Balancing Machining Lines: a Two-phase Heuristic

Mohamed Essafi, Xavier Delorme, Alexandre Dolgui

Ecole Nationale Supérieure des Mines de Saint-Etienne,
Centre for Industrial Engineering and Computer Science,
158 Cours Fauriel, 42023, Saint-Etienne, France
{essafi,delorme,dolgui}@emse.fr

Abstract: This paper considers balancing machining lines with parallel machines and sequence-dependent setup times. The goal is to minimize the number of machines for a given cycle time. Such lines are paced, i.e. parts are moved from one station to the next with a given cadence (defined by the line cycle time). At least one machine is installed at each station. Parallel machines are to be used when the corresponding station is overloaded, i.e. the total time of all operations assigned to the station exceeds the line cycle time. Moreover, station workload depends on the sequence in which the operations are executed because the setup times related to the changes and displacements of tools, rotations of the part, etc. In this paper, a heuristic method is proposed for the resolution of this problem. An industrial example is presented and numerical results are reported.

Keywords: Machining lines, parallel machines, sequence-dependent setup times, heuristic method, optimization.

1. Introduction

Machining lines are widely used in automotive and other industries. They are expensive with heavy investments in their installation and implementation. This investment influences to a great extent the cost of the finished products. Therefore, machining line manufacturers are increasingly interested in the optimization of the line design process. The objective is to optimize some criteria such as the total investment cost, total number of workstations, cost of operations (tool, men power, energy, etc.), or the cycle time. Note that the profitability of the line depends directly on the expense of its design and so should be minimized. In this manner, optimization has become a crucial issue for the machining line design.

In this paper, we deal with the machining line balancing problem which appears at the preliminary design stage. We consider the serial-parallel lines composed of CNC (Computer Numerical Control) machines. There are parallel machines at each station and setup times between operations at each machine (sequence dependent setup-times). The main objective of the line balancing problem is to minimize the total number of machines for a given cycle time. We propose a two-phase heuristic method for the resolution of this problem.

This work is developed in co-operation with an industrial partner (PCI-SCEMM). PCI proposes solutions for the machining of complex parts such as cylinder heads. A peculiarity of this type of line is the necessity to put several machines in parallel at each station to respect a given cycle time. An additional characteristic of these lines is that there are sequence-dependent setup times at each station. To our knowledge, such a problem has not been studied in the literature.

The machining line is a special case of assembly lines [15]. A machining line is equipped with a set of machines by which different operations are executed. Each operation is characterized by: an operational time; a set of operations which must be assigned before (precedence constraints); a set of operations which must be executed on the same workstation (inclusion constraints); a set of operations which cannot be executed on the same workstation (exclusion constraints). The specificity of the considered problem deals with two main factors. The first is the necessity to consider unproductive times between the operations (setup times) varying according to the sequence in which the operations are assigned. The second factor is the possibility (and even the necessity) to parallelize the flow at each station using several parallel machines. The utilization of parallel machines creates an additional difficulty to balance the load, but it is necessary in the case of a station with the processing time higher than cycle time.

The paper is composed of six sections. In Section 0 a state of the art for the assembly line balancing problems is presented. Particular attention is given to those problems with parallel machines or sequence-dependent setup times. In Section 0, a complete definition of the problem as well as the notations used is given. A heuristic method for approximate resolution of this problem is then proposed in Section 0. Section 0 deals with computational experiments and, finally Section 0 provides concluding remarks and perspectives of this work.

2. State of the Art

The assembly line balancing problem (ALBP) has been widely studied since the first publication of Salveson [22]. Furthermore, exhaustive studies have been made by several researchers in the last fifty years with many interesting applications covered. Comprehensive surveys for this problem are presented in [6], [9], and [20].

In [6], the author distinguishes two traditional problems, namely, the simple assembly line balancing problem SALBP and the general assembly line balancing problem GALBP. The exact methods are mainly based on Branch & Bound and dynamic programming approaches, see [1], [10], [23]. The ALBP are NP-hard [8]. Therefore, much research has been generated to solve the problem by developing heuristic techniques [3], [16], [18].

Among the extensions proposed for the SALBP, two are related to our problem, namely the use of parallel machines and the consideration of setup times between operations. We will now present the principal research work which has considered these two extensions.

