Available online at www.sciencedirect.com
ScienceDirect
Applied Mathematics and Computation 181 (2006) 1471–1481
APPLIED MATHEMATICS AND COMPUTATION
www.elsevier.com/locate/amc

Multi-criteria sequencing problem for a mixed-model assembly line in a JIT production system

R. Tavakkoli-Moghaddam *, A.R. Rahimi-Vahed

Department of Industrial Engineering, Faculty of Engineering, University of Tehran, P.O. Box 11365/4563, Tehran, Iran

Abstract

Mixed-model assembly lines (MMAL) are a type of production lines where a variety of products models similar to product characteristics are assembled in a just-in-time (JIT) production system. There is a set of criteria on which to judge sequences of product models in terms of the effective utilization of these lines. In this paper, we consider three objectives simultaneously: (i) total utility work cost, (ii) total production rate variation cost, and (iii) total setup cost. In this study, these three objectives are first weighted by their relative importance weights and then a new mathematical model is presented. To solve this model, a memetic algorithm (MA) is proposed to determine suitable sequences. The performance of the MA is compared with the Lingo 6 software. A number of test problems are carried out to verify the good ability of the proposed MA in terms of the solution quality and computational time. The computational results reveal that the MA finds promising results, especially in the case of large-sized problems.

Keywords: Multi-criteria sequencing; Mixed-model assembly line; JIT system; Memetic algorithm

1. Introduction

Mixed-model assembly lines are a type of production lines that are capable of the diversified small lot production and are able to respond promptly to sudden demand changes for a variety of models without holding large inventories. The effective utilization of a mixed-model assembly line requires solving two problems in the sequential manner: (1) design and balance the line and (2) determine the production sequence for different models. In this paper, we assume that the line has already been balanced and a sequencing problem is only considered.

The sequence of introducing models to the mixed-model assembly line should be determined considering the main goals which are crucial to efficiently implement a just-in-time (JIT) production system. Numerous research efforts have been directed towards the development of computer approximation algorithms or heuristics as well as their subsequent implementation to solve mixed-model assembly lines. Monden [1] defines

* Corresponding author.
E-mail address: tavakoli@ut.ac.ir (R. Tavakkoli-Moghaddam).

0096-3003/$ - see front matter
doi:10.1016/j.amc.2006.02.033

two goals for the sequencing problem: (1) leveling the load of the total assembly line for each station on the line and (2) keeping a constant rate of usage for every part used in the line. To handle these problems, goal chasing I and II (GC-I and GC-II) are developed by Toyota Corporation. The GC-I minimizes the one-stage assuming that the length of the unique workstation is equal to zero. The GC-II solves the GC-I under special assumptions regarding the product structure.

Miltenburg [2] developed a non-linear programming model for the second above-mentioned goal. The time complexity function of the proposed program was exponential. Thus, he developed and solved the problem by applying two heuristic procedures. Miltenburg et al. [3] solved the same problem with a dynamic programming algorithm. Inman and Bulfin [4] solved the problem introduced in [2] by converting it to a new mathematical model. Other objectives were also considered by a number of researchers. Yano and Rachamdugu [5] minimized the total utility work. Bard et al. [6] considered an objective to minimize the overall line length. Okamura and Yamashina [7] developed a heuristic algorithm to minimize the risk of conveyor stoppage.

