The ASALB Problem with Processing Alternatives Involving Different Tasks: Definition, Formalization and Resolution*

Liliana Capacho[1,2] and Rafael Pastor[2]

[1] Universidad de Los Andes, Dpto. de I.O. - EISULA y CESIMO, Mérida, Venezuela
[2] Technical University of Catalonia, IOC Research Institute, Av. Diagonal 647, Edif. ETSEIB, p.11, 08028 Barcelona, Spain
{liliana.capacho, rafael.pastor}@upc.edu

Abstract. The Alternative Subgraphs Assembly Line Balancing Problem (ASALBP) considers assembly alternatives that determine task processing times and/or precedence relations among the tasks. Capacho and Pastor [3] formalized this problem and developed a mathematical programming model (MILP) in which the assembly alternatives are determined by combining all available processing alternatives of each existing sub-assembly. In this paper an extended definition of the ASALBP is presented in which assembly sub-processes involving different tasks are also considered. Additionally, a mathematical programming model is proposed to formalize and solve the extended version of the ASALBP, which also improves the performance of the former MILP model. Some computational results are included.

1. Introduction

The classical Assembly Line Balancing Problem (ALBP) consists in assigning a set of tasks to a group of workstations in order to optimize certain efficiency measure (e.g. the number of workstations). Each task is characterized by a processing time and a set of precedence relations, which specifies the allowed processing order.

ALBP are classified into two well-known categories [1]: Simple Assembly Line Balancing Problems (SALBP) and General Assembly Line Balancing Problems (GALBP). SALBP involve very simple and restrictive problems which consider, for example, a unique serial line that processes a single model of one product. GALBP are those in which one or more assumptions of the simple case are varied. Amongst such problems the following groups are usually considered: UALBP that involve U-shaped lines that may be used to overcome the inflexibility of serial lines; MALBP which appear when a line produces units of different models of a unique product (see, for example, Milterburg [8]); MOALBP which consider several optimization objectives; and RALBP that consider robotic lines. Other less common GALBP considered in the literature include problems that involve parallel workstations;

* Supported by the Spanish MCyT project DPI2004-03472 and co-financed by FEDER

parallel tasks; stations not equally equipped, which may imply equipment selection; two-sided or buffered lines; workstation capacity constrained; problems involving processing times that are dependent on the sequence, stochastic or fuzzy; and problems considering multi-product lines (e.g. [5], [9], [12]).

Assembly line balancing problems have been widely studied (e.g. Baybars [1], Becker and Scholl [2], Scholl and Becker [10]). Although most published research work on assembly line balancing focuses on the simple case, in recent years a significant amount of research effort has been directed at the general case in order to address more realistic problems.

Numerous procedures have been developed to solve assembly line balancing problems, which are usually grouped into two main categories: exact methods and approximate methods. According to Becker and Scholl [2], most exact methods are based on linear and dynamic programming and branch and bound procedures (e.g. Scholl and Klein [11]). A wide range of approximate methods have been developed to efficiently solve more realistic ALBP. Among these there are heuristics based on priority rules and enumeration procedures (e.g. Lapierre and Ruiz [7]); metaheuristics such as genetic algorithms (e.g. Kim et al. [6]); simulated annealing (e.g. Suresh and Sahu [12]); tabu search (e.g. Pastor et al. [9]); and others that include procedures based on ant colonies, constrained logic programming, fuzzy logic, expert systems and algorithms based on networks theory (e.g. Gen et al. [5]).

A common feature of ALBP is that a unique and predetermined precedence graph is used to represent all relations among the tasks required to assemble a particular product. In real-life problems, however, is possible that some parts of a product admit several alternative assembly variants that cannot be represented in a standard precedence graph.

In this regard, Capacho and Pastor [3] presented and formalized a new GALBP entitled ASALBP (Alternative Subgraphs Assembly Line Balancing Problem), which considers the possibility of processing alternatives in which the processing time and/or precedence relations of some tasks depend on the selected sub-assembly alternative. In this problem, each processing alternative consists of a particular processing order of a subset of tasks.