[11] was the first to introduce the use of parallel machines for ALBP. Parallel machines execute the same operations on different parts of the product. The author presented the advantages of such a line: reduced non-productive time (idle time), cycle time respected (when there are operations with an operational time exceeding the cycle time), improved production output imposed by the longest operation and, finally, transport time and matter flows on the line are reduced. In [19], balancing problem with paralleling of workstations is considered. A resolution approach for this problem with a branch and bound algorithm is proposed. The paper [5] proposed an extension of the problem considering dead times. A heuristic approach was proposed in [4] for the multi-product line balancing problem with parallel workstations. The article [17] studied the multi-objective assembly line balancing problem with stochastic operational times and parallel machines. In [10] the authors proposed another interesting approach for assembly line design with parallel stations.

Setup times between operations were examined at first in [26] as tool changes times; the author proposed a column generation approach to solve this problem. Since this publication, there were a few articles on this subject, we can only mention [24], which defined the sequence-dependent assembly line problem (SDALBP), setup times varying according to the sequence in which the operation was processed, and [2], which notably proposed a heuristic approach for this problem.

Thus, the line balancing problems with parallel stations have been extensively studied in the literature since the publication of [19]. Several methods have been developed. At the same time, very little research has been devoted to the assembly line balancing problem with sequence dependent setup-times. In this paper, we explore a novel problem in which we treat both paralleling of stations and sequence-dependent setup times.

Another family of line balancing problems was described for machining lines used in mass production. These lines are composed of multi-spindle machines where several operations are executed simultaneously using special equipment (multi-spindle heads). The problem is thus to chose spindle heads and to assign them to workstation minimizing the total cost (cost of spindle heads and stations) while executing all the operations and respecting all the constraints. In [14] a shortest path algorithm was proposed for the case where all the available spindle heads are known beforehand. For the same problem, in [7] two MIP models were suggested and tested. In [13], the same type of problem is considered but where the set of all possible spindle heads is not known before optimization. A MIP model and a decomposition heuristic was proposed. In [12], for the same problem two heuristics were developed. These transfer line balancing problems are different from that considered in this paper, because there were not parallel machines and setup times before. However, in the problem considered in this paper, there are no multi-spindle heads. The operations are executed sequentially one by one. Finally, the use of CNC machines is usual for flexible machining lines, but most research papers in this field focus on the evaluation of the flexibility obtained (see e.g. [21]) rather than on the balancing problem.

3. Problem Statement

We consider the following line balancing problem. The set of operations ($N, i = 1,2,\ldots,|N|$) is determined by the processing plan for the product for which the line will be designed. A part to be machined passes through a sequence of workstations (k_0) in the order of their installation. Each workstation is provided with at least one machine which carries out within the line cycle time (T_0) the set of operations $N(k)$ (operations assigned to the station k). In the case where a workstation workload exceeds the line cycle time, identical parallel machines can be installed. Then, the local cycle time ($T_{0,k}$) is equal to the number of parallel machines multiplied by the line cycle time. The same operations are duplicated and processed in parallel but on different machines and items.

There are constraints on the assignment of operations, namely:

- The precedence constraints: relations of order between operations. These relations define feasible sequences of operations. We consider P_i^* as the set of all predecessors (direct and indirect) of operation i;
- The exclusion constraints which prohibit assigning two operations to the same station (technological incompatibility), where $\overline{ES}$ is a set of pairs (i, j) (or (j,i)) of operations which cannot be assigned to the same workstation;
- The inclusion constraints: the need to carry out fixed groups of operations on the same station, where ES is a collection of subsets e ($e \subset N$) of operations which must be imperatively assigned to the same workstation;
- The sequence dependent setup time: setup time depends on the sequence in which the operations are assigned. t_{ij} is the additional setup time for the execution of operation j after operation i on the same workstation.

4. Heuristic Method

In this section, we propose a heuristic method for the resolution of the considered problem. This is a two-phase heuristic: in the first phase (Algorithm 1), we create the stations and assign operations while satisfying all the constraints of the problem using a greedy algorithm; in the second phase (Algorithm 2), we regroup stations to minimize the total number of stations while taking into account the exclusion constraints of the problem. We will introduce the following additional notations:

- M is the subset of operations which have not yet been assigned;
- T_k is the slack time (available time) of station k;
- Op is the selected operation to assign to the current station;
- t_{si} is the setup time which is necessary for the execution of operation i in the sequence in which it's assigned;
- $N_{machine}(k)$ is the number of machines in station k;
- μ_0 is the limit on utilization rate of stations (maximum authorized rate).