Tavakkoli-Moghaddam et al. [8,9] presented the optimal schedule and sequence of a set of jobs for a single machine with idle insert, in which the objective function is to minimize the sum of maximum earliness and tardiness (n/1/I/ETmax). Sequencing mixed-model assembly lines have also been studied as a multi-criteria problem. Bard et al. [10] developed a model involving two objectives as follows: (1) minimizing the overall line length and (2) keeping a constant rate of part usage. They solved the problem by using the weighted sum and they proposed a tabu search (TS) method to solve such a problem. Hyun et al. [11] addressed three objectives as follows: (1) minimizing total utility work, (2) keeping a constant rate of part usage, and (3) minimizing total setup cost. This problem was solved by proposing a new genetic evaluation and selection mechanism. Korkmazel and Meral [12] developed a weighted sum approach for two goals introduced by Monden [1]. McMullen and Fraizer [13] developed a simulated annealing (SA) method for the model used by McMullen [14] and they compared this SA against the TS method. McMullen [15–17] also solved the same problem by using genetic algorithms, Kohonen self-organizing map (SOM), and ant colony optimization, respectively. He also compared the performance of these three methods with SA and TS methods. Mansouri [18] has also solved the same problem with genetic algorithms in which a new selection mechanism has been introduced. A number of other metaheuristic methods can be applied to any other combinatorial optimization problem. Tavakkoli-Moghaddam et al. [19] proposed an efficient memetic algorithm (MA) with a simulated annealing-based local search engine in order to solve a new model of a cell formation problem (CFP) for a multi-period planning horizon.

In this paper, we consider three objectives simultaneously: (i) total utility work cost, (ii) total production rate variation cost, and (iii) total setup cost. The structure of this paper is as follows: In Section 2, we present the detailed description of the mixed-model assembly line (MMAL). In Section 3, we discuss about the complexity of the proposed model and propose the memetic algorithm to solve such a hard model. Section 4 provides experimental results in which a number of test problems are solved to show the efficiency of the proposed MA. Finally, we present our conclusions in Section 5.

2. Multi-criteria sequencing problem for the MMAL

2.1. Mixed-model assembly line

In this paper, the MMAL is a conveyor system moving at a constant speed (v_c). Similar products are launched onto the conveyor at a fixed rate. The line is partitioned into J stations. It is assumed that the stations are all closed types. A closed station has boundaries in which workers cannot cross. Such a closed station is often found in reality where the use of facilities is restricted within a certain boundary. The tasks allocated to each station are properly balanced and their operating times are deterministic. The worker moves downstream on the conveyor while performing his/her tasks to assemble a product. To complete the job, the worker moves upstream to the next product. Suppose that the worker's moving time is ignored.

The design of the MMAL involves several issues such as determining operator schedules, product mix, and launch intervals. Two types of operator schedules (i.e., early start schedule and late start schedule) are found in [6]. An early start schedule is more common in practice and is used in this paper. Second, a minimum part set (MPS) production, which is a strategy widely accepted in the mixed-model assembly lines, is also used in this

paper. MPS is a vector representing a product mix, such that $(d_1,\ldots,d_M)=(D_1/h,\ldots,D_M/h)$; where M is the total number of models, D_M is the number of products of the model type M which needs to be assembled during the entire planning horizon and h is the greatest common divisor or highest common factor of $D_1,D_2,\ldots,D_M$. This strategy operates in a cyclical manner. The number of products produced in one cycle is given by $I=\sum_{i=1}^{M}d_i$. Obviously, h times the repetition of producing the MPS products that can meet the total demand in the planning horizon. Third, the launch interval (γ) is set to $T/(I\times J)$ where T is the total operation time required to produce one cycle of MPS products [11] (Fig. 1).

2.2. Objective functions

2.2.1. Minimizing the total utility work cost

The utility work is typically handled by the use of utility workers who assist the regular workers during work overload. Let L_j be the fixed line length of station j and U_{ij} be the amount of utility work required for the ith product in a sequence at station j. The following model is an extension of the model presented in [11].

$$\text{Min} \quad \sum_{j=1}^{J}\left[w_j\left(\sum_{i=1}^{I}U_{ij}+Z_{(i+1)j}/v_c\right)\right] \tag{1}$$

$$\text{s.t.} \quad \sum_{m=1}^{M}x_{im}=1 \quad \forall i, \tag{2}$$

$$\sum_{i=1}^{I}x_{im}=d_m \quad \forall m, \tag{3}$$

$$Z_{(i+1)j}=\max\left[0,\min\left(Z_{ij}+v_c\sum_{m=1}^{M}x_{im}t_{jm}-w,L_j-w\right)\right] \quad \forall i,j, \tag{4}$$

$$U_{ij}=\max\left[0,\left(Z_{ij}+v_c\sum_{m=1}^{M}x_{im}t_{jm}-L_j\right)\Big/v_c\right] \quad \forall i,j, \tag{5}$$

