Multi-product lot-sizing and scheduling on unrelated parallel machines

Alexandre Dolgui[1]

Scientific Methods for Industrial Management Department (MSGI), Industrial Engineering and Computer Science Division (G21), Ecole des Mines de Saint Etienne, 158, cours Fauriel, 42023 Saint Etienne Cedex 2, France, E-mail: dolgui@emse.fr

Anton V. Eremeev

Omsk Branch of Sobolev Institute of Mathematics SB RAS, 13 Pevtsov street, 644099 Omsk, Russia, E-mail: eremeev@ofim.oscsbras.ru

Mikhail Y. Kovalyov

Faculty of Economics, Belarusian State University, and United Institute of Informatics Problems, National Academy of Sciences of Belarus, Nezavisimosti 4, 220030 Minsk, Belarus, E-mail: koval@newman.bas-net.by

Abstract

We study a problem of optimal scheduling and lot-sizing a number of products on m unrelated parallel machines to satisfy given demands. A sequence dependent setup time is required between lots of different products. The products are assumed to be all continuously divisible or all discrete. The criterion is to minimize the time, at which all the demands are satisfied, $C_{\max}$, or the maximum lateness of the product completion times from the given due dates, $L_{\max}$. The problem is motivated by the real-life scheduling applications in multi-product plants. We derive properties of optimal solutions, NP-hardness proofs, enumeration and dynamic programming algorithms for various special cases of the problem. The major contribution is an NP-hardness proof and pseudo-polynomial algorithms linear in m for the case, in which the number of products is a given constant. The results can be adapted for solving a production line design problem.

Keywords: Scheduling; Lot-sizing; Parallel machines; Production line design.

[1]Corresponding author

1 Introduction

There are m unrelated parallel machines, which are used for manufacturing n products in lots. The products can be either all continuously divisible, or all discrete. A lot is the maximal quantity of the same product, which is manufactured on the same machine with no inserted quantity of another product. Each lot is preceded by a sequence dependent setup time. The size of a lot is the quantity of the product contained in it. In the continuous case, it is a positive real number, and in the discrete case, it is a positive integer number. The following parameters are given for each product i:

D_i - a demand (at least this quantity of product i should be manufactured);

B_i - an upper bound on total production (at most this quantity of product i should be manufactured), $B_i \geq D_i$;

d_i - a due date, $d_i \geq 0$;

M_i - a subset of eligible machines (machines from the set $\{1, \ldots, m\} \backslash M_i$ cannot be used for manufacturing product i), $M_i \neq \phi$;

p_{li} - a per unit processing requirement for product i on machine l (it is required $p_{li} \cdot x$ time units for machine l to produce x units of product i), $l \in M_i$;

q^0_{li} - a lower bound on the size of a lot on machine l (a lot of a size $x < q^0_{lj}$ is not allowed on machine l), $l \in M_i$;

s_{lij} - a setup time required to switch from processing a lot of product i to a lot of product j, $j \neq i$, on machine l, $l \in M_i \cap M_j$;

s_{l0i} - a setup time required to start processing a lot of product i, if it is manufactured first on machine l, $l \in M_i$.

Let N_l denote the set of products eligible for machine l, i.e., $N_l := \{i \mid l \in M_i,\ i = 1, \ldots, n\}$, and let $n_l = |N_l|$, $l = 1, \ldots, m$. Denote $n_{\max} = \max\{n_l \mid l = 1, \ldots, m\}$.

We assume that all the numerical input parameters are non-negative integer numbers. A schedule specifies decision variables, which are product lots (their sizes), assignment of the lots to the machines, and their sequences on the machines.

We consider two objective functions to be minimized: the makespan, $C_{\max} = \max\{C_i | i = 1, \ldots, n\}$, and the maximum lateness, $L_{\max} = \max\{C_i - d_i | i = 1, \ldots, n\}$, where C_i is the time when the last unit of product i has been manufactured. Following the traditional three-field notation for scheduling problems, see Graham et al. [8], we denote our problem as $R|s_{lij}, \beta|\gamma$, where $\beta \in \{cntn, dscr\}$ specifies continuous and discrete cases, respectively, and $\gamma \in \{C_{\max}, L_{\max}\}$ specifies the objective function to be minimized. If the number of machines m is a given constant, then the descriptor Rm will be used instead of R.

Note that the formulated problem is solvable if and only if $B_i \geq \min\{q_{li}^0 \mid l \in M_i\}$ for all $i = 1, \ldots, n$. Therefore, we assume without loss of generality that $B_i \geq q_{li}^0,\ l \in M_i,\ i = 1, \ldots, n$.

Any of the four versions of the problem $R1|s_{lij}, \beta|\gamma$, $\beta \in \{cntn, dscr\}$, $\gamma \in \{C_{\max}, L_{\max}\}$, is *NPO-complete* (see Ausiello et al. [3] for definition) because it contains the problem *Hamiltonian Path of Minimum Weight* as a subproblem, and the latter problem is polynomially equivalent to the *Travelling Salesman Problem (TSP)*, which is NPO-complete (Orponen and Mannila [14]). It follows that the problem $R1|s_{lij}, \beta|\gamma$ cannot be approximated with any constant or polynomial factor of the optimum in polynomial time, unless $\mathcal{P} = \mathcal{NP}$.

