Intelligent Information Management, 2016, 8, 41-65
Published Online May 2016 in SciRes. http://www.scirp.org/journal/iim
http://dx.doi.org/10.4236/iim.2016.83005

Heuristics for Mixed Model Assembly Line Balancing Problem with Sequencing

Panneerselvam Sivasankaran[1], Peer Mohamed Shahabudeen[2]

[1]Department of Mechanical Engineering, Manakula Vinayagar Institute of Technology, Pondicherry, India
[2]Department of Industrial Engineering, College of Engineering, Anna University, Chennai, India
Email: sivasankaran.panneerselvam@yahoo.com, psdeen@gmail.com

Received 13 March 2016; accepted 28 May 2016; published 31 May 2016

Abstract

The growing global competition compels organizations to use many productivity improvement techniques. In this direction, assembly line balancing helps an organization to design its assembly line such that its balancing efficiency is maximized. If the organization assembles more than one model in the same line, then the objective is to maximize the average balancing efficiency of the models of the mixed model assembly line balancing problem. Maximization of average balancing efficiency of the models along with minimization of makespan of sequencing models forms a multi-objective function. This is a realistic objective function which combines the balancing efficiency and makespan. This assembly line balancing problem with multi-objective comes under combinatorial category. Hence, development of meta-heuristic is inevitable. In this paper, an attempt has been made to develop three genetic algorithms for the mixed model assembly line balancing problem such that the average balancing efficiency of the model is maximized and the makespan of sequencing the models is minimized. Finally, these three algorithms and another algorithm in literature modified to solve the mixed-model assembly line balancing problem are compared in terms of the stated multi-objective function using a randomly generated set of problems through a complete factorial experiment.

Keywords

Assembly Line Balancing, Genetic Algorithm, Crossover Operation, Mixed-Model, Model Sequencing, Makespan

1. Introduction

Companies engaged in mass production use assembly line balancing to design its assembly line by taking necessary input. Assembly line balancing is the process of grouping the tasks of precedence network of a product

How to cite this paper: Sivasankaran, P. and Shahabudeen, P.M. (2016) Heuristics for Mixed Model Assembly Line Balancing Problem with Sequencing. *Intelligent Information Management*, **8**, 41-65. http://dx.doi.org/10.4236/iim.2016.83005

that is assembled into a minimum number of workstations such that the sum of the task times in each of the workstations is less than or equal to the cycle time.

The cycle time is computed based on a given production volume per shift using the following formula [1].

$$\text{Cycle Time}(\text{CT}) = \text{Effective time available per shift} / \text{Production volume per shift} \tag{1}$$

The process of minimizing the number of workstations amounts to maximizing the balancing efficiency of the assembly line. This type of problem is called as assembly line balancing problem 1 (*ALBP* 1). In contrary, for a given number of workstations, the objective may be to minimize the maximum of the sum of the task times of workstations. This problem is called as assembly line balancing problem 2 (*ALBP* 2). This in turn maximizes the production volume per shift.

The balancing efficiency of the solution of the assembly line balancing problem is given by the following formula [1].

$$\begin{aligned}\text{Balancing efficiency} &= \left[\frac{\text{Sum of all task times}}{\text{Number of workstations} \times \text{Cycle time}}\right] \times 100 \\ &= \left[\left(\sum_{j=1}^{N} t_j\right) \Big/ (NS \times CT)\right] \times 100\end{aligned} \tag{2}$$

where,

N is the number of tasks.

t_j is the required time of the task j.

CT is the cycle time.

NS is the number of workstations.

Now-a-days, companies assemble more than one model in the same line mainly to meet the demands of its customers on continuous basis, which necessitates the use of mixed-model assembly line balancing. Here, the objective is to maximize the average balancing efficiency of the models. Further, sequencing of the models with minimum makespan along with the maximization of the average balancing efficiency of the models will improve the productivity of the company very much.

In this paper, the *ALBP* 1 problem with a mixed-model is considered. In this problem, there will be many models which are to be produced in batches using the same assembly line. The presence of a mixed-model makes the design of the assembly line more complex, in terms of processing times of the tasks and cycle times of the model. The average processing time (T_j) of each task as given by the following formula is normally taken as the representative value of the task time for that task.

$$T_j = \left(\sum_{i=1}^{m} t_{ij}\right) \Big/ NM_i \quad \text{for } j = 1, 2, 3, \cdots, N, \text{ where } t_{ij} \neq 0 \tag{3}$$

where,

t_{ij} is the time of the task j in the model i.

N is the number of tasks.

m is the number of models.

T_j is the average time of the task j.

NM_j is the number of models in which the processing time of the task j is more than zero.

Researchers in the past except [2] computed average task time for each of the tasks in the combined model. But, in this research, the average task time is not computed. Instead, the original task times of the models are used as such in the design of the assembly line without any modification, because it introduces perfection in the design of the assembly line in terms of reduced number of workstations.

The concept of combined network in the mixed-model assembly line balancing problem is demonstrated using two models, viz. Model 1 and Model 2, whose precedence networks are shown in **Figure 1** and **Figure 2**, respectively. The number by the side of each node represents the respective task time. The combined network of the two models is shown in **Figure 3**, which consists of the tasks of both models. The set of tasks in the combined model is the union of the tasks of the individual models.

In this paper, the mixed model assembly line balancing problem with sequencing of models is considered with the objective of maximizing the average balancing efficiency of the models and minimizing the makespan of sequencing the models through the workstations.

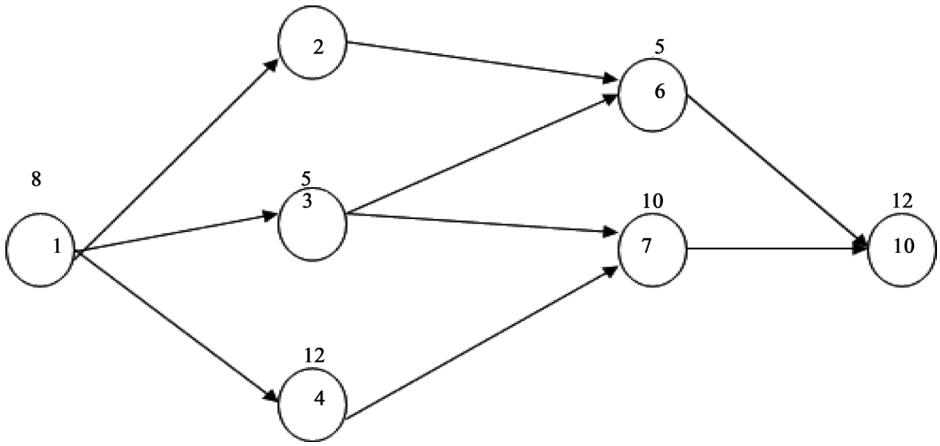

Figure 1. Precedence network of model 1.

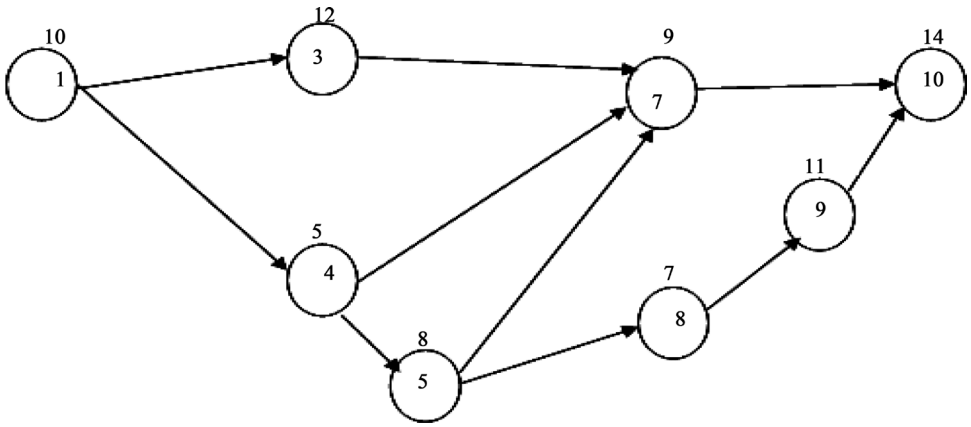

Figure 2. Precedence network of model 2.

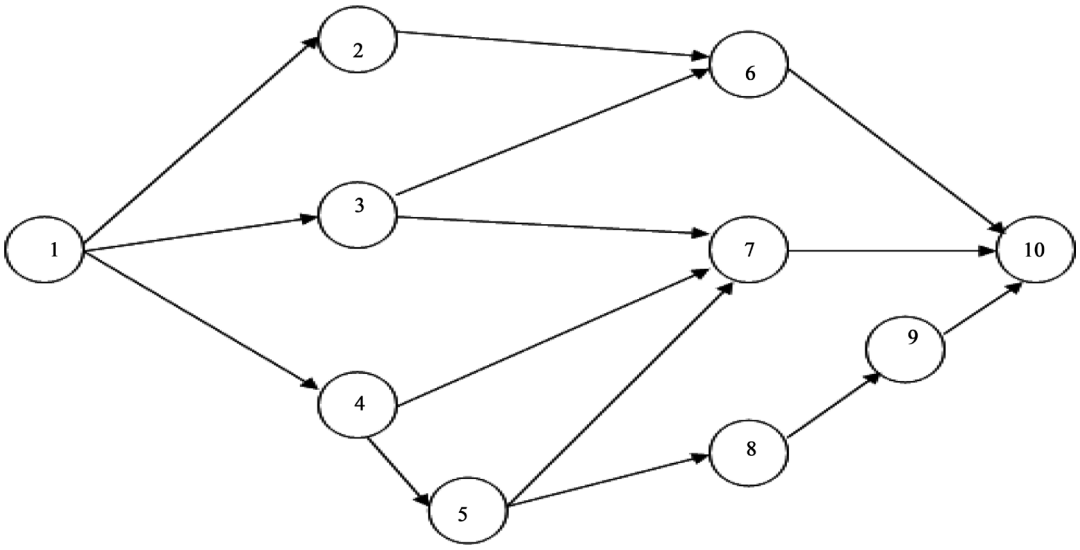

Figure 3. Combined model of model 1 and model 2.