Usually, in assembly line balancing processes an alternative for each sub-assembly is selected a priori and the line is then balanced. To solve the ASALBP efficiently, two subproblems need to be solved simultaneously: the decision problem, which selects the assembly sequence of those parts that admit alternatives, and the balancing problem, which assigns the tasks to the workstations.

The remainder of this paper is organized as follows. Section 2 introduces the Alternative Subgraphs Assembly Line Balancing Problem. Section 3 presents the extended definition of the ASALBP. Section 4 describes the model proposed to solve the ASALBP regarding the extended definition. Section 5 provides some computational results. Finally, Section 6 presents the conclusions and proposes future research work.

2. ASALBP: the Alternative Subgraphs Assembly Line Balancing Problem

As previously mentioned, the ASALBP is a new general assembly line balancing problem that considers assembly alternatives, in which an alternative has to be selected for each part that admits assembly variants. Each processing alternative could be represented by a precedence graph as can be seen in Figure 1, which shows two assembly alternatives for a simple example involving six tasks. In alternative 2 the processing order changes for tasks B, C, D and E. Furthermore, the processing time of task B increases 2 time units when it is processed after task E.

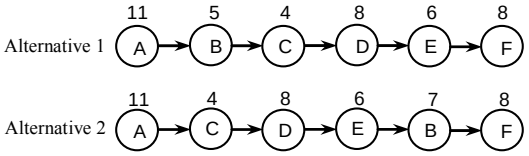

Fig. 1. Two assembly alternatives

Normally, when a problem with assembly alternatives has to be solved one of the available alternatives for each sub-assembly is selected a priori and the line is then balanced. Smallest total processing time is a common criterion that system designers usually use to choose an assembly alternative. If these two processes – the selection process and the line balancing – are carried out independently, it cannot be guaranteed that the global problem can be solved optimally. However, better solutions can be obtained if the two problems are solved simultaneously, as illustrated in Figure 1. If the two resulting balancing problems are solved optimally, minimizing the number of workstations given an upper bound on the cycle time of 15 time units, the results shown in Table 1 are obtained. Even though it has a longer total processing time (44), Alternative 2 provides the best number of workstations for the problem. If the selection process had been carried out a priori, Alternative 1 would have been selected, since its total processing time is 42, and the best solution would have been discarded.

Table 1. Results of balancing the assembly alternatives

	Tasks per station (time)				Total	No.
Alt	I	II	III	IV	processing time	of stations
1	A (11)	B,C (9)	D,E (14)	F (8)	42	4
2	A,C (15)	D,E (14)	B,F (15)	-	44	3

The assembly alternatives of Figure 1 represent two variants of an assembly process, which determine two alternative subgraphs: S1, which consists in performing tasks B, C, D and E (in the shown order); and S2, which consists in performing tasks C, D, E and B (in that order). A diagramming scheme called S-Graph, proposed by Capacho and Pastor [3], is used here to represent all alternative subgraphs in a unique precedence graph, as shown in Figure 2.

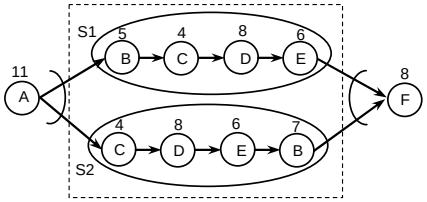

Fig. 2. Precedence S-Graph

To solve the ASALBP, Capacho and Pastor [3] developed an integer linear programming mathematical model (hereafter referred to as the preliminary model, M1), which decides on both the assembly subgraphs and the line balancing. The task-workstation assignment variables in M1 are defined for each total assembly precedence graph (i.e., global route) obtained by combining the alternative subgraphs of each available subassembly contained in the S-Graph. Therefore, each of the global routes considers the whole set of tasks required to assemble a product. Figure 3 shows two global routes (GR1 and GR2) for the example shown in Figure 1. The computational experiment carried out with M1 shows that optimal solutions can be obtained in a reasonable amount of time for small problems.