An important special case appears if the setup times satisfy the *triangle inequality*:

$$s_{lij} + s_{ljk} \geq s_{lik}, \quad i = 0, 1, \ldots, n,\ j = 1, \ldots, n,\ k = 1, \ldots, n,\ l \in M_i \cap M_j \cap M_k. \qquad (1)$$

We denote this special case by placing symbol Δ in front of s_{lij} in the second field of the three-field notation. Papadimitriou and Vempala [15] showed that TSP with asymmetric arc lengths and the triangle inequality satisfied cannot be approximated better than $\frac{220}{219}$ times the optimum in polynomial time. Therefore, the non-approximability bound of $\frac{220}{219}$ applies for $R1|\Delta s_{lij}, \beta|\gamma$, $\beta \in \{cntn, dscr\}$, $\gamma \in \{C_{\max}, L_{\max}\}$.

Our primary interest in the problem $R|s_{lij}, \beta|\gamma$ stems from the medium-range production scheduling applications in multi-product chemical plants (see, e.g., Bitran and Gilbert [4], Lin et al. [12], and Shaik et al. [20]). Other applications of this model can be found in metal production in foundries (dos Santos-Meza, dos Santos and Arenales [2], and de Araujo, Arenales and Clark [1]), and textile industry (Silva and Magalhaes [18], and Taner et al. [22]). In these large-scale problems, the efficient utilization of the critical production units constitutes the main source of complexity. Our problem can be applied for optimal scheduling of the critical production units in situations where other units (e.g. feed transfer, storage and final product filling units in chemical plants) are not the bottleneck for the whole

system.

Problem $R|s_{lij},\beta|\gamma$ belongs to a class of problems combining scheduling with batching or lot-sizing. Surveys of this line of research are given by Potts and Van Wassenhove [17], Potts and Kovalyov [16] and Zhu and Wilhelm [24]. The most closely related problems were studied by Monma and Potts [13] and Brucker et al. [5]. The difference is that Monma and Potts considered identical machines, the triangle inequality case, and assumed that each product i consists of D_i different items having their own processing times and due dates. Notice that the latter assumption implies that the length of the input of their problem is $O(mn^2 + \sum_{i=1}^{n} D_i)$, while it is $O(mn^2)$ for our problem. Brucker et al. considered sequence independent setup times. Both papers did not study the continuous case.

The rest of the paper is organized as follows. In Section 2, solution procedures are presented for the triangle inequality case. They are combinations of enumeration, dynamic programming and linear programming techniques. In Section 3, the case of a given number of products is studied. The problem is proved NP-hard even if there are $n = 2$ products. Dynamic programming algorithms are developed for the discrete case, which are linear in m and exponential in n. An application of the obtained results for a production line design problem is discussed in Section 4. The paper concludes with a summary of the results and suggestions for future research.

2 The triangle inequality case

In this section we assume that the setup times satisfy the triangle inequality (1). A lot shifting technique can be used to show that there exists an optimal solution for the problem $R|\Delta s_{lij},\beta|\gamma$, $\beta \in \{cntn, dscr\}$, $\gamma \in \{C_{\max}, L_{\max}\}$, in which each product is produced in at most one lot on each machine. In the rest of this section, we consider only such schedules and assume that any schedule is fully specified if for each machine we are given a set of products to be manufactured, their sequence and the corresponding lot sizes.

Let us introduce an $m \times n$ *allocation matrix* $Y = ||y_{li}||$ such that

$$y_{li} = \begin{cases} 1, & \text{if product } i \text{ is manufactured on machine } l, \\ 0, & \text{otherwise.} \end{cases}$$

We call an allocation matrix Y *feasible* if $\{l \mid y_{li} = 1\} \subseteq M_i$ and $\sum_{l=1}^{m} y_{li} \geq 1$ for $i = 1, \ldots, n$. The total number of feasible allocation matrices is $O\left(\Pi_{l=1}^{m} 2^{n_l}\right) = O\left(2^{mn_{\max}}\right)$. Given a feasible allocation matrix Y, let $S(l, Y)$ denote the set of products allocated to machine l. The

matrix Y also induces a set $P(Y,l)$ of product permutations *consistent* with Y for each machine l:

$$P(Y,l) = \left\{(i_1^{(l)},\ldots,i_{k_l}^{(l)}) \mid i_1^{(l)},\ldots,i_{k_l}^{(l)} \in S(l,Y),\ k_l = |S(l,Y)|\right\},\quad l=1,\ldots,m.$$

Given permutation $\pi^{(l)} = (i_1^{(l)},\ldots,i_{k_l}^{(l)}) \in P(Y,l)$, the total setup time on machine l can be calculated as:

$$t(\pi^{(l)},l) := s_{l0i_1^{(l)}} + \sum_{j=1}^{k_l-1} s_{li_j^{(l)}i_{j+1}^{(l)}}.$$

A schedule is fully specified if we are given an allocation matrix Y, permutations $\pi^{(l)} \in P(Y,l)$, $l=1,\ldots,m$, and an $m\times n$ matrix of lot sizes $X=||x_{li}||$, which is consistent with the allocation matrix Y:

$$q_{li}^0 y_{li} \le x_{li},\quad l\in M_i,\quad i=1,\ldots,n,\quad D_i \le \sum_{l=1}^{m} x_{li} \le B_i,\quad i=1,\ldots,n.$$

Here x_{li} is the size of the lot of product i allocated to machine l. The total number of $(m+1)$-tuples $(Y,\pi^{(1)},\ldots,\pi^{(m)})$, where $\pi^{(l)} \in P(Y,l)$, $l=1,\ldots,m$, is equal to $O\left(\Pi_{l=1}^m (2^{n_l} n_l!)\right) = O\left(2^{mn_{\max}}(n_{\max}!)^m\right)$.

