Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnLine 50-1 (2017) 5706–5711

Stochastic assembly line balancing with learning effects

F. Lolli, E. Balugani, R. Gamberini, B. Rimini

* *Department of Sciences and Methods for Engineering, University of Modena and Reggio Emilia, Via Amendola 2, Padiglione Morselli, 42122 Reggio Emilia, Italy*

Abstract: Human learning is nowadays taken into account in several research fields, including the assembly line balancing problem. Despite the plethora of contributions and different approaches to solving the problem, the autonomous learning phenomenon, that is to say, the time-dependent or position-dependent reduction of assembly task times due to repetition, should also be explored using stochastic models which, to the best of our knowledge, have been disregarded. In this paper, a well-established cost-based stochastic balancing heuristic has been coupled with a time-dependent learning curve in order to investigate the role of learning in the rebalancing of assembly lines with repetitive tasks. Finally, a real case study has been conducted with the aim of demonstrating the applicability of our proposal.

Keywords: Stochastic assembly line balancing problem; Learning effect; Kottas-Lau heuristic

1. INTRODUCTION

The assembly line balancing problem (ALBP) represents one of the most debated problems in operations management, and aims to assign a set of assembly tasks to workstations in order to optimise a given performance measure while taking the precedence relations between the tasks into account. In particular, most approaches refer to type I and II ALBP, which aim to minimise the number of workstations for satisfying a given cycle time and vice versa respectively (Baybars, 1986). The pioneering work on ALBP dates back to 1955 (Salveson, 1955), and since then, a plethora of academic contributions have been made. Following the taxonomy proposed by Boysen et al. (2007), the contributions on ALBP may be classified into three families: single-model ALPB, where only one product is manufactured; mixed-model ALPB, where one product is manufactured in multiple models on the same assembly line; and multi-model ALPB, where multiple products are manufactured in batches. A further classification factor is the nature of the task times, which may be either deterministic or stochastic. The reader can also refer to Battaïa and Dolgui (2013) for a review of this topic. In fact, a labour-intensive assembly line intrinsically exhibits a higher variability in task times because of the human nature of the work, and thus probabilistic modelling appears more appropriate in this case. The reader can refer to Bentaha et al. (2016) for a review of stochastic assembly/disassembly line balancing approaches. It is worth noting that a special case of ALBP, i.e. the simplest one without precedence constraints, is also an NP-hard problem in the strong sense (Erel and Sarin, 1998) because it may be modelled as a bin-packing problem, which is NP-hard. This remark justifies the large number of contributions in recent literature focused on heuristic and meta-heuristic strategies for solving practical size problems, which may be modelled both as single (e.g. Li et al., 2017) and as multi-objective problems (e.g. Yuguang et al., 2016; Chica et al., 2016).

Learning is a crucial phenomenon in human-related activities, and deserves to be taken into account in several fields. Accounting (e.g. Lolli et al., 2016b), quality improvement (e.g. Lolli et al., 2016a; Lolli et al., 2017), inventory control (e.g. Jaber and Glock, 2013) and scheduling (e.g. Dolgui et al., 2012; Pan et al., 2014; Chutima and Naruemitwong, 2014) are some examples of fields where the learning concept can be applied. With regard to assembly lines, one of the first attempts to deal with learning effects was made by Cohen and Dar-El (1998), where the optimum number of workstations was analytically derived via makespan formulation with deterministic task times. Cohen et al. (2006) investigated the inverse problem, i.e. allocating the task to a given number of stations to minimise the makespan, by adopting Wright's (1936) standard time-dependent (or product-dependent) learning curve with homogenous learning slopes between workstations and deterministic (but dynamic) task times. A type I ALBP with learning effects has been solved by Toksari et al. (2008), who adopted the position-dependent learning curve proposed by Biskup (1999) to demonstrate that simple and U-type line balancing problems with homogenous learning effects are polynomially solvable. Toksari et al. (2010) dealt once again with a type I ALBP by means of a mixed nonlinear integer programming model, but in this case by combining the position-dependent learning curve proposed by Biskup (1999) with a linear deterioration of jobs, which leads to an increase in the task time of a job due to the delay in its starting time. Again, the task times in these contributions are deterministic and dynamic. Hamta et al. (2013) introduced a meta-heuristic solution approach for a multi-objective ALBP with learning, which was modelled using Biskup's position-dependent curve. Again, the task times are dynamic, but in this case they are forced to vary between lower and upper bounds. Hamta et al. (2011) called this type of processing time "flexible operation time".