$$x_{im}=0 \text{ or } 1 \quad \forall i,m, \tag{6}$$

$$Z_{1j}=0, \quad Z_{ij}\geqslant 0 \quad \forall i,j, \tag{7}$$

$$U_{ij}\geqslant 0 \quad \forall i,j, \tag{8}$$

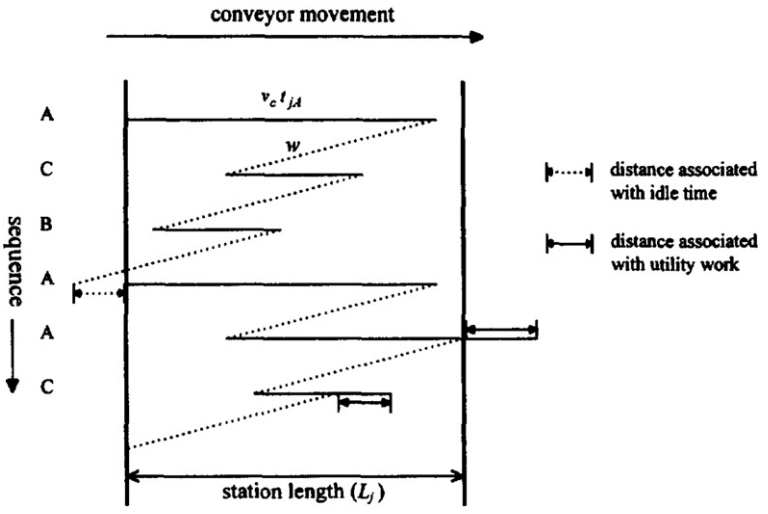

Fig. 1. Operations in a closed workstation.

where w_j is the sequence-independent utility work cost per unit time at station j involving direct and indirect utility labor costs. Z_{ij} is the starting position of the work on the ith product in a sequence at station j, and x_{im} is 1 if the ith product in a sequence is in model m; otherwise x_{im} is 0. The second term in the objective function accounts for the utility work that may be required at the end of a cycle. Eq. (2) ensures that exactly one product is assigned to each position in a sequence. Eq. (3) guarantees that demand for each model is satisfied. Eq. (4) indicates the starting position of the worker at each station j on product $i+1$ in a sequence. Utility work U_{ij} for the ith product in a sequence at station j is determined by Eq. (5).

2.2.2. Minimizing total production rate variation cost

One basic requirement of JIT systems is continual and stable for the part supply. Since this can be realized when the demand rate of parts is constant over time, the objective is significant to a successful operation of the system. Thus, the objective can be achieved by matching demand with the actual production. In this paper, the following model is used which is developed and modified according to the model proposed by Miltenberg [2].

$$\text{Minimize} \quad \sum_{i=1}^{I}\sum_{m=1}^{M}\left(v_{im}\left|\sum_{l=1}^{i}\frac{x_{lm}}{i}-\frac{d_m}{I}\right|\right) \tag{9}$$

s.t. Eqs. (2), (3) and (6),

where v_{im} is the sequence-dependent production rate variation cost for the product model m in the ith position of the sequence. This cost penalizes the difference between the production rate and actual demand. The first term in the objective function is the production ratio of model m until the ith product is produced. The second term is the demand ratio of model m.

2.2.3. Minimizing total setup cost

In many industries, sequence-dependent setups are considered as an important issue in assembly operations. The model considering sequence-dependent setups developed by Hyun et al. [11] is considered in this paper.