The problem $R|\Delta s_{lij},\beta|\gamma$, $\beta\in\{cntn,dscr\}$, $\gamma\in\{C_{\max},L_{\max}\}$, can be solved by the following two-stage procedure. In the first stage, a complete enumeration of all feasible allocations Y, and given Y, all m-tuples of permutations $(\pi^{(1)},\ldots,\pi^{(m)})$, $\pi^{(l)}\in P(Y,l)$, $l=1,\ldots,m$, is carried out. In the second stage, for each relevant $(m+1)$-tuple $(Y,\pi^{(1)},\ldots,\pi^{(m)})$, a lot-sizing subproblem is formulated as a linear program with $O(mn)$ variables. For problem $R|\Delta s_{lij},\beta|C_{\max}$, the lot-sizing subproblem is

$$\text{Minimize}\ \ C_{\max},\ \text{subject to} \tag{2}$$

$$t(\pi^{(l)},l) + \sum_{i\in N_l} p_{li}x_{li} \le C_{\max},\ l=1,\ldots,m, \tag{3}$$

$$D_i \le \sum_{l\in M_i} x_{li} \le B_i,\ i=1,\ldots,n, \tag{4}$$

$$q_{li}^0 y_{li} \le x_{li} \le D_i y_{li},\ l=1,\ldots,m,\ i=1,\ldots,n. \tag{5}$$

For problem $R|\Delta s_{lij},\beta|L_{\max}$, the lot-sizing subproblem is

$$\text{Minimize}\ \ L_{\max},\ \text{subject to (4)-(5) and}$$

$$s_{l0i_1^{(l)}} + \sum_{j=1}^{k-1} s_{li_j^{(l)}i_{j+1}^{(l)}} + \sum_{j=1}^{k} p_{li_j^{(l)}}x_{li_j^{(l)}} - d_{i_k^{(l)}} \le L_{\max},\ k=1,\ldots,k_l,\ l=1,\ldots,m,$$

where $(i_1^{(l)},\ldots,i_{k_l}^{(l)}) = \pi^{(l)}$. The variables are $C_{\max}$, $L_{\max}$ and x_{li}, $l=1,\ldots,m$, $i=1,\ldots,n$. They are restricted to integer and rational numbers if $\beta=dscr$ and $\beta=cntn$, respectively.

In the continuous case, the lot-sizing subproblems can be solved in polynomial time by the ellipsoid method of Shor [21] and Khachiyan [10] or the strongly polynomial time algorithm of Vavasis and Ye [23]. In the discrete case, the algorithm of Lenstra [11] can be used, which is polynomial if the number of variables, $O(mn)$, is a constant. Thus, the problem $R|\Delta s_{lij}, \beta|\gamma$, $\beta \in \{cntn, dscr\}$, $\gamma \in \{C_{\max}, L_{\max}\}$, can be solved in $O(\tau_\beta 2^{mn_{\max}}(n_{\max}!)^m)$ time, where τ_β is the running time of the corresponding linear programming ($\beta = cntn$) or integer linear programming ($\beta = dscr$) algorithm.

The above two-stage solution procedure can be adjusted for the following case, in which the triangle inequality is violated. Assume that the minimal lot sizes incur sufficiently long *minimal lot processing times* $p^0_{li} := q^0_{li} p_{li}$ such that, though the triangle inequality (1) is violated, the following inequalities are satisfied:

$$s_{lij} + s_{ljk} + p^0_{lk} \geq s_{lik},\ l \in M_i \cap M_j \cap M_k,\ i = 0, 1, \ldots, n,\ j = 1, \ldots, n,\ k = 1, \ldots, n. \quad (6)$$

In the algorithm for this case, we add p^0_{li} to the corresponding setup times and, for each feasible allocation matrix Y, deduct quantities $q^0_{li} y_{li}$ from the corresponding demands, thus obtaining an equivalent situation with the new demands $\tilde{D}_i(Y)$, new upper bounds $\tilde{B}_i(Y)$, new (zero) minimal lot sizes and new setup times $\tilde{s}_{lij}$ for which the triangle inequality is satisfied:

$$\tilde{D}_i(Y) := D_i - \sum_{l=1}^{m} q^0_{li} y_{li},\ \tilde{B}_i(Y) := B_i - \sum_{l=1}^{m} q^0_{li} y_{li},\ i = 1, \ldots, n,$$

$$\tilde{s}_{lij} := s_{lij} + p^0_{li},\ l \in M_i \cap M_j,\ i = 1, \ldots, n,\ j = 1, \ldots, n,$$

$$\tilde{q}^0_{li} := 0,\ l \in M_i,\ i = 1, \ldots, n.$$

We stress that the new values of $\tilde{s}_{lij}$ and $\tilde{q}^0_{li}$ do not depend on matrix Y, while the new demands and upper bounds depend on it. An optimal solution found by the modified algorithm for the problem $R|s_{lij}, \beta|\gamma$, $\beta \in \{cntn, dscr\}$, $\gamma \in \{C_{\max}, L_{\max}\}$, with the inequalities (6) satisfied can be easily transformed into an optimal solution of the original problem.

The running time $O(\tau_\beta 2^{mn_{\max}}(n_{\max}!)^m)$ of the suggested algorithms can be reduced by about a factor of $(n_{\max}!)^m$ in the case of $C_{\max}$ criterion by using a dynamic programming algorithm, which is similar to the well-known algorithm of Held and Karp [9] developed for the TSP with triangle inequality. A description of such an algorithm is given in the following theorem.

