL) of Meyr (2002) and Meyr and Mann (2013). This is a single-stage formulation (for multiple stages see the outlook in Section 3.7). The GLSPPL is adapted to a common time structure across all lines as it has been done in the GLSP for multiple production stages (GLSPMS) in order to synchronize the different stages (c.f., Meyr 2004, Seeanner and Meyr 2013 and Seeanner et al. 2013). Just one state is allowed per line and microperiod. Thus, the synchronization of disjunctive SRs across the parallel lines of a single stage of production can also be based on a common microperiod time grid. To assure more flexibility, period-overlapping setup times (so-called "continuous setups") are allowed. Suerie (2005) applies this approach of spreading long setup times over several consecutive periods of large-bucket and small-bucket models. We adapt his formulation to the GLSPPL as it has in a similar way been done by Seeanner (2013) concerning the GLSPMS.

The production system comprises multiple parallel production lines l ($l = 1,2,...,L$). These production lines are used to produce several "real" products j ($j = 1,2,...,J$). An additional product $j = 0$ is added to represent a "neutral state" of a line when it is not set up for a certain real product. There does not exist any demand for this fictitious dummy product. For all other products $j > 0$ a demand d_{jt} is given for each macroperiod t ($t = 1,2,...,T$). The production coefficient a_{lj} defines the production time which is necessary to produce one unit of product j on line l. Before production of a product j can take place on line l, a changeover from the previous product i to product j has to be done. These setups cause sequence-dependent setup costs s_{lij} and times st_{lij}. Shutdown costs to switch to the neutral state of a line l are indicated by s_{li0}. On the other hand, the activation of lines from the neutral state triggers startup costs s_{l0j}. If a line does not produce, it is called “idle”. If a line l is not shut down but nevertheless idle, the current setup state j is merely conserved. This conservation of setup states causes standby costs b_l per time unit. Holding costs h_j are accounted for inventory of each product $j > 0$ at the end of each macroperiod. Moreover, c_{lj} defines the production costs which incur when producing one unit of a product j on line l.

The planning horizon is divided into S microperiods. At most one product j can be produced in each microperiod s ($s = 1,...,S$). Thus, these microperiods are used to define the product sequence. Each macroperiod t consists of $|S_t|$ microperiods, whereat S_t defines the set of microperiods within macroperiod t. The first microperiod of a macroperiod has always a fixed starting time $\overline{w}_s$. Therefore, the lengths of macroperiods are defined by the starting times of fixed microperiods, which are subsumed by the set Φ. An additional microperiod $S+1$ is used to represent the end of the planning horizon. Macroperiod lengths represent the production capacities of the lines. Microperiod lengths are flexible and, in our representation, identical for all lines. The common time structure of all lines is realized using variables w_s which represent the starting times of the microperiods s.

Continuous variables x_{ljs} and $\overline{x}_{ljs}$ are used to measure the production quantity of a product j in microperiod s on line l and the idle time, where the setup state is conserved, respectively. Minimum lotsizes m_{lj} must be respected. I_{jt} denotes the inventory of product j at the end of macroperiod t. The binary variables $y_{ljs} \in \{0;1\}$ and $v_{ljs} \in \{0;1\}$ represent whether there is production of product j on line l in microperiod s and whether there is conservation of the setup state, respectively.

The following variables are necessary to consider continuous setups: x^f_{ls} defines the (potentially) fractional time of a setup spent on line l in microperiod s. The continuous variables z_{lijs} take the value 1 if a changeover from product i to product j on line l is completed in microperiod s. Otherwise, their value is 0. If a changeover from product i to product j is spread over several consecutive microperiods, all periods s of this changeover except for this last completion period are marked by a binary variable $z^c_{lijs} \in \{0;1\}$ taking on the value 1. These microperiods will be denoted as “to be continued (tbc)” in the following. Note that it is important for the resource consideration that the information about the concerned products is known, thus, the indices i and j in the variable z^c_{lijs} are necessary. Furthermore, two variables are used to accumulate fractional setup times. τ_{ls} determines a lower bound of the up to period s cumulated setup times and σ_{ls} defines an upper bound.

All parameters and variables used in the model are summarized in Table 3.3. The model formulation is stated below.

Objective function of the GLSPPL with a common time structure:

$$\text{Min} \sum_{t,j} h_j I_{jt} + \sum_{l,i,j \neq i,s} s_{lij} z_{lijs} + \sum_{l,j,s} c_{lj} x_{ljs} + \sum_{l,j,s} b_l \overline{x}_{ljs} \tag{3.1}$$

Table 3.3: Symbols of the GLSPPL with a common time structure

Indices and sets:	
$i, j = 1, ..., J$	products; $i, j = 0$ neutral product
$l = 1, ..., L$	production lines
$s = 1, ..., S$	microperiods
$s = S+1$	dummy microperiod modeling the end of the last macroperiod
$t = 1, ..., T$	macroperiods
S_t	set of microperiods s belonging to macroperiod t
Φ	set of all microperiods with fixed starting times
Data:	
a_{lj}	capacity consumption (time) needed to produce one unit of product j on line l
b_l	standby costs of line l (per time unit)
c_{lj}	production costs of product j (per unit) on line l
d_{jt}	demand of product j in macroperiod t (units)
h_j	holding costs of product j (per unit and per macroperiod)
I_{j0}	initial inventory of product j at the beginning of planning (units)
m_{lj}	minimum lotsize of product j (units) if produced on line l
m_{l0}	minimum time line l has to remain shut down
s_{lij}	setup cost of a changeover from product i to product j on line l
st_{lij}	setup time of a changeover from product i to product j on line l
v_{lj0}	equals 1 if the setup state of product j is conserved on line l at the beginning of planning (0 otherwise)
$\overline{w}_s$	starting time of fixed microperiod $s \in \Phi$
y_{lj0}	equals 1 if line l is set up for product j at the beginning of planning (0 otherwise)
z_{lij0}	equals 1 if a changeover from product i to product j has been completed on line l before the beginning of planning (0 otherwise)
$z^c_{lij0} = 0$	continued setups are not allowed before the beginning of planning
Variables:	
$I_{jt} \geq 0$	inventory of product j at the end of macroperiod t (units)
$v_{ljs} \in \{0; 1\}$	equals 1 if the setup state of product j is conserved on line l in microperiod s (0 otherwise)
$w_s \geq 0$	starting time of microperiod s
$\sigma_{ls} \geq 0$	cumulated setup time on line l until the end of microperiod s (upper bound)
$\tau_{ls} \geq 0$	cumulated setup time on line l until the end of microperiod s (lower bound)

$x_{ljs} \geq 0$	quantity of product j produced during microperiod s on line l (units)
$x^f_{ls} \geq 0$	(potentially) fractional setup time on line l in microperiod s
$\bar{x}_{ljs} \geq 0$	time used for conserving the setup state j on line l in microperiod s (idle time)
$y_{ljs} \in \{0;1\}$	equals 1 if production of product j takes place on line l in microperiod s (0 otherwise)
$z_{lijs} \geq 0$	equals 1 if a changeover from product i to product j is completed on line l in microperiod s (0 otherwise)
$z^c_{lijs} \in \{0;1\}$	setup to be continued (tbc); equals 1 if a changeover from product i to j takes place on line l in microperiod s, but is not yet completed in this microperiod (0 otherwise)

Constraints of the GLSPPL with a common time structure:

$$w_s = \overline{w}_s \qquad \forall s \in \Phi \qquad (3.2)$$

$$I_{jt} = I_{j,t-1} + \sum_{l,s \in S_t} x_{ljs} - d_{jt} \qquad \forall j,t \qquad (3.3)$$

$$\sum_{i,j \neq i} z^c_{lijs} + \sum_{i,j \neq i} z_{lijs} + \sum_j y_{ljs} + \sum_j v_{ljs} = 1 \qquad \forall l,s \qquad (3.4)$$

$$\sum_j a_{lj} x_{ljs} + \sum_j \overline{x}_{ljs} + x^f_{ls} = w_{s+1} - w_s \qquad \forall l,s \qquad (3.5)$$

$$a_{lj} x_{ljs} \leq \overline{w}_{S+1} y_{ljs} \qquad \forall j,l,s \qquad (3.6)$$

$$\overline{x}_{ljs} \leq \overline{w}_{S+1} v_{ljs} \qquad \forall j,l,s \qquad (3.7)$$

$$x^f_{ls} \leq \overline{w}_{S+1} \sum_{i,j \neq i} (z^c_{lijs} + z_{lijs}) \qquad \forall l,s \qquad (3.8)$$

$$\sum_{r=s}^{s+1} x_{ljr} \geq m_{lj} \sum_{i \neq j} z_{lij,s-1} \qquad \forall j,l,s \qquad (3.9)$$

$$y_{lj,s-1} + \sum_{i \neq j} z^c_{lji,s-1} + \sum_{i \neq j} z_{lij,s-1} + v_{lj,s-1} = y_{ljs} + \sum_{i \neq j} z^c_{ljis} + \sum_{i \neq j} z_{ljis} + v_{ljs} \qquad \forall j,l,s \qquad (3.10)$$

$$z^c_{lij,s-1} \leq z^c_{lijs} + z_{lijs} \qquad \forall l,i,j \neq i,s \qquad (3.11)$$

$$\tau_{ls} \geq \sum_{i,j \neq i} st_{lij} z_{lijs} \qquad \forall l,s \qquad (3.12)$$

$$\tau_{ls} \leq \tau_{l,s-1} + x^f_{ls} \qquad \forall l,s \qquad (3.13)$$

$$\tau_{ls} \leq x^f_{ls} + \overline{w}_{S+1} \sum_{i,j \neq i} z^c_{lij,s-1} \qquad \forall l,s \qquad (3.14)$$

$$\sigma_{ls} \geq \sigma_{l,s-1} + x^f_{ls} - \sum_{i,j \neq i} st_{lij} z_{lij,s-1} \qquad \forall l,s \qquad (3.15)$$

$$\sigma_{ls} \leq \sum_{i,j \neq i} st_{lij} z_{lijs} + \sum_{i,j \neq i} \overline{w}_{S+1} z^c_{lijs} \qquad \forall l,s \qquad (3.16)$$

$$z^c_{lijS} = 0 \qquad \forall l,i,j \neq i \qquad (3.17)$$

The objective function (3.1) minimizes the total costs, namely, the sum of holding costs, setup costs, production costs and standby costs.

Constraints (3.2) create the macroperiod time structure using the fixed starting times $\overline{w}_s$ of microperiods $s \in \Phi$. Equations (3.3) are the typical inventory balancing equations for each macroperiod. Exactly one of the states “setup to be continued”, “setup completion”, “production” or “conservation” is allowed per production line and microperiod (3.4). This restriction is important to synchronize SRs as will be shown later. Equations (3.5) define the length of microperiod s as the difference between the starting time of the following and the current microperiod. The time budget of such a microperiod must be completely used for either production or conservation of the setup state or for setups.

The values of the binary variables y_{ljs} and v_{ljs} are defined by constraints (3.6) and (3.7), respectively. If there is production $x_{ljs} > 0$ or conservation of a setup state $\overline{x}_{ljs} > 0$, the corresponding binary variable takes the value 1. Constraints (3.8) assure that positive setup time can only be charged if a setup takes place. In these three types of constraints, the planning horizon $\overline{w}_{S+1}$ serves as a big number linking the continuous with the binary variables.

Similarly to Koçlar and Süral (2005), it is sufficient to fulfill the minimum lotsizes during the first two microperiods after a setup (3.9). This approach offers more flexibility than just using x_{ljs} on the left-hand side. Thus, for instance, conservation of the setup state in the first microperiod after a setup is possible.

Equations (3.10) ensure the correct flow of the different states of a line. For example, if product j has been set up in period $s-1$ (left-hand side = 1), but is not needed any longer in following period ($y_{ljs} = v_{ljs} = 0$), a changeover to another product i has to be started in period s, which can either also be finished in period s (i.e., $\sum_{i \neq j} z_{ljis} = 1$) or has to be continued in period $s+1$ (i.e., $\sum_{i \neq j} z^c_{ljis} = 1$). Note that Equations (3.11) and not Equations (3.10) are responsible for the correct flow of the tbc-periods. Nevertheless, Equations (3.10) include a $\sum_{i \neq j} z^c_{lji,s-1}$ on the left-hand side. Otherwise, it would never be possible to switch from a tbc-period to the completion of a setup.[16] Further note that because of (3.10), in any optimal solution, the variables z_{lijs} will only take zero or one as values.

The remaining constraints are necessary to model the period-overlapping setups. Constraints (3.11) assure that a continuous setup is not interrupted. Once started ($z^c_{lij,s-1} = 1$) in or before period $s-1$, because of (3.4), it either has to be continued ($z^c_{lijs} = 1$) or finished ($z_{lijs} = 1$) in the following period s. Constraints (3.12) ensure that sufficient setup time τ_{ls} will be accumulated until the setup has been completed in period s. The accumulation is put into practice by constraints (3.13). It finally needs to reach st_{lij}, but can at most be increased from a preceding period to its subsequent period by the fractional setup time x^f_{ls}. Reading constraint (3.13) as $x^f_{ls} \geq \tau_{ls} - \tau_{l,s-1} \forall l,s$ clarifies why τ is called a “lower bound” for the setup time to be spent. Because of (3.5), its increment has to be reserved on line l for each continuous setup period affected. After a setup has been completed ($z_{lij,s-1} = 1$) in a period $s-1$, at the *beginning* of the next period s the cumulated setup time τ needs to be reset to zero. Constraints (3.14) do this indirectly by limiting the setup time τ_{ls}, that has been accumulated until the *end* of period s, to the fractional setup time x^f_{ls} accounted for period s. The added term $\overline{w}_{S+1} \sum_{i,j} z^c_{lij,s-1}$ turns this constraint non-active in tbc-periods.

[16]Using the same index sequence ji three times for $z^c_{lji,s-1}$, z^c_{ljis} and z_{ljis} allows (3.10) to carry the information, which product had been produced last, over all microperiods of a continued setup.

If the costs for conserving the setup state were very high, the model presented so far would artificially stretch the setup times and accumulate more fractional setup times x^f_{ls} than actually necessary. If this shall be prevented, penalty costs for x^f_{ls} could be imposed or constraints (3.15)–(3.17) could be introduced. Like τ_{ls} the variables σ_{ls} accumulate the fractional setup times. Constraints (3.15) force the σ_{ls} of period s to sum the preceding period's $\sigma_{l,s-1}$ and the current period's fractional setup time x^f_{ls} as long as the setup has not been completed in the preceding period ($\sum_{i,j} st_{lij} z_{lij,s-1} = 0$). If the setup has been completed, σ_{ls} is reset to zero. Reading (3.15) as $x^f_{ls} \leq \sigma_{ls} - \sigma_{l,s-1} + \sum_{i,j} st_{lij} z_{lij,s-1}$ illustrates why σ is called an "upper bound" on the fractional setup times. According to constraints (3.16), the accumulated setup time σ_{ls} itself is bounded by st_{lij} if a changeover from product i to product j has been completed ($\sum_{i,j} z_{lijs} = 1, \sum_{i,j} z^c_{lijs} = 0$) in period s. In tbc-periods ($\sum_{i,j} z^c_{lijs} = 1, \sum_{i,j} z_{lijs} = 0$), however, constraints (3.16) do not show any effect. Equations (3.17) finally forbid unfinished setups at the end of the planning horizon (see also Table 3.3).

3.4 Extension for secondary resources

The following subsections introduce practically relevant extensions to the base formulation to consider secondary resources. We structure the presentation according to the four different types of SRs that emerged from the literature review in Section 3.2. If all presented constraints are simultaneously used in a single model, each type of SR can be represented. We indicate this by using different indices for the different types of SRs. Of course, it is also possible to neglect some type of SR and omit its corresponding constraints if this was sufficient for a certain industrial application. Some additional optional features, helping to further adapt the model formulation to industrial needs, will be presented in Section 3.5.

3.4.1 Disjunctive resources without substitutes

The basic model stated up to now will in the following be extended to consider disjunctive resources without substitutes. An example would be a scenario with two setup operators who are necessary to perform setups, a tool which is bounded to the machines during setups, production and conservation of setup states and a worker who is responsible for the machine during production. In terms of our classification scheme of Section 3.2.4, a process may need several SRs, i.e., multiple resources are considered and the resource-to-process-relation is flexible (*r:pr*). Nevertheless, there are no substitutes.

There are several disjunctive resources $p = 1, 2, ..., P$. We assume that each resource is available for the complete planning horizon. For each potential process (conservation of the setup state of product j, production of product j and changeover from product i to j) it is possible to define whether resource p is necessary or not by introducing further binary parameters b^c_{jp}, b^p_{jp} and b^s_{ijp}, respectively. Of course, these parameters could also be declared as line-dependent if necessary. These additional parameters are summarized in Table 3.4 (new variables are not needed). The necessary additional constraints are stated below.

Table 3.4: Symbols for disjunctive resources without substitutes

Indices:	
$p = 1,...,P$	disjunctive resources without substitutes
Data:	
b^c_{jp}	$= 1$ if resource p is needed for conservation of setup state j (0 otherwise)
b^p_{jp}	$= 1$ if resource p is needed for production of product j (0 otherwise)
b^s_{ijp}	$= 1$ if resource p is needed for a changeover from product i to product j (0 otherwise)

Additional constraints for disjunctive resources without substitutes:

$$\sum_{l,i,j\neq i} b^s_{ijp}(z_{lijs} + z^c_{lijs}) + \sum_{l,j} b^p_{jp} y_{ljs} + \sum_{l,j} b^c_{jp} v_{ljs} \leq 1 \qquad \forall s, p \tag{3.18}$$

Constraints (3.18) avoid double usage of SRs and thus synchronize the use of SRs across all parallel lines. Since there is a common time structure with exactly one state per microperiod, it is sufficient to check that each SR p is used on at most one line in every microperiod s. Note that — now and in the following — for ease of simplicity we assume that there do not occur any transportation times if an SR is transferred from one production line to another.

Figure 3.2 shows an exemplary production plan with a common time structure and a single state per microperiod. If resource $p = 1$ is necessary for the setup on line 1 in microperiod 1, this resource cannot be used on another line in the same microperiod. A binary parameter like b^s_{ijp} ensures that several setups can be executed in the same microperiod if they use different SRs. Thus, both setups (from product 1 to 2 on line 1 and from product 3 to 4 on line 2) in microperiod $s = 1$ are possible if there are two different tools $p = 1$ and $p = 2$ with $b^s_{121} = b^s_{342} = 1$.

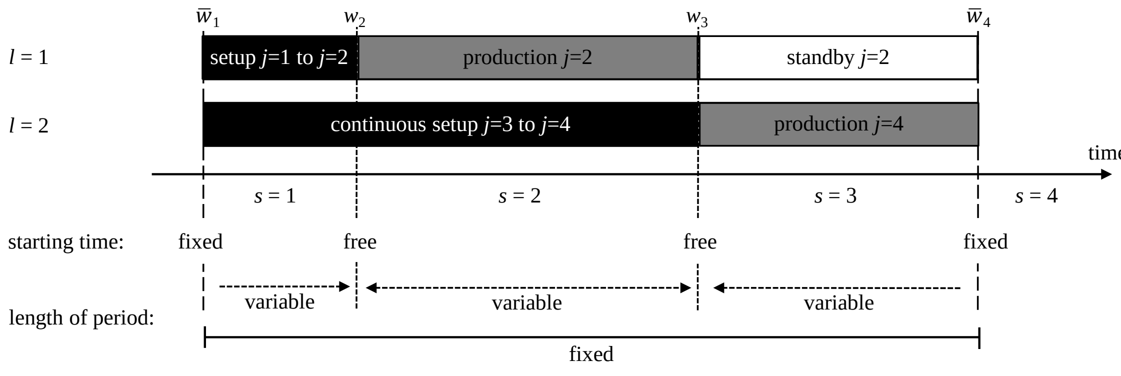
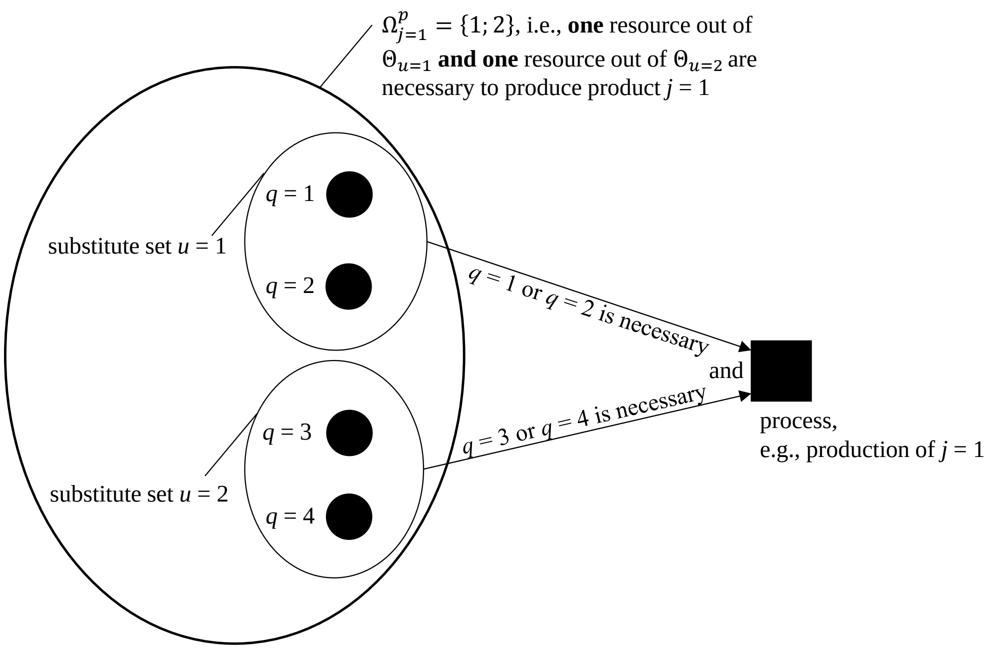


Figure 3.2: Example production plan with common time structure and exactly one state per line and microperiod

3.4.2 Disjunctive resources with substitutes

Now, the base model (Equations (3.1)–(3.17)) is extended to consider disjunctive resources with substitutes. For instance, it is possible to choose whether worker 1 or worker 2 performs a given setup. This opportunity is of particular interest if the workers are heterogeneous, e.g., if they have different skills and can take care of different processes. For instance, worker 1 can perform every setup operation and worker 2 can only perform simple setup operations. Then, substitutability leads to more flexibility to construct feasible production plans. Again, the formulation is capable of handling the most general case concerning the resource-to-process-relations (*r:pr*). In this section, we only consider the case s:p:c where substitutable SRs may be switched from period to period. Later on in Section 3.5.2, we will present an extension for the s-p-c case.

Resources are defined using the index $q = 1, 2, ..., Q$. The index $u = 1, 2, ..., U$ distinguishes different skills, e.g., the skill of a worker to execute a simple or a more complex setup or the skill of a tool to execute a certain production process. The substitute set Θ_u contains all resources q with skill u which are substitutes for each other (see also Figure 3.3 for a better understanding). For example, if worker



$\Omega^p_{j=1} = \{1; 2\}$, i.e., **one** resource out of $\Theta_{u=1}$ **and one** resource out of $\Theta_{u=2}$ are necessary to produce product j = 1
q = 1
q = 2
q = 3
q = 4
substitute set u = 1
substitute set u = 2
q = 1 or q = 2 is necessary
q = 3 or q = 4 is necessary
and
process, e.g., production of j = 1

Figure 3.3: Example for substitute consideration

1 is defined as $q = 1$ and worker 2 as $q = 2$, both are substitutes for each other because both show skill $u = 1$, then $\Theta_1 = \{1; 2\}$. A process may require several skills simultaneously. For example, a low-skilled worker ($u = 1$) and a 9mm drill ($u = 2$) may be necessary simultaneously to produce a certain product $j = 1$. The process sets Ω^c_j, Ω^p_j and Ω^s_{ij} describe which skills u are necessary for conservation of setup state j, for production of product j and for a changeover from product i to

product j, respectively.[17] Then the process set $\Omega^p_1 = \{1;2\}$ represents the simultaneous necessity of both skills in the above example.

A resource can have more than one skill and thus belong to more than one substitute set. For example, worker $q = 1$ may be able to execute a simple setup ($u = 1$) and a complex setup ($u = 3$), whereas worker $q = 2$ is only able to execute the simple setup ($u = 1$). In addition, a 9mm diamond drill $q = 3$ may be able to drill 9mm holes in soft ($u = 2$) and hard ($u = 4$) surfaces, whereas a 9mm metal drill $q = 4$ may only be able to perforate soft surfaces ($u = 2$), altogether resulting in the four substitute sets $\Theta_1 = \{1;2\}$, $\Theta_2 = \{3;4\}$, $\Theta_3 = \{1\}$ and $\Theta_4 = \{3\}$. However, for ease of simplicity, we assume that the same resource cannot be in two different substitute sets of the same process set. When looking at the graphical representation in Figure 3.3, this would mean that a resource cannot be found more than once in the large circle (e.g., $q = 1$ cannot be in $\Theta_{u=1}$ and $\Theta_{u=2}$ because they are both part of $\Omega^p_{j=1}$). Relaxing this restriction would lead to significantly more variables in the model. As the above example demonstrates, this assumption is not really crucial because reasonable skill requirements can nevertheless be modeled, e.g., production processes needing low-skilled workers and simple drills ($\Omega^p_1 = \{1;2\}$) as well as processes needing high-skilled workers and complex drills ($\Omega^p_2 = \{3;4\}$).

Table 3.5: Symbols for disjunctive resources with substitutes

Indices and sets:	
$q = 1,...,Q$	disjunctive resources with substitutes
$u = 1,...,U$	skills
Θ_u	substitute set, listing alternative resources q with skill u
Ω^c_j	process set, listing all skills u necessary for conserving the setup state j
Ω^p_j	process set, listing all skills u necessary for producing product j
Ω^s_{ij}	process set, listing all skills u necessary for a changeover from product i to product j
Variables:	
$y^c_{lqs} \in \{0;1\}$	equals 1 if resource q is used to conserve a setup state on line l in microperiod s (0 otherwise)
$y^p_{lqs} \in \{0;1\}$	equals 1 if resource q is used to produce a product on line l in microperiod s (0 otherwise)
$y^s_{lqs} \in \{0;1\}$	equals 1 if resource q is used to perform a setup on line l in microperiod s (0 otherwise)

It is necessary to track which resources are used in order to ensure the correct synchronization of substitutes. Thus, three new variables are introduced: y^c_{lqs} indicates whether resource q is used to conserve a setup state on line l in microperiod s, y^p_{lqs} indicates if resource q is used during a production

[17]We omit the index l of the production lines for ease of readability.

process on line l in microperiod s and y^s_{lqs} triggers the use of resource q to perform a setup on line l in microperiod s. Additional symbols are summarized in Table 3.5 and the additional constraints are stated afterwards.

