Lead Time Considerations for the Multi-Level Capacitated Lot-sizing Problem*

Christian Almeder[a,c], Diego Klabjan[b], Bernardo Almada-Lobo[c]

August 26, 2009

[a]University of Vienna, Faculty of Economics, Business and Statistics, Brünnerstr. 72, A-1210 Vienna, Austria.
[b] Department of Industrial Engineering and Management Sciences, Northwestern University, 2145 Sheridan Road, Tech M239 Evanston, IL 60208-3119, USA.
[c]Faculdade de Engenharia da Universidade do Porto, Rua Dr. Roberto Frias s/n, Porto 4200-465, Portugal.

Abstract

The classical multi-level capacitated lot-sizing problem is often not suitable to correctly capture resource requirements and precedence relations. We tackle this issue by explicitly modeling these two aspects. Two models are presented; one considering batch production and the other one allowing lot-streaming. Comparisons with traditional models demonstrate the capability of the new approach to deliver more realistic and practically feasible results. Large-scale cases are solved by Benders, which has been enhanced in several novel ways.
Keywords: *Lot-sizing, Scheduling, Mathematical Programming, Optimization*
Mathematics Subject Classification: *90B30, 90C11*

1 Introduction

Extensive work has been reported in tackling the multi-level capacitated lot-sizing problem (ML-CLSP), which lies at the core of many production planning systems. Numerous models and solution approaches to support MLCLSP have been practically integrated with Manufacturing Resource Planning, Master Production Scheduling or Enterprize Resource Planning systems. MLCLSP fits the highest level of planning in one of the most prominent hierarchical systems, namely two-level production planning and scheduling. In fact, the complexity of the product structure (bill-of-materials or BOM) of real-world cases has motivated researchers to decompose hierarchically the overall planning system into a set of more manageable subsystems. At the upper level of two-level production planning and scheduling, the tactical driver is to determine the amount of each product that must be produced in each time-period across several resources (sizing of lots). The production scheduling (sequencing of lots) is left to the lower operational level of planning. In order to ensure that the plans generated at the upper-level can be scheduled

*This research has been supported, in part, by Grant # J2815-N13 from the Austrian Science Foundation.

afterwards, the interdependencies between both levels should be incorporated in the solution approaches, through top-down and bottom-up influences.

The majority of the models and algorithms proposed for MLCLSP rely on one of the following two assumptions: either lead times are neglected, thus allowing predecessors and successors to be produced in the same period; or lead times account for at least one period for each component, forcing the throughput time (in number of periods) of the finished products to be at least equal to the number of levels of the BOM. Buschkühl et al. (2008) report in their review that out of 16 papers dealing with metaheuristic solutions to MLCLSP, only one methodology considers lead time, while all others neglect it. The present work is motivated by the observation that the majority of the solutions published for well-known MLCLSP instances suffer from this assumption. Indeed, the zero lead time assumption leads to plans that are not implementable, as the lower-level scheduling problem is likely to be infeasible. This is more pronounced as the capacity tightens. On the other hand, positive lead time usually results in huge amounts of work-in-process, which tends to increase with a larger number of levels in the BOM.

We study synchronization of the batches of products as an important extension of standard MLCLSP. In each time period, sizing and sequencing of production lots are simultaneously addressed in such a way that precedence relationships of products are respected. We propose two models, one restricted to batch production and the other one allowing lot-streaming. In the latter case modeling is tricky and it requires a rigorous argument to establish correctness.

In order to solve large-scale instances, we resort to a well known decomposition approach, Benders decomposition, as state-of-the-art commercial solvers are inappropriate. Benders decomposition (Benders, 1962) is a classical solution approach for certain linear optimization problems. The overall formulation is partitioned into two models, the restricted master problem containing only a subset of original variables and constraints, and the subproblem, obtained from the original one by fixing some of the variables to values obtained in the restricted master problem. In MLCLSP with synchronization of batches, the restricted master problem deals with the binary setup-related variables defining the sequence of batches, while the subproblem models the continuous variables representing the production and inventory quantities, as well as the starting and finishing times of the batches. It is well known that the basic Benders decomposition algorithm often exhibits slow convergence. We implement new variants and refine some well known extensions of the standard Benders algorithm to improve its efficiency to better solve large instances.

The main contributions of our work are as follows. We show that most of the solutions reported in the literature obtained by standard MLCLSP with zero lead times are infeasible, and those that consider positive lead times entail significant work-in-process. We propose two new linear mixed-integer programming formulations. The batching formulation considers production taking place in batches such that units produced within each lot can only be processed as raw materials on the subsequent BOM level once the whole batch is completed. The lot-streaming or non-batching model relaxes this requirement, allowing components to be transformed further on as soon as they are released. We present results establishing correctness of the two models and comparing them with respect to the produced cost. Regarding Benders decomposition, our computational experiments show that the newly developed penalty variant outperforms all

other versions, and consistently beats a state-of-the-art commercial solver. The penalty variant stabilizes the solutions of the restricted master problem by imposing a penalty for deviating from the best known solution where the solution quality is specified by the quality of the approximation of the subproblem. To the best of our knowledge such an enhancement has not been studied in the past.

In Section 2 we introduce MLCLSP and the major hurdles of its standard formulation proposed in the literature. In Section 3 two novel multi-level formulations are developed for the capacitated lot-sizing and scheduling problem, based assumptions for batching and lot-streaming production environments, respectively. We then address several possible extensions to these models. Section 4 is devoted to Benders decomposition. The overall algorithm is shown, along with several variants and sets of valid inequalities to speed up the convergence of the algorithm. Computational results for a variety of well-known test instances reported in literature are given in Section 5. The introduction is concluded with a literature review.

Related Literature

Lot-sizing problems have been studied by researchers in different variation during the last decades. One of the initial works describing the trade-off between the setup and holding costs for a dynamic demand scenario is provided by Wagner and Within (1958). Introducing capacity limitations and considering several products simultaneously lead to the capacitated lot-sizing problem, which Bitran and Yanasse (1982) have shown to be NP-hard. In the case of positive setup times Maes et al. (1991) proved that already the feasibility problem is NP-complete. MLCLSP dealt in this paper was first introduced by Billington et al. (1983) and it extended the lot-sizing models towards material requirements planning. Before this work lot-sizing problems have mainly been applied on the final product level only.

A recent review about different model formulations and solution methods for MLCLSP can be found in Buschkühl et al. (2008). This review points out that it is necessary to consider lead times, but many researchers neglect them. For example, 15 out 16 papers on metaheuristic solution methods for MLCLSP mentioned in that review neglect lead time. The remaining work Berretta et al. (2005) assumes that some items have no lead times and others have positive lead times of one period or more.

The present paper deals with the lead time consideration for MLCLSP. This involves also scheduling decisions. The class of small-bucket lot-sizing and scheduling problems try to capture both lot-sizing and scheduling decisions, Drexl and Kimms (1997). Wolsey (2002) provides a comprehensive study and classifications scheme for different small- and big-bucket models. His analysis shows that the LP relaxation of small-bucket models usually delivers very weak lower bounds. Only with customized reformulations and valid inequalities added to the problem an improvement of the lower bound is possible. In contrast, most big-bucket models (MLCLSP is classified as such) provide much better lower bounds.

For the class of big-bucket lot-sizing models, there are extensions to incorporate scheduling decisions. Haase (1996), Gupta and Magnusson (2005), and Almada-Lobo et al. (2008) developed extensions for the capacitated lot-sizing problem to deal with sequence dependent setups. The review by Zhu and Wilhelm (2006) analyzes the research on the intersection of lot-sizing and

scheduling from the scheduling point of view.

Another subject related to this work is the problem of synchronizing the use of common resources. Tempelmeier and Buschkühl (2008) describe a problem where a common setup operator has to perform the setup operations on different machines. Hence, it is necessary to ensure that there are no two setup operations at the same time. In Almeder and Almada-Lobo (2009) a model describing the synchronization of a secondary resource used in a parallel machine environment is developed.

The paper most closely related to our work is the work by Fandel and Stammen-Hegene (2006). They model a similar problem, but in contrast to our approach the authors develop a non-linear formulation. The complexity of that formulation prohibits to solve even small instances. To the best of our knowledge, efficient solution methodologies for their model are not known as of this date.

2 Motivation

In this paper, we study extensions to the following classical MLCLSP as formulated by Billington et al. (1983).

We are given N products, T time periods and M machines. Each product has a manufacturing lead time and deterministic demand in each time period. In addition, we are given the underlying BOM. The problem is to find production quantities in each time period that obey BOM requirements, demand requirements, limited capacity resources and the production and holding costs are minimized. It reads

$$\min \sum_{i=1}^{N} \sum_{t=1}^{T} (c_i \cdot Y_{it} + h_i \cdot I_{it}) \tag{1}$$

subject to

$$I_{it} = I_{i(t-1)} + X_{i(t-l_i)} - \sum_{j \in \Gamma(i)} a_{ij} \cdot X_{jt} - E_{it} \qquad i, t > 1 \tag{2}$$

$$\sum_{i=1}^{N} (p_{mi} \cdot X_{it} + s_{mi} \cdot Y_{it}) \leq L_{mt} \qquad m, t \tag{3}$$

$$X_{it} - G \cdot Y_{it} \leq 0 \qquad i, t \tag{4}$$

$$I_{it} \geq 0, X_{it} \geq 0, Y_{it} \in \{0, 1\} \qquad i, t \tag{5}$$

and with the decision variables:

I_{it} inventory level of item i at the end of period t

X_{it} production amount of item i in period t

Y_{it} $\begin{cases} 1 & \text{if item } i \text{ is produced in period } t \\ 0 & \text{otherwise.} \end{cases}$

The parameters read

a_{ij}	quantity of item i required to produce one unit of item j (*gozinto-factor*)
p_{mi}	time for producing one unit of item i on machine m
c_i	setup cost of item i
E_{it}	(external) demand of item i in period t
I_{i0}	initial inventory level of item i
h_i	holding cost of item i
L_{mt}	available capacity of machine m in period t
l_i	lead time of item i (nonnegative integer corresponding to the number of periods)
s_{mi}	time for setting up machine m for the production of item i
$\Gamma(i)$	set of immediate successors of item i based on BOM.

