Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 44 (2016) 102 – 107

www.elsevier.com/locate/procedia

6th CIRP Conference on Assembly Technologies and Systems (CATS)

Worker skills and equipment optimization in assembly line balancing by a genetic approach

Michela Dalle Mura[a,]*, Gino Dini[a]

[a]*Department of Civil and Industrial Engineering, University of Pisa, Via Diotisalvi, 2 , Pisa 56122, Italy*

* Corresponding author. Tel.: +39-050-2218011. *E-mail address:* m.dallemura@ing.unipi.it

Abstract

The Assembly Line Balancing Problem (ALBP) is to determine the optimal allocation of assembly operations to a set of workstations, with respect to precedence constraints. This paper proposes a multi-objective optimization to solve the ALBP using a Genetic Algorithm (GA) approach. The aim is to minimize, besides the number of workstations, two aspects, very important from an economic point of view, but poorly treated in literature: the number of high skilled workers needed to correctly accomplish the operations and the number of assembly equipment along the line. A case study was finally discussed in order to demonstrate the capability of the proposed method in finding optimized solutions in different scenarios.

Peer-review under responsibility of the organizing committee of the 6th CIRP Conference on Assembly Technologies and Systems (CATS)
Keywords: Assembly line balancing; Genetic algorithm; Worker skill; Assembly equipment

1. Introduction

The combinatorial problem to determine the optimal allocation of assembly operations among workstations, with respect to precedence constraints, is known as the Assembly Line Balancing Problem (ALBP). Because of its complexity and great significance from an economic point of view, ALBP is considered to be one of the main issues in design and planning of manufacturing systems. The classical single-model version is the Simple Assembly Line Balancing Problem (SALBP), whose main assumptions are [1]:

- mass production of one homogeneous product;
- paced line with fixed cycle time;
- deterministic execution times;
- serial line layout, one-sided stations;
- fixed launch interval corresponding to cycle time.

Depending on the objective to be pursued, several versions of the problem can be defined [2,3]:

- feasibility problem (SALBP-F): it consists in finding a task assignment for a given number of workstations operating with a certain cycle time;
- first optimization problem (SALBP-1): the objective is to minimize the number of workstations for a given cycle time;
- second optimization problem (SALBP-2): the objective is to minimize the cycle time for a given number of workstations;
- efficiency optimization (SALBP-E): the objective is to maximize the balance efficiency, defined as the ratio between the total operation time and the total available time.

Various techniques are currently employed in the literature to solve these balancing problems [4]. Firstly, heuristic methods can be used to find feasible solutions in a short time, giving good results for simple problems. In addition, iterative algorithms can reach a solution close to the optimal in a reasonable computing time, thanks to a finite number of steps that lead to a gradual convergence moving through successive approximations. Algorithms inspired by the biological world

Peer-review under responsibility of the organizing committee of the 6th CIRP Conference on Assembly Technologies and Systems (CATS)
doi:10.1016/j.procir.2016.02.033

can also be used to solve these problems, such as Neural Networks and Ant Colony Optimization. Among them, the Genetic Algorithm (GA) has shown promising results in this field and has been chosen by many researchers because of its success in solving a wide variety of complex balancing problems [5,6].

2. State of the art and objectives

ALBP has been widely treated in the literature. Fig.1 shows the distribution of balancing objectives pursued in 25 recent papers.

As shown, most researches focuses on the minimization of the number of workstations [7], trying simultaneously to maximize the workload smoothness, to obtain better balanced solutions [8,9,10]. The maximization of the line efficiency is pursued in [11], whereas in other works the minimization of the cycle time and of the frequency of tool changes are treated, with the aim of maximizing the workload smoothness [12] or minimizing the frequency of direction changes [13] at the same time. Unlike the above-mentioned objectives, the minimization of aspects such as equipment costs and worker skills have been poorly discussed in the literature, despite the importance from an economic point of view. Contributions approaching these aspects can be found in [14], where a method to choose the type of equipment to place in each workstation in order to minimize the total equipment cost is proposed, and in [15], where the problem to minimize the number of temporary workers is taken into account.

