On the Discrete Lot-Sizing and Scheduling Problem with Sequence-Dependent Changeover Time

Céline Gicquel, Michel Minoux, Yves Dallery

On the discrete lot-sizing and scheduling problem with sequence-dependent changeover times

C. Gicquel [a,1], M. Minoux [b] , Y. Dallery [a]

[a] *Ecole Centrale Paris, Laboratoire de Génie Industriel*
Grande Voie des Vignes, 92290 Chatenay-Malabry, France
[b] *Laboratoire d'Informatique de Paris 6*
4 place Jussieu, 75005 Paris, France

Abstract

We consider the discrete lot-sizing and scheduling problem with sequence-dependent changeover costs and times and propose to solve it as a mixed-integer program using a commercial solver. Our approach is based on the extension of an existing tight formulation for the case without changeover times. Computational results confirm the benefits of the proposed solution procedure.

Keywords: Lot-sizing, Sequence-dependent setup costs, Setup times, Mixed-integer programming

1 Introduction

A wide variety of models for production planning and inventory management has been investigated in operations research. Among them, capacitated lot-sizing models aim at determining the optimal timing and level of production complying with given capacity restrictions and such that demand for all products is satisfied without backlogging. Recent overviews on the lot-sizing literature can be found among others in [2] and [4].

In the present paper, the discrete lot-sizing and scheduling problem (DLSP) is considered. The DLSP relies on several basic assumptions (see e.g. [3]) :
- Demand for products is deterministic and time-varying.
- The production plan is established for a finite time horizon subdivided in several discrete periods.
- At most one item can be produced per period ("small bucket" model) and the facility processes either one product at full capacity or is completely idle ("all-or-nothing assumption").
- Costs to be minimized are the inventory holding costs and the changeover costs.

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

The DLSP has received much attention in the literature. Complexity results for this problem can be found in [8]. They show that the single machine multi-product case without setup times is NP-hard and that the problem of finding a feasible solution in the presence of sequence-independent setup times is NP-complete. We deal here with the extension of this problem to the case of sequence-dependent changeover costs and times. The DLSPSD under study in the present paper is thus NP-hard.

We now discuss into more detail specific contributions on the DLSPSD. [9] reformulate the problem as a travelling salesman problem with time windows and use a dynamic programming-based algorithm to solve it. [5] show the equivalence between the DLSPSD and the batch sequencing problem (BSP) and use a specific branch and bound type algorithm for solving the BSP to optimality. In both papers, the mixed-integer programming formulation proposed for the problem is weak and does not provide lower bounds good enough to solve the problem using a commercial solver (see results in section 3.2). However, as pointed out by [7], there is now a good knowledge about the "right" way to formulate many simple production planning sub-models as mixed-integer programs and, thanks to it, many practical production planning problems can be (approximately) solved using commercial solvers. To the best of our knowledge, these results

[1] Corresponding author.
Tel.: +33 141 13 18 04; fax: +33 141 13 12 72.
E-mail address: celine.gicquel@ecp.fr

have not yet been exploited to solve the DLSPSD. In the present paper, we attempt to close this gap by proposing a new tight formulation for this specific variant of the problem.

The purpose of this paper is thus to introduce a strengthened formulation for the DLSP with sequence-dependent changeover costs and times. This formulation is an extension of the formulation proposed by [11] for the DLSP with sequence-dependent changeover costs and zero changeover times. Thanks to this strengthened formulation, the lower bounds provided by the linear relaxation of the problem are significantly better, enabling a branch and bound type procedure to solve the problem more efficiently.

The paper is organized as follows. In section 2, we present the proposed tight formulation for the DLSPSD. In section 3, we discuss the results of some computational experiments carried out to evaluate it.

2   A tight formulation for the DLSPSD

We present a tight formulation for the DLSP with sequence-dependent changeover costs and times. This formulation is an extension of the formulation proposed by [11] for the DLSP with sequence-dependent changeover costs and zero changeover times. In the sequel, we denote it the DSLPSD2 formulation whereas we denote DLSPSD1 the formulation proposed by [9].