The index set (i, j, t) is defined by $i, j \in \{1, 2, \ldots, N\}$, $t \in \{1, 2, \ldots, T\}$ and $m \in \{1, 2, \ldots, M\}$. Objective function (1) captures the fixed setup cost and the underlying holding cost. Constraints (2) are standard lot-sizing flow requirements capturing BOM and lead times. Limited machine capacity is reflected by (3) and (4) captures the definition of setup variables.

Many researchers dealing with this model assume that lead times are negligible, i.e. predecessors and successors might be produced in the same period ($l_i = 0$ for every i). Since MLCLSP is a big-bucket model and periods are assumed to cover long time slots with several different production batches, the exact scheduling process is postponed to the next planning level. Assuming that the capacity utilization is not too high and/or that there is always the possibility of using overtime to compensate a potential lack of capacity, neglecting the lead time might be reasonable. Nevertheless, in case of capacity tight scenarios and of a limited usage of overtime (which happens for example in a 24-7 production system) this zero lead time assumption may lead to plans that are not implementable in practice.

On the other hand, positive lead times of at least one period ($l_i = 1$ for every i) do deliver feasible solutions that can be implemented. But positive lead times contradict somehow the big-bucket assumption, because in practice they would lead to huge amounts of work-in-process. For example, consider a time period of one day and an average production batch of 3 hours. For a BOM with 10 levels it would take ten days to finish a final item according to MLCLSP, but less than two days in practice. Note, that in the case of positive lead times the objective function (1) does not capture the additional work-in-process. An extra constant cost term depending on the total demand of all periods has to be added.

2.1 Drawbacks of MLCLSP

Let us assume a simple example with four items, three machines, and two periods. Table 1 contains all parameters and Figure 1 depicts the product structure. It is assumed that resource requirements are measured in time units, each machine has a capacity of 1 meaning it is available throughout the whole period and setup times are zero.

The optimal solution of the according MLCLSP model without lead time can be easily obtained (see Table 2). The total cost is 22 units consisting of the setup cost of 20 units and the holding cost of 2 units. It is obvious that this solution is not feasible in practice, because before item 1 can be produced in the first period, items 3 and 4 have to be scheduled. Figure 2 shows two possible implementations of the solution. The first one violates the demand requirement, as

Table 1: Data for the example

item	machine	d_{i1}	d_{i2}	h_i	c_i	p_{mi}
1	A	3	0	3	5	0.1
2	B	0	2	2	5	0.1
3	C	0	0	2	5	0.1
4	C	0	0	1	5	0.1

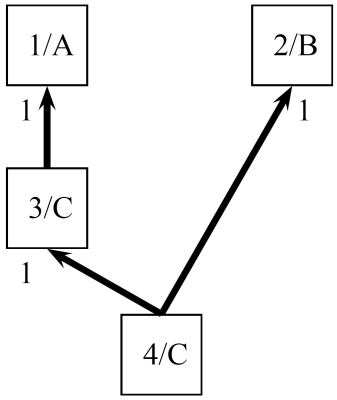

Figure 1: BOM for the example

item 1 is finished in period 2 and not in period 1. The second implementation would solve this issue, but at a cost of an additional setup for item 4. The optimal feasible solution is to move the second batch of item 4 to the second period, ending up with a total cost of 25 units (with no holding costs), but this solution is essentially different from that of the MLCLSP model. Clearly, if we consider one period lead time for MLCLSP, the problem is infeasible.

Table 2: MLCLSP´solution with no lead time for the example

item	x_{i1}	x_{i2}	I_{i1}	I_{i2}
1	3	0	0	0
2	0	2	0	0
3	3	0	0	0
4	5	0	2	0

3 Models

In order to avoid these pitfalls we suggest to embed scheduling to MLCLSP.

3.1 Batching

In this section we develop a multi-level capacitated lot-sizing formulation that schedules the production lots and synchronizes the batches of predecessors and successors. It is an extension of (1)-(5). We assume that each item is assigned to a single machine, which is the case for most problems published and test instances available (see Tempelmeier and Derstroff, 1996). The following additional parameters are introduced:

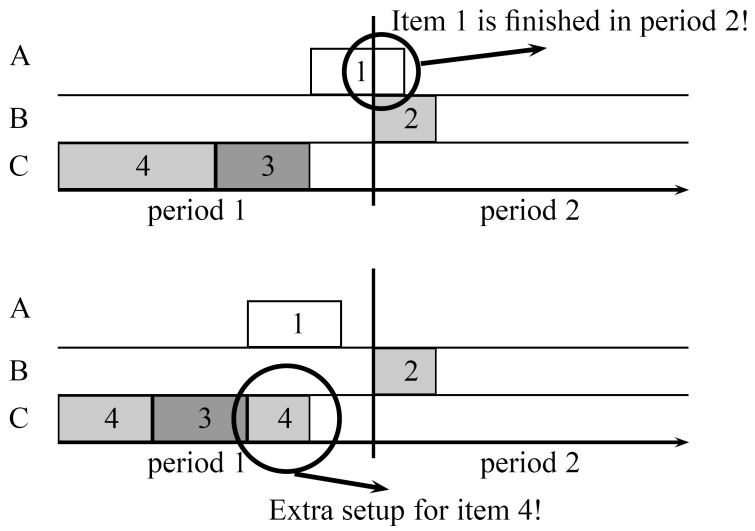

Figure 2: Possible adjustments for the example

$\phi(m)$ set of items that can be assigned to machine m

p_i time for producing one unit of item i (we can omit index m because each item is assigned to a specific machine)

s_{ij} setup time for changeover from product i to product j on the appropriate machine, $s_{ii} = 0$ for each i

c_{ij} cost incurred to set up the machine from product i to product. j, $c_{ii} = 0$, $i, j \in \phi(m)$

We need the following decision variables:

μ^s_{it} start time of the production of item i in period t

$\hat{X}_{ijt}$ production amount of item $j \in \Gamma(i)$ starting before the production of item i is finished in period t

α_{itm} $\begin{cases} 1 & \text{if machine } m \text{ is set up for item } i \text{ at the beginning of period } t \\ 0 & \text{otherwise} \end{cases}$

T_{ijtm} $\begin{cases} 1 & \text{if there is a switch from item } i \text{ to } j \text{ on machine } m \text{ in period } t \\ 0 & \text{otherwise} \end{cases}$

W_{ijt} $\begin{cases} 1 & \text{if production of item } j \text{ starts after the whole batch of item } i \\ & \text{is completed in period } t \text{ and } j \in \Gamma(i) \\ 0 & \text{otherwise.} \end{cases}$

By scaling, we assume without loss of generality that the duration of a period is a single time unit. We first consider the case where the production takes place in batches. In other words, at the start of the production of item i the whole amount of necessary predecessors (raw materials) must be available and the finished items are available for other production processes only after the whole batch is finished. Hence, the available stock level of item i is a discontinuous step function (see Figure 3) where a decrease in stock is due to the consumption of item i by a different item and an increase in inventory results from a finished batch.

Note that $T_{ijtm} = 0$ for every $i \notin \phi(m)$ or $j \notin \phi_m$, and for $i = j$. We also have $\alpha_{imt} = 0$ for

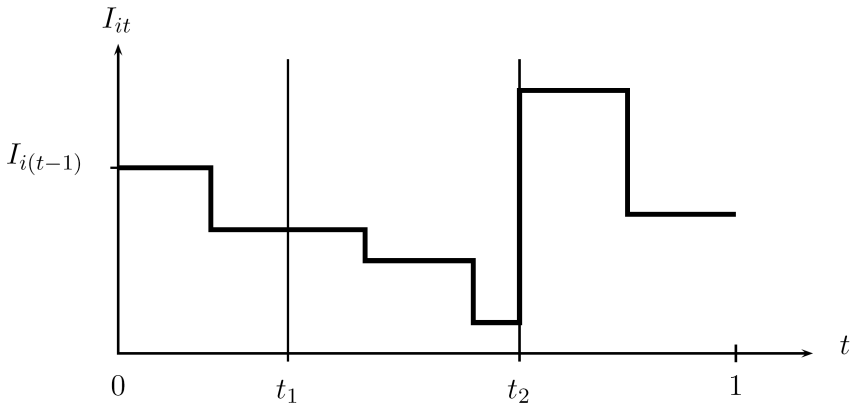

Figure 3: Change of the available stock of item i: t_1 denotes the start and t_2 the finish time of the production of item i.