$$\text{Minimize} \quad \sum_{j=1}^{J}\sum_{i=1}^{I}\sum_{m=1}^{M}\sum_{r=1}^{M}x_{imr}c_{jmr} \tag{10}$$

$$\text{s.t.} \quad \sum_{m=1}^{M}\sum_{r=1}^{M}x_{imr}=1 \quad \forall i, \tag{11}$$

$$\sum_{m=1}^{M}x_{imr}=\sum_{p=1}^{M}x_{(i+1)rp} \quad i=1,\ldots,I-1,\ \forall r, \tag{12}$$

$$\sum_{m=1}^{M}x_{Imr}=\sum_{p=1}^{M}x_{1rp} \quad \forall r, \tag{13}$$

$$\sum_{i=1}^{I}\sum_{r=1}^{M}x_{imr}=d_m \quad \forall m, \tag{14}$$

$$x_{imr}=0 \text{ or } 1 \quad \forall i,m,r, \tag{15}$$

where c_{jmr} is the setup cost required when the model type is changed from m to r at station j and x_{jmr} is 1 if model types m and r are assigned, respectively at the ith and $(i+1)$th position in a sequence; otherwise x_{jmr} is 0. Eq. (11) is a set of position constraints indicating that every position in a sequence is occupied by exactly one product. Eqs. (12) and (13) ensure that the sequence of products must be maintained while repeating the cyclic production. Eq. (14) imposes the restriction that all the demands must be satisfied in terms of MPS.

It is possible to combine the above three objectives, since the objective functions are as the same form; moreover, the set of constraints construct a feasible solution. Thus in the objective function, three objectives are weighted by their relative weights named λ_1, λ_2, and λ_3, respectively. Each of these weights must be less than or equal to 1.

3. MMAL solution algorithms

3.1. Combinatorial complexity

Finding production sequences with desirable levels of all objectives is NP-hard [11]. The total number of sequences for a mixed-model assembly sequencing problem can be computed as follows:

$$\text{Total sequences} = \frac{\left(\sum_{m=1}^{M} d_m\right)!}{\prod_{m=1}^{M}(d_m!)}. \tag{16}$$

As the problem increases in size, the number of feasible solutions increases in the exponential way. Thus, problems with large number possible solutions cannot be usually solved optimality within a reasonable amount of time [15].

3.2. The proposed memetic algorithm

In the 1980s, because of the genetic algorithm's weakness for local searches, a new class of knowledge-augmented genetic algorithms (GAs) was introduced in soft computer sciences. To solve our proposed model in this paper, a memetic algorithm is implemented as described in the following section.

3.2.1. Initialization

In the first step, a number of feasible solutions have to be introduced as the initial population named the first generation. To do so, a set of N chromosomes are generated at random. The chromosome ch_i is represented by a sequence of genes $(g_j|j = 1, \ldots, I)$. Each individual gene contains one bit of information: the jth product in a sequence is represented by the position of gene g_j in the sequence of the chromosome.

3.2.2. XP local search

If a new chromosome is produced in the initial population, then it is improved by using the pair wise exchange procedure (XP or 2-Opt method). In XP, the position of every pair of adjacent jobs is exchanged. This local search must be performed until the fitness function value is unchanged for η consecutive times.

3.2.3. Parent selection

A number of individuals are selected according to their fitness to create and form the next generation. This is carried out by using the roulette wheel selection without replacement [20,21]. The simplified roulette wheel considers the quality of parents. Each chromosome ch_i in the population has an associated selection probability P_i that is proportional to its fitness value f_i. First, all chromosomes are ranked in ascending order based on their fitness value. Then, the selection probability of each chromosome is calculated as follows:

$$p_i = \frac{1 + f_i}{N + \sum_{l=1}^{N} f_l}; \quad f_i = -\left(\frac{\lambda_1 \times f_{1i}(x)}{\max\limits_{l} f_{1l}(x)} + \frac{\lambda_2 \times f_{2i}(x)}{\max\limits_{l} f_{2l}(x)} + \frac{\lambda_3 \times f_{3i}(x)}{\max\limits_{l} f_{3l}(x)}\right); \quad l = 1, \ldots, N, \tag{17}$$

where $f_{ki}(x)$ is the kth objective function value for ith chromosome in the population. Then, the cumulative probability c_i is calculated for each chromosome and a uniform random number in the interval $[0, 1]$ will determine the selected parents until the desired number of parent is reached. In addition to the above assumptions, we also assume that each chromosome must be selected only once time. According to this assumption, the diversity of mating pool will increase.