2. Literature Review

This section presents the review of literature of the mixed-model assembly line balancing problem. The contri-

butions of the researchers are broadly classified into the following three categories:

1) Mixed model assembly line balancing problems;
2) Sequencing in mixed model assembly lines;
3) Mixed model assembly line balancing problems with model sequencing.

2.1. Mixed Model Assembly Line Balancing Problems

The literature of the mixed model assembly line balancing problem with the primary objective of balancing the line is presented under the following categories:

- Mathematical models;
- Branch and bound method;
- Heuristics;
- Genetic algorithms;
- Simulated annealing algorithms;
- ACO algorithm;
- Multiple colony bees algorithm.

Mathematical Models

Gokcen and Erel [3] considered the mixed-model assemble line balancing problem and developed a goal programming model with the objective of minimizing the number of workstations. In this model, they introduced assignment constraints, precedence constraints, cycle time constraint, zoning constraints and station constraints. Gokcen and Erel [4] developed a binary integer formulation for the mixed-model assembly line balancing problem in which the number of stations is minimized for given cycle times of the models. Kara *et al*. [5] considered the mixed-model assembly line balancing problem with model-mix having precedence conflict and duplicate common tasks. They developed a mathematical model for this problem. Sivasankaran and Shahabudeen [6] developed a hybrid mathematical model for the single model assembly line balancing problem. In the first phase, they presented a mathematical model to design the workstations such that the balancing efficiency is maximized for a given cycle time and in the second phase, another mathematical model is presented to minimize the cycle time for the number of workstations which is obtained in the first phase. Akpinar and Baykasoglu [7] developed a mixed-integer linear mathematical programming model to minimize the number of workstations of the mixed-model assembly line balancing problem.

Mathematical models cannot solve large size problems of mixed-model assembly line balancing problems, because the number of variables and the number of constraints that can be used in the software that solves the models are limited in the form of upper limit. So, there use is limited to very small size problems.

Branch and Bound Method

Bukchin and Rabinowitch [8] made an attempt to develop a branch and bound based solution approach for the mixed-model assembly line balancing problem with the objective of minimizing the number of workstations and task duplication costs. Li and Gao [9] considered balancing manual mixed-model assembly lines using overtime work in a demand variation environment with the objective of satisfying the demand in each possible scenario with minimum labour costs paid for both normal shifts and overtime work. A lower bound on the labour costs and a heuristic to find feasible solution are proposed. Then branch, bound and remember (BB&R) algorithm is proposed to find improved solutions. Through an experiment, they found that the use of overtime work and adjustable cycle time significantly reduces the labour costs.

Brach and bound method is a curtailed enumeration method. So, it can solve relatively large size problem when compared to mathematical models. For some problems, the total number of nodes evaluated may be closer to the maximum number of nodes of the branching tree. On such occasion, the time required to solve the mixed model assembly line balancing problem with the objective of minimizing the number of workstations by this method will be too large.

Heuristics

Matanachai and Yano [10] considered the mixed-model assembly line balancing problem with the objective of reducing work overload as well as maintaining reasonable workload balance among the workstations. They developed a heuristic based on filtered beam search for this problem. Jin and Wu [11] developed a heuristic called “variance algorithm” to balance the assembly line of the mixed-model assembly line balancing problem. Hop [12] developed a mixed zero-one programming model for the mixed-model assembly line balancing prob-

lem to minimize the number of workstations. Also, the author presented a fuzzy heuristic to minimize the number of workstations of the assembly line. Bock [13] developed distributed search methods for balancing the mixed-model assembly lines in an auto industry with the objective of minimizing the number of workstations. Al-Mamun *et al*. [14] developed a genetic algorithm based heuristic to design the mixed model assembly line with parallel workstations, zoning constraint and resource limitations such that the number of workstations of a plastic bag manufacturing company is minimized. Su, Wu and Yu [15] considered the mixed-model assembly line balancing problem with the objective of minimizing the number of workstations in the first stage and then minimizing the cycle time in the second stage. They developed a Petri net-based heuristic called P-invariant algorithm (PA) to minimize the number of workstations and a heuristic by combining the PA with a binary search algorithm (BSA) to minimize the cycle time.

Use of heuristics is considered to be a better choice for the mixed-model assembly line balancing problem, because it comes under combinatorial problem. But, the solution of a heuristic may not be very closer to the optimal solution of this problem. Hence, researchers directed their researches towards developing meta-heuristic such as genetic algorithm, ant colony optimization algorithm, etc. for the assembly line balancing problems, which are presented in the following sections.

Genetic Algorithms

Chutima and Iammi [16] considered the mixed-model assembly line balancing problem with the objectives of minimizing the number of workstations and total idle time. They developed a genetic algorithm for this problem. They reported that their algorithm outperforms COMSOAL, which is a primitive algorithm. So, the proposed algorithm needs to be compared with powerful meta-heuristic. Bai, Zhao and Zhu [17] developed a new hybrid algorithm by combing genetic algorithm with simulated annealing algorithm for the mixed-model assembly line balancing problem to find good solution in terms of minimizing the number of workstations and maximizing the balancing efficiency.

Sivasankaran and Shahabudeen [2] considered the mixed-model assembly line balancing problem with the objective of maximizing balancing efficiency for a common cycle time which is computed based on the cycle times of the models. They used the original task times of the individual models while forming the workstations, which is a realistic approach. They developed a genetic algorithm with cyclic crossover method [18] to solve this problem. Further, they found that their approach of using individual task times instead of average task times while forming workstations performs better.

The performance of a genetic algorithm may be affected by the crossover method used in it. Hence, future research may be directed to design genetic algorithms with different crossover methods and compare them for selecting the best among them.

ACO Algorithm

Yagmahan [19] considered the mixed-model assembly line balancing problem with the objective of minimizing the number of workstations. They developed a multi-objective ant colony optimization algorithm for this problem.

Multiple Colony Hybrid Bees Algorithm

Akpinar and Baykasoglu [20] considered the mixed-model assembly line balancing problem with zoning constraints, setup times, etc. They developed multiple bees algorithm for this problem with the objective of minimizing the number of workstations. Through an experiment, they found that this algorithm has superior performance.

From literature, it is seen that researchers concentrated in developing mathematical models, branch and bound algorithm, heuristics, genetic algorithms, ant colony optimization algorithm, etc for the mixed-model assembly line balancing problem with the objectives of minimizing the number of workstations and improving other measures.

2.2. Sequencing in Mixed Model Assumedly Lines

In addition to balancing the mixed model assembly lines, the study of effects of sequencing the models through the designed workstations forms a significant research. So, this section presents review of literature on sequencing models in mixed-model assembly lines for a given configuration of workstations under the following classification:

- Mathematical model;

- Heuristics;
- Tabu search algorithm;
- Genetic algorithms;
- Simulated annealing algorithms;
- Hybrid algorithm.

Mathematical Models

Dar-El and Cucuy [21] developed an integer programming model to find the optimal sequencing of models in the mixed-model assembly line such that the production demand is satisfied and overall line length is minimized for zero station idle time.

Heuristics

Ding and Cheng [22] developed a simple sequencing algorithm for mixed-model assembly lines in just-in-time production systems with the objective of maintaining near-constant production rate for every model in the line. They studied the repeating property of this algorithm. Leu, Huang and Rusell [23] developed beam search techniques for sequencing models in the mixed-model assembly line with the objectives of minimizing parts consumption variation and minimizing workload variation. They showed that their algorithm performs better than existing heuristics. Ding, Zhu and Sun [24] compared two weighted approaches for sequencing models in the mixed-model assembly line with multiple objectives., viz. product level objectives aiming to smooth finished goods production, work load balance in assembly lines and better sequence of various products by considering their assembly time impact to each assembly station. Rabbani *et al*. [25] developed a fuzzy goal programming approach to sequence the models in the mixed model assembly line with the objectives of minimizing total utility work, total production rate variation and total setup cost. It uses desirability of decision makers (DM) and tolerances considered on goal values. Through a set of instances, its computational performance is reported. Rahimi-Vahed *et al*. [26] developed a multi-objective scatter search algorithm (MOSS) for a mixed model assembly line sequencing problem with the objectives of minimizing total utility cost, total production rate variation and total setup cost. The performance of this algorithm has been compared with prominent algorithms in literature and showed that it performs well. Erel, Gocgun and Sabuncuoglu [27] developed six beam search algorithms to sequence the models in the mixed-model assembly line with the objectives of minimizing part usage variation and maximizing load-levelling.

Bautista and Cano [28] considered the mixed model assembly line balancing problem with sequencing of models in the mixed model assembly line with the objective of minimizing work overload. They developed a bounded dynamic programming algorithm for this problem. Gujjula, Werk and Giinther [29] developed a heuristic based on Vogel's approximation method to sequence models in the mixed-model assembly line through workstations with the objective of minimizing utility work. Lin and Chu [30] developed a Lagrangian relaxation approach to the mixed-product assembly line sequencing problem with objectives of minimizing the overall cost and meeting customer demand, and applied it to a door-lock comply in Taiwan.

Tabu Search Algorithm

Scholl, Klein and Domschke [31] considered the problem of sequencing models in the mixed-model assembly line with fixed rate launching and closed stations. They developed an informed tabu search procedure with a pattern based vocabulary building strategy.