The novel of this work is therefore to propose a software tool to solve SALBP-1 through a multi-objective genetic algorithm that optimizes, besides of the number of workstations, the two following important aspects:

- distribution of the worker skills along the line. The objective is to distribute the assembly tasks in order to group, in the minimum number of workstations, those ones requesting high skills;
- distribution of the equipment along the line. The objective is to distribute the assembly tasks in order to group, in the minimum number of workstations, those ones requesting a specific assembly equipment.

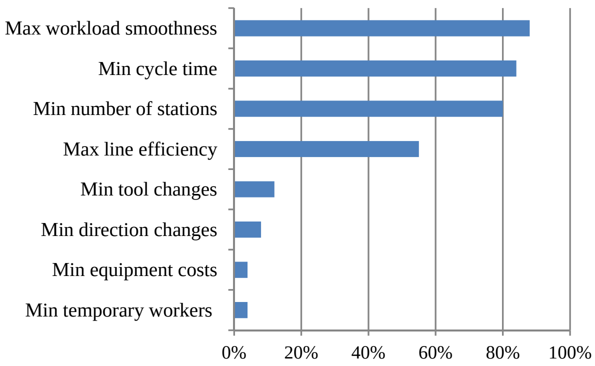

Fig. 1: Frequency of ALBP objectives in the examined literature

In balancing algorithms where this first aspect is not considered, solutions may lead to assembly plans that minimize the number of workstations, even though operations are distributed so that the use of a high number of skilled workers is needed. These resources may not be available, or alternatively expensive training or movement of personnel can be required to satisfy the assembly plan.

An algorithm that allows to group as much as possible operations requiring high skills in the minimum number of workstations has therefore to be developed, in order to reduce the number of high skilled workers to be placed on the line.

The same concerning equipment availability: solutions grouping in few workstations the assembly tasks with the same machines and tools should be preferred to avoid the duplication of resources along the line and to obtain a reduction of costs and complexity.

As a result, the optimization of these new parameters leads to obtaining efficient assembly line configurations.

3. Description of the proposed system

The developed genetic approach, named GenIAL (Genetic Iteration for Assembly Lines), has been created on a MatLab® platform. The operating mode of the software system, schematically reported in Fig.2, can be described by the following steps:

- Random generation of the initial population through a sequence planner, which operates as follows:

    1. create a subset of assembly operations without any precedence with respect to other tasks;
    2. if the subset is empty, stop, otherwise go to step 3;
    3. randomly choose and remove one of the operations from the subset and assign it to the first available position in the sequence;
    4. put in the subset operations that follow the removed one in the precedence graph, only if not constrained by operations not yet allocated;
    5. return to step 2.

- Fitness evaluation by using a multi-objective function described in detail in section 3.3.
- Genetic operations: the individuals of the current population are selected to produce offspring according to their fitness value. Order-based crossover is applied with a probability of 98%, ensuring populations made only of feasible individuals, in contrast to the swap mutation, applied with a probability of 2%. Elitism is then applied and the roulette wheel selection is used.
- Infeasible sequences correction: the action of the mutation operator may generate individuals that violate the precedence relationships, meaning infeasible assembly sequences. For this reason a repairing procedure, operating similarly to the sequence planner, has been used at this stage.

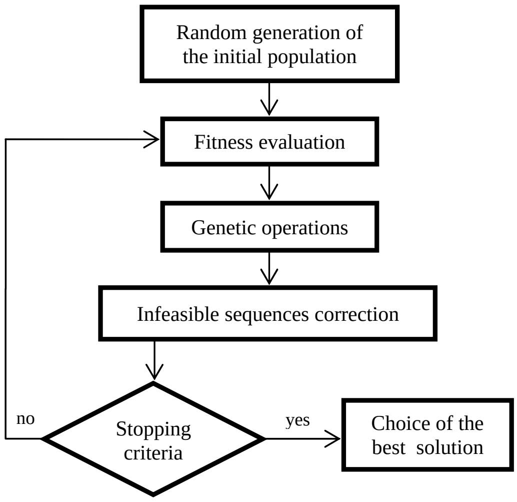

Fig.2: Flowchart of GenIAL algorithm

- Choice of the best solution: when the predetermined stopping criteria has been reached (i.e. maximum number of iterations), the system returns the best solution of the problem. The individual with the highest value of the fitness represents the solution that best balances the assembly line in terms of the previously mentioned objectives.