Theorem 1 *The problem $R|\Delta s_{lij}, \beta|C_{\max}$, $\beta \in \{cntn, dscr\}$, is solvable in $O(mn^2_{\max} 2^{n_{\max}} + \tau_\beta 2^{mn_{\max}})$ time.*

Proof. Recall that $S(l, Y)$ denotes the set of products allocated to machine l in accordance with the allocation matrix Y. Let $\pi^*(l, Y)$ denote an optimal permutation of products of the set $S(l, Y)$. Permutation $\pi^*(l, Y)$ minimizes the total setup time $t(\pi, l)$ on the set of permutations $\pi = (i_1, i_2, \ldots, i_{|S(l,Y)|})$, $i_j \in S(l, Y)$, $j = 1, \ldots, |S(l, Y)|$, $i_0 = 0$. We now show how to construct the optimal values $T^*(l, Y) := t(\pi^*(l, Y), l)$ for *all* feasible allocation matrices Y and $l = 1, \ldots, m$. We will use a dynamic programming algorithm, which is similar to the algorithm of Held and Karp [9].

In our algorithm values $T(l, S, i)$ are recursively computed, where $T(l, S, i)$ is the minimum total setup time for processing a set of products $S \subseteq N_l$ on machine l, $l = 1, \ldots, m$, provided that product $i \in S$ is processed last. The initialization is $T(l, S, i) = s_{l0i}$ for $S = \{i\}$, $i \in N_l$, $l = 1, \ldots, m$, and the recursion for $S \subseteq N_l$, $|S| = 2, 3, \ldots, n_l$, is given by

$$T(l, S, i) = \min_{j \in S \setminus \{i\}} \Big\{ T(l, S \setminus \{i\}, j) + s_{lji} \Big\}.$$

For any set $S(l, Y)$, the minimum total setup time $T^*(l, Y)$ can be calculated as $T^*(l, Y) = \min_{i \in S(l,Y)} \{T(l, S(l, Y), i)\}$ in $O(|S(l, Y)|)$ time. All the relevant values $T^*(l, Y)$ can be computed in $O\left(\sum_{l=1}^{m} \sum_{k=0}^{n_l} k(k-1)\binom{n_l}{k}\right) = O(mn_{\max}^2 2^{n_{\max}})$ time. Given Y and $T^*(l, Y)$, $l = 1, \ldots, m$, an optimal lot-sizing decision can be made in $O(\tau_\beta)$ time. Then the optimal $C_{\max}$ value, $C^*_{\max}$, can be determined in $O(mn_{\max}^2 2^{n_{\max}} + \tau_\beta 2^{mn_{\max}})$ time by enumerating all the feasible allocation matrices Y. If $C^*_{\max}$ is found for the values $T^*(l, Y^*)$, $l = 1, \ldots, m$, then the corresponding optimal permutations $\pi^*(l, Y^*)$, $l = 1, \ldots, m$, can be found in $O(mn_{\max})$ time by backtracking the dynamic programming algorithm described above. Thus, the problem $R|\Delta s_{lij}, \beta|C_{\max}$, $\beta \in \{cntn, dscr\}$, can be solved in $O(mn_{\max}^2 2^{n_{\max}} + \tau_\beta 2^{mn_{\max}})$ time, as indicated in the theorem. ∎

If the triangle inequality (1) is violated but the inequalities (6) hold, the algorithm described in Theorem 1 can be modified in the same fashion as it is suggested for the two-stage procedure given before this theorem. To see this, notice that the algorithm in Theorem 1 works with the setup times, and the modified setup times $\tilde{s}_{lij}$ do not depend on matrix Y.

3 Given number of products

In this section we consider the case, in which the number of products n is a given constant. Firstly, we will show that all four versions of the problem $R|s_{lij}, \beta|\gamma$ are NP-hard if there are

at least 2 products. Then we will present dynamic programming algorithms for the problems $R|s_{lij}, dscr|C_{\max}$ and $R|s_{lij}, dscr|L_{\max}$, which are linear in m and exponential in n. These results are novel and apply to the practically relevant situations, in which there is a small number of products and a large number of machines.

Theorem 2 *For any given $n \geq 2$, the problem $R|s_{lij}, \beta|\gamma$, $\beta \in \{cntn, dscr\}$, $\gamma \in \{C_{\max}, L_{\max}\}$, is NP-hard, even if all setup times are equal, all minimal lot-sizes are equal to zero, all processing times are product independent, all demands are equal, all upper bounds B_j are equal to infinity, and any two processing times differ by at most a factor of 2.*

Proof. Assume that $n = 2$. The case $n \geq 3$ can be handled similarly by introducing dummy products. We will use a reduction from the following NP-complete special case of the problem PARTITION (see, e.g., Schuurman and Woeginger [19]), which we call BOUNDED PARTITION: Given $2k+1$, where $k \geq 3$, positive integer numbers $e_1, \ldots, e_{2k}$ and E, which satisfy $\sum_{l=1}^{2k} e_l = 2E$ and $\frac{E}{k+1} < e_l < \frac{E}{k-1}$, $l = 1, \ldots, 2k$, is there a subset $X \subset K := \{1, \ldots, 2k\}$ such that $\sum_{l \in X} e_l = E$? Notice that set X is a solution to BOUNDED PARTITION only if $|X| = k$. Furthermore, $e_r/e_l \leq (k+1)/(k-1) \leq 2$ for any r and l from the set K.