Genetic Algorithms

Kim, Hyun and Kim [32] considered the sequencing of models in the mixed model assembly line and developed a genetic algorithm with the issues of worker schedules, product mix and launch interval. Leu, Matheson and Rees [33] developed a genetic algorithm for sequencing models through the workstations of the mixed-model assembly line. Hyun, Kim and Kim [34] developed a genetic algorithm called Pareto stratum-niche cubicle, for the mixed-model assembly line balancing problem with sequencing of models. The objectives considered by them are minimizing total utility work, keeping a constant rate of part usage and minimizing total setup cost. They compared the performance of this algorithm with existing genetic algorithms and found that the proposed algorithm performs better. Ponnambalam, Aravindan and Subba Rao [35] developed genetic algorithms for sequencing models in the mixed model assembly line with the objectives of minimizing total utility work, variability in parts usage and total setup cost. Su and Lu [36] developed a genetic algorithm for the mixed-model assembly line balancing problem to find the sequence of the models, for minimizing the cycle time.

Simulated Annealing Algorithms

Xiaobo and Ohno [37] developed a branch bound algorithm and a simulated annealing algorithm for se-

quencing mixed models on an assembly line in a JIT production system with the objective of minimizing line stoppages. The branch and bound algorithm is used to solve small problems optimally and the simulated annealing algorithm is used to solve large size problems near optimally. Fattahi and Salchi [38] developed a hybrid meta-heuristic based on simulated annealing algorithm to sequence the models through the workstations of the mixed-model assembly line with variable launching interval. Amlashi and Zandieh [39] considered the sequencing of models in the mixed-model assembly line with the objective of minimizing line stoppage cost. They developed a cloud theory-based simulated annealing algorithm (CSA) and compared its performance with that of simulated annealing algorithm (SA). It is found that the CSA algorithm performs better than SA algorithm.

Hybrid Algorithm

Rahimi-Vahed and Mirzaei [40] considered the sequencing of models in a given mixed model assembly line configuration. They developed a hybrid multi-objective shuffled frog-leaping algorithm for this problem with the objectives of minimizing total utility work, total production rate variation and total set-up cost. They compared the results of this algorithm with existing algorithms and found that it is better.

From the literature on sequencing the models through workstations of the mixed-model assembly line with varied objectives, it is clear that researches have been carried in developing mathematical models, tabu search, genetic algorithms, simulated annealing algorithms and hybrid algorithm. Since, sequencing falls under combinatorial problem researchers can continue to develop much efficient meta-heuristics to improve their performance measures.

2.3. Mixed Model Assembly Line Balancing with Sequencing of Models

Simultaneous consideration of the design of assembly lines and the sequencing of models through the designed workstations brings manifold benefits in terms of realistic improvement on several performance measures. This section presents a review of literature on the mixed-model assembly line balancing problem with model sequencing under the following classifications:

- Heuristics;
- Simulated annealing algorithms.

Heuristics

Thomopoulos [41] described a procedure of adapting the single-model assembly line balancing technique to the mixed-model assembly line balancing. Further a procedure to sequence the models in the assembly line is also proposed by the author. Merengo, Nava and Pozzetti [42] considered the mixed-model assembly line balancing problem with the objectives of minimizing the rate of incomplete jobs and work-in-process. They developed a balancing methodology to reduce the number of workstations and a sequence methodology to provide uniform parts usage. Kim and Kim [43] developed a co-evolutionary algorithm for balancing and sequencing in the mixed model assembly line with the objective of minimizing utility work. Lovgren and Racer [44] developed algorithms for sequencing of models in the mixed-model assembly line with due date restrictions. The objectives of this research are maximizing the balancing of the assembly line and minimizing the sum of the lateness values of the models. They studied a set of heuristics and analyzed their performance in terms of lateness measure and smoothing component utilization. The Border Swap heuristic proves to be the best among all the heuristics in the set. Hwang and Katayama [45] developed an integrated procedure to balance the mixed-model assembly line and sequence the models through the workstations with the objectives of minimizing the number of workstations as a static criterion and minimizing workload variance as dynamic criterion. The approach developed is a hierarchical design procedure based on an amelioration procedure. They found that this approach gives better results.

Simulated Annealing Algorithm

Ozcan *et al*. [46] developed a simulated annealing algorithm for balancing and sequencing of parallel mixed assembly lines in which two or more assembly lines with the objectives of minimizing the number of workstations and attaining equalization of workloads among workstations. Mosadegh, Zandieh and Ghomi [47] researched the simultaneous solving of balancing and sequencing of the assembly line balancing problem with the objective of minimizing total utility work. They developed a mixed-integer programming model and then developed a simulated annealing algorithm. The results of this algorithm were compared with those of co-evolutionary genetic algorithm and showed that the simulated annealing algorithm gives better result. The performance of the simulated annealing algorithm can be improved by setting its parameters, viz. temperature, reduction factor, etc. at optimal levels and designing efficient seed generation algorithms. *From literature on simul-*

taneous consideration of balancing and sequencing of the models in the mixed-model assembly line balancing problem, it is clear that a limited work is carried out for this problem

From the literature, it is clear that considerable number of researchers contributed to balance the mixed-model assembly line balancing problem as well as sequencing of models in the mixed model assembly line. A very few research is carried out on simultaneous consideration of balancing and sequencing of models in the mixed-model assembly line.

Hence, in this paper, the simultaneous consideration of the mixed-model assembly line balancing with model sequencing through the workstations with multi-objective is considered. The components of the multi-objective function are maximizing the average balancing efficiency of the models and minimizing makespan of sequencing the models through workstations. Among different algorithms, genetic algorithm plays a significant role in improving the performance measures of this problem, because its variance can be studied through different crossover methods as well as methods of forming workstations. So, in this paper, four different genetic algorithms are presented and compared using a complete factorial experiment with four factors, viz. problem size, problem structure, algorithm and cycle time.

3. Problem Statement

The problem considered in this paper is the mixed-model assembly line balancing problem with M models. Each model has a specific set of tasks to be performed. The tasks of the combined model form a set of tasks which is the union of the tasks of the individual models. Let, the number of tasks in model 1 and model 2 and the combined model of the model 1 and model 2 be n_1, n_2 and n, respectively. Then the value of n will be less than $n_1 + n_2$. In combined model, if a task is present in a model, then it has a deterministic task time; otherwise, it is assumed as zero.

The objective of this research is to group the tasks of the combined model into a minimum number of workstations without violating precedence constraints for a common cycle time, which in turn maximizes the average balancing efficiency of the models, and sequence the models in the assembly line with minimum makespan.

In this paper, three different genetic algorithms are designed for the mixed-model assembly line balancing problem, in which the original times of the tasks are used while concurrently designing the workstations of the models, such that the average balancing efficiency of the models is maximized and the makespan of sequencing the models is minimized. Then, these three algorithms and the genetic algorithm developed by Sivasankaran and Shahabudeen (2013b) adopted to this problem are compared using a randomly generated data set as per a complete factorial experiment with four factors, viz. Problem Size, Problem Structure, Algorithm and Cycle Time, in terms of the multi-objective stated above.

The above multi-objective is represented as given below.

$$\begin{aligned} F &= \text{Maximize average balancing efficiency of the models and Minimize makespan of sequencing models} \\ &= \text{Maximize}\left[\frac{\sum_{i=1}^{m}\eta_i}{m}\right] + \text{Minimize}\left[M\right] \end{aligned} \tag{4}$$

where, η_i is the efficiency of the model i.

m is the number of models.

M is the makespan of sequencing models in the assembly line.

In this function F, the average balancing efficiency is in percentage and the makespan is in time unit. To covert them in common unit, the makespan is converted into excess percentage of makespan (EPMS) using the following formula, if genetic algorithm is used to solve the problem.

$$\begin{aligned} \text{EPMS} &= \left[\frac{\text{Actual makespan of a chromosome}}{\text{Minimum of the makespans of the chromosomes of the population}}\right]\times 100 - 100 \\ &= \left[\frac{M}{\text{Min}\left\{MS_j\right\}, j = 1,2,3,\cdots,K}\right]\times 100 - 100 \end{aligned} \tag{5}$$

where,

M is the actual makespan of sequencing models in the assembly line.

K is the number of chromosomes in the population.

MS_j is the makespan of the j^{th} chromosome in the population.

The above conversion matches the unit of the excess percentage of makespan with that of balancing efficiency in terms of percentage.

The multi-objective after matching the units of its components is presented below.

$$\text{Maximize } F = \text{Average Balancing Efficiency of Models} - \text{EPMS}$$

$$= \left[\frac{\sum_{i=1}^{m}\eta_i}{m}\right] - \left\{\left[M \Big/ \text{Min}\left(MS_j\right), j = 1,2,3,\cdots,K\right] \times 100 - 100\right\}$$

Further, if the actual makespan of a chromosome is equal to the minimum of the makespan values of the chromosomes of the population, then the excess percentage of makespan (EPMS) will be equal to zero, which coincides with the objective of the second component of the objective function F. This forms a rationale of minimizing the makespan of the model chromosome.

$$\text{Maximize } F = \left[\frac{\sum_{i=1}^{m}\eta_i}{m}\right] - \left\{\left[M \Big/ \text{Min}\left(MS_j\right), j = 1,2,3,\cdots,K\right] \times 100 - 100\right\} \quad (6)$$

4. Design of Genetic Algorithms

In this research, four different genetic algorithms as given below are presented to solve the mixed-model assembly line balancing problem with model sequencing to maximize the average balancing efficiency of the models and minimize the makespan of sequencing the models.

- Genetic algorithm with cyclic crossover method for task chromosomes [2] and reflection crossover method for model chromosomes, $ALGS_1$.
- Genetic algorithm with forward crossover method for task chromosomes and reflection crossover method for model chromosomes, $ALGS_2$ (Proposed).
- Genetic algorithm with reverse crossover method for task chromosomes and reflection crossover method for model chromosomes, $ALGS_3$ (Proposed).
- Genetic algorithm with cyclic crossover method for task chromosomes and reflection crossover method for model chromosomes, and modified workstation formation , $ALGS_4$ (Proposed).

