Contents lists available at GrowingScience

International Journal of Industrial Engineering Computations

homepage: www.GrowingScience.com/ijiec

Simple assembly line balancing problem under task deterioration

M. Emrani Noushabadi[a], U. Bahalke[a*], K. Dolatkhahi[a], S. Dolatkhahi[b] and A. Makui[a]

[a]*Department of Industrial Engineering, Iran University of Science & Technology, Tehran, Iran*
[b]*Payam e noor University of Tehran department of Industrial Engineering, Iran, Tehran*

A R T I C L E I N F O

Article history:
Received 18 December 2010
Received in revised form
18 February 2011
Accepted 21 February 2011
Available online
22 February 2011

Keywords:
Assembly line balancing
Scheduling
Task deterioration
Genetic algorithm

A B S T R A C T

This paper introduces the effect of task deterioration in simple assembly line balancing problem. In many realistic assembly lines, a deterioration task is considered when a task is started earlier than the assigned time since the station time is constant and the earliness of the task does not reduce the cycle time. This phenomenon is known as deteriorating tasks. Therefore, we seek an optimal assignment and schedule of tasks in workstations, in order to minimize the number of stations for a given cycle time, which is known as SALBP-1. For this purpose, a mathematical model is proposed. Since the pure SALBP-1 is proved to be NP-hard and considering task deterioration complicates problem further, we propose a genetic algorithm for solving such problem. Several well-known test problems are solved to study the performance of the proposed approach.

1. Introduction

The initial study in assembly line balancing (ALB) problem is in assigning tasks to work concentrating on precedence relationship among the tasks to optimize some measurable performance (Erel & Sarin, 1998). Bryton (1954) conducted the first attempt in assembly line balancing and Salveson (1955) aimed to publish the first work in ALB problem. The objective functions could be categorized in two distinctive points of view, including minimizing cycle time in constant work stations, which is defined in (SALBP-2), and minimizing the workstation in fixed cycle times, which is defined in (SALBP-1) (Becker & Scholl, 2006). The workstation is a set of tasks that are to be assigned to some resources and the cycle time is defined as the time between completions of successive products. Kara et al. (2009), Özcan and Toklu (2009), Scholl and Becker (2006), Lapierre et al.(2006), Kim et al.(2009), Levitin et al.(2006), Andre´s et al.(2008) and Toksarı et al.(2008) addressed the ALB problem with different objective functions and the cycle time is considered to be fixed in most of these papers.

Although in a real situation, the cycle time could be increased due to tardiness and lag caused by the precedence workstation, Gupta and Gupta (1990) and Browne and Yechialli (1990) made the first endeavor in this field by considering a dependent function on starting time for processing of elements in workstation. A fire fighting example is introduced by Kunnathur and Gupta (1990), in which the time for necessary measures to fight the fire increases if the actions are started by delay. Other similar

* Corresponding author. Tel: +982173220000
E-mail: unesbahalke@gmail.com (U. Bahalke)

doi: 10.5267/j.ijiec.2011.02.003

examples for the aforementioned could be found in line production systems, financial and accounting management, etc.

Owing to the wide application of the processing time in different conditions, a wide variety of models based on time for processing have been developed by Ji and Cheng (2009, 2008), Wang et al. (2009a, 2009b, 2009c), Low et al. (2008) and Yin et al. (2010). Especially, in line-balancing problem, Toksar (2010) considered machine-human fault and considered aspects altogether to provide an optimal solution in (SALBP-1). A comprehensive review of literature of the assembly-line balancing problem reveals that there is no hypothesis that considers task-deterioration. Considering the fact that this characteristic aids in applying more realistic definitions of SALBP compared with traditional models and formula in processing time of deteriorating tasks, the hypothesis is of great importance. The problem description is asserted in section 2, which is followed by mathematical modeling in section 3. A developed genetic model for SALBP is presented in section 4 and ultimately two well-known examples are presented and solved by mathematical and developed GA and the results are compared for two methods for small and large-scale problems. In small-scale problem, the mathematical modeling is proposed to provide the optimum solution and for large-scale problems, the comparison is executed while the effects of deterioration are not considered.