Additional constraints for disjunctive resources with substitutes read:

$$\sum_{q \in \Theta_u} y^p_{lqs} \geq y_{ljs} \qquad \forall l, j, s, u \in \Omega^p_j \tag{3.19}$$

$$\sum_{q \in \Theta_u} y^c_{lqs} \geq v_{ljs} \qquad \forall l, j, s, u \in \Omega^c_j \tag{3.20}$$

$$\sum_{q \in \Theta_u} y^s_{lqs} \geq (z_{lijs} + z^c_{lijs}) \qquad \forall l, i, j \neq i, s, u \in \Omega^s_{ij} \tag{3.21}$$

$$\sum_{l} (y^c_{lqs} + y^p_{lqs} + y^s_{lqs}) \leq 1 \qquad \forall s, q \tag{3.22}$$

For each skill u of process set Ω^p_j, inequalities (3.19) attach a suitable SR if product j needs to be produced on line l in microperiod s. As an example, let substitute set 1 again consists of workers $q = 1$ and $q = 2$ with skill $u = 1$ ($\Theta_1 = \{1;2\}$) and substitute set 2 consists of tools $q = 3$ and $q = 4$ with skill $u = 2$ ($\Theta_2 = \{3;4\}$), respectively. Product 1 is produced ($y_{l1s} = 1$), requiring both skills ($\Omega^p_1 = \{1;2\}$). On the one hand, (3.19) turns into $\sum_{q \in \Theta_1} y^p_{lqs} \geq 1$, forcing y^p_{l1s} or y^p_{l2s} to be set to 1. On the other hand, (3.19) yields $\sum_{q \in \Theta_2} y^p_{lqs} \geq 1$. Consequently y^p_{l3s} or y^p_{l4s} must take 1, assuring that at least one resource out of each substitute set is assigned to the production process of product 1.

The variables could also be continuous. This would lead to production plans with processes performed by combinations of disjunctive resources, e.g., 30% of the work is done by worker 1 and 70% of the work is done by worker 2. This is possible (when assuming zero transfer times), but normally not desired. Inequalities (3.20) and (3.21) are constructed in the same way as (3.19). They assure correct standbys and setups, respectively. Up to now, the decision variables only indicate the usage of the resources. Inequalities (3.22) enforce that the same resource q cannot be attached to several lines in the same microperiod.

Note the difference between the case without (Sect. 3.4.1) and with (Sect. 3.4.2) substitutes: in (3.18), resource usage only depends on whether the process is active or not. Inequalities (3.19)–(3.21) additionally provide the possibility to choose which resources are used if a process is active.

3.4.3 Cumulative resources without substitutes

In this subsection, the model is extended to consider cumulative resources without substitutes, e.g., a raw material which is used for parallel production of two different products on two lines. Raw materials can also be necessary during setups for test runs and adjustment processes of the production lines. It is predefined how many units of a resource are necessary to perform a certain process. Storing of resources is not possible, i.e., if the resource's capacity is not completely needed in a certain macroperiod, the remaining capacity cannot be used in the following macroperiod. An extension for storing resources will be presented in Section 3.5.5. The resource-to-process-relation is general (*r:pr*). The resources' availability is already known.

The index $r = 1, 2, ..., R$ denotes the different resources. Each resource has a given capacity K_{rt} per macroperiod t. The consumption of resource r is defined by parameters e^c_{jr}, e^p_{jr} and e^s_{ijr} for conservation, production and setup processes. Table 3.6 shows concrete definitions of these additional parameters.

Table 3.6: Parameters for cumulative resources without substitutes

Indices and sets:	
$r = 1, ..., R$	cumulative resources without substitutes
Data:	
e^c_{jr}	consumption of resource r while the setup state of product j is conserved for one time unit
e^p_{jr}	consumption of resource r while one unit of product j is produced
e^s_{ijr}	consumption of resource r while a setup from product i to product j is performed
K_{rt}	capacity of resource r in macroperiod t

With the above assumptions, the additional constraints (3.23) suffice to model cumulative resources without substitutes:

$$\sum_{l,i,j\neq i,s\in S_t} e^s_{ijr} z_{lijs} + \sum_{l,j,s\in S_t} e^p_{jr} x_{ljs} + \sum_{l,j,s\in S_t} e^c_{jr} \bar{x}_{ljs} \leq K_{rt} \qquad \forall r,t \tag{3.23}$$

They assure that the aggregate capacity of each resource r is respected in every macroperiod t. Since we only consider an SR's aggregate capacity per macroperiod in this section, it is sufficient to assume that the total amount e^s_{ijr} of a cumulative setup resource r (e.g., a raw material used for cleaning) will be merely consumed in the last microperiod of a continuous setup. A more detailed modeling of permanently used SRs will be discussed in Section 3.5.4.

3.4.4 Cumulative resources with substitutes

Finally, cumulative SRs with substitutes are considered. For example, it is possible to produce product 1 using raw material 1 or raw material 2. The index $n = 1, 2, ..., N$ defines the different cumulative resources. The index $o = 1, ..., O$ distinguishes different properties these resources may show, e.g., whether they stem from a local or a global supplier or from organic or conventional cultivation. Similarly to Θ_u in Subsection 3.4.2, substitute sets Ξ_o are introduced, which list the resources n that share property o and can alternatively be used. For example, the raw materials of a conventional final product could stem from both conventional and organic cultivation, whereas the raw materials of an organic final product would not allow such a substitution. We assume that the resources of a substitute set can be combined to fulfill a process, e.g., if 30 units of product 1 are produced in microperiod s, 10 of them can be produced using raw material 1 and 20 using raw material 2.

Table 3.7: Symbols for cumulative resources with substitutes

Indices and sets:	
$n = 1,...,N$	cumulative resources with substitutes
$o = 1,...,O$	properties
Ξ_o	substitute set, listing all cumulative resources n with property o
Λ^c_j	process set, listing all properties o necessary for conserving the setup state j
Λ^p_j	process set, listing all properties o necessary for producing product j
Λ^s_{ij}	process set, specifying all substitute sets o necessary for changing from product i to product j
Data:	
f^c_{jo}	consumption of resources with property o while the setup state of product j is conserved for one time unit
f^p_{jo}	consumption of resources with property o while one unit of product j is produced
f^s_{ijo}	consumption of resources with property o while a setup from product i to j is performed
K^c_{nt}	capacity of resource n in macroperiod t
Variables:	
$x^c_{lns} \geq 0$	consumption of resource n to conserve a setup state on line l in microperiod s
$x^p_{lns} \geq 0$	consumption of resource n to produce a product on line l in microperiod s
$x^s_{lns} \geq 0$	consumption of resource n to perform a setup on line l in microperiod s

The properties o that are necessary for conservation of the setup state of product j, for production of product j, and for setups from i to j are declared by the process sets Λ^c_j, Λ^p_j and Λ^s_{ij}, respectively. K^c_{nt} denotes the overall capacity of resource n in macroperiod t. f^c_{jo} denotes the overall amount of SRs with property o that is necessary to conserve setup state j for one time unit. f^p_{jo} and f^s_{ijo} state the amount of SRs with property o necessary for the production of one unit of product j and during a setup from product i to j, respectively. Similarly to the case of cumulative resources without substitutes, f^s_{ijo} is only used in the last microperiod of a continuous setup. The continuous variables x^c_{lns}, x^p_{lns} and x^s_{lns} distinguish the consumption of resource n on line l in microperiod s for conservation, production and

setups. This notation is summarized in Table 3.7.
Additional constraints for cumulative resources with substitutes:

$$\sum_{n\in\Xi_o} x^p_{lns} \geq f^p_{jo} x_{ljs} \qquad \forall l,j,s,o \in \Lambda^p_j \tag{3.24}$$

$$\sum_{n\in\Xi_o} x^c_{lns} \geq f^c_{jo} \bar{x}_{ljs} \qquad \forall l,j,s,o \in \Lambda^c_j \tag{3.25}$$

$$\sum_{n\in\Xi_o} x^s_{lns} \geq f^s_{ijo} z_{lijs} \qquad \forall l,i,j \neq i,s,o \in \Lambda^s_{ij} \tag{3.26}$$

$$\sum_{l,s\in S_t} x^p_{lns} + \sum_{l,s\in S_t} x^c_{lns} + \sum_{l,s\in S_t} x^s_{lns} \leq K^c_{nt} \qquad \forall n,t \tag{3.27}$$

Inequalities (3.24) determine the values of variables x^p_{lns}. It is assured that enough resource quantities from substitute set Ξ_o are reserved for the production quantity x_{ljs}. These quantities can be fulfilled by only one resource or a combination of different resources of the substitute set of property o. Note that here the same resource may be in several substitute sets of the same process set. For example, assume that multivitamin juice ($j = 1$) has to be mixed from orange and pineapple concentrate. Orange concentrate can be bought from a Spanish ($n = 1$) and Mexican ($n = 2$) supplier, pineapple concentrate can be bought from a Thai ($n = 3$) and Brazilian ($n = 4$) supplier. The juice needs to have shares of at least 20 % of both orange ($o = 1$) and pineapple ($o = 2$) concentrate ($f^p_{11} = f^p_{12} = 0.2$). However, the overall share of fruit concentrate ($o = 3$) has to be at least 50 % ($f^p_{13} = 0.5$). Then, the three substitute sets $\Xi_1 = \{1;2\}$, $\Xi_2 = \{3;4\}$ and $\Xi_3 = \{1;2;3;4\}$ result, which all belong to the same process set $\Lambda^p_1 = \{1;2;3\}$.[18]

Inequalities (3.25) and (3.26) are constructed in the same way and assure that enough resources are assigned for the conservation of setup states and for the setups, respectively. Equations (3.27) ensure that the available resource capacities K^c_{nt} are respected for each resource n in each macroperiod t.

3.5 Considering additional features

This section demonstrates how the model can be applied and how it may easily be adapted to incorporate further SR-relevant features. For some companies, these features can be of great help to create realistic production plans. It is possible to combine the constraints of the different scenarios described in Sections 3.4.1–3.4.4. For instance, if a company has disjunctive resources with substitutes and cumulative resources without substitutes, as well, Equations (3.1)–(3.17), (3.19)–(3.22) and (3.23) can be combined to address this case.

3.5.1 Split of setups into dismounting, cleaning and mounting

In some industrial settings it may be important to split a changeover from product i to product j into a dismounting operation of product i, a cleaning operation and a mounting operation of product j.

[18] If a property o does only refer to some ingredient of the SRs, as for example the sugar content of orange and pineapple concentrate, a corresponding factor a_{on} can be introduced and (3.24) can be changed to $\sum_{n\in\Xi_o} a_{on} x^p_{lns} \geq f^p_{jo} x_{ljs}$.

Sections 3.4.1 and 3.4.2 assume that disjunctive resources (e.g., cutting tools), which are necessary for product i, and disjunctive resources, which are necessary for product j, are assigned to the production line for the complete setup from product i to j. This assumption can be too restrictive. For instance, if the overall setup time is 6 hours and dismounting, cleaning and mounting last 1, 2 and 3 hours, the disjunctive resources which are necessary for product i will be assigned to the production line for 6 hours. By splitting the operation into dismounting, cleaning and mounting, the resources only necessary for product i already get available 1 hour after starting the changeover.

The three new states "dismounting", "cleaning" and "mounting" replace the state "setup". We introduce the identifiers D, E and M to distinguish these states. For example, the former aggregate setup time $st_{lij} = 6$ will be replaced by $st^D_{lij} = 1$ for dismounting, $st^E_{lij} = 2$ for cleaning and $st^M_{lij} = 3$ for mounting. Sequence dependency of these times is still important, as it could be the case that some tools, which have been necessary for product i, are still needed for product j and can be left mounted, whereas others have to be dismounted. This fact enforces to track the sequence. Note that the restriction of having exactly one state per line and microperiod is still valid. Furthermore, we assume that these three processes are always in the order dismounting $\rightarrow$ cleaning $\rightarrow$ mounting and that there is no idle time in between. Nevertheless, each of these three states may be spread over several periods.

Besides setup times further parameters (z_{lij0}, z^c_{lij0}; cf. Table 3.3) and variables (σ_{ls}, τ_{ls}, x^f_{ls}, z_{lijs}, z^c_{lijs}) have to differentiate the three new states in order to adapt the basic model accordingly. We use the same logic to distinguish them, but introduce an abbreviation for our notation: an asterisk * marks that a constraint has to be executed for each of the three states with the corresponding state-specific parameters and variables. For example, $x^{f*}_{ls} \geq 0 \, \forall l,s,*$ would abbreviate the non-negativity constraints $x^{fD}_{ls} \geq 0, x^{fE}_{ls} \geq 0$ and $x^{fM}_{ls} \geq 0 \, \forall l,s$ of the fractional setup times x^{fD}_{ls}, x^{fE}_{ls} and x^{fM}_{ls} of the states "dismounting", "cleaning" and "mounting".

Compared to basic model's objective (3.1) the only change of the new objective function (3.28) is that the z_{lijs} are substituted by z^M_{lijs}:

$$\text{Min} \sum_{t,j} h_j I_{jt} + \sum_{l,i,j\neq i,s} s_{lij} z^M_{lijs} + \sum_{l,j,s} c_{lj} x_{ljs} + \sum_{l,j,s} b_l \bar{x}_{ljs} \tag{3.28}$$

The adapted constraints of the basic model (3.1)–(3.17) are presented and explained in the following (constraints (3.2), (3.3), (3.6) and (3.7) are still valid):

$$\sum_{i,j\neq i} (z^{cD}_{lijs} + z^D_{lijs} + z^{cE}_{lijs} + z^E_{lijs} + z^{cM}_{lijs} + z^M_{lijs}) + \sum_j y_{ljs} + \sum_j v_{ljs} = 1 \qquad \forall l,s \tag{3.29}$$

$$\sum_j a_{lj} x_{ljs} + \sum_j \bar{x}_{ljs} + x^{fD}_{ls} + x^{fE}_{ls} + x^{fM}_{ls} = w_{s+1} - w_s \qquad \forall l,s \tag{3.30}$$

$$x^{f*}_{ls} \leq \overline{w}_{S+1} \sum_{i,j\neq i} (z^{c*}_{lijs} + z^*_{lijs}) \qquad \forall l,s,* \tag{3.31}$$

$$\sum_{r=s}^{s+1} x_{ljr} \geq m_{lj} \sum_{i\neq j} z^M_{lij,s-1} \qquad \forall j,l,s \tag{3.32}$$

$$y_{lj,s-1} + \sum_{i\neq j} z^{cD}_{lji,s-1} + \sum_{i\neq j} z^M_{lij,s-1} + v_{lj,s-1} = y_{ljs} + \sum_{i\neq j} z^{cD}_{ljis} + \sum_{i\neq j} z^D_{ljis} + v_{ljs} \qquad \forall j,l,s \tag{3.33}$$

$$z^{c*}_{lij,s-1} \leq z^{c*}_{lijs} + z^{*}_{lijs} \qquad \forall l,i,j \neq i,s,* \qquad (3.34)$$

$$z^{D}_{lij,s-1} \leq z^{cE}_{lijs} + z^{E}_{lijs} \qquad \forall l,i,j \neq i,s \qquad (3.35)$$

$$z^{E}_{lij,s-1} \leq z^{cM}_{lijs} + z^{M}_{lijs} \qquad \forall l,i,j \neq i,s \qquad (3.36)$$

$$\tau^{*}_{ls} \geq \sum_{i,j\neq i} st^{*}_{lij} z^{*}_{lijs} \qquad \forall l,s,* \qquad (3.37)$$

$$\tau^{*}_{ls} \leq \tau_{l,s-1} + x^{f*}_{ls} \qquad \forall l,s,* \qquad (3.38)$$

$$\tau^{*}_{ls} \leq x^{f*}_{ls} + \overline{w}_{S+1} \sum_{i,j\neq i} z^{c*}_{lij,s-1} \qquad \forall l,s,* \qquad (3.39)$$

$$\sigma^{*}_{ls} \geq \sigma^{*}_{l,s-1} + x^{f*}_{ls} - \sum_{i,j\neq i} st^{*}_{lij} z^{*}_{lij,s-1} \qquad \forall l,s,* \qquad (3.40)$$

$$\sigma^{*}_{ls} \leq \sum_{i,j\neq i} st^{*}_{lij} z^{*}_{lijs} + \sum_{i,j\neq i} \overline{w}_{S+1} z^{c*}_{lijs} \qquad \forall l,s,* \qquad (3.41)$$

$$z^{c*}_{lijS} = 0 \qquad \forall l,i,j \neq i,* \qquad (3.42)$$

Equations (3.4) are substituted by (3.29) to assure that at most one state is allowed per microperiod and production line. The capacity restrictions (3.5) are adapted to (3.30) in order to respect all five states that are now possible. Inequalities (3.8) are changed to (3.31) to respect all three states involved into a setup. As in (3.9), the minimum lotsizes of (3.32) still have to be produced within two subsequent microperiods. However, now z^{M}_{lijs} indicates the end of a setup.

The correct flow of states is assured by (3.33)–(3.36) which substitute (3.10) and (3.11). Equations (3.33) switch from a preceding state in period $s-1$ to one of the states "production", "dismounting" or "conservation" in period s. If a setup had been started in $s-1$ by switching to the dismounting state, constraints (3.34) enforce a correct flow of states during a continuous dismounting. Likewise, continuity of cleaning and mounting are ensured if the $*$ in (3.34) is substituted by E and M, respectively. Constraints (3.35) enable a change from dismounting to cleaning. Constraints (3.36) enforce the subsequent transition from cleaning to mounting. If a mounting had been completed in period $s-1$ because of $z^{M}_{lij,s-1} = 1$, Equations (3.33) again control the flow of states until the next setup starts with a dismounting operation. For example, let us assume that a changeover takes place from product i' to product j' where a continued dismount is finished in period s', i.e., $\sum_{i\neq j} z^{cD}_{lji,s-1} = 1$ and $\sum_{i\neq j} z^{D}_{ljis} = 1$ for $j = i'$ and period $s = s'$. Then the left-hand side of (3.33) has to become 0 for all j in period $s = s'+1$ because of (3.29). Furthermore, $z^{cE}_{li'j',s'+1}$ or $z^{E}_{li'j',s'+1}$ have to take the value 1 because of (3.35). This is not hindered by (3.33) because none of the variables indicating a cleaning operation appear in (3.33). If this cleaning ends in period $s'' \geq s'+1$ by $z^{E}_{li'j's''} = 1$, a mounting process has to start in period $s''+1$ (either $z^{cM}_{li'j',s''+1} = 1$ or $z^{M}_{li'j',s''+1} = 1$) because of (3.36). If this mounting is finished in some period $s''' \geq s''+1$, in the following period, the left-hand side of (3.33) becomes 1 for product $j = j'$ and production, conservation (or dismounting) of product j' might start (see Table 3.8). All in all, $\sum_{i\neq j} z^{cD}_{lji,s-1}$ on the left-hand side of (3.33) serves the same purpose as $\sum_{i\neq j} z^{c}_{lji,s-1}$ did in (3.10).

Constraints (3.37)–(3.42) replace (3.12)–(3.17), however, mirrored for the states "dismounting", "cleaning" and "mounting".

Table 3.8: Example for the transition from dismounting of product i' to the mounting of product j'

period $s =$	**active state**	**all other states**	**LHS and RHS of** (3.33)
$s'-1$	$\sum_{i \neq i'} z^{cD}_{li'is} = 1$	0	$= 1 \quad \text{for } j = i'$
s'	$\sum_{i \neq i'} z^{D}_{li'is} = 1$	0	$= 1 \quad \text{for } j = i'$
$s'+1$	$z^{cE}_{li'j's}$ or $z^{E}_{li'j's} = 1$	0	$= 0 \quad \forall j$
$s'' \geq s'+1$	$z^{E}_{li'j's} = 1$	0	$= 0 \quad \forall j$
$s''+1$	$z^{cM}_{li'j's}$ or $z^{M}_{li'j's} = 1$	0	$= 0 \quad \forall j$
$s''' \geq s''+1$	$\sum_{i \neq j'} z^{M}_{lij's} = 1$	0	$= 0 \quad \forall j$
$s'''+1$	$y_{lj's}$ or $v_{lj's} = 1$	0	$= 1 \quad \text{for } j = j'$

The constraints for the different resource types can easily be adapted to consider dismounting, cleaning and mounting, as well. As an example, this is done for disjunctive resources with substitutes (c.f. Section 3.4.2). The process sets Ω^s_{ij}, representing the information which skills u are necessary for a changeover from product i to product j, are replaced by new process sets Ω^D_i, Ω^E_j and Ω^M_j representing the information which skills u are necessary during dismounting of product i and cleaning and mounting of product j, respectively. The variables y^s_{lqs} are replaced by corresponding variables y^*_{lqs}.

Then, constraints (3.21) and (3.22) of Sect. 3.4.2 have to be replaced by the following adapted constraints (3.43)–(3.45) and (3.46):

$$\sum_{q \in \Theta_u} y^D_{lqs} \geq \sum_{j \neq i} (z^D_{lijs} + z^{cD}_{lijs}) \qquad \forall l, i, s, u \in \Omega^D_i \tag{3.43}$$

$$\sum_{q \in \Theta_u} y^E_{lqs} \geq \sum_{i \neq j} (z^E_{lijs} + z^{cE}_{lijs}) \qquad \forall l, j, s, u \in \Omega^E_j \tag{3.44}$$

$$\sum_{q \in \Theta_u} y^M_{lqs} \geq \sum_{i \neq j} (z^M_{lijs} + z^{cM}_{lijs}) \qquad \forall l, j, s, u \in \Omega^M_j \tag{3.45}$$

$$\sum_{l} (y^c_{lqs} + y^p_{lqs} + y^D_{lqs} + y^E_{lqs} + y^M_{lqs}) \leq 1 \qquad \forall s, q \tag{3.46}$$

Constraints (3.43)–(3.45) determine which disjunctive SRs of the substitute sets Θ_u are actually used in the different states. Because of (3.46), each SR q is at most applied once per microperiod s.

3.5.2 S-p-c model for disjunctive substitutes and splitting of setups

In this section, we introduce additional constraints to represent the s-p-c case. If splitting of setups is allowed, too, this means that it is mandatory that the same resource (e.g., a tool) is used during mounting, production or conservation and dismounting.

Constraints (3.47)–(3.50) assure that the same resource is used during all subsequent microperiods of this sequence of processes:

$$y^D_{lq,s-1} \leq y^D_{lqs} + \sum_{i,j\neq i} z^D_{lij,s-1} \qquad \forall l,s,q \tag{3.47}$$

$$y^M_{lq,s-1} \leq y^M_{lqs} + \sum_{i,j\neq i} z^M_{lij,s-1} \qquad \forall l,s,q \tag{3.48}$$

$$y^p_{lq,s-1} + y^c_{lq,s-1} \leq y^p_{lqs} + y^c_{lqs} + \sum_{i,j\neq i} (z^{cD}_{lijs} + z^D_{lijs}) \qquad \forall l,s,q \tag{3.49}$$

$$y^M_{lq,s-1} + y^p_{lq,s-1} + y^c_{lq,s-1} \leq y^M_{lqs} + y^p_{lqs} + y^c_{lqs} + y^D_{lqs} \qquad \forall l,s,q \tag{3.50}$$

Constraints (3.47) assure that an SR q, which is used for dismounting on line l in microperiod $s-1$, is also used for dismounting on the same line in microperiod s. This flow can only be interrupted if dismounting also had been finished in microperiod $s-1$, i.e., if $\sum_{i,j\neq i} z^D_{lij,s-1} = 1$. Constraints (3.48) work in the same way to consider resource flows during mounting. The resource flow during production and conservation of a setup state is respected using constraints (3.49). They allow an arbitrary change between the states production and conservation until dismounting begins. Constraints (3.50) enforce that mounting can only be followed by mounting or the sequence defined by (3.49). Overall, the resource can only be released from this production line using a dismounting operation.

If a cleaning process required the same SR q during all cleaning periods, this could be modeled in a similar fashion.

3.5.3 Capacity restriction of disjunctive resources

Section 3.4.1 assumed that a disjunctive SR p is — like the primary resource — always available for the complete planning horizon. In the literature of disjunctive resources without substitutes, this assumption is sometimes relaxed (c.f. Table 3.2). For example, a setup operator p might only be available for 7 hours within an 8 hour macroperiod. With K^d_{pt} denoting the available capacity of SR p in macroperiod t, constraints (3.51) and (3.52) extend the model of Section 3.4.1:

$$x^h_{lijs} \geq x^f_{ls} - \overline{w}_{S+1} * (1 - z_{lijs} - z^c_{lijs}) \qquad \forall l,i,j,s \tag{3.51}$$

$$\sum_{l,i,j\neq i,s\in S_t} b^s_{ijp} x^h_{lijs} + \sum_{l,j,s\in S_t} b^p_{jp} a_{lj} x_{ljs} + \sum_{l,j,s\in S_t} b^c_{jp} \bar{x}_{ljs} \leq K^d_{pt} \qquad \forall p,t \tag{3.52}$$

The fractional setup time x^f_{ls} of the basic model does not identify the products i and j responsible for a changeover. Thus constraints (3.51) do establish this missing link by setting the newly introduced variables $x^h_{lijs} (\geq 0)$ to x^f_{ls} if i and j cause a setup and to 0, otherwise. With this knowledge, constraints (3.52) can aggregate the usage of SR p (c.f. Table 3.4) within macroperiod t and limit it by its available capacity K^d_{pt}.

3.5.4 Capacity restriction of cumulative resources for a continuously provided resource

If non-substitutable cumulative resources are provided in a continuous manner, e.g., because there is a continuous flow from a pipeline, it might not be sufficient to respect the SR's capacity only per macroperiod in an aggregate manner, as it has been done using constraints (3.23) of Section 3.4.3. For example, if a macroperiod consisted of 10 microperiods, each lasting 1 minute, if an SR was provided with a rate of 1 unit per minute and if there were two production lines, each needing 5 units of the SR in the first 5 minutes, such a schedule would indeed satisfy (3.23), but nevertheless be unrealistic because in each of the first five microperiods two units of the resource were required with only one being provided. In this case, the SR's capacity needs to be modeled in more detail, e.g., on a microperiod basis. Then, constraints (3.23) have to be substituted by (3.53) where the capacity of a secondary resource r is defined by a maximum flow rate Kf_r (measured in units of the SR per unit of time):

$$\sum_{l,i,j\neq i} e^s_{ijr} x^h_{lijs} + \sum_{l,j} e^p_{jr} x_{ljs} + \sum_{l,j} e^c_{jr} \bar{x}_{ljs} \leq Kf_r(w_{s+1} - w_s) \qquad \forall r, s \tag{3.53}$$

In this case of a continuous supply it is also no longer reasonable that an SR is only consumed in the last microperiod of a continuous setup. Thus, the auxiliary variables x^h_{lijs} of Section 3.5.3 need to be used again and e^s_{ijr} needs to be re-defined as the consumption of resource r during one time unit of a setup from product i to product j (c.f. Table 3.6). Then, the left-hand side of (3.53) represents the SR's total consumption during microperiod s, whereas the right-hand side represents its maximum availability in the same microperiod. Note that nevertheless all microperiods are still of flexible length.

3.5.5 Inventory balancing of cumulative resources

If unused cumulative resources can be stored, the development of the resulting inventories should be tracked over time. This can also be done on a macro- or microperiod basis by introducing variables $\bar{I}_{rt} \geq 0$ or $\hat{I}_{rs} \geq 0$ representing the inventory of resource r at the end of macroperiod t or microperiod s, respectively. Let $\bar{K}_{rt}$ now denote a predefined, given supply of resource r in macroperiod t (e.g., by a mid-term contract with a supplier of r) and $\hat{Kf}_r$ denote a constant inflow rate of resource r per unit of time (e.g., from a preceding, independent stage of production).

Then the model of Section 3.4.3 can be adapted by replacing constraints (3.23) with

$$\bar{I}_{rt} = \bar{I}_{r,t-1} + \bar{K}_{rt} - \sum_{l,i,j\neq i,s\in S_t} e^s_{ijr} z_{lijs} - \sum_{l,j,s\in S_t} e^p_{jr} x_{ljs} - \sum_{l,j,s\in S_t} e^c_{jr} \bar{x}_{ljs} \qquad \forall r, t \tag{3.54}$$

and the model of Section 3.5.4 can be adapted by replacing constraints (3.53) with

$$\hat{I}_{rs} = \hat{I}_{r,s-1} + \hat{Kf}_r(w_{s+1} - w_s) - \sum_{l,i,j\neq i} e^s_{ijr} x^h_{lijs} - \sum_{l,j} e^p_{jr} x_{ljs} - \sum_{l,j} e^c_{jr} \bar{x}_{ljs} \qquad \forall r, s. \tag{3.55}$$

Constraints (3.54) and (3.55) are standard inventory balancing constraints analogous to constraint (3.3). Note that holding costs could easily be introduced for $\bar{I}_{rt}$ since the length of a macroperiod is known in advance. However, this is not the case for $\hat{I}_{rs}$ because the length of a microperiod is variable.