The genetic algorithm begins its working starting from an initial population of say K chromosomes. Each chromosome represents a possible solution of the problem of interest. It evaluates each and every chromosome and obtains the value of its fitness function. Then it sorts them in descending order of the values of the fitness function in the case of maximization problem. Then a subpopulation is selected for crossover operation from the top of the sorted list and then each offspring that is obtained after performing crossover operation and mutation, is evaluated and placed in the respective chromosome of the population. The whole exercise is repeated for a specified number of iterations called generation and finally the best chromosome is selected for implementation [48] [49].

In this paper, two types of population, viz. task population and model population are used. The tasks of the combined model are randomly assigned to different positions of a chromosome to form a task chromosome. Similarly, the different model numbers are randomly assigned to different gene positions of a chromosome to form a model chromosome. The sequence of gene entries in a model chromosome represents the model sequencing (*scheduling*), *which means that sequencing of the models through the workstations of the assembly line. The proportion of model entries in the model chromosome is in proportion to the production volumes of the models.*

4.1. Crossover Methods for Task Chromosomes

The solution of the assembly line balancing problem using genetic algorithm may be affected by crossover methods. Hence, in this paper, the following crossover methods for task chromosomes are incorporated in the genetic algorithm to derive three different GA based algorithms.

- Cyclic crossover method;

- Forward crossover method;
- Reverse crossover method.

A chromosome consists of a sequence of genes. For the mixed model assembly line balancing problem, a combined model is derived using the tasks of all the models. The chromosome consists of tasks of the combined model as shown in **Table 1** and **Table 2** for the combined model shown in **Figure 3**.

4.1.1. Cyclic Crossover Method for Task Chromosomes

In this paper, the cyclic crossover method implemented in the genetic algorithm developed by Senthilkumar and Shahabudeen [25] for open shop scheduling problem to create two offspring is adapted to create offspring of the task chromosomes of the mixed model assembly line balancing problem. Using task chromosome 1 and the task chromosome 2, which are shown in **Table 1** and **Table 2**, respectively, the cyclic crossover method is explained below.

Step 1: Generate two crossover points randomly. Let them be 4 and 7.

Step 2: Identify the values of the genes in the task chromosome 1 between the two crossover points inclusive of the crossover points. These values are 10, 4, 9 and 8.

Step 3: Form the offspring 2 whose gene positions are filled with the genes of the task chromosome 2 as shown in **Table 3**.

Step 4: In the offspring 2, wherever the gene value is equal to each of the gene values identified between the two crossover points of the task chromosome 1, set it to 0 as shown in **Table 4**.

Step 5: Read the non-zero values of the genes from left to right of the offspring 2 and write them in cyclic order starting from the next position of the second crossover point, except the points between the two crossover points as shown in **Table 5**.

Step 6: Write the values of the genes between the two crossover points of the task chromosome 1 in the empty positions of the offspring 2 starting from the first vacant position from left as shown in **Table 6** to get the final offspring 2.

Table 1. Task chromosome 1.

Position *i*	1	2	3	4	5	6	7	8	9	10
Task *i*	6	1	2	10	4	9	8	3	7	5

Table 2. Task chromosome 2.

Position *i*	1	2	3	4	5	6	7	8	9	10
Task *i*	4	2	5	7	1	10	9	6	3	8

Table 3. Offspring 2 with initial genes.

Position *i*	1	2	3	4	5	6	7	8	9	10
Task *i* of Offspring 2	4	2	5	7	1	10	9	6	3	8

Table 4. Modified partial offspring 2.

Position *i*	1	2	3	4	5	6	7	8	9	10
Task *i* of Offspring 2	0	2	5	7	1	0	0	6	3	0

Table 5. Rearranged partial offspring 2.

Position *i*	1	2	3	4	5	6	7	8	9	10
Task *i* of Offspring 2	1	6	3					2	5	7

Table 6. Final offspring 2 using cyclic crossover method.

Position *i*	1	2	3	4	5	6	7	8	9	10
Task *i* of Offspring 2	1	6	3	10	4	9	8	2	5	7

Similarly, the offspring 1 to replace the chromosome 1 is created using the cyclic crossover method and it is as shown in **Table 7**.

4.1.2. Forward Crossover Method for Task Chromosomes

The steps of the forward crossover method for task chromosomes proposed in this paper are same as in the cyclic crossover method except Step 5.

Step 5: Read the non-zero values of the genes from left to right of the offspring 2 in **Table 4** and write them in forward direction starting from the first available gene position from left to right, except the points between the two crossover points. The corresponding offspring 2 using forward crossover method is shown in **Table 8**.

Similarly, the offspring 1 to replace the task chromosome 1 is created using the forward crossover method and it is as shown in **Table 9**.

4.1.3. Reverse Crossover Method

The steps of the reverse crossover method proposed in this paper are same as in the cyclic crossover method except Step 5.

Step 5: Read the non-zero values of the genes from left to right of the offspring 2 in **Table 4** and write them in reverse direction starting from the last available gene position from right to left, except the points between the two crossover points. The final offspring 2 as per reverse crossover method is shown in **Table 10**.

Similarly, the offspring 1 to replace the chromosome 1 is created using the reverse crossover method and it is shown in **Table 11**.

4.2. Reflection Crossover Method for Model Chromosomes

As stated earlier, the fitness function for each combination of task chromosome and model chromosome consists of two components, viz. average balancing efficiency of the models and excess percentage of makespan. For each chromosome in the initial population of the model chromosomes, the makespan of sequencing the models as per sequence of genes in that model chromosome is to be obtained. This is done by computing completion times of the models represented by the genes of that model chromosome in different workstations and finally treating the completion time of the model represented by the last gene in the model chromosome in the last workstation as the makespan.

Table 7. Offspring 1 using cyclic crossover method.

Position *i*	1	2	3	4	5	6	7	8	9	10
Task *i* of Offspring 1	8	3	5	7	1	10	9	6	2	4

Table 8. Final offspring 2 using forward crossover method.

Position *i*	1	2	3	4	5	6	7	8	9	10
Task *i* of Offspring 2	2	5	7	10	4	9	8	1	6	3

Table 9. Offspring 1 using forward crossover method.

Position *i*	1	2	3	4	5	6	7	8	9	10
Task *i* of Offspring 1	6	2	4	7	1	10	9	8	3	5

Table 10. Final offspring 2 using reverse crossover method.

Position *i*	1	2	3	4	5	6	7	8	9	10
Task *i* of Offspring 2	3	6	1	10	4	9	8	7	5	2

Table 11. Offspring 1 using reverse crossover method.

Position *i*	1	2	3	4	5	6	7	8	9	10
Task *i* of Offspring 1	5	3	8	7	1	10	9	4	2	6

In this paper, reflection crossover method is proposed to perform crossover operation between a pair of model chromosomes in the subpopulation of model chromosomes. It is followed by mutation on each offspring for a given probability. Then, the makespan of sequencing the models through the workstations is computed. This makespan is used to compute the excess percentage of makespan (EPMS), which forms the second part of the objective function.

Guidelines to Form Model Chromosome

Let the ratio of production volume of two models which are to be assembled in the same line be V1:V2, where V1 is the volume of production of model 1 and V2 be the volume of production of model 2.

The ratio V1:V2 should the least integral multiples of the production volumes of the model 1 and model 2.

Let V1:V2 be 4:5.

Assume 9 genes for the model chromosome. Now, randomly assign model number 1 and model number 2 to different gene positions of the model chromosome in proportion to their production volumes 4:5 to create model chromosome 1 and model chromosome 2 as shown in **Table 12** and **Table 13**, respectively.

Steps of Reflection Crossover Method

Step 1: Input the model chromosome 1 (MC1) and the model chromosome 2 (MC2) as shown in **Table 14** and **Table 15**, respectively.

Step 2: Initialize offspring gene position (P) to 1.

Step 3: Randomly select a gene position in the model chromosome 1(MC1) and let it be X and the corresponding gene content is $MC1_X$.

Also, set starting index (STARTIND) of the model chromosome 1 to X.

Step 4: Store $MC1_X$ as the P^{th} gene of the model offspring O [$O_P = MC1_X$] and remove the gene entry at the location X of the model chromosome 2.

Step 5: In the model chromosome 2 [MC2], find the gene positions from among the available genes, for which their gene values are equal to $MC1_X$.

Step 6: Randomly select one of the genes of the model chromosome 2 identified in Step 5 and identify its location. Let that location be Z.

Step 7: If Z = STARTIND, then go to Step 12; otherwise go to Step 8.

Step 8: Remove the gene entry at the location Z of the model chromosome 2.

Step 9: Increment the gene position of the model offspring by 1 [P = P + 1].

Step 10: Store the Z^{th} gene of the model chromosome 1 as the P^{th} gene of the model offspring and store Zin X, $O_P = MC1_Z$, X = Z.

Step 11: Go to Step 5.

Step 12: Copy the remaining (unselected) genes of the model chromosomes 1 (MC1) as the genes of the model offspring starting from its $P + 1^{th}$ gene position to form the complete model offspring 1.

Table 12. Model chromosome 1 [MC1].

Gene Position I	1	2	3	4	5	6	7	8	9
Gene $MC1_I$	1	2	1	2	2	2	1	2	1

Table 13. Model chromosome 2 [MC2].

Gene Position I	1	2	3	4	5	6	7	8	9
Gene $MC2_I$	2	2	1	1	2	2	2	1	1

Table 14. Model chromosome 1 [MC1].

Gene Position I	1	2	3	4	5	6	7	8	9
Gene $MC1_I$	1	2	1	2	2	2	1	2	1

Table 15. Model chromosome 2 [MC2].