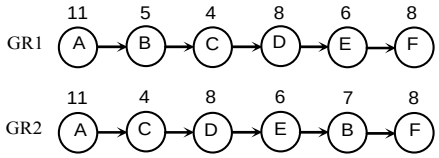

Fig. 3. Alternative global routes

To address more general problems, the definition of the ASALBP is extended in this paper: it is possible to consider alternative assembly processes that involve different and independent sets of tasks. Additionally, a new integer linear programming mathematical model (hereafter referred to as the enhanced model, M2) has been developed to formalize and solve this extended version of the ASALBP.

3. The Extended ASALBP Definition

As mentioned above, the former ASALBP definition considers the possibility of assembly alternatives in such a way that all alternatives of a particular sub-assembly (one of which must be selected) involve the same tasks. In practice, however, industrial problems can involve alternative assembly processes that consider different and mutually exclusive sets of tasks. For instance, consider the toy-manufacturing example given in Das and Nagendra [4], in which a large number of products are assembled from molded plastic parts or from metal stamping. In this example, the tasks are performed only if the assembly process they belong to is selected. Figure 4 shows the S-Graph for an example with three assembly processes involving 3 different and independent sets of tasks: S1 with tasks B1 to B3; S2 with tasks C1 to C4; and S3 with tasks D1 to D3.

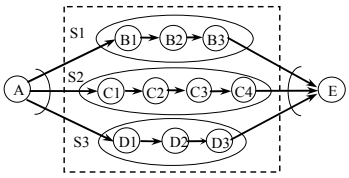

Fig. 4. S-Graph with alternative assembly processes

The new ASALBP definition considers alternative assembly precedence subgraphs that can involve either the same or different sets of tasks. In these two cases, task processing times and/or precedence relations can be dependent on the selected subgraph. As in the former definition, it is assumed that assembly alternatives do not overlap each other; thus, each alternative of each subassembly is represented by a unique and independent precedence subgraph. This new ASALBP definition enables more realistic problems to be addressed and on the other hand, relaxes the SALBP hypothesis which states that all tasks must be processed only once.

4. Enhanced Model for the Alternative Subgraphs Assembly Line Balancing Problem (M2)

Before presenting model M2, some aspects concerning assignment variables and precedence relations are discussed.

4.1. Modeling assumptions

In model M1, each overall assembly alternative referred to as a global route was determined by a precedence graph. As a result, there were as many global assembly routes as combinations of alternative subgraphs for each available subassembly contained in the S-Graph. Consider, for example, the S-graph in Figure 5, which involves 9 assembly tasks. In this example, 9 global routes need to be considered as follows. GR1: A-S1-E-S4-I (i.e. A-B-C-D-E-F1-F2-F3-I), GR2: A-S1-E-S5-I (i.e. A-B-C-D-E-G1-G2-G3-I), GR3: A-S1-E-S6-I, GR4: A-S2-E-S4-I, GR5: A-S2-E-S5-I, GR6: A-S2-E-S6-I, GR7: A-S3-E-S4-I, GR8: A-S3-E-S5-I and GR9: A-S3-E-S6-I.

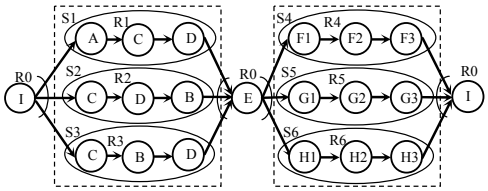

Fig. 5. S-Graph for an example with 9 tasks

Observe that tasks that do not admit processing alternatives are also involved in all global routes. This implies that a large number of task-workstation assignment variables have to be defined for even a small number of routes. In the proposed model, alternative subgraphs (referred to as partial routes) are considered only for those tasks that admit

processing alternatives and a unique route (known as a base route) is considered for those tasks without processing alternatives. Task-workstation assignment variables are thus defined only for the partial routes (alternative subgraphs), thereby considerably reducing the size of the model to be solved. For the example in Figure 5, tasks A, E and I have only one possible processing alternative. Therefore, such tasks are processed according to the base route (referred to as R0). Partial routes R1, R2 and R3 represent the processing alternatives involving tasks B, C and D; R4 is the processing alternative that involves tasks F1 to F3; R5 involves G1 to G3 and R6 involves H1 to H3. For each subset of routes that are alternative to each other (e.g., R4, R5 and R6), one partial route must be selected. Observe that only 7 partial routes are involved; the difference between the number of global and partial routes is even greater because in M1 the number of global routes increases exponentially with the number of partial routes.