Given an instance of BOUNDED PARTITION, we construct the following instance of the problem $R|s_{lij}, \beta|\gamma$. Calculate $A = \Pi_{r=1}^{2k} e_r$. Set $n = 2$, $m = 2k$, $D_j = E$, $B_j = \infty$, $d_j = 2A$, $p_{lj} = A/e_l$, $s_{lij} = A$ $(j \neq i)$, and $q_{lj}^0 = 0$ for $i = 0, 1, 2$, $j = 1, 2$, and $l = 1, \ldots, 2k$. Observe that any two processing times differ by at most a factor of 2: $p_{li}/p_{rj} = e_r/e_l \leq 2$, $l \in K$, $r \in K$, $i = 1, 2$. Since $\log A = \sum_{r=1}^{2k} \log e_r$, our reduction is polynomial with respect to the input length of BOUNDED PARTITION.

We show that BOUNDED PARTITION has a solution if and only if there exists a feasible schedule for the constructed instance of the problem $R|s_{lij}, \beta|\gamma$, $\gamma \in \{C_{\max}, L_{\max}\}$, such that $C_{\max} \leq 2A$, or equivalently, $L_{\max} \leq 0$. Consider a feasible schedule for which $C_{\max} \leq 2A$. Since all setup times are equal to A, each machine l can process at most e_l units of the same product within the remaning A available time units. Denote by X the set of machines each of which processes product 1. Then the following inequlities must be satisfied: $\sum_{l \in X} e_l \geq D_1 = E$ and $\sum_{l \in K \setminus X} e_l \geq D_2 = E$. We deduce that $\sum_{l \in X} e_l = E$, i.e., set X is a solution for BOUNDED PARTITION. Conversely, if some set X is a solution to BOUNDED PARTITION, then for a schedule, in which each machine $l \in X$ process e_l units of product 1, and each machine $r \in K \setminus X$ process e_r units of product 2, we have that the demand of E units for each product is satisfied, $C_{\max} \leq 2A$ and $L_{\max} \leq 0$, as required. ∎

We now pass to describing a dynamic programming algorithm for the problem $R|s_{lij}, dscr|C_{\max}$. Notice that the triangle inequality is not required to be satisfied for this problem. Our algorithm assigns product lots to machines $1, \ldots, m$ in this order, and enumerates the total numbers of products assigned so far, the product assigned last, and the completion time of the current machine. First of all, we determine an upper bound on the optimal $C_{\max}$ value:

$$C^*_{\max} \leq T := \max_{1 \leq l \leq m} \{ \sum_{j \in N_l} p_{lj} B_j + n_l \max_{i,j \in N_l} \{ s_{lij}, s_{l0j} \} \} \leq n_{\max}(p_{\max} B_{\max} + s_{\max}), \qquad (7)$$

where $C^*_{\max}$ is the optimal $C_{\max}$ value, $p_{\max} = \max\{p_{li} \mid l \in M_i, i = 1, \ldots, n\}$, $s_{\max} = \max\{\max\{s_{lij} \mid l \in M_i \cup M_j, i, j = 1, \ldots, n\}, \max\{s_{l0j} \mid l \in M_j, j = 1, \ldots, n\}\}$, and $B_{\max} = \max\{B_i \mid i = 1, \ldots, n\}$.

In our algorithm, values $C_l(z_1, \ldots, z_n, j, t)$ are recursively computed, where $C_l(z_1, \ldots, z_n, j, t)$ is the minimum $C_{\max}$ value for a partial schedule, in which z_i units of product i, $i = 1, \ldots, n$, are processed on the machines $1, \ldots, l$, product $j \in N_l$ is processed last on machine l, and the last unit of this product completes at time t. The initialization is $C_0(z_1, \ldots, z_n, j, t) = 0$ for $(z_1, \ldots, z_n, j, t) = (0, \ldots, 0)$, and $C_0(z_1, \ldots, z_n, j, t) = \infty$ for $(z_1, \ldots, z_n, j, t) \neq (0, \ldots, 0)$. The recursion for $l = 1, \ldots, m$, $z_i \in \{0, 1, \ldots, B_i\}$, $j = 0, 1, \ldots, n_l$, and $t = 0, 1, \ldots, T$, is given by the following formula.

$$C_l(z_1, \ldots, z_n, j, t) = \begin{cases} \min_{i \in N_{l-1} \cup \{0\}, t \in \{0,1,\ldots,T\}} \{C_{l-1}(z_1, \ldots, z_n, i, t)\}, & \text{if } (j,t) = (0,0), \\ \min_{i \in (N_l \cup \{0\}) \setminus \{j\}, \delta \in \{q^0_{lj}, q^0_{lj}+1, \ldots, z_j\}} \{\max\{t, & \\ C_l(z_1, \ldots, z_{j-1}, z_j - \delta, z_{j+1}, \ldots, z_n, i, t - (s_{lij} + \delta p_{lj}))\}\}, & \text{if } (j,t) \neq (0,0). \end{cases}$$

Here $(j, t) = (0, 0)$ means that no product is processed on the corresponding machine.

The optimal objective function value $C^*_{\max}$ can be determined from

$$C^*_{\max} = \min \Big\{ C_m(z_1, \ldots, z_n, i, t) \mid i \in N_m \cup \{0\}, t \in \{0, 1, \ldots, T\}, z_k \in \{D_k, D_k + 1, \ldots, B_k\}, k = 1, \ldots, n \Big\},$$

and the corresponding optimal schedule can be found by backtracking.

