A Heuristic to Minimize the Equipment Cost for Large Transfer Lines with Operations Parallelization

Antoneta Iuliana Bratcu
Advanced Control Systems Research Centre
"Dunărea de Jos" University of Galați
111, Domnească, RO-800021 Galați
Romania
Antoneta.Bratcu@ugal.ro

Abstract – **This paper approaches the design of single product lines without buffers (transfer lines), disposing of several types of multifunctional equipment, which allow the parallel execution of many different tasks by the same unit (blocks of tasks). A multi-criteria optimization is aimed at: balancing the line and minimizing the equipment cost. This problem is decomposed into two sub-problems: first, obtain the blocks by assigning the tasks to equipments, subject to the precedence constraints; secondly, these blocks are assigned to workstations, subject to the cycle time constraint, which is just the ALB problem. In a previous paper a branch-and-bound (optimal) algorithm, enhanced by using a lower bound and a dominance rule, was proposed to solve the first sub-problem. This paper enriches this algorithm with a (suboptimal) heuristic for dealing with large scale problems.**

I. INTRODUCTION

The production lines approached in this paper are designed for machining a type of product (or a family of products) [1]; the absence of buffers gives the name of "transfer lines". Each workstation is equipped with several types of equipment and must perform a certain set of tasks. Grouping the tasks by subsets allows to using dedicated equipment. The tasks assigned to the same equipment are realized *simultaneously* (*in parallel*) and form a *block*. Therefore, the time of a block is the maximal task time for the tasks assigned to the corresponding equipment. The workstation time is equal to the sum of the block times. The cycle time is the time of the bottleneck workstation.

The several types of equipment have different costs and are able to perform some (or all) of the tasks. Therefore, the processing time of a given task depends on the chosen equipment. For equipments, their costs and task times are known.

For a transfer line, the design goal is a desired productivity (predetermined cycle time). Given a product, all the precedence constraints between tasks are known. So, for this type of transfer lines, at the preliminary design stage, it is necessary to find a good allocation of tasks by blocks and workstations, along with the equipment selection for the blocks. Therefore, under the constraint of an imposed cycle time, the design problem can be stated as to find the cheapest equipment-to-workstations assignment, meanwhile establishing which tasks should be assigned in which of the workstations and blocks.

In the Simple Assembly Line Balancing (SALB) problem [2], no alternative equipment is considered, the goal being the balancing of the stations' workloads in the case of *sequential* execution of tasks. The same problem for the transfer lines is studied in [1].

The problem of Resource Planning (RP) – standing for the equipment selection – was considered while ignoring the precedence constraints [3]. The problem was optimally solved using linear programming techniques and branch-and-bound algorithms, and next enriched with heuristics and extended to the case of multiple product lines [4], [5]. Meta-heuristics have also been proposed to deal with "resource dependent task times", like, for example, genetic algorithms [6]. In [7], the design of flexible assembly lines with multiple equipment variants, taking into account the precedence constraints, is solved by a branch-and-bound algorithm. But in [7] each station has a single piece of equipment and the tasks are sequential. A design issue in the case of stations paralleling is analyzed in [8].

The same problem for the case of parallel execution of tasks is approached in [9], where the global optimization goal for the considered type of lines – the line balancing and the resource cost minimization – is split in two. First, the blocks result by assigning the tasks to equipments, subject to the precedence constraints, and then the blocks are assigned to workstations, subject to the cycle time constraint. Differently from [1], two tasks can be performed in parallel (by the same equipment, in the same time) if and only if there is not any precedence relation between them. It is said that they are *compatible* (or *indifferent* [10]). A branch-and-bound algorithm is proposed to solve the first sub-problem (the second one being just the well-known SALB problem). Because of the decomposition, the final solution will be not necessarily optimal, but well balanced and cheap.

This paper continues the work presented in [9], by proposing a heuristic method for the first sub-problem, dedicated to large scale cases.