For global routes, the immediate predecessors of a task for each route are fixed and the precedence constraints can be easily established. However, this is not the case for partial routes. The difficulty arises because an immediate predecessor or the task itself may have alternatives and only one of these is to be selected. Therefore, all possible immediate predecessors of a task must be considered.

In order to account for all possible precedence relations implied when considering partial routes and to facilitate their formalization, tasks can be divided into two categories: fixed, which are those without alternatives (i.e., those processed throughout the base route); and mobile, which are those that form a part of alternative routes. As can be seen in Figure 6, tasks A, F and G are fixed whereas tasks B, C, D and E are mobile because they are involved in alternative assembly routes: R1 and R2 involving tasks B and C, and R3 and R4 involving tasks D and E. A fictitious task with nil processing time, α, is used in the S-Graph to represent precedence relations that involve mobile tasks with predecessors that are also mobile but affected by different sets of alternative routes. This case is represented in Figure 6 by the mobile tasks D and E, whose predecessors C and B are also mobile tasks affected by different routes.

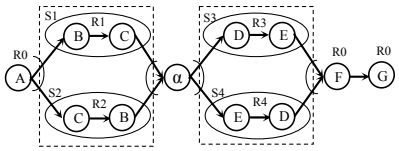

Fig. 6. Fixed and mobile tasks - precedence relations

Table 2 shows the 5 basic cases of task-predecessor relations, formalized below, which are based on the example in Figure 6.

Table 2. Task-predecessor relation typology

	Cases	*i*	*p*
1	Task *i* fixed and its predecessor *p* fixed.	G	F
2	Task *i* fixed and its predecessor *p* mobile.	F	E,D
3	Task *i* mobile and its predecessor *p* fixed.	B	A
4	Task *i* mobile and its predecessor *p* mobile, with *i* and *p* in the same route.	C	B
5	Task *i* mobile and its predecessor *p* mobile, with *i* and *p* in different route.	D	C,B

4.2. Model M2

Parameters:

n	Number of tasks (i = 1,…,n).
nr	Number of routes (r = 0,…,nr).
nsr	Number of different sets of routes (subgraphs) such that the routes within a set are alternatives to each other (q=1,…,nsr). For instance, in the example of Figure 6 there are 2 such subsets (nsr=2), one containing routes R1 and R2, and one containing routes R3 and R4.
m_{min}, m_{max}	Lower and upper bounds on the number of stations.
R_i	Set of all routes through which task i can be processed (i = 1,…,n).
ct	Cycle time.
t_{ir}	Duration of task i when processed through route r (i = 1,…,n; $r \in R_i$).
TR_r	Set of tasks that are affected by route r.
P_{ir}	Set of the possible immediate predecessors of task i, if task i is processed through route r ($i = 1,...,n; r \in R_i$).
PT_i	Set of all possible immediate predecessors of task i ($PT_i = \bigcup_{\forall r \in R_i} P_{ir}$).
E_{ir}, L_{ir}	Earliest and latest station that task i can be assigned to, if task i is processed through route r ($i = 1,...,n; r \in R_i$).
SCR_q	Subset q of routes that are alternative among one another (q=1,…,nsr). For the example in Figure 6, there are two of such subsets: SCR_1 involving R1 and R2 and SCR_2 involving R3 and R4.