every $i \notin \phi(m)$. Let $\nu(\cdot)$ denote the optimal value of the underlying optimization problem. The corresponding lot-sizing and scheduling problem reads:

$$\nu(F_B) = \min \sum_{i=1}^{N} \sum_{t=1}^{T} h_{it} \cdot I_{it} + \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{t=1}^{T} \sum_{m=1}^{M} c_{ij} \cdot T_{ijtm} \tag{6}$$

subject to

$$I_{it} - I_{i(t-1)} - X_{it} + \sum_{j \in \Gamma(i)} a_{ij} \cdot X_{jt} + E_{it} = 0 \qquad i,t \tag{7}$$

$$p_i \cdot X_{it} \leq \sum_{j \in \phi(m)} T_{jitm} + \alpha_{itm} \qquad i \in \phi(m), t, m \tag{8}$$

$$\sum_{i \in \phi(m)} \alpha_{itm} = 1 \qquad t, m \tag{9}$$

$$\sum_{j \in \phi(m)} T_{jitm} + \alpha_{itm} = \sum_{j \in \phi(m)} T_{ijtm} + \alpha_{i(t+1)m} \qquad i \in \phi(m), t, m \tag{10}$$

$$\sum_{j \in \phi(m)} T_{jitm} \leq 1 \qquad i \in \phi(m), t, m \tag{11}$$

$$\mu^s_{it} + p_i \cdot X_{it} + s_{ij} \cdot T_{ijtm} + T_{ijtm} - 1 - \alpha_{j(t+1)m} \leq \mu^s_{jt} \qquad i \in \phi(m), j \in \phi(m), t, m \tag{12}$$

$$\mu^s_{it} + p_i \cdot X_{it} + \alpha_{j(t+1)m} - 1 - \alpha_{jtm} \leq \mu^s_{jt} \qquad i \in \phi(m), j \in \phi(m), t, m \tag{13}$$

$$\mu^s_{it} + p_i \cdot X_{it} + \sum_{j \in \phi(m)} s_{ij} \cdot T_{ijtm} \leq 1 \qquad i \in \phi(m), t \tag{14}$$

$$(\mu^s_{it} + p_i \cdot X_{it}) - \mu^s_{jt} \leq 1 - W_{ijt} \qquad i,j \in \Gamma(i), t \tag{15}$$

$$\hat{X}_{ijt} \geq X_{jt} - \frac{1}{p_j} W_{ijt} \qquad i,j \in \Gamma(i), t \tag{16}$$

$$I_{i(t-1)} \geq \sum_{j \in \Gamma(i)} a_{ij} \hat{X}_{ijt} \qquad i,t \tag{17}$$

$$I_{it}, X_{it}, \hat{X}_{ijt}, \mu^s_{it} \geq 0; \alpha_{itm}, T_{ijtm}, W_{ijt} \in \{0,1\} \quad i,j,t,m. \tag{18}$$

Objective function (6) aims at minimizing the sum of sequence-dependent setup and holding

costs. Constraints (7) represent the inventory balances for both components and end-items. Constraints (8) guarantee that a product is produced only if the machine is set up for it. Note that we assume that the length of one period is one time unit and capacities are measured in time units. Therefore, in contrast to constraints (4), we can omit parameter G in this case. Each machine has to be set up for exactly one product at the beginning of each period from (9). Constraints (10) carry the setup configuration state of the machines into the next period and constraints (11) avoid that there is more than one setup for each item. Disconnected subtours are removed from any feasible solution by (12). It is easy to see that such constraints also cut off subtours that appear in the middle of the sequence (see e.g. Almada-Lobo et al., 2007 or Almada-Lobo et al., 2008 for details). These constraints also link the starting and finishing times of two batches produced one immediately after another. We do not impose the starting times of the first batch in any period to be zero, as there might be some idle time before production starts (as well as afterwards). Requirements (12) do not define the starting time of the last product to be produced on each machine, as they are non-active for such cases. We assume that for a cycle of size greater than one involving item j ($\alpha_{jtm} = \alpha_{j(t+1)m} = 1$), there occurs only one batch of j in the first position of the production sequence. In case of a path, requirements (13) determine the starting time of the last product of the sequence. Clearly, (13) are redundant for a cycle. Inequalities (12)–(14) ensure that the machines are used no longer than their available capacity. Constraints (15) ensure the coherency between variables W_{ijt} and the start and finish times of the batches of predecessors and successors. Note that $\mu^s_{it} + p_i \cdot X_{it}$ computes the production finish time of product i in period t. In case item j starts production before the batch of its predecessor i is completed in the same period (or in case there is not such batch), i.e. $W_{ijt} = 0$, constraints (16) and (17) force the quantity of item i required to produce j to be supplied solely from the initial stock in that period. If $W_{ijt} = 1$, then (16) and (17) are non-active, and (7) allow the gross requirements of item i to produce j to be matched from stock and from production in the same period.

We remark that the way depended batches are synchronized is motivated by Almeder and Almada-Lobo (2009) where the usage of a secondary resource is synchronized across parallel machines.

Next we show that formulation F_B really models the problem correctly. Let Q be the set of solutions satisfying constraints (7)–(15) and (18). Set Q admits solutions that have nonnegative inventory stock for every product at the end of every period (imposed by (7)), but potentially negative in-period stock. We show that constraints (16) and (17) avoid this case. Now consider subsets $S(i)$ of items in a given time period t defined as

$$S(i) = \{j \in \Gamma(i) : \mu^s_{jt} \leq \mu^s_{it} + p_i \cdot X_{it}\}.$$

The following lemma states that there are no negative in-period stocks (note that due to the batching assumption the condition $\sum_{j \in S(i)} a_{ij} \cdot X_{jt} > I_{i(t-1)}$ is equivalent to the statement of having negative in-period stock).

Lemma 1. *If $x^\star \in Q$ and $x^\star$ fulfills (16) and (17), then $\sum_{j \in S(i)} a_{ij} \cdot X_{jt} \leq I_{i(t-1)}$.*

Proof. Let $x^\star \in Q$. We first observe that if $j \in S(i)$, then $\mu^s_{jt} \leq \mu^s_{it} + p_i \cdot X_{it}$, which in turn from

(15) implies $W_{ijt} = 0$. Thus if $j \in S(i)$, from (16) we obtain $\hat{X}_{ijt} \geq X_{jt}$.

On the other hand, if $j \in \Gamma(i) \backslash S(i)$, then clearly $\hat{X}_{ijt} \geq 0$. Combining these two factors we derive

$$I_{i(t-1)} \geq \sum_{j \in \Gamma(i)} a_{ij} \cdot \hat{X}_{ijt} \geq \sum_{j \in S(i)} a_{ij} \cdot \hat{X}_{ijt} + \sum_{j \in \Gamma(i) \backslash S(i)} a_{ij} \cdot \hat{X}_{ijt} \geq \sum_{j \in S(i)} a_{ij} \cdot \hat{X}_{ijt} \geq \sum_{j \in S(i)} a_{ij} \cdot X_{jt}.$$

This establishes the lemma. □

In addition to showing that solutions have no negative in-period stock, we need to assert that no solutions to the problem are left out.

Lemma 2. *Any $x^\star \in Q$ satisfying $\sum_{j \in S(i)} a_{ij} \cdot X_{jt} \leq I_{i(t-1)}$ for every i and t (x^* has nonnegative in-period inventory) fulfills constraints (16) and (17).*

Proof. We show this by contradiction. We first note that in (16) and (17) we have the freedom of selecting $\hat{X}$. As a result we can assume that we have an underlying $\hat{X}_{ijt}$ satisfying (16) but not (17) for at least one item i' in a period t' since otherwise we can always increase $\hat{X}_{ijt}$ to satisfy (16), but possibly violating (17) even more. Thus we have

$$\hat{X}_{i'jt'} \geq X_{jt'} - \frac{1}{p_j} W_{i'jt'} \qquad j \in \Gamma(i')$$
$$I_{i'(t'-1)} < \sum_{j \in \Gamma(i')} a_{i'j} \hat{X}_{i'jt'}$$

Among all such $\hat{X}_{i'jt'}$, we pick the smallest one, which we denote by $\hat{X}^{\min}_{i'jt'}$. By choice we clearly have

$$I_{i'(t'-1)} < \sum_{j \in \Gamma(i')} a_{i'j} \hat{X}^{\min}_{i'jt'}. \tag{19}$$

According to the definition of $S(i)$, the minimal values satisfying (16) are defined by

$$\hat{X}^{\min}_{i'jt'} = \begin{cases} X_{jt'} & j \in S(i') \\ 0 & j \in \Gamma(i') \backslash S(i'). \end{cases}$$

Inequality (19) yields

$$I_{i'(t'-1)} < \sum_{j \in \Gamma(i')} a_{i'j} \hat{X}^{\min}_{i'jt'} = \sum_{j \in S(i')} a_{i'j} \hat{X}^{\min}_{i'jt'} + \sum_{j \in \Gamma(i') \backslash S(i')} a_{i'j} \hat{X}^{\min}_{i'jt'} = \sum_{j \in S(i')} a_{i'j} X_{jt'}.$$

This constraint indicates that there is negative in-period inventory and it contradicts our assumption that $\sum a_{i'j} \cdot X_{jt} \leq I_{i'(t'-1)}$. □

3.2 Lot-streaming Production (Non-batching)

In the previous model we assumed that components can only be used for the production of successors if the production batch is completely finished. If we relax this assumption, we allow to

use some already finished parts of the current production batch for the production of successors. For this case we assume the idealized situation that there are no transportation times between different production stages and that inventory levels increase and decrease continuously due to production and consumption of an item. In principle we have to ensure that during the whole period the inventory level is nonnegative. Instead of constraints (16) and (17) it is necessary to replace them with

$$I_{i(t-1)} + \min\left(X_{it}, \frac{1}{p_i}\left(\tau - \mu^s_{it}\right)^+\right) \geq \sum_{j\in\Gamma(i)} a_{ij} \min\left(X_{jt}, \frac{1}{p_j}\left(\tau - \mu^s_{jt}\right)^+\right) \quad i,t,\tau \in [0,1], \quad (20)$$

where $x^+ := \max(x, 0)$.

This set of constraints ensures that, for every time point τ in period t, the initial inventory plus the production of item i until τ is not less than the consumption of item i due to the production of successors. Note that the external demand E_{it} only occurs at the end of the period. Unfortunately there is an infinite number of such constraints. The next result asserts that only a finite number of τ values suffices.