The algorithm runs in

$$O\left(\sum_{l=1}^{m} \left(\Pi_{j=1}^{n} (B_j + 1) n_l^2 T (B_{\max} + 1) \right) \right) =$$

$$O(m n_{\max}^2 T (B_{\max} + 1)^{n+1}) = O\left(m n_{\max}^3 (B_{\max} + 1)^{n+1} (s_{\max} + p_{\max} B_{\max}) \right) \qquad (8)$$

time, which is pseudopolynomial if n is a constant. The idea of this algorithm is different from the other dynamic programming algorithms derived for the batch scheduling problems.

To the best of our knowledge, in the parallel machine case, all of them are exponential in m, while our algorithm is linear in m.

If the setup times satisfy the triangle inequality, then the running time of the above algorithm can be reduced by using the fact that there exists an optimal solution in which each product has at most one lot on each machine. If the setup times are sequence independent, then the order of the (non-empty) lots on the same machine is immaterial. This fact can also be used to reduce the time complexity of the above algorithm.

Our algorithm for the problem $R|s_{lij}, dscr|L_{\max}$ is slightly different from the algorithm for the problem $R|s_{lij}, dscr|C_{\max}$. It also enumerates the completion time of the current machine, but due to the different objective function, it needs to enumerate more such values. Specifically, the value of $C_{\max}$ for an optimal solution to the problem $R|s_{lij}, dscr|L_{\max}$ can be greater than T, where T is given in (7). However, it does not exceed $T + d_{\max}$, where $d_{\max} = \max\{d_i \mid i = 1, \ldots, n\}$. Indeed, assume the contrary: $C_i > T + d_{\max}$ for some product i in an optimal solution. In this case, the optimal value $L^*_{\max}$ of the objective function satisfies

$$L^*_{\max} \geq C_i - d_i > T + d_{\max} - d_i \geq T. \tag{9}$$

However, from (7) we know that there exists a feasible solution, in which $C_i - d_i \leq T - d_i \leq T$ for all $i = 1, \ldots, n$, i.e., $L_{\max} \leq T$, which contradicts (9). We deduce that the machine completion times can be limited by $T + d_{\max}$ in the dynamic programming for the problem $R|s_{lij}, dscr|L_{\max}$. In our algorithm for this problem, values $L_l(z_1, \ldots, z_n, j, t)$ are recursively computed, where $L_l(z_1, \ldots, z_n, j, t)$ is the minimum $L_{\max}$ value for a partial schedule, in which z_i units of product i, $i = 1, \ldots, n$, are processed on all the machines $1, \ldots, l$, product $j \in N_l$ is processed last on machine l, and the last unit of this product completes at time t. The initialization is $L_0(z_1, \ldots, z_n, j, t) = 0$ for $(z_1, \ldots, z_n, j, t) = (0, \ldots, 0)$, and $L_0(z_1, \ldots, z_n, j, t) = \infty$ for $(z_1, \ldots, z_n, j, t) \neq (0, \ldots, 0)$. The recursion for $l = 1, \ldots, m$, $z_i \in \{0, 1, \ldots, B_i\}$, $j = 0, 1, \ldots, n_l$, and $t = 0, 1, \ldots, T + d_{\max}$, is given by the following formula.

$$L_l(z_1, \ldots, z_n, j, t) = \begin{cases} \min_{i \in N_{l-1} \cup \{0\}, t \in \{0,1,\ldots,T\}} \{L_{l-1}(z_1, \ldots, z_n, i, t)\}, & \text{if } (j,t) = (0,0), \\ \min_{i \in (N_l \cup \{0\}) \setminus \{j\}, \delta \in \{q^0_{lj}, q^0_{lj}+1, \ldots, z_j\}} \{\max\{t - d_j, & \\ L_l(z_1, \ldots, z_{j-1}, z_j - \delta, z_{j+1}, \ldots, z_n, i, t - (s_{lij} + \delta p_{lj}))\}\}, & \text{if } (j,t) \neq (0,0). \end{cases}$$

The optimal objective function value can be determined from

$$L^*_{\max} = \min \Big\{ L_m(z_1, \ldots, z_n, i, t) \mid$$

$$i \in N_m \cup \{0\}, t \in \{0, 1, \ldots, T\}, z_k \in \{D_k, D_k + 1, \ldots, B_k\}, k = 1, \ldots, n\Big\},$$

and the corresponding optimal schedule can be found by backtracking. The time complexity estimation of this algorithm can be obtained from that for the $C_{\max}$ criterion by replacing T with $T + d_{\max}$ in (8). Thus, we have

Theorem 3 *The problems* $R|s_{lij}, dscr|C_{\max}$ *and* $R|s_{lij}, dscr|L_{\max}$ *are solvable in* $O\left(mn_{\max}^3(B_{\max}+1)^{n+1}(s_{\max}+p_{\max}B_{\max})\right)$ *and* $O(mn_{\max}^3(B_{\max}+1)^{n+1}(s_{\max}+p_{\max}B_{\max}+d_{\max}))$ *time, respectively.*