Surprisingly, the stochastic ALBP with learning effects has not been investigated yet. However, the stochastic components of task times, as well as the human-related learning effects, are relevant in the case of labour-intensive assembly lines. Moreover, non-negligible incompletion costs occur in stochastic assembly lines. This indicates the opportunity of adopting a cost-oriented heuristic approach involving total labour cost and expected incompletion costs. In particular, the Kottas-Lau heuristic (1973) is adopted here, where the task times are modelled by means of a time-dependent learning curve with a plateau based on Wright's well-known curve (1936). In short, the Kottas-Lau heuristic, which addresses a cost-based stochastic type I ALBP, has been coupled with learning effects resulting from repetitive tasks, with the aim of investigating the possibility of rebalancing the assembly line. In fact, the rebalancing problem represents a relevant issue in real situations when, for instance, changes in market conditions, product design and quantities occur (see Gamberini et al., 2006; Gamberini et al., 2009). This problem is addressed here solely as a consequence of the learning process involving the assembly workstations.

The rest of the paper is organised as follows. After the problem assumptions are set out (Section 2), the solution approach is explained in detail (Section 3), and then applied to a real case study (Section 4). Section 5 closes the paper by presenting the conclusions and the further research agenda.

2. PROBLEM ASSUMPTIONS

The standard set of assumptions for the cost-based stochastic type I ALBP is imposed on the assembly tasks:

- The assembly line is paced. Each operator stands at a station and has a fixed takt time T to perform the assigned assembly tasks on a single product.
- A precedence matrix is defined between the tasks. These precedences are the only constraints related to task execution.
- The tasks that are still incomplete after T are executed outside of the line at a higher cost, along with the tasks that were blocked due to the precedence constraints. Each task i is thus assigned an out-of-line assembly cost I_i'. All the tasks which are not unfinished are still executed on the line.
- All the operators are defined by the same characteristics and are paid the same hourly amount c.
- Each task time is normal-distributed, with an expected value and variance that can differ from task to task. These probability distributions are independent between tasks.

An extra set of assumptions related to the learning process:

- The expected value for performing a task decreases as experience is accumulated. The driver of this learning phenomenon is the amount of products assembled by an operator (autonomous learning, i.e. simply 'learning by doing').
- Each experience curve follows the shape of Wright's curve:

$$C_{ink} = \left((1-r)\cdot C_{i1}\right)\cdot n^{-b_k} + r\cdot C_{i1} \tag{1}$$

where n is the number of products assembled, C_{i1} is the initial expected time for the task i (e.g. the standard task time), and r is the fraction, fixed for all the tasks and stations, of C_{i1} that is unaffected by the learning process. The learning curve will then converge to a plateau value of $r\cdot C_{i1}$ for all stations. b_k is the positive learning rate related to each assembly station k. C_{ink} is the expected time for the task i after n products have been assembled by the station k.

- For each task i, the relative standard deviation is given by:

$$s_i = \sigma_i / C_{i1} \tag{2}$$

s_i is fixed, making the variance adjust to follow the changes in the expected value over time. This assumption avoids an unrealistic increase in relative task variability as a result of the learning phenomenon. Modelling variance decrease due to autonomous learning may be part of the further research agenda.

- The operators rotate in order to evenly distribute the learning experience across all tasks. This prevents the task assignment from being stuck in a single configuration due to the operators becoming overspecialised. The experience gained for each product assembled is split evenly between the stations involved in the assembly process. This consideration is already accounted for by the value n, calculated as:

$$n = N/S \tag{3}$$

where N is the overall number of units produced and S is the number of stations.

3. SOLUTION APPROACH

The objective is to find an effective and efficient way of computing the assembly line changes generated by the learning phenomenon without recalculating the whole line for each amount of processed products. Given a unitary increment of N, the Kottas-Lau methodology could be reinitialised, and the positioning of all the tasks could be redefined. Unless the learning effect is particularly effective or the initial solution is unstable under limited variations of C_{ink}, the calculation would probably lead to the same initial solution with a unitary increment of N. This procedure could be continued for different values of N until a change arises. At this point, some tasks would change their positions, being assigned to different stations. In this scenario, the learning curve for the moved tasks changes. In order to use the new curve while retaining the accumulated experience, a fictitious number of products n_{ik}^f must be computed for each moved task i. From (1) it follows that:

$$n_{ik}^f = \left(\frac{C_{ink'} - r\cdot C_{i1}}{C_{i1}\cdot(1-r)}\right)^{-\frac{1}{b_k}} \tag{4}$$

where b_k is the learning rate of the new station k to which the task i has been assigned, and k' indicates the station to which the task was previously assigned.