Decision variables:

$x_{ijr} \in \{0,1\}$	=1 if task i is assigned to workstation j and processed through route r ($i = 1,...,n; \forall r \in R_i; \ \forall \ j \in [E_{ir}, L_{ir}]$).
$y_j \in \{0,1\}$	=1 if there is any task assigned to workstation j (j=m_{min}+1,…,m_{max}).
$ar_r \in \{0,1\}$	= 1 if there is any task processed through route r (r = 1,…,nr).

Mathematical Model for the ASALBP-1: to minimize the number of workstations given ct

$$\text{Minimize } Z = \sum_{j=m_{min}+1}^{m_{max}} j \cdot y_j \ . \tag{1}$$

$$\sum_{j=E_{i0}}^{L_{i0}} x_{ij0} = 1 \quad \forall i \mid i \in TR_0 \ . \tag{2}$$

$$\sum_{\forall r \in R_i} \sum_{j=E_{ir}}^{L_{ir}} x_{ijr} = \sum_{\forall r \in R_i} ar_r \quad \forall i \mid i \notin TR_0 \ . \tag{3}$$

$$\sum_{r=0}^{nr} \sum_{\forall i \mid (r \in R_i) \wedge (j \in [E_{ir}, L_{ir}])} t_{ir} \cdot x_{ijr} \le ct \quad j = 1,...,m_{min} \ . \tag{4}$$

$$\sum_{r=0}^{nr} \sum_{\forall i \mid (r \in R_i) \wedge (j \in [E_{ir}, L_{ir}])} t_{ir} \cdot x_{ijr} \le ct \cdot y_j \quad j = m_{min}+1,...,m_{max} \ . \tag{5}$$

$$\sum_{j=E_{p0}}^{L_{p0}} j \cdot x_{pj0} \le \sum_{j=E_{i0}}^{L_{i0}} j \cdot x_{ij0} \quad \forall i \in TR_0, \forall p \in PT_i \mid p \in TR_0 \ . \tag{6}$$

$$\sum_{\forall s \in R_p} \sum_{j=E_{ps}}^{L_{ps}} j \cdot x_{pjs} \le \sum_{j=E_{i0}}^{L_{i0}} j \cdot x_{ij0} \quad \forall i \in TR_0, \forall p \in PT_i \mid p \notin TR_0 \ . \tag{7}$$

$$\sum_{j=E_{p0}}^{L_{p0}} j \cdot x_{pj0} \le \sum_{\forall r \in R_i} \sum_{j=E_{ir}}^{L_{ir}} j \cdot x_{ijr} + m_{max} \cdot (1 - \sum_{\forall r \in R_i} ar_r) \quad \forall i \notin TR_0, \forall p \in PT_i \mid p \in TR_0 \ . \tag{8}$$

$$\sum_{j=E_{pr}}^{L_{pr}} j \cdot x_{pjr} \le \sum_{j=E_{ir}}^{L_{ir}} j \cdot x_{ijr} \quad \forall i \notin TR_0, \forall r \in R_i, \forall p \in P_{ir} \mid [p \notin TR_0 \wedge r \in R_p] \ . \tag{9}$$

$$\sum_{\forall s \in R_p} \sum_{j=E_{ps}}^{L_{ps}} j \cdot x_{pjs} \le \sum_{\forall r \in R_i} \sum_{j=E_{ir}}^{L_{ir}} j \cdot x_{ijr} \quad \forall i \notin TR_0, \forall p \in PT_i \mid [p \notin TR_0 \wedge (R_i \bigcap R_p = \varnothing)] \ . \tag{10}$$

$$\sum_{r \in SCR_q} ar_r = 1 \quad q = 1, \ldots, nsr \quad . \tag{11}$$

$$\sum_{\forall i \in TR_r} \sum_{j=E_{ir}}^{L_{ir}} x_{ijr} \le ar_r \cdot |TR_r| \quad r = 1, \ldots, nr \ . \tag{12}$$

The objective function (1) minimizes the number of workstations for a given upper bound on the cycle time. The constraints are: (2) and (3), which ensure that all tasks belonging to a selected route are assigned to one and only one workstation, and otherwise tasks are not assigned; (4) and (5) ensure that the total processing time assigned to workstation *j* does not exceed the cycle time; (6) to (10) are the precedence constraints, which guarantee that no task is assigned to an earlier workstation than an immediate predecessor; (11) are the route uniqueness constraints that ensure that one and only one route for each subassembly is selected from among the possible routes; and (12) guarantees that tasks belonging to a particular precedence subgraph are assigned to the same route.

