Two Customized Parallel Machines Scheduling Problem with Precedence Relations

Evgeny R. Gafarov

Ecole Nationale Superieure des Mines, FAYOL-EMSE, CNRS:UMR6158, LIMOS, F-42023 Saint-Etienne, France,
Institute of Control Sciences of the Russian Academy of Sciences, Profsoyuznaya st. 65, 117997 Moscow, Russia,
email: axel73@mail.ru

Alexandre Dolgui

Ecole Nationale Superieure des Mines, FAYOL-EMSE, CNRS:UMR6158, LIMOS, F-42023 Saint-Etienne, France
email: dolgui@emse.fr

Abstract

In this paper, we consider two customized parallel machines scheduling problem with precedence relations to minimize makespan. Complexity and approximation results are presented.

Keywords: Scheduling, Two customized machines, precedence relations, makespan

Introduction

The two-dedicated-parallel-machines scheduling problem is formulated as follows:

We are given a set $N = \{1, 2, \ldots, n\} = N_1 \bigcup N_2 \bigcup N_{1or2} \bigcup N_{1and2}$ of n jobs that must be processed on two machines. Jobs from the subset N_1 have to be processed on the first machine, jobs from the subset N_2 on the second one, jobs from the subset N_{1or2} can be processed on any of them, jobs from the subset N_{1and2} use both machines simultaneously. Job preemption is not allowed. Each machine can handle only one job at a time. All the jobs are assumed to be available for processing at time 0. For each job j, $j \in N$, a processing time $p_j \geq 0$ is given. Furthermore, arbitrary finish-start precedence relations $i \to j$ are defined between the jobs according to an acyclic directed graph G. The objective is to determine the starting time S_j for each job j, $j = 1, 2, \ldots, n$, in such a way that the given precedence relations are fulfilled and the makespan $C_{max} = \max_{j=1}^{n} C_j$, where $C_j = S_j + p_j$, is minimized. Denote this problem as $P2|prec, N_1, N_2, N_{1or2}, N_{1and2}|C_{max}$.

This problem originally appeared as a sub-problem of the well-known two-sided assembly line balancing problem. To define it, firstly,

we describe a simple assembly line balancing problem. A single-model paced assembly line which continuously manufactures a homogeneous product in large quantities is considered (mass production). The simple assembly line balancing problem (SALBP-1) is to find an optimal line balance for a given cycle time c, i.e., to find a feasible assignment of given operations to stations in such a way that the number of stations used m reaches its minimal value. The SALBP-1 is defined as follows.

Given a set $N = \{1, 2, \ldots, n\}$ of operations and K stations (machines) $1, 2, \ldots, K$. For each operation $j \in N$ a processing time $t_j \geq 0$ is defined. The cycle time $c \geq \max\{t_j,\ j \in N\}$ is given. Furthermore, finish-start precedence relations $i \to j$ are defined between the operations according to an acyclic directed graph G. The objective is to assign each operation $j,\ j = 1, 2, \ldots, n$, to a station in such a way that:
- number $m \leq M$ of stations used is minimized;
- for each station $k = 1, 2, \ldots, m$ a total load time $\sum_{j \in N_k} t_j$ does not exceed c, where N_k – a set of operations assigned to a station k;
- given precedence relations are fulfilled, i.e. if $i \to j,\ i \in N_{k_1}$ and $j \in N_{k_2}$ then $k_1 \leq k_2$.

In the two-sided assembly line balancing problem (TSALBP-1) instead of single stations, pairs of opposite stations on either side of the line (left and right side stations) work in parallel, i.e., they work simultaneously at opposite sides of the same workpieces. Some operations have to be performed on the right-hand-side and on the left-hand-side, respectively, while others may be done on either side of the line or can require both sides simultaneously.

While for SALBP-1 all jobs from a set N_k where $\sum_{j \in N_k} t_i \leq c$ can be processed on the single station, for TSALBP-1 the question appears: Is it possible to process all jobs from a set N_l where $c < \sum_{j \in N_l} t_i \leq 2c$ on a pair of opposite stations? So, the problem $P2|prec, N_1, N_2, N_{1or2}, N_{1and2}|C_{max}$ is obtained.