3.1. Input data

The initial data for GenIAL are the line efficiency, the production rate and the total number of tasks of the assembly problem to be solved. The other data needed to describe each assembly operation are:

- Precedence relationships: these constraints are codified in the software by a square matrix, where each element a_{ij} can be:

$$a_{ij} = \begin{cases} 1 & \text{if task i is a precedence for task j} \\ 0 & \text{if no precedence occurs between i and j.} \end{cases}$$

- Execution time, expressed in minutes;
- Worker skills: considering the different abilities of assembly workers on a production line, in the proposed system a skill level s_i is associated to each task, as detailed below:

$$s_i = \begin{cases} 0 & \text{if task i does not request specific skill;} \\ 1 & \text{if task i requests an intermediate skill;} \\ 2 & \text{if task i requests a high skill.} \end{cases}$$

- Assembly equipment, codified with an equipment matrix whose elements e_{ik} can be:

$$e_{ik} = \begin{cases} 1 & \text{if task i requests equipment k} \\ 0 & \text{if task i does not request equipment k.} \end{cases}$$

Further input data concern the algorithm parameters, namely the number of generations, the size of the population, the crossover probability and the mutation probability.

3.2. Chromosome structure

The encoding method implemented in this paper is the task-oriented representation. In this genetic representation each gene of a chromosome represents one assembly task, expressed by an integer number. The length of the chromosome string corresponds to the total number of tasks to be performed for a given assembly process, with an order corresponding to the processing sequence.

For each chromosome the parameters to be evaluated are: i) the number *N* of workstations needed to perform the assembly sequence, ii) the worker skill index *S*, iii) the equipment index *E*. N is established by allocating the tasks into the workstations such that the sum of related task times in each station does not exceed the cycle time. The index *S* is calculated for each chromosome as the sum of the maximum value of skills requested on each workstation:

$$S = \sum_{m=1}^{N} S_m \tag{1}$$

with

$$S_m = \max\{s_i\}, \quad i = \text{subset of } 1,..., n_t \tag{2}$$

where n_t is the number of tasks and the *subset of 1,...,* n_t consists of tasks assigned to workstation *m*.

Lastly, the index *E* is evaluated by the sum of different equipment among workstations (Eq.(3), Eq.(4)), considering in each workstation a unique occurrence for each equipment:

$$E = \sum_{m=1}^{N} E_m \tag{3}$$

with

$$E_m = \sum_{k=1}^{n_k} \left(\bigcup_{i=1}^{n_t} e_{ik} \right) \tag{4}$$

where n_k is the number of equipment and e_{ik} is the element of the equipment matrix.

In Table 1 an example of chromosome for a 8-task problem is given, where S and E are calculated as follows:

$$S = \sum_{m=1}^{4} S_m = 2+1+1+2 = 6$$

$$E = \sum_{m=1}^{4} E_m = 1+2+1+1 = 5$$

Table 1: Example of chromosome. The genes represent the tasks as they are processed, subdivided in workstations. Worker skill and equipment parameters are associated to each gene.

Chromosome	**1**	**2**	**5**	**6**	**3**	**4**	**7**	**8**
Workstation		1		2		3		4
Worker skill	0	0	2	0	1	1	1	2
Equipment	-	-	1	2	2,3	1	1	4

3.3. Fitness function

The system objective is to determine the assembly sequence with tasks allocation into workstations in order to minimize the previously explained indexes N, S and E.

To accomplish that, the following fitness function for each chromosome has been defined:

$$F = w_1F_1 + w_2F_2 + w_3F_3 \quad (5)$$

where w_i represent the weights of objectives and F_1, F_2, F_3 represent respectively the normalized values derived from the indexes N, S and E.

The greater the value of the fitness function is, the greater the probability for the chromosome to survive to the next generation will be.

The generic F_i is therefore calculated through the following expression:

$$F_i = 1 + \frac{(LB - X)}{(UB - LB)} \quad (6)$$

where X is the current value of the considered index that need to be normalized, LB and UB are the lower and upper bound established for each of the three indexes as detailed below:

- The lower bound of the number of workstations is established by the following expression:

$$LB(N) = \lceil T_p / T_c \rceil \quad (7)$$

where T_p is the total assembly time of the product and T_c is the cycle time.