3.5.6 No substitution of cumulative resources during a production lot

The formulation in Section 3.4.4 allows a combination of substitutable cumulative resources when producing a certain product. Sometimes, it is not desired to switch between alternative raw materials while producing a single lot. For instance, one would like to avoid switching the providing tank several times for a single production lot lasting just a few minutes. To hinder substitution within a single lot, the model from Section 3.4.4 needs to be adapted. We just sketch the general idea but do not present the complete model:

$$x^{p}_{lns} \leq M y^{bp}_{lns} \qquad \forall l,n,s \tag{3.56}$$

$$\sum_{n \in \Xi_o} y^{bp}_{lns} \leq 1 \qquad \forall l,s,o \tag{3.57}$$

The additional binary variables y^{bp}_{lns} are set to 1 if resource n is used for production on line l in microperiod s (otherwise 0). Constraints (3.56) assure this by means of a big constant M. To forbid switching, at most one type of SR of each substitute set Ξ_o is allowed per line, microperiod and property o (3.57). If desired, analogous constraints must be formulated for the other potential states of the line, too. Since a production lot may span over several microperiods, it is still possible that line l uses one raw material of substitute set Ξ_o in microperiod s and another one in microperiod $s+1$. To prevent this, flow constraints similar to (3.49) are necessary.

3.5.7 All lines must consume the same resource

The following extension represents the case of Camargo et al. (2012). The authors consider a scenario where all production lines must consume the same resource at the same time. This way they model a furnace which feeds several lines in parallel. The material can differ from period to period. Our model from Section 3.4.3 can be used as a basis. Thus, cumulative resources without substitutes are considered, and the following constraints added:

$$\sum_{k,i,j \neq i} e^{s}_{ijr} z_{kijs} + \sum_{k,j} e^{p}_{jr} x_{kjs} + \sum_{k,j} e^{c}_{jr} \bar{x}_{kjs} \leq M \left(\sum_{i,j \neq i} e^{s}_{ijr} z_{lijs} + \sum_{j} e^{p}_{jr} x_{ljs} + \sum_{j} e^{c}_{jr} \bar{x}_{ljs} \right) \qquad \forall l,r,s \tag{3.58}$$

The left-hand side of (3.58) constitutes the total consumption of resource r on all lines in microperiod s, whereas the brackets of the right-hand side constitute the consumption of the same resource in the same microperiod, but only on line l. Thus, with M again being a large positive constant, (3.58) ensure that all other lines are forced to use resource r if at least one line uses this resource. Note that a line may still require more than just a single SR.

3.6 Examples

The following examples demonstrate the functionality of the model. The focus is on the secondary resources. Thus, the basic production scenario is very simple and all models are kept small in order to remain comprehensive. In Section 3.6.1 disjunctive resources without substitutes are addressed. The subsequent section is devoted to disjunctive resources with substitutes. Section 3.6.3 addresses cumulative resources with substitutes. Afterwards an example with all types of resources is presented. The final scenario of Section 3.6.5 requires continuous setups.

The exemplary models have been implemented using GMPL 4.50 as a modeling language and GLPK 4.55 as a solver. All experiments have been performed on an IntelCore i5-4300 CPU 1.9 Ghz, 8 GB RAM. However, computation time is not the focus of this section, but rather the flexibility of the underlying mixed-integer programming models.

3.6.1 Disjunctive resources without substitutes

The basic production scenario comprises two lines and two products (plus a product $j = 0$ to represent the neutral state). It lasts 1 time unit to produce 1 unit of each product. Production of product 1 is only possible on line 1 and product 2 can be produced exclusively on line 2. Production costs are 2 monetary units per unit of each product. Setup costs are set to 1 monetary unit and setup times are set to 1 time unit for every product combination. Both lines are in the neutral state ($j = 0$) at the beginning of the planning horizon. Two macroperiods containing three microperiods each are considered. Each of them has a capacity of 10 time units. If 1 unit of a product is stored for one macroperiod, holding costs of 1 monetary unit occur. Standby costs are zero and the demand of product 1 and 2 is 8 units, each, at the end of the second macroperiod. Minimum lotsizes are set to one. The optimal production plan without consideration of secondary resources is presented in Figure 3.4.

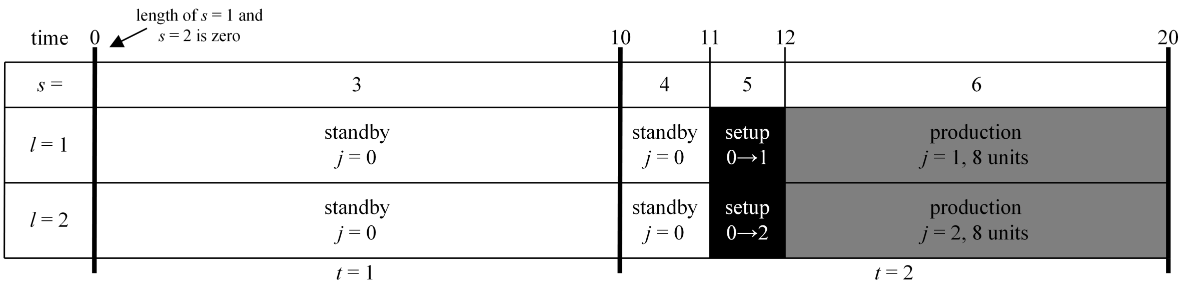


Figure 3.4: Production plan without consideration of secondary resources

The length of microperiods $s = 1$ and $s = 2$ is zero. In microperiods $s = 3$ and $s = 4$ the initial setup state ($j = 0$) is conserved (standby) on both lines. Microperiod $s = 5$ is used to perform the setups from product 0 to 1 on line 1 and from product 0 to 2 on line 2. The last microperiod has a length of 8 time units and is used for production on both lines. The total costs are 34 monetary units.

Now, there are three different secondary resources: workers A and B, and tool C. Worker A is necessary for every setup which involves product 1 and worker B is necessary for every setup which involves product 2. Tool C is necessary for every setup which involves product 1 and 2 and for production and standby of the products 1 and 2. The production plan resulting from solving model (3.1)-(3.18) is shown in Figure 3.5.

Since tool C is necessary for the production of both products, it is not possible to produce them at the

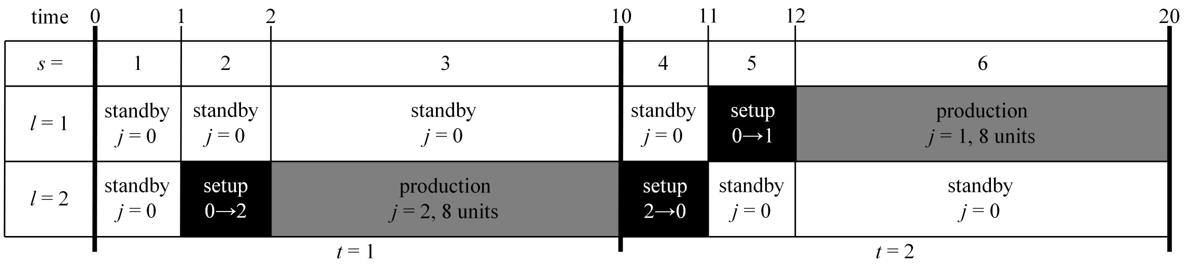


Figure 3.5: Production plan with consideration of disjunctive resources

same time. Furthermore, tool C is also necessary for the standby of the products. Thus, it is not possible that line 2 stays set up for product 2 after production. Additional costs occur due to the changeover from product 2 to 0 in microperiod 4. The tool is still necessary during this setup operation, thus, this setup has to be finished before the setup on line 1 can start in microperiod 5. Due to preproduction in microperiod 3 holding costs of 8 monetary units occur. Nevertheless, an optimal production plan, which would be feasible in this practical application, has been created.

3.6.2 Disjunctive resources with substitutes

The considered scenario is similar to the scenario described in Section 3.6.1. The only difference is the existence of an additional tool D, which can be installed instead of tool C for both products. For workers A and B, constraints (3.18) of the model without substitutes can still be used. Nevertheless, it is also possible to apply the constraints for resources with substitutes. In this case, there would be just one resource in the substitute set and the model would involve unnecessary variables. However, for tools C and D the formulation of Section 3.4.2 is mandatory. The resulting production plan is presented in Figure 3.6.

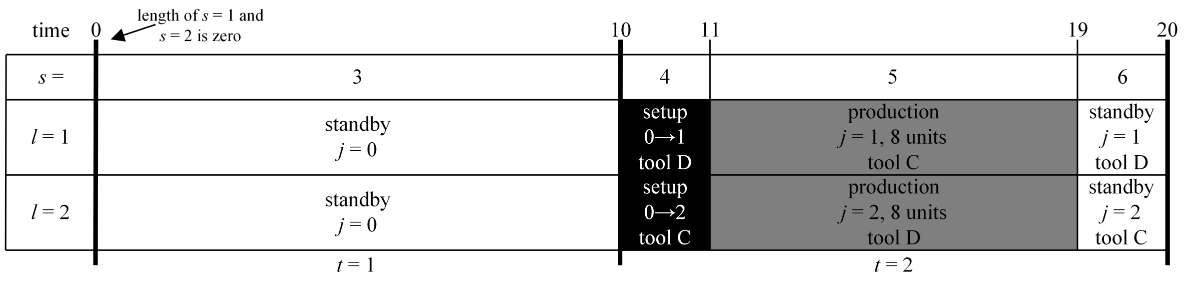

time
0
length of $s = 1$ and $s = 2$ is zero
10
11
19
20
$s =$
3
4
5
6
$l = 1$
standby $j = 0$
setup $0 \rightarrow 1$ tool D
production $j = 1$, 8 units tool C
standby $j = 1$ tool D
$l = 2$
standby $j = 0$
setup $0 \rightarrow 2$ tool C
production $j = 2$, 8 units tool D
standby $j = 2$ tool C
$t = 1$
$t = 2$

Figure 3.6: Production plan with consideration of disjunctive resources with substitutes

As can be seen, complete production takes place in macroperiod $t = 2$. Thus, the total costs are 34 monetary units. The usage of tools C and D is indicated by the variables y^s_{lqs}, y^p_{lqs} and y^c_{lqs}. As shown in Figure 3.6, each resource is used at most on one line in each microperiod.

Since a frequent switching of SRs is not always welcome, in the following we apply the s-p-c formulation of Section 3.5.2. We assume that dismounting, cleaning and mounting lasts 1 time unit, each. Substitute set 1, consisting of tools C and D, is necessary during the mounting and production of the products $j = 1$ and $j = 2$. The resulting production plan is presented in Figure 3.7.

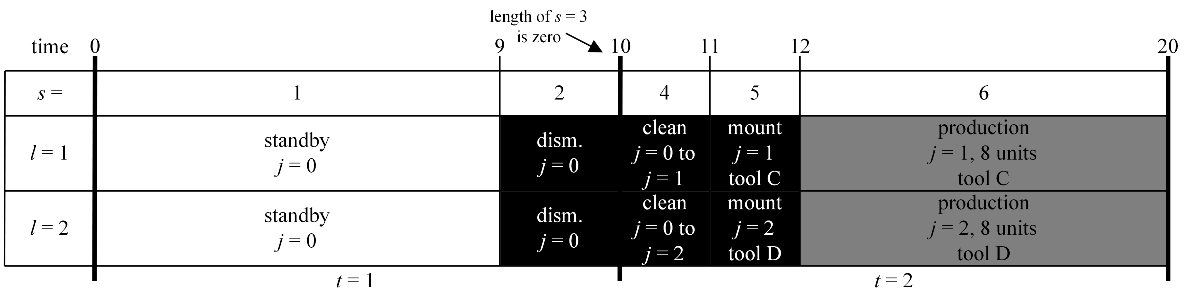


Figure 3.7: Production plan of the s-p-c case

Setups are split into dismounting in microperiod 2, cleaning in microperiod 4 and mounting in microperiod 5. Production takes place in microperiod 6. The total costs are 34 monetary units. Tool C is used on line 1 during mounting and production. Tool D is used on line 2 for the same sequence of states. A comparison with Figure 3.6 reveals that s-p-c model works as expected. The schedule of Figure 3.7 is also a feasible and optimal plan for the previous model. However, as already mentioned, the schedule of Figure 3.6 would not be feasible for the s-p-c case.

3.6.3 Cumulative resources with substitutes

Now, the basic production scenario is extended by cumulative SRs only. We assume that 2 units of resource 1 (e.g., a raw material) are necessary to produce product 1. Furthermore, we assume that 3 units of resource 1, 2 or 3 are necessary to produce product 2. The model with substitutes of Section 3.4.4 is used. Substitute set 1 consists of resource 1 and substitute set 2 consists of resources 1, 2 and 3. The availability of resource 1 is 10 units per macroperiod. Each of the two other resources has an availability of 5 units per macroperiod. The resulting production plan is presented in Figure 3.8.

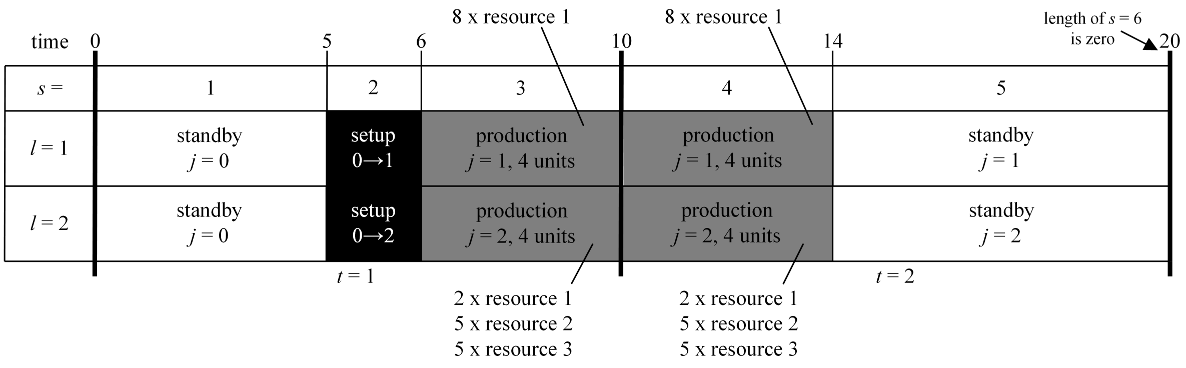


Figure 3.8: Production plan with consideration of cumulative resources

The setups from product 0 to 1 and from product 0 to 2 take place in microperiod 2. Product 1 is set up on line 1 and product 2 on line 2. Production of 4 units of each product takes place in microperiod 3. Note that resource 1 is used on both lines in parallel in microperiod 3 as expected under the setting of cumulative resources. As can be seen, the resources of substitute set 2 are combined to fulfill the requested quantity of resources to produce 4 units of product 2. The missing units of products 1 and 2 are produced in macroperiod 2. Since the pre-production of 4 units each is necessary due to resource restrictions, the costs sum up to 42 monetary units including 8 monetary units for storing 4 units of product 1 and 4 units of product 2.

3.6.4 A combination of different types of resources

The following scenario demonstrates the combination of disjunctive and cumulative SRs. Furthermore, it also considers the case that a process needs resources of more than just one substitute set. The basic scenario is not changed. A setup which involves product 1 needs the following resources: one worker of a group of two high skilled workers (substitute set Θ_1 including disjunctive resources 1 and 2), one worker of a group of three lower skilled workers (substitute set Θ_2 including disjunctive resources 3, 4 and 5) and one crane (disjunctive resource 6) whereof just one replica exists. There are two different substitute sets of cumulative resources (Ξ_1 and Ξ_2), each consisting of two raw materials. A setup consumes 10 units of each of these substitute sets to perform test runs and adjustments of the production line. Production of product 1 requires 2 units of each of these substitute sets for each produced unit. Furthermore, one worker of substitute set Θ_2 is necessary. Setup and production of product 2 require exactly the same resources. Disjunctive resources are available during the complete planning horizon. Raw materials 1 and 2 (substitute set Ξ_1) are limited to 20 units per period, each. The same holds for substitute set Ξ_2 (raw materials 3 and 4). Figure 3.9 shows the resulting production plan.

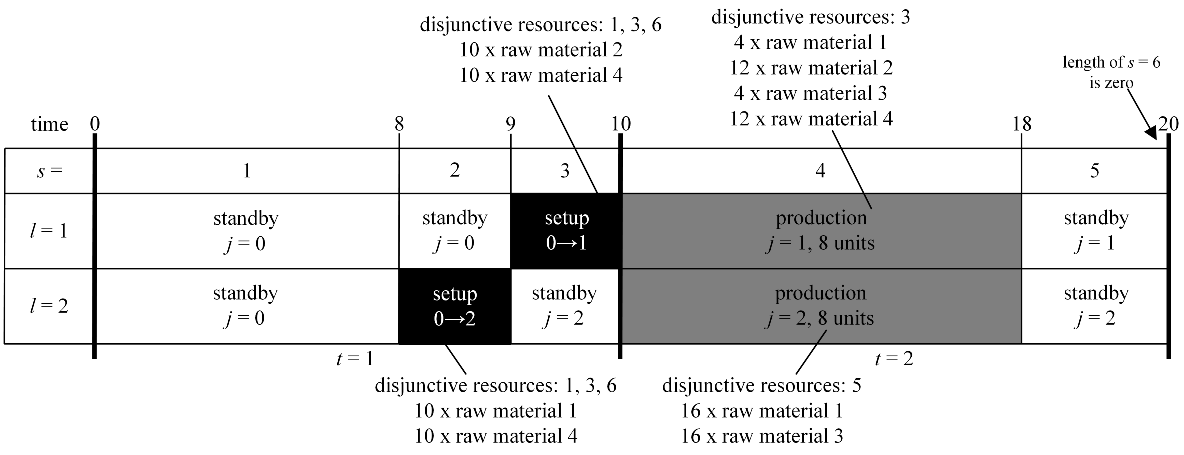


Figure 3.9: Production plan with different types of resources

The total costs are 34 monetary units. As can be seen, all requirements are considered. For instance, disjunctive resource 3 is used for production on line 1, and disjunctive resource 5 for production on line 2 in microperiod 4. Both are substitutes for each other and disjunctive. Thus, for example, resource 3 cannot be used in parallel on both lines and another resource is used for line 2. Setting up both lines in parallel is not possible because disjunctive resource 6 is necessary for both of them. Also note that the minimum lotsize on line 2 is ensured in the second microperiod after the setup operation (c.f. constraint

(3.9)).

3.6.5 A scenario requiring continuous setups

The last scenario necessitates continuous setup times. Besides the following changes all parameters remain the same as before. Demand occurs only in the second macroperiod: 10 units of product 1 and 5 units of product 2. The setup times are re-defined: each setup on line 1 lasts 3 time units and each setup on line 2 lasts 11 time units. Only one cumulative resource with a capacity of 10 units per macroperiod is considered. To produce one unit of product 1 or 2 one unit of the cumulative resource is necessary. The resulting production plan is shown in Figure 3.10.

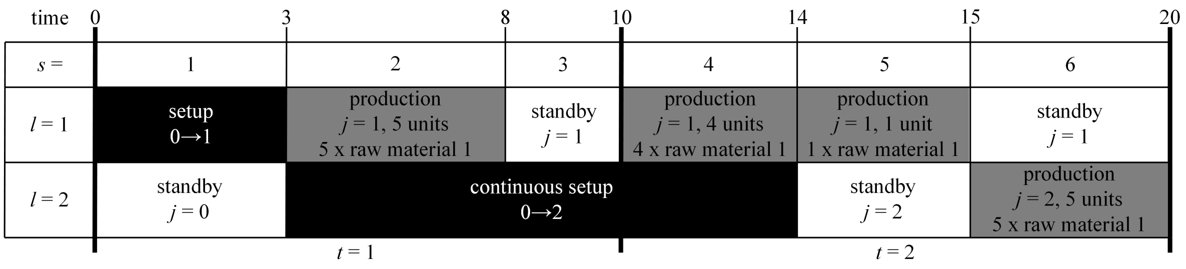


Figure 3.10: Production plan with cumulative resources and a continuous setup

Obviously, it is impossible to produce all products in macroperiod 2. The limited availability of the cumulative resource is the reason for this. Thus, 5 units of product 1 are produced in macroperiod 1 and stored until macroperiod 2. As intended, on line 2, there is a setup which continues over three microperiods and even exceeds a macroperiod boundary.

3.7 Summary and outlook

A mixed-integer, linear programming model for single-stage, simultaneous lotsizing and scheduling considering secondary resources (SRs) has been presented. In this field of research, besides the limited capacity of the primary resources (several parallel production lines of a single stage of production) also the limited availability of further (“secondary”), potentially scarce resources like setup tools, setup operators or raw materials has to be respected.

A comprehensive literature research has revealed that most of the existing SR-models are tailored to specific practical applications. It has also helped to develop a classification scheme for SRs, which comprises four different types of resources: disjunctive resources with and without substitutes and cumulative resources with and without substitutes. While disjunctive SRs can only serve a single production line at a single point in time, do not become part of the final product and can be used several times consecutively (like setup tools), cumulative SRs can serve several production lines simultaneously, do become part of the final product and can be consumed only once (e.g., raw materials). Substitutability distinguishes whether only a single type of SR or several alternative types of SRs could be applied for a certain setup or production process.

The developed model is based on the general lotsizing and scheduling problem for parallel production lines (GLSPPL) and can represent general situations which combine all four types of SRs. Synchronization of SRs is realized using a common time structure on all parallel production lines. Substitutes for different skills and properties of SRs are incorporated using substitute sets. Additional index sets define which skills and properties are necessary to perform a certain process.

The major advantage of the model is that it unifies nearly all SR-constraints and -applications found in the literature within a single formulation. This formulation still remains compact since features, which were unnecessary for a certain application, could easily be left out by omitting the corresponding constraints and variables. Some features, which have not been dealt with in science so far but are of practical relevance, can also be incorporated. Examples are cumulative SRs with substitutes or the ability to refine the modeling of changeovers by distinguishing between dismounting, cleaning and mounting. Such an approach allows to construct more flexible and thus more realistic schedules.

Some examples have been presented which demonstrate the applicability of the new model. However, extensive numerical tests on the computational performance of the new model have not been performed. This would have gone beyond the scope of a single publication. Thus, future research has to analyze and, where possible, improve the performance of the formulation, but mainly to design scalable solution heuristics for problem instances of industrial size. If these base on the new model, they promise to be more generally applicable than current SR-heuristics are. Nevertheless, the already existing models and heuristics for specific applications, which have been referred to in Section 3.2, may serve as benchmarks for comparison. Another challenge for future research is to extend the model for multiple stages of production.

Bibliography

Almada-Lobo, B., Klabjan, D., Carravilla, M. A., Oliveira, J. F., 2010. Multiple machine continuous setup lotsizing with sequence-dependent setups. Computational Optimization and Applications 47 (3), 529–552.

Almeder, C., Almada-Lobo, B., 2011. Synchronisation of scarce resources for a parallel machine lotsizing problem. International Journal of Production Research 49 (24), 7315–7335.

Camargo, V. C. B., Toledo, F. M. B., Almada-Lobo, B., 2012. Three time-based scale formulations for the two stage lot sizing and scheduling in process industries. Journal of the Operational Research Society 63, 1613–1630.

Camargo, V. C. B., Toledo, F. M. B., Almada-Lobo, B., 2014. HOPS – hamming-oriented partition search for production planning in the spinning industry. European Journal of Operational Research 234 (1), 266–277.

Copil, K., 2016. Capacitated Lot-Sizing and Scheduling with Scarce Setup Resources. Books on Demand, Norderstedt.

Copil, K., Wörbelauer, M., Meyr, H., Tempelmeier, H., 2017. Simultaneous lotsizing and scheduling problems: a classification and review of models. OR Spectrum 39 (1), 1–64.

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

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

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

Ferreira, D., Clark, A. R., Almada-Lobo, B., Morabito Neto, R., 2012. Single-stage formulations for synchronised two-stage lot sizing and scheduling in soft drink production. International Journal of Production Economics 136 (2), 255–265.

Ferreira, D., Morabito Neto, R., Rangel, S., 2009. Solution approaches for the soft drink integrated production lot sizing and scheduling problem. European Journal of Operational Research 196 (2), 697–706.

Ferreira, D., Morabito Neto, R., Rangel, S., 2010. Relax and fix heuristics to solve one-stage one-machine lot-scheduling models for small-scale soft drink plants. Computers & Operations Research 37 (4), 684–691.

Figueira, G., Santos, M. O., Almada-Lobo, B., 2013. A hybrid VNS approach for the short-term production planning and scheduling: a case study in the pulp and paper industry. Computers & Operations Research 40 (7), 1804–1818.

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

Fleischmann, B., Meyr, H., 1997. The general lotsizing and scheduling problem. OR Spectrum 19 (1), 11–21.

Furlan, M. M., Almada-Lobo, B., Santos, M. O., Morabito Neto, R., 2015. Unequal individual genetic algorithm with intelligent diversification for the lot-scheduling problem in integrated mills using multiple-paper machines. Computers & Operations Research 59, 33–50.

Göthe-Lundgren, M., Lundgren, J. T., Persson, J. A., 2002. An optimization model for refinery production scheduling. International Journal of Production Economics 78 (3), 255–270.

Haase, K., 1996. Capacitated lot-sizing with sequence dependent setup costs. OR Spectrum 18 (1), 51–59.

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

Karmarkar, U. S., Schrage, L., 1985. The deterministic dynamic product cycling problem. Operations Research 33 (2), 326–345.

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

Koçlar, A., Süral, H., 2005. A note on "The general lot sizing and scheduling problem". OR Spectrum 27 (1), 145–146.

Lasdon, L., Terjung, R. C., 1971. An efficient algorithm for multi-item scheduling. Operations Research 19 (4), 946–969.

Mac Cawley, A. F., 2014. The international wine supply chain: challenges from bottling to the glass. Ph.D. thesis, Georgia Institute of Technology.

Maldonado, M., Rangel, S., Ferreira, D., 2014. A study of different subsequence elimination strategies for the soft drink production planning. Journal of Applied Research and Technology 12 (4), 631–641.

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

Meyr, H., 2004. Simultane Losgrößen- und Reihenfolgeplanung bei mehrstufiger kontinuierlicher Fertigung. Zeitschrift für Betriebswirtschaft 74 (6), 585–610.

Meyr, H., Mann, M., 2013. A decomposition approach for the general lotsizing and scheduling problem for parallel production lines. European Journal of Operational Research 229 (3), 718–731.

Persson, J. A., Göthe-Lundgren, M., Lundgren, J. T., Gendron, B., 2004. A tabu search heuristic for scheduling the production processes at an oil refinery. International Journal of Production Research 42 (3), 445–471.

Santos, M. O., Almada-Lobo, B., 2012. Integrated pulp and paper mill planning and scheduling. Computers & Industrial Engineering 63 (1), 1–12.

Seeanner, F., 2013. Multi-stage simultaneous lot-sizing and scheduling – planning of flow lines with shifting bottlenecks. Produktion und Logistik. Springer Gabler, Wiesbaden.

Seeanner, F., Almada-Lobo, B., Meyr, H., 2013. Combining the principles of variable neighborhood decomposition search and the fix&optimize heuristic to solve multi-level lot-sizing and scheduling problems. Computers & Operations Research 40 (1), 303–317.