4 Extension for a production line design problem

Consider the following problem. A production line has to be designed for a cyclic execution of n non-intersecting groups of operations on the same machine part, where each group i consists of D_i identical operations. The production line comprises m workstations and, in each cycle, every operation has to be executed exactly once on one of the workstations eligible for its execution. A transporting equipment like a conveyor is used to move the machine part from workstation l to workstation $l+1$, $l = 1, \ldots, m-1$. Operations of the same workstation are performed sequentially. Furthermore, a setup time s_{lij} is needed if an operation of group j is performed after an operation of group i on machine l. Precedence relations are given on the set of groups of operations. If group i precedes group j, which we denote as $i \to j$, then there is a one-to-one correspondence between the operations of groups i and j (which implies $D_i = D_j$), and each operation of group j cannot start earlier than the corresponding operation of group i. If $o(i)$ and $o(j)$ are the above mentioned operations of groups i and j, and $o(i)$ is performed on workstation l, then $o(j)$ cannot be performed on any workstation $r \in \{1, \ldots, l-1\}$ and on the workstation l before the operation $o(i)$. The precedence relations reflect the technological requirements for the operations. For example, group i may consist of 10 drilling operations to make 10 identical holes, and group j may consist of 10 threading operations, each for one of the drilled holes. The total setup and processing time of the workstation is called its *cycle time*. The problem is find an allocation of the operations to the workstations and their sequence on each workstation such that the *line cycle time* is minimized. The line cycle time is the maximum among all the workstation cycle times.

The production line design problem formulated above is closely related to the flexible assembly line design problem studied by Bukchin and Tzur [6]. However, in their problem

the setup times are negligibly small, the number of workstations is the decision variable, and the objective is to minimize the cost of the operations plus the cost of the workstations, provided that an upper bound on the line cycle time is not exceeded.

It is easy to see that our production line design problem can be modeled as the problem $R|s_{lij}, dscr|C_{\max}$ with the additional precedence constraints given on the set of products, and the assumptions $q_{lj}^0 = 1$, $l \in M_j$, and $D_j = B_j$, $j = 1, \ldots, n$. In this problem, the products will represent the groups of operations, and the machines will represent the workstations. We will denote this problem as $R|s_{lij}, dscr, prec^{LD}, q_{lj}^0 = 1, D_j = B_j|C_{\max}$, where descriptor $prec^{LD}$ indicates that there are precedence constraints on the set of products specific for the production line design (LD) problem.

The dynamic programming algorithm for the problem $R|s_{lij}, dscr|C_{\max}$ presented in Section 3 can be easily adapted to handle the problem $R|s_{lij}, dscr, prec^{LD}, q_{lj}^0 = 1, D_j = B_j|C_{\max}$. The only modification is that we should limit the enumeration of the state variables $z_1, \ldots, z_n$ so that $z_i \geq z_j$ if product i precedes product j. Therefore, the time complexity estimation of this algorithm remains unchanged.

The algorithms presented in Section 2 can be adapted to solve the problem $R|s_{lij}, dscr, prec^{LD}, q_{lj}^0 = 1, D_j = B_j|C_{\max}$ with an additional constraint that each workstation l can process at most one lot of any product $i \in M_l$. In the two-stage solution procedure of Section 2 and in the algorithm described in Theorem 1, the integer linear programming formulation should include additional constraints such that if $\pi^{(l)} = (i_1^{(l)}, \ldots, i_{k_l}^{(l)}) \in P(Y, l)$ and $i \rightarrow i_j^{(l)}$, $1 \leq j \leq k_l$, then for the lot of the product $i_j^{(l)}$ on machine l, the total number of units of this product allocated to machines $1, \ldots, l$ should not exceed the total number of units of product i allocated to machines $1, \ldots, l-1$ and those sequenced before the product $i_j^{(l)}$ on machine l :

$$\sum_{k=1}^{l} x_{ki_j^{(l)}} \leq \sum_{r=1}^{l-1} x_{ri} + x'_{li}, \text{ for } i \rightarrow i_j^{(l)}, \ 1 \leq j \leq k_l, \ i = 1, \ldots, n. \tag{10}$$

where

$$x'_{li} = \begin{cases} x_{li}, & \text{if } i \in \{i_1^{(l)}, i_2^{(l)}, \ldots, i_{j-1}^{(l)}\}, \\ 0, & \text{otherwise.} \end{cases}$$

Thus, the problem $R|s_{lij}, dscr, prec^{LD}, q_{lj}^0 = 1, D_j = B_j|C_{\max}$ with the additional constraint that each workstation l can process at most one lot of any product $i \in M_l$ can be solved in $O(mn_{\max}^2 2^{n_{\max}} + \tau_{dscr}^{(LD)} 2^{mn_{\max}})$ time, where $\tau_{dscr}^{(LD)}$ is the running time of the integer linear programming algorithm for the problem (2)-(5), (10).

5 Conclusions

We have derived the following computational complexity and algorithmic results for various special cases of the problem $R|s_{lij},\beta|\gamma$:

- The problem $R|\Delta s_{lij},\beta|C_{\max}$, $\beta \in \{cntn, dscr\}$ is solvable in $O(mn^2_{\max}2^{n_{\max}} + \tau_\beta 2^{mn_{\max}})$ time.

- For any given $n \geq 2$, the problem $R|s_{lij},\beta|\gamma$, $\beta \in \{cntn, dscr\}$, $\gamma \in \{C_{\max}, L_{\max}\}$, is NP-hard, even if all setup times are equal, all minimal lot-sizes are equal to zero, all processing times are product independent, all demands are equal, all upper bounds B_j are equal to infinity, and any two processing times differ by at most a factor of 2.