- The upper bound of the number of workstations is difficult to establish due to the potential risk of overestimating it. For example, to set it equal to the number of tasks is an excessive evaluation. For this reason, in the present work, the upper bound of the number of workstations has been established empirically by analyzing a great quantity of case studies and benchmark from the literature. It can be noticed that UB(N) strongly depends on LB(N), and its value can be reasonably established as a percentage increase η by the following logical conditions:

$$UB(N) = \lceil LB(N) * (1 + \eta) \rceil \quad (8)$$

Considering that:

If LB(N) <10 then η = 30%
Else if LB(N) >=10 and LB(N) <30 then η = 20%
Else η = 10%

- The lower bound of the worker skills is the maximum value of skill in the entire assembly sequence:

$$LB(S) = \max \{s_i\}, \qquad i=1,\dots,n_t \quad (9)$$

- The upper bound of the worker skills is obtained by simulating the tasks assignment to stations, whose number is taken equal to the UB(N), in order to distribute the high and intermediate skills in every station:

$$UB(S) = \sum_{m=1}^{UB(N)} S_m \quad (10)$$

- The lower bound of the equipment is the total number of different equipment needed to perform the entire assembly sequence:

$$LB(E) = n_k \quad (11)$$

- The upper bound of the equipment is obtained by simulating the tasks assignment to stations in order not to repeat the same equipment into workstations, whose number is taken equal to the UB(N) (Eq.(12)). This distribution allows to configure the worst situation:

$$UB(E) = \sum_{m=1}^{UB(N)} E_m \quad (12)$$

4. Case study

The system has been tested using several examples of products. In this section a case study is discussed to investigate the effectiveness of the proposed approach. The example, selected from the industrial reality, is a carburetor of a four-stroke engine, whose exploded view is shown in Fig. 3.

This case study has been analyzed using different scenarios, obtained by varying skill and equipment parameters one at a time. The assembly sequence is formed by 25 tasks shown in Table 2, together with the input data concerning scenario 1. The problem has a production rate of 25 products/hour and the line efficiency is supposed to be 95%; the cycle time is therefore 2.28 min/product.

The other scenarios have been configured as follows:

- Scenario 2 (lower worker skills): considering column 2 of Table 3 as the skills of the tasks;
- Scenario 3 (higher worker skills): considering column 3 of Table 3 as the skills of the tasks;

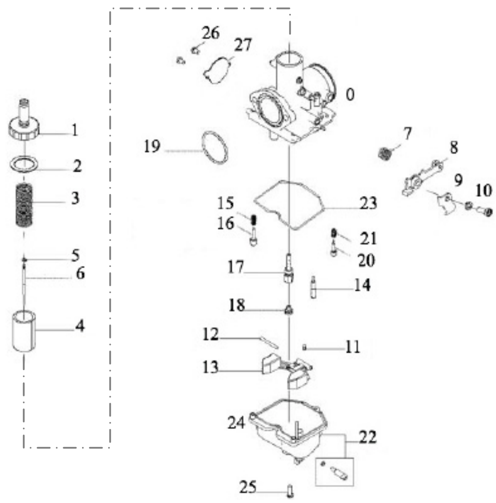

Fig. 3: Case study: exploded view of a carburetor.

- Scenario 4 (lower number of equipment) considering column 4 of Table 3 as the equipment of the tasks;
- Scenario 5 (higher number of equipment): considering column 5 of Table 3 as the equipment of the tasks.

Table 2: Input data for the scenario 1

Task	Precedence constraints	Execution time [min]	Skill	Equipment		
1	-	0.52	0	1	0	0
2	1	0.35	0	1	0	0
3	2	0.52	0	0	0	0
4	3	0.35	0	0	0	0
5	-	0.35	1	0	0	0
6	4,5	0.73	1	0	0	0
7	-	0.52	0	1	0	0
8	7	0.52	0	0	0	0
9	8	0.52	1	0	0	0
10	9	0.73	1	2	0	0
11	-	0.35	1	0	0	0
12	-	0.78	2	2	0	0
13	-	0.73	0	4	0	0
14	13	0.73	0	4	0	0
15	-	0.35	0	0	0	0
16	15	0.73	0	4	0	0
17	-	0.73	1	0	0	0
18	-	0.35	0	0	0	0
19	18	0.78	0	0	0	0
20	-	0.35	0	0	0	0
21	14,16,17, 19,20	0.93	2	2	0	0
22	-	0.73	1	3	0	0
23	-	0.52	1	0	0	0
24	23	0.73	1	2	0	0
25	6,10,11,12, 21,22,24	1.50	2	5	0	0