2. SALBP under task deterioration

The assembly line is defined as a group of workstations, which is a place in which some specific operations are performed on products, and certain task scheduled for each workstation. Passing time in workstation for products is known as cycle time, (C) and equivalent to the time interval between successively completed production units. The number of tasks is shown by N and should be assigned in workstations. The processing time function is presented as the following equation,

$$P_j = a_j + b_j \times T_j \tag{1}$$

Task-deterioration is considered in Eq. (1). P_j is the task time of task j, a_j is the fixed time in processing time, b_j is the deterioration growth rate and T_j denotes the delay in processing time and is defined as a gap time from the beginning up to starting of the tasks. The assumption of lag-time at the beginning time of tasks in this paper differs from the specific scheduling problem due to precedence relationship among the tasks. The tasks are considered available from the beginning in scheduling problem, although in assembly line the precedence relationships and the nature of tasks result in preventing to use this assumption, which is typically used in scheduling problem. The Eq. (1) is not used in this field and some adjustment seems to be necessary. Consequently, the developed form of Eq. (1) is presented in Eq. (2) which considers the available time and start time of each task, separately,

$$P_j = a_j + b_j \times (ST_j - AV_j) \tag{2}$$

The start-time of task is denoted by ST_j and the available-time is presented by AV_j. In assembly line, each task can be started only when all predecessors have finished and the available-time is known as the time when all the related predecessors are finished. The delay-time of the task is modeled as the difference between the start-time and available-time. A clarifying example of such predecessors is shown in Fig. 1. The available-time for task d, is the maximum end-time of all its predecessors (b, c). The delay-time for task d, in Fig. 2 equals the difference of start-time and end-time. The problem is formulated as follows,

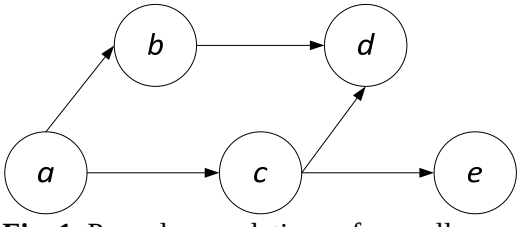

Fig. 1. Precedence relations of a small sample

$a \rightarrow b$	$c \rightarrow d$	…	…
Station 1	Station 2	…	Station k

Fig. 2. Sequence of tasks in stations for a small sample

3. Mathematical formulation

The objective is to assign and schedule tasks to workstations so as to minimize the number of work stations for a given cycle time (SALBP-1). So there are N tasks ($i \in \{1 \ldots N\}$) that should be assigned and scheduled and it is evident that the minimum value of the number of stations is equal to $(\sum_{j=1}^{N} P_j)/C_t$ and the maximum value of it equals the number of all tasks (each task is positioned in one separate station). Note that this model includes one assembly cycle, which means that every output product is essentially an output of the line. Additional parameters and variables are as follows,

AV_i The available time of task *i*,
a_i Fixed part of the processing time of job *i*,
b_i The deterioration growth rate of job *i*,
C_i Completion time of task *i*,
Ct Cycle time,
pre_i Set of immediate predecessors of job *i*,
i Designates the Tasks,
j Designates the positions ({1,.., N}) of jobs in a sequence $j \in I = \{1,...,N\}$,
Jp Set of jobs that have predecessor,
k It designates the workstations. $k \in m = \{Z_{\min},...,Z_{\max}\}$,
M A large positive number,
N Number of all tasks,
Np Set of tasks that have no predecessor,
P_j Processing time of task in position *j*,
q_k The processing time of station *k*,
$Start_i$ The starting time of task *i*,
ST_j The starting time of task in position *j*,
S_{jk} Starting time of task in position *j* and assigned in station *k*,
T_k The starting time of station *k* in each production cycle that final product exists from line,
Z Number of occupied work stations ($Z \in \{Z_{\min},.., Z_{\max}\}$).

The optimization model can be formulated as follows.