Gene Position I	1	2	3	4	5	6	7	8	9
Gene $MC2_I$	2	2	1	1	2	2	2	1	1

Step 13: Swap the original model chromosomes and perform the above steps to obtain the model offspring 2.
Step 14: Stop.

The implementation of the steps of the reflection crossover method is shown in **Figure 4**. The offspring 1 by treating MC1 as the first chromosome and MC2 as the second chromosome is as shown at the bottom of the **Figure 4**. Similarly, the offspring 2 can be obtained by treating MC2 as the first chromosome and MC1 as the second chromosome.

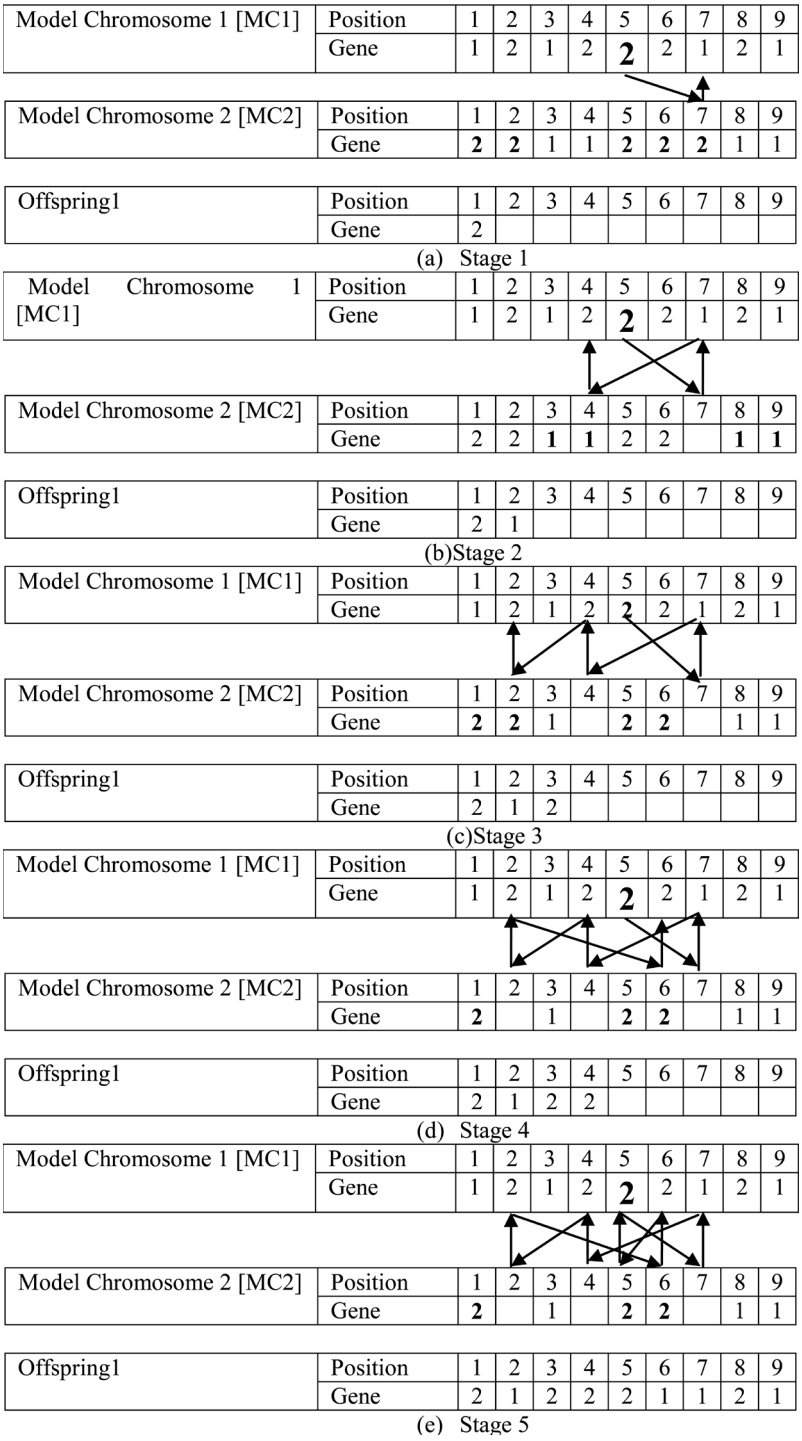

Figure 4. Stages of reflection crossover method for model chromosome.

4.3. Construction of Ordered Vector of Task Chromosome

The serial assignment of tasks in a task chromosome to workstations from left to right without violating precedence constraints of the combined model gives a feasible solution for the assembly line balancing problem. But, the randomly placed tasks of the combined model in a task chromosome may not form workstations without violating immediate precedence constraints of the combined model. Hence, the genes of each task chromosome are to be ordered (rearranged) such that the serial assignment of the genes from the ordered vector to workstations does not violate the precedence constraints as shown in **Figure 3**. The steps of constructing the ordered vector for a given task chromosome/task offspring as presented by Sivasankaran and Shahabudeen [2] are given below.

Step 1: Input the chromosome.

- Let the task chromosome I, which is already shown in the **Table 1** be as shown in **Table 16** along with the values for the STATUS row as zero. If the value of the STATUS for a gene (task) is zero, then it signifies that it is not assigned to any workstation; otherwise, it signifies that it is assigned to some workstation.
- Form the immediate predecessor(s) matrix of the tasks shown in **Figure 3** as shown in **Table 17**. The maximum of the number of immediate predecessors of the tasks in the combined model in the **Figure 3** is 3.
- Number of tasks (genes of the task chromosome), N.
- Initialize the gene position of the ordered vector, K = 1.
- Set the task chromosome number.

Step 2: Set the gene position of the task chromosome, J = 1

Step 3: If the STATUS of the gene J of the task chromosome I is equal to 1, then go to Step 9; else, go to Step 4.

Step 4: If all the values in the row of **Table 17** corresponding to the task at the gene position J of the task chromosome I are zero, then go to Step 5; otherwise, go to Step 9.

Step 5: Assign the task at the gene position J of the task chromosome I to the gene position K of the ordered vector.

$$OV_K = C_{IJ}$$

Step 6: Set the status of the gene position *J* of the task chromosome *I* to 1 in **Table 16** ($STATUS_J = 1$).

Step 7: Change the value of C_{IJ} in immediate predecessor matrix to zero, wherever it is equal to the task at the gene position J.

Step 8: Increment the gene position (K) of the ordered vector by 1 and go to Step 2.

Step 9: Increment the gene position (J) of the task chromosome 1 by 1.

Table 16. Task chromosome I.

Gene Position J	1	2	3	4	5	6	7	8	9	10
STATUS	0	0	0	0	0	0	0	0	0	0
Gene J of Chromosome 1	6	1	2	10	4	9	8	3	7	5

Table 17. Immediate predecessors matrix [IP_{pq}].

Task (p)	Immediate Predecessor(s) (q)		
1	0	0	0
2	1	0	0
3	1	0	0
4	1	0	0
5	4	0	0
6	2	3	0
7	3	4	5
8	5	0	0
9	8	0	0
10	6	7	9

Step 10: If $J \leq N$, then go to Step 3; otherwise go to Step 11.

Step 11: Stop.

The application of the above steps to the task chromosome 1 which is shown in **Table 1** gives an ordered vector as shown in **Table 18**.

4.4. Determination of Average Balancing Efficiency of Models (Part of Fitness Function) of Task Chromosome

This section illustrates the method of finding the average balancing efficiency of the models of a task chromosome. This forms the first part of the multi-objective fitness function.

The cycle time of the model 1 as well as that of the model 2 is assumed as 20 minutes. Hence, the cycle time of the combined model is also 20 minutes, which is the average of the cycle times of both the models. The fitness function of the task chromosome 1, namely balancing efficiency is obtained by assigning the tasks serially from left to right from its ordered vector 1-2-4-3-6-5-8-9-7-10 into workstations for the given cycle time of 20 minutes of the combined model as shown in **Table 19**.

While assigning a task into a workstation, that task pertaining to all the models should be assigned to the same workstation. If a task is available in only one model then that can be independently assigned to the current workstation.

Note: *The number of workstations of model* 1 *and that of model* 2 *may not be same, because some workstations of the combined model may be assigned with the tasks of either model* 1 *or model* 2 *only.*

The formulas for the balancing efficiencies of model 1, model 2 and average balancing efficiency of the models are as given below.

$$\text{Balancing efficiency of model 1}\left(\eta_1\right) = \left[\frac{\text{Sum of the task times of model 1}}{\text{No.of workstations of Model 1} \times \text{Common Cycle Time}}\right] \times 100$$

Table 18. Ordered vector of task chromosome 1shown in **Table 1**.

Gene Position J	1	2	3	4	5	6	7	8	9	10
Gene J of Chromosome 1	1	2	4	3	6	5	8	9	7	10

Table 19. Solution and fitness function value for ordered vector 1-2-4-3-6-5-8-9-7-10.

Workstation	Assigned Tasks		Unassigned Time	
	Model 1	Model 2	Model 1	Model 2
I	1	1	12	10
	2	-	1	10
	Idle time		**1**	**10**
II	4	4	8	15
	3	3	3	3
	Idle time		**3**	**3**
III	6	-	15	20
	-	5	15	12
	-	8	15	5
	Idle time		**15**	**5**
IV	-	9	20	9
	7	7	10	0
	Idle time		**10**	**0**
V	10	10	8	6
	Idle time		**8**	**6**
	Balancing efficiency		63%	76%
	Combined efficiency			69.5%

$$\text{Balancing efficiency of model 2}\left(\eta_2\right) = \left[\frac{\text{Sum of the task times of model 2}}{\text{No.of workstations of Model 2} \times \text{Common Cycle Time}}\right] \times 100$$

$$\text{Average balancing efficiency of the models} = \left(\eta_1 + \eta_2\right)/2$$

4.5. Evaluation of Excess Percentage of Makespan of Sequencing Models

The second part of the multi-objective fitness function represents the excess percentage of makespan and it is subtracted from the first part, because the second part is a minimization type and the fitness function is of maximization type.