At this point, the accumulated experience calculation no longer relies on the overall relative number of products assembled n. The focus shifts to the incremental number of products after the last change ΔN and the incremental relative number of products Δn:

$$\Delta n = \Delta N / S_{new} \tag{5}$$

where S_{new} is the number of stations after the last positioning change. It can differ from the initial S since the movements that occurred might have eliminated one or more stations.

These calculations lead to a different way of computing Wright's curve:

$$C_{ink} = \left((1-r)\cdot C_{i1}\right)\cdot\left(n_{ik}^{f} + \Delta n\right)^{-b_k} + r\cdot C_{i1} \tag{6}$$

This new curve starts from $C_{ink'}$, i.e. where the other curve stopped, and proceeds with the correct slope for the newly assigned station k.

As has been shown, recalculation is a viable way for mapping the impact of the learning effect over time, but it is not very efficient. There is a different way of computing the changes generated by the produced units, without explicitly recalculating the system for each unitary ΔN. This way can be slightly more efficient, and it also provides a technical framework for further improvements that could lead to a better long-term mapping of the learning effect for this assembly problem, as explained in the following. The Kottas-Lau method is based on task categorisation. Given a set of tasks, free of precedence constraints, to be assigned to an open station k, a value of z_{ink} is calculated for each task:

$$z_{ink} = \frac{T - \sum_{i\in J} C_{ink}}{\sqrt{\sum_{i\in J}\sigma_{ink}^2}} \tag{7}$$

where J is the set of tasks already assigned to the station k. σ_{ink}^2 is the variance of the task i as calculated from the expected task time C_{ink} and the relative standard deviation s_r:

$$\sigma_{ink}^2 = (s_i \cdot C_{ink})^2 \tag{8}$$

The value of z_{ink} is directly affected both by the tasks already inside the station and by the learning phenomenon. This amount is confronted with a reference value, independent from the station and different among the tasks (named z'_{ink}), which is indirectly affected by the learning via the variations of C_{ink}:

$$z'_{ink} = inv\left(1 - \frac{c\cdot C_{ink}}{\sum_{w\epsilon K_i} I'_w}\right) \tag{9}$$

where $inv(X)$ is the inverse normal standard distribution given a cumulative probability X. K_i is a set containing the task i and the tasks that cannot be executed unless the task i has already taken place.

If the station is empty, the priority is given to the critical tasks where $z_{ink} < z'_{ink}$. Among these tasks, the ones with the largest amount of directly subsequent tasks in terms of precedence are prioritised. If no critical task is available, the task to be placed in the station is chosen among those where $z_{ink} \geq z'_{ink}$. Some of these tasks may have a value of z_{ink} higher than a certain safety level:

$$z^{safety} = inv(0.995) \tag{10}$$

If present, these safe tasks have priority over the unsafe ones, among which the choice is based on $\sum_{w\epsilon K_i} I'_w$, the higher the better. If only desirable tasks are available, i.e. tasks where $z_{ink} \geq inv(0.995) \geq z'_{ink}$, the choice is performed based on the $\sum_{w\epsilon K_i} I'_w$ value, this time the lower the better. The case with a non-empty station is similar, while in this scenario, the critical operations as defined above are not considered viable. If only such operations remain, a new station must be opened. It has been shown that both z_{ink} and z'_{ink} are affected by learning. In particular, both increase while C_{ink} rises for an incremental N. This means that the difference between the two values fluctuates in response to an increase in the amount of produced units.

A matrix P can be introduced, having a number of columns equal to the assignment attempts and a number of lines equal to the number of tasks. The matrix is filled with these values:

$$P_{ij} = \begin{cases} z_{ink} - z'_{ink} & case\ 1 \\ z_{ink} - inv(0.995) & case\ 2 \\ 0 & case\ 3 \end{cases} \tag{11}$$

Where the value of k considered in each cell is the one corresponding to the attempt j.

Case 1:

- In the attempt j, either a desirable task different from i or no task at all was chosen. While the attempt took place i was an unconstrained undesirable task and, if a different task was selected, $\sum_{w\epsilon K_i} I'_w$ was equal to or smaller than the same characteristic on the selected task.
- Or in the attempt j the critical task i was chosen.