Decision variables:

$$x_{jk} = \begin{cases} 1 & \text{if task in position } j \text{ is assigned to station } k \\ 0 & \text{otherwise} \end{cases}$$

$$y_k = \begin{cases} 1 & \text{if station } k \text{ is occupied by any task} \\ 0 & \text{otherwise} \end{cases}$$

$$u_{ij} = \begin{cases} 1 & \text{if task } i \text{ is assigned to position } j \\ 0 & \text{otherwise} \end{cases}$$

Objective function:

$$\min Z \tag{3}$$

subject to:

$$Ct \geq \sum_{j=1}^{N} (x_{jk} \times p_j) \qquad k \in m \tag{4}$$

$$q_k = \sum_{j=1}^{N} (x_{jk} \times p_j) \qquad k \in m \tag{5}$$

$$\sum_{i=1}^{N} u_{ij} = 1 \qquad j \in I \qquad (6)$$

$$\sum_{j=1}^{N} u_{ij} = 1 \qquad i \in I \qquad (7)$$

$$Z = \sum_{k=1}^{Z\max} y_k \qquad (8)$$

$$y_k \geq y_{k+1} \qquad k \in m \qquad (9)$$

$$y_k \geq x_{jk} \qquad j \in I,\ k \in m \qquad (10)$$

$$\sum_{k=1}^{Z} x_{jk} = 1 \qquad j \in I \qquad (11)$$

$$P_j = \sum_{i=1}^{N} u_{ij} \times (a_i + b_i \times (ST_j - AV_i)) \qquad j \in I \qquad (12)$$

$$ST_i = ST_{i-1} + P_{i-1} \qquad j \in I \qquad (13)$$

$$T_S = \sum_{k=1}^{S-1} q_k \qquad S \in m \qquad (14)$$

$$S_{jk} + M \times (1 - x_{jk}) \geq T_k \qquad j \in I,\ k \in m \qquad (15)$$

$$\sum_{k=1}^{Z} (S_{jk} \times x_{jk}) = ST_j \qquad j \in I \qquad (16)$$

$$Start_i = \sum_{j=1}^{N} u_{ij} \times ST_j \qquad i \in I \qquad (17)$$

$$C_i = \sum_{i=1}^{N} u_{ij} \times (ST_j + P_j) \qquad i \in I \qquad (18)$$

$$AV_i = \max\{C_r\} \qquad r \in pre_i \quad , i \in Jp \qquad (19)$$

$$Start_i \geq C_r \qquad i \in Jp,\quad r \in pre_i \qquad (20)$$

$$AV_i = 0 \qquad i \in Np$$

$$ST_1 = 0$$

Eq. (3) is the objective function of the model, which aims to minimize the number of workstations; Eq. (4) describes the workstation time limitation that must be less than cycle time. Constraint (5) denotes the total operating time for each station. Eq. (6) requires that each position in sequence must be filled with a task and the one-to-one correspondence between the tasks and positions is shown in Eq. (7). Eq. (8) calculates the number of workstations, and Eq. (9) ensures that the action of task assignments starts from the first workstation; as a result of such constraint, one can conclude that there is s no empty station between two active stations. Eq. (10) ensures that if a position is assigned to a station, the station must be necessarily selected and counted as the number of stations. Eq. (11) ensures that each position can be assigned only to one station. Eq. (12) denotes the task-time of position *j*. Eq. (13) calculates the start-time of the task in position *j*, and Eq. (14) counts for the start-time of station *k* in each cycle. Constraint (15) requires that the start-time of the task in position *j* and station *k* must be equal or more than the start-time of station. Eq. (16) denotes the equality of start-time of the task in position *j* and the task in position *j* in station k. Eqs. (17-19) are respectively in charge of the start time, completion time and available time for task *i*, and finally, constraint (20) guarantees the precedence relationships among the tasks.

SALBP is classified in NP-hard problems and the considered problem in this paper with additional issue of task deterioration is more complicated. As known, the mathematical models are not able to solve this complex category of problems in the reasonable computational time for large-scale

problems. Therefore, a GA is developed, which is known as a popular and the most practical meta-heuristic algorithm for this field of problems.