2.1   Basic formulation

We wish to optimize the production schedule for a set of N items over an horizon featuring T planning periods. A period is indexed by $t = 1, ..., T$, an item by $i = 0, .., N$. We agree to use item $i = 0$ to represent idle periods.
We use the following notation:
- d_{it}: demand (in units) for item i in period t.
- P_{it}: production capacity (in units per period) for item i in period t.
- h_i: holding costs per unit and period for item i.
- c_{ij}: changeover costs from item i to item j.
- T_{ij}: changeover time from item i to item j. T_{ij} is assumed to be an integer number of planning periods.
Decision variables are defined as follows:
- I_{it}: inventory level corresponding to item i at the end of period t.
- y_{it}: setup variables. y_{it} equals 1 if the resource is setup for production of item i in period t, and 0 otherwise.
- w_{ijt}: changeover cost variables. If $T_{ij} > 0$, w_{ijt} equals 1 during the first period of a changeover

from item i to item j, and 0 otherwise. If $T_{ij} = 0$, w_{ijt} equals 1 in the first period of production of j, and 0 otherwise.
- v_t: changeover time variables. v_t equals 1 during each period in which a changeover between two items occurs, and 0 otherwise.

In the mixed-integer formulation proposed in table 1, the objective (3) minimizes the sum of inventory holding costs and changeover costs. Note that, in the DLSPSD2 formulation, variables w_{iit} are introduced: $w_{iit} = 1$ means that the resource is setup for item i both in period $t - 1$ and in period t, i.e. that a production run for item i takes place over periods $t - 1$ and t.

Constraints (4) express the inventory balance. Together with constraints (8), they ensure that demand for each item is fulfilled without backlogging.

Equalities (5) and (6) link the setup variables with the changeover cost variables. (5) guarantee that item i can be produced in period $t - 1$ if and only if a changeover from i to another item j (possibly $j = i$) takes place at the beginning of period t. Similarly, (6) guarantee that item j can be produced in period t if and only if a changeover from another item i (possibly $i = j$) to item j begins early enough (i.e. in period $t - T_{ij}$) to be finished at the beginning of period t.

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

The binary character of the setup variables is represented by (9). (10) and (11) state the non-negativity of the changeover variables: observe, as pointed out by [1], that thanks to constraints (5)-(7) and (9), there is no need to define variables w_{ijt} and v_t as binary variables.

2.2   Valid inequalities

As shown in [11] for the case without changeover times, the DLSPSD2 formulation can be further strengthened through a family of valid inequalities adapted from the ones developed by [10]. We investigate here an extension of this idea to the case of positive changeover times and propose a family of valid inequalities for the problem (3)-(11).

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

2

periods for i in $\{t, ..., \tau\}$.

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

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

$$z_{jt} = \sum_{i:i \neq j} w_{ij,t-T_{ij}} \quad \forall j = 1...N, \forall t = 1...T \quad (1)$$

Equalities (1) state that the production of a new lot of item j begins in period t if and only if a changeover from another item $i \neq j$ starts "early enough" (i.e. in period $t - T_{ij}$) to be finished at the beginning of period t.

With this notation, we have:

Proposition 1. *The following inequations (2) are valid inequalities for the DLSP with sequence-dependent changeover costs and times:*

$$I_{it} \geq \sum_{q=1}^{p} \left(1 - y_{i,t+q} - \sum_{\tau=t+q+1}^{S_{i,D_{i,1,t+q}}} z_{i\tau} \right) \quad (2)$$

$$\forall i = 1...N, \forall t = 1...T, \forall p \in \{0...D_{i,t+1,T}\}$$

Proof. A sketch of proof is as follows. First note that $y_{i,t+q} + \sum_{\tau=t+q+1}^{S_{i,D_{i,1,t+q}}} z_{i\tau} = 0$ if and only if the resource is not setup for item i in period $t+q$ and no startup for i takes place between the period $t+q+1$ and the period where the qth demand after period t occurs, i.e. if and only if no production of item i is possible in the interval $\{t+q, ..., S_{i,D_{i,1,t+q}}\}$. In this case, the quantity needed to satisfy the qth demand after period t should be in stock at the end of period t. Thus we see that constraints (2) force an increase of the stock of item i at the end of period t by one for each index q for which no production occurs in the interval $\{t+q, ..., S_{i,D_{i,1,t+q}}\}$. A detailed proof of the validity of (2) can easily be derived from the above (see also [10]). $\square$

In the computational experiments to be presented in section 3, we use a standard cutting-plane generation method (see e.g. [6]) to strengthen the DSLPSD2 formulation by adding violated valid inequalities (2). The resulting improved formulation is denoted DLSPSD2*.

2.3 Remark

It is interesting to keep in mind that equalities (5)-(6) can be interpreted as flow conservation constraints in a network.