The paper is organized as follows. Section II summarizes the results from [9]: the two sub-problems are described and the main steps of the proposed branch-and-bound algorithm are listed. Section III presents the optimal algorithm results for a middle scale case. The heuristic proposed for solving large scale problems is described in section IV and its effectiveness is analyzed in relation with the optimal results in section V; an example is also provided. Some concluding remarks are listed in section VI, ending this paper.

II. PROBLEM DESCRIPTION AND OPTIMAL BRANCH-AND-BOUND SOLVING ALGORITHM

A. First sub-problem: producing the blocks of parallel tasks

Each block of parallel tasks is performed by a single unit of equipment. Each task can be performed by several equipments. The cost of a unit from each type of equipment and the processing time of each task by each type of equipment are known. There are no constraints on the availability of units of each type (they are supposed infinitely available). If task i cannot be performed by equipment j, then it is supposed that the respective processing time, t_{ij}, overflows the cycle time of the line, T_0, and is set to infinity.

Parallelization of tasks is possible if and only if there is no precedence relation between them. In this paper, this situation is described by the *compatibility* relation (and the corresponding non-oriented graph, complementary with the precedence graph). Therefore, a set of tasks can be paralleled iff they are two-by-two compatible. Because of parallelization, the processing time of a block will be the *maximum* of the tasks durations; so, the cycle time constraint may be considered implicitly met for each task.

In the sequel, the following notations will be used: n is the number of tasks, m is the number of equipment types (tools), EC_j the cost of a unit of type j (j=1...m) and t_{ij} the processing time of task i (i=1...n) by equipment j.

The integer programming formulation of the first sub-problem can be found in [9].

B. Second sub-problem: assigning the blocks to workstations

A solution of the first sub-problem serves as input data for the second one. Let K be the number of blocks. Given the manner of parallelisation, one can easily deduce that these blocks are *totally ordered* [10]. They must now be grouped into the minimum number of workstations, subject to the cycle time constraint and without violating the total order between them. Execution of blocks inside of a workstation is sequential. Therefore, this is a SALB problem, for which minimizing the stations' number, M, is the same as minimizing the total idle time, $\text{Min}\left\{M \cdot T_0 - \sum_{i=1}^{K} tb_i\right\}$, where tb_i is the processing time of block i.

The second sub-problem admits a linear programming formulation, which is presented in [9].

C. Branch-and-bound algorithm

The first sub-problem is much less approached in the literature than the second one (identified as the SALB problem). To solve it, a branch-and-bound algorithm was proposed, which is resumed next. The algorithm uses a lower bound and a dominance rule. To introduce the lower bound, let some new variables be introduced:

$$y_j = \sum_{k=1}^{n} y_{jk} = \text{total number of type } j \text{ tools units}$$

$$x_{ij} = \sum_{k=1}^{n} x_{ijk} = \begin{cases} 1, & \text{if task } i \text{ is executed by tool } j \\ 0, & \text{otherwise} \end{cases}$$

If one chooses:

$$x_{ij} = \begin{cases} 1, & \text{if } EC_j = \min_l \left\{ EC_l \mid t_{il} \le T_0 \right\} \\ 0, & \text{otherwise} \end{cases}, \tag{1}$$

$$y_j = \max_{i=1...n} \left\{ x_{ij} \right\}, \quad \forall j, \tag{2}$$

then the quantity:

$$LB_1 = \sum_{j=1}^{m} EC_j \cdot y_j \tag{3}$$

is a lower bound to the value of the first sub-problem. The proof of this statement is presented in [9].

The scope of the B&B algorithm is to optimally select the equipment for the *parallel* execution of tasks.

The nodes of the B&B tree represent partial solutions of the first sub-problem. Each node is thus associated with a set of already assigned tasks, σ, grouped by blocks, each block having assigned the equipment selected to perform it. Among the blocks, only the last one may still allow new assignments. A task is a *candidate* to be assigned to the last block if: 1) the task has no predecessors or they all have been already assigned, and 2) the task is compatible with all the tasks assigned to the (last) block.