The "current station" is where we assign operations in the current step of the algorithm. As long as the current station is not entirely filed, we cannot open a new station. If a station is opened, it becomes the current station. The proposed algorithms utilize three lists of operations:

- Lp: subset of operations of M which haven't predecessors not assigned;
- Lf: subset of operations of Lp which can be assigned to the current station; i.e.: i) the needed time to the execution of the operation (operational time plus the corresponding set-up time) is less than

the available workload time of the current station, ii) at least one of the possible positions of the operation is available on the current station, iii) at least one possible type for the current station is compatible with the operation assignment, iv) there is no exclusion constraints between the operation and the already assigned operations of the station;

- LI: subset that includes all the operations which must be assigned on the same station with operation i (inclusion constraints) and their predecessors of the subset M.

4.1 Algorithm 1

The first algorithm starts and creates a station. At each iteration, the set Lp (Step 3) is updated. This set contains only operations with all predecessors assigned or none at all. Afterwards, the set Lf (Step 4) is created. It is composed of operations which can be assigned to the current station, i.e. the operational times are less than the station slack time and they do not have exclusion constraints with the operations already affected to this station. If Lf is empty (Step 5), a new station is created, Lp and Lf are updated. If Lf is not empty, it selects the operation which has the longest operational time. Let Op be this operation (Step 6). Then, the set of operations LI is defined (Step 7). This set contains operations which must be processed on the same station with Op and all their predecessors not yet assigned (For example, if operation i is selected as Op with i and j that belong to the same inclusion constraint, and with l a predecessor of j which is not already assigned, then the set $LI = \{i, j, l\}$ is defined). Then, the operations of LI are assigned to the current station (Step 9-12). If the current station is full (slack time is less than the minimum operation time), but the remaining operations of LI must be assigned to the current station (for example, operation i has an inclusion constraint with j already assigned, therefore we cannot assign i to another station). In this case, a parallel machine to the current station is added (Step 10). Otherwise, a new station is created. This algorithm is processed every time an operation Op is selected to be assigned, until all operations of the set LI have been assigned.

Algorithm 1

```
Begin
1:  $k \leftarrow 0$, $M \leftarrow N$;
2:  Create station: $k \leftarrow k+1$,
    $N_{machine}(k) \leftarrow 1$, $T_{0,k} \leftarrow T_0 * \mu_0$,
    $T_k \leftarrow T_{0,k}$, $N(k) \leftarrow \varnothing$, $Lp \leftarrow \varnothing$,
    $Lf \leftarrow \varnothing$;
3:  $Lp \leftarrow \{ i \in M \mid P_i^* \cap M = \varnothing \}$;
    If $Lp = \varnothing$ Then Go to END;
4:  $Lf \leftarrow \{ i \in Lp \| (t_j + t_{sj} \le T_k)$
            $\text{and } \nexists j \in N(k), (i,j) \in \overline{ES} \}$;
5:  If $Lf = \varnothing$ Then Go to Step 2;
6:  Select operation $Op$ in $Lf$,
    $Op \in \left\{ i \in Lf \,\middle|\, t_i = \max_{j \in Lf} \{t_j\} \right\}$;
7:  $LI \leftarrow \{ i \in M \mid \exists e \in ES, e = (i, Op) \}$;
    $LI \leftarrow LI \cup \left\{ \bigcup_{i \in LI} \{P_i^*\} \right\} \cap M$;
8:  If $\exists i \in LI, j \in N(k) \mid (i,j) \in \overline{ES}$ Then
    $Lf \leftarrow Lf \setminus Op$ And Go to Step 5;
9:  Select operation $I$ in $LI$,
    $I \in \{ i \in LI \mid t_i = \max_{j \in LI \mid P_j^* = \varnothing} \{t_j\}$
            $\text{and } P_i^* = \varnothing \}$;
10: If $T_k < (t_I + t_{sI})$ Then
        If $\exists i \in LI, j \in N(k), e \in ES$
            $\|(i,j) \in e$ Then add one
        machine to station $k$:
        $N_{machine}(k) \leftarrow N_{machine}(k) + 1$,
        $T_{0,k} \leftarrow T_{0,k} + T_0 * \mu_0$,
        $T_k \leftarrow T_k + T_0 * \mu_0$;
        Else $Lf \leftarrow Lf \setminus \{Op\}$ And Go to
        Step 5;
11: $N(k) \leftarrow N(k) \cup \{I\}$,
    $T_k \leftarrow T_k - (t_I + t_{sI})$, $LI \leftarrow LI \setminus \{I\}$,
    $M \leftarrow M \setminus \{I\}$;
12: If $LI = \varnothing$ Go to Step 3 Else Go to Step 9;
END
```