Theorem 3. *It is sufficient and necessary to fulfill constraints (7) and (20) at the beginning of the production of item i ($\tau = \mu^s_{it}$) and at the end of the production of each successor $k \in \Gamma(i)$ ($\tau = \mu^s_{kt} + p_k X_{kt}$) in order to maintain a nonnegative inventory level throughout the whole period t.*

Proof. We fix item i. Let us first consider the case when there is no production of item i in period t ($X_{it} = 0$). If there is no production, the in-period inventory level of item i is non-increasing throughout the period. Hence it reaches the minimum at the end of the period. Since constraints (7) ensure that there is a nonnegative stock level at the end of each period, the in-period inventory level can never be negative. To show the other way around, because of (7), the following derivation shows that (20) is fulfilled for all τ:

$$I_{i(t-1)} + \min\left(X_{it}, \frac{1}{p_i}\left(\tau - \mu^s_{it}\right)^+\right) = I_{i(t-1)} = I_{it} - X_{it} + E_{it} + \sum_{j\in\Gamma(i)} a_{ij} X_{jt} \geq$$
$$\geq \sum_{j\in\Gamma(i)} a_{ij} X_{jt} \geq \sum_{j\in\Gamma(i)} a_{ij} \min\left(X_{jt}, \frac{1}{p_j}\left(\tau - \mu^s_{jt}\right)^+\right).$$

Now let us consider the case of a production of item i in period t ($X_{it} > 0$). We divide the period into three parts. Let t_1 define the start of the production of item i ($t_1 = \mu^s_{it}$), and t_2 the end of the production of item i ($t_2 = \mu^s_{it} + p_i X_{it}$). Furthermore, note that $\frac{1}{p_i}$ is the production rate of item i, $\frac{a_{ij}}{p_j}$ is the consumption rate of item i due to the production of successor j, and the in-period inventory level $I_{it}(\tau)$ is a piecewise linear function of τ.

Let first $\tau \in [0, t_1]$. During this period only consumption of item i might occur ($I_{it}(\tau)$ is non-increasing), so it is sufficient and necessary that at time t_1 the inventory level is nonnegative, which is equivalent to constraints (20) at $\tau = t_1 = \mu^s_{it}$.

Consider now $\tau \in [t_2, 1]$. After t_2 again only consumption might occur and the in-period inventory level is non-increasing ($I_{it}(\tau)$ is again non-increasing). For the the time span from t_2

to the end of the period it is sufficient and necessary to have nonnegative inventory at the end of the period. This is assured by constraints (7).

Finally, let $\tau \in [t_1, t_2]$. During the production period of item i between t_1 and t_2 the in-period inventory level may increase or decrease depending on the difference between the constant production rate $\frac{1}{p_i}$ and the consumption rate of all currently processed successors (see Figure 4). This consumption rate might vary depending on the number of successors produced at a specific time point. If we consider a specific successor k of item i, during the production of item k the consumption rate of item i is increased by $\frac{a_{ik}}{p_k}$. Since the consumption rate due to item k stays constant over the whole production period $[t_1, t_2]$ it is sufficient and necessary that at the end of this production period the in-period inventory level is nonnegative, i.e. $I_{it}(\tau) \geq 0$ for $\tau = \mu^s_{kt} + p_k X_{kt}$. We need to guarantee this for all k and thus $I_{it}(\tau) \geq 0$ for $\tau \in \{\mu^s_{kt} + p_k X_{kt} | k \in \Gamma(i) \text{ and } t_1 \leq \mu^s_{kt} + p_k X_{kt} \leq t_2\}$. Using the same argument as in the case of no production it can be shown that (20) for $\tau < t_1$ and $\tau > t_2$ are valid and not cutting off solutions with nonnegative inventory. Hence, (20) must hold for $\tau \in \{\mu^s_{kt} + p_k X_{kt} | k \in \Gamma(i)\}$. □

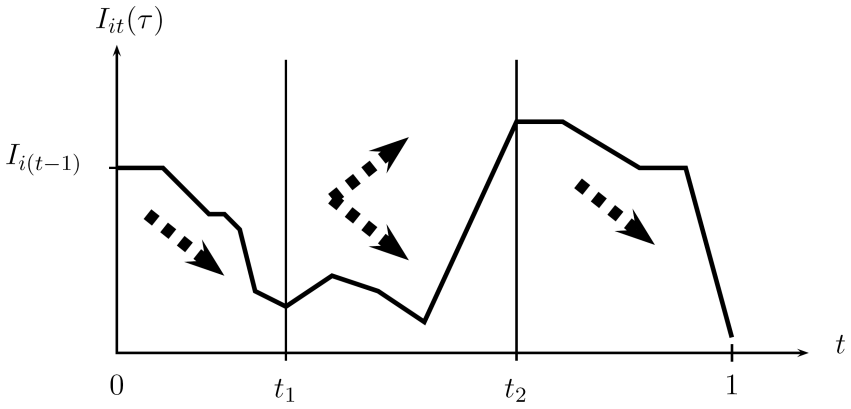

Figure 4: Example of the in-period inventory

From Theorem 3 we conclude that replacing (16) and (17) with a set of constraints avoiding negative stock levels at the beginning and end of the production of each item i for each successor k by

$$I_{i(t-1)} \geq \sum_{j \in \Gamma(i)} a_{ij} \min\left(X_{jt}, \frac{1}{p_j}\left(\mu^s_{it} - \mu^s_{jt}\right)^+\right) \qquad i, t \qquad (21)$$

$$I_{i(t-1)} + \min\left(X_{it}, \frac{1}{p_i}\left(\mu^s_{kt} + p_k X_{kt} - \mu^s_{it}\right)^+\right) \geq \sum_{j \in \Gamma(i)} a_{ij} \min\left(X_{jt}, \frac{1}{p_j}\left(\mu^s_{kt} + p_k X_{kt} - \mu^s_{jt}\right)^+\right) \qquad i, k \in \Gamma(i), t \qquad (22)$$

leads to a model that allows immediate use of produced items but maintains a nonnegative in-period inventory.

Hence, the model for the non-batching case reads:

$$\nu(F_{NB}) = \min \sum_{i=1}^{N}\sum_{t=1}^{T} h_{it}\cdot I_{it} + \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{t=1}^{T}\sum_{m=1}^{M} c_{ij}\cdot T_{ijtm}$$

$$\text{subject to } (7)-(15), (21), (22)$$

$$I_{it}, X_{it}, \mu^s_{it} \geq 0; \alpha_{itm}, T_{ijtm}, W_{ijt} \in \{0,1\} \qquad\qquad i,j,t,m$$

This is a nonlinear model that can be linearized. We show the underlying linearization of (21) and (22) in Appendix A.

We prove in the following theorem that F_B is a special case of F_{NB} and, as such, the optimal value of F_{NB} is at least as good as F_B's.

Theorem 4. *We have $v(F_B) \geq v(F_{NB})$.*

Proof. We demonstrate the statement by showing that adding constraints (16) and (17) to F_{NB} constraints (21) and (22) become redundant and can be dropped.

Consider $S(i) = \{j \in \Gamma(i) : \mu^s_{it} \geq \mu^s_{jt}\}$. Constraints (21) are then equivalent to $I_{i(t-1)} \geq \sum_{j\in S(i)} a_{ij} \min\left(X_{jt}, \frac{1}{p_j}\left(\mu^s_{it} - \mu^s_{jt}\right)\right)$. If $j \in S(i)$, then (15) imposes $W_{ijt} = 0$ and $\hat{X}_{ijt} \geq X_{jt}$. Using (16) and (17), we obtain

$$I_{i(t-1)} \geq \sum_{j\in\Gamma(i)} a_{ij}\hat{X}_{ijt} \geq \sum_{j\in S(i)} a_{ij}\cdot X_{jt} \geq \sum_{j\in S(i)} a_{ij}\min\left(X_{jt}, \frac{1}{p_j}\left(\mu^s_{it} - \mu^s_{jt}\right)\right) \quad i,t,$$

making constraint (21) redundant.

Let $Q(i,k)$, $k \in \Gamma(i)$ denote $Q(i,k) = \{j : j \in \Gamma(i) \text{ and } \mu^s_{kt} + p_k X_{kt} \geq \mu^s_{jt}\}$. The right hand-side of (22) can then be simplified to

$$\sum_{j\in Q(i,k)} a_{ij}\min\left(X_{jt}, \frac{1}{p_j}\left(\mu^s_{kt} + p_k X_{kt} - \mu^s_{jt}\right)\right).$$

Let us now distinguish two cases: $\mu^s_{kt} + p_k X_{kt} \leq \mu^s_{it}$ or $\mu^s_{kt} + p_k X_{kt} > \mu^s_{it}$.

We start by $\mu^s_{kt} + p_k X_{kt} \leq \mu^s_{it}$. Since $\mu^s_{kt} \leq \mu^s_{it} + p_i \cdot X_{it}$, from (15) clearly $W_{ikt} = 0$. Moreover, $W_{ijt} = 0$ for every j in $Q(i,k)$, which implies $\hat{X}_{ijt} \geq X_{jt}$. Thus,

$$I_{i(t-1)} \geq \sum_{j\in Q(i,k)} a_{ij}\cdot X_{jt} \geq \sum_{j\in Q(i,k)} a_{ij}\min\left(X_{jt}, \frac{1}{p_j}\left(\mu^s_{kt} + p_k X_{kt} - \mu^s_{jt}\right)\right) \quad i,k \in \Gamma(i), t.$$

Let now $\mu^s_{kt} + p_k X_{kt} > \mu^s_{it}$. We have to further consider two different scenarios.