Two-machines problems are considered as fundamental scheduling problems, which are a special case of parallel machines problems (see, e.g., a survey [1]). Papers on different two-machines models appear permanently (see, e.g., [2]). If $N_{1or2} = N$, i.e., $N_1 = N_2 = N_{1and2} = \emptyset$, then we have the classical two identical parallel machines problem $P2|prec|C_{max}$ which is NP-hard [1]. A two-parallel machines early-tardy scheduling problem where some jobs need to be processed by one machine, while the others have to be processed by both machines simultaneously is presented in [4]. In this paper we consider the special case of the problem, where $N_{1or2} = N_{1and2} = \emptyset$, which is denoted as $P2|prec, N_1, N_2|C_{max}$. A similar problem without precedence relations was considered in [3], where jobs are assigned to the machine in advance and an incompatibility relation was defined over the tasks which forbids any two incompatible tasks to be processed at the same time.

SALBP-1 is NP-hard in the strong sense. Surveys on results for SALBP-1 are published periodically (e.g., [5]). There exists a special electronic library http://www.assembly -line-balancing.de of experimental data to test solution algorithms for this problem.

The rest of the paper is organized as follows. In the first Section some complexity results for special subcases are presented. Approximation results are discussed in Section 2. This paper is finishing up in Section 3 with the conclusion.

1 Complexity Results

Denote by $P2|chain, N_1, N_2|C_{max}$ a special subcase of the problem, where G consists only chains of jobs and by $P2|prec, p_j = 1, N_1, N_2|C_{max}$ a special subcase with equal-processing-times of jobs.

3-Partition problem:
A set $N = \{b_1, b_2, \ldots, b_n\}$ of $n = 3m$ positive integers is given, where $\sum_{i=1}^{n} b_j = mB$ and $\frac{B}{4} < b_j < \frac{B}{2},\ j = 1, 2, \ldots, n$. Does there exist a partition of N into m subsets $\overline{N}_1, \overline{N}_2, \ldots, \overline{N}_m$ such that each subset consists exactly three numbers and the sum of the numbers in each subset is equal, i.e.,

$$\sum_{b_j \in \overline{N}_1} b_j = \sum_{b_j \in \overline{N}_2} b_j = \cdots = \sum_{b_j \in \overline{N}_m} b_j = B?$$

Lemma 1 *$P2|chain, N_1, N_2|C_{max}$ is NP-hard in the strong sense.*

Proof. We give a reduction from the 3-Partition problem. Given an instance of the 3-Partition problem with $3m$ numbers. Construct an instance of $P2|chain, N_1, N_2|C_{max}$ with $5m - 1$ jobs. The first $3m$ jobs are independent, $p_j = b_j,\ j = 1, 2, \ldots, 3m$, and there is a chain of jobs $3m + 1 \to 3m + 2 \to 3m + 3 \to \cdots \to 5m - 1$, where $p_j = B,\ j = 3m+1, 3m+3, \ldots, 5m-1$ and $p_j = 1,\ j = 3m+2, 3m+4, \ldots,\ 5m-2$. Furthermore, $N_1 = \{3m + 1, 3m + 3, \ldots, 5m - 1\}$ and $N_2 = \{1, 2, \ldots, 3m, 3m + 2, 3m + 4, \ldots, 5m - 2\}$. See Fig.1(a).

If and only if the instance of the 3-Partition problem has the answer "YES", there is a schedule in which a subset of jobs which corresponds to the set $\overline{N}_i$ is processed in parallel with the job $3m + (2i - 1),\ i = 1, 2, \ldots, m$. Starting times $S_{3m+2i-1} = (B + 1)(i - 1),\ i = 1, 2, \ldots, m$, and $S_{3m+2i} = Bi + (i - 1),\ i = 1, 2, \ldots, m - 1$. For such the schedule $C_{max} = mB + m - 1$.

□

We can present the similar reduction from a decision version of SALBP-1 to $P2|prec, N_1, N_2|C_{max}$. In the decision version, we have to answer the question, whether there is a line balance with m stations.