More precisely, we consider a graph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$. A node $v \in \mathcal{V}$ corresponds to a item-period pair (i,t). There is an oriented arc $a \in \mathcal{A}$ from node v_1 to node v_2 if and only if $v_1 = (i,t)$ and $v_2 = (j, t+T_{ij}+1)$. The setup variable y_{it} corresponds to the flow through node (i,t) and the changeover variable $w_{ij,t+1}$ corresponds to the flow between node (i,t) and node $(j, t + T_{ij} + 1)$. With this interpretation, a production sequence on the resource corresponds to a flow of a single unit through the network, starting from a node $(i,0)$ and arriving in a node (j,T). Thus equalities (5) and (6) can be seen as flow conservation constraints, stating that the flow through a node is equal to the sum of the flows on the arcs directed away from it or to the sum of the flows on the arcs directed toward it.

Due to the presence of positive changeover times, the structure of the graph $\mathcal{G}$ used here is seen to be different from the one used in [1]. Namely, when there are no changeover times, all arcs in graph $\mathcal{G}$ link pairs of nodes related to successive planning periods, which is not the case anymore with positive changeover times.

3 Computational results

In this section, we discuss the results of some computational experiments carried out to evaluate the DLSPSD2* formulation proposed in section 2.

3.1 Problem instance generation

We created several sets of randomly generated instances following the procedure described in [9] and [5]. The reader is referred to these references for more details. The generated instances differ with respect to the following characteristics:

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

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

For each possible combination of problem dimension and production capacity utilization, 5 problems were generated, resulting in $7 \times 6 \times 5 = 210$ instances. All tests were run on a Pentium 4 (2.8 Ghz) with 505 Mb of RAM, running under Windows XP. We used a standard MIP software (CPLEX 8.1.0) with the solver default settings to solve the problem, using either the DLSPSD1 formulation presented in [9] or the DSLPSD2 and DLSPSD2* formulations presented in section 2.

3.2 Comparison of DSLPSD1, DL-SPSD2 and DLSPSD2* formulations

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

Table 2 shows the results obtained with the DSLPSD1, DLSPSD2 and DLSPSD2* formulations for the sets A and B instances. For each formulation, we provide:
- *Variables* and *Constraints*: the average number of variables and constraints.
- $\#VI$: for the DLSPSD2* formulation, the average number of valid inequalities of type (2) added by the cutting-plane generation procedure.
- $\#Opt$: the number of instances out of the corresponding 30 instances that could be solved to optimality within 20 minutes of computation.
- Gap_0: the integrality gap, i.e. the relative difference between the lower bound provided by the linear relaxation of the problem and the value of an optimal solution. For the DLSPSD2* formulation, we consider the lower bound obtained after the cutting-plane generation procedure has stopped.
- $\#Nodes$: the number of nodes of the search tree explored before a guaranteed optimal solution is found or the computation time limit of 20 minutes is reached.
- CPU_{IP}: the computation time in seconds required to find a guaranteed optimal solution. If one could not be found, we use the computation time limit of 1200 seconds.
- Gap: the gap obtained after 20 minutes of computation between the best integer solution and the best lower bound found.
For performance measures Gap_0, $Nodes$, CPU_{IP} and Gap, we provide the average value (on the first line) and the minimum and maximum values (in brackets on the second line) for the considered set of randomly generated instances.

Table 2 shows that the results obtained with the DLSPSD2* formulation are much better than the ones obtained with the DLSPSD1 formulation. Namely, computation times are significantly reduced and more instances can be solved to optimality within the time limit while using the DLSPSD2* formulation.

This can be explained by the combination of two advantages:
1. The lower bounds provided by the linear relaxation of the DLSPSD2* formulation are much

better than the ones obtained with the DLSPSD1 formulation. This formulation improvement is achieved to a large extent thanks to the use of a small number of valid inequalities (2). This can be seen e.g. for set A instances for which the integrality gap is reduced in average from 84% with the DLSPSD1 formulation to 59% with the basic DLSPSD2 formulation and 4% with the strengthened DLSPSD2* formulation.
2. The MIP size (number of variables and constraints) is significantly reduced with the DLSPSD2* formulation. As a consequence, the time spent at each node of the branch and bound tree to solve the linear relaxation is shorter. This size reduction is explained by the fact that using the DLSPSD2* formulation, there is no need to introduce explicit changeover time variables v_{ijt} for each possible transition between pairs of items (i, j). Moreover, the set of equalities (5)-(6) are sufficient to ensure that positive changeover times between production runs for different items are respected. As a consequence, the numerous inequalities needed in the DLSPSD1 formulation to link changeover time variables to setup and changeover cost variables can be eliminated from the formulation.