If $\mu^s_{kt} + p_k X_{kt} < \mu^s_{it} + p_i X_{it}$, then similarly to the above case, it follows that $W_{ikt} = 0$ and

$W_{ijk} = 0$ for every j in $Q(i,k)$. In terms it yields

$$I_{i(t-1)} + \min\left(X_{it}, \frac{1}{p_i}\left(\mu^s_{kt} + p_k X_{kt} - \mu^s_{it}\right)\right) \geq I_{i(t-1)} \geq$$
$$\geq \sum_{j \in Q(i,k)} a_{ij} \cdot X_{jt} \geq \sum_{j \in Q(i,k)} a_{ij} \min\left(X_{jt}, \frac{1}{p_j}\left(\mu^s_{kt} + p_k X_{kt} - \mu^s_{jt}\right)\right),$$

making (22) redundant.

Finally let $\mu^s_{kt} + p_k X_{kt} \geq \mu^s_{it} + p_i X_{it}$. In this case (22) reads

$$I_{i(t-1)} + \min\left(X_{it}, \frac{1}{p_i}\left(\mu^s_{kt} + p_k X_{kt} - \mu^s_{it}\right)\right) = I_{i(t-1)} + X_{it} \geq$$
$$\geq \sum_{j \in Q(i,k)} a_{ij} \min\left(X_{jt}, \frac{1}{p_j}\left(\mu^s_{kt} + p_k X_{kt} - \mu^s_{jt}\right)\right).$$

It is easy to see that this condition is dominated by $I_{i(t-1)} + X_{it} \geq \sum_{j \in \Gamma(i)} a_{ij} \cdot X_{jt}$, which is derived from (7) given that I_{it} and E_{it} are non-negative. □

3.3 Extensions

An interesting extension is a combination of the batching and lot-streaming cases, i.e. the raw materials might be delivered continuously to the production process of item i, but the newly produced items are only available if the whole batch is finished. In such a case we have to ensure that the whole production of successors can be supplied from stock of item i until the production is finished. In order to achieve this we replace (22) by

$$I_{i(t-1)} \geq \sum_{j \in \Gamma(i)} a_{ij} \min\left(X_{jt}, p_j\left(\mu^s_{it} + p_i X_{it} - \mu^s_{jt}\right)^+\right) \quad i, t. \tag{23}$$

In (23) we have replaced the start time μ^s_{it} of the production of item i by the finish time $\mu^s_{it} + p_i X_{it}$.

The proposed models could be extended in several different directions in order to adapt them to other production scenarios. The option of using overtime to compensate for a lack of capacities is easily introduced by relaxing constraints (14). Furthermore, in constraints (8), (12), (13), and (16) a large number M must be introduced and the objective function must be extended by a penalty term for the use of overtime.

The synchronization model could also be extended for a parallel machine scenario where each item might be produced on several different machines. For that purpose it is necessary to differenciate the production amounts X_{it} and the starting times μ^s_{it} for the the different machines m, as well as the synchronization variables would have to be extended to $W_{ijtmm'}$ and $\hat{X}_{ijtmm'}$.

Both versions, the batching and lot-streaming cases, represent extreme situations, especially the continuous production case where it is possible to produce subsequent items in the BOM at the same time. If we want to model a situation where it is not necessary to wait for the whole batch to be finished before items can be retrieved and used for further production steps, we might introduce a minimum production lead time τ_i and substitute in constraints (21) and (22)

the starting time of item i with the term $\mu^s_{it} + \tau_i$.

4 Benders Decomposition

Large-scale instances of the proposed models cannot be efficiently solved by commercial IP solvers. For this reason we resort to decomposition approaches, namely Benders decomposition, Benders (1962). We focus on the batching case, but all other cases can be treated in a similar fashion.

To this end, let us introduce the following function, where $\overrightarrow{T} = (T_{ijtm})_{i,j,t,m}, \overrightarrow{W} = (W_{ijt})_{i,j,t}$, $\overrightarrow{\alpha} = (\alpha_{itm})_{i,t,m}$ are given:

$$f(\overrightarrow{T}, \overrightarrow{W}, \overrightarrow{\alpha}) = \min \sum_{i=1}^{N} \sum_{t=1}^{T} h_{it} \cdot I_{it} \tag{24}$$

$$I_{it} - I_{i(t-1)} - X_{it} + \sum_{j \in \Gamma(i)} a_{ij} \cdot X_{jt} = -E_{it} \qquad i, t \tag{25}$$

$$p_i \cdot X_{it} \leq \sum_{j \in \phi(m)} T_{jitm} + \alpha_{itm} \qquad i \in \phi(m), t, m \tag{26}$$

$$\mu^s_{it} - \mu^s_{jt} + p_i \cdot X_{it} \leq -s_{ij} \cdot T_{ijtm} - T_{ijtm} + 1 + \alpha_{j(t+1)m} \qquad (i,j) \in \phi(m), t, m \tag{27}$$

$$\mu^s_{it} + p_i \cdot X_{it} - \mu^s_{jt} \leq -\alpha_{j(t+1)m} + 1 + \alpha_{jtm} \qquad (i,j) \in \phi(m), t, m \tag{28}$$

$$\mu^s_{it} + p_i \cdot X_{it} \leq 1 - \sum_{j \in \phi(m)} s_{ij} \cdot T_{ijtm} \qquad i \in \phi(m), t \tag{29}$$

$$\mu^s_{it} - \mu^s_{jt} + p_i \cdot X_{it} \leq 1 - W_{ijt} \qquad i, j \in \Gamma(i), t \tag{30}$$

$$X_{jt} - \hat{X}_{ijt} \leq \frac{1}{p_j} W_{ijt} \qquad i, j \in \Gamma(i), t \tag{31}$$

$$\sum_{j \in \Gamma(i)} a_{ij} \hat{X}_{ijt} - I_{i(t-1)} \leq 0 \qquad i, t \tag{32}$$

$$I, X, \hat{X}, \mu \geq 0. \tag{33}$$

We note that evaluation of f corresponds to solving an LP. Then our batching model can be rewritten as

$$\nu(F_B) = \min \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{t=1}^{T} \sum_{m=1}^{M} c_{ij} \cdot T_{ijtm} + f(\overrightarrow{T}, \overrightarrow{W}, \overrightarrow{\alpha}) \tag{34}$$

$$\sum_{i \in \phi(m)} \alpha_{itm} = 1 \qquad t, m \tag{35}$$

$$\sum_{j \in \phi(m)} T_{jitm} + \alpha_{itm} = \sum_{j \in \phi(m)} T_{ijtm} + \alpha_{i(t+1)m} \qquad i \in \phi(m), t, m \tag{36}$$

$$\sum_{j \in \phi(m)} T_{jitm} \leq 1 \qquad i \in \phi(m), t, m \tag{37}$$

$$\alpha, T, W \text{ binary.} \tag{38}$$

We consider the dual polyhedron of the LP defining f. To this end, let Π be a dual vector corresponding to (25)-(32), respectively. For ease of notation we denote by $q(\overrightarrow{T}, \overrightarrow{W}, \overrightarrow{\alpha})$ the vector of all right-hand sides in the same LP. Let K be the finite set of all extreme vertices and J the finite set of all extreme rays of the dual polyhedron. Then by basic LP theory and the principles of Benders we have

$$\nu(F_B) = \min \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{t=1}^{T} \sum_{m=1}^{M} c_{ij} \cdot T_{ijtm} + s \tag{39}$$

$$\text{subject to (35)} - \text{(37)} \tag{40}$$

$$q(\overrightarrow{T}, \overrightarrow{W}, \overrightarrow{\alpha})\Pi - s \geq 0 \qquad \Pi \in K \tag{41}$$

$$q(\overrightarrow{T}, \overrightarrow{W}, \overrightarrow{\alpha})\Pi \leq 0 \qquad \Pi \in J \tag{42}$$

$$\overrightarrow{\alpha}, \overrightarrow{T}, \overrightarrow{W} \text{ binary.} \tag{43}$$

This reformulation has a lower number of variables, but many more rows. Indeed, the rows need to be generated dynamically on the fly. Algorithm 1 presents the basic row generation algorithm. (The exposition in Step 16 exploits the fact that $\nu(F_B) > 0$.)

1: Start with $K = J = \emptyset$.
2: **loop**
3: Solve (39)-(43) by considering the incumbent K, J.
4: **if** this mixed integer program is infeasible **then**
5: Exit, the original problem is infeasible.
6: **end if**
7: Let $\overrightarrow{T}^*, \overrightarrow{W}^*, \overrightarrow{\alpha}^*, s^*$ be the resulting optimal solution.
8: Solve the linear program $f(\overrightarrow{T}^*, \overrightarrow{W}^*, \overrightarrow{\alpha}^*)$.
9: **if** this linear program is feasible **then**
10: Let Π^* be an optimal dual solution.
11: Set $K = K \cup \{\Pi^*\}$.
12: **else**
13: Let Π^* be an extreme ray.
14: Set $J = J \cup \{\Pi^*\}$.
15: **end if**
16: **if** $|s^*| > 0$ and $\frac{s^* - f(\overrightarrow{T}^*, \overrightarrow{W}^*, \overrightarrow{\alpha}^*)}{|s^*|} \leq$ some relative tolerance **then**
17: Exit.
18: **end if**
19: **end loop**

Algorithm 1: The Benders algorithm

The problem (39)-(43) over a subset of K, J is called the restricted master problem (RMP). The Benders algorithm is known to exhibit slow convergence mostly due to degeneracy of the underlying LP behind f. We have tried several variants, some of them being new. Next we describe them one by one.

Convex Combinations This idea originates in Klabjan et al. (2000) in the context of column generation, but it was later used in a Benders framework by Smith (2004). The key concept is to consider prior solutions to the RMP and then to modify the evaluation of f to also optimize over best convex combination multipliers.