Case 2:

- In the attempt j, a desirable task different from i was chosen. While the attempt took place, the unconstrained task i (desirable or not) had a $\sum_{w\epsilon K_i} I'_w$ value equal to or higher than the same characteristic on the selected task.
- Or in the attempt j a safe task, different from i, was chosen. While the attempt took place i was unconstrained and not safe, $\sum_{w\epsilon K_i} I'_w$ was equal or higher than the same characteristic on the selected task.

Case 3:

- In the attempt j, the task i was not described by the cases above.

When the matrix is generated, all its values are zero or negative. The non-zero cells contain all the cases in which the chosen task for the attempt j might have been different if i

had a positive matrix value. P can be recalculated for different values of N. When a value in the matrix switches from negative to zero or to positive, it means that, in some assignments, the precedence has switched and the assembly line needs to be recalculated. For the P_{ij} cells exhibiting case 2, it is also necessary to check $z'_{ink} \geq z^{safety} z'_{ink} \geq z^{safety}$. If this condition is not satisfied, that cell automatically becomes a case 1 and must be verified accordingly. Since both z_{ink} and z'_{ink} grow while N increases, a second matrix Q is needed to assess the potential decreases of $z_{ink} - z'_{ink}$ in the attempts in question. The second matrix is filled with these values:

$$Q_{ij} = \begin{cases} z_{ink} - z'_{ink} & case\ 1 \\ 0 & case\ 2 \end{cases} \tag{12}$$

Case 1:

- In the attempt j, a desirable or safe task i was chosen.
- Or in the attempt j, a critical task different from i was chosen. While the attempt took place, i was an unconstrained desirable or safe task. It had an amount of directly subsequent tasks equal or larger than the selected task.

Case 2:

- In the attempt j, the task i was not described by the case above.

When this matrix is generated, all its values are zero or positive. Over time, some differences can switch as an effect of the learning process. This makes a task that was once suitable for the assignment at hand no longer valuable, or in the case of empty stations, makes some tasks more critical and worth rearranging. Starting with $N = 1$ and increasing, both matrices can be recalculated. Their shapes do not change until an inversion is found. At this point, the initial assembly line structure is still valid, and the task positioning needs to be recomputed only from the attempt j, where the first inversion took place, to the end. The new line can feature a different number of stations and a different amount of attempts, thus the new matrices are shaped differently to the previous ones. While this search process is slightly more efficient than the standard line computation, it still requires various configurations of the same matrices to be checked before reaching a turning point. The process can be accelerated, for low learning coefficient, using numerical algorithms. For each non-zero cell, the approximate value of N required to make it zero can be computed. The turning point is achieved by constructing the lowest of these values. The difference between z_{ij} and z'_i can be expressed equivalently as:

$$\left(\int_{-\infty}^{T} \frac{1}{\sqrt{2\pi\sigma_{ink}^2}} \cdot e^{-\frac{\left(t - C_{ink} - \sum_{w \in J} C_{wnk}\right)^2}{2\sum_{w \in J} \sigma_{wnk}^2}} dt \right) - 1 + \frac{c \cdot C_{ink}}{\sum_{w \in K} l'_w} \tag{13}$$

While the difference between z_{ij} and $inv(0.995)$ becomes:

$$\left(\int_{-\infty}^{T} \frac{1}{\sqrt{2\pi\sigma_{ink}^2}} \cdot e^{-\frac{\left(t - C_{ink} - \sum_{w \in J} C_{wnk}\right)^2}{2\sum_{w \in J} \sigma_{wnk}^2}} dt \right) - 0.995 \tag{14}$$

In both expressions, the C_{ink} value can be substituted by the Wright's curve formula described above, leading to a compact representation of the learning phenomenon on the single operation given a task selection attempt. While the derivative of the integral part of the expression is hard to compute, an approximate derivative can still be applied in order for the expression to converge to its first zero. Since an iterative method of this kind is more costly than the original matrix recalculation, it is recommended to only apply it if the learning coefficients are low. In fact, low learning coefficients may require values of N far from 1 in order to achieve the first turning point.

4. CASE STUDY

The matrix methodology has been applied to a real assembly line. The assembly process involves 35 constrained tasks and a set of learning parameters with the values given in Table 1.