Seeanner, F., Meyr, H., 2013. Multi-stage simultaneous lot-sizing and scheduling for flow line production. OR Spectrum 35 (1), 33–73.

Suerie, C., 2005. Time continuity in discrete time models: new approaches for production planning in process industries. Lecture Notes in Economics and Mathematical Systems. Springer, Berlin et al.

Tempelmeier, H., Buschkühl, L., 2008. Dynamic multi-machine lotsizing and sequencing with simultaneous scheduling of a common setup resource. International Journal of Production Economics 113 (1), 401–412.

Tempelmeier, H., Copil, K., 2016. Capacitated lot sizing with parallel machines, sequence-dependent setups and a common setup operator. OR Spectrum 38 (4), 819–847.

Toledo, C. F. M., Arantes, M. d. S., de Oliveira, R. R. R., Almada-Lobo, B., 2013. Glass container production scheduling through hybrid multi-population based evolutionary algorithm. Applied Soft Computing 13 (3), 1352–1364.

4 Decomposing large-scaled simultaneous lotsizing and scheduling problems using product aggregation

Abstract During the last decades, many heuristics have been introduced to solve simultaneous lotsizing and scheduling problems. However, many problems of practical relevance are large-scaled in terms of the number of products and machines which must be considered simultaneously. Present heuristics often provide an unsatisfying performance to solve these problems, i.e., it is impossible to find cost-efficient, or at least feasible production plans in an adequate time. We present a new heuristic, which creates a modified multi-line master problem by aggregating products into groups. The resulting problem is less complex and its solution can be used to define single-line sub problems. These sub problems are solved by present heuristics and the results are then combined to form a solution to the original problem. The new heuristic provides a superior alternative to solve large-scaled problems.

Keywords Scheduling, Heuristics, Simultaneous lotsizing and scheduling, Production

4.1 Introduction

Consumer goods, such as dairy products or drugstore products, will always be in demand. From a producer's perspective, these products have certain characteristics such as involving only a small number of production stages within the production process. We use the dairy industry as example (see e.g., Smith-Daniels and Smith-Daniels 1986 and Seeanner and Meyr 2013): producing a yoghurt utilizes two stages. One stage for blending yoghurt and a second stage to fill the final product into cups. This type of production is called make-and-pack. A stage may involve several consecutive machines, showing similar production speeds. Since products run through these machines in the same order and transportation times between the machines can be neglected, they can be planned as one unit, which is called (production) line. Furthermore, in make-and-pack scenarios, often only one stage defines a bottleneck, i.e., a stage which has significantly less production capacity than the other stages. Therefore, it is sufficient to plan this bottleneck stage in detail and propagate the resulting plan to the non-bottleneck stages. A bottleneck stage normally consists of several parallel production lines which often have the same functionality, i.e., are capable of producing the same products. On the other hand, these lines normally are heterogeneous, i.e., they have different levels of efficiency, since they have been bought step by step over a long period of time. As a result, there might be line-dependent parameters like production costs and production speeds. All in all, this leads to dependencies between the lines and they must be planned simultaneously.

Sequence-dependent changeover costs and times may occur if there is a changeover from one product to another product on a line. Thus, not only the line assignment, but also the sequence of the products must be planned. Due to a high indifference of customers between similar products of two consumer goods producing companies, it is very important to fulfill the demand without backlogging to avoid lost sales. Normally, the demand is forecasted and can be assumed as deterministically known. Furthermore, the demand for each product may change in each period (days or weeks) of a finite planning horizon (months or quarters). If due to the necessity of short lead-times and the occurrence of large setup times, consumer goods are produced on stock, which is the normal case, holding costs must be considered until stocked products are used to fulfill the forecasted demand. I.e., a lotsizing problem must be solved too. Due to given limited capacities of the production lines, a high interdependency between the different planning problems occurs and requires a simultaneous planning. For example, due to sequence-dependent setup times, it is impossible to define lotsizes without considering the scheduling. Previously described problems are named simultaneous lotsizing and scheduling problems. (see, e.g., Meyr 1999, Chapter 3)

Many publications examine simultaneous lotsizing and scheduling problems. Usually, they focus on extended model formulations or new solution heuristics. An up-to-date review can be found in Copil et al. (2017). Former reviews are provided by Drexl and Kimms (1997) and Zhu and Wilhelm (2006). We focus on the general lotsizing and scheduling problem for parallel lines (GLSPPL) of Meyr (2002), since it is quite general and can be used for problems of practical relevance (Meyr and Mann 2013). A solution heuristic for the GLSPPL has been introduced in Meyr (2002). Another solution procedure, which focuses more on large-scaled problems, can be found in Meyr and Mann (2013). In fact, the GLSPPL and other simultaneous lotsizing and scheduling models are not used to create plans for products but for (setup) families. A setup family includes several products which have no or only very small setup times between each other. Nevertheless, switching from one setup family to another one causes long setup times. (see, e.g., Meyr 2002, p. 277) Thus, the problems solved are smaller compared to the product-based original problems. Indeed, considering setup families is a reasonable approach. Nevertheless, companies often do not know which products constitute a setup family or they aggregate products to families in a way which still provides optimization potential to further reduce the model size.

We take-up this issue and propose a solution heuristic which creates setup families to reduce the model size. This heuristic significantly helps to solve large-scaled problems and also generates advantages if problems already consist of setup families since often further aggregation is possible to improve the solution performance. At the same time, we provide helpful insights about creating setup families in the case of sequence-dependent setup times which, to the best of our knowledge, has not been considered in literature so far. In detail, the first step of the heuristic aggregates products to setup families to formulate a modified master problem which is less complex than the original problem. In the following step, an up-to-date heuristic is used to solve the modified multi-line problem. The result is disaggregated to determine line-dependent demands. The resulting single-line problems are solved with another up-to-date heuristic and the production plan of the original problem is formed. Different settings are examined in numerical tests and prove the applicability of the approach to solve problems of practical relevance.

In Section 4.2, a short review provides insight into the literature of simultaneous lotsizing and scheduling. The GLSPPL is presented in Section 4.3. Afterwards, Section 4.4 is devoted to the explanation of the new solution approach. Numerical tests and their results are explained in Section 4.5. Finally, Section 4.6 summarizes the results and provides a short outlook on further research.

4.2 Literature review

The GLSPPL, mentioned in Section 4.1, generalizes other formulations of simultaneous lotsizing and scheduling problems. I.e., it is possible to restrict the GLSP by adapting the input parameters to form all of the following basic models (see Meyr 1999, pp. 82-84 or Copil et al. 2017, pp. 6-8): the discrete lotsizing and scheduling problem (DLSP) was first named by Fleischmann (1990). It consists of so-called microperiods, which allow at most one setup per period. Furthermore, production must take place for the complete period, or not at all (all-or-nothing assumption). Lasdon and Terjung (1971) already introduced a formulation based on this assumption but did not name it DLSP. The continuous setup lotsizing problem (CSLP) by Karmarkar and Schrage (1985) eliminates the all-or-nothing assumption but still allows only one product per microperiod. The proportional lotsizing and scheduling problem (PLSP) of Haase (1994) allows two products per microperiod if the first product was already set up in a former period. Finally, the capacitated lotsizing problem with sequence-dependent setups (CLSD) of Haase (1996) uses so-called macroperiods which allow several products per period. The sequence of the products during a macroperiod is formed by numbering the products during the period. Many extensions and different solution methods for all of these models exist and are summarized in Copil et al. (2017). The following paragraphs describe publications which put their focus on setup families and publications concerning the GLSP.

The main characteristic of setup families is having only small or even no setup times (minor setups) between products of the same family and longer setup times (major setups) between different families (see, e.g., Fleischmann 1994, p. 401, Smith-Daniels and Smith-Daniels 1986, p. 278 or Hax and Meal 1975). In most publications concerning simultaneous lotsizing and scheduling the term “product” is used in the model description. Of course, depending on the aggregation level of the input data, the term “product” can also represent setup families. Only a few publications directly use the word “setup family” (see, e.g., Toso et al. 2009). Nevertheless, there also exist model formulations which explicitly include the differentiation between minor and major setups. For example, Tempelmeier and Buschkühl (2008) consider products which differ in the characteristics color and tool use. Products which need the same tool are combined in setup families. Tool changes cause major sequence-independent setups. Additionally, color changes cause minor sequence-independent setups. Almeder and Almada-Lobo (2011) propose a model for a different problem and assume that minor setup times are zero and that sequence-dependent setup times between products which need different tools exist. Gicquel et al. (2009) propose an approach to reduce the changeover variables of a DLSP model. They assume that all products differ in the specification of several physical attributes, like color or dimension. Setup costs are accounted on basis of attribute changes. The authors also point out that if there are only two attributes, a major-minor-setup structure can be easily defined. However, none of the aforementioned publications put the focus on defining setup families and the corresponding parameters like production costs of a family.

The first GLSP formulation has been introduced by Fleischmann and Meyr (1997). This formulation represents sequence-dependent setup costs, minimum lotsizes and continuous setup states, i.e., the setup state is not lost during idle periods. In Meyr (1999), the formulation additionally includes sequence-dependent setup times and allows a continuous setup state or the loss of the setup state by defining the input parameters. Subsequently, many extensions of the model have been introduced. For example, several of the formulations incorporate multi-level bill-of-material structures and multiple production stages (to mention only a few: Meyr 2004, Fandel and Stammen-Hegener 2006, Lang 2010, Seeanner and Meyr 2013 or Mohammadi and Poursabzi 2014). Often, model development was inspired by practical applications. For instance, by the consumer goods industry in general (Günther et al. 2006, Pahl et al. 2011, Tiacci and Saetta 2012, Meyr and Mann 2013, Seeanner 2013, Seeanner and Meyr 2013 or Camargo et al. 2014) or more specifically by the food industry (Fleischmann and Meyr 1997, Meyr 2000 or Marinelli et al. 2007) or the beverage industry (Toledo et al. 2008a, Toledo et al. 2008b, Ferreira et al. 2009, Toledo et al. 2009, Ferreira et al. 2010, Toledo et al. 2010, Ferreira et al. 2012, Baldo et al. 2014, Toledo et al. 2014 or Toledo et al. 2015).

Research takes not only place within the scope of modeling, but also in the field of developing solution heuristics. Quite a lot of these heuristics are meta-heuristics based on local search or evolutionary algorithms (e.g., Meyr 2000, Meyr 2002, Toledo et al. 2008a or Figueira et al. 2013). Furthermore, MIP-based[19] approaches like fix&relax, fix&optimize and rolling horizon approaches are used in contemporary models (examples are Toso et al. 2009, Ferreira et al. 2010, Mohammadi 2010 or Baldo et al. 2014).

In the following, the focus is placed on heuristics being used during our decomposition approach to solve sub problems. Meyr (2000) proposes a heuristic for the single-line GLSP, which uses the local search meta-heuristic threshold accepting. In each iteration, he generates a setup sequence randomly from the neighborhood of the previous setup sequence. A new sequence is accepted if the resulting objective value is better or not worse than a certain threshold. The threshold values are lowered to reach convergence of the algorithm. The setup costs can be directly calculated by the sequence. For the calculation of the minimum holding costs, a network flow problem must be solved. To solve this problem, a dual network flow problem algorithm is used and the current solution is re-optimized each time the setup pattern is adapted. Using the dual network flow problem it becomes possible to refuse candidates without solving the problem to optimality. This algorithm is called TADR. It is adapted to the multi-line case in Meyr (2002). The main difference is having to solve a generalized network flow problem. This makes the approach only useful for medium-sized instances with two lines, eight periods and 19 products. This approach is called TAPLS.

[19]MIP – Mixed integer programming.

Meyr and Mann (2013) propose a decomposition approach (TA-agg), which shows similarities in the basic structure compared to our heuristic. The authors initiate a time aggregation of the original multi-line problem to create a less complex master problem. I.e., several macroperiods of the original model are aggregated to form one macroperiod of the adapted model. The reduced model is solved by a reimplementation of TAPLS which uses the standard solver GLPK instead of the algorithm for the generalized network flow problem (this algorithm is called TA-GLPK). After solving the aggregated multi-line problem, line-dependent demands are calculated based on the result. Afterwards, the single-line problems are solved using TADR. The results of the single-line problems serve as fixed setup pattern for a linear programming representation of the original model. Finally, the linear program (LP) is solved to further optimize the solution.

The solution approach we present is sketched in Meyr (1999, pp. 175-181). However, in this approach the time is aggregated as well which will be omitted in the following. Furthermore, we essentially improve the approach and provide important insights about the realization of the several steps of the algorithm.

Additionally, a decomposition approach which also relies on setup families is proposed by Mac Cawley (2014). However, his approach solves a very specialized simultaneous lotsizing and scheduling problem for wine bottling (see also Copil et al. 2017, p. 49). Based on the bottle type, the products are assigned to setup families. Major sequence-independent setups occur for changing the bottle type and minor sequence-dependent setups are respected for product changeovers. Macroperiods are used to represent eight-hour shifts. A major setup always lasts for a complete shift and only two minor setups are allowed per macroperiod. Product-specific demands must be fulfilled at the end of the planning horizon. Aggregating the demand for each setup family leads to a problem which is solved by a standard solver. Afterwards, the sequences and lotsizes of the products are determined. All in all, the model has many additional restrictions compared to the basic GLSPPL formulation. Therefore, the solution approach cannot be directly applied to solve GLSPPL models missing these special characteristics.

To sum up, the GLSP is a widely used and accepted model formulation for simultaneous lotsizing and scheduling problems. Several extensions and solution approaches exist as shown in the previous paragraphs. Nevertheless, there is still a need for faster solution approaches, especially for large-sized practical problems (see, e.g., Meyr and Mann 2013 concerning the GLSP).

4.3 GLSPPL model formulation

The GLSPPL formulation considers several products j ($j = 1,2,...,J$) which can be produced on multiple parallel production lines l ($l = 1,2,...,L$). The planning horizon is divided into multiple macroperiods t ($t = 1,2,...,T$). A given (time) capacity K_{lt} limits production on line l in macroperiod t. The consumed capacity while producing one unit of product j on line l is defined by a_{lj}. If a changeover from product i to product j takes place on line l, the given capacity is further reduced by a setup time st_{lij}. Additionally, setup costs s_{lij} occur. Upon completion of a setup, minimum lotsizes m_{lj} must be respected. Moreover, production costs c_{lj} must be calculated for each produced unit of product j on line l. A given demand d_{jt} must be met without backlogging for product j in macroperiod t. However, positive inventory is allowed and has to be penalized by holding costs h_j for storing one unit of product j for the length of one macroperiod.

Additionally to the macroperiod time grid, the planning horizon is further divided into microperiods s ($s = 1, 2, ..., S$). S_t defines the set of microperiods within macroperiod t. Thus, $|S_t|$ defines the number of microperiods in macroperiod t. The main functionality of microperiods is to define the sequence of produced products. This is realized by allowing at most one product per microperiod. The lengths of microperiods are not defined in advance and it is also possible that a microperiod has a length of zero.

Table 4.1: Symbols of the GLSPPL

Indices and sets:	
$i, j = 1, ..., J$	products; $i, j = 0$ neutral product
$l = 1, ..., L$	production lines
$s = 1, ..., S$	microperiods
$t = 1, ..., T$	macroperiods
S_t	set of microperiods s belonging to macroperiod t
Data:	
a_{lj}	capacity consumption (time) needed to produce one unit of product j on line l
c_{lj}	production costs of product j (per unit) on line l
d_{jt}	demand of product j in macroperiod t (units)
h_j	holding costs of product j (per unit and macroperiod)
I_{j0}	initial inventory of product j at the beginning of planning (units)
K_{lt}	capacity of line l in macroperiod t (time)
m_{lj}	minimum lotsize of product j (units) if produced on line l
s_{lij}	setup cost of a changeover from product i to product j on line l
st_{lij}	setup time of a changeover from product i to product j on line l
y_{lj0}	equals 1 if line l is set up for product j at the beginning of planning (0 otherwise)
Variables:	
$I_{jt} \geq 0$	inventory of product j at the end of macroperiod t (units)
$x_{ljs} \geq 0$	quantity of product j produced during microperiod s on line l (units)
$y_{ljs} \in \{0; 1\}$	setup state: y_{ljs} equals 1 if line l is set up for product j in microperiod s (0 otherwise)
$z_{lijs} \geq 0$	equals 1 if a changeover from product i to product j takes place on line l in microperiod s (0 otherwise)

x_{ljs} defines the production quantity of product j on line l in microperiod s. The inventory of product j at the end of macroperiod t is denoted by I_{jt}. It is possible to define a starting inventory I_{j0} of product j which is already available before macroperiod $t = 1$. The following variables are used to define and track the changeovers: z_{lijs} takes on 1 if a changeover from product i to product j takes place on line l in microperiod s (otherwise, its value is 0). If line l is set up for product j in microperiod s, the variable y_{ljs} takes on 1 (otherwise, its value is 0). y_{lj0} defines the initial setup state. It is equal to 1, if line l is set up for product j before the beginning of planning.

A product $j = 0$ is introduced to represent an idle state of a line at the beginning of planning. There is no demand of this product and after the first changeover on each line the product $j = 0$ is never used again, since the model formulation assumes that setup states are conserved during idle times ($st_{ljj} = s_{ljj} = 0 \ \forall l, j$).

All parameters and variables used in the model are summarized in Table 4.1. The model formulation is stated below.

Objective function:

$$\text{Min} \sum_{t,j} h_j I_{jt} + \sum_{l,i,j,s} s_{lij} z_{lijs} + \sum_{l,j,s} c_{lj} x_{ljs} \tag{4.1}$$

Constraints:

$$I_{jt} = I_{j,t-1} + \sum_{l,s \in S_t} x_{ljs} - d_{jt} \qquad \forall j,t \tag{4.2}$$

$$\sum_{j,s \in S_t} a_{lj} x_{ljs} \leq K_{lt} - \sum_{i,j,s \in S_t} st_{lij} z_{lijs} \qquad \forall l,t \tag{4.3}$$

$$x_{ljs} \leq \frac{K_{lt}}{a_{lj}} y_{ljs} \qquad \forall l,j,t,s \text{ with } s \in S_t \tag{4.4}$$

$$x_{ljs} \geq m_{lj}(y_{ljs} - y_{lj,s-1}) \qquad \forall l,j,s \tag{4.5}$$

$$\sum_j y_{ljs} = 1 \qquad \forall l,s \tag{4.6}$$

$$z_{lijs} \geq y_{li,s-1} + y_{ljs} - 1 \qquad \forall l,i,j,s \tag{4.7}$$

The objective function (4.1) sums holding, sequence-dependent setup and production costs. The resulting sum must then be minimized. The inventory of product j in macroperiod t is defined as the inventory of the previous period plus the production quantities of the current macroperiod minus the demand in macroperiod t (4.2). Constraints (4.3) assure that the capacity minus the time consumed during setups is not exceeded by the time used for production. Inequalities (4.4) assure that production of product j on line l in microperiod s can only take place ($x_{ljs} > 0$) if the line is set up for product j ($y_{ljs} = 1$). After a setup in microperiod s, the production quantity in microperiod s must be at least as big as the minimum lotsize m_{lj} (4.5). As already mentioned, only one product is allowed per microperiod and the setup state is conserved during idle periods, thus line l must be set up for exactly one product in each microperiod s (4.6). The correct recording of changeovers is realized by constraints (4.7). It should be mentioned that in an optimal solution the continuous variable z_{lijs} only takes on 1 or 0 (see (4.7) and (4.1)). Since the setup states are conserved during idle time, it is assumed that $st_{lij} = s_{lij} = 0 \ \forall l, i, j = i$ holds in the following.

4.4 Solution approach

In the following, a new decomposition heuristic to solve the GLSPPL is explained. The heuristic uses setup families to define an aggregated problem. Therefore, we propose two heuristics which differ in the assignment of products to families. Aggr-P constructs setup families based on considerations of how the aggregated plan represents the original problem as detailed as possible. A second approach, Aggr-PR, mainly uses random numbers to assign products to setup families. The general framework of the solution approach is explained in Subsection 4.4.1. The subsequent subsections provide detailed information about each step of the algorithm.

4.4.1 Framework of the heuristic

The starting point of the heuristic is the detailed model (4.1)-(4.7). The model is named PL/O and the iteration counter i of the heuristic is set to zero (see here and in the following Figure 4.1). Without loss of generality, let us assume that the initial inventory is zero and that there is positive demand for each product in at least one macroperiod. Section 4.4.2 explains why these assumptions are necessary for the heuristic and how they can easily be realized without changing the basic problem. At first, the complexity of the model is reduced by aggregating products to setup families g ($g = 1, 2, ..., G$) (step 1) and defining parameters (like production coefficients) for the resulting families (step 2). The following subsections start with the explanation of step 2. Afterwards, the assignment of products to setup families is explained. We changed the order of explanation because the difficulties at determining parameters of setup families provide many insights on how setup families should be constructed.

After this preliminary work, the first iteration $i = 1$ starts. The results of step 1 and 2 form a model which is identical to the formulation (4.1)-(4.7) despite of the fact that setup families are considered instead of products. Since this master problem $PL/Aggr$ is of less complexity compared to the original problem, it can be solved with up-to-date heuristics within a satisfying amount of time (step 3). The resulting schedule is disaggregated to obtain line-dependent product-specific demand quantities which are used to decompose the problem into several single-line problems SL_l (step 4). Each single-line problem is less complex compared to the original model and can be solved using up-to-date heuristics (step 5). Afterwards, the single-line production plans are used to form a fixed setup pattern for the original problem PL/O. Using this fixed setup pattern, PL/O can be formulated as linear program (LP) which is solved by, e.g., Gurobi Optimizer (step 6). This step may lead to a further optimized production plan.

It might happen that the final solution is infeasible due to an aggregation error, e.g., the accounted setup time of a setup family's lot is shorter than the sum of the setup times of the products which are assigned to this lot during disaggregation. If this is the case, it might happen that there is not enough remaining capacity to fulfill the complete line-dependent demand. Therefore, a further iteration is started, using reduced capacities K_{lt} (step 7). The intention is to reserve capacities which can be used in the single-line problems to compensate the lack of capacity caused by aggregation errors. The heuristic is stopped if step 6 creates a feasible plan or if a maximum number of iterations i_{max} is reached.

Comparing our algorithm to Meyr and Mann (2013), one will realize similarities. However, since

Meyr and Mann (2013) apply time aggregation, the heuristics have strong differences in their implementations.

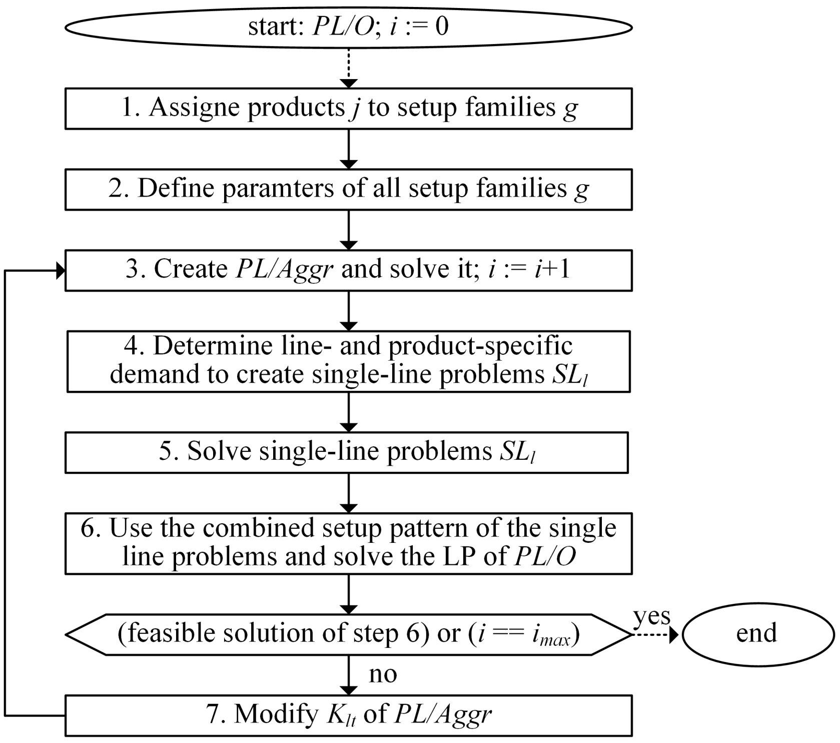

start: *PL/O*; *i* := 0
1. Assigne products *j* to setup families *g*
2. Define paramters of all setup families *g*
3. Create *PL/Aggr* and solve it; *i* := *i*+1
4. Determine line- and product-specific demand to create single-line problems SL_l
5. Solve single-line problems SL_l
6. Use the combined setup pattern of the single line problems and solve the LP of *PL/O*
(feasible solution of step 6) or ($i == i_{max}$)
yes
end
no
7. Modify K_{lt} of *PL/Aggr*

Figure 4.1: Framework of the heuristic

4.4.2 Determining parameters of setup families

The aim is to create an aggregated model which represents the original model as detailed as possible. Problems arise due to the aggregation of parameters because information gets lost. For example, in the detailed model, there are different production coefficients for product 1 and 2 and both products are assigned to setup family 1 which has only one production coefficient, thus, the detailed information of the production coefficients gets lost. In the following, we will look at all product-dependent parameters (i.e., K_{lt} is omitted since it is identical in the original and the aggregated model). There will be information about the arising difficulties, about how the parameters are aggregated and about the impacts on the process of assigning products to setup families. Used symbols can be found in Table 4.1. Additionally, the indices g and h ($g,h = 1,2,...,G$) define setup families. The set SF_g defines which products are included in setup family g, e.g., if product 1 and 2 are products of setup family 1, this is indicated by $SF_1 = \{1,2\}$. Furthermore, the superscript ^{g} is used to explicitly define parameters of setup families or the aggregated model $PL/Aggr$ in general, e.g., d^g_{gt} defines the demand of setup family g in period t.

The **demand** d^g_{gt} of setup family g in period t should be calculated by summing the demand parameters d_{jt} of all products of family g (see Eq. (4.8)) like it has already been done in Bitran and Hax (1977,

p. 42) for example. This approach directly implies that each product should be assigned exactly to one setup family (see Section 4.4.3.1 for a more detailed description of the basic assumptions concerning setup families).

$$d_{gt}^{g} = \sum_{j \in SF_g} d_{jt} \quad \forall g,t \tag{4.8}$$

Directly connected to the demand is the **initial inventory** I_{j0}, since it will be used to fulfill fractions of the demand. If the initial inventory is positive and would be summed up similarly to the demand (Eq. (4.8)), most probably aggregation errors will occur. Consider the following example: the initial inventories of products 1 and 2 are 3 and 2 units, respectively. Furthermore, the demand of product 1 is 5 units in period 1 (no demand of product 2). If both products belong to setup family 1 (i.e., $I_{10}^{g} = 5$), it might result in a feasible aggregated plan which defines to produce nothing in period 1. However, disaggregation is impossible because only 3 units of the necessary 5 units can be satisfied by the initial inventory of product 1. These considerations lead to the following procedure: calculate the net demand (also called effective demand) and set I_{j0} and I_{g0}^{g} to 0 for all products and setup families. The net demand d_{jt}^{net} of each product j is calculated in Equations (4.9) and should be applied in Equations (4.8) to calculate the net demand of all setup families.

$$d_{jt}^{net} = \begin{cases} \max\left\{0, \sum_{u=1}^{t} d_{ju} - I_{j0}\right\}, & \forall j, t = 1,...,t^* \\ d_{jt} & \forall j, t = t^* + 1,...,T \end{cases} \tag{4.9}$$

In Equations (4.9), t^* defines the period in which the initial stock level reaches zero. I.e., from period $t^* + 1$ the net demand is equal to the original demand. (See Bitran and Hax 1977, p. 41 and Stadtler 1988, pp. 97f). Notice, for a fair comparison with other heuristics, holding costs of initial inventories have to be externally calculated.