4.2 Algorithm 2

After balancing line with Algorithm 1, we try to minimize the number of stations while grouping stations by using the following second algorithm. The entire sequence of operations assigned to the stations is examined. The algorithm verifies if there are exclusion constraints between the sub-sequences of two successive stations. If so, the algorithm goes to the next station, otherwise these two stations are grouped together and the parameters of the line are updated: the local cycle time of the station, number of machines at the station, and sequence of assigned operations, etc. At the end of the iteration, the new stations verify all the initial constraints of the problem. A new iteration begins from the first station of the line. The algorithm stops when we have an iteration without line modifications.

Algorithm 2

```
Begin
1:  $k \leftarrow 1$;
2:  If $\exists i \in N(k), j \in N(k+1), (i,j) \in \overline{ES}$
    Then $k \leftarrow k+1$;
    Else
        Merge stations $k$ and $k+1$,
        $N(k) \leftarrow N(k) \cup N(k+1)$;
        Update $T_k$, $N_{machine}(k)$ and
        $T_{0,k}$;
        Delete station $k$,
        For $\lambda \in \{k+1, \ldots, k_0 - 1\}$ Do
            $N(\lambda) \leftarrow N(\lambda+1)$,
            $T_\lambda \leftarrow T_{\lambda+1}$;
            Update $N_{machine}(\lambda)$
            and $T_{0,\lambda}$;
        $k_0 \leftarrow k_0 - 1$;
3:  If $k < k_0$ Then Go to Step 2;
END
```

4.3 Illustrative example

In order to illustrate the suggested procedures, we present a numerical example with 10 operations ($T_0 = 40$, μ_0=0.7). Figure 1 gives the precedence graph and the times of the operations.

The inclusion constraints are:

$\{(2,4);(8,9);(5,6)\}$,

and the exclusion constraints are:

$\{(2,7);(3,4)\}$.

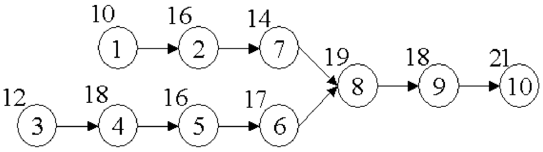

Figure 1. Precedence graph

First, we apply Algorithm 1 to this example. The solution obtained is composed of five stations. Only one station (Station 4) contains two machines, the other stations containing only one machine. The total unproductive time is equal to 76 units of time, and there are 3 stations which have a utilization rate less than the threshold rate (maximum limit), see Table 1. Afterwards, we apply Algorithm 2. The new solution is composed only of 4 stations and 5 machines. The total unproductive time is equal to 36 units of time, and there is only one station where the utilization rate is less than the threshold, see Table 1.

5. Experimental Study

In order to evaluate the performance of the developed heuristics, the two algorithms have been implemented and applied on different instances of the studied problem.

Table 1: Experimental results

	Algorithm 1					Algorithm 2			
Station	1	2	3	4	5	1	2	3	4
Number of machines	1	1	1	2	1	1	1	1	3
Work time	23.5	39	35.5	60.7	21	23.5	39	35.5	84.4
Unproductive time	16.5	1	4.5	19.3	19	16.5	1	4.5	35.6
Rate of utilization	0.59	0.98	0.89	0.75	0.53	0.59	0.98	0.89	0.71

19 instances have been randomly generated with characteristics close to those of actual industrial problems. To generate these instances, we have considered different numbers of operations (between 10 and 100). Concerning the numbers of constraints, we have used the following measure of constraint density: $Dp = (2 * z)/(N * (N-1))$, where z is the number of constraints. We have set the densities of precedence, inclusion and exclusion constraints to 5%, 2% and 3%, respectively.