3.2.4. Crossover

The order crossover (OX) [21] was selected for the model. To illustrate how it works, consider the following two parent sequences:

Parent 1: A A A A | A A B B B | B C C D D
Parent 2: D A B A | B C B A A | B C A D A

Brackets designate the portion of the sequences that will remain intact and become part of the offspring in the crossover process. The location of the brackets is determined at random, however the left bracket must be to the right of the first character in a sequence and the right bracket must be to the left of the last character in the sequence. A new parent is then created by moving all the characters appearing after the right bracket of the original parent at the beginning of the sequence. The results of this are shown below:

Parent 1′: B C C D | D A A A A | A A B B B (C D D A A A A B B)
Parent 2′: B C A D | A D A B A | B C B A A (C D D A A C B A A)

From this new sequence, characters that match the characters between the brackets of the other original parent are removed. For instance, from Parent 1′, the first two A's, the first two B's, and the first C are removed because Parent 2′ has the sequence BCBAA between its brackets. The sequence between the brackets of the original parent and this shortened list as shown in parentheses above from the other parent is then used to construct an offspring. For instance, the sequence AABBB is taken from Parent 1 and the shortened list from Parent 2′ is added starting at the right bracket and wrapping around to the beginning of the sequence. Using this technique, the following offspring are created [15].

Offspring 1: C B A A | A A B B B | C D D A A
Offspring 2: A A B B | B C B A A | C D D A A

3.2.5. Inversion

Inversion is a un-array operator that generates offspring from a single parent. It first chooses two random cut points in a parent. The elements between the cut points are then reversed. An example of the inversion operator is presented below:

Before inversion: C B A | B A B C | C A
After inversion: C B A | C B A B | C A

3.2.6. Mutation

The mutation operator changes one or some of the genes in a single parent randomly. This operator has been used to increase the diversity. In this paper, swapping mutation was used. Consider the following sequence:

Before mutation: B A A **A** C A B A B **B** C D D D

The two gray elements are randomly selected unique elements that are targeted for swapping. After swapping, or mutation, the sequence is as follows:

After mutation: B A A **B** C A B A B **A** C D D D

3.2.7. IP local search

If a chromosome is provided by the mutation, then to prevent the slow convergence of the algorithm, the insert procedure (IP) is utilized. In IP, each job is removed from its current position and inserted into another position. This local search must be performed until the fitness function value is unchanged for η consecutive times.

3.2.8. Elitist strategy

The idea of Elitist strategy is to bring the best chromosomes from the previous stage to the present stage without changing the gene structure. This ensures the best chromosomes can survive in the succeeding generation. In this paper, an elitist strategy is used during the optimization process. During the evolution, the best chromosomes in the solution pool will be identified and stored.

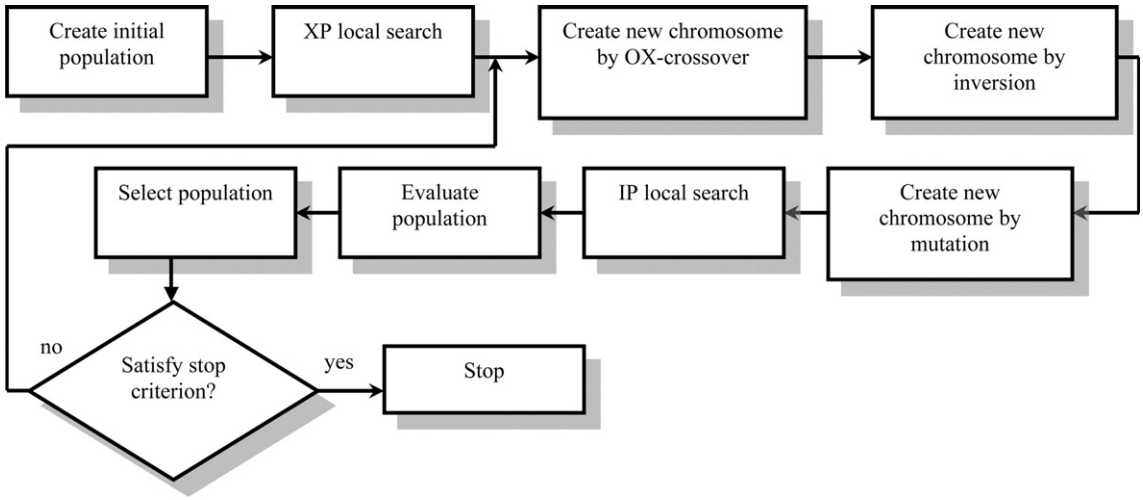

Fig. 2. Flowchart of the memetic algorithm.