Concerning the assignment of products to setup families, which is described in detail in Section 4.4.3, it is necessary to decide if products which show zero net demand in all periods should be considered in the new solution approach. Of course, if a company plans frequently due to a short planning horizon and the scenario changes only in the demand pattern, computing time to determine setup families could be saved by doing the assignment of products to families only once. Nevertheless, we have decided to exclude all products with zero net demand, to reduce the complexity of the model and to avoid any influence of those products to the families' parameters (influences might occur if the average of product-dependent parameters is calculated and serves as parameter of a family).

To avoid disaggregation problems, it is preferable that the **production coefficient** a_{lj} is identical for all products of a family. In this case it would be possible to take the production coefficient of one product of the family to represent the production coefficient of the setup family[20] (see Leisten 1996, p. 20; this approach is also called projection, see Steven 1994, p. 49). For example, if the production coefficient of each product of a family is 1 time unit per produced unit, the production coefficient of the family would also be set to 1. If an aggregated plan schedules 5 units of the family in period 1, regarding the production coefficients, this reserved time can be used to produce 5 units as an arbitrary

[20]Notice, the production coefficients could still be different for each line since a_{lg}^{g} is also line-dependent.

combination of products of the affected family. On the contrary, assume that one product of this family has a production coefficient of 2 time units per produced unit, all other products show 1 as production coefficient and 1 is chosen as production coefficient of the family. Then, there might arise problems during disaggregation, since the resulting production quantities depend on the products assigned to the reserved time.

Of course, it will be necessary to assign products with different production coefficients to the same setup family in order to reach a sufficiently small number of setup families. In this case, it seems reasonable to create the production coefficients of the families by weighting the production coefficients of the corresponding products using the net demand. The process of weighting to create parameters of a family (also called aggregation, see Kleindienst 2004, p. 5) is a very common approach and can be found in many publications, see, e.g., Hallefjord et al. (1993, p. 105), Storoy (1996, p. 30) and Schneeweiß (2003, p. 175). Equations (4.10) show how the production coefficient a^g_{lg} of family g of line l is determined (see Kleindienst 2004, p. 72).

$$a^g_{lg} = \frac{\sum_{j \in SF_g} (\sum_t d^{net}_{jt}) a_{lj}}{\sum_{t, j \in SF_g} d^{net}_{jt}} \quad \forall l, g \tag{4.10}$$

In the enumerator of Equations (4.10), the production coefficient of each product and line is multiplied by the total net demand of this product. Afterwards, the resulting values are summed over all products of the currently calculated setup family g. The denominator shows the total net demand of all products of setup family g.

In practical cases, not every product can be produced on all lines. This could be implemented by choosing a very high production coefficient for products which are forbidden on a line. However, this could induce numerical problems. Therefore, we define that a production coefficient $a_{lj} = -1$ indicates that product j cannot be produced on line l and incorporate this in our implementation of the decomposition heuristic. Furthermore, we introduce the term "a_{lj}-functionality" and define that two products i and j have the same a_{lj}-functionality if the following condition is true: $(a_{lj} = a_{li} = -1)$ or $(a_{lj} > 0$ and $a_{li} > 0)$ $\forall l$.

Since $a_{lj} = -1$ will lead to unreasonable production coefficients, it seems useful to substitute the first sum in the denominator of Eq. (4.10) by $\sum_{j \in SF_g, a_{lj} > 0}$, i.e., exclude products which cannot be produced on line l. However, this could lead to problems as the following example shows. Think about two lines and two products of the same setup family ($g = 1$). Both products show positive net demand and can be produced on line 2, thus, Equations (4.10) can be used to calculate the family's production coefficient of line 2. For line 1 product 1 has a production coefficient of $a_{11} = 1$ and product 2 cannot be produced at all ($a_{12} = -1$). Two general possibilities exist in this case. In the first one, setup family 1 can be produced on line 1, in the second one, this is forbidden. In the first case, it might happen that in the aggregated plan no capacity for family 1 is left over on line 2 since it is used for other families and all necessary capacity for family 1 is reserved on line 1. In this case, it is impossible to disaggregate the plan, since product 2 cannot be produced on line 1. In the other case, where the production of setup family 1 is forbidden on line 1, it might happen that the capacity of line 2 might not be enough to produce the complete demand of product family 1 which also leads to feasibility problems. It should be mentioned that there is a chance that in both cases no problems arise but this is mainly based on fortune. In order to avoid this problem and following the main assumption that only products which have identical or nearly identical production coefficients should be aggregated in a family, products with different a_{lj}-functionalities are not aggregated in a setup family (see also Section 4.4.3). Therefore, a_{lg}^{g} is set to -1 for all l,g with $a_{lj} = -1$ and $j \in SF_g$.

Considering the **production costs** c_{lj}, it is ideal as well if they are identical or at least quite similar among the products of a setup family. Obviously, they have no direct influence on the possibility to disaggregate the aggregated plan in a feasible way. However, they have an influence on the quality of the resulting plans. Since the total production costs of the detailed plan are dependent on the production quantities, production costs parameters should be weighted by the net demand as it is shown in Equations (4.11).

$$c_{lg}^{g} = \frac{\sum_{j \in SF_g} (\sum_t d_{jt}^{net}) c_{lj}}{\sum_{t, j \in SF_g} d_{jt}^{net}} \quad \forall l, g \tag{4.11}$$

Holding costs h_j have no direct influence on the feasibility of the disaggregation process, but will influence the resulting production plans. Again, it is preferable that they are identical or nearly identical for all products of a setup family. Since the stored quantities of the different products are not known in advance, it might be a good approach to average the different holding costs like it is done in Equations (4.12) ($|SF_g|$ defines the number of products in setup family g).

$$h_g^g = \frac{\sum_{j \in SF_g} h_j}{|SF_g|} \quad \forall g \tag{4.12}$$

Contrary to the aforementioned parameters, identical **minimum lotsizes** m_{lj} of products belonging to the same setup family provide no advantage. For minimum lotsizes, it is preferable that they are zero or at least very small. To clarify this, think about a setup family $g = 1$ consisting of eight products. Each product has a minimum lotsize of 20 units and the production coefficients of each product-line-combination are $a_{lj} = 1$. It is impossible to estimate how much time has to be reserved to produce minimum lotsizes if a lot of setup family 1 is started, because it is unpredictable how many different product lots will be assigned to this setup family's lot during disaggregation. If all products are produced during this lot, the minimum lotsize is 160 units, if only one product is assigned to this lot, the minimum lotsize is 20 units. Over- and underestimation of the family's minimum lotsize can lead to problems. If the minimum lotsize is too big, it might be impossible to find a feasible plan for $PL/Aggr$ due to the restricted overall capacity. If the reserved time is too short, there might be problems to solve the single line problems SL_l since not enough time is reserved to fulfill the minimum lotsizes of all products which are assigned to a setup family's lot.

However, if the minimum lotsize is underestimated in $PL/Aggr$, it could happen that the lotsize of the setup family is big enough to fulfill all minimum lotsizes of the associated products. Thus, the minimum lotsize is defined as the average of all minimum lotsizes of all products of a setup family. In the case of different production coefficients of the products of a family, it seems reasonable to transfer the minimum lotsizes to minimum time spans, take the average and transfer this value back to units by multiplying it by the production coefficient of the family. The approach of transferring a measuring unit of a parameter from units to time units is already used in Kleindienst (2004, p. 56). Equations (4.13) formally show how the minimum lotsizes of setup families are defined.

$$m^g_{lg} = \frac{\sum_{j \in SF_g} m_{lj} a_{lj}}{|SF_g|} * \frac{1}{a^g_{lg}} \quad \forall l, g \tag{4.13}$$

The **initial setup state** y^g_{lg0} can be easily defined by calculating the maximum of the initial setup states of all products of a family.[21] I.e., if the product which is set up on line l at the beginning of the planning horizon is included in family g, line l is initially setup for family g. This can be represented formally using Equations (4.14).

$$y^g_{lg0} = \max_{j \in SF_g} \{y_{lj0}\} \quad \forall l, g \tag{4.14}$$

This paragraph explains how **setup times** st^g_{lgh} of changeovers from family g to h are calculated. The following example, which can easily be generalized, shows why it is difficult to define the setup time of a changeover from setup family $g = 1$ to setup family $h = 2$. Setup family 1 consists of products 1 and 2 and setup family 2 consists of products 3 and 4. The corresponding (non-preferable) setup times of line 1 are shown in Table 4.2. Obviously, there exist "setup times between products of the same family" (i.e., for the switchover from product 1 to 2 and vice versa and for the switchover from 3 to 4 and vice versa) and there are "setup times between the products of the different families" (i.e., for the changeovers from 1 to 3, from 2 to 3, from 1 to 4, from 2 to 4 and all resulting vice versa combinations).

Table 4.2: Example: setup times between products

st_{1ij}	1	2	3	4
1	0	5	2	13
2	3	0	4	15
3	6	7	0	6
4	8	9	4	0

In order to estimate the setup time of the changeover from family 1 to 2, it is necessary to determine the arising setup time in the disaggregated case. At first, consider the setup times between the products of different families (setup times between products of the same family are considered later on by adapting the production coefficients). The disaggregated setup time depends on the product produced last of family 1 ($j = 1$ or 2) and it also depends on the product which is the first product of family 2 (i.e., whether product 3 or 4 is produced), i.e., it differs between 2 and 15 time units. It is impossible to estimate the first and last product of each lot of a setup family in advance, thus, it is reasonable to choose the average ($st^g_{112} = 8.5$ time units in the example). Obviously, the aggregation error is smaller if the setup times are nearly identical or is even zero if the setup times are identical. Thus, it is favored to construct setup families g having identical setup times from all products of family g to all

[21] It is also possible to take the sum of all initial setup state parameters of all products of a family.

products of setup family $h(\neq g)$ (see Section 4.4.3.2). Equations (4.15) define how the average of the product dependent setup times is calculated to determine the setup times st^g_{lgh} (remember, each product is assigned to exactly one family). Setup times of changeovers from a setup family to itself are set to 0 (see Eq. (4.16)) since setup states are conserved during idle periods in the model of Section 4.3.

$$st^g_{lgh} = \frac{\sum_{i\in SF_g, j\in SF_h} st_{lij}}{|SF_g| * |SF_h|} \quad \forall l, g, h \neq g \tag{4.15}$$

$$st^g_{lgh} = 0 \quad \forall l, g, h = g \tag{4.16}$$

In the following, the influences of the setup times between the products of a setup family are discussed. In the example, it is possible to increase the setup time st^g_{112} to also consider the setup times which occur between products in the lot of setup family $h = 2$. Estimating this setup time is even more difficult because it depends on the number of product lots and on the products which are affected. Thus, it would be preferable if the setup times between products of the same family are zero or at least very small. Nevertheless, in most cases the setup times will not be small enough to be ignored. However, the approach of increasing the setup time seems to be too imprecise. Therefore, the following approach is applied.

It can be assumed that there is a positive correlation between the length of a setup family lot and the number of products which are produced during the reserved time of this lot. Since the lotsizes are not known in advance, it seems to be a good approach to add a supplement on the production coefficient which represents the occurring setup times between products of a setup family. The advantage is that in a bigger lot more time and in a smaller lot less time is reserved for setups between products of a family. Equations (4.17) show how the production coefficient supplement can be calculated. The first term in the enumerator defines the average setup time between two products of setup family g ($-|SF_g|$ excludes the changeovers from one product to itself). For the calculation, we assume that each setup family is only set up once in the planning horizon and all products of the family are produced during this lot. This assumption is represented by the second term of the enumerator (notice, the setup time of the first product of the lot is already included in the family setup time). All in all, the enumerator defines the time which should be reserved for setups. This time is divided by the total net demand of family g. The resulting value defines how many time units should be reserved for setups per produced unit of setup family g. Depending on the demand structure, it might happen that there is a very large lot which is used to produce only two products. In this case, the reserved time for setups will be too long. To counteract this effect, the supplement to the production coefficient is divided in half, since not further documented tests runs have shown that this value seems to be reasonable. This correction supplement still is an approximation but increases the probability of feasible solutions of $PL/Aggr$. The production coefficient supplement must be added to the production coefficient of Eq. (4.10).

$$\text{correction supplement of } a^g_{lg} := \frac{\frac{\sum_{i\in SF_g, j\in SF_g, i\neq j} st_{lij}}{|SF_g| * |SF_g| - |SF_g|} * (|SF_g| - 1)}{\sum_{t, j\in SF_g} d^{net}_{jt}} * 0.5 \quad \forall l, g \tag{4.17}$$

Setup costs s_{lij} are treated similar to the setup times. Nevertheless, they have no direct influence on the feasibility of the plans. The setup costs of a changeover from family g to h are defined as the average setup costs of all changeovers from all products of family g to all products of family h (see Eq. (4.18)). The setup costs of a changeover from a family to itself are set to 0 (see Eq. (4.19)), since the setup state is conserved. The costs of changeovers between products of the same setup family are ignored. Of course, it is possible to adapt the production costs like it was done for the production coefficient in Eq. (4.17) or add a supplement to the setup costs of Equations (4.18). However, the benefit is unpredictable, thus, it is omitted.

$$s^g_{lgh} = \frac{\sum_{i \in SF_g, j \in SF_h} s_{lij}}{|SF_g| * |SF_h|} \quad \forall l, g, h \neq g \tag{4.18}$$

$$s^g_{lgh} = 0 \quad \forall l, g, h = g \tag{4.19}$$

Finally, the **number of microperiods per macroperiod** $|S^g_t|$ of $PL/Aggr$ must be defined. A common approach is to set the number of microperiods per macroperiod to the number of products (see Meyr 1999, p. 84) and increase it if necessary. This approach crucially reduces the number of resulting variables compared to the detailed model. Equations (4.20) show the formal representation of this approach.

$$|S^g_t| = G \quad \forall t \tag{4.20}$$

4.4.3 Assigning products to setup families

The following sections describe how products are assigned to setup families. Generally, this process of aggregating objects with similar characteristics into groups is called “clustering” (for a detailed review see, e.g., Jain et al. 1999). To the best of our knowledge, no publications which consider clustering in the context of sequence-dependent setup times or different a_{lj}-functionalities exist. Also common clustering methods, like k-means (see, e.g., Jain et al. 1999, pp. 278f), do not directly consider these topics. Thus, the insights of Section 4.4.2 are used to formulate a clustering algorithm. Since setup times are one of the main aspects, the algorithm is called Cluster-S. Additionally, an approach which does the assignment mainly based on random choice (Cluster-R) is proposed to serve as benchmark. It should be mentioned that sometimes there is a natural clustering pre-defined by the product characteristics and setup process itself. For example, if there are two product characteristics, like size and color, and it is known that a size change is complicated and consumes a lot of time and a color change consumes only a few minutes, obviously, all products of the same size should be aggregated in a setup family. Of course, Cluster-S is intended to exactly lead to the same result. Nevertheless, Cluster-S will also work well if only the setup times without further information of characteristics are available.

Section 4.4.3.1 defines some basic assumptions concerning the clustering algorithms. The subsequent section explains the Cluster-S algorithm. Later on, in Section 4.4.3.3 Cluster-R is introduced.

4.4.3.1 Basic assumptions

In the following, we assume that each product must exactly be assigned to one setup family. However, it is allowed that a setup family consists of only one or multiple products. Equations (4.21)-(4.23) summarize these assumptions in a mathematical way (see Steven 1994, p. 47). Notice, the assignment of products to families is identical for all lines.

$$SF_g \cap SF_h = \emptyset \quad \forall g, h \neq g \tag{4.21}$$

$$SF_g \neq \emptyset \quad \forall g \tag{4.22}$$

$$\bigcup_g SF_g = \{1, ..., J\} \tag{4.23}$$

Product $j = 0$, the product which indicates the idle state and which only might be used at the beginning of planning, is always assigned to family $g = 0$. It is not allowed to assign an additional product to this setup family.

4.4.3.2 Cluster-S

This section describes the Cluster-S algorithm. The algorithm is based on the insights of Section 4.4.2, thus, one main aspect is the consideration of setup time characteristics. Figure 4.2 shows an example of setup times and product-to-family-assignments which, concerning the setup times, will lead to no aggregation error.

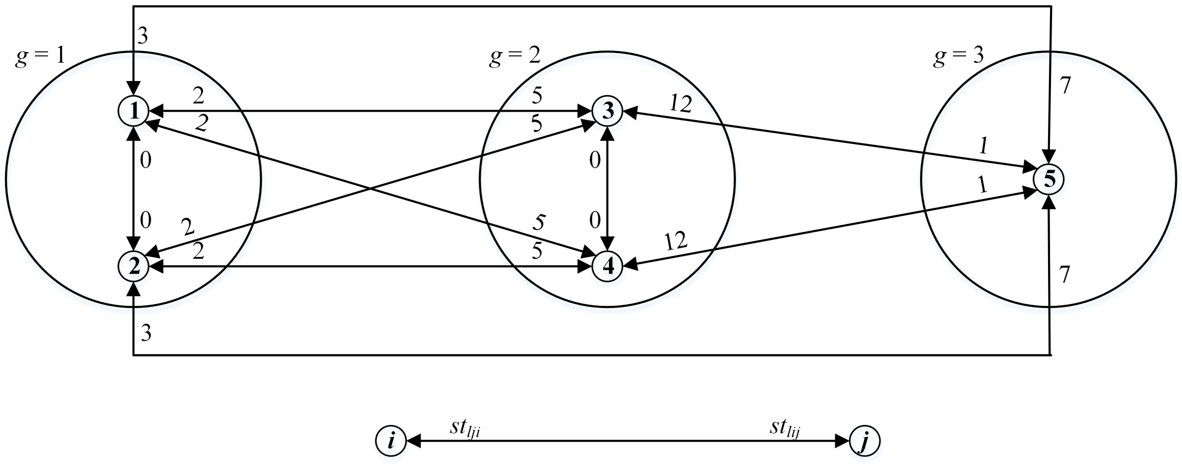


Figure 4.2: Setup times and product-to-family-assignments which, concerning the setup times, will lead to none aggregation error

Setup family $g = 1$ consists of products 1 and 2, setup family $g = 2$ includes products 3 and 4 and product 5 is solely assigned to setup family $g = 3$. The setup times between products of the same family are zero. Furthermore, the setup times of changeovers from all products of family 1 to all products of family 2 are identical (5 time units). Additionally, all setup times of changeovers from products of family 1 to the single product of setup family 3 are identical as well (7 time units). Looking at the other setup times, the same scheme can be identified. Both characteristics shown in the example are mathematically represented in the Equations (4.24) and (4.25).

$$st_{lij} = 0 \quad \forall l, g, i \text{ and } j \in SF_g, i \neq j \tag{4.24}$$

$$st_{lik} = st_{ljn} \quad \forall l, g, h \neq g, i \text{ and } j \in SF_g, k \text{ and } n \in SF_h \tag{4.25}$$

Equations (4.24) define that the setup time of a changeover between two products of the same family g must be zero. This must hold for the concerned setup times of all lines and for all setup families. Recognize, by iterating over all i and j, not only the changeovers from product i to j but also vice versa changeovers are included (e.g., if there is one line and one setup family consisting of the products 1 and 2, the following i-j combinations arise: 1-2 and 2-1).

Equations (4.25) postulate that the setup time of a changeover from product i of family g to product k of family $h(\neq g)$ on line l must be identical to the setup time of a changeover from product j of family g to product n of family h on line l. This equation must be feasible for all combinations of lines, setup families and concerned products. Notice, Equations (4.25) include all changeovers from family g to h and vice versa (e.g., if there are three setup families and only one line, there will be six equations with the following g-h combinations (for demonstrational reasons, combinations from and to family 1 are underlined): $\underline{\text{1-2}}$, $\underline{\text{1-3}}$, $\underline{\text{2-1}}$, 2-3, $\underline{\text{3-1}}$ and 3-2).

Since Cluster-S defines the setup families step by step, it is necessary to formulate Equations (4.25) more general like it is done in Equations (4.26).

$$st_{lik} = st_{ljk} \text{ and } st_{lki} = st_{lkj} \quad \forall l, g, i \text{ and } j \in SF_g, k \notin SF_g \tag{4.26}$$

Equations (4.26) claim that the setup time from one product i of family g to product k, which is not included in family g, is identical to the setup time from product j of family g to the same product k. Additionally, the setup time from k to i must be identical to the setup time from k to j. For the Cluster-S algorithm, it is necessary to interpret Equations (4.26) in the following way: two products i and j should be in the same family if they fulfill the characteristic represented in Equations (4.27).

$$st_{lik} = st_{ljk} \text{ and } st_{lki} = st_{lkj} \quad \forall l, k \tag{4.27}$$

This can be formulated similarly for Equations (4.24): the products i and j should be aggregated in a setup family if Equations (4.28) hold.

$$st_{lij} = 0 \text{ and } st_{lji} = 0 \quad \forall l \tag{4.28}$$

It is unreasonable to construct an algorithm that generates setup families which strictly fulfill Equations (4.27) and (4.28) because in this case, the number of families will not be much smaller than the original number of products. The extreme case would be that each product has to be solely assigned to a setup family, resulting in an aggregated problem which is identical to the original problem. Thus, the postulations of the equations are relaxed in the Cluster-S algorithm. Equations (4.28) are relaxed considering two aspects. One aspect ("relaxation 1") is that setup times must only be lower than or equal to a certain value $st^{small}(\geq 0)$. If this is the case, we will refer to such a setup time as "small setup time" in the following. The other aspect is that even the postulation $st_{lij} \leq st^{small}$ and $st_{lji} \leq st^{small}$ $\forall l$ can be disregarded for some changeovers ("relaxation 2"). I.e., two products i and j can be aggregated in a family even if some setup times are higher than st^{small} ($st_{lij} > st^{small}$ or $st_{lji} > st^{small}$ for some l). Equations (4.27) could be relaxed in a similar way. However, many equations would have to be considered. Therefore, we decided to formulate it more compact as shown in Equation (4.29). Two products i and j should be in the same family if they fulfill the characteristic represented in Equations (4.29).

$$\sum_{l,k} |st_{lik} - st_{ljk}| + \sum_{l,k} |st_{lki} - st_{lkj}| = 0 \tag{4.29}$$

I.e., all absolute values of deviations of setup times st_{lik} and st_{ljk} and of setup times st_{lki} and st_{lkj} are summed up. The outcoming sum must be equal to zero. The postulation of Equation (4.29) can easily be relaxed by allowing values greater than zero on the right-hand side ("relaxation 3"). Later on, we will refer to the characteristic represented in Equation (4.29) as "similarity of two products" or as "deviation between two products". For example, Equation (4.29) shows that product i and j are very "similar" (in this case, even identical) or accordingly, the products are only a little bit (in this case, even not at all) "different". Defining similarity measures is a common approach used in clustering methods (see, e.g., Jain et al. 1999, pp. 271-274). Notice, the intensity of the relaxations, i.e., in the case of relaxation 3, the allowed deviation from zero on the right-hand side of Eq. (4.29), influences the number of resulting setup families G, thus, the number of setup families cannot be defined in advance. The intensity of the relaxations must be defined considering the tradeoff between a low aggregation error, reached due to a high number of homogeneous setup families (weak relaxation, i.e., in the case of relaxation 3, low value on the right-hand side of Eq. (4.29)), and a low number of heterogeneous setup families (strong relaxation, i.e., in the case of relaxation 3, high value on the right-hand side of Eq. (4.29)) which will lead to a less complex aggregated problem $PL/Aggr$.

The Cluster-S algorithm divides into two phases. The first phase defines product pairs (i, j) which show small setup times for the changeovers from i to j and vice versa (Eq. (4.28)). The second phase examines if the products i and j of a pair of phase 1 are similar as postulated by Equation (4.29). If the products are similar, they are assigned to the same setup family. Furthermore, it is examined if other setup pairs (i, j) should be assigned to this family as well. The following paragraphs explain the Cluster-S algorithm in detail. The main aspects discussed are the detailed definition of the relaxations and the handling of difficulties which arise due to products which are forbidden on some lines ($a_{lj} = -1$).

The first step of phase 1 (see Algorithm 1 Line 1)[22] is to define a list which includes all possible product pairs (i, j). At the beginning, this list of *possible pairs* consists of all combinations $M \times N | M = \{i | i = 1, 2, ..., J\}, N = \{j | j = 1, 2, ..., J\}, i < j, i \neq j$.[23] For example, for a scenario consisting of three products the list is defined as $\{(1,2), (1,3), (2,3)\}$.

The next step (see Line 2) deletes all product pairs (i, j) which do not have the same a_{lj}-functionality (see page 123) from the list of *possible pairs*. I.e., product pairs which show $a_{li} = -1$ (i is forbidden on line l) and $a_{lj} > 0$ (j can be produced on line l) on at least one line are deleted since otherwise problems arise by defining production coefficients of the families (see Section 4.4.2 for a deeper motivation).

Product pairs, of which both products cannot be produced on one or several lines, remain in the list of *possible pairs*. Additionally, a second list *relevant info* is created on basis of the current list of *possible*

[22]Pseudocodes in this publication does not show every step in detail.

[23]$\times$ defines the cartesian product ($A \times B = \{(x,y) | x \in A, y \in B\}$).

pairs. The new list consists of (l,i,j)-combinations and will be used later on to identify changeovers which show small setup times. To provide a very small example how the list is constructed (see Lines 3-9 of Algorithm 1) consider the following scenario. There are two production lines, the current list of *possible pairs* consists of the single pair $(2,3)$ and the products 2 and 3 can only be produced on line 2. In this case, the list *relevant info* is defined as $\{(2,2,3),(2,3,2)\}$, i.e., changeovers on the forbidden line are excluded. As it can be seen in Line 4 of the algorithm, a counter $c^{irr\_l}_{(i,j)}$ is introduced. This counter of irrelevant lines of pair (i,j) is increased by 2 if both products i and j cannot be produced on a line. This is necessary, since later on the number of small setup times of a product pair (i,j) will be counted to create a ranking. For this ranking, setup times of changeovers of possible pairs (i,j) which can never occur, like those counted by $c^{irr\_l}_{(i,j)}$, are defined as small setup times to assure a fair ranking. The following example should clarify the problem. Assume two production lines and four products i, j, m and n. Products i and j can be produced on both lines, while product m and n can only be produced on line 1. The setup times for a changeover from i to j and vice versa on line 1 are 0. The setup times for a changeover from i to j and vice versa on line 2 are 900, which is assumed as very large setup times. As one can see, the pairs (i,j) and (m,n) have an identical number of small setup times. However , (m,n) only consists of small setup times while (i,j) also shows large setup times.