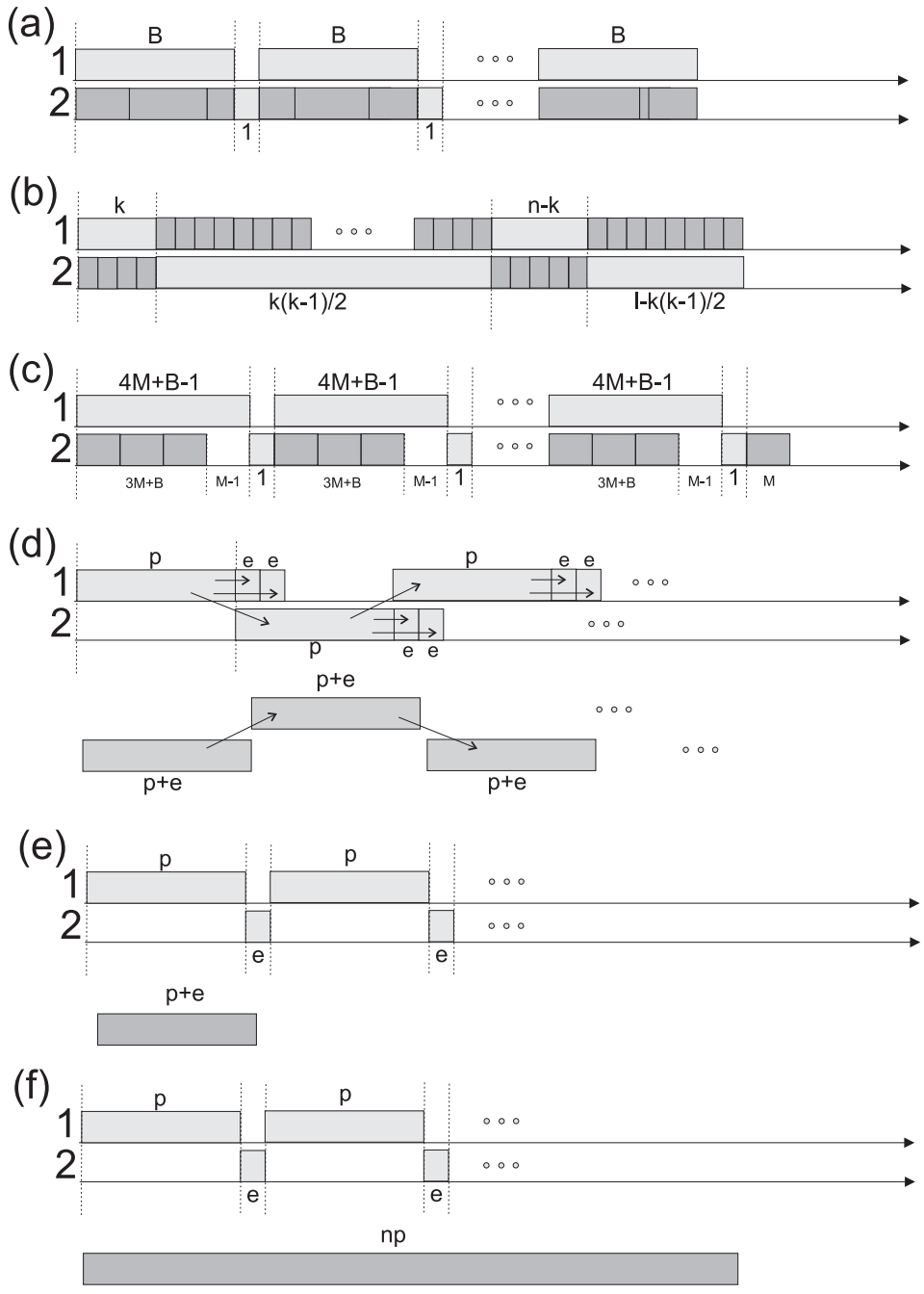

Figure 1: Examples

For this reduction there is a subset of jobs which corresponds to the set of operations from SALBP-1 with the processing times $p_j = t_j,\ j = 1, 2, \ldots, n$, and the same precedence relations. Furthermore, a chain of long and short jobs $n+1 \to n+2 \to n+3 \to \cdots \to n+2m-1$ is given, where $p_j = c,\ j = n+1, n+3, \ldots, n+2m-1$ and $p_j = 1,\ j = n+2, n+4, \ldots, n+2m-2$. If and only if the instance of SALBP-1 has the answer "YES", there is a schedule for which $C_{max} = mc+(m-1)$. So, the following Lemma is proven.