The mathematical formulation for ASALBP-2, considering the extended definition, can be easily obtained by modifying model M2 developed to solve ASALBP-1, in which the objective function to be considered optimizes the cycle time *ct* instead of the number of workstations, which is a given parameter.

5. Computational Experiment

To solve problems involving alternative processes with different sets of tasks, model M1 was easily adapted by properly defining the global routes. To compare models M1 and M2 and evaluate their performance regarding industrial-sized problems, a computational experiment was carried out. The test-problem instances were designed using some of the benchmark data sets available at the webpage www.assembly-line-balancing.de for assembly line balancing research (i.e., the problems of Bowman, Mansor, Buxey, Gunther, Kilbrid, Hann, Warnecke, Tonge and Arc with 8, 11, 29, 35, 53, 58, 70 and 111 tasks respectively). The problems were adapted by incorporating a number of assembly alternatives (between 2 and 14) and using 3 or 4 different cycle-time values. When alternative assembly processes involving different tasks were considered, new sets of tasks

were also added to the original problems. For example, 9 new tasks were added to Hann's problem to contemplate 4 new assembly processes involving 2, 3, 2 and 2 tasks respectively. A total of 82 test-problem instances were defined and solved with both models using the optimization software ILOG CPLEX 9.0 on a PC Pentium 4, CPU 2.88 GHz with 512 Mb of RAM. Table 3 presents the data and results for some of these problems, including the name of the problem, the number of tasks *n*, the cycle time *ct*, and the number of global routes for model M1 and partial routes for model M2.

Table 3. Results of optimally solving ASALBP instances

Problem	n	ct	No. of routes		Constraints		Variables		Solving Time		% of Improv.
			Global	Partial	M1	M2	M1	M2	M1	M2	
Bowman	8	20	18	9	366	77	1744	888	0.53	0.03	94,3
Mansor	11	62	12	8	288	74	804	547	0.63	0.09	85,7
Mansor	11	62	15	9	352	78	1002	614	2.10	0.16	91,0
Buxey	29	54	12	8	861	147	3850	2581	61547	92.03	99,9
Buxey	29	54	6	6	444	134	1936	1941	18485	0.86	100
Gunther	35	41	32	11	2633	205	25806	8911	89558	14805	83,5
Gunther	40	81	60	13	4287	189	28824	6276	467	0.31	99,9
Kilbrid	45	56	12	8	1383	204	10840	7247	213	1.41	99,3
Kilbrid	45	69	24	10	2505	217	17312	7241	830	1.06	99,9
Hann	53	4676	18	9	2424	238	4780	2403	114	0.13	99,9
Hann	58	2004	24	10	3400	263	19516	8157	8356	3.48	100
Hann	62	2806	36	12	5210	280	22340	7471	19785	249	98,7
Warnecke	58	111	2	3	368	235	3186	4754	7200	638	91,1
Warnecke	58	111	4	5	648	253	6318	7888	17709	1410	92,0
Tonge	70	185	8	7	1428	342	21356	18702	259200	80122	69,1
									Average percentage of improvement of M2 over M1		93,6

Table 3 shows that model M2 outperformed model M1 in all cases. Furthermore, M2 achieved around 94 % of average improvement; reaching a 100% in nearly half of the problems solved. On the other hand, the number of variables and constraints was significantly reduced (as intended) and the solving time was considerably smaller than with the preliminary model M1. Therefore, optimal solutions could be obtained and guaranteed in a reasonable amount of time for problem instances involving up to 70 tasks and 40 assembly alternatives (i.e., 14 partial routes).