Table 1. Task and learning parameters

Parameter	Min	Max	Mean	STD
$C_{i1}\ [min]$	0.1030	3.0600	0.5336	0.5494
s_r	0.1178	1.0454	0.3354	0.1783
$l'_i\ [€/min]$	0.0145	1.6830	0.2898	0.3048
b_k	0.0074	0.0515	0.0276	0.0144

The line parameters are:

Table 2. Line parameters

$T\ [min]$	$c\ [€/h]$	r
2	22	0.5

The calculations are performed via the following steps:

- The assembly line is solved with the Kottas-Lau algorithm using the initial task characteristics. This is the same as computing the system for $n = 1$. While performing the attempts, the two matrices are created.
- The leaning process takes place and the non-zero elements of the matrix are recalculated with $\Delta N = 1$.
- If one or more elements of the matrices change their sign, the line is recalculated with the Kottas-Lau algorithm using the updated characteristics. If not, the previous step continues with a higher ΔN value, until a change is found or a maximum amount of iterations is reached – the latter is an early stopping condition.

The algorithm stops when a given number of Kottas-Lau recalculations have taken place. This is the time the assembly line has been rearranged.

The initial line in our example contains 12 stations. It is filled by tasks 1, 2, 3 and 24. After 17 steps ($\Delta N = 17$) a change is triggered in the third attempt, the task 5 becomes safe and disputes the position of task 3. After a line recalculation, the number of stations is reduced to 11 and task 5 takes the place of task 3 at the first station. As expected, all the tasks assigned during the first two attempts, i.e. 1 and 2, remain unaffected, while the tasks assigned after the third attempt are moved. For example, task 24 is moved to the end of station 2. The matrices are recalculated, and the increment on ΔN restarts from one. The system does not return to the previous state, since the second Kottas-Lau calculation has been performed using the values that have been updated to account for experience, and the matrices have been affected as well. In fact, the new matrix recalculations are based on these same updated characteristics. The next change takes place after 5 steps and involves the fourth attempt. In this attempt, no suitable station had been found, but, after the learning phase, task 24 becomes desirable and takes place as the fourth task at the first station. No station is eliminated by this update, and the tasks are simply rearranged more efficiently. This process can continue as long as the learning effect is able to move the status quo. Over time, the average assembly time approaches a theoretical limit, generated by the plateau in the experience curve. At this point the changes are no longer meaningful enough to trigger a line change. If the various matrices are memorised after each change, interpreting them can help us understand which tasks are less stable and which zones are more prone to changes. Interestingly, in the scenario at hand, the second matrix is never relevant to the triggering of a change. The low values of the learning rate also generate a diluting effect. In the matrices priming a change, no more than a single inversion can be found in the first twenty switches.

5. CONCLUSIONS AND EXTENSIONS

As has been shown, learning can significantly impact the performance of assembly systems. This is particularly true for those systems highly dependent on human labour. From the variability perspective, labour-intensive assembly lines cannot be correctly managed without taking their stochastic nature into account when assigning tasks to individual operators. In this paper, an effort has been made to join these two perspectives, by maintaining the algorithmic structure of a stochastic assembly line balancing and evaluating the changes imposed on the solution by the experience phenomenon over time. In particular, the recalculation of the whole system has been substituted by the calculation of two matrices signalling meaningful changes in the incumbent solution. Since this formulation slightly diminishes the computational effort, it could lead to even more significant savings if coupled with numerical algorithms, in particular in scenarios with low learning coefficients. A natural extension of this reasoning could focus on a deeper understanding of the savings brought about by the learning process. Changes in the assembly line take place after a defined numbers of units, and the unitary production cost follows a stair-shaped graph. A deeper analysis of the savings involved could lead to a more refined task allocation algorithm. At the present time, the Kottas-Lau heuristic does not account for the learning process, and a temporary, suboptimal solution could instead take advantage of the experience gained in the long term, leading to an overall better system. This strategy should take the product life cycle into account directly, since the final line configuration savings are not the only ones impacting on the production costs.

REFERENCES

Battaïa, O. and Dolgui, A. (2013). A taxonomy of line balancing problems and their solution approaches. *International Journal of Production Economics*, 142 (2), 259-277.

Baybars, I. (1986). A survey of exact algorithms for the simple assembly line balancing problem. *Management Science*, 32 (8), 909-932.

Bentaha, M.L., Dolgui, A. and Battaïa, O. (2016). A bibliographic review of production line design and balancing under uncertainty. *IFAC-PapersOnLine*, 8 (3), 70-75.

Biskup, D. (1999). Single-machine scheduling with learning considerations. *European Journal of Operational Research*, 115 (1), 173-178.

Boysen, N., Fliedner, M. and Scholl, A. (2007). A classification of assembly line balancing problems. *European Journal of Operational Research*, 183 (2), 674-693.

Chica, M., Bautista, J., Cordón, Ó and Damas, S. (2016). A multi objective model and evolutionary algorithms for robust time and space assembly line balancing under uncertain demand. *Omega*, 58, 55-68.