Lemma 2 *Decision version of SALBP-1 can be reduced to* $P2|prec, N_1, N_2|C_{max}$ *in polynomial time.*

Since to solve TSALBP-1, solution methods for SALBP-1 can be used, where $P2|prec, N_1, N_2, N_{1or2}, N_{1and2}|C_{max}$ has to be solved as a subproblem, it seems to be interesting that SALBP-1 can be reduced to a special case of $P2|prec, N_1, N_2|C_{max}$. Furthemore, there are instances of SALBP-1 for which any known Branch and Bound algorithm with a lower bound computed in polynomial time can not solve instances with $n \geq 60$ operations in appropriate time [6]. So, it seems to be inadvisable to try to construct an effective solution algorithm for the general case of the problem under consideration.

Clique Problem:
Given a graph $G = (V, E)$ and an integer k, does G have a clique (i.e., a complete subgraph) on k vertices?

Lemma 3 $P2|prec, p_j = 1, N_1, N_2|C_{max}$ *is NP-hard in the strong sense.*

Proof. We give a reduction from the Clique problem[1]. We introduce a job J_v for every vertex $v \in V$ and a job J_e for every edge $e \in E$, with $J_v \to J_e$ whenever v is endpoint of e. Denote $\overline{n} = |V|$ and $l = |E|$. The processing times of all the jobs equal 1. Jobs $J_v \in N_2, \forall v \in V$, and $J_e \in N_1, \forall e \in E$. We also add the chain of jobs $\overline{n}+1 \to \overline{n}+2 \to \overline{n}+3 \to \overline{n}+4$, where $p_{\overline{n}+1} = k,\ p_{\overline{n}+2} = k(k-1)/2,\ p_{\overline{n}+3} = n-k,\ p_{\overline{n}+4} = l-k(k-1)/2$ and $\overline{n}+1, \overline{n}+3 \in N_1,\ \overline{n}+2, \overline{n}+4 \in N_2$. See Fig.1(b).

If and only if the instance of clique problem has the answer "YES", there is a schedule for which $C_{max} = \overline{n} + l = \sum_{i=\overline{n}+1}^{\overline{n}+4} p_i$. Denote the clique by $G'(V', E')$. Jobs $J_v,\ v \in V'$, are processed in parallel with the job $\overline{n}+1$. Jobs $J_e,\ e \in E'$, are processed in parallel with the job $\overline{n}+2$. Jobs $J_v,\ v \in V \setminus V'$ are processed in parallel with the job $\overline{n}+3$. Jobs $J_e,\ e \in E \setminus E'$ are processed in parallel with the job $\overline{n}+4$.

If there is no clique of size k, then after scheduling of k jobs $J_v,\ v \in V$, we will be able to schedule no more than $k(k-1)/2-1$ jobs $J_e,\ e \in E$, in parallel with the job $\overline{n}+2$.

[1] We use a similar idea like in [1] for $P|prec, p_j = 1|C_{max}$ problem

The jobs $\overline{n}+1, \overline{n}+2, \overline{n}+3, \overline{n}+4$ can be substituted for chains of k, $k(k-1)/2$, $\overline{n}-k$, and $l-k(k-1)/2$ equal-processing-time jobs, respectively, i.e. the special case, where $p_j = 1$, is received.

So, the Lemma is proven.

□

As a consequence from Lemma 5, for the special case $P2|prec, p_j = 1, N_1, N_2|C_{max}$ the approximation ratio of polynomial time algorithms is not less than $2/n$ and there is no FPTAS (fully-polynomial time approximation schema) for the special case.
Denote the problem, where preemptions of jobs are allowed by $P2|prec, pmtn., N_1, N_2|C_{max}$.

Corollary 1 *$P2|prec, pmtn., N_1, N_2|C_{max}$ is NP-hard in the strong sense.*

Denote $C^*_{max}(pmtn.)$ – the minimal makespan for the problem with preemptions.

Lemma 4 *For the problem $P2|prec, N_1, N_2|C_{max}$ an inequality $\frac{C^*_{max}}{C^*_{max}(pmtn.)} < 2$ holds and there is an instance for which $\frac{C^*_{max}}{C^*_{max}(pmtn.)} \approx 2$.*

Proof. It's obvious that

$$\frac{1}{2}\sum_{j\in N} p_j \le C^*_{max}, C^*_{max}(pmtn.) < \sum_{j\in N} p_j.$$

So, the first part of Lemma is true.