For the problems for which no optimal solution was guaranteed, models M1 and M2 were solved with the computing time restricted to 1800 seconds (a realistic time window in an industrial environment). Table 4 shows the results obtained with model M2; it presents no data for M1 because the established time limit was exceeded without any solution being provided for these problems. Observe that model M2 obtained solutions with one workstation deviation from the benchmark optimum. Furthermore, the known optimal solution was found for Gunther's and Tonge's problems (marked in Table 4 with an asterisk).

Table 4. Problems solved within a 1800-seconds time window

Problem name	No. of tasks	Cycle time	No. of routes		No. of workstations	
			Global	Partial	Obtained	Optimal
Gunther*	35	41	32	11	14	14
Warnecke	58	111	2	3	15	14
Tonge	70	320	8	7	12	11
Tonge*	70	220	8	7	17	17
Arc2	111	17067	4	5	10	9

6. Conclusions and Further Research

In this paper an extended definition of the ASALBP has been presented. More realistic problems can be addressed with this new definition since it allows alternative assembly processes to be considered that can involve either the same or different and mutually exclusive sets of tasks. Therefore, the hypothesis that states that tasks must be processed only once is relaxed because some tasks are not processed if the alternative they belong to is not selected.

Additionally, a mathematical programming model has been presented, which solves the ASALBP considering the extended definition. With this new model, the problems are solved in a significantly shorter time than with the preliminary model adapted to the new definition; moreover, medium-sized problems are solved in a very reasonable amount of time. For bigger problems, good feasible solutions were obtained in 1800 seconds, a realistic time window in an industrial environment. Nevertheless, since the solving time increases exponentially with the number of tasks and assembly alternatives, further research is needed in order to develop heuristics to solve the ASALBP regarding the extended definition presented and formalized in this paper.

References

1. Baybars, I. A survey of exact algorithms for the simple assembly line balancing problem. *Management Science*, 32, 909-932 (1986).
2. Becker, C. and Scholl, A. A survey on problems and methods in generalized assembly line balancing. *Eur. J. of Op. Res.*, 168, 694-715 (2006).
3. Capacho, L. and Pastor, R. ASALBP: the Alternative Subgraphs Assembly Line Balancing Problem. *Technical Report* IOC-DT-P-2005-5. UPC. Spain. Jan. (2005).
4. Das, S. and Nagendra, P. Selection of routes in a flexible manufacturing facility. *Int. Journal of Production Economics*, 48, 237-247 (1997).
5. Gen, M., Tsujimura, Y. and Li, Y. Fuzzy assembly line balancing using genetic algorithms. *Comp. & Ind. Eng.*, 31, 631-634 (1996).
6. Kim, Y.K., Kim, Y.J. and Kim, Y. Genetic algorithms for assembly line balancing with various objectives. *Comp. and Ind. Eng.*, 30, No.3, pp. 397-409 (1996).
7. Lapierre, S.D. and Ruiz, A.B. Balancing assembly lines: an industrial case study. *J. of the Operational Research Society*, 55, 559-597 (2004).
8. Miltenburg, J. Balancing and scheduling mixed-model U-shaped production lines, *International Journal of Flexible Manufacturing Systems*, 14, 119-151. (2002).
9. Pastor, R., Andres, C., Duran, A. and Perez, M. Tabu search algorithms for an industrial multi-product, multi-objective assembly line balancing problem, with reduction of task dispersion, *J. Op. Res. Society*, 53, 1317-1323 (2002).
10. Scholl, A. and Becker, C. State-of-the-art exact and heuristic solution procedures for simple assembly line balancing. *European J. of Op. Res.*, 168, 666-693 (2006).
11. Scholl, A. and Klein, R. SALOME: A bidirectional branch and bound procedure for assembly line balancing. *INFORMS Journal on Comp.*, 9, 319-334 (1997).
12. Suresh, G. and Sahu, S. Stochastic assembly line balancing using simulated annealing, *Int. Journal of Production Research* 32, 1801-1810 (1994).