The steps of computing the excess percentage of makespan of sequencing the models through the workstations are presented below.

Step1: Input the following:

- Number of workstations (NS);
- Tasks of model *i*, $i = 1, 2, 3, \cdots, m$ assigned to each workstation;
- Task times of model *i*, $i = 1, 2, 3, \cdots, m$;
- Model sequence, which is a model chromosome/ model offspring and the number of genes in it(R).

Step 2: Set model number, I = 1.

Step 3: Set workstation number, J = 1.

Step 4: Find the total time of tasks of model *I* assigned to workstation J [TTT_{IJ}].

Step 5: Increment workstation number by 1 (J = J + 1).

Step 6: If $J \le NS$, then go to Step 4; otherwise go to Step 7.

Step 7: Set gene position of model chromosome to 1 [GP = 1]

Step 8: Set completion time of model that is present in gene position GP at workstation J to 0, for $J = 1, 2, 3, \cdots, NS$.

$$CT(J) = 0,\ J = 1, 2, 3, \cdots, NS$$

Step 9: Let the model at the gene position GP be X.

Step 10: Set workstation number, J = 1.

Step 11: Compute the completion time of the model X at the workstation J by adding TTT_{XJ} to CT_J.

$$CT_J = CT_J + TTT_{XJ}$$

Step 12: Increment the workstation number by 1 (J = J + 1).

Step 13: If $J \le WS$, then go to Step 10; otherwise go to Step 14.

Step 14: Set the gene position of model chromosome to 2 [GP = 2].

Step 15: Let the model at the gene position GP be X.

Step 16: Set workstation number, J = 1.

Step 17: If $J \ne 1$, then go to Step 23.

Step 18: Compute the completion time of the model X at the workstation J by adding TTT_{XJ} to CT_J.

$$CT_J = CT_J + TTT_{XJ}$$

Step 18: Go to Step 23.

Step 19: IF $CT_J \ge CT_{J-1}$, then go to Step 22.

Step 20: $CT_J = CT_{J-1} + TTT_{XJ}$.

Step 21: Go to Step 23.

Step 22: $CT_J = CT_J + TTT_{XJ}$.

Step 23: Increment J by 1 (J = J + 1).

Step 24: If $J \le WS$, then go to Step 19.

Step 25: Increment the gene position of the model chromosome by 1 [GP = GP + 1].

Step 26: If $GP \le R$, then go to Step 14; otherwise go to Step 27.

Step 27: Makespan of the model chromosome is given by the latest completion time of the last workstation.

$$\text{Makespan} = CT_{WS}$$

Step 28: Repeat the above steps to obtain the makespan values of all the model chromosomes while evaluating the initial population or while evaluating the offspring of the subpopulation.

Step 29: Find the minimum makespan of the makespan values of the model chromosomes of the current population and find the excess percentage of makespan (EPMS) of each model chromosome using the following formula.

$$\text{EPMS} = \left[\frac{\text{Actual makespan of the chromosome}}{\text{Minimum of the makespans of the chromosomes of the population}} \right] \times 100 - 100$$

Step 30: Stop.

4.6. Genetic Algorithm with Cyclic Crossover Method for Task Chromosome and Reflection Crossover Method for Model Chromosome ($ALGS_1$)

The steps of the genetic algorithm with the cyclic crossover method for task chromosome and the reflection crossover method for model chromosome to group the tasks of the mixed-model assembly line balancing problem into a minimum number of workstations which amounts to maximizing average balancing efficiency of the models with sequencing of models to minimize makespan are presented below. For each combination of task chromosome (offspring) and model chromosome (offspring), the balancing efficiencies of the individual models and the average balancing efficiency of the models are computed. Then the excess percentage of makespan (EPMS) is computed. Next, it is subtracted from the average balancing efficiency of the models to obtain the value of the maximization multi-objective function (F) as given in Equation (6). So, the objective of this algorithm is to group the tasks of the models in an integrated manner based on the combined network into different workstations such that the function F (multi-objective function) is maximized.

As explained in the previous subsection, for each of the task offspring that will be obtained using crossover operation, an ordered vector will be constructed, which gives a sequence of the tasks in the immediate precedence network such that their sequential assignment to workstations satisfies the immediate precedence relationships among the tasks in the network.

The steps $ALGS_1$ are presented below.

Step 1: Input the following:

- Number of models, M;
- Production volume per shift of each model;
- Ratio of production volumes of the models;
- Precedence networks of the models;
- Combined precedence network;
- Task times of the models;
- Common cycle time, C;
- Size of the population of the task chromosomes as well as that of the model chromosomes, N (50);
- Number of generations to be carried out, Q (20);
- Mutation probability, α(0.3).

Step 2: Initialize the generation count (r) to 1.

Step 3: Generate the desired number (N) of task chromosomes (Population) by randomly placing the tasks of the combined model as the genes of each task chromosome.

Step 4: Find the ordered vector of each task chromosome in the population.

Step 5: Design the workstations of each task chromosome for the common cycle time using the task times of the individual models and the immediate predecessor(s) matrix of the combined model by serially assigning the tasks from the ordered vector and obtain the following:

- The balancing efficiencies of the models ($\eta_1, \eta_2, \eta_3, \cdots, \eta_M$);
- The average balancing efficiency of the models

$$\eta_{Ave} = \left[\frac{\sum_{i=1}^{M} \eta_i}{M} \right]$$

Step 6: Generate N model chromosomes by randomly placing the model numbers in each of them to represent N model sequences.

- The number of genes of the models in each model chromosome is based on the proportion of the production volumes of the models.
- If the number of genes in a model chromosome is 9 and if the ratio of production volumes of model 1 and model 2 is 4:5, then a sample model chromosome is given as 2-1-2-2-1-1-2-2-1.

Step 7: Compute the excess percentage of makespan (EPMS) using the following steps.

- For each model chromosome in the population of model chromosomes, find the makespan of sequencing the models through the workstations of the corresponding task chromosome in the population of task chromosomes.
- Find the excess percentage of makespan (EPMS) with reference to the least makespan of the current population of model chromosomes using the following formula.

$$\text{EPMS} = \left\{ \frac{L}{\left[\text{Min}\left(MS_i \right), i = 1, 2, 3, \cdots, N \right]} \right\} \times 100 - 100$$

Step 8: Find the multi-objective fitness function (F) value of each combination of task chromosome and model chromosome of the population using the following formula.

$$F = \eta_{Ave} - \text{EPMS}$$

Step 9: Simultaneously sort the task chromosomes and model chromosomes in descending order of their multi-objective fitness function (F) values.

Step 10: Select a subpopulation (30% of population rounded to even number) from the top of the sorted list of the task chromosomes and a subpopulation of the same size from the top of the sorted population of the model chromosomes.

Step 11: Perform crossover operation using cyclic crossover operation as explained in the Section 4.1.1 for different pairs of task chromosomes in the subpopulation of task chromosomes and obtain their offspring.

Step 12: For each task offspring, perform mutation for a given mutation probability.

Step 13: For each of the task offspring after mutation, find its ordered vector.

Step 14: Design the workstations of each task offspring for the common cycle time using the task times of the individual models and the immediate predecessor(s) matrix of the combined model by serially assigning the tasks from the ordered vector and obtain the following:

- The balancing efficiencies of the models ($\eta_1, \eta_2, \eta_3, \cdots, \eta_M$);
- The average balancing efficiency of the models.

$$\eta_{Ave} = \left[\frac{\sum_{i=1}^{M} \eta_i}{M} \right]$$

- Replace the corresponding task chromosome with this task offspring along with its average balancing efficiency/

Step 15: Perform reflection crossover operation as explained in the Section 4.2 for different pairs of model chromosomes in the subpopulation of the model chromosomes and obtain their model offspring.

Step 16: For each model offspring, perform mutation for a given mutation probability and do the following:

- Obtain its makespan to schedule the models through the workstations designed as per the corresponding task offspring;
- Replace the corresponding model chromosome in the current population with this model offspring along with its makespan.

Step 17: For each model chromosome in the current population, find the excess percentage of makespan (EPMS) with reference to the least makespan of the model chromosome of the current population using the following formula.

$$\text{EPMS} = \left\{ \frac{L}{\left[\text{Min}\left(MS_i \right), i = 1, 2, 3, \cdots, N \right]} \right\} \times 100 - 100$$

Step 18: Find the multi-objective fitness function (F) value of each combination of task chromosome and the corresponding model chromosome of the current population using the following formula.

$$F = \eta_{Ave} - \text{EPMS}$$

Step19: Simultaneously sort the task chromosomes and model chromosomes of the population in descending order of their fitness function (F) values.

Step 20: Increment the generation count r by 1 (r = r + 1).

Step 21: If the generation count (r) is less than or equal to Q, then go to Step 10; otherwise, go to Step 22.

Step 22: Rework the details of the workstations and sequencing of the models with respect to the topmost task chromosome and model chromosome in the current sorted population and print the results along with the corresponding fitness function value.

Step 23: Stop.

4.7. Genetic Algorithm with Forward Crossover Method for Task Chromosome and Reflection Crossover Method for Model Chromosome ($ALGS_2$)

The steps of the genetic algorithm with the forward crossover method for task chromosome and the reflection crossover method for model chromosome proposed in this paper for the mixed model assembly line balancing problems are same as that of $ALGS_1$, except the Step 11. The required Step 11 of this algorithm is given below.

Step 11: Perform crossover operation using the forward crossover operation for different pairs of task chromosomes in the subpopulation of task chromosomes and obtain their offspring.