All these instances have been solved on a SUN UltraSPARC IIIi with 1593 Mhz CPU and 16 GB of memory. The computational times are very low (less than 0.5 second per instance). The impact of the second algorithm on, respectively, the number of stations and the number of machines is significant. Actually, when two stations are grouped, the sum of their idle times can exceed the cycle time of the line and thus permits to remove a machine. However, if the cycle time is respected for each station, the utilization rate can be very high for some stations. Taking into account the time of transport along stations and possible random variations on operations times, it is necessary to adopt a margin by considering the cycle time in the model shorter that the objective line cycle time.

We test the model for various values of this margin: the cycle time considered by the heuristic varied between 85% and 100% of the objective cycle time. Indeed, the station can only be fully loaded when we accept 100% cycle time.

Figure 2 presents the total number of machines obtained for four different values (i.e. the cycle time of the model is of 85%, 90%, 95% and 100% of the objective cycle time). Obviously, the total number of machines increases with the margin. However, it is interesting to note that the difference is generally low (less than 3 machines) and does not increase significantly with the size of instances. There is no appreciable difference concerning the number of stations. This slight increase of the number of machines can lead to a significant improvement concerning the utilization rate for some stations. Figure 3 shows the repartition of these rates for each value of the

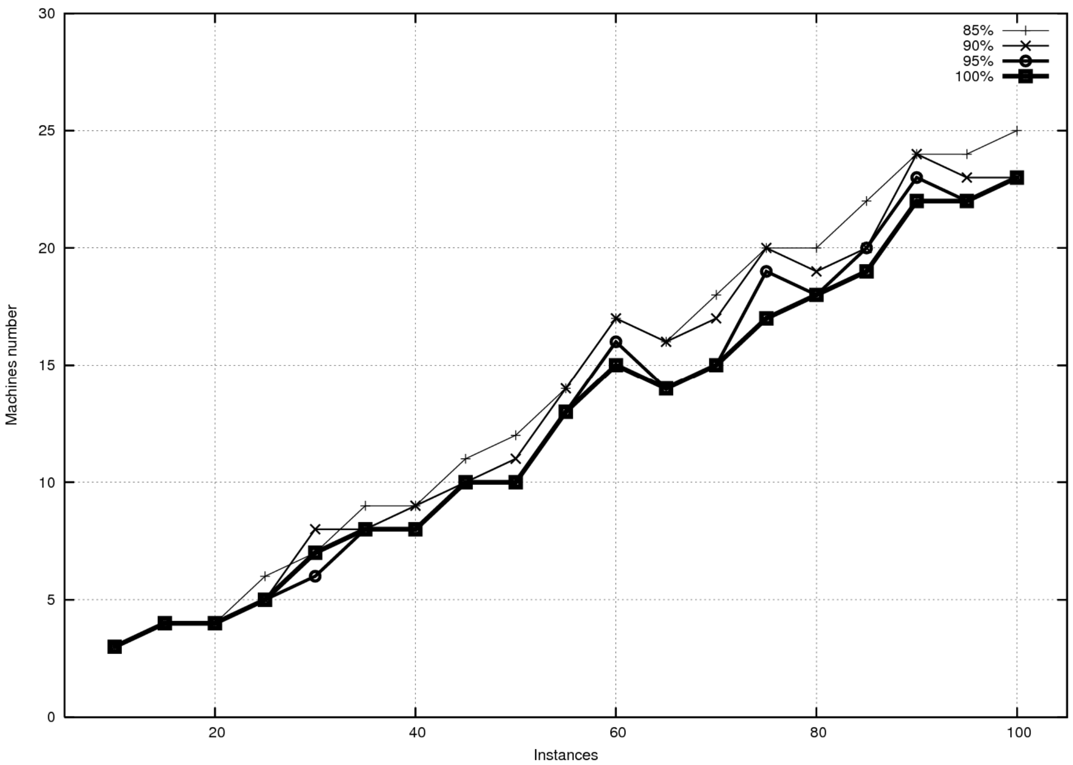

Figure 2. Total number of machines for different margins for the cycle time

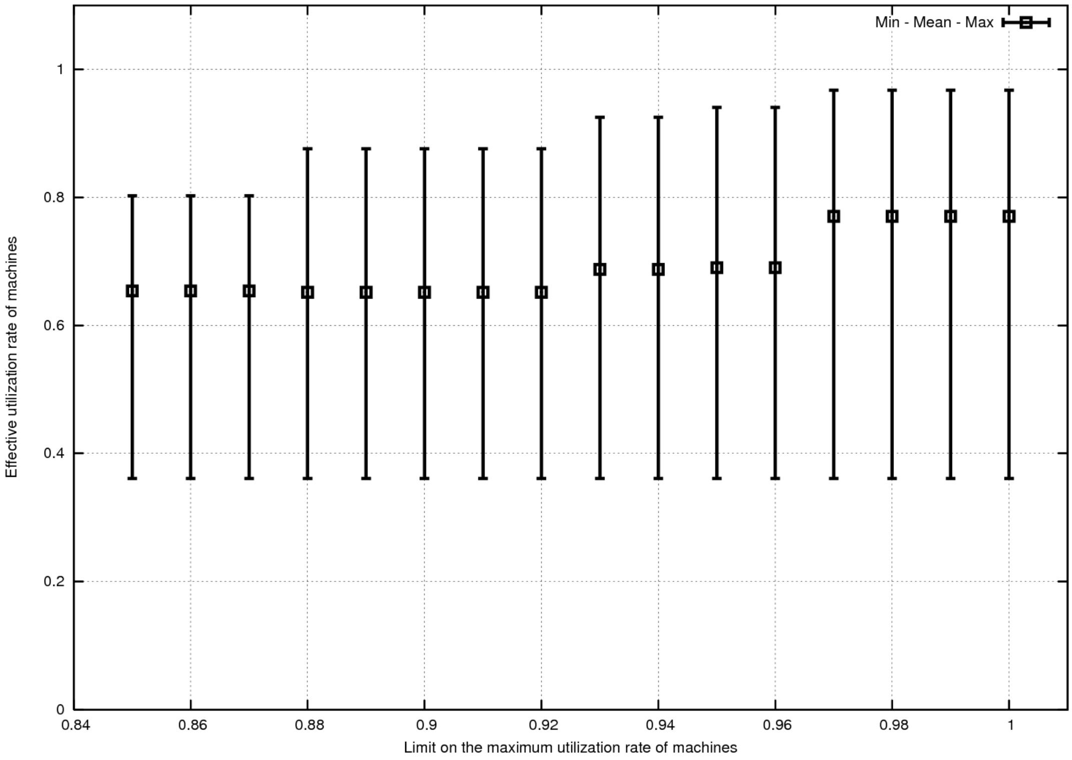

Figure 3. Utilization rates of machines for the instance with 75 operations

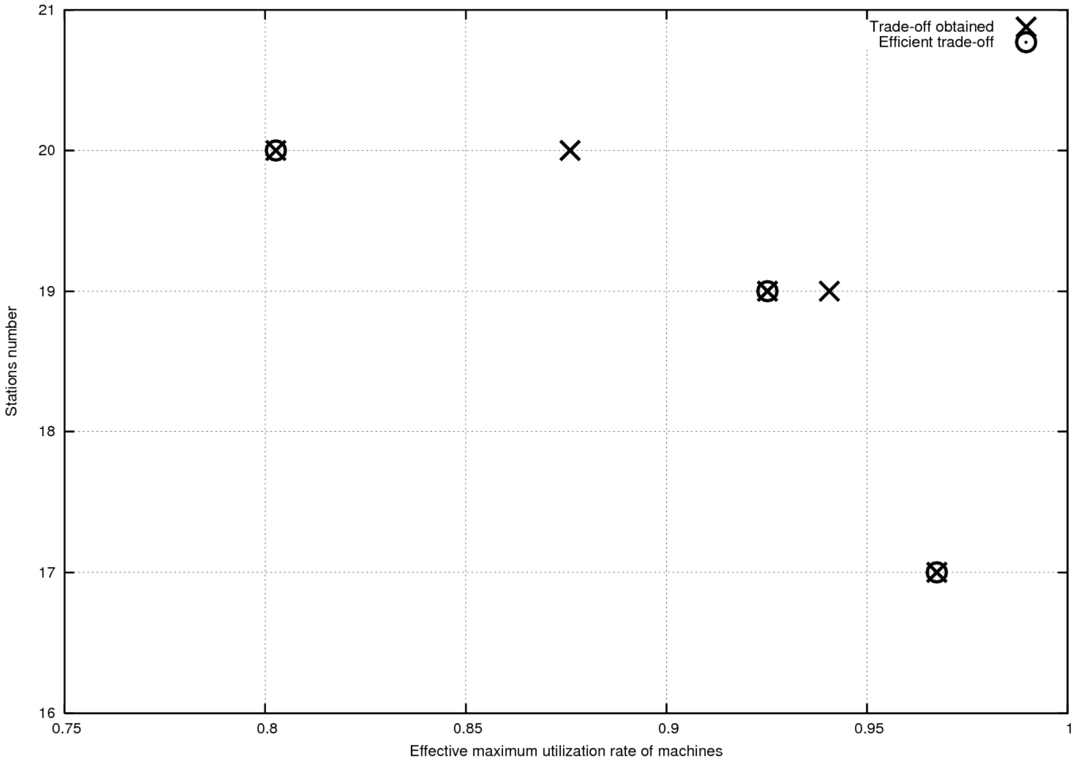