If the set of candidate tasks is not empty, the block is said *open*; otherwise, it is said *closed*. These definitions are extended to the corresponding node. The lower bound of a node X is defined by:

$$LB_X = TC_X + LB_{1(\sigma')},$$

where TC_X is the total cost of equipments assigned to closed blocks and σ' is the complement of σ in the original set of tasks. The first element reflects the past assignment decisions, whereas the second concerns future decisions. The main stages of the B&B algorithm are:

1. *Creation of the first level* of the B&B tree: a node is created for each task without predecessors, assigned to a tool capable to perform it.

2. *Selection of a node for extension*: from the nodes without descendants, the one with the lowest lower bound is chosen for extension.

3. *Extension*: each descendant contains the assignment of a new single task; if the last block is open, the extension concerns all candidate tasks; if it is closed, a new block is opened and the extension is made for every feasible pair of a tool and an unassigned task.

4. *Elimination of dominated nodes*: each time a block is closed, the current node is compared with the other nodes having the last block closed, using the following dominance rule: if $\sigma(X) \supseteq \sigma(Y)$ and $TC_X \le TC_Y$ (where $\sigma(\cdot)$ denotes the set of already assigned tasks), then node Y is dominated and will be eliminated.

5. *Stop condition*: an optimal solution has been found if an extended node contains all the tasks and its solution value is no larger than the lower bound of any of the nodes from the last level.

The presented algorithm is an optimal one; therefore, it is expected to behave worse as the problem scale grows. The solving of a middle scale problem is illustrated next.

III. MIDDLE SCALE EXAMPLE

This example concerns a product obtained according to a precedence graph of fifteen tasks (see Fig. 1), using four multifunctional equipments. Table I contains the costs per unit for each type of equipment and the durations of performing each task by each type of equipment (grey cases denote impossibility).

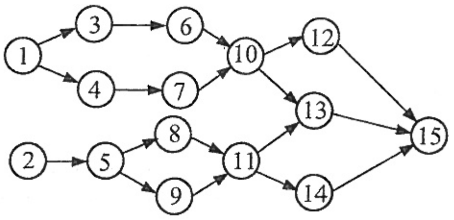

Fig. 1 Precedence graph for a middle scale example (n=15 tasks)

TABLE I

TASK TIMES DEPENDING ON THE EQUIPMENT IN A MIDDLE SCALE CASE (n=15, m=4)

Equipment	E_1	E_2	E_3	E_4
Cost/unit	10000	7500	6000	8500
Task 1	10	8		9
Task 2	20	18	17	19
Task 3	27			29
Task 4	18	19		19
Task 5	14	10	8	
Task 6	15	11	8	16
Task 7	19	16	14	
Task 8	9	8	7	8
Task 9	48		49	45
Task 10	12	11		13
Task 11	10	10		11
Task 12	11			11
Task 13	12	10	9	9
Task 14	10	7	6	
Task 15	10		14	13

The final solution is presented in Fig. 2. The blocks – each representing one type of equipment – were subsequently grouped into workstations, subject to a cycle time of T_0=50 time units (t.u.). The cost of the optimal solution is 53500 and the total idle time of the line is 21 t.u. Obviously, the optimal solution contains those types of equipment which achieve good trade-offs between cost and flexibility (note, on the contrary, the absence of the fourth type, whose flexibility does not therefore justify its cost).