In Lines 10-20 (l,i,j)-combinations which show large setup times are deleted from the list *relevant info*. I.e., after this procedure, the list consists merely of (l,i,j)-combinations which have small setup times. The first step is to sort the list *relevant info* by increasing setup times (see Line 10). The next step is to iterate over all (l,i,j)-combinations of the list *relevant info*. If the associated setup time is smaller than or equal to a certain boundary st^{small} (relaxation 1), the (l,i,j)-combination is left in the list and it is continued with the next iteration. However, a strict boundary can cause undesired effects. Think about the following setup times of a possible product pair (i,j) sorted in increasing order: $0.5, 0.8, 1.1, 4.7, 4.7, 5$ time units. If a setup time smaller than or equal to $st^{small} = 1$ time unit is defined as small setup time, only the first two setup times are defined as small. Nevertheless, it seems reasonable that the setup time 1.1 should also be defined as small because it is very close to the previous setup time. However, 4.7 should not be defined as small setup time since it is much higher than 1.1. However, practical instances have much more products and thus much more setup times. In such cases, it is desired to detect the boundary between small and large setup times in a more automatic way. Therefore, setup times which are greater than st^{small} and smaller than or equal to an upper bound st^{large} are analyzed in the following way (see Line 14). Calculate the difference between the setup time of the current iteration of the for-loop which iterates over the sorted list *relevant info* (Line 12) and the setup time of the previous iteration $st_{lij,prev}$. If this difference is smaller than or equal to a given boundary st^{max_diff}, the setup time is defined as small. Otherwise, it is a large setup time and must be deleted from the list of *relevant info* (see Line 17). After the first large setup time is identified, all setup times of the following iterations are defined as large setup times and deleted from the list. The variable v^{help} helps to implement this functionality (see Lines 11, 13, 18 and 20). Notice, setup times which are higher than st^{large} are definitely deleted from the list *relevant info* (see Line 14).

To allow simple setting, the values of st^{small}, st^{large} and st^{max_diff} are defined as follows. The parameters st^{small} and st^{large} are defined as percentage of the maximum setup time of the current scenario.

Algorithm 1: Cluster-S phase 1 (main input: a_{lj}, st_{lij})

```
1  possible pairs := $\{M \times N | M = \{i | i = 1,2,...,J\}, N = \{j | j = 1,2,...,J\}, i < j, i \neq j\}$; // create the list possible pairs
2  if $i$ and $j$ do not have the same $a_{lj}$-functionality, delete $(i,j)$ from the list possible pairs;
3  forall $(i,j) \in$ possible pairs do // create list relevant info:
4      $c^{irr\_l}_{(i,j)} := 0$; // counter of irrelevant lines
5      forall $l$ do
6          if $a_{li}$ and $a_{lj} > 0$ then
7              add $(l,i,j)$ and $(l,j,i)$ to the list relevant info;
8          else // i.e., $a_{li}$ and $a_{lj} = -1$
9              $c^{irr\_l}_{(i,j)} := c^{irr\_l}_{(i,j)} + 2$;
10 sort the list relevant info by increasing setup times;
11 $v^{help} := 0$;
12 forall $(l,i,j) \in$ relevant info do // delete large setup times from the list relevant info:
13     if $v^{help} == 0$ then
14         if $st_{lij} \leq st^{small}$ or ($st_{lij} \leq st^{large}$ and $st_{lij} - st_{lij,prev} \leq st^{max\_diff}$) then
15             continue with next iteration of forall-loop;
16         else
17             delete $(l,i,j)$ from the list relevant info;
18             $v^{help} := 1$;
19     else
20         delete $(l,i,j)$ from the list relevant info;
21 forall $(i,j) \in$ possible pairs do // consider relaxation 2:
22     $small\_st^{num}_{(i,j)} := 0$;
23     forall $l$ do
24         if $(l,i,j) \in$ relevant info then
25             $small\_st^{num}_{(i,j)} := small\_st^{num}_{(i,j)} + 1$;
26         if $(l,j,i) \in$ relevant info then
27             $small\_st^{num}_{(i,j)} := small\_st^{num}_{(i,j)} + 1$;
28     $small\_st^{num}_{(i,j)} := small\_st^{num}_{(i,j)} + c^{irr\_l}_{(i,j)}$;
29     $small\_st^{perc}_{(i,j)} := (small\_st^{num}_{(i,j)})/(2L)$;
30     if $small\_st^{perc}_{(i,j)} < small\_st^{perc\_allowed}$ then
31         delete $(i,j)$ from the list possible pairs;
```

And st^{max_diff} is defined as percentage of the maximum difference which arises between two consecutive setup times of a list including all setup times in an increasing order. As already mentioned, the choice of the aforementioned parameters which relax Equations (4.28) is affected by the tradeoff between a high number of homogenous setup families and a low number of inhomogeneous setup families.

The next step considers relaxation 2. At first, iterate over all pairs (i,j) of the list of *possible pairs* (see Line 21) and calculate how often Equation (4.28) or the relaxation of it is fulfilled. I.e., calculate the number of combinations (l,i,j) and (l,j,i) which are associated with small setup times (see Lines 22-27) and store the result in the variable $small\_st^{num}_{(i,j)}$. Additionally, the number of irrelevant lines c^{irr_l} is added to $small\_st^{num}_{(i,j)}$ since setup times of changeovers which do not occur (both products are forbidden on a line) are defined as small setup times as well (see Line 28). The maximum number of small setup times which can arise for a pair (i,j) is $2L$, i.e., both directions of changeovers (i to j and j to i) are considered for all lines. As shown in Line 29, the percentage of small setup times, in relation to the maximum possible number of small setup times, is calculated for all pairs (i,j) and stored in the variable $small\_st^{perc}_{(i,j)}$. Equation (4.28) does not have to be fulfilled for all possible changeovers of a pair (i,j) (relaxation 2). I.e., pairs (i,j) which have a small $small\_st^{perc}_{(i,j)}$-value which is greater or equal to a parameter $small\_st^{perc\_allowed} (> 0)$ are still potential candidates which may be assigned together to one setup family. As result, all pairs not fulfilling $small\_st^{perc}_{(i,j)} \geq small\_st^{perc\_allowed}$ must be deleted from the list of *possible pairs* (see Lines 30-31). Notice, all product pairs (i,j) of which all changeovers cause large setup times are deleted from the list of *possible pairs*. Consequently, products i and j of such a pair will never be aggregated in one setup family.

Phase 2 of Cluster-S (see Algorithm 2) decides which products should be actually aggregated in a setup family. The list of *possible pairs* of phase 1 of Cluster-S serves as starting point. Since pairs having a high $small\_st^{perc}_{(i,j)}$-value fulfill Equation (4.28) in many cases, it is reasonable to start the decision process with such candidates. I.e., the list of *possible pairs* is ordered by decreasing $small\_st^{perc}_{(i,j)}$ values (see Line 1 of Algorithm 2). There might be pairs which have an identical $small\_st^{perc}_{(i,j)}$ value. In these cases, the pairs are ranked by increasing average deviations $dev^{av}_{(i,j)}$ (see Lines 2-4). Equations (4.30) are an adaption of Equation (4.29) and show how $dev^{av}_{(i,j)}$ is calculated. Since these deviations are also used later on to decide if two products i and j are similar enough to be assigned to one setup family, $dev^{av}_{(i,j)}$ is directly calculated for all product pairs of the list of *possible pairs* (see Equations (4.30) and Lines 2-3 of the algorithm).

$$dev^{av}_{(i,j)} = \frac{\sum\limits_{l,k,a_{lk}>0,a_{li}>0} |st_{lik} - st_{ljk}| + \sum\limits_{l,k,a_{lk}>0,a_{li}>0} |st_{lki} - st_{lkj}|}{\sum\limits_{l,k,a_{lk}>0,a_{li}>0} 2} \quad \forall (i,j) \in possible\ pairs \tag{4.30}$$

The enumerator of Equation (4.30) defines the sum of the absolute values of the deviations between setup times of changeovers from i to k and from j to k and vice versa. Different to Equation (4.29), all changeovers which never occur because at least one of the products cannot be produced on a line, are excluded ($a_{lk} > 0$ and $a_{li} > 0$ in the sigma signs; notice, $a_{lj} > 0$ can be omitted since product pairs (i,j) in the list of *possible pairs* have the same a_{lj}-functionality). Finally, the denominator of Equation (4.30) defines the number of summands in the enumerator and is used to calculate the average deviation. If the value of $dev^{av}_{(i,j)}$ is small, the products i and j are similar in their setup characteristics and should be aggregated in a setup family. Notice, changeovers from i to j and vice versa are also considered in Equations (4.30). This situation is motivated as following: not all setup times between products i and

j of a pair (i,j) of the list *possible pairs* have been defined as small setup times (remember phase 1 of Cluster-S and relaxation 2). Up to now, the lengths of the large setup times have not been analyzed in detail. I.e., there might be a wide range of setup times (starting from $> st^{small}$ up to the maximum setup time of a scenario) which have been defined as large. Thus, it is reasonable to incorporate them in the decision process whether two products should be aggregated in a setup family or not. For instance, the term $|st_{l11} - st_{l21}|$ will arise for the calculation of $dev^{av}_{(i,j)}$ for the products $i = 1$ and $j = 2$. The parameter st_{l11} is equal to 0 (see page 119) and parameter st_{l21} defines the setup time of a changeover from j to i. As one can see, smaller values of st_{l21}, i.e., small setup times between two products which will possibly be assigned to the same family, lead to smaller values of $dev^{av}_{(i,j)}$. Therefore, the intention of Equations (4.24) is additionally incorporated in Equations (4.30).

After sorting the list of *possible pairs*, create an empty list storing the products which are *already assigned* to setup families (see Line 6). Start iterating over all entries of the list *possible pairs* (see Line 7). In the first iteration (see Lines 8-13), both products i and j are not represented in the list *already assigned*, thus, verify if the deviation $dev^{av}_{(i,j)}$ between product i and j is lower than or equal to a given limit dev^{limit} which is defined as percentage of the maximum setup time (relaxation 3). If the deviation is greater than dev^{limit}, continue with the next pair (i,j). Otherwise, assign i and j to a new setup family ($g = 1$ in this case) and add these products to the list *already assigned*. Additionally, the variable $family\_updated$ is set to 1 which indicates that one or two products have been added to a family. The next step is to search for products which should also be added to the newly-created family. The applied procedure is described in the following (see Lines 14-24). At first, start a second forall-loop which iterates over all product pairs (m,n) of the list of *possible pairs*. If exactly one of the products, e.g., product m, of a pair (m,n) is part of the list of *already assigned* products, proceed with the following (see Lines 19-24). Calculate all deviation values $dev^{av}_{(n,k)}$ between product n and all products k which are included in family g. For example, the first setup family consists of the products 1 and 3 and the next pair is $(m,n) = (3,9)$. Obviously, product $m = 3$ is part of setup family $g = 1$. Thus, the following deviations $dev^{av}_{(n,k)}$ are calculated: $dev^{av}_{(9,1)}$ and $dev^{av}_{(9,3)}$. If all resulting deviations are smaller than or equal to the given limit dev^{limit}, product 9, or in general, product n, is assigned to setup family g.[24] Additionally, product n is added to the list of *already assigned* products and the variable $family\_updated$ is set to 1 to assure a further iteration of the while-loop (see Line 14). I.e., whenever an additional product is added to a family ($family\_update = 1$), it is required to repeat the procedure one more time, since the list *already assigned* has been updated and the if-question of Line 19 could switch to a “yes” for pairs which have led to a “no” before. If at least one deviation is higher than dev^{limit}, product 9 is not assigned to a family right now and the algorithm continues with the next iteration of the forall-(m,n)-loop. If there is no further product which can be added to this family, the main forall-(i,j)-loop continues (see Line 7). If both products of the next iteration are again not in the list of *already assigned* products and fulfill $dev^{av}_{(i,j)} \leq dev^{limit}$, create another setup family and repeat the previously described process.

[24]In some cases, it also might be helpful to check if all setup times which could occur between products of the family are small enough. Additionally, it might also be necessary to define a maximal number of products which can be assigned to a family. For our numerical tests, both adaptions have not been necessary.

Algorithm 2: Cluster-S phase 2 (main input: output of phase 1)

```
1  sort the list possible pairs by decreasing $small\_st^{perc}_{(i,j)}$ values;
2  forall $(i,j) \in$ possible pairs do // calculate the deviation between i and j:
3      $dev^{av}_{(i,j)} = \frac{\sum_{l,k,a_{lk}>0,a_{li}>0} |st_{lik}-st_{ljk}| + \sum_{l,k,a_{lk}>0,a_{li}>0} |st_{lki}-st_{lkj}|}{\sum_{l,k,a_{lk}>0,a_{li}>0} 2}$;
4  sort all $(i,j)$ in possible pairs which have identical $small\_st^{perc}_{(i,j)}$ values by increasing $dev^{av}_{(i,j)}$ values;
5  $g := 0$; // family counter
6  already assigned$= \{\}$; // list of products which are already assigned to a setup
     family
7  forall $(i,j) \in$ possible pairs do // assign products to families:
8      if i and j $\notin$ already assigned then
9          if $dev^{av}_{(i,j)} \leq dev^{limit}$ then
10             $g := g+1$;
11             assign i and j to family g;
12             add i and j to the list already assigned;
13             $family\_updated := 1$; // indicates that a family has been updated
14             while $family\_updated == 1$ do
15                 $family\_updated := 0$;
16                 forall $(m,n) \in$ possible pairs do
17                     if (m and n $\notin$ already assigned) or (m and n $\in$ already assigned) then
18                         continue with next iteration of forall-$(m,n)$-loop;
19                     else if m (respectively n) $\in$ already assigned then
20                         calculate $dev^{av}_{(n,k)}$ (respectively $dev^{av}_{(m,k)}$) $\forall k \in SF_g$;
21                         if all $dev^{av}_{(i,j)}$ values calculated in Line 20 $\leq dev^{limit}$ then
22                             assign n (respectively m) to family g;
23                             add n (respectively m) to the list already assigned;
24                             $family\_updated := 1$;
25     else if i and j $\in$ already assigned then
26         continue with next iteration of forall-loop;
27 assign all remaining products solely to newly-created families;
```

If in an iteration (i,j) both products are in the list of *already assigned* products, the algorithm directly starts the next iteration (see Lines 25 and 26). For instance, the families $(1,3)$ and $(6,8)$ already exist and the next pair is $(3,8)$. Obviously, both products are already in setup families, consequently, skip to the next iteration. Since the iteration order is based on a ranking, it has been reasonable to assign the products 3 and 8 to two different setup families and not to merge them into one as proposed by the pair $(3,8)$.

After iterating over all pairs of the list of *possible pairs*, there might be products which have not been assigned to setup families so far. Either, because they are not in the list of *possible products*, or the deviations $dev^{av}_{(i,j)}$ are not small enough. Each of these remaining products is solely assigned to a newly-created setup family (see Line 27).

To sum up, Cluster-S is capable of handling sequence-dependent setup times and products which are forbidden on some production lines. The outcome of the algorithm are setup families which have low setup times between products of the same family. Additionally, the products of a family are as homogenous as possible in the characteristic of occurring setup times to and from products of other families. The approach does not allow to define the number of setup families in advance. However, the number of setup families is a result of the algorithm. Nevertheless, it seems simple to create an iteration process which adapts the boundaries, such as st^{small} or dev^{limit}, to create a higher or lower number of setup families.

4.4.3.3 Cluster-R

The Cluster-R algorithm assigns products to setup families based on random numbers. The basic assumptions are the same as for Cluster-S: initial inventory is zero, only products which show net demand in at least one period are included and product $j = 0$ is assigned to family $g = 0$. To make the resulting setup families comparable to the first algorithm, it is necessary to define the desired number of setup families $G^{des}(\leq J)$ in advance.

As discussed earlier, it is a crucial problem to define production coefficients if products with different a_{lj}-functionalities would be assigned to the same setup family. Thus, the first step (see Algorithm 3 Line 1) is to assign products which show the same a_{lj}-functionality to the same family. If there is a single product showing a certain a_{lj}-functionality, this product is solely assigned to a family.

If the resulting number of families (see the paragraph before) is higher than the desired number of families ($G > G^{des}$), the algorithm is stopped. The reason is, it is impossible to reach the desired number of families without violating the basic request of only grouping products together which show the same a_{lj}-functionality. Nevertheless, since Cluster-S respects the same request, Cluster-S will not define a smaller number of setup families than Cluster-R. I.e., both algorithms will always be comparable.

Algorithm 3: Cluster-R (main input: a_{lj}, G^{des})

```
1 assign products $j$ which show the same $a_{lj}$-functionality to the same family $g$;
2 // increase the number of families $G$ to the desired number $G^{des}$:
3 while $G < G^{des}$ do
4     randomly choose a family $g$ which consists of more than one product;
5     randomly choose a number $n$ (between 1 and $|SF_g| - 1$) which defines how many products of family
        $g$ will be reassigned to a new family;
6     randomly choose $n$ different products of family $g$ and reassign them to a newly-created family;
7     $G := G + 1$;
```

If there are less families compared to the desired number of families, new families must be created and some products of the existing families must be reassigned to the new families (see Lines 3-7). The reassignment process works as following. Randomly choose an existing setup family g which consists of more than one product (splitting a setup family which consists of a single product does not work).

Then, randomly choose a number n which defines how many products of family g will be reassigned to a new family. Notice, only values between 1 and the number of products of family g reduced by 1 make sense, since at least one product must be reassigned to a new family and at least one product must stay in family g. Afterwards, n different products of setup family g are randomly chosen and reassigned to a newly-created setup family. If the number of setup families is still smaller than the desired number of setup families, the aforementioned process is repeated until the desired number G^{des} is reached.

4.4.4 Disaggregation

Linking the GLSPPL model formulation (4.1)-(4.7) with aggregated input data (see Sections 4.4.2 and 4.4.3) leads to the aggregated model $PL/Aggr$. In our case, this model is solved using a heuristic, known from Meyr and Mann (2013) (for details see Section 4.5). The resulting solution defines production quantities x^g_{lgs} of family g on line l in microperiod s. These quantities must be disaggregated to determine line- and product-specific demand parameters d_{ljt} of each period t which will be used to form independent single-line problems SL_l. If a family is exclusively scheduled on one line within the aggregated plan, disaggregation can be done applying a backward oriented heuristic. However, if a family is assigned to several lines, the problem is more complex due to different production costs[25]. In this case, it is useful to formulate an assignment problem (inspired by Meyr 1999, pp. 178f, but essentially adapted). Since it happens quite often that a family is assigned to several lines, we decided to solve the following optimization model for each setup family $g = 1, 2, ..., G$, no matter if the family is produced on one or several lines.

The variable x_{lujt} reports the assignment of quantities of family g which are produced on line l in macroperiod u to the demand of product j ($j \in SF_g$) in macroperiod t. Correspondingly, c_{lujt} defines the costs for producing one unit of product j on line l in macroperiod u and store it until the demand arises in macroperiod t. As shown in Equations (4.31), these costs consist of the production costs of product j on line l plus the holding costs which occur for storing one unit of product j from macroperiod u until macroperiod t.

$$c_{lujt} = c_{lj} + (t-u) * h_j \quad \forall l, u, j, t \tag{4.31}$$

If a setup family is scheduled on several lines, it might happen that one or several products of this family are assigned to several lines as well. However, assigning products to several lines increases the number of products which must be simultaneously considered in each single-line problem SL_l. Additionally, if a product is produced on several lines, it must be set up on several lines which causes

[25]Example: assume two production lines, two macroperiods and one setup family consisting of two products. Production costs of product 2 are identical on both lines, but, production of product 1 is cheaper on line 1 than on line 2. Furthermore, holding costs of product 1 are more expensive compared to the holding costs of product 2. Setup times, setup costs and minimum lotsizes are neglected. The aggregated production plan schedules setup family 1 on line 1 in macroperiod 1 and on line 2 in macroperiod 2. Both production slots must be completely used to fulfill the identical demand of both products at the end of macroperiod 2. Looking at the production costs, it is preferable to produce product 1 on line 1. However, looking at the holding costs, it is preferable to produce product 1 as late as possible which would mean to produce it on line 2. Obviously, the problem is not trivial.

higher setup times and costs. Normally, the costs c_{lujt} control the assignment of products to lines. I.e., if possible, a product will be produced exclusively on the cheapest line. However, it is possible that a constellation of production- and holding-costs arises which unnecessarily assigns a product to several lines. For example, if the costs c_{lujt} of product j are identical for all lines, no incentive exists to assign product j exclusively to a single line. Thus, a binary variable lu_{lj} which indicates whether product j is assigned to line l at least once within the planning horizon ($lu_{lj} = 1$) or not ($lu_{lj} = 0$) is introduced and connected to penalty costs cl.

The penalty costs cl arise if a product is produced on a line at least once within the planning horizon. The costs for line use cl are set to a high value, e.g., to the maximum of all cost parameters ($cl = \max_{l,i,j}\{s_{lij}, c_{lj}, h_j\}$).

To assure solvability, the heuristic applied to solve the aggregated problem $PL/Aggr$ (step 3 of Fig. 4.1) allows unlimited supply of all products, i.e., lost sales are possible, but charged with high penalty costs. In this way, the total production quantity in real periods ($s = 1,...,S$) of the aggregated plan ($\sum_{l,g,s} x^g_{lgs}$) does not always represent the necessary quantity to fulfill the total demand ($\sum_{j,t} d_{jt}$) of the original problem PL/O. Therefore, it is possible to extend the provided production quantities x^g_{lgs} of setup family g. The variables xa_{lt} define the additionally used quantities on line l in macroperiod t (notice, the measuring unit of xa_{lt} is quantity units of family g).

It also seems reasonable to use additional quantities xa_{lt} to reduce the number of lines employed to produce a product. The following example explains this in more detail. If the assignment model presented later on assigns 30 units of product 1 to line 1 and only 2 units to line 2, using the provided capacities x^g_{lgs}, it seems reasonable to add two additional units ($\sum_t xa_{1t} = 2$) to the provided capacity of line 1 in order to schedule product 1 exclusively on line 1.

The costs ca_{lt} for using one additional unit of family g on line l in macroperiod t are defined based on the following considerations. The costs ca_{lt} should be low if there is idle production time e^g_{lt} (time units) on line l in macroperiod t, i.e., if there is time which is neither used for production nor for setups within the aggregated plan. On the other hand, if there is no idle time on line l in macroperiod t, the costs should be high. This idle time e^g_{lt} can be calculated as shown in Equations (4.32).

$$e^g_{lt} = K_{lt} - \sum_{g,s\in S^g_t} x^g_{lgs} a^g_{lg} - \sum_{g,h,s\in S^g_t} st^g_{lgh} z^g_{lghs} \quad \forall l,t \tag{4.32}$$

Equations (4.32) subtract the used production time on line l in period t of model $PL/Aggr$ and the associated setup times from the given capacity K_{lt}.

However, it is only reasonable to use this idle production time if it is necessary due to lost sales within the aggregated plan or if costs for line use cl can be saved. Otherwise, the reserved time in terms of x^g_{lgs} should be used. Nevertheless, the additional quantities xa_{lt} cannot be limited to the corresponding idle production time e^g_{lt} since solvability must be assured. Notice, even in the case of zero idle time and positive additional quantities, it is possible to find feasible plans later on in the single-line problems SL_l (step 5 of Fig. 4.1). The reason is that the reserved setup and production time of a setup family's lot might allow producing more units of a product due to aggregation errors concerning setup times and production coefficients.

Using idle time should be cheaper than using an additional line for a certain product. I.e., in the extreme case, all idle time should be used to avoid producing a product on an additional line. The basic approach to determine the costs for using one additional unit ca_{lt} is to determine how many quantity units of family g can be produced within the idle time e^g_{lt} and divide the costs for line use cl by this value. For instance, if it is possible to produce 20 units within the idle time and the costs for line use are $cl = 500$ monetary units, the costs for using one additional unit ($xa_{lt} = 1$) are $ca_{lt} = 500/20 = 25$ monetary units. In this case, using 1 to 20 additional units is cheaper or equally expensive (costs: 25 to 500) compared to producing the product on one additional line. As desired, 21 units will never be used to avoid producing the product on an additional line, since the costs of $21 * 25 = 525$ monetary units are higher than the line use costs. The following paragraphs describe the determination of ca_{lt} in detail.

The first step is to calculate the number of quantity units q_{lt} of family g which can be produced during idle time on line l in macroperiod t.

$$q_{lt} = \frac{\sum_u d^g_{gu}}{\sum\limits_{h \geq g, u : a^g_{lh} > 0} d^g_{hu}} e^g_{lt} \frac{1}{a^g_{lg}} \quad \forall l,t : a^g_{lg} > 0 \tag{4.33}$$

Equations (4.33) weight the idle capacities e^g_{lt} (time units) and transfer the resulting values to quantity units of family g. The weighting is calculated by dividing the demand of family g which maximally could occur on line l by the total demand of all families which could occur on line l. The total demand of all families which could occur depends on the families which must still be disaggregated. Remember, the described assignment problem is solved consecutively for each family $g = 1,2,...,G$, thus, if the demand of a family is already assigned to single-line problems, it is not useful to still reserve idle time for this family. As shown in Equations (4.34), q_{lt} is set to zero if family g cannot be produced on a line.

$$q_{lt} = 0 \quad \forall l,t : a^g_{lg} = -1 \tag{4.34}$$

Finally, the costs for line use cl are divided by the number of quantity units of family g which can be produced during idle time on line l in macroperiod t (see Equations (4.35)). As one can see, using idle time to save line use costs should only be done if there is enough idle capacity to produce at least one unit. In all other cases, the costs ca_{lt} are set to line use costs times five (see Equations (4.36)), since not further documented tests runs have shown that this value seems to be reasonable. Table 4.3 summarizes the symbols used in the following model.

$$ca_{lt} = \frac{cl}{q_{lt}} \quad \forall l,t : q_{lt} \geq 1 \tag{4.35}$$

$$ca_{lt} = 5 * cl \quad \forall l,t : q_{lt} < 1 \tag{4.36}$$

Table 4.3: Symbols for disaggregation model of setup family g

Indices:	
$u = 1, ..., U$	macroperiods
Data:	
c_{lujt}	costs for producing one unit of product j on line l in macroperiod u and store it until macroperiod t
ca_{lt}	costs for using one additional unit of setup family g to extend the provided quantities on line l in macroperiod t
cl	line use costs which occur once in the planning horizon if the production quantity of a product on a line is positive
d_{jt}	demand of product j in macroperiod t (known from the original problem PL/O)
x^{g}_{lgs}	production quantity of family g on line l in microperiod s (known from the aggregated production plan)
Variables:	
$x_{lujt} \geq 0$	quantities of setup family g which are scheduled on line l in macroperiod u and assigned to the demand of product j ($j \in SF_g$) in macroperiod t
$xa_{lt} \geq 0$	number of additionally used quantity units of setup family g on line l in macroperiod t
$lu_{lj} \in \{0;1\}$	equals 1 if product j is produced on line l at least once in the planning horizon (0 otherwise)

Objective function:

$$\text{Min} \sum_{l,u,t,j \in SF_g} c_{lujt} x_{lujt} + \sum_{l,t} ca_{lt} xa_{lt} + \sum_{l,j \in SF_g} cl * lu_{lj} \tag{4.37}$$

Constraints:

$$\sum_{j \in SF_g} \sum_{t=u}^{T} x_{lujt} \leq \sum_{s \in S_u} x^{g}_{lgs} + xa_{lu} \qquad \forall l, u : a_{lg} > 0 \tag{4.38}$$

$$\sum_{l : a^{g}_{lg} > 0} \sum_{u=1}^{t} x_{lujt} = d_{jt} \qquad \forall t, j \in SF_g \tag{4.39}$$

$$\sum_{u,t} x_{lujt} \leq lu_{lj} \sum_{t} d_{jt} \qquad \forall l, j \in SF_g : a_{lg} > 0 \tag{4.40}$$

The objective function (4.37) minimizes the total costs. These costs consist of the costs for assigning quantities of family g to product-dependent demand, the costs for additional quantities and the costs for line use.

Constraints (4.38) assure that the quantities of family g which are assigned to product-dependent demand x_{lujt} do not exceed the provided production quantities of setup family g plus the additional used quantities xa_{lt}. The given product-dependent demand d_{jt} of each product $j \in SF_g$ in each period t must exactly be fulfilled by the assigned quantities x_{lujt} (see Equations (4.39)). Notice, $a_{lg} > 0$ in the sigma sign assures that products are only assigned to lines which are capable of producing these products. Inequalities (4.40) assure that if product j is assigned to line l, the line use indicator lu_{lj} is set to 1. After solving the model for family g and before solving it for family $g+1$, the idle times used in Equations (4.33) should be adapted in the following way: $e^g_{lt} := e^g_{lt} - xa_{lt} \quad \forall l,t$.