3.2.9. Overall procedure

Fig. 2 depicts the flowchart for the general procedure of the proposed MA.

4. Experimental results

The performance of the proposed memetic algorithm (MA) is compared with the Lingo software. This algorithm is coded into the visual basic 6 and run on the Pentium 4, processor at 2.8 GHz and Windows XP using 256 MB of RAM. The experiments are implemented in two folds: first, for small-sized problems, the other for large-sized ones. For both of these experiments, we consider the following assumptions:

- *General assumptions:*
  1. All relative weights have the same importance and are equal to 1.
  2. All sequence-independent utility work costs, w_j, at each work station are identical and equal to 1.
  3. All sequence-dependent production rate variance costs, v_{im}, are identical and equal to 1.
  4. Sequence-dependent setup costs occur only at the first station.
  5. The velocity of conveyor, v_c, is set to 1 for the sake of computational convenience.
  6. Each experiment is repeated 10 times.

- *Memetic algorithm's assumption:*
  1. The number of new offspring is fixed to 50% of the population size.
  2. The ratio of the crossover OX, inversion, and mutation is set to 1, 0.5, and 0.076, respectively.
  3. The ratio of elitism is set to 0.5.

4.1. Small-sized problems

The first experiment is carried out on a set of small-sized problems. The basic configuration of the problem set follows the sequencing problem found in [6]. In this paper, the problem is considered for four workstations and three product models. The assembly time and the length of each station are shown in Table 1. Sequence-dependent setup costs are presented in Table 2. The nine MPSs shown in Table 3 are tested. From these MPSs, the number of possible sequences is computed from Eq. (16). The last column refers to the launch interval for each problem instance.

The population size, η, and termination criterion of MA are fixed to 50, 5, and 50 generations, respectively. For each instance, the results obtained are compared with the Lingo 6. Fig. 3 depicts the comparison of the

Table 1
Assembly time and workstation length

Workstation	Model			Workstation length
	1	2	3	
1	4	8	7	12
2	6	9	4	14
3	8	6	6	12
4	4	7	5	11

Table 2
Sequence-dependent setup cost

Model	Model		
	1	2	3
1	0	1	2
2	3	0	1
3	2	3	0

Table 3
Problem sets

Problem	I	MPS	No. of feasible solutions	Launch interval
1	9	(4, 3, 2)	1260	6.2
2	9	(3, 5, 1)	504	6.6
3	10	(5, 3, 2)	2520	6.1
4	10	(4, 4, 2)	3150	6.3
5	10	(4, 3, 3)	4200	6.1
6	11	(4, 6, 1)	2310	6.6
7	11	(6, 3, 2)	4620	6.0
8	11	(5, 3, 3)	9240	6.0
9	12	(6, 4, 2)	13,860	6.2

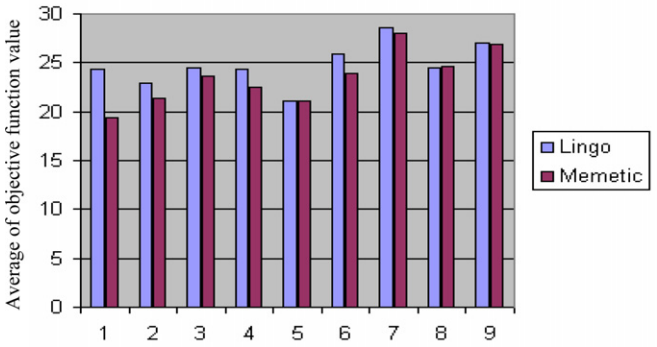

Fig. 3. Comparison of average of objective function value.

average of the objective function value obtained by the Lingo and MA, respectively. Table 4 shows total obtained results for nine test problems.