Chutima, P. and Naruemitwong, W. (2014). A Pareto biogeography-based optimisation for multi-objective two-sided assembly line sequencing problems with a learning effect. *Computers & Industrial Engineering*, 69 (1), 89-104.

Cohen, Y. and Dar-El, M.E. (1998). Optimizing the number of stations in assembly lines under learning for limited production, *Production Planning and Control*, 9 (3), 230-240.

Cohen, Y., Vitner, G. and Sarin, S.C. (2006). Optimal allocation of work in assembly lines for lots with homogenous learning, *European Journal of Operational Research*, 168 (3), 922-931.

Dolgui, A., Gordon, V. and Strusevich, V. (2012). Single machine scheduling with precedence constraints and positionally dependent processing times. *Computers & Operations Research*, 39 (6), 1218-1224.

Erel, E. and Sarin, S.C. (1998). A survey of the assembly line balancing procedures. *Production Planning and Control*, 9 (5), 414-434.

Gamberini, R., Gebennini, E., Grassi, A. and Regattieri, A. (2009). A multiple single-pass heuristic algorithm solving the stochastic assembly line rebalancing problem.

International Journal of Production Research, 47 (8), 2141-2164.

Gamberini, R., Grassi, A. and Rimini, B. (2006). A new multi-objective heuristic algorithm for solving the stochastic assembly line re-balancing problem. *International Journal of Production Economics*, 102 (2), 226-243.

Hamta, N., Fatemi Ghomi, S.M.T., Jolai, F. and Akbarpour Shirazi, M. (2013). A hybrid PSO algorithm for a multi-objective assembly line balancing problem with flexible operation times, sequence-dependent setup times and learning effect. *International Journal of Production Economics*, 141 (1), 99-111.

Hamta, N., Fatemi Ghomi, S.M.T., Jolai, F. and Bahalke, U. (2011). Bi-criteria assembly line balancing by considering flexible operation times. *Applied Mathematical Modelling*, 35 (12), 5592-5608.

Jaber, M.Y. and Glock, C.H. (2013). A learning curve for tasks with cognitive and motor elements. *Computers & Industrial Engineering*, 64 (3), 866-871.

Kottas, J.F. and Lau, H.S. (1973). A cost oriented approach to stochastic line balancing. *AIIE Transactions*, 5 (2), 164-171.

Li, Z., Tang, Q. and Zhang, L. (2017). Two-sided assembly line balancing problem of type I: Improvements, a simple algorithm and a comprehensive study. *Computers & Operations Research*, 79, 78-93.

Lolli, F., Gamberini, R., Gamberi, M. and Bortolini, M. (2017). The training of suppliers: a linear model for optimizing the allocation of available hours. *International Journal of Industrial and Systems Engineering*, in press.

Lolli, F., Gamberini, R., Giberti, C., Gamberi, M., Bortolini, M. and Bruini, E. (2016a). A learning model for the allocation of training hours in a multistage setting. *International Journal of Production Research*, 54 (19), 5697-5707.

Lolli, F., Messori, M., Gamberini, R., Rimini, B. and Balugani, E. (2016b). Modelling production cost with the effects of learning and forgetting. *IFAC-PapersOnLine*, 49 (12), 503-508.

Pan, E., Wang, G., Xi, L., Chen, L. and Han, X. (2014). Single-machine group scheduling problem considering learning, forgetting effects and preventive maintenance. *International Journal of Production Research*, 52 (19), 5690-5704.

Salveson, M.E. (1955). The assembly line balancing problem. *The Journal of Industrial Engineering*, 6 (3), 18-25.

Toksarı, M.D., Isleyen, S.K., Güner, E. and Baykoç, Ö.F. (2008). Simple and U-type assembly line balancing problems with a learning effect. *Applied Mathematical Modelling*, 32 (12), 2954-2961.

Toksarı, M.D., Isleyen, S.K., Güner, E. and Baykoç, Ö.F. (2010). Assembly line balancing problem with deterioration tasks and learning effect. *Expert Systems with Applications*, 37 (2), 1223-1228.

Wright, T.P. (1936). Factors affecting the cost of airplanes. *Journal of Aeronautical Science*, 3 (2), 122-128.

Yuguang, Z., Bo, A. and Yong, Z. (2016). A PSO algorithm for multi-objective hull assembly line balancing using the stratified optimization strategy. *Computers & Industrial Engineering*, 98, 53-62.