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

3.3 Results with the MIP formulation DLSPSD2*

In order to further validate our approach, we carried out additional computational experiments. More precisely, we considered instances similar to the ones studied in [5] and [9], i.e. instances for which $(N, T) = \{(5, 60), (10, 60)\}$ (sets C-D). We also used 3 additional sets of larger instances for which $(N, T) = \{(15, 60), (10, 90), (15, 90)\}$ (sets E-G).

Table 3 displays the detailed results obtained with the DLSPSD2* formulation. We observe that:
- For medium size instances (sets C-D), 98% of the generated instances could be solved to optimality within 20 minutes of computation.
- For large size instances (sets E-G), 42% of the generated instances could be solved to optimality within the computation time limits. Moreover the average remaining gap obtained after 20 minutes of computation between the best integer solution and the best lower bound found is small (3.6% on average).

Thus, even if the proposed approach was implemented on a computer with more computing power, these results suggest the potential of the MIP modelling approach to solve instances larger than the

ones considered in [5] and [9].

References

[1] G. Belvaux, L.A. Wolsey, Modelling practical lot-sizing problems as mixed-integer programs, Management Science 47(7) (2001) 993-1007.

[2] A. Drexl, A. Kimms, Lot sizing and scheduling - Survey and extensions, European Journal of Operational Research 9 (1997) 221-235.

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

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

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

[6] G. Nemhauser, L.A. Wolsey, Integer and Combinatorial Optimization., John Wiley & Sons, 1988.

[7] Y. Pochet, L.A. Wolsey, Production planning by mixed integer programming, Springer Science, 2006.

[8] M. Salomon, L.G. Kroon, R. Kuik R, L.N. van Wassenhove, Some extensions of the discrete lotsizing and scheduling problem, Management Science 37(7) (1991) 801-812.

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

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

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

$$min \sum_{i=1}^{N} \sum_{t=1}^{T} h_i I_{it} + \sum_{i=0}^{N} \sum_{j=0}^{N} \sum_{t=1}^{T} c_{ij} w_{ijt} \tag{3}$$

$$I_{it} = I_{i,t-1} + P_{it} y_{it} - d_{it} \qquad \forall i = 1...N, \forall t = 1...T \tag{4}$$

$$y_{i,t-1} = \sum_{j=0}^{N} w_{ijt} \qquad \forall i = 0...N, \forall t = 1...T \tag{5}$$

$$y_{jt} = \sum_{i=0...N \text{ st } t-T_{ij}>0} w_{ij,t-T_{ij}} \qquad \forall j = 0...N, \forall t = 1...T \tag{6}$$

$$\sum_{i=0}^{N} y_{it} + v_t = 1 \qquad \forall t = 0...T \tag{7}$$

$$I_{it} \geq 0 \qquad \forall i = 1...N, \forall t = 1...T \tag{8}$$

$$y_{it} \in \{0,1\} \qquad \forall i = 0...N, \forall t = 0...T \tag{9}$$

$$w_{ijt} \in [0,1] \qquad \forall i = 0...N, \forall j = 0...N, \forall t = 1...T \tag{10}$$

$$v_t \in [0,1] \qquad \forall t = 0...T \tag{11}$$

Table 1: Mixed integer formulation DLSPSD2

MIP formulation	set A			set B		
	DLSPSD1	DLSPSD2	DLSPSD2*	DLSPSD1	DLSPSD2	DLSPSD2*
Variables	1167	960	960	7880	5720	5720
Constraints	2392	361	361	19946	1321	1321
#VI	_	_	124	_	_	479
#Opt	27	30	30	0	0	30
$\text{Gap}_0(\%)$	84	59	4	92	68	5
	[65;95]	[47;69]	[0;12]	[80;96]	[57;73]	[0;13]
#Nodes	49362	738	3	2497	35497	100
	[16;191523]	[11;2780]	[0;19]	[391;4969]	[24705;52968]	[2;319]
$\text{CPU}_{\text{IP}}(s)$	343	3	1	1200	1200	30
	[3;1200]	[0;8]	[0;3]	[1200;1200]	[1200;1200]	[3;100]
Gap (%)	2	0	0	81	33	0
	[0;29]	[0;0]	[0;0]	[60;91]	[11;44]	[0;0]

Table 2: Results for set A and B instances

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

Table 3: Results for set C-G instances obtained with the DLSPSD2* formulation