Column 4 in Table 4 presents the gap between the Lingo and MA. As shown in this column, the proposed MA is able to achieve and find better solutions than the Lingo in the given seven problems.

Table 4
Comparison of solution quality

Problem	Lingo experiment[a]	Memetic algorithm				$\frac{OFV_{Lingo} - Ave_{MA}}{OFV_{Lingo}} \times 100$
	OFV[b]	OFV				
		min	max	Ave	STD[c]	
1	24.36	19.4	19.4	19.4	0	20.36
2	22.89	21.07	21.42	21.32	0.13	6.86
3	24.42	23.39	24.32	23.67	0.47	3.07
4	24.30	22.01	22.66	22.48	0.15	7.49
5	21.05	21.05	21.42	21.15	0.23	−0.05
6	25.84	23.91	24.1	23.97	0.11	7.24
7	28.53	27.9	28.14	27.97	0.08	1.96
8	24.50	24.5	25.48	24.6	0.4	−0.4
9	27.05	26.84	27.05	26.92	0.22	0.04

[a] This column represents the Best-IP of Lingo.
[b] Objective function value.
[c] Standard deviation.

Table 5
Problem sets

Problem	I	MPS	No. of feasible solutions
1	20	(4,4,4,5,3)	2.44×10^{11}
2	20	(5,3,3,4,5)	1.95×10^{11}
3	20	(6,2,2,5,5)	5.87×10^{10}
4	30	(6,6,6,7,5)	1.18×10^{18}
5	30	(7,6,4,6,7)	8.39×10^{17}
6	40	(8,9,8,7,8)	6.80×10^{24}
7	40	(9,9,7,7,8)	6.05×10^{24}

4.2. Large-sized problems

Another experiment is implemented on large-sized problems. In this experiment, the assembly time (t_{jm}), the length of each workstation (L_j), and the setup cost (c_{jmr}) are generated from uniform distributions of U(4, 9), U(12, 15), and U(1, 3), respectively. The number of stations is fixed to 10 and the assembly line must produce five product models. Seven MPSs shown in Table 5 are provided and the proposed algorithm is applied to each of them. The population size, η, and the total number of generations of MA are increased to 200, 5, and 100, respectively. Table 6 shows total obtained results for seven test problems. The computational results shown in

Table 6
Comparison of the solution quality

Problem	Lingo experiment	Memetic algorithm			
		OFV			STD
		min	max	Ave	
1	*	93.85	97.8	95.66	0.72
2	*	94.26	98.88	96.14	1.32
3	*	93.27	96.85	95.32	1.23
4	*	134.99	142.21	137.92	2.66
5	*	137.67	144.52	141.44	2.14
6	*	172.82	181.41	177.01	2.50
7	*	176.64	185.21	182.67	3.31

* denotes that lingo cannot solve the problem.

Table 6 reveal that the proposed MA has the ability to compete with the Lingo from the quality point of view. That is very clearly true for large-sized problems in which the Lingo software cannot produce any results.

5. Conclusions

In this paper, a multi-criteria sequencing problem is studied for a mixed-model assembly line in just-in-time (JIT) environment. We have considered three objectives simultaneously, i.e. minimizing total utility work cost, total production rate variance cost, and total setup cost. Mathematical formulations for these objectives are proposed and the objective functions are combined with the weighted sum approach. Since the problem belongs to NP-hard one, memetic algorithm (MA) is proposed to solve it. The selection mechanism of MA is the roulette wheel without replacement. This algorithm uses three basic genetic operators such as crossover, inversion, and mutation. A hybrid local search is also implemented in this paper. Extensive computational experiments are performed for the proposed MA in two folds: for small and large-sized problems. The computational results are shown that the proposed MA has the ability to compete with the Lingo software, especially for large-sized problems.

Acknowledgements

The authors would like to acknowledge the Iran National Science Foundation (INSF) for the financial support of this work. We would also thank the scholars who recommended and helped through the preparation of this paper.