4. Genetic algorithm for SALBP-1

Most of the problems dealing with the design of production systems are categorized as combinational and NP-Hard problems. The assembly-line balancing problems, among the combinational optimization problems, are also categorized as NP-Hard (Karp, 1972). In recent years, most researchers have used genetic algorithm to solve such problems (Kim et al. 2009, Chen et al. 2002).
In genetic algorithm, the initial population of solutions, which appear as a set of feasible solutions (*chromosomes*), are selected randomly. The population is evaluated and the corresponding fitness function is calculated. The amount of fitness function provides a relative preference for the chromosomes; chromosomes with higher fitness, present a better solution. Taking steps toward optimization is achieved by *generation*, which is stated in the following form:

1. *Selection*: Some chromosomes are selected from the current population, note that chromosomes with higher fitness have more chance to be selected.
2. *Crossover*: The selected chromosomes mate and generate offsprings
3. *Mutation*: Some chromosomes of the new population mutate

This process is repeated until the solution is achieved (Goldberg 1989, Michalewicz 1996). In the developed GA, the number of iterations depends on the scale of the problems, which is considered $100\times n$ (n is the number of all tasks) as this number of iterations has resulted in good outputs after testing several values for the number of iterations. The structure of proposed GA is described in following sections.

4.1 Representation and initial population

Each feasible solution is represented by a permutation of tasks, which satisfies the precedence relationships among the tasks. This type of representation clearly determines the sequence of tasks. The number of stations for each individual is calculated through the following statements. Beginning from the first positioned task, tasks are assigned to the current station such that their sequence is maintained. Whenever the sum of processing times in a station exceeds the cycle time, the task is assigned to the next station. Initial population is generated for a certain number such that the precedence relations are observed. We have chosen 50 through experimental evidences. The structure of a chromosome is depicted in Fig. 3.

Station 1				Station 2			Station 3	
1	2	4	-1	7	5	-1	3	6

Fig. 3. Structure of the chromosome

4.2.Selection mechanism

The popular Roulette wheel selection mechanism is used in this work (Sivanandam & Deepa, 2008). After selection, the chromosomes are mated to produce a new population.

4.3. Genetic operators

The developed GA applies the popular precedence preservative crossover (PPX) (Sivanandam & Deepa 2008). After exerting crossover, some of individuals may undergo a mutation process, which selects a random task and exchanges it with preceding or following task randomly in a way that the precedence relationships are satisfied. This operator is executed with the probability of 0.1.

5. Computational results

First, the small example of Mertens, available at www.assembly-line-balancing.de, is investigated to examine the developed GA and the results are compared with the optimized solution. The precedence

relationships and task-times are denoted respectively in Fig 2, and Table 1. The model results and the applied assignments from the genetic algorithm, which is derived from Mertens model, are shown in Table 2. The remaining results from the model and the genetic algorithm solution of Mertens and Jackson are presented in Table 3. In this table, the number of workstation is determined considering different cycle-times proposed for the model. Also, the genetic algorithm is run 20 times for each instance; and the maximum, the minimum and the standard deviation (SD) of the results are calculated and finally, the optimized solution of preceding examples are presented in the last column, which denotes the efficiency of the presented model and genetic algorithm.

Next, from Tables 1 to Table 4, Mertens example is solved for different cycle times (10, 15 and 18) considering task deterioration and using the presented model. Deterioration rate is considered the same for all tasks. As can be seen, the results obtained from the genetic algorithm are the same as the results of the presented model. Next, in order to evaluate the efficiency of genetic algorithm, some examples are solved for large-scale problems using the model without considering task deterioration, and the solutions are compared with the optimal solutions. The final solutions of the proposed model of this paper for large-scale models are the same as the one reported by Buxey and Killbridge. For Lutz model, also, the solution of genetic algorithm is close to the optimal one. Table 10 includes the results showing the efficiency of genetic algorithm. Finally, the large-sized examples are solved including the task deterioration rates (0.1 and 0.2) and using the developed genetic algorithm. The results are presented in Table 11.