The resulting quantities x_{lujt} can easily be transferred to line-dependent, product-specific demand parameters d_{ljt}, like it is formalized in Equations (4.41).

$$d_{ljt} = \sum_{u=1}^{t} x_{lujt} \quad \forall l,j,t \tag{4.41}$$

After solving the model (4.37)-(4.40), the resulting demand d_{ljt} can be combined with the GLSP model (4.1)-(4.7) to form single-line problems SL_l. It should be mentioned that the model size of the single-line problems can be reduced further by adapting the number of microperiods per macroperiod $|S_t|$ to the number of products which will be produced on this line. Furthermore, all variables and parameters must only be defined for the relevant products and product $j = 0$ for each line. By doing so, one special case may arise. If the initial product of a line is not product $j = 0$, this is no problem as long as demand of the initial product is assigned to this line. If, e.g., product 1 is the initial product of line 1, but only demand for products 2 and 3 occurs on this line, necessary parameters are missing. For example, the setup times from product 1 to 2 or 1 to 3 are not incorporated in the provided data. To avoid further implementation difficulties, the initial product is named as product 0 and all necessary parameters of the original initial product are transferred to product 0. I.e., the new setup times concerning the initial product of a line l which shows the aforementioned characteristic are defined as $st_{10j} := st_{1aj}\ \forall j$ with a representing the original initial state, i.e., in the aforementioned example $a = 1$.

4.4.5 Iterative solution

After solving the single line problems SL_l, the resulting setup patterns are used to formulate a linear programming representation of the original model (step 6 of Fig. 4.1). The generated model can be solved using a standard solver (in our case, Gurobi Optimizer). If no feasible solution exists, but the model ($PL/Aggr$) has been feasibly solved (i.e., no lost sales), the feasibility-problems are induced by aggregation errors. I.e., the time spans reserved for setup families do not match the time which is necessary for production and setup of the single products of the families. Thus, it is worthwhile to reserve some production capacity which cannot be used in the aggregated model $PL/Aggr$, but is available in the single-line problems SL_l (see step 7 of Fig. 4.1). This approach hopefully leads to different aggregated production plans and consequently to different single-line demand values compared to the previous iteration. The reduced capacities are calculated using a slightly adapted[26] version of the algorithm proposed in Meyr and Mann (2013, pp. 722f).

[26]Meyr and Mann (2013) reserve both production time and setup time for the missing quantities. We omit the consideration of setup times, since it often leads to a too strong reduction of capacity in the numerical tests.

Algorithm 4: Calculation of adapted capacity (main input: d_{ljt}, x_{ljs}, original K_{lt})

```
forall $l, j, t = 1, ..., T$ do // forward calculation of shortages:
    $I_{ljt} := I_{lj,t-1} + \sum_{s \in S_t} x_{ljs} - d_{ljt}$;
    if $I_{ljt} < 0$ then
        $SQ_{ljt} := -I_{ljt}$; // short quantity
        $I_{ljt} := 0$;
    else
        $SQ_{ljt} := 0$;
forall $l$ do
    $CR := 0$; // capacity reduction
    forall $t = T, ..., 1$ do // backward calculation of reduced capacities:
        $CR := CR + \sum_j a_{lj} SQ_{ljt}$;
        // capacity reduction:
        if $CR > K_{lt}$ then
            $CR := CR - K_{lt}$;
            $K_{lt} := 0$;
        else
            $K_{lt} := K_{lt} - CR$;
            $CR := 0$;
```

Lines 1-7 of Algorithm 4 show the calculation of the shortage quantities SQ_{ljt} of product j on line l in macroperiod t. Afterwards, a backward calculation (see Lines 8-18) is used to determine the reduced capacities K_{lt} which will be used in $PL/Aggr$ in the next iteration.

4.5 Numerical tests

In the following, the performances of the new heuristics Aggr-P (using Cluster-S) and Aggr-PR (using Cluster-R) are tested. The heuristics are used to solve two large-scaled practical problems. The decomposition heuristic TA-agg of Meyr and Mann (2013) (for a short description see Section 4.2) serves as benchmark. Section 4.5.1 describes the basic settings of the test environment. Section 4.5.2 introduces the test instances and discusses the results of the experiments.

4.5.1 Basic settings of the test environment

All tests haven been executed using a laptop showing the following technical specifications: intel Core i7-6500U 2.5 GHz DC CPU, 8 GB RAM, Win 10 Pro 64 bit. The new heuristics have been implemented using Python 2.7.12. Furthermore, Gurobi Optimizer has been used to solve the disaggregation problem (4.37)-(4.40) and the linear programming representation of PL/O (see step 6 in Figure 4.1). Depending on the experiment, the aggregated problem $PL/Aggr$ (step 3 of Fig. 4.1) has been solved using TA-GLPK or TA-agg (see Section 4.2). To avoid re-implementation of these heuristics, the original implementations of Meyr and Mann (2013) have been embedded as exe-files. Additionally, TADR (see Section 4.2) has been used to solve the single-line problems SL_l (step 5 of Fig. 4.1).

For the threshold accepting concepts of TADR and TA-GLPK, the threshold values of Meyr (2000) and Meyr (2002) are used, respectively. Furthermore, the threshold multiplier *TM* is set to 100 for TADR and to 2000 for TA-GLPK. The threshold multiplier TM is used to control the threshold accepting process. In detail, after TM iterations without any improvement of the objective function value, the next lower threshold value is applied. After $5 * TM$ iterations without any change, the complete run is finished. Finally, $2 * TM$ defines the maximal number of iterations executed for each threshold value. Notice, these parameters are applied for TADR and TA-GLPK in every experiment, no matter in which heuristic they are used.

Each time TADR is used in an iteration of a heuristic it is repeated 10 times using different seed values for generating the random numbers which are applied during the threshold accepting procedures. Afterwards, the best result of these 10 runs is chosen.

TA-agg provides further setting options: we choose the version TA-agg1-s for the benchmark. In this version, the aggregation factor is set to 1 which means that the problem is not aggregated at all. However, the problem is decomposed into single-line problems. The “s” in TA-agg1-s indicates that the number of microperiods per macroperiod are reduced if reasonable (for details see Meyr and Mann 2013, p. 729). If TA-agg is used in Aggr-P or Aggr-PR, an aggregation factor of 4 is applied, i.e., the aggregated model comprises $\frac{T}{4}$ macroperiods (T is the number of macroperiods in the original model). The number of microperiods per macroperiod is not decreased. In the following, this heuristic will be called TA-agg4.

If the result of the first iteration of TA-agg1-s, Aggr-P or Aggr-PR is infeasible, the capacities of the aggregated problem are reduced based on the missing quantities (see algorithm 1 of Meyr and Mann (2013) and Algorithm 4 of the present paper) and a second iteration is started. However, the heuristics are stopped after completion of the second iterations, no matter if feasible solutions have been found or not.

Notice, we choose TA-agg1-s combined with the above mentioned threshold multipliers, because it performed very well in Meyr and Mann (2013), especially for the problem instances considered in the following.

4.5.2 Test instances and results

The first test instance will be called *J51* since it comprises $J = 51$ products (notice, only 50 products show net demand). This instance has been introduced by Meyr (1999, pp. 180f). It is a practical problem occurring in the acrylic glass production and was formulated to execute a mid-term planning. It consists of $L = 3$ lines and $T = 12$ macroperiods and has several characteristics which make it difficult to solve: e.g., minimum lotsizes must be respected for each product and not every product can be produced on all lines as well as sequence-dependent setup times and costs exist.

The second problem instance comprises $J = 72$ setup families and will be called *J72*. It is provided by a label printing company. The labels are used to declare consumer goods. The company already did a basic assignment of products to setup families. However, the high number of 72 setup families suggests that potential for further aggregation exists. Identically to instance *J51*, the problem comprises $T = 12$ macroperiods and is applied for mid-term planning. Furthermore, $L = 7$ lines must be considered. Due to the high number of lines and the presumption that the necessary setups will be spread over all lines, the number of microperiods per macroperiod $|S_t|$ is set to 20. Identically to problem *J51*, not every product can be produced on all lines and minimum lotsizes as well as sequence-dependent setup times and costs exist.

Both problems are solved using TA-agg1-s, Aggr-P(TA-GLPK), Aggr-P(TA-agg4), Aggr-PR(TA-GLPK) and Aggr-PR(TA-agg4). The information in brackets defines the heuristic applied to solve the aggregated multi-line problem $PL/Aggr$. Since the threshold accepting concepts of TADR and TA-GLPK use random numbers, all combinations of instances and heuristics are performed at least 60 times. Due to different seed values of the random number generators, these runs will lead to different results.

We decided to do the aggregation in a way that the number of setup families is approximately one third of the number of products. In detail, the number of setup families of instance *J51* is 19 and the number of setup families of instance *J72* is 25. The following parameters have been used to fulfill the desired number of setup families: for *J51*: $st^{small} = 0.1$, $st^{large} = 0.2$, $st^{max\_diff} = 0.1$, $small_{st}^{perc\_allowed} = 1$ and $dev^{limit} = 0.002$. For *J72*: $st^{small} = 0.5$, $st^{large} = 0.6$, $st^{max\_diff} = 0.1$, $small_{st}^{perc\_allowed} = 0.4$ and $dev^{limit} = 0.1$. For comparability reasons, Aggr-PR uses the same number of setup families as Aggr-P. Notice, Aggr-PR randomly generates new setup families for each run since the associated seed value of the random number generator is changed as well (see Algorithm 3).

Table 4.4 summarizes the results. There, one can find the average percentage deviation from the best known objective function value[27] ("av. obj"). In the following, the aforementioned performance indicator will be called "objective deviation". Furthermore, the percentage of runs with lost sales is documented in italic letters. Additionally, the average computing times (excluding the solutions with lost sales) in seconds ("av. cpu") can be found in the table. For each problem instance, the best values of objective deviation, computing time and percentage of lost sales are marked in bold letters.

Table 4.4: Results of solving the instances *J51* and *J72* with different heuristics. "Av. obj" defines the average percentage deviations from the best known objective values. "Av. cpu" defines the average computing time. Percentages of runs with lost sales are documented in italic letters.

		TA-agg1-s		Aggr-P (TA-GLPK)		Aggr-P (TA-agg4)		Aggr-PR (TA-GLPK)		Aggr-PR (TA-agg4)	
J51	av. obj [%] *ls* [%]	4.0	*50*	2.5	*65*	**2.4**	***32***	4.1	*55*	4.3	*63*
	av. cpu [s]	432		257		**119**		232		130	
J72	av. obj [%] *ls* [%]	0.7	***0***	**0.5**	*32*	0.9	*12*	0.6	*78*	0.8	*40*
	av. cpu [s]	1426		**549**		561		794		753	

[27]Best objective function value known from Meyr and Mann (2013), updated if better solutions have been found.

This paragraph discusses the results of the test instance *J51*: as one can see in Table 4.4, Aggr-P(TA-agg4) performs best. Especially in comparison to TA-agg1-s and both Aggr-PR heuristics, it shows a significantly lower average objective deviation. Additionally, Aggr-P(TA-agg4) is approximately 3.5 times faster compared to TA-agg1-s. Comparing Aggr-P with Aggr-PR, in terms of average objective deviations, Aggr-P performs better. Nevertheless, comparing the percentage of lost sales, Aggr-PR performs not that bad compared to the other heuristics. The reason might be that the assignment of products to setup families is changed for each run. The new heuristics which use TA-agg4 are faster compared to the ones which apply TA-GLPK. Additionally, there is no relevant difference between the computing times of Aggr-P and Aggr-PR while applying the same heuristic to solve the aggregated problem.

In the following paragraph, the results of test instance *J72* are discussed: in terms of the average objective deviation, all heuristics perform very well (average objective deviations are less than 1%). However, Aggr-P(TA-GLPK) performs best and Aggr-P(TA-agg4) performs worst. Both Aggr-PR heuristics show the highest lost sales and TA-agg1-s shows no lost sales at all. However, TA-agg1-s is quite slow, taking up 1426 seconds on average. Compared to instance *J51*, the difference is that Aggr-P and Aggr-PR do not show similar computing times if the same sub-heuristic (e.g. TA-GLPK) is applied. In detail, Aggr-PR needs approximately 200-250 seconds longer compared to the corresponding Aggr-P heuristic.

Since it is difficult to get a complete picture by analyzing three different performance parameters like it was done in Table 4.4, it seems worthwhile to additionally use another way of result representation (see Meyr and Mann 2013, pp. 726f). The basic approach is to set solution quality and computing time in relation. Obviously, during the time necessary to solve *J72* using TA-agg1-s, it is possible to run Aggr-P(TA-agg4) twice and choose the smaller resulting objective function value. I.e., if a heuristic with stochastic components is executed twice, the two resulting running times (including running times of results which show lost sales) have to be summed and the best run concerning the objective function value has to be chosen. In our approach, we will do this type of calculation for different sizes of "packages" $(1, 2, 3, ...)$. Afterwards, the computing times and the objective deviations of all packages of a package size will be averaged.

The results of the (at least) 60 runs of, e.g., instance $J51$ solved by Aggr-P(TA-GLPK), are stored as a list. Since they are independent, each run has the same probability to occur. Thus, each combination of runs should be considered. I.e., if the package size is 2, it would be the best to calculate the objective deviations and computing times for all possible combinations of two runs of this list of 60 runs. However, a package size of 6 will already lead to $60^6 = 4.7 * 10^{10}$ combinations. Thus, to keep the handling simple, the packages are merely built by iterating subsequently over the given list of 60 runs. I.e., if the package size is 2, the first two runs, run 3 and 4, run 5 and 6 and so on are combined. Lost sales are considered in more detail as described in the following. If all runs of a package show lost sales, this package will be marked as "lost sales package", otherwise the package will be marked as "feasible package". Now, the average running times of each package size are multiplied using the following factor $\frac{number\ of\ all\ packages}{number\ of\ feasible\ packages}$ (this calculation is only possible if there is at least one feasible package). This factor defines the average number of runs of a certain package size which must be performed to find a feasible solution. Using the aforementioned factor, the average running time to find a feasible solution for each package size is estimated.

Figure 4.3 shows the results of instance *J51*. The complete running time is limited to 7200 seconds (i.e., the package size is increased until an average computing time of 7200 seconds is reached) and only the convex hull of each heuristic is plotted.

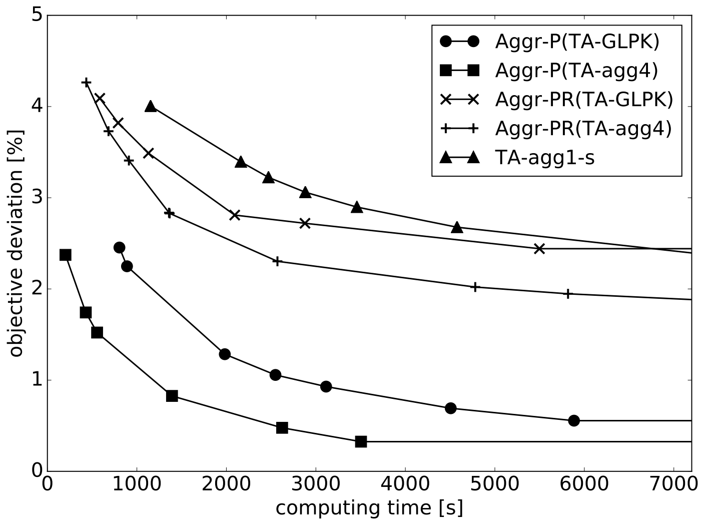


Figure 4.3: Objective deviation depending on computing time for instance *J51*

In Figure 4.3, one can see that both Aggr-P heuristics perform very well. As expected, due to the randomly created setup families, both Aggr-PR heuristics perform not that well. However, up to the computing time of nearly two hours, both are preferable compared to TA-agg1-s. Additionally, one can see that the time to find a first feasible solution is quite short for Aggr-P(TA-agg4) and nearly identical for both Aggr-PR heuristics. Aggr-P(TA-GLPK) needs approximately 805 seconds and TA-agg1-s need approximately 1150 seconds.

A similar figure is created to represent the results of instance *J72*. In this case, the running time was extended to three hours. The results can be found in Figure 4.4.

Figure 4.4 shows that Aggr-P(TA-GLPK) performs best. However, Aggr-P(TA-agg4) finds the first feasible solution already within 677 seconds (Aggr-P(TA-GLPK) needs 940 seconds). This difference is caused by a higher percentage of lost sales of Aggr-P(TA-GLPK) (see Table 4.4). TA-agg1-s needs 1426 seconds to find a first feasible solution despite zero lost sales (see Table 4.4). However, from finding its first feasible solution up to approximately 3850 seconds, TA-agg1-s performs better compared to Aggr-P(TA-agg4). Aggr-PR(TA-GLPK) needs approximately 4300 seconds to find a first feasible solution. Whereas, Aggr-PR(TA-agg4) does so in a much shorter time. This must be due to the higher percentage of lost sales of Aggr-PR(TA-GLPK) (see Table 4.4).

A further test was initiated using the test instance *J51* and the heuristic Aggr-P(TA-agg4) which was the best heuristic to solve this problem. Now, the setting was changed in a way that the 51 products

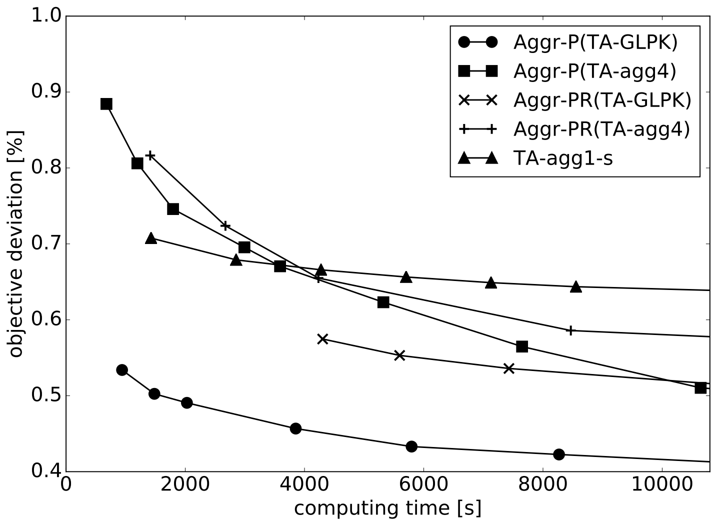


Figure 4.4: Objective deviation depending on computing time for instance *J72*

have been condensed to 5, 14 and 37 families[28]. The results are shown in Figure 4.5 up to a computing time of 3300 seconds. For comparison reasons the result of the experiment with 19 families and Aggr-P(TA-agg4) is plotted as well.

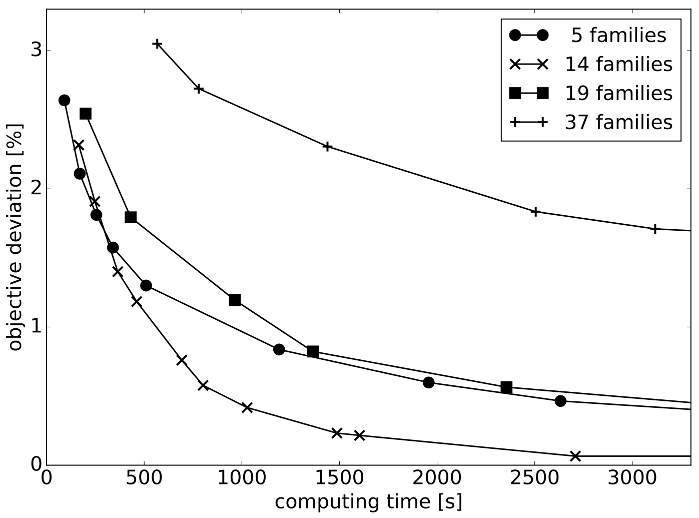


Figure 4.5: Objective deviation depending on computing time for instance *J51* solved by Aggr-P(TA-agg4) applying different numbers of setup families

As Figure 4.5 shows, a high number of setup families leads to bad results. The reason for this is the higher computing time to solve an aggregated problem with 37 families compared to an aggregated

[28]Applied settings defined in the order st^{small}, st^{large}, st^{max_diff}, $small_{st}^{perc\_allowed}$, dev^{limit}: for 5 families: $0.99, 0.999, 0.1, 0, 0.99$. For 14 families: $0.1, 0.2, 0.1, 1, 0.003$. For 37 families: $0.1, 0.2, 0.1, 1, 0.001$.

model with e.g., 19 families. As expected, an aggregated problem of bigger size is more difficult to solve. The experiments with 5 and 14 families show better results than the experiment with 19 families. However, using the lowest possible number of setup families also shows disadvantages. As one can see, the experiment with 5 families never reaches objective deviations which are as small as for the experiment with 14 families. One reason might be that too much information has been lost during the aggregation of parameters. However, in the scenario with 5 families only 6.7% of the runs show lost sales. In the case with 14 families, 30% of the runs have lost sales. I.e., the reason is not an aggregation error which occurs due to an underestimation of necessary production capacity in the aggregated case. Most probably, the reason of worse objective function values is that the line assignment which is done during the aggregation-disaggregation process does not allow finding better production plans for the single-line problems. The following example explains this case. Assume that in the optimal solution of a multi-line problem product 1 has to be produced on line 1. If this product is assigned to line 2 during the aggregation-disaggregation process, the optimal solution of the original problem never can be found during the further steps of the decomposition heuristic. However, Figure 4.5 shows that Aggr-P provides very good results even for a relatively wide range of numbers of setup families.

4.6 Summary and outlook

A new solution approach for large-scaled GLSPPL problem instances has been introduced. The approach is based on decomposition and works as follows: by aggregating products to setup families, the original multi-line problem is transformed into a smaller problem. The problem consisting of setup families is solved using the heuristics TA-GLPK or TA-agg4 (both known from Meyr and Mann 2013). The resulting production plan is used to form single-line problems which consider products. These single-line problems are subsequently solved by the heuristic TADR (see Meyr 2000) within a few seconds. Afterwards, the setup patterns of the single-line production plans are used to formulate a linear program of the original problem. Since all variables are continuous, the problem is solved by a standard solver within a very short time. If the final solution includes lost sales, the capacities of the aggregated model are reduced and a second iteration of the algorithm is started.

One important step of the new heuristic is the assignment of products to setup families. Therefore, we proposed two different heuristics. The first method, Cluster-S, defines setup families based on setup characteristics of the test instances. The approach is capable of handling sequence-dependent setup times which can be line-dependent as well. Different settings of this algorithm lead to more, but homogenous, or less, but heterogeneous setup families. At the same time, Cluster-S is also capable of handling cases with products which cannot be produced on all lines. As a benchmark, the algorithm Cluster-R is proposed which only considers the fact that not every product can be produced on all lines. The further assignment of products to families is done randomly. Cluster-R allows to define the number of desired setup families in advance which enables a fair comparison between heuristics using Cluster-R and heuristics using Cluster-S. For Cluster-S and Cluster-R, we defined identical formulas how parameters, like production coefficients, of the families should be calculated after assigning products to families.

For the disaggregation process (defining line- and product-dependent demands based on the aggregated plan), we formulated a mixed integer programming model. One important part of this model are the "line use costs". I.e., costs are accounted each time a product j is produced at least once during the planning horizon on a production line. These penalty costs assure that a product is not assigned to a unnecessarily high number of lines which could happen if a setup family is produced on several lines. Therefore, this approach saves a high amount of setup costs and setup times in the single-line problems SL_l compared to a disaggregation which does not avoid unnecessary assignment of a product to several lines.

Four different heuristics have been defined: Aggr-P(TA-GLPK), Aggr-P(TA-agg4), Aggr-PR(TA-GLPK) and Aggr-PR(TA-agg4). Both Aggr-P heuristics use Cluster-S to define the setup families and both Aggr-PR heuristics use Cluster-R. The heuristics named in brackets are used to solve the aggregated problems. Two practical applications showing 51 and 72 products have been solved in the numerical tests. Since the heuristic TA-agg1-s has performed very well for both test instances (*J51* and *J72*) in Meyr and Mann (2013), it serves as a benchmark. Notice, the instance with 72 products originally consists of even more products but has already been aggregated to these 72 products by the company.

For the numerical tests, the instance with 51 products has been condensed to 19 families and the instance with 72 products has been condensed to 25 families. For both problem instances Aggr-P(TA-GLPK) is superior to TA-agg1-s in terms of objective deviation and computing time. For the smaller instance, Aggr-P(TA-agg4) is even better than Aggr-P(TA-GLPK). In the case of 72 products, Aggr-P(TA-agg4) performs similar to TA-agg1-s. However, it is able to find a first feasible solution earlier and finds better solutions if the running time of both heuristics is longer than approximately 3850 seconds.

As expected, both Aggr-PR heuristics perform worse compared to the Aggr-P heuristics. However, in case of the instance with 51 products, both show better results compared to TA-agg1-s. In the case of 72 products, the objective deviations are also quite good with less than 1%. However, in this case, Aggr-PR(TA-GLPK) takes a very long time to find a first feasible solution due to a high number of solutions showing lost sales.

A further experiment tests Aggr-P(TA-agg4) with different numbers of families. The instance with 51 products is considered and the results show that a stronger aggregation, i.e., less setup families, leads to better results. The reason is that a smaller aggregated problem can be solved in a better way concerning solution time and objective function value. However, an experiment with 5 families shows that a number of families which is too low leads to disadvantages. In this case, the objective deviations are never as small as for the scenario with 14 families. However, the percentage of lost sales is much lower in the case of 5 families. The reason might be that the aggregation-disaggregation process creates single-line problems which do not allow to find better production plans for the original problem. However, the result is still very good and better compared to TA-agg1-s.

All in all, it has been shown that the new heuristics Aggr-P(TA-GLPK) and Aggr-P(TA-agg4) are very good at solving the considered large-scaled simultaneous lotsizing and scheduling problems. Aggr-PR is a bit worse. However, this difference between Aggr-P and Aggr-PR shows that Cluster-S works very well and provides advantages.

Further research should focus on the refinement and a deeper analysis of the approaches. Therefore, it will be useful to examine additional or adapted test instances. Especially by adapting the current test instances, it will be possible to find out why the solution heuristics perform so differently for instance $J51$ and $J72$. As one can see in Section 4.5.2, all tested heuristics perform similarly well in terms of the objective deviation for instance $J72$. However, this is not the case for instance $J51$, where Aggr-P(agg-4) performs best. These differences do not only occur for the new heuristics, TA-agg1-s also shows strong differences in the number of runs having lost sales ($J51$ has 50% lost sales, whereas, $J72$ has 0 lost sales). Adapting, for example, the production capacities, the minimum lotsizes or the cost structure of the problems will provide further insights of why the aforementioned differences between $J51$ and $J72$ exist. Afterwards, it is possible to think about the impact of these insights on the heuristics. Correlating with this, it should be examined how the best number of families can be determined in advance and in which cases TA-GLPK or TA-agg4 or even another heuristic (e.g., TA-agg1-s) is preferable to solve the aggregated problem.

Moreover, the percentage of lost sales is still very high for all new heuristics of this thesis (see Table 4.4). As Meyr and Mann (2013) already proposed, it could be helpful to increase production coefficients instead of reducing capacities in a second iteration of the decomposition heuristic. This approach enables a more detailed reaction to the arising lost sales since it is setup-family-specific and not only line-specific. However, it could happen that an increased production coefficient leads to feasibility problems due to scarce capacities in the aggregated model. Therefore, it might also be useful to introduce time-dependent production coefficients. Thereby, it will be possible to increase the production coefficient of family g only in macroperiods with occurring lost sales for the products of this setup family.