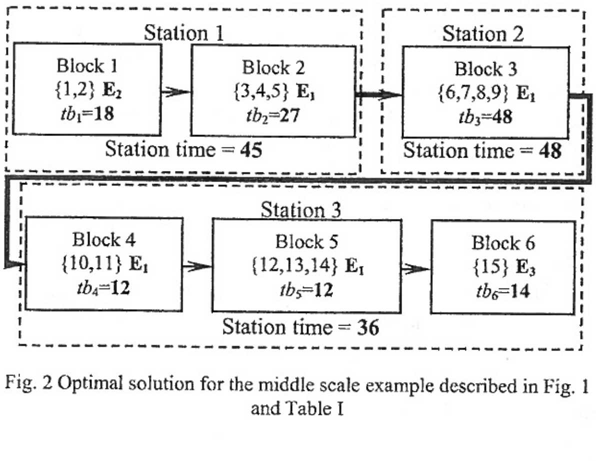

Fig. 2 Optimal solution for the middle scale example described in Fig. 1 and Table I

IV. HEURISTIC FOR SOLVING LARGE PROBLEMS

As it is designed to provide the optimal solution, the presented B&B algorithm requires large computer resources to solve very large problems. Although an exact study on the impact of the main variation parameters [7] has not been done, the algorithm complexity dependence on these parameters may be qualitatively described as follows:

a) n, the number of tasks – its growth generally induces an increased complexity and execution time;

b) m, the *equipment alternatives* – with the same kind of influence as n;

c) the *variability of task duration* has the less predictable influence; thus, a small variability can lead to a large size of the B&B tree, the most probable because the number of candidate tasks at each step grows;

d) *F-ratio* is an inverse measure of the precedence graph's flexibility and is defined as [11]:

$$F\text{-}ratio = \frac{2Z}{n(n-1)} \in [0;1],$$

where Z is the number of zeroes in the precedence matrix, P, of the precedence graph:

$$P_{ij} = \begin{cases} 1, & \text{if task } i \text{ precedes task } j \\ 0, & \text{otherwise} \end{cases};$$

the F-ratio contributes negatively to the algorithm complexity, that is, a low F-ratio (large flexibility in creating assembly sequences, equivalent with many two-by-two compatible tasks, which is the case in the most real world problems) leads to many nodes in the B&B tree;

e) *E-ratio* measures the equipment flexibility and is defined as [12]:

$$E\text{-}ratio = 1 - \frac{I}{n(m-1)} \in [0;1],$$

where I is the number of elements set to infinity in the matrix of task processing times by each type of equipment (for example, I is the number of grey cases of Table I); a large E-ratio (close to 1) – meaning a large equipment flexibility – induces the growth of the algorithm complexity.

The contribution of this paper consists in proposing a heuristic for dealing with the large problems, mostly encountered in the real world.

Let the second step of the B&B algorithm be reconsidered; it concerns the selection of the node with the smallest lower bound for extension. But some of these nodes have a very small probability of leading to the optimal solution, their extension being important for proving the optimality of the solution. The proposed heuristic concerns the modification of the node selection rule.

Let Y be an open node at the tree level N_Y, with a lower bound LB_Y. Note that the tree levels are numbered such that the root is placed on level 0 and the highest index level is n, where all tasks are assigned. In the heuristic, the probability of choosing node Y for extension must reflect increased chances of nodes with large N_Y to be chosen. Therefore, this probability, $p(Y)$, can be measured by the ratio N_Y/LB_Y.

Let now X be the node with the smallest lower bound, LB_X, situated on the tree level N_X. The node selection is modified as follows:

If $\frac{N_X}{LB_X} > K \cdot \max_{Y \neq X}\left\{\frac{N_Y}{LB_Y}\right\}$, then select node Y, otherwise select node X.

Parameter K is the heuristic's control parameter; it is expected that, more the value of K grows, more the solution provided is far from optimality. The choice of K is discussed in the next section.

V. LARGE SCALE EXAMPLE AND CHOICE OF THE HEURISTIC'S CONTROL PARAMETER

First, an example of large scale problem is provided. It is the case of a product assembled according to a precedence graph of thirty tasks (see Fig. 3), disposing of six types of equipment. The equipment costs and tasks durations are shown in Table II. The B&B algorithm provided the optimal solution, of cost 323000, in approximately 72 seconds (which is thousands times more than for a small or even a middle scale problem).