Table 1
Processing times of the tasks

Task	1	2	3	4	5	6	7
Processing time	1	5	4	3	5	6	5

As shown in most of the examples for large-scale problems, the effect of task deterioration in increasing the number of workstation is significantly more than the case of small-scale problems. For example, by considering Tables 4 and Table 5, when the deterioration rate changes from 0.1 to 0.2 there is no increase in the number of stations for the small-scale problem of Mertens, while for the same change in deterioration rate for Lutz's example with cycle-time of 20, the number of work-stations increases from 31 to 36 (Table 11).

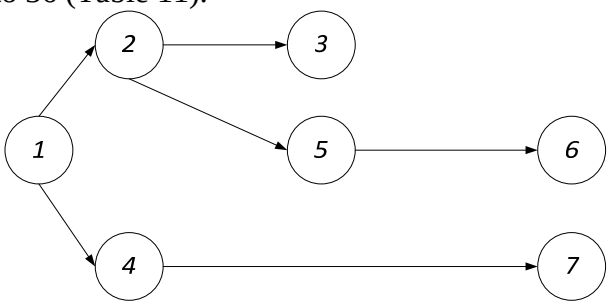

Fig 4. Precedence relations of Mertens

Table 2
The results of SALBP-1 with the developed model and proposed GA without the effect of task deterioration for Mertens (Ct=10)

Solution with developed model			Solution with proposed Genetic algorithm		
No. of station	Task	Task time	No. of station	Task	Task time
1	1	1	1	1	1
	2	5		4	3
	4	3		2	5
2	7	5	2	5	5
	5	5		7	5
3	3	4	3	6	6
	6	6		3	4

Table 3
The other results of SALBP-1 with proposed GA and developed model for the small-scale problems without task deterioration effect

Instance	Given cycle time	Model result (NO. of stations)	GA result (NO. of Stations)			Optimum solution (NO. of Stations (m*))
			Min	Max	S.D	
Mertens (7 tasks)	6	6	6	6	0	6
	7	5	5	5	0	5
	15	2	2	2	0	2
	18	2	2	2	0	2
Jackson (11 tasks)	7	8	8	8	0	8
	9	6	6	6	0	6
	11	5	5	5	0	5

Table 4
The results of SALBP-1 with the developed model and proposed GA with the effect of task deterioration for Mertens (Ct=10, deterioration=0.1)

Solution with developed model			Solution with proposed Genetic algorithm		
No. of station	Task	Task time	No. of station	Task	Task time
1	1	1	1	1	1
	4	3		2	5
	2	5.3		3	4
2	5	5	2	5	5.3
	3	4.5		4	4.43
3	6	6.45	3	7	5
4	7	7.125	4	6	6.943

Table 5
The results of SALBP-1 with the developed model and proposed GA with the effect of task deterioration for Mertens (Ct=10, deterioration=0.2)

Solution with developed model			Solution with proposed Genetic algorithm		
No. of station	Task	Task time	No. of station	Task	Task time
1	1	1	1	1	1
	4	3		4	3
	7	5		7	5
2	2	6.6	2	2	6.6
3	5	5	3	5	5
	3	5		3	5
4	6	7	4	6	7

Table 6
The results of SALBP-1 with the developed model and proposed GA with the effect of task deterioration for Mertens (Ct=15, deterioration=0.1)

Solution with developed model			Solution with proposed Genetic algorithm		
No. of station	Task sequence (up to down)	Task time	No. of station	Task sequence (up to down)	Task time
1	1	1	1	1	1
	2	5		2	5
	3	4		5	5
	4	3.9	2	3	4.5
2	7	5		4	4.45
	5	6.29		7	5
3	6	6	3	6	7.395

Table 7
The results of SALBP-1 with the developed model and proposed GA with the effect of task deterioration for Mertens (Ct=15, deterioration=0.2)

Solution with developed model			Solution with proposed Genetic algorithm		
No. of station	Task sequence (up to down)	Task time	No. of station	Task sequence (up to down)	Task time
1	1	1	1	1	1
	4	3		2	5
	7	5		4	4
2	2	6.6		3	4.8
	5	5	2	7	5.96
3	6	6		5	7.952
	3	6.2	3	6	6