Table 3: Skill and equipment parameters for scenarios from 2 to 5

Scenario	2	3	4			5		
Task	Skill	Skill	Equipment			Equipment		
1	0	0	1	0	0	1	0	0
2	0	0	1	0	0	2	0	0
3	0	0	0	0	0	1	2	0
4	0	2	0	0	0	0	0	0
5	1	1	0	0	0	3	0	0
6	0	2	0	0	0	0	0	0
7	0	0	1	0	0	1	2	0
8	0	0	0	0	0	0	0	0
9	0	1	0	0	0	0	0	0
10	1	2	2	0	0	2	0	0
11	1	1	0	0	0	3	0	0
12	2	2	0	0	0	2	0	0
13	0	1	1	0	0	1	3	4
14	0	1	1	0	0	4	0	0
15	0	1	0	0	0	3	0	0
16	0	0	1	0	0	1	4	0
17	0	1	0	0	0	3	0	0
18	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0
20	0	0	0	0	0	3	0	0
21	2	2	0	0	0	2	0	0
22	1	1	2	0	0	3	0	0
23	0	1	0	0	0	0	0	0
24	1	2	2	0	0	3	0	0
25	1	2	2	0	0	2	5	0

4.1. Results

The experiments were performed with a population size of 80 and a number of generations of 100. The final results computed by the system for the five scenarios are reported in Table 4. The best solutions in terms of tasks allocation for the scenarios 1, 3 and 5 are respectively reported in Tables 5, 6 and 7.

Table 4: Results referred to the best solution of each scenario

Scenario	N	S	E
1	8	8	6
2	8	7	7
3	8	10	8
4	8	8	5
5	8	8	13

Table 5: Results for the best solution of scenario 1

Station	Tasks	Skill	Equipment
1	15-11-20-17-5	1	-
2	1-7-2-3	-	1
3	8-23-9-18	1	-
4	13-14-16	-	4
5	4-19-6	1	-
6	24-10-22	1	2,3
7	12-21	2	2
8	25	2	5

Table 6: Results for the best solution of scenario 3

Station	Tasks	Skill	Equipment
1	23-20-5-11	1	-
2	17-15-22-18	1	3
3	1-19-7	-	1
4	13-14-2	1	1,4
5	12-24-8	2	2
6	9-3-16	1	4
7	10-21-4	2	2
8	6-25	2	5

Table 7: Results for the best solution of scenario 5

Station	Tasks	Skill	Equipment
1	23-22-20	1	3
2	12-15-17	2	2,3
3	13-16-1	-	1,3,4
4	2-18-3-7	-	1,2
5	19-8-14	-	4
6	5-11-24-9	1	3
7	21-10-4	2	2
8	6-25	2	2,5

4.2. Discussion

The results computed by the system have been compared with the solution obtained through the heuristic Largest Candidate Rule (LCR), shown in Table 8. This heuristic only minimizes the number of workstations and has been chosen as a reference in order to highlight how the proposed GA is able to optimize the distribution of worker skills and assembly equipment along the line.

Table 8: Results for scenario 1 obtained by LCR heuristic

Station	Tasks	Skill	Equipment
1	12-13-1	2	1,2,4
2	14-17-22	1	3,4
3	7-8-9-23	1	1
4	10-24-2-5	1	1,2
5	3-4-6-11	1	-
6	15-16-18-19	0	4
7	20-21	2	2
8	25	2	5

Obtained index values: N = 8; S = 10; E = 11

The value 8 for the number of workstations is obtained for every configuration and, compared with the lower bound (LB(N) = 7), it highlights the capability of the proposed method to minimize the index N.

The computed results show how the system groups worker skills into workstations and reduces the total number of equipment on the line. Comparing the solution of the first scenario obtained with the proposed GA to the one obtained with LCR, the difference between indexes S and E can be detected. Using the proposed genetic approach, both the total worker skills and the total number of equipment requested on the line are lower than those ones obtained with the LCR solution. In fact, in the GA solution, the grouping of operations characterized by the same skill level, such as tasks 12 and 21, or requiring the same equipment, such as tasks 13, 14 and 16, can also be noticed. In the heuristic solution these operations are separated, so the same assembly cycle requires the duplication of resources. As far as the other scenarios are concerned, the obtained solutions show how the system groups tasks requiring high skills such as 10, 21 and 4 in one station for scenario 3, whereas the same operations in scenario 1 are divided in three different stations because they present three different skill levels. Similar considerations apply to solutions referred to other scenarios; in particular, in the line configuration for the last scenario, the grouping of operations requiring the same equipment, such as tasks 13, 16 and 1, can be noticed.