4.8. Genetic Algorithm with Reverse Crossover Method for Task Chromosome and Reflection Chromosome for Model Chromosome ($ALGS_3$)

The steps of the genetic algorithm with the reverse crossover method for task chromosome and the reflection crossover method for model chromosome proposed in this paper are same as that of $ALGS_1$, except the Step 11. The required Step 11 of this algorithm is presented below.

Step 11: Perform crossover operation using the reverse crossover operation for different pairs of task chromosomes in the subpopulation of task chromosomes and obtain their offspring.

4.9. Genetic Algorithm with Cyclic Crossover Method for Task Chromosome and Reflection Crossover Method for Model Chromosome and Modified Workstation Formation ($ALGS_4$)

The steps of the genetic algorithm with the cyclic crossover method for task chromosome and the reflection crossover method for model chromosome and modified workstation formation proposed in this paper are same as that of $ALGS_1$, except the Step 5 and Step 14. The required Step 5 and Step 14 of this algorithm are shown below.

Step 5: Design the workstations of each task chromosome for the common cycle time using the task times of the individual models and the immediate predecessor(s) matrix of the combined model by serially assigning the tasks from the ordered vector and obtain the following:

- The balancing efficiencies of the models ($\eta_1, \eta_2, \eta_3, \cdots, \eta_M$);
- The average balancing efficiency of the models

$$\eta_{Ave} = \left[\frac{\sum_{i=1}^{M} \eta_i}{M} \right]$$

While moving from one workstation to another workstation, if there is idle time in the current workstation, then look for alternate succeeding task(s) which can best fit into the current workstation subject to precedence constraints that will result with either zero idle time or least idle time in that workstation.

Step 14: Design the workstations of each task offspring for the common cycle time using the task times of the individual models and the immediate predecessor(s) matrix of the combined model by serially assigning the tasks from the ordered vector and obtain the following:

- The balancing efficiencies of the models ($\eta_1, \eta_2, \eta_3, \cdots, \eta_M$);
- The average balancing efficiency of the models;

$$\eta_{Ave} = \left[\frac{\sum_{i=1}^{M} \eta_i}{M} \right]$$

- Replace the corresponding task chromosome with this task offspring along with its average balancing efficiency.

While moving from one workstation to another workstation, if there is idle time in the current workstation, then look for alternate succeeding task(s) which can best fit into the current workstation subject to precedence constraints that will result with either zero idle time or least idle time in that workstation.

5. Comparison of Genetic Algorithms

This section presents a comparison of the four genetic algorithms, viz. ALGS1, ALGS2, ALGS3 and ALGS4 using a complete factorial experiment with three factors, viz. Problem Size (*A*), Algorithm (*B*) and Cycle Time (*C*). The levels of the Problem Size (*A*) are from 40 tasks to 65 tasks in steps of 5 tasks. The levels of Algorithm (*B*) are ALGS1, ALGS2, ALGS3 and ALGS4. For each problem, the Cycle Time (*C*) is set at two levels, viz.75 min and 100 min. The number of replications under each experimental combination of the three factors is 2. So, the total number of observations of this experiment is 96. The ANOVA model of this complete factorial experiment [50] is shown below.

$$Y_{ijkl} = \mu + A_i + B_j + AB_{ij} + C_k + AC_{ik} + BC_{jk} + ABC_{ijk} + e_{ijkl}$$

where,

μ is the overall mean

Y_{ijkl} is the multi-objective function value of the l^{th} replication under i^{th} level of the factor *A* (Problem Size), j^{th} level of the factor *B* (Algorithm) and k^{th} level of the factor *C* (Cycle Time).

A_i is the effect of the i^{th} level of the factor *A* (Problem Size) on the multi-objective function value.

B_j is the effect of the j^{th} level of the factor *B* (Algorithm) on the multi-objective function value.

AB_{ij} is the effect of the i^{th} level of the factor *A* (Problem Size) and the j^{th} level of the factor *B* (Algorithm) on the multi-objective function value.

C_k is the effect of the k^{th} level of the factor *C* (Cycle Time) on the multi-objective function value.

AC_{ik} is the effect of the i^{th} level of the factor *A* (Problem Size) and the k^{th} level of the factor *C* (Cycle Time) on the multi-objective function value.

BC_{jk} is the effect of the j^{th} level of the factor *B* (Algorithm) and the k^{th} level of the factor *C* (Cycle Time) on the multi-objective function value.

ABC_{ijk} is the effect of the i^{th} level of the factor *A* (Problem Size), j^{th} level of the factor *B* (Algorithm) and the k^{th} level of the factor *C* (Cycle Time) on the multi-objective function value.

e_{ijkl} is the random error associated with the multi-objective function value of the l^{th} replication under the i^{th} level of the factor A, j^{th} level of the factor B and the k^{th} level of the factor *C*.

The number of models considered in all the replications is 2. The ratio of production volumes of the model 1 and that of model 2 is assumed to be 2:3. This ratio is scaled up to 4:6 to fit into the size of the model chromosome of 10. The multi-objective function values (η_{Ave} – EPMS) of the replications under all the experimental combinations as per the design of the complete factorial experiment are given in **Table 20**. The results of ANOVA for the data given in the **Table 20** are summarized in **Table 21**. From the **Table 21**, it is clear that the components *A*, *B*, *AB* and *C* are significant and the remaining components are insignificant at a significance level of 0.05. Based on these results, the inferences of the ANOVA model are listed below.

- There is significant difference between the treatments of "Problem Size (*A*)" in terms of the multi-objective function value.
- There is significant difference between the treatments of "Algorithm (*B*)" in terms of the multi-objective function value.
- There is significant difference between the treatment combinations of "Problem Size (*A*) and Algorithm (*B*)" in terms of the multi-objective function value.
- There is significant difference between the treatments of "Cycle Time (*C*)" in terms of the multi-objective function value.
- There is no significant difference between the treatment combinations of "Problem Size (*A*)" and "Cycle

Table 20. Multi-objective (η_{AVE} – EPMS) function values.

Problem size (A)	Replication	Algorithm (B)							
		$ALGS_1$		$ALGS_2$		$ALGS_3$		$ALGS_4$	
		Cycle Time (C)		Cycle Time (C)		Cycle Time (C)		Cycle Time (C)	
		75	100	75	100	75	100	75	100
40	1	71.40	74.63	73.57	77.25	73.57	73.17	79.23	79.30
	2	74.20	76.70	74.20	76.70	74.20	76.70	77.00	79.08
45	1	67.72	76.96	69.20	71.25	67.72	71.25	87.31	85.50
	2	67.05	68.45	73.57	68.45	66.81	70.84	81.43	85.50
50	1	68.78	73.08	73.08	73.08	74.22	77.91	83.52	86.19
	2	68.55	72.55	68.55	72.83	68.55	72.55	77.32	78.92
55	1	70.61	70.17	67.52	70.17	67.52	70.17	83.37	87.48
	2	69.62	73.44	66.35	74.64	66.35	71.95	77.35	80.39
60	1	72.46	72.30	67.21	71.81	64.75	72.30	81.10	82.82
	2	69.50	69.50	63.01	76.72	68.29	71.60	82.34	83.46
65	1	75.78	75.78	68.70	75.78	68.70	71.34	81.74	80.83
	2	74.37	74.38	70.28	66.11	66.11	73.23	84.48	83.61

Table 21. Results of ANOVA.

Source of Variation	Sum of Squares	Degrees of Freedom	Mean Sum of Squares	F_{Cal}	F_{α} = 0.05.	Remark
A	83.500	5	16.700	2.420	2.418	Significant
B	2075.563	3	691.854	100.273	2.808	Significant
AB	299.313	15	19.954	2.892	1.888	Significant
C	177.688	1	177.688	25.753	4.048	Significant
AC	21.250	5	4.25	0.616	2.418	Insignificant
BC	20.875	3	6.958	1.009	2.808	Insignificant
ABC	92.125	15	6.142	0.890	1.888	Insignificant
Error	331.188	48	6.900			
Total	3101.500	95				

Time (*C*)" in terms of the multi-objective function value.

- There is no significant difference between the treatment combinations of "Algorithm (*B*)" and "Cycle Time (*C*)" in terms of the multi-objective function value.
- There is no significant difference between the treatment combinations of "Problem Size (*A*)", "Algorithm (*B*)" and "Cycle Time (*C*)" in terms of the multi-objective function value.

Since, there is significant difference between the algorithms (Factor *B*), viz. ALGS1, ALGS2, ALGS3 and ALGS4 in terms of the multi-objective function value, the next step is to identify the best algorithm using Duncan's multiple range test.

Determination of Best Algorithm Using Duncan's Multiple Range Test

The best algorithm is identified using Duncan's multiple range test.

Step: 1: The ascending order of the mean multi-objective function values of the algorithms is as shown below. The mean multi-objective fitness function value of an algorithm means the mean of the multi-objective fitness function values under that algorithm irrespective of the levels of other factors.

Algorithm	**ALGS3**	**ALGS2**	**ALGS1**	**ALGS4**
Mean multi-objective value(%)	70.83	71.25	71.99	82.05

Step 2: The standard error of algorithm-mean of the multi-objective function value is computed as shown below.

$$\text{Std Error} = \left[\text{MSS}_{\text{error}} / \text{No. of replications under each algorithm}\right]^{1/2}$$

where,

$\text{MSS}_{\text{error}}$ is as shown in **Table 21**, which is 6.9.

Std. Error = $[6.9/24]^{1/2}$ = 0.53619.

Step 3: The three (4 - 1) significant ranges from Duncan's table for error degrees of freedom of 48 at α = 0.05 are shown below.

j	2	3	4
Significant range	2.848	2.998	3.092

Step 4: The three least significant ranges (LSR) are obtained by multiplying the respective significant ranges with the standard error and they are as shown below.

j	2	3	4
LSR_j	1.527	1.608	1.658

Step 5: The actual ranges (AR) between different means of the algorithms along with corresponding LSR values are shown in **Figure 5**.