Let $(\overrightarrow{T}^{*i}, \overrightarrow{W}^{*i}, \overrightarrow{\alpha}^{*i})$ for $i = 1, \ldots, \kappa$ be optimal solutions in Step 7 over the last κ iterations with feasible f, where κ is a parameter. Instead of evaluating f, we evaluate

$$\min f(\sum_{i=1}^{\kappa} \lambda_i \overrightarrow{T}^{*i}, \sum_{i=1}^{\kappa} \lambda_i \overrightarrow{W}^{*i}, \sum_{i=1}^{\kappa} \lambda_i \overrightarrow{\alpha}^{*i}) + \sum_{k=1}^{\kappa} \lambda_k \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{t=1}^{T} \sum_{m=1}^{M} c_{ij} \cdot T^{*k}_{ijtm}$$

$$0 \leq \lambda \leq 1$$

$$\sum_{i=1}^{\kappa} \lambda_i = 1 \,.$$

It is not difficult to see that this is an LP with variables $\lambda, I, X, \hat{X}, \mu$.

After solving this optimization problem in a given iteration, we get the underlying dual values and add the corresponding cut. This convex combination problem is not solved in each iteration of Algorithm 1, but every κ iteration, i.e., once it is solved in an iteration, standard Benders is applied for the next κ iterations, which are then followed by a single iteration with convex combination.

Duplicate Variables A potential drawback of our reformulation is the fact that binary variables W are not present in constraints (35)-(37). To remedy this we can replicate these variables in both the RMP and in evaluating f. To carry out this idea, the RMP remains intact, but f is modified by considering W as a continuous variable. The resulting modified f, denoted by $\bar{f}$, becomes only a function of $\overrightarrow{T}$ and $\overrightarrow{\alpha}$, i.e., $\bar{f} = \bar{f}(\overrightarrow{T}, \overrightarrow{\alpha})$ and W's are continuous variables in $\bar{f}$. The dual prices are obtained as before and the underlying cuts do not change.

It can be shown that the optimal value of such a Benders approach is less than or equal to $\nu(F_B)$ and larger than or equal to the optimal value of the LP relaxation of $\nu(F_B)$.

Deviation Penalties In column generation there are known box-step type approaches to stabilize the underlying dual prices, Briant et al. (2008), du Merle et al. (1999). These approaches impose a deviation penalty in the subproblem to smoothly guide or stabilize the dual prices.

In Benders, the dual prices from the subproblem do not require stabilization (even though there are other issues with the dual prices such as degeneracy and dominance), however the solutions to RMP tend to bounce around. It thus seems plausible to speed up the algorithm by stabilizing the RMP solutions. To this end, let $\overrightarrow{T}', \overrightarrow{W}', \overrightarrow{\alpha}'$ be a past solution of RMP and R a penalty. We modify the objective function of RMP to

$$\sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{t=1}^{T} \sum_{m=1}^{M} c_{ij} \cdot T_{ijtm} + s + R\|(\overrightarrow{T}, \overrightarrow{W}, \overrightarrow{\alpha}) - (\overrightarrow{T}', \overrightarrow{W}', \overrightarrow{\alpha}')\| \,.$$

The challenge now becomes how to update R, which is initially set to a very large number, and $\overrightarrow{T}', \overrightarrow{W}', \overrightarrow{\alpha}'$. For the latter, we keep track of the minimum $h = \frac{s^* - f(\overrightarrow{T}^*, \overrightarrow{W}^*, \overrightarrow{\alpha}^*)}{|s^*|}$ observed so far. Let us denote this value by h_{min}. In a given iteration, if $h < h_{min}$, we reset $h_{min} = h$ and update $(\overrightarrow{T}', \overrightarrow{W}', \overrightarrow{\alpha}') = (\overrightarrow{T}^*, \overrightarrow{W}^*, \overrightarrow{\alpha}^*)$. If $h_{\min}$ is below the accepted tolerance level,

then we decrease R, e.g., $R = R/2$. If R becomes too small, we remove the penalty term altogether.

A known technique for generating Pareto non-dominant cuts is presented in Magnanti and Wong (1981). We did not evaluate it since, first, it is hard to find a core point, which is required in their approach, and second, our past experience has shown no benefits by using the suggested core approach.

To further strengthen the RMP, we add the following valid inequalities.

$$\begin{aligned} V_{jt} &\geq V_{it} + Q_m T_{ijmt} - (Q_m - 1) - Q_m \alpha_{imt} && m, i \in \phi(m), j, t \\ W_{ijt} &\leq \alpha_{imt} + \sum_{k \in \phi(m)} T_{kimt} && m, i \in \phi(m), j, t \\ W_{ijt} &\leq \alpha_{jmt} + \sum_{k \in \phi(m)} T_{kjmt} && m, i \in \phi(m), j, t \end{aligned} \tag{44}$$

Here Q_m is the total number of items that can be assigned to machine m and V_{it} are new auxiliary variables. The full derivation of these inequalities can be found in Almada-Lobo et al. (2007) with appropriate adjustments. Valid inequalities (44) follow from the fact that T variables either form a path or cycle in an appropriately defined network with α controlling if it is a path or cycle.

5 Computational Experiments

All computational experiments were coded in C++ using CPLEX 11.1 and Concert 2.6 on an Intel Pentium D with 3.2 GHz CPU and 4 GB of random access memory. In Section 5.1 we focus on the IP formulations F_B and F_{NB} and their comparison with MLCLSP. Secion 5.2 is dedicated to Benders.

5.1 Comparing with MLCLSP

All models in this section are solved by CPLEX. Furthermore, we used the well-known (l, S) valid inequalities adapted by Clark and Armentano (1995) for the multi-level case. We add them dynamically within the branch-and-cut framework of CPLEX.

5.1.1 MLCLSP with Zero Lead Time and no Setup Carryover

Considering that the proposed model incorporates more details than MLCLSP, we first evaluate if solutions derived from the traditional MLCLSP model would also be optimal or feasible for our approach. For this purpose we considered 4 classes of instances proposed by Tempelmeier and Derstroff (1996). Classes A and B are small instances with 10 items produced on 3 machines and a planning horizon of 4 periods. Class A instances consider setup cost but no setup times, whereas class B instances include also setup times. These classes are small enough that MLCLSP with zero lead time and no setup carryover can be solved easily with CPLEX within a second. Classes C and D are larger instances (40 items, 6 machines, 16 periods). For most of these instances optimal solutions of MLCLSP are unknown. We take the best known solutions for

these classes obtained by a heuristic method proposed in Almeder (2009). All classes contain instances with a general and assembly product structure, with different demand profiles and different levels of capacity utilization.

The classical MLCLSP does not consider setup carryover. To avoid setup carryover in our formulation we introduce additional items (one for each machine) with no demand and force that setup carryovers are only possible for those additional items. The setup times and costs into these items are set to zero. The production quantities obtained by solving the MLCLSP are fixed in our formulation (6)-(18) such that we only determine the production sequences and start and finish times. We let CPLEX run for at most ten minutes and classify the outcomes in three different categories:

Infeasible: CPLEX reports that the problem is infeasible, i.e. it is not possible to find a capacity feasible solution with the given production amount respecting synchronization conditions,

Feasible: CPLEX is able to find a capacity feasible solution with the given production quantities, which considers synchronization,

Unknown: CPLEX is neither able to generate a feasible solution, nor to prove infeasibility.

Table 3 shows the results of these tests. The numbers in brackets correspond to the number of tested solutions for each of the classes of test instances. We clearly observe that the vast majority of the solutions are infeasible. This further solidifies the need of our work. Since the batching constraint is more restrictive, fewer solutions are feasible under this assumption. But even without considering batching, which implies that predecessor and successor can be produced simultaneously, more than 90% of small instances are infeasible. It seems, while more than 60% of large instances are infeasible, that the zero lead-time assumption leads to solutions that cannot be implemented in practice. On the other hand, having only a few periods, a 1-period lead time is also not meaningful.

Table 3: Checking the feasibility of solutions of the classical MLCLSP model.

	batching assumption			lot-streaming assumption		
class	infeasible	feasible	unknown	infeasible	feasible	unknown
A (1500)	99.67%	0.33%	0.00%	94.80%	5.20%	0.00%
B (600)	99.67%	0.33%	0.00%	93.67%	6.33%	0.00%
C (599)	99.50%	0.00%	0.50%	86.64%	0.00%	13.36%
D (573)	98.08%	0.00%	1.92%	63.35%	0.00%	36.65%

In Table 4 we aggregate the results of the different classes according to various criteria. The number in brackets correspond to the number of tested solutions in each group. Differentiating by the product structure (general and assembly structure) and by the way items are assigned to machines (cyclic - the same machine is used for items on different levels; non-cyclic - only items at the same level or adjacent levels in the BOM are assigned to the same machines), we observe that most of the feasible solutions are obtained for problems with a general product structure and a non-cyclic machine-item relation (for the non batching assumption). Furthermore, it is obvious that capacity-tight problems are more likely to be infeasible, because free capacity usually means more flexibility for the scheduling problem. The capacity utilization is defined by

the total capacity requirement for production and setup for each machine divided by the total available capacity of that machine. Also, the setup cost level has an impact on the feasibility of the solution. High setup costs tend to generate solutions with large batches which are more difficult to schedule than smaller batches.

Table 4: Breakdown with respect to BOM structure, capacity utilization, and setup cost.

	batching assumption			lot-streaming assumption		
group	infeasible	feasible	unknown	infeasible	feasible	unknown
general-noncyclic (823)	99.51%	0.00%	0.49%	73.27%	12.76%	13.97%
general-cyclic (814)	99.02%	0.86%	0.12%	94.35%	1.35%	4.30%
assembly-noncyclic (811)	99.38%	0.00%	0.62%	91.86%	0.00%	8.14%
assembly-cyclic (824)	99.51%	0.00%	0.49%	91.02%	0.00%	8.98%
capacity util. 90% (641)	100.00%	0.00%	0.00%	97.97%	0.00%	2.03%
capacity util. 70% (657)	100.00%	0.00%	0.00%	87.67%	3.04%	9.28%
capacity util. 50% (660)	97.42%	1.06%	1.52%	80.45%	5.91%	13.64%
setup cost - low (653)	96.78%	1.07%	2.14%	81.62%	5.97%	12.40%
setup cost - med. (656)	100.00%	0.00%	0.00%	88.41%	3.51%	8.08%
setup cost - high (656)	100.00%	0.00%	0.00%	91.62%	2.13%	6.25%

5.1.2 MLCLSP with Zero Lead Time and Setup Carryover

In addition to the classical MLCLSP the proposed model considers setup carryover. We have applied the full model with synchronization to the test classes A and B in order to gain some insights about the computational behavior of the model. The results are compared to the solution of the classical MLCLSP with setup carryover. The solution of that model is a lower bound for the solution of model (6)-(18).