To prove the second part let us consider an instance with chain of $2n-1$ jobs, $p_j = p,\ j = 1, 3, \ldots, 2n-1$, and $p_j = e,\ j = 2, 4, \ldots, 2n-2$. In addition, an independent job $2n$ is given with $p_{2n} = np$. $N_1 = \{1, 3, \ldots, 2n-1\}$ and $N_2 = \{2, 4, \ldots, 2n-2, 2n\}$ (see Fig.1(f)). For such the instance $C^*_{max} = (n-1)(p+e) + np$ and $C^*_{max}(pmtn.) = np + (n-1)e$. Then for $e \to 0$ the second part of the Lemma is true.
□

2 Approximation by List Scheduling Algorithm

To solve problems with precedence relations (e.g., SALBP-1, $P2|prec|C_{max}$) enumeration schemas based on the well-known List Scheduling (LS) Algorithm are usually used. The problem $P2|prec, N_1, N_2|C_{max}$ can be solved by an algorithm based on LS as well. The main idea of LS is as follows: on each step $j = 1, 2, \ldots, n$, choose a job (operation) for which all predecessors are already scheduled and assign it to be performed from the earliest possible starting

time according to precedence relations and resource restrictions. According to such the algorithm only *active* schedules will be constructed for which there is no job which can be shifted to an earlier starting time without violating of precedence or resource constraints. It's obvious, that among active schedules there are optimal ones, that's why, an optimal solution can be presented as a sequence (permutation) of n jobs, which denotes the order of jobs' choice in LS. Different dominations rules are used in LS, which define the jobs' choice, e.g., choose a job with the maximal processing time among ready to be scheduled jobs (LPT), or choose a job which belongs to a critical path (CP) etc.

List Scheduling is widely used for scheduling problems with precedence relations to compute an Upper Bound, i.e., to find an feasible solution. The question appears, which approximation ratios LS with different domination rules has. Let us denote the optimal objective function value by C^*_{max} and the objective function value for the solution constructed by LS with domination rules α by $C_{max}(LS_\alpha)$. It is known[1], that for the problem $P|prec|C_{max}$ we have $\frac{4}{3} \leq \frac{C_{max}(LS_\alpha)}{C^*_{max}} < 2$ for any domination rule α checked in polynomial time. For SALBP-1 we have $\frac{3}{2} \leq \frac{m(LS_\alpha)}{m^*} < 2$, where m^* – minimal number of stations and $m(LS_\alpha)$ – number of stations for the solution constructed by LS with any domination rule α.

It is obvious that for the problem $P2|prec, N_1, N_2|C_{max}$, we have $\frac{C_{max}(LS_\alpha)}{C^*_{max}} < 2$, since

$$\frac{1}{2}\sum_{j\in N} p_j \leq C^*_{max}, C_{max}(LS_\alpha) < \sum_{j\in N} p_j$$

.

For some problems it can be useful to know the worst possible active schedule constructed by LS. Such problems with opposite optimality criteria have both theoretical and practical significance [7]. For the problem under consideration, we can note such a problem with opposite optimality criteria, namely to maximize the makespan where only active schedules are considered, by $P2|prec, N_1, N_2|C_{max} \to \max$. Unfortunately, the proposed maximization problem is strongly NP-hard, too.

Lemma 5 *$P2|chains, N_1, N_2|C_{max} \to \max$ is NP-hard in the strong sense.*

Proof. We give a reduction from the 3-Partition problem. Given an instance of the 3-Partition problem with $3m$ numbers. Let $M = (mB)^2$. Construct an instance of $P2|chain, N_1, N_2|C_{max}$ with $5m+1$ jobs. The first $3m+1$ operations are independent, $p_j = M + b_j,\ j = 1, 2, \ldots, 3m$, and $p_{3m+1} = M$. In addition, there is a chain of jobs $3m+2 \to 3m+3 \to 3m+4 \to \cdots \to 3m+2m+1$, where $p_j = 4M +$

$B-1,\ j=3m+2,3m+4,\ldots,3m+2m$ and $p_j=1,\ j=3m+3,3m+5,\ldots,3m+2m+1$. Furthermore, $N_1=\{3m+2,3m+4,\ldots,3m+2m\}$ and $N_2=\{1,2,\ldots,3m+1,3m+3,3m+5,\ldots,3m+2m+1\}$. See Fig.1(c).