Table 8
The results of SALBP-1 with the developed model and proposed GA with the effect of task deterioration for Mertens (Ct=18, deterioration=0.1)

solution with developed model			solution with proposed Genetic algorithm		
No. of station	Task sequence (up to down)	Task time	No. of station	Task sequence (up to down)	Task time
1	1	1	1	1	1
	2	5		4	3
	5	5		2	5.3
	4	4		7	5.53
2	7	5	2	3	4.553
	6	6.9		5	6.0083
	3	6.09		6	6

Table 9
The results of SALBP-1 with the developed model and proposed GA with the effect of task deterioration for Mertens (Ct=18, deterioration=0.2)

solution with developed model			solution with proposed Genetic algorithm		
No. of station	Task sequence (up to down)	Task time	No. of station	Task sequence (up to down)	Task time
1	1	1	1	1	1
	4	3		4	3
	7	5		7	5
	2	6.6		2	6.6
2	5	5	2	3	4
	3	5		5	5.8
	6	7		6	6

Table 10
The results of SALBP-1 with proposed GA for large-sized problems without task deterioration effect

		GA result			Optimum solution
Instance	Given cycle time	Min	Max	S.D	
Buxey (29 tasks)	30	12	12	0	12
	33	11	11	0	11
	36	10	10	0	10
	41	8	8	0	8
Killbridge (45 tasks)	57	10	10	0	10
	110	6	6	0	6
	138	4	4	0	4
Lutz2 (89 tasks)	11	51	51	0	49
	12	46	47	0.5	44
	19	28	28	0	26
	20	26	26	0	25
	21	25	25	0	24

Table11
The results of SALBP-1 with the proposed GA with the effect of task deterioration for large-scale problems with task deterioration effect of 0.1 and 0.2

Instances	Given cycle time	Task Deterioration 0.1			Task Deterioration 0.2		
		Min	Max	S.D	Min	Max	S.D
Buxey	30	14	14	0	15	15	0
	33	13	13	0	14	14	0
	36	11	11	0	13	13	0
	41	10	10	0	11	11	0
Killbridge	57	13	13	0	15	17	0.9574
	110	7	7	0	8	8	0
	138	5	5	0	7	7	0
Lutz2	19	33	33	0	39	41	1
	20	31	31	0	36	36	0
	21	30	31	0.5	35	35	0

6. Conclusion

In this paper, the effect of task deterioration with a new concept in simple assembly-line balancing problem type1 (SALBP-1) is studied for the first time. A mathematical model and a genetic algorithm are proposed. Several well-known examples have been solved without the effect of deterioration for verifying the effectiveness of proposed model and GA, and then the same problems have been considered with the effect of task deterioration. The results indicated that by increasing the impact of task deteriorations, the number of workstations or the value of cycle time increase. Therefore, the increasing in the number of workstations and cycle time stimulate managers to keep both factors in minimum range while tasks are affected by deterioration. Finally, we have concluded that the existence of deterioration manages to increase the number of workstations depending on the task deterioration rates, but these increments can be kept in minimum ranges by optimum scheduling and sequencing of deteriorated tasks in workstations.

References

Andre´s, C., Miralles, C., & Pastor, R. (2008). Balancing and scheduling tasks in assembly lines with sequence-dependent setup times. *European Journal of Operational Research*, 187, 1212–1223.

Becker, C. & Scholl, A. (2006). A survey on problems and methods in generalized assembly line balancing. *European Journal of Operational Research*, 168, 694–715.

Browne, S. & Yechiali, U. (1990). Scheduling deteriorating jobs on a single processor. *Operations Research*, 38, 495–498.

Bryton, B. (1954). Balancing of a continuous production line. MS Thesis, Northwestern University, Evanston, Illinois.

Chen, R.-S., Lu, K.-Y., & Yu, S.-C. (2002). A hybrid genetic algorithm approach on multi-objective of assembly planning problem. *Engineering Applications and Artificial. Intelligence*, 15, 447–457.

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

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization and machine learning*. Addision-Wesley, Reading, Massachusetts.