Table 5 shows the results. The columns report the number of solutions solved to optimality, the number of feasible solutions found by CPLEX plus the additional number of feasible solutions obtained using a relax-and-fix method, the number of infeasible solutions, and the number of solutions with an unknown status, i.e. we were neither able to find a feasible solution, nor could we prove infeasibility. We also report the average gap reported by CPLEX for all feasible solutions. The last column shows the average increase of the total costs of an optimal solution compared with the solution of the classical MLCLSP with setup carryover but no synchronization.

Table 5: Results for our model formulation.

class	optimal	feasible	infeasible	unknown	avg. gap	cost incr.
batching assumption						
A (1500)	120 (8.0%)	179+21 (13.3%)	680 (45.3%)	500 (33.3%)	9.86%	27.20%
B (600)	87 (14.5%)	84+17 (16.8%)	221 (36.8%)	191 (31.8%)	10.26%	25.40%
lot-streaming assumption						
A (1500)	378 (25.2%)	650+95 (49.7%)	215 (14.3%)	162 (10.8%)	9.80%	8.82%
B (600)	201 (33.5%)	264+32 (49.3%)	38 (6.3%)	65 (10.8%)	7.63%	8.53%

Unfortunately there are quite a few instances with unknown status. It is encouraging that the average gap for solutions classified as feasible is fairly good. The last column clearly shows that synchronization bears necessary additional costs, which are neglected by classical MLCLSP.

5.1.3 MLCLSP with One Period Lead Time and Setup Carryover

One way to overcome the feasibility problem of the MLCLSP is to consider a minimum lead time of one period. As aforementioned this lead time may cause a substantial increase of inventory. Tempelmeier and Buschkühl (2009) used an extension of the class B test instances for MLCLSP with lead time and setup carryover. In order to guarantee feasibility they added two additional periods at the beginning with no external demand (in total now 6 periods). Since the problems have a three level product structure it is possible to satisfy an end-item demand after the third period. We will denote the new class of instances as B6.

We solve the test instances using the classical MLCLSP without synchronization, but with one period lead time (as for example described by Tempelmeier and Buschkühl, 2009), as well as using our new model formulation with synchronization. Since the model with synchronization is much harder to solve, we apply a 10 minute run time limit to CPLEX. Table 6 compares the new solutions obtained by using the synchronization feature. The columns show the number of solutions solved to optimality, the number of feasible solutions, the number of instances where no solution has been obtained, the number of improved solutions and the average cost improvement (improvement of the upper bound) of the synchronization model compared with the MLCLSP with lead time.

For the batching case we can solve 114 instances to optimality, for 483 instances we find a feasible solution and only for 3 instances we are not able to compute a feasible solution within the specified run time. All those three instances have an assembly product structure, cyclic machine-item relations and a high capacity utilization. For the non-batching case we are able to solve more instances to optimality. For the non-batching case 140 instances can be solved to optimality and for all other instances we obtain feasible solutions.

The column labeled *UB impr.* denotes the average improvement of the objective function of the best solution found (optimal or feasible) compared to the optimal solution of the MLCLSP with lead time. Considering lead time in a big-bucket model leads to a clear overestimate of the optimal costs of at least 27.5% for batching and 37.6% for lot-streaming due to the increased inventory levels carried from one-period to the next one just to maintain feasibility of the solution.

5.2 Benders Algorithms

In this section we compare the enhancement strategies of the Benders algorithm. In addition, we compare Benders with CPLEX. All of the experiments were carried out on large instances of class D (same as those used before) and class 6 proposed by Tempelmeier and Buschkühl (2009) with a time limit of one hour for both Benders and CPLEX. Class 6 instances are of the same size and similar structure as class D instances. The difference is that class 6 instances were designed to be feasible for MLCLSP with lead time and setup carryover. Thus, they are also feasible for our model formulation. Class D instances were designed for the MLCLSP without lead time. Hence, some of them might be infeasible and backlogging is necessary. In order to avoid technical and development difficulties with a possibly infeasible f, we allow backlogging (see (25)), but we heavily penalize it.

Tables 7 and 8 summarize the most important findings. All of the values in both tables present the relative difference with respect to the objective value of the solution obtained by

Table 6: Comparing the MLCLSP with one period lead time with the new model with synchronization for the class B6 test instances.

	batching assumption					no batching assumption				
group	opt.	feas.	unk.	improved	UB impr.	opt.	feas.	unk.	improved	UB impr.
general-noncyclic (150)	39	111	0	144 (96%)	30.0%	84	66	0	150 (100%)	43.8%
general-cyclic (150)	49	101	0	146 (97%)	27.1%	21	129	0	149 (99%)	30.1%
assembly-noncyclic (150)	2	145	3	110 (73%)	21.3%	5	145	0	150 (100%)	36.1%
assembly-cyclic (150)	24	126	0	149 (99%)	30.0%	38	112	0	150 (100%)	40.4%
capacity util. 60% (120)	0	117	3	95 (79%)	17.6%	6	114	0	119 (99%)	30.6%
capacity util 47% (120)	22	98	0	111 (93%)	26.6%	31	89	0	120 (100%)	38.4%
capacity util. 33% (120)	58	62	0	120 (100%)	37.5%	66	54	0	120 (100%)	43.6%
setup cost - low (120)	6	114	0	120 (100%)	48.2%	14	106	0	120 (100%)	66.6%
setup cost - med. (120)	11	109	0	119 (99%)	27.6%	25	95	0	120 (100%)	41.3%
setup cost - high (120)	48	69	3	90 (75%)	9.9%	56	64	0	119 (99%)	15.0%
all (600)	**114**	**483**	**3**	**549 (92%)**	**27.5%**	**148**	**452**	**0**	**599 (100%)**	**37.6%**

CPLEX, e.g., a value of 4.38% indicates that CPLEX solutions are on average 4.38% better, while a value of -4.19% implies that the underlying Benders outperforms CPLEX by 4.19%. By *penalty*, *duplicate*, *convex* we denote the variant *Deviation Penalties*, *Duplicate Variables*, and *Convex Combinations*, respectively, from Section 4. Standard Benders as shown in Algorithm 1 is labeled as *standard*.

Table 7: Comparison of Benders for class 6 instances

	penalty	standard	duplicate	convex
general product structure	-39.32%	30.64%	18.40%	43.45%
assembly structure	-46.66%	4.27%	12.61%	5.26%
overall	-43.13%	15.06%	14.72%	20.88%

Table 8: Comparison of Benders for class D instances

	penalty	standard	duplicate	convex
general product structure	-4.19%	10.24%	10.76%	14.72%
assembly structure	4.38%	29.14%	1.81%	31.73%
overall	0.47%	20.55%	5.58%	24.44%

The most important message is that the penalty variant outperforms all other versions. For class 6 it fares drastically better than CPLEX, while for class D it is slightly outperformed by CPLEX. The remaining three variants are much worse. Due to the lower overall number of iterations, the convex variant results are not indicative since the convex combination step is very seldom executed. More than 90% of instances have less than 5 iterations of the algorithm. The duplicate variant performs better than standard Benders, however it is outperformed by CPLEX.

In Figure 5 we show the improvement in the objective value of the best solution for two typical instances. The trends exhibit the typical tailing off effect where substantial improvements are made in initial iterations and then the decreases tapper off in subsequent iterations.

6 Conclusions

Our extensive computational experiments clearly indicate that the solutions obtained by ML-CLSP are either infeasible or lead to excessive work-in-process (from 15% to 60% increase in overall cost). The need for synchronization is thus clearly established. The presented work is the first step in such a direction. While commercial solvers are capable of solving small to mid size instances and many larger instances, large-scale cases need further investigations. The presented Benders algorithms pave the inroads to tackling computational challenges. To this end, our brand new Benders variant with penalties is a significant contribution drastically improving the computational effort to find satisfactory solutions. While it has only be tested on the models presented herein, we see no reason for its above par performance in other problems. Our experiments assert that the idea of convex combinations does not yield advantages, but this conclusion

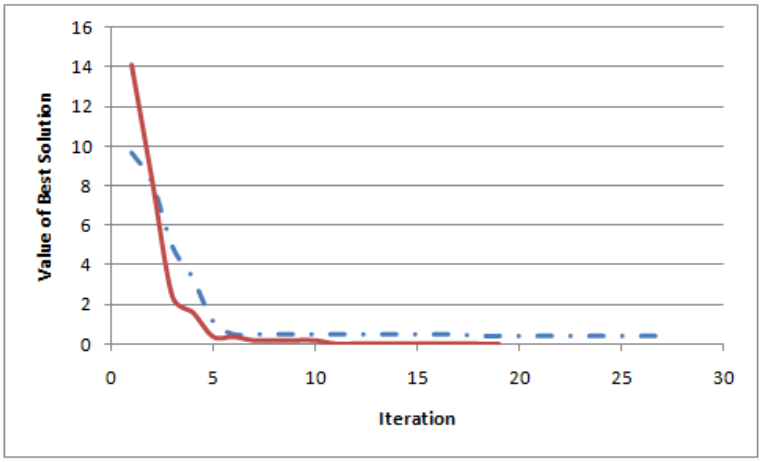

Figure 5: Objective value improvements of Benders with penalty.

is misleading due to the lower number of iterations. Separate experiments on other problems with a much larger number of iterations show a substantial improvement of convex combinations.