Figure 3. Trade-off between maximum utilization rate and number of stations for the instance with 75

maximum utilization rate limit. The intervals include the values for different machines. The utilization rates of the four highest loaded machines decrease when we change the value of maximum limit on the utilization rates from 85% to 100%.

These heuristics are not time consuming. Therefore, they can be easily used to calculate different trade-offs between the number of stations and the maximum authorized utilization rate (maximum utilization rate limit), see Figure 3. For example, we have obtained for this instance 5 trades-offs, where 3 of them are efficient. The 2 others do not appear to be interesting since they necessitate a larger maximum utilization rate to obtain the same number of stations.

6. Conclusion and Perspectives

A heuristic approach for balancing machining lines with parallel stations and sequence-dependent setup times is presented. In such a line, the parts to be machined pass through a sequence of serial workstations. The line is paced. Parallel machines can be installed at each workstation. The same operations are duplicated and processed in parallel but on different machines and different items. Setup times between operations are considered, they depend on the sequence in which operations are processed. This balancing problem consists in assigning operations to a sequence of workstations defining the number of parallel machines installed at each workstation while minimizing the number of machines used. A new heuristic has been proposed for this problem, and computational results are reported. A first study of the possible trade-offs between the cycle time and the number of machines has also been presented. Future research will investigate further these trade-offs. The integration in a decision support system for the resolution of industrial problems will be proposed then. Also, these heuristics could provide