This problem has been considered for an analysis of the heuristic's performance when varying the value of K. Another two problems have been also considered: one of twenty tasks and four equipment types and the second one of twenty-five tasks and five equipment types. For the three problems, the values of the F-ratio and of the E-ratio are also given.

The heuristic results for three values of K are listed in Table III.

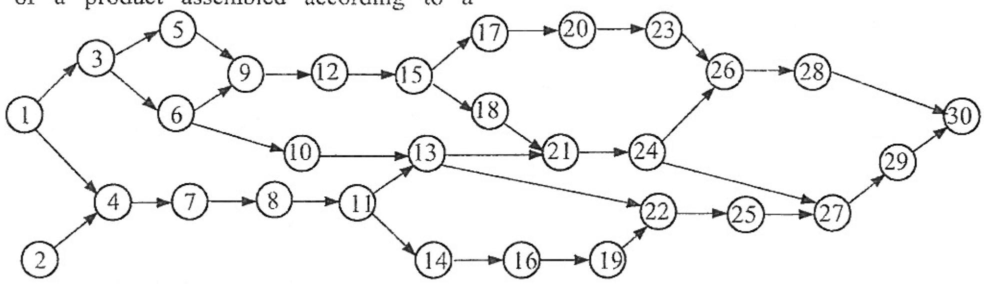

Fig. 3 Precedence graph for a large scale case, taken from [8] (n=30 tasks)

TABLE II

TASK TIMES DEPENDING ON THE EQUIPMENT IN A LARGE SCALE CASE (n=30, m=6)

Equipment	E_1	E_2	E_3	E_4	E_5	E_6
Cost/unit	20000	40000	35000	38000	41000	37000
Task 1	11	10		12	14	13
Task 2	13	15	14		15	14
Task 3	49		38	48		42
Task 4	19		17	16	18	
Task 5	18	19		20	19	
Task 6		18	21	20		19
Task 7	20		27	25	24	23
Task 8	7	8	10		7	9
Task 9	27		25	24		26
Task 10		8	9	5	10	
Task 11	11	15		10	11	12
Task 12	29		25		22	23
Task 13	12	14	12		10	12
Task 14	30	34	35	30	31	
Task 15		11	11	12	14	
Task 16	15	12		10	11	13
Task 17	9	11	13	12	11	10
Task 18	5		6	4	6	7
Task 19	4	4	4		9	5
Task 20	13	12	14	13	13	11
Task 21	12		17	15		14
Task 22		31	34	30	32	
Task 23	15	18		14	15	16
Task 24		20	21	22	20	20
Task 25	20		19	18	18	
Task 26	15	16	14	12		
Task 27	18	20		21	19	18
Task 28		12	15	14		16
Task 29	5		6		4	5
Task 30	10	10		11	12	

The proposed heuristic's efficiency may be judged upon the capacity of providing solutions not far from optimality in a reasonable time. The considered problems have almost the same (low) flexibility of the precedence graph; the first problem has a smaller equipment flexibility than the other two. So, the dimension differences between the three problems are mainly due to n and m.

Analyzing Table III, one can note the dramatic reduction of the execution time for the first and the third problem, whereas differences from the optimal solution are acceptable. These differences are larger as the problem dimension is larger. Indeed, by increasing the value of K one obtains more quickly a solution, but also more different from the optimal one.

VI. CONCLUSION

In this paper, the production line design problem has been formulated such that both balancing and resource cost minimization to be achieved. Traditionally, the task-to-workstations assignment supposes the *serial* execution by the same equipment. But if disposing of multifunctional equipment, the *parallel* execution can be an alternative. The *compatibility* constraints – which are complementarily deduced in relation to the precedence constraints – govern the generation of the blocks of parallel tasks. This is feasible and desirable when disposing of several types of flexible equipment.