Gupta, J. N. D. & Gupta, S. K. (1988). Single facility scheduling with nonlinear processing times. *Computers & Industrial Engineering*, 14, 387–393.

Ji, M., & Cheng, T. C. E. (2009). Batch scheduling of simple linear deteriorating jobs on a single machine to minimize makespan. *European Journal of Operational Research*, 202, 90-98.

Ji, M., & Cheng, T. C. E. (2008), Parallel-machine scheduling with simple linear deterioration to minimize total completion time. *European Journal of Operational Research*, 188, 342–347.

Kara, Y., Paksoy, T., & Chang, C.-T. (2009). Binary fuzzy goal programming approach to single model straight and U-shaped assembly line balancing. *European Journal of Operational Research*, 195(2), 335–347.

Karp, R. M. (1972). Reducibility among combinatorial problems. In Complexity of computer applications, Eds R. E. Miller and J. W. Thatcher, 85–104, Plenum Press, New York.

Kim, Y. K., Song, W. S., & Kim, J. H. (2009). A mathematical model and a genetic algorithm for two-sided assembly line balancing. Computers & Operations Research, 36, 853–865.

Kunnathur, A. S. & Gupta, S. K. (1990). Minimizing the makespan with late start penalties added to processing times in a single facility scheduling problem. *European Journal of Operational Research*, 47, 56–64.

Lapierre, S. D., Ruiz, A., & Soriano, P. (2006). Balancing assembly lines with tabu search. *European Journal of Operational Research*, 168, 826–837.

Levitin, G., Rubinovitz, J., & Shnits, B. (2006). A genetic algorithm for robotic assembly line balancing. *European Journal of Operational Research*, 168, 811–825.

Low, C., Hsu, C.-J., & Su, C.-T. (2008). Minimizing the makespan with an availability constraint on a single machine under simple linear deterioration. *Computational Mathematics & Applications*, 56, 257–265.

Michalewicz, Z. (1996). Genetic algorithms + data structures evolution programs, 3rd edition, Springer, Berlin/Heidelberg, Germany.

Özcan, U., & Toklu, B. (2009). Multiple-criteria decision making in two-sided assembly line balancing: a goal programming and a fuzzy goal programming model. *Computers & Operations Research*, 36(6), 1955–1965.

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

Scholl, A., & Becker, C. (2006). State-of-the-art exact and heuristic solution procedures for simple assembly line balancing. *European Journal of Operational Research*, 168, 666–693.

Sivanandam, S. N., & Deepa, S. N. (2008). *Introduction to genetic algorithms*, Springer, Berlin/Heidelberg/New York.

Toksari, M. D., Isleyen, S. K., Güner, E., & Baykoc, Ö. F. (2008), Simple and U-type assembly line balancing problems with a learning effect. *Applied Mathematical Modeling*, 32, 2954–2961.

Toksari, M.D., Isleyen, S. K.,Güner, E., & Bayko¸c, Ö. F. (2010). Assembly line balancing problem with deterioration tasks and learning effect. *Expert Systems with Applications*, 37(2), 1223–1228.

Wang, J.-B., Gao, W.-J., Wang, L.-Y., & Wang, D. (2009a). Single machine group scheduling with general linear deterioration to minimize the makespan. *International Journal of Advanced Manufacturing Technology*, 43, 146–150.

Wang, J.-B., Huang, X., Wang, X.-Y., Yin, N., & Wang, L.-Y. (2009b). Learning effect and deteriorating jobs in the single machine scheduling problems. *Applied Mathematical Modeling*, 33, 3848–3853.

Wang, J.-B., Wang, L.-Y., Wang, D., & Wang, X.-Y. (2009c). Single-machine scheduling with a time dependent deterioration. *International Journal of Advanced Manufacturing Technology*, 43(7–8), 805–809.

Yin, N., Wang, J.-B., Wang, D., Wang, L.-Y., & Wang, X.-Y. (2010). Deteriorating jobs and learning effects on a single-machine scheduling with past-sequence dependent setup times. *International Journal of Advanced Manufacturing Technology*, 46(5–8), 707–714.