5. Conclusion

The paper proposes a software tool able to solve the SALBP-1 through a multi-objective genetic algorithm.

The results have demonstrated the capability of the developed genetic approach to group in few workstations the necessary resources to accomplish the assembly operations, simultaneously reducing the number of workstations, the distribution of high skilled workers and the number of assembly equipment among workstations. The obtained solutions are good results in terms of the proposed objectives, because they avoid the duplication of resources and allow the reduction of the assembly line complexity, creating efficient configurations in terms of production costs. Such an algorithm might be much in demand in industrial sectors characterized by assembly operations requiring dexterity and manual skills, acquired with years of experience (e.g.: assembly of luxury leather goods such as bags, wallets, etc.). In such companies, these professionals are usually in a limited number, and, thanks to this algorithm, assembly plans able to satisfy this restriction can be obtained, ensuring, at the same time, high level quality products.

In future works, further aspects which contribute to the reduction of costs and cycle time could be also considered in the fitness function evaluation, such as the reduction of assembly direction changes and the minimization of the workload variance.

References

[1] Scholl A. Balancing and Sequencing of Assembly Lines, 2nd ed., Physica, Heidelberg; 1999.
[2] Boysen N, Fliedner M, School A. A classification of assembly line balancing problems. European Journal of Operational Research 183; 2007. p. 674-693.
[3] Becker C, Scholl A. A survey on problems and methods in generalized assembly line balancing. European Journal of Operational Research 168; 2006. p. 694-715.
[4] Rekiek B, Dolgui A, Delchambre A, Bractu A. State of art of optimization methods for assembly line design. Annual Reviews in Control 26; 2002. p. 163–174.
[5] Rashid MFF, Hutabarat W. A review on assembly sequence planning and assembly line balancing optimisation using soft computing approaches. International Journal of Advanced Manufacturing Technology 59; 2012. p. 335–349.
[6] Tasan SO, Tunali S. A review of the current applications of genetic algorithms in assembly line balancing. Journal of Intelligent Manufacturing 19; 2008. p. 49–69.
[7] Pitakaso R. Differential evolution algorithm for simple assembly line balancing tyoe 1 (SALBP-1). Journal of Industrial and Production Engineering 32:2; 2015. p. 104-114.
[8] Yu J, YinY. Assembly line balancing based on an adaptive genetic algorithm. International Journal of Advanced Manufacturing Technology 48 :1; 2010. p. 347–354.
[9] Sabuncuoglu I, Ere, E, Tanyer M. Assembly line balancing using genetic algorithms. Journal of Intelligent Manufacturing 11; 2000. p. 295–310.
[10] Dou J, Jun L, Chun S. A novel feasible task sequence-oriented discrete particle swarm algorithm for simple assembly line balancing problem of type 1. The International Journal of Advanced Manufacturing Technology 69.9-12; 2013. p. 2445-2457.
[11] Al-Hawari T, Ali M, Al-Araidah O, Mumani A. Development of a genetic algorithm for multi-objective assembly line balancing using multiple assignment approach. The International Journal of Advanced Manufacturing Technology, 77:5-8; 2015. p.1419-1432.
[12] Chen R S, Lu, K Y, & Yu S. C. A hybrid genetic algorithm approach on multi-objective of assembly planning problem. Engineering Applications of Artificial Intelligence, 15,; 2002; p. 447–457.
[13] Tseng HE, Chen MH, Chang CC, Wang WP. Hybrid evolutionary multi-objective algorithms for integrating assembly sequence planning and assembly line balancing. Int J Prod Res 46:21; 2008. p.5951–5977.
[14] Bukchin J, Tzur M. Design of flexible asembly line to minimize equipment cost. IIE Transactions 32; 2000. p. 585– 598.
[15] Corominas A, Pastor R, Plans J. Balancing assembly line with skilled and unskilled workers. Omega 36:6; 2008. p. 1126–1132.