Unfortunately the presented models do not allow more than one setup in a time period. Let us now consider the example from Section 2.1 with modified demand for item 2, $d_{21} = 2$ and $d_{22} = 0$. The only feasible solution to this instance is shown in Figure 6. Such a solution enforces two batches of item 4 in period 1 on machine C, which is not a feasible solution to our model. Belvaux and Wolsey (2001) develop a lot-sizing model, which accounts for multiple setups of the same item. In their approach the production sequence is not unique anymore. Adapting it to our case would require to compute starting times for each batch and thus blowing up the model size. Fandel and Stammen-Hegene (2006) develop a general lot-sizing and scheduling approach for that problem but they end up with a highly complex non-linear model. Experiences with such model formulations in similar settings show that the computational tractability of these kind of models is worse (see e.g. Almeder and Almada-Lobo, 2009). It remains a challenge to develop a tractable model to address this issue.

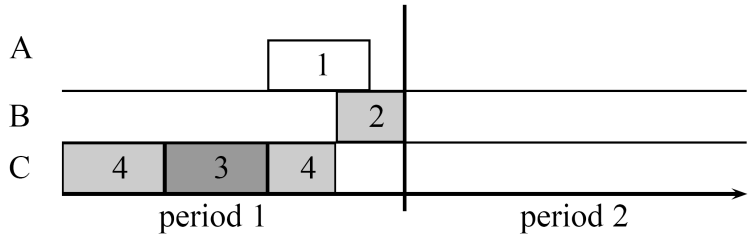

Figure 6: Example from Section 2.1 with changed demand for item 2.

References

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

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

Almeder, C., 2009. A hybrid optimization approach for multi-level capacitated lot-sizing problems. European Journal of Operational Research, to appear, doi:10.1016/j.ejor.2009.01.019.

Almeder, C., Almada-Lobo, B., 2009. Synchronization of scarce resources for a parallel machine lotsizing problem, submitted for publication.
(http://prolog.univie.ac.at/research/publications/downloads/Alm_2009_397.pdf)

Belvaux, G., Wolsey, L., 2001. Modelling practical lot-sizing problems as mixed-integer programs. Management Science 47, 993–1007.

Benders, J., 1962. Partitioning procedures for solving mixed-variables programming problems. Numerische Mathematik 4, 238–252.

Berretta, R., Franca, P., Armentano, V., 2005. Metaheuristic approaches for the multilevel resource-constraint lot-sizing problem with setup and lead times. Asia - Pacific Journal of Operational Research 22 (2), 261–286.

Billington, P., McClain, J., Thomas, L., 1983. Mathematical programming approach to capacity-constrained MRP systems: Review, formulation and problem reduction. Management Science 29, 1126–1141.

Bitran, G., Yanasse, H., 1982. Computational complexity of the capacitated lot size problem. Management Science 28, 1174–1186.

Briant, O., Lemaréchal, C., Meurdesoif, P., Michel, S., Perrot, N., Vanderbeck, F., 2008. Comparison of bundle and classical column generation. Mathematical Programming 113, 299–344.

Buschkühl, L., Sahling, F., Helber, S., Tempelmeier, H., 2008. Dynamic capacitated lot-sizing problems: a classification and review of solution approaches. OR Spectrum, to appear, doi:10.1007/s00291-008-0150-7.

Clark, A., Armentano, V., 1995. The application of valid inequalities to the multi-stage lot-sizing problem. Computers & Operations Research 22, 669 – 680.

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

du Merle, O., Villeneuve, D., Desrosiers, J., Hansen, P., 1999. Stabilized column generation. Discrete Mathematics 194, 229–237.

Fandel, G., Stammen-Hegene, C., 2006. Simultaneous lot sizing and scheduling for multi-product multi-level production. International Journal of Production Economics 194, 308–316.

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

Haase, K., 1996. Capacitated lot-sizing with sequence dependent setup costs. Operations Research Spektrum 18, 51–59.

Klabjan, D., Johnson, E., Nemhauser, G., 2000. A parallel primal-dual algorithm. Operations Research Letters 27, 47–55.

Maes, J., McClainLuk, J., Van Wassenhove, N., 1991. Multilevel capacitated lotsizing complexity and LP-based heuristics. European Journal of Operational Research 53, 131–148.

Magnanti, T., Wong, R., 1981. Accelerating Benders decomposition: Algorithmic enhancement and model selection criteria. Operations Research 29, 464–484.

Smith, B., 2004. Robust airline fleet assignment. Ph.D. thesis, Georgia Institute of Technology, Atlanta, GA.

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, 401–412.

Tempelmeier, H., Buschkühl, L., 2009. A heuristic for the dynamic multi-level capacitated lot-sizing problem with linked lotsizes for general product structures. OR Spectrum 31, 385–404.

Tempelmeier, H., Derstroff, M., 1996. A Lagrangean-based heuristic for dynamic multilevel multi-item constrained lotsizing with setup times. Management Science 42, 738–757.

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

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

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

A Linearized Model for the Lot-streaming Case

In order to linearize constraints (21) and (22), it is necessary to introduce two sets of binary and two sets of continuous decision variables for each time period t. We will use $\Gamma^{-1}(j)$ to denote the set of direct predecessors of item j and $\Gamma\left(\Gamma^{-1}(j)\right)$ to denote the set of items k which have a common direct predecessor with item j.

For each item i and $j \in \Gamma(i)$ we introduce $X^{(1)}_{ijt}$ to denote the production amount of item j, which has to be supplied from the stock of item i at the beginning of period t. Similarly we define for items j and $k \in \Gamma\left(\Gamma^{-1}(j)\right)$ variables $X^{(2)}_{jkt}$ to denote the production amount of item j, which is finished at the end of production of item k in period t.
For every item i and $j \in \Gamma(i)$ we define a binary variable

$$W^{(1)}_{ijt} \quad \begin{cases} 1 & X_{jt} \geq \frac{1}{p_j}\left(\mu^s_{it} - \mu^s_{jt}\right) \\ 0 & \text{otherwise} \end{cases}$$

and for every item j and $k \in \Gamma(j)$, and for items j and $k \in \Gamma\left(\Gamma^{-1}(j)\right)$ we define

$$W^{(2)}_{jkt} \quad \begin{cases} 1 & X_{jt} \geq \frac{1}{p_j}\left(\mu^s_{kt} + p_k X_{kt} - \mu^s_{jt}\right) \\ 0 & \text{otherwise.} \end{cases}$$

For large enough constant M, the constraints necessary to ensure the correct synchronization in the case of the lot-streaming assumptions are therefore

$$X_{jt} - \frac{1}{p_j}\left(\mu^s_{it} - \mu^s_{jt}\right) \leq M \cdot W^{(1)}_{ijt} \qquad i,j \in \Gamma(i), t \tag{45}$$

$$-X_{jt} + \frac{1}{p_j}\left(\mu^s_{it} - \mu^s_{jt}\right) \leq M \cdot \left(1 - W^{(1)}_{ijt}\right) \qquad i,j \in \Gamma(i), t \tag{46}$$

$$X_{jt} - \frac{1}{p_j}\left(\mu^s_{kt} + p_k X_{kt} - \mu^s_{jt}\right) \leq M \cdot W^{(2)}_{jkt} \qquad j,k \in \Gamma(j) \cup \Gamma\left(\Gamma^{-1}(j)\right), t \tag{47}$$

$$-X_{jt} + \frac{1}{p_j}\left(\mu^s_{kt} + p_k X_{kt} - \mu^s_{jt}\right) \leq M \cdot \left(1 - W^{(2)}_{jkt}\right) \qquad j,k \in \Gamma(j) \cup \Gamma\left(\Gamma^{-1}(j)\right), t \tag{48}$$

$$X^{(1)}_{ijt} \geq X_{jt} - \frac{1}{p_j} W^{(1)}_{ijt} \qquad i,j \in \Gamma(i), t \tag{49}$$

$$X^{(1)}_{ijt} \geq \frac{1}{p_j}\left(\mu^s_{it} - \mu^s_{jt}\right) - \frac{1}{p_j}\left(1 - W^{(1)}_{ijt}\right) \qquad i,j \in \Gamma(i), t \tag{50}$$

$$X^{(2)}_{jkt} \geq X_{jt} - \frac{1}{p_j} W^{(2)}_{jkt} \qquad j,k \in \Gamma\left(\Gamma^{-1}(j)\right), t \tag{51}$$

$$X^{(2)}_{jkt} \geq \frac{1}{p_j}\left(\mu^s_{kt} + p_k X_{kt} - \mu^s_{jt}\right) - \frac{1}{p_j} W^{(2)}_{jkt} \qquad j,k \in \Gamma\left(\Gamma^{-1}(j)\right), t \tag{52}$$

$$I_{i(t-1)} \geq \sum_{j \in \Gamma(i)} a_{ij} X^{(1)}_{ijt} \qquad i,t \tag{53}$$

$$I_{i(t-1)} + X_{it} + M \cdot W^{(2)}_{ikt} \geq \sum_{j \in \Gamma(i)} a_{ij} X^{(2)}_{jkt} \qquad i,k \in \Gamma(i), t \tag{54}$$

$$I_{i(t-1)} + \frac{1}{p_i}\left(\mu^s_{kt} + p_k X_{kt} - \mu^s_{it}\right) + M \cdot \left(1 - W^{(2)}_{ikt}\right) \geq \sum_{j \in \Gamma(i)} a_{ij} X^{(2)}_{jkt} \qquad i,k \in \Gamma(i), t. \tag{55}$$