If and only if the instance of the 3-Partition problem has the answer "YES", there is an active schedule in which the subset of jobs which corresponds to the set $\overline{N}_i$ is processed in parallel with a job $3m+1+(2i-1),\ i=1,2,\ldots,m$. Starting times $S_{3m+1+(2i-1)}=(4M+B-1+1)(i-1),\ i=1,2,\ldots,m$, and $S_{3m+1+2i}=(4M+B-1)i+(i-1),\ i=1,2,\ldots,m$. The job $3m+1$ is processed independently from the time $(4M+B-1+1)m$. For such the schedule $C_{max}=(4M+B)m+M$.

If the answer is "NO" then $C_{max}=(4M+B)m$, and there is a job $3m+1+(2i-1),\ i\in\{1,2,\ldots,m\}$, which is processed in parallel with 4 jobs from the set $\{1,2,\ldots,3m+1\}$ (including the job $3m+1$).

□

We show that approximation ration of LS with the following domination rules is ≈ 2: CP – critical path rule (choose a job which belongs to a critical path [5, 1]), LPT–choose a job with the maximal processing time, MS – choose a job with the maximal number of immediate successors.

Lemma 6 *There are instances for which*

$$\frac{C_{max}(LS_\alpha)}{C^*_{max}}\approx 2,\ \alpha\in\{CP,LPT,MS\}.$$

Proof.

For the rule MS we consider an instance from Fig.1(d). In this instance we have a chain of k jobs with processing times p. k is odd. Each such a job precedes two jobs with processing times e. Additionally, there is a chain of k jobs with processing times $p+e$. Then $C_{max}(LS_{MS})=(k-1)p+2e+k(p+e)$ and $C^*_{max}=k(p+e)+ke/2$. For $k\to\infty,\ e\to 0$, the lemma is true.

For the rule CP consider an instance from Fig.1(e). In this instance we have a chain of $2k$ jobs with processing times p and e. Additionally, there is k independent jobs with processing times $p+e$ which have to be processed on the second machine. Then $C_{max}(LS_{CP})=2k(p+e)$ and $C^*_{max}=k(p+2e)$. For $e\to 0$ the lemma is true.

If we modify the instance for CP by adding a job $2k+1$ with processing time $e/2$, which precedes all independent jobs with the processing time $p+e$, then for the modified instance $\frac{C_{max}(LS_{LPT})}{C^*_{max}}\approx 2$.

□

We conjecture that the same relation is true for other rules α computed in polynomial time.

3 Conclusion

In this paper, we present some complexity and approximation results for the two-dedicated-parallel-machines scheduling problem with precedence relations to minimize makespan, which is a sub-problem of two-sided assembly line balancing problem. The presented results shows that the two-machines problem is not easier than well-known SALBP-1, i.e. there is no Branch and Bound algorithm with a polynomial time computed Lower Bound that solve instances of a special case even for $n = 60$ jobs in appropriate time. For the future research a question appears: if there is a constant $a, 1 < a < 2$, in which the problem is either approximable or not.

Acknowledgement

The authors are grateful to Chris Yukna for his help regarding the English presentation.

References

[1] E.L. Lawler, J.K. Lenstra, A.H.G. Rinnoy Kan , D.B. Shmoys, Sequencing and Scheduling: Algorithms and Complexity, Report BS-8909,Stichting Matematisch Centrum, Amsterdam, 1989.

[2] Manzhan Gu, Xiwen Lu, Preemptive stochastic online scheduling on two uniform machines, Information Processing Letters, 109 Issue 7 (2009) 369–375.

[3] I. N. Lushchakovaa, Vitaly A. Strusevich, Scheduling incompatible tasks on two machines, European Journal of Operational Research, 200 Issue 2 (2010) 334–346.

[4] Sheng-Fuu Lin, Jaw-Yeh Chen, Two parallel machines scheduling, Systems Analysis Modelling Simulation, 42 Issue 10 (2002) 1429-1437.

[5] A. Scholl, Balancing and Sequencing of Assembly Lines, Physica Verlag, A Springer-Verlag Company, 1999.

[6] E.R. Gafarov, A.Dolgui, Notes on Complexity of the Simple Assembly Line Balancing Problem, Preprint, Ecole Nationale Superieure des Mines, 2012.

[7] M.A. Aloulou , M.Y. Kovalyov, M.-C. Portmann, Maximization Problems in Single Machine Scheduling, Annals of Operations Research, 129 (2004) 21 – 32.