Acknowledgments The author is very grateful to Herbert Meyr for providing the implementation of the decomposition heuristic of Meyr and Mann (2013).

Bibliography

Almeder, C., Almada-Lobo, B., 2011. Synchronisation of scarce resources for a parallel machine lot-sizing problem. International Journal of Production Research 49 (24), 7315–7335.

Baldo, T. A., Santos, M. O., Almada-Lobo, B., Neto, R. M., 2014. An optimization approach for the lot sizing and scheduling problem in the brewery industry. Computers & Industrial Engineering 72, 58–71.

Bitran, G. R., Hax, A. C., 1977. On the design of hierarchical production planning systems. Decision Sciences 8 (1), 28–55.

Camargo, V. C. B., Toledo, F. M. B., Almada-Lobo, B., 2014. HOPS – hamming-oriented partition search for production planning in the spinning industry. European Journal of Operational Research 234 (1), 266–277.

Copil, K., Wörbelauer, M., Meyr, H., Tempelmeier, H., 2017. Simultaneous lotsizing and scheduling problems: a classification and review of models. OR Spectrum 39 (1), 1–64.

Drexl, A., Kimms, A., 1997. Lot sizing and scheduling – survey and extensions. European Journal of Operational Research 99 (2), 221–235.

Fandel, G., Stammen-Hegener, C., 2006. Simultaneous lot sizing and scheduling for multi-product multi-level production. International Journal of Production Economics 104 (2), 308–316.

Ferreira, D., Clark, A. R., Almada-Lobo, B., Morabito Neto, R., 2012. Single-stage formulations for synchronised two-stage lot sizing and scheduling in soft drink production. International Journal of Production Economics 136 (2), 255–265.

Ferreira, D., Morabito Neto, R., Rangel, S., 2009. Solution approaches for the soft drink integrated production lot sizing and scheduling problem. European Journal of Operational Research 196 (2), 697–706.

Ferreira, D., Morabito Neto, R., Rangel, S., 2010. Relax and fix heuristics to solve one-stage one-machine lot-scheduling models for small-scale soft drink plants. Computers & Operations Research 37 (4), 684–691.

Figueira, G., Santos, M. O., Almada-Lobo, B., 2013. A hybrid VNS approach for the short-term production planning and scheduling: a case study in the pulp and paper industry. Computers & Operations Research 40 (7), 1804–1818.

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

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

Fleischmann, B., Meyr, H., 1997. The general lotsizing and scheduling problem. OR Spectrum 19 (1), 11–21.

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

Günther, H.-O., Grunow, M., Neuhaus, U., 2006. Realizing block planning concepts in make-and-pack production using MILP modelling and SAP APO. International Journal of Production Research 44 (18-19), 3711–3726.

Haase, K., 1994. Lotsizing and scheduling for production planning. Springer, Berlin.

Haase, K., 1996. Capacitated lot-sizing with sequence dependent setup costs. OR Spectrum 18 (1), 51–59.

Hallefjord, A., Jörnsten, K. O., Värbrand, P., 1993. Solving large scale generalized assignment problems - an aggregation/disaggregation approach. European Journal of Operational Research 64 (1), 103–114.

Hax, A. C., Meal, H. C., 1975. Hierarchical integration of production planning and scheduling. In: Geisler, M. (Ed.), Logistics. TIMS Studies in the Management Sciences, Vol. 1. Elsevier Amsterdam, pp. 53–69.

Jain, A. K., Murty, M. N., Flynn, P. J., 1999. Data clustering: A review. ACM Computing Surveys 31, 264–323.

Karmarkar, U. S., Schrage, L., 1985. The deterministic dynamic product cycling problem. Operations Research 33 (2), 326–345.

Kleindienst, E., 2004. Aggregation und Allokation in der hierarchischen Produktionsplanung. Dt. Universitäts-Verl., Wiesbaden.

Lang, J. C., 2010. Production and Inventory Management with Substitutions. Lecture Notes in Economics and Mathematical Systems. Springer, Heidelberg et al.

Lasdon, L., Terjung, R. C., 1971. An efficient algorithm for multi-item scheduling. Operations Research 19 (4), 946–969.

Leisten, R., 1996. Iterative Aggregation und mehrstufige Entscheidungsmodelle - Einordnung in den planerischen Kontext, Analyse anhand der Modelle der Linearen Programmierung und Darstellung am Anwendungsbeispiel der Hierarchischen Produktionsplanung. Physica-Verl., Heidelberg.

Mac Cawley, A. F., 2014. The international wine supply chain: challenges from bottling to the glass. Ph.D. thesis, Georgia Institute of Technology.

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

Meyr, H., 1999. Simultane Losgrößen- und Reihenfolgeplanung für kontinuierliche Produktionslinien. Deutscher Universitätsverlag, Wiesbaden.

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

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

Meyr, H., 2004. Simultane Losgrößen- und Reihenfolgeplanung bei mehrstufiger kontinuierlicher Fertigung. Zeitschrift für Betriebswirtschaft 74 (6), 585–610.

Meyr, H., Mann, M., 2013. A decomposition approach for the general lotsizing and scheduling problem for parallel production lines. European Journal of Operational Research 229 (3), 718–731.

Mohammadi, M., 2010. Integrating lotsizing, loading, and scheduling decisions in flexible flow shops. The International Journal of Advanced Manufacturing Technology 50 (9-12), 1165–1174.

Mohammadi, M., Poursabzi, O., 2014. A rolling horizon-based heuristic to solve a multi-level general lot sizing and scheduling problem with multiple machines (MLGLSP_MM) in job shop manufacturing system. Uncertain Supply Chain Management 2 (3), 167–178.

Pahl, J., Voß, S., Woodruff, D. L., 2011. Discrete lot-sizing and scheduling with sequence-dependent setup times and costs including deterioration and perishability constraints. In: Proceedings of the 44th Hawaii international conference on system sciences. IEEE Computer Society, Los Alamitos, pp. 1–10.

Schneeweiß, C., 2003. Distributed Decision Making, 2nd Edition. Springer, Berlin, Heidelberg.

Seeanner, F., 2013. Multi-stage simultaneous lot-sizing and scheduling – planning of flow lines with shifting bottlenecks. Produktion und Logistik. Springer Gabler, Wiesbaden.

Seeanner, F., Meyr, H., 2013. Multi-stage simultaneous lot-sizing and scheduling for flow line production. OR Spectrum 35 (1), 33–73.

Smith-Daniels, V. L., Smith-Daniels, D. E., 1986. A mixed integer programming model for lot sizing and sequencing packaging lines in the process industries. IIE Transactions 18 (3), 278–285.

Stadtler, H., 1988. Hierarchische Produktionsplanung bei losweiser Fertigung. Physica-Schriften zur Betriebswirtschaftslehre. Physica, Heidelberg.

Steven, M., 1994. Hierarchische Produktionsplanung, 2nd Edition. Physica–Verlag, Heidelberg.

Storoy, S., 1996. Optimal weights and degeneracy in variable aggregated linear programs. Operations Research Letters 19 (1), 29–31.

Tempelmeier, H., Buschkühl, L., 2008. Dynamic multi-machine lotsizing and sequencing with simultaneous scheduling of a common setup resource. International Journal of Production Economics 113 (1), 401–412.

Tiacci, L., Saetta, S., 2012. Demand forecasting, lot sizing and scheduling on a rolling horizon basis. International Journal of Production Economics 140 (2), 803–814.

Toledo, C. F. M., de Jesus Filho, J. E. F., França, P. M., 2008a. Meta-heuristic approaches for a soft drink industry problem. In: IEEE International Conference on Emerging Technologies and Factory Automation, 2008. pp. 1384–1391.

Toledo, C. F. M., de Oliveira, L., de Freitas Pereira, R., França, P. M., Morabito, R., 2014. A genetic algorithm/mathematical programming approach to solve a two-level soft drink production problem. Computers & Operations Research 48 (1), 40–52.

Toledo, C. F. M., de Oliveira, L., de Oliveira, R. R. R., Pereira, M. R., 2010. Parallel genetic algorithm approaches applied to solve a synchronized and integrated lot sizing and scheduling problem. In: Proceedings of the 2010 ACM Symposium on Applied Computing. New York, pp. 1148–1152.

Toledo, C. F. M., França, P. M., Morabito, R., Kimms, A., 2009. Multi-population genetic algorithm to solve the synchronized and integrated two-level lot sizing and scheduling problem. International Journal of Production Research 47 (11), 3097–3119.

Toledo, C. F. M., França, P. M., Rosa, K. A., 2008b. Evaluating genetic algorithms with different population structures on a lot sizing and scheduling problem. In: Proceedings of the 2008 ACM Symposium on Applied Computing (Fortaleza, Ceara, Brazil). pp. 1777–1781.

Toledo, C. F. M., Kimms, A., França, P. M., Morabito, R., 2015. The synchronized and integrated two-level lot sizing and scheduling problem: Evaluating the generalized mathematical model. Mathematical Problems in Engineering Article ID 182781.

Toso, E. A. V., Morabito, R., Clark, A. R., 2009. Lot sizing and sequencing optimisation at an animal-feed plant. Computers & Industrial Engineering 57 (3), 813–821.

Zhu, X., Wilhelm, W. E., 2006. Scheduling and lot sizing with sequence-dependent setup: a literature review. IIE Transactions 38 (11), 987–1007.

5 Summary and outlook

The present thesis focuses on simultaneous lotsizing and scheduling. The intention was to create significant improvements for solving simultaneous lotsizing and scheduling problems arising in practical applications. In detail, the thesis is structured into three main topics: providing a comprehensive literature review concerning simultaneous lotsizing and scheduling, formulating a general model to consider secondary resources and developing a heuristic to solve large-scaled simultaneous lotsizing and scheduling problems. The following section describes the results in detail. Additionally, further research topics are discussed in Section 5.2.

5.1 Summary

Simultaneous lotsizing and scheduling problems often consist of only one bottleneck production stage which may comprise multiple parallel production lines having limited capacities. Since a product can be produced on more than one line, a simultaneous planning of all lines is necessary. Furthermore, parameters like production costs or production speeds are line- and product-dependent. Changeovers from one product to another product cause sequence-dependent setup times and setup costs. I.e., the previously produced product influences the amount of setup times and setup costs. Moreover, holding costs occur for storing the products until they are used to fulfill a dynamic demand for several products. Therefore, it is necessary to solve a lotsizing problem as well. All in all, due to strong interdependencies, the lotsizes, the sequences and the scheduling on the different lines must be planned simultaneously. Usually, simultaneous lotsizing and scheduling problems are formulated as mixed integer programming models having an objective function which sums up all costs and has to be minimized. Since the problems are complex, especially for real world applications, several solution heuristics have been developed to solve them. (see, e.g., Meyr 1999, Chapter 3 and Copil et al. 2017)

Chapter 2 provides a comprehensive literature review of simultaneous lotsizing and scheduling problems. The focus of the review is placed on the different model formulations. Therefore, more than 160 different formulations have been identified and are briefly described. To enable the identification of differences between the model formulations, a detailed classification scheme is used (see Meyr 1999). Aspects, included in the classification scheme are, for example, the number of considered production stages, the used time structure of the model formulations and the setup characteristics, i.e., among others, whether setup times and setup costs exist and whether they are sequence-dependent or sequence-independent. If model formulations have been developed to solve real world applications, the concerned industries are specified as well. Moreover, used solution heuristics and additional features such as the consideration of perishable products are documented too. Using this classification scheme, it is possible to describe the development of simultaneous lotsizing and scheduling and particularly to identify research gaps.

One important key finding is that recent models describe the planning problems in more detail. One reason for this are the improved possibilities (better hardware and improved standard solvers) to solve more complex models. Moreover, the models are often formulated to be applied in practice. Therefore, it is essential that all relevant aspects are considered. For example, many recent models consider multi-level bill-of-materials structures and multiple production stages which is important if the bottleneck production stage shifts depending on the demand. Additionally, some recent formulations do not only consider the production lines (*primary resources*) as scarce capacities, but also some *secondary (scarce) resources*, like personnel or raw materials. This is picked up in Chapter 3 of the present thesis and will be summarized in the following.

As described before, some models include secondary resources. Secondary resources have to be considered if their availability is limited which is usually the case if they are expensive, like personnel, or if purchasing problems exist as it could be the case for some raw materials. The importance of including secondary resources can be easily shown using the following example. Assume that there is only one setup operator which is responsible for the setups on two production lines, but is only capable of setting up one line at the same time. If the setup operator is ignored during planning, it might happen that both lines should be set up simultaneously. Obviously, this plan is infeasible in reality, since the setup operator cannot set up both lines in parallel.

Since the literature review of Chapter 2 is focused on simultaneous lotsizing and scheduling in general, a further review is provided which analyzes the models incorporating secondary resources in more detail (see Section 3.2). It is described which secondary resources are incorporated in the different problems of the literature and how they are modeled. Moreover, a new classification scheme has been developed to structure the reviewed literature. Two main attributes of secondary resources – *shareability* and *substitutability* – have been identified. Shareability has two different potential values: on the one hand, a *disjunctive* resource can only be used on a single line at the same time, but it can be used several times such as a setup operator. On the other hand, a *cumulative* resource can be supplied simultaneously to multiple lines, but it can only be used once like a raw material. Substitutability differentiates between the following potential values: in case *without substitutes*, it is clearly defined which resources have to be used. If substitutes exist (*with substitutes*), alternative resources (e.g., high and low skilled workers) are available which can perform the same (simple) operation. Using substitutes, the flexibility of the model is increased. Especially, if the skill level is considered more generally as capability of performing a task. For example, if there are three lines and two setup operators, and the first setup operator is specialized in line 1 and 2 and the second setup operator is specialized in line 2 and 3, all combinations of two simultaneous setups are possible (line 1 and 2 or line 1 and 3 or line 2 and 3). This is not the case if only one of the two setup operators is specialized in line 2. Based on the aforementioned four main attributes, the following types of secondary resources have been identified: *disjunctive resources with and without substitutes* and *cumulative resources with and* *without substitutes*. Among others, the classification scheme also describes which *states* (production, setup and conservation of a setup state) require secondary resources and whether a state requires only one or several different secondary resources.

The review shows that many models are very specialized. For example, there are models which only consider a single setup operator or only raw materials. If this is the case, it is impossible to represent a production scenario which considers two setup operators and raw materials simultaneously. Therefore, a general model (see Section 3.3-3.5) has been developed which has the following characteristics concerning secondary resources:

- All four types of secondary resources can be represented at the same time. For example, it is possible to have a scenario which exactly defines which raw materials are necessary for which *processes* (changeover from product i to j, conservation of the setup state of product j (standby) and production of product j) and at the same time considers two setup operators of which one can do every setup and the other one is only capable of doing simple setups.
- Unnecessary variables and constraints can be "deactivated". I.e., if a planning problem does not consider substitutes, all constraints and variables concerning substitutes can be easily omitted in the model to reduce complexity.
- It is possible that a process requires several different secondary resources. I.e., it is possible to represent a scenario in which two different setup operators and a tool are necessary during the changeover from product i to j. Additionally, it is possible that a secondary resource is capable of taking care of different processes, e.g., the tool can also be necessary for producing product j.
- The model formulation allows that each process needs secondary resources. For example, it is possible that a worker is necessary during the changeover from product 1 to 2, and a tool is required for the changeover from product 2 to 3 and during production of product 3. Additionally, it is also possible to assure that a substitute resource must be used for several consecutive processes on a certain line. Consider that tool A and B are substitutes for each other and one of the two tools is required for the setup of product 1 and for production of product 1. In this case, it is possible to assure that the tool mounted during the changeover also has to be used for production.

Most but not all of these characteristics are already incorporated in model formulations. However, often, only a few of these characteristics are considered in a single model formulation and no model exists which considers all of them simultaneously.

The new model is based on a simultaneous lotsizing and scheduling model named *general lotsizing and scheduling problem for parallel lines* (GLSPPL) (see Meyr 2002). The model considers a limited planning horizon which is divided into several macroperiods. Each macroperiod itself consists of several microperiods. In the original model formulation these microperiods can have different lengths on all lines. However, in the new model, the microperiod time grid is identical on all lines. Moreover, only one process is allowed per microperiod. This enables the synchronization of disjunctive secondary resources without unnecessarily long assignments of secondary resources to production lines. For example, if a setup operator is assigned to line 1 in microperiod 1, he can be used on another line in microperiod 2. A common time grid combined with the restriction of only allowing one state per microperiod reduces the flexibility of the model. Assume two production lines which should be set up in microperiod 1. The setup on line 1 lasts for 30 minutes and the setup on line 2 lasts for 40 minutes. Obviously, it is impossible to set up both lines in the first microperiod, since only one state is allowed per microperiod and the starting and ending times of all microperiods must be identical on all lines. Therefore, the new model allows *continuous setups*, i.e., setups can last for several microperiods. Thus, the first microperiod has a length of 30 minutes and the second microperiod will be 10 minutes long to finish the setup on line 2.

Section 3.5 proposes additional features which only seem relevant in some practical applications. One important topic is the splitting of setup operations into dismounting, cleaning and mounting operations. This splitting provides more flexibility to find feasible production plans. Assume a setup operation which takes five hours, but dismounting takes only 10 minutes. Using the model without splitting of the different setup operations, a tool which has been mounted on a machine to produce a product is assigned to the machine for the complete changeover from this product to the following product, i.e., for five hours. If the model considers splitting the setup operations, the tool can be set free after 10 minutes and can then be used on another line. To the best of our knowledge, this is the first simultaneous lotsizing and scheduling model which considers the setup process in such detail.

Small experiments demonstrate the applicability of the general model formulation. Future research should focus on extensive numerical tests and the development of solution heuristics. This will be further discussed in Section 5.2.

Finally, since up-to-date heuristics still need long computation times to solve large-scaled instances, Chapter 4 proposes a decomposition heuristic for the GLSPPL. At first, products of the original problem are aggregated to *setup families*. Therefore, two algorithms – Cluster-S and Cluster-R – have been developed. Since some products cannot be produced on all lines, both algorithms respect the following assumption: product i and j only can be assigned to the same setup family if they are identical in the characteristic of being able to be produced on line l or not ($\forall\ l$). I.e., if both products can be produced on line 1 and both products cannot be produced on line 2, they can be assigned to the same setup family. If only product j can be produced on line 1, the products i and j cannot be assigned to the same family. Additionally, Cluster-S considers the setup times to decide which product should be assigned to which setup family. The intention is to reduce the loss of information which could arise if the setup times of all products of a family are aggregated to a single setup time of the family. Cluster-R assigns products to setup families, despite of the abovementioned restriction, based on random numbers.

Using the setup families, an aggregated version of the original problem is generated. The resulting problem is solved using the heuristics TA-GLPK or TA-agg4 (see Meyr and Mann 2013). The scheduled setup families' lotsizes are disaggregated to define line- and product-specific demands using an assignment problem formulated as mixed integer program. If a setup family, e.g., consisting of three products is scheduled on, e.g., two lines, it is possible that all three products are assigned to both lines.

In most cases, this is unnecessary and thus undesired. Disadvantages which arise based on such an assignment are that the resulting single-line problems comprise a high number of products and that setup costs and setup times for all products will arise on both lines. Sometimes, the described effect can be avoided by merely incorporating production and holding costs in the assignment problem. However, depending on the cost structure, the desired effect is not always achieved. Therefore, penalty costs are introduced to avoid the aforedescribed problem.

The line-dependent demands are used to formulate single-line problems which are solved independently using the fast single-line heuristic TADR (see Meyr 2000). The single-line schedules altogether form a solution for the multi-line problem. However, sometimes further optimization potential exists, thus, the setup patterns of the single-line schedules are used to generate a linear programming formulation of the original multi-line problem. If the solution of this problem does not completely satisfy the total demand, a second iteration of the heuristic is started. In this second iteration, the production capacities of the aggregated problem are modified to hopefully influence the aggregated production schedule in a way that a final solution without lost sales can be found.

For numerical tests, four new heuristics have been defined Aggr-P(TA-GLPK), Aggr-P(TA-agg4), Aggr-PR(TA-GLPK) and Aggr-PR(TA-agg4). Aggr-P uses Cluster-S and Aggr-PR uses Cluster-R to define setup families. The heuristic names in parentheses define the heuristic used to solve the aggregated problem. The heuristic TA-agg1-s of Meyr and Mann (2013) serves as benchmark since it performed better than other heuristics to solve large-scaled problems so far. Two test instances from industrial applications have been solved. The first one is from the acrylic glass production. It comprises 51 products, three production lines and 12 macroperiods. The second instance is from a label printing company and comprises 72 products, seven lines and 12 macroperiods. Since all heuristics have stochastic components, several runs of each instance-heuristic-combination have been performed and performance measures like average computing time, "objective deviation"[29] and "percentage of lost sales"[30] are considered. In the case of 51 products (condensed to 19 setup families), both Aggr-P heuristics perform very well (Aggr-P(TA-agg4) performs best having an objective deviation of 2.4% and an average computing time of 119 seconds). As expected, both Aggr-PR heuristics perform a bit worse compared to Aggr-P (Aggr-PR(TA-GLPK) has an objective deviation of 4.1% and Aggr-PR(TA-agg4) of 4.3%). However, they are still preferable to TA-agg1-s which has an average objective deviation of 4.0% but is slower (432 seconds compared to Aggr-PR(TA-GLPK) needing 232 seconds and Aggr-PR(TA-agg4) needing only 130 seconds). In the case of 72 products (condensed to 25 setup families) the picture is not that clear. Each heuristic performs acceptably well having an objective deviation below 1%. Aggr-P(TA-agg4) is the fastest heuristic in finding a first feasible solution. It is even a bit faster compared to Aggr-P(TA-GLPK) which performs best concerning the objective deviation. Aggr-PR(TA-GLPK) needs a very long time to find a first feasible solution. However, since Aggr-PR was initially intended to serve as a benchmark meaning to show that defining setup families by a reasonable approach generates advantages compared to a mainly random assignment, such a result has been expected.

[29]Average percentage deviation from the best known objective function value.

[30]Percentage of runs with lost sales.

A further experiment using instance *J51* shows that the number of setup families has a high influence on the solution quality. As expected, high numbers of setup families perform worse, since the aggregated problem is still very complex. However, a scenario with 5 families has never reached objective deviations as small as for a scenario with 14 families. The reason might be the loss of information while defining the aggregated parameters of the problem. As a consequence, the aggregation-disaggregation process might create single-line problems which do not include the optimal solution of the original problem anymore. Nevertheless, the scenario with 5 families has a lower percentage of runs with lost sales compared to the scenario with 14 families.

Altogether, the new heuristics can serve as alternatives to previous heuristics. Further numerical tests are necessary to analyze if the heuristics provide relevant improvements like, e.g., for test instance $J51$ or if they only perform similarly or sometimes even worse to former heuristics like, e.g., for test instance $J72$.

5.2 Outlook

One aspect worth considering in future research is the incorporation of multi-stage bill-of-materials and multiple production stages. Chapter 2 has shown this to be a relevant topic, especially in practical applications. As one can see for example in Seeanner and Meyr (2013), synchronization between the different production stages is an important issue. One approach is to assume that quantities produced in a period on one production stage can serve as pre-products for another production stage in the following period (see, e.g., Meyr 2004). Obviously, comparing large-bucket and small-bucket models, the resulting lead time from producing a product until its usage on another production stage is shorter in the latter case. However, in the GLSPPL formulation of Meyr (2002), microperiods cannot be used for synchronization of production stages directly since the starting times of the microperiods can be different on each line. Therefore, Meyr (2004) has introduced an identical microperiod time grid on all lines to synchronize the different production stages. (see also Seeanner and Meyr 2013) This common time grid has already been used in the model formulation of Chapter 3 to assure the synchronization of disjunctive resources. Thus, extending the new model to multiple production stages does not seem difficult. However, up-to-date multi-stage models allow several states (e.g., setup and production) in one microperiod and enable to use a product of a previous production stage on a successor stage in the same microperiod (see, e.g. Seeanner and Meyr 2013). Allowing several states per microperiod requires significant adaptions of the model. However, this adaption will reduce the necessary number of microperiods as shown in the following.

Allowing only one state per microperiod in the model formulation of Chapter 3 may lead to a high number of microperiods and thus to a high number of variables. Therefore, it seems reasonable to relax this restriction (see the example on page 158: allowing conservation of a setup state and a changeover in one microperiod, enables to do both setups in the first microperiod instead of needing two microperiods). However, on average, microperiods will be longer and therefore it seems reasonable to allow disjunctive resources to be assigned consecutively to two lines in one microperiod. As shown before, multi-stage simultaneous lotsizing and scheduling models already allow several states per microperiod. Therefore, for example Seeanner and Meyr (2013) propose constraints which assure the synchronization between different production stages within the same microperiod. Such constraints would also be necessary for the synchronization of secondary resources within a microperiod. To give an example of how this concept could be applied, consider two production lines and a worker who is responsible for changeovers on both lines. Both lines should be set up in microperiod 1. Therefore, a constraint is necessary which assures that on one of both lines the previous setup state is conserved until the setup on the other line is finished. I.e., the restriction must assure that the microperiod is long enough to subsequently perform both setups. Of course, this approach has to be worked out in detail to consider all types of resources and all concerned states.

Moreover, it is necessary to do further numerical tests using different and bigger instances. Despite the aforementioned approach of reducing microperiods and therefore variables, it will be necessary to develop heuristics which are capable of solving the problem for test instances of practical relevance.

Since the decomposition heuristics of Chapter 4 have only been tested using two instances, it is necessary to do further extensive numerical tests. Particularly, since, for test instance $J72$, the new heuristics showed similar performance results compared to previous heuristics. In this case, it seems worthwhile to adapt instance $J72$, e.g., in terms of the given capacities, the minimum lotsizes or the setup costs and times. This will help to identify if this instance has a particular characteristic which has led to low objective deviations for all tested heuristics and has led to the result that Aggr-P(TA-agg4) has not performed best like it was the case for test instance $J51$. The insights of the further numerical tests will help to improve the solution approach.

The heuristics of Chapter 4 still have high percentages of lost sales (see Table 4.4). I.e., the approach of reducing capacities in the second iteration of the decomposition heuristic does not perform well in every case. As already proposed by Meyr and Mann (2013, p. 730), it could help to increase the production coefficients of those setup families showing lost sales. This approach reserves time which only can be used in the single-line problems specifically for the products of the setup family which has had lost sales in the previous iteration. However, it could lead to feasibility problems in the aggregated problem, since more production time is needed than before. To counteract this, time-dependent production coefficients (see Meyr and Mann 2013, p. 723) will be useful. In this case, it is possible to increase the production coefficients only for setup families and macroperiods which have had lost sales in the previous iteration of the decomposition heuristic.

Bibliography

Copil, K., Wörbelauer, M., Meyr, H., Tempelmeier, H., 2017. Simultaneous lotsizing and scheduling problems: a classification and review of models. OR Spectrum 39 (1), 1–64.

Meyr, H., 1999. Simultane Losgrößen- und Reihenfolgeplanung für kontinuierliche Produktionslinien. Deutscher Universitätsverlag, Wiesbaden.

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

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

Meyr, H., 2004. Simultane Losgrößen- und Reihenfolgeplanung bei mehrstufiger kontinuierlicher Fertigung. Zeitschrift für Betriebswirtschaft 74 (6), 585–610.

Meyr, H., Mann, M., 2013. A decomposition approach for the general lotsizing and scheduling problem for parallel production lines. European Journal of Operational Research 229 (3), 718–731.

Seeanner, F., Meyr, H., 2013. Multi-stage simultaneous lot-sizing and scheduling for flow line production. OR Spectrum 35 (1), 33–73.