Tasks parallelization is expected to produce time savings – especially when the processing times inside of a block do not have a great variance – which makes it superior to the serial variant. To the proposed B&B algorithm for obtaining the blocks of tasks along with the cheapest equipment able to perform it – providing the optimal solution – has been added a heuristic for large-scale problems. Its principal advantage is the reduction of the execution time, while obtaining a quite good solution.

The future research will be directed to the use of meta-heuristics (such as, for example, genetic algorithms or ant colonies optimization) for solving the described problem.

TABLE III

HEURISTIC RESULTS

Problem's features		n=20, m=4, F-ratio=0.62, E-ratio=0.55			n=25, m=5, F-ratio=0.7, E-ratio=0.76			n=30, m=6, F-ratio=0.67, E-ratio=0.71		
Problem solving		**cost**	**CPU time [s]**	**difference to optimal [%]**	**cost**	**CPU time [s]**	**difference to optimal [%]**	**cost**	**CPU time [s]**	**difference to optimal [%]**
B&B algorithm: optimal solution		364000	1.214	0	720000	0.52	0	323000	72.633	0
Heuristic: K	0.1	371000	0.07	1.92	720000	0.215	0	343000	0.555	6.19
	0.9	380000	0.04	4.39	790000	0.163	9.72	360000	0.505	11.45
	1.5	379000	0.045	4.12	740000	0.17	2.77	387000	0.74	19.81

VII. ACKNOWLEDGMENT

This work was partially supported by the Romanian National Council of Scientific Research, under Grant AT/80.

VIII. REFERENCES

[1] A. Dolgui, N. Guschinsky and G. Levin, *Approaches to Balancing of Transfer Lines with Blocks of Parallel Operations*, Research Report, Institute of Engineering Cybernetics/University of Technology of Troyes, Preprints No. 8, Minsk, Academy of Science, 2000.

[2] I. Baybars, "A Survey of Exact Algorithms for the Simple Line Balancing Problem", *Management Science*, Vol. 32, pp. 909-932, 1986.

[3] S.C. Graves and D.E. Whitney, "A Mathematical Programming Procedure for Equipment Selection and System Evaluation in Programmable Assembly", *Proceedings of the IEEE Conference on Decision and Control*, IEEE Press, pp. 531-536, 1979.

[4] S.C. Graves and B.W. Lamar, "An Integer Programming Procedure for Assembly System Design Problems", *Operations Research*, Vol. 31, No. 3, pp. 522-545, 1983.

[5] R.E. Gustavson, *Design of Cost Effective Assembly Systems*, Technical Report No. P-2661, C.S. Draper Laboratory, Cambridge, MA, 1986.

[6] E. Falkenauer, "A Hybrid Grouping Genetic Algorithm for Bin Packing", *Journal of Heuristics*, Vol. 2, No. 1, pp. 5-30, 1996.

[7] J. Bukchin and M. Tzur, "Design of Flexible Assembly Line to Minimize Equipment Cost", *IIE Transactions*, Vol. 32, pp. 585-598, 2000.

[8] J. Bukchin and J. Rubinovitz, "A Weighted Approach for the Assembly Line Design with Station Paralleling and Equipment Selection", *IIE Transactions*, Vol. 35, pp. 73-85, 2003.

[9] A.I. Bratcu, L. Makdessian and A. Dolgui, "Minimisation of Equipment Cost for Transfer Lines with Blocks of Parallel Tasks", *Proceedings of the 5th IEEE International Symposium on Assembly and Task Planning – ISATP '03*, pp. 109-114, 2003.

[10] A. Bratcu, Détermination systématique des graphes de précédence et équilibrage des lignes d'assemblage, Ph.D. Thesis, Franche-Comté University, Besançon, France, 2001.

[11] E.M. Mansoor and M. Yadin, "On the problem of assembly line balancing", *Developments in Operations Research*, Gordon and Breach, New York, p. 361.

[12] J. Rubinovitz and J. Bukchin, "RALB – a heuristic algorithm for design and balancing of robotic assembly lines", *Annals of the CIRP*, Vol. 42, pp. 497-500, 1993.