a good upper bound on the objective function for the problem considered and might be used within an exact resolution method. A possible improvement of the proposed heuristic could be inspired by COMSOAL (Computer Method of Sequencing Operations for Assembly Lines) heuristic, see [3]. COMSOAL is a multi-pass iterative method which generates the solutions using a random assignment of operations to workstations. This approach was already used for balancing machining lines with multi-spindle heads, see [12]. Future development of the heuristic using a more powerful metaheuristic framework (see e.g. [25]) could improve its performance as well.

Acknowledgements

The authors would like to thank the LT' 2007 program committee for recommending our work to *Studies in Informatics and Control* Journal. They also appreciate the collaboration of PCI-SCEMM.

REFERENCES

1. AMEN, M., **Heuristic Methods for Cost-oriented Assembly Line Balancing: a Comparison on Solution Quality and Computing Time**, International Journal of Production Economics, vol. 69, 2001, pp. 255–264.

2. ANDRÉS, C., C. MIRALLES, R. PASTOR, **Balancing and Scheduling Tasks in Assembly Lines with Sequence-dependent Setup Times**, European Journal of Operational Research, vol. 187, 2008, pp. 1212–1223.

3. ARCUS, A. L., COMSOAL: **A Computer Method of Sequencing Operations for Assembly Lines**, International Journal of Production Research, vol. 4, 1966, pp. 259–277.

4. ASKIN, R. G., M. ZHOU, **A Parallel Station Heuristic for the Mixed-model Production Line Balancing Problem**, International Journal of Production Research, vol. 35, nr. 11, 1997, pp. 3095–3105.

5. BARD, J. F., **Assembly Line Balancing with Parallel Workstations and Dead Time**, International Journal of Production Research, vol. 27, nr. 6, 1989, pp. 1005–1018.

6. BAYBARS, I., **A Survey of Exact Algorithms for the Simple Assembly Line Balancing Problem**, Management

Science, vol. 32, nr. 8, 1986, pp. 909–932.