References

[1] Y. Monden, Toyota Production System, Institute of Industrial Engineers Press, Atlanta, 1983.
[2] J. Miltenburg, Level schedules for mixed-model assembly lines in just-in-time production systems, Management Science 35 (2) (1989) 192–207.
[3] J. Miltenburg, G. Steiner, S. Yeomans, A dynamic programming algorithm for scheduling mixed-model just-in-time production systems, Mathematical Computation Modeling 13 (1990) 57–66.
[4] P.R. Inman, R.L. Bulfin, Note on sequencing JIT mixed-model assembly lines, Management Science 3 (7) (1991) 904–910.
[5] C.A. Yano, R. Rachamadugu, Sequencing to minimize work overload in assembly lines with product options, Management Science 37 (1991) 572–586.
[6] J.F. Bard, E.M. Dar-El, A. Shtub, An analytic framework for sequencing mixed model, International Journal of Production Research 30 (1992) 35–48.
[7] K. Okamura, H. Yamshina, A heuristic algorithm for the assembly line model-mix sequencing problem to minimize the risk of stopping the conveyor, International Journal of Production Research 17 (1979) 233–247.
[8] R. Tavakkoli-Moghaddam, G. Moslehi, M. Vasei, A. Azaron, Optimal scheduling for a single machine to minimize the sum of maximum earliness and tardiness considering idle insert, Applied Mathematics and Computation 167 (2) (2005) 1430–1450.
[9] R. Tavakkoli-Moghaddam, G. Moslehi, M. Vasei, A. Azaron, A branch-and-bound algorithm for a single machine sequencing to minimize the sum of maximum earliness and tardiness with idle insert, Applied Mathematics and Computation 17 (1) (2006) 388–408.
[10] J.F. Bard, A. Shtub, S.B. Joshi, Sequencing mixed-model assembly lines to level parts usage and minimize the length, International Journal of Production Research 32 (1994) 2431–2454.
[11] C.J. Hyun, Y. Kim, Y.K. Kim, A genetic algorithm for multiple objective sequencing problems in mixed model assembly lines, Computers and Operations Research 25 (7/8) (1998) 675–690.
[12] T. Korkmazel, S. Meral, Bi-criteria sequencing methods for the mixed-model assembly line in just-in-time production systems, European Journal of Operational Research 131 (2001) 188–207.
[13] P.R. McMullen, G.V. Frazier, A simulated annealing approach to mixed-model sequencing with multiple objectives on a JIT line, IIE Transactions 3 (8) (2000) 679–686.
[14] P.R. McMullen, JIT sequencing for mixed-model assembly lines with setups using tabu search, Production Planning and Control 9 (5) (1998) 504–510.
[15] P.R. McMullen, An efficient frontier approach to addressing JIT sequencing problems with setups via search heuristics, Computers and Industrial Engineering 41 (2001) 335–353.
[16] P.R. McMullen, A Kohonen self-organizing map approach to addressing a multiple objective, mixed-model JIT sequencing problem, International Journal of Production Economics 72 (2001) 59–71.
[17] P.R. McMullen, An ant colony optimization approach to addressing a JIT sequencing problem with multiple objectives, Artificial Intelligence in Engineering 15 (2001) 309–317.
[18] S.A. Mansouri, A multi-objective genetic algorithm for mixed-model sequencing on JIT assembly lines, European Journal of Operational Research 167 (3) (2005) 696–716.

[19] R. Tavakkoli-Moghaddam, N. Safaei, M. Babakhani, Solving a dynamic cell formation problem with machine cost and alternative process plan by memetic algorithms, in: O.B. Lupanov, O.M. Kasim-Zade, A.V. Chaskin, K. Steinhofel (Eds.), Stochastic Algorithms: Foundation and Applications, Lecture Notes in Computer Science, vol. 3777, Springer-Verlag, Berlin, 2005, pp. 213–227 (IDS Number: BDQ12).
[20] D.E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley, Reading, MA, 1989.
[21] Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs, Springer, Berlin, 1996.