- The problem $R|s_{lij}, dscr|C_{\max}$ is solvable in $O\left(mn^3_{\max}(B_{\max}+1)^{n+1}(s_{\max}+p_{\max}B_{\max})\right)$ time.

- The problem $R|s_{lij}, dscr|L_{\max}$ is solvable in $O(mn^3_{\max}(B_{\max}+1)^{n+1}(s_{\max}+p_{\max}B_{\max}+d_{\max}))$ time.

- The problem $R|s_{lij}, dscr, prec^{LD}, q^0_{lj}=1, D_j = B_j|C_{\max}$ is solvable in $O(mn^3_{\max}(B_{\max}+1)^{n+1}(s_{\max}+p_{\max}B_{\max}))$ time.

- The problem $R|s_{lij}, dscr, prec^{LD}, q^0_{lj}=1, D_j = B_j|C_{\max}$ with the additional constraint that each workstation l can process at most one lot of any product $i \in M_l$ is solvable in $O(mn^2_{\max}2^{n_{\max}} + \tau^{(LD)}_{dscr}2^{mn_{\max}})$ time.

The latter two results can be used for modeling and solving the production line design problem described in Section 4. Further research can be undertaken to develop efficient approximation algorithms for the problem $R|s_{lij},\beta|\gamma$ and its important special cases.

References

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

[2] dos Santos-Meza, E., dos Santos, M.O., Arenales, M.N., A lot-sizing problem in an automated foundry, European Journal of Operational Research 139 (2002) 490-500.

[3] Ausiello, G., Crescenzi, P., Gambosi, G., Kann, V., Marchetti-Spaccamela, A., Protasi, M., Complexity and approximation: combinatorial optimization problems and their approximability properties, Springer Verlag, Berlin, 1999.

[4] Bitran, G.R., Gilbert, S.M., Sequencing production on parallel machines with two magnitudes of sequence-dependent setup cost, Journal of Manufacturing and Operations Management 3 (1990) 24-52.

[5] Brucker, P., Kovalyov,M.Y., Shafransky, Y.M., Werner, F., Batch scheduling with deadlines on parallel machines, Annals of Operations Research 83 (1998) 23-40.

[6] Bukchin, J., Tzur, M., Design of flexible assembly line to minimize equipment cost, IIE Transactions 32 (2000) 585-598.

[7] Garey, M.R., Johnson, D.S., Computers and intractability. A guide to the theory of NP-completeness, W.H. Freeman and Company, San Francisco, 1979.

[8] Graham, R.L., Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A.H.G., Optimization and approximation in deterministic sequencing and scheduling and scheduling, Annals of Discrete Mathematics 5 (1979) 287-326.

[9] Held, M., Karp, R. M., A dynamic programming approach to sequencing problems. Journal of the Society for Industrial and Applied Mathematics 10 (1962) 196-210.

[10] Khachiyan, L.G., A polynomial algorithm in linear programming, Dokladi Akademii Nauk SSSR 244 (1979) 1093-1096.

[11] Lenstra, H.W. Jr., Integer programming with a fixed number of variables, Mathematics of Operations Research 8 (1983) 538-548.

[12] Lin, X., Floudas, C.A., Modi, S., Juhasz, N.M., Continuous-time optimization approach for medium-range production scheduling of a multiproduct batch plant, Industrial and Engineering Chemistry Research 41 (2002) 3884-3906.

[13] Monma, C.L., Potts, C.N., On the complexity of scheduling with batch setup times, Operations Research 37 (1989) 798-804.

[14] Orponen, P., Mannila, H., On approximation preserving reductions: Complete problems and robust measures, Technical Report C-1987-28, Department of Computer Science, University of Helsinki, 1987.

[15] Papadimitriou, C.H., Vempala, S., On the approximability of the traveling salesman problem, Combinatorica 26 (2006) 101-120.

[16] Potts, C.N., Kovalyov, M.Y., Scheduling with batching: A review, European Journal of Operational Research 120 (2000) 228-249.

[17] Potts, C.N., Van Wassenhove, L.N., Integrating scheduling with batching and lot-sizing: a review of algorithms and complexity, Journal of the Operational Research Society 43 (1992) 395-406.

[18] Silva, C., Magalhaes, J.M. Heuristic lot size scheduling on unrelated parallel machines with applications in the textile industry, Computers & Industrial Engineering 50 (2006) 76-89.

[19] Schuurman, P., Woeginger, G., Approximation schemes - a tutorial. Manuscript, to appear in Lectures on Scheduling, edited by R.H. Mohring, C.N. Potts, A.S. Schulz, G.J. Woeginger and L.A. Wolsey, 2006.

[20] Shaik, M.A., Floudas, C.A., Kallrath, J., Pitz, H.-J., Production scheduling of a large-scale industrial continuous plant: short-term and medium-term scheduling. In: Proc. of 17th European Symposium on Computer Aided Process Engineering (ESCAPE17), edited by V. Plesu and P.S. Agachi, Elsevier, 2007.

[21] Shor, N.Z., Convergence rate of the gradient descent method with dilatation of the space, Kibernetika 2 (1970) 80-85.

[22] Taner, M.R., Hodgson, T.J., King, R.E., Schultz, S.R., Satisfying due-dates in the presence of sequence dependent family setups with a special comedown structure, Computers and Industrial Engineering 52 (2007) 57-70.

[23] Vavasis, S.A., Ye, Y., A primal-dual interior point method whose running time depends only on the constraint matrix, Mathematical Programming 74 (1996) 79-120.

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