7. BELMOKHTAR, S., A. DOLGUI, N. GUSCHINSKY, G. LEVIN, **An Integer Programming Model for Logical Layout Design of Modular Machining Lines**, Computers and Industrial Engineering, vol. 51, nr. 3, 2006, pp. 502–518.

8. BHATTACHAJEE, T. K., S. SAHU, **Complexity of Single Model Assembly Line Balancing Problems**, Engineering Costs and Production Economics, vol. 18, 1990, pp. 203–214.

9. BOYSON, N., M. FLIEDNER, A. SCHOLL, **A Classification of Assembly Line Balancing Problems**, European Journal of Operational Research, vol. 183, 2007, pp. 674–693.

10. BUKCHIN, J., J. RUBINOVITZ, **A Weighted Approach for Assembly Line Design with Station Paralleling and Equipment Selection**, IIE Transactions, vol. 35, 2002, pp. 73–85.

11. BUXEY, G. M., **Assembly Line Balancing with Multiple Stations**, Management Science, vol. 20, 1974, pp. 1010–1021.

12. DOLGUI, A., B. FINEL, N. GUSCHINSKY, G. LEVIN, F. VERNADAT, **An Heuristic Approach for Transfer Lines Balancing**, Journal of Intelligent Manufacturing, vol. 16, nr. 2, 2005, pp. 159–171.

13. DOLGUI, A., B. FINEL, N. GUSCHINSKY, G. LEVIN, F. VERNADAT, **MIP Approach to Balancing Transfer Lines with Blocks of Parallel Operations**, IIE Transactions, vol. 38, 2006, pp. 869–882.

14. DOLGUI, A., N. GUSCHINSKY, G. LEVIN, **A Special Case of Transfer Lines Balancing by Graph Approach**, European Journal of Operational Research, vol. 168, nr. 3, 2006, pp. 732–746.

15. DOLGUI, A., J. M. PROTH, **Les systèmes de production modernes**, Hermes, 2006.

16. HELGESON, W. P., D. P. BIRNIE, **Assembly Line Balancing Using the Ranked Positional Weight Technique**, Journal of Industrial Engineering, vol. 12, 1961, pp. 394–398.

17. MCMULLEN, P. R., G. V. FRAZIER, **Using Simulated Annealing to Solve a Multiobjective Assembly Line Balancing Problem with Parallel Workstations**, International Journal of Production Research, vol. 39, nr. 10, 1998, pp. 2717–2741.

18. MOODI, C. L., H. H. YOUNG, **A Heuristic Method for Assembly Line Balancing for Assumption of Constant or Variable Elements Time**, Journal of Industrial Engineering, vol. 16, 1965, pp. 23–29.

19. PINTO, P. A., D. G. DANNENBRING, B. M. KHUMAWALA, **A Branch and Bound Algorithm for Assembly Line Balancing with Paralleling**, International Journal of Production Research, vol. 13, nr. 2, 1975, pp. 183–196.

20. REKIEK, B., A. DOLGUI, A. DECHAMBRE, A. BRATCU, **State of Art of Assembly Lines Design Optimization**, Annual Reviews in Control, vol. 26, nr. 2, 2002, pp. 163–174.

21. WADHWA, S., Y. DUCQ, M. ALI, A. PRAKASH, **Performance Analysis of a Flexible Manufacturing System under Planning and Control Strategies**, Studies in Informatics and Control, vol. 17, nr. 3, 2008, pp. 273–284.

22. SALVESON, M. E., **The Assembly Line Balancing Problem**, Journal of Industrial Engineering, vol. 6(4), 1955.

23. SCHOLL, A., **Balancing and Sequencing Assembly Lines**, Physica-Verlag Heidelberg, 1999.

24. SCHOLL, A., N. BOYSEN, M. FLIEDNER, **The Sequence-dependent**

Assembly Line Balancing Problem, Operation Research Spectrum, vol. 30, nr. 3, 2008, pp. 579–609.

25. TANGOUR, F., P. BORNE, **Presentation of Some Metaheuristics for the Optimization of Complex Systems**, Studies in Informatics and Control, vol. 17, nr. 2, 2008, pp. 169–180.

26. WILHELM, W. E., **A Column-generation Approach for the Assembly System Design Problem with Tool Changes**, The International Journal of Flexible Manufacturing Systems, vol. 11, 1999, pp. 177–205.