From **Figure 5**, it is clear that the algorithm ALGS4 is significantly different from the algorithms ALGS1, ALGS2 and ALGS3 in terms of the multi-objective function value. The actual ranges of the remaining combinations of the algorithms are less than the respective LSR values. This means that there is no significant difference between the corresponding pairs of algorithms (ALGS1 and ALGS3, ALGS1 and ALGS2, and ALGS2 and ALGS3) in terms of the multi-objective function value.

Since the algorithm ALGS4 is significantly different and superior to ALGS1, ALGS2 and ALGS3 in terms of the multi-objective function value (η_{AVE} − EPMS), the algorithm ALGS4 is identified as the best algorithm to solve this mixed- model assembly line balancing problem with sequencing of models to maximize the average balancing efficiencies of the models and minimize the makespan of sequencing the models.

6. Conclusions

It is well known that the mixed model assembly line balancing problem attracts researchers and practitioners because of its use to cope up with growing global competition. Further, it comes under combinatorial category, which gives challenge in terms of designing efficient algorithm to obtain assembly line balancing solution. In this paper, the mixed model assembly line balancing problem with model sequencing maximizes the average balancing efficiency of the models and minimizes the makespan of sequencing the models through the workstations. Since, the two components of the objective function are in different units as well as with opposing direction of optimality, the second component, that is the makespan is converted into excess percentage of makespan (EPMS) and subtracted from the first component to maintain the multi-objective function as a maximization type.

In this research, an attempt has been made to develop a set of genetic algorithms to maximize the multi-objective function (η_{AVE} − EPMS) for the mixed model assembly line balancing problem with sequencing of models

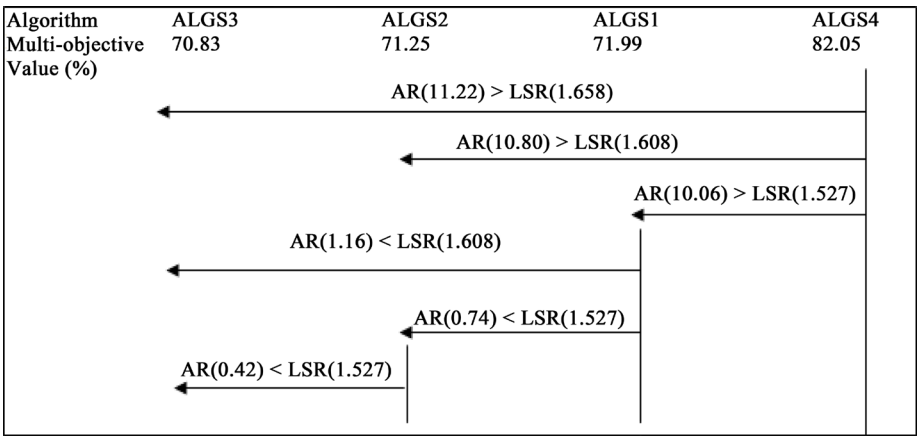

Figure 5. Comparison of actual ranges (AR) with respective least significant ranges (LSR).

and select the best amongst them through a carefully designed complete factorial experiment.

The three genetic algorithms as presented below have been developed. Along with these three genetic algorithms, the authors consider the genetic algorithm with cyclic crossover method (ALG_1) developed by Sivasankaran and Shahabudeen [2] with necessary modification to handle model sequencing ($ALGS_1$), which is called as *Genetic algorithm with cyclic crossover method for task chromosome and reflection crossover method for model chromosome, for comparison.*

- Genetic algorithm with forward crossover method for task chromosome and reflection crossover method for model chromosome, $ALGS_2$.
- Genetic Algorithm with reverse crossover method for task chromosome and reflection crossover for model chromosome, $ALGS_3$.
- Genetic algorithm with cyclic crossover method for task chromosome and reflection crossover method for model chromosome and modified workstation formation, $ALGS_4$.

These four algorithms are compared using a complete factorial experiment with four factors, viz. Problem Size (*A*), Algorithm (*B*) and Cycle Time (*C*). The number of levels of the factor "Problem Size (*A*)" is 6, viz. 40 tasks, 45 tasks, 50 tasks, 55 tasks, 60 tasks and 65 tasks. The factor "Algorithm (*B*)" is with two levels, viz. $ALGS_1$, $ALGS_2$, $ALGS_3$ and $ALGS_4$. The cycle time (Factor *C*) is set at two levels, viz. 75 sec. and 100 sec. The number of replications under each experimental combination of the four-factor experiment is 2.

Based on the results of the four factor ANOVA experiment, it is observed that the Factor *A* (Problem Size), Factor *B* (Algorithm), *AB* (Problem Size x Algorithm) and Factor *C* (Cycle Time) are having significant effect on the multi-objective function value, which is a combination of the average balancing efficiency of the models and makespan of sequencing the models at a significance level of 0.05. Other components of the ANONA model do not have effect on the multi-objective function value of the mixed model assembly line balancing problem with model sequencing at a significance level of 0.05. From this observation, it is clear that there is significant difference between the genetic algorithms (Factor *C*) in terms of multi-objective function values.

Since, there is significant difference between the algorithms in terms of the mean of the multi-objective function value [η_{AVE} − (EPMS)], the best algorithm is determined using Duncan's multiple range test, which gives ALGS4 (Genetic algorithm with cyclic crossover method for task chromosome and reflection crossover method for model chromosome, and modified workstation formation) as the best algorithm to the mixed model assembly line balancing problem in which the multi-objective function value is maximized.

In future research, probabilistic task times for the mixed model assembly line balancing may be assumed and accordingly the results may be analyzed.

Acknowledgements

We thank the anonymous referees for their constructive suggestions, which improved the paper.

References

[1] Panneerselvam, R. (2012) Production and Operations Management. 3rd Edition, PHI Learning Private Ltd., New Delhi

[2] Sivasankaran, P. and Shahabudeen, P. (2013) Genetic Algorithm for Concurrent Balancing of Mixed-Model Assembly Lines with Original Task Times of Models. *Intelligent Information Management*, **5**, 84-92. http://dx.doi.org/10.4236/iim.2013.53009

[3] Gokcen, H. and Erel, E. (1997) A Goal Programming Approach to Mixed Model Assembly Line Balancing Problem. *International Journal of Production Economics*, **48**, 177-185.

[4] Gokcen, H. and Erel, E. (1998) Binary Integer Formulation for Mixed-Model Assembly Line Balancing Problem. *Computers & Industrial Engineering*, **34**, 451-461. http://dx.doi.org/10.1016/S0360-8352(97)00142-3

[5] Kara, Y., Ozgiiven, C., Seeme, N.Y. and Chang, C.T. (2011) Multi-Objective Approaches to Balance Mixed-Model Assembly Lines for Model Mixes Having Precedence Conflicts and Duplicate Common Tasks. *International Journal of Advanced Manufacturing Technology*, **52**, 725-737. http://dx.doi.org/10.1007/s00170-010-2779-z

[6] Sivasankaran, P. and Shahabudeen, P. (2013) Modelling Hybrid Single Model Assembly Line Balancing Problem. *Udyog Pragati*, **37**, 26-36.

[7] Akpinar, S. and Baykasoglu, A. (2014) Modelling and Solving Mixed-Model Assembly Line Balancing Problem with Setups. Part I: A Mixed Integer Programming Model. *Journal of Manufacturing Systems*, **33**, 177-187. http://dx.doi.org/10.1016/j.jmsy.2013.11.004

[8] Bukchin, Y. and Rabinowitch, I. (2006) A Branch-and-Bound Based Solution Approach for the Mixed-Model Assembly Line-Balancing Problem for Minimizing Stations and Task Duplication Costs. *European Journal of Operational Research*, **174**, 492-508. http://dx.doi.org/10.1016/j.ejor.2005.01.055

[9] Li, J. and Gao, J. (2014) Balancing Manual Mixed-Model Assembly Lines Using Overtime Work in a Demand Variation Environment. *International Journal of Production Research*, **52**, 3552-3567. http://dx.doi.org/10.1080/00207543.2013.874603

[10] Matanachai, S. and Yano, C.A. (2001) Balancing Mixed-Model Assembly Lines to Reduce Work Overload. *IIE Transactions*, **33**, 29-42. http://dx.doi.org/10.1080/07408170108936804

[11] Jin, M. and Wu, S.D. (2002) A New Heuristic Method for Mixed-Model Assembly Line Balancing Problem. *Computers & Industrial Engineering*, **44**, 159-169. http://dx.doi.org/10.1016/S0360-8352(02)00190-0

[12] Hop, N.V. (2006) A Heuristics Solution for Fuzzy Mixed-Modelling Balancing Problem. *European Journal of Operational Research*, **168**, 798-810. http://dx.doi.org/10.1016/j.ejor.2004.07.029

[13] Bock, S. (2008) Using Distributed Search Methods for Balancing Mixed-Model Assembly Lines in the Automotive Industry. *OR Spectrum*, **30**, 551-578. http://dx.doi.org/10.1007/s00291-006-0069-9

[14] Mamun, A.A., Khaled, A.A., Ali, S.M. and Chowdhury, M.M. (2012) A Heuristic Approach for Balancing Mixed-Model Assembly Line of Type I Using Genetic Algorithm. *International Journal of Production Research*, **50**, 5106-5116. http://dx.doi.org/10.1080/00207543.2011.643830

[15] Su, P., Wu, N. and Yu, Z. (2014) A Petrinet-Based Heuristic for Mixed-Model Assembly Line Balancing Problem of Type E. *International Journal of Production Research*, **52**, 1542-1556. http://dx.doi.org/10.1080/00207543.2013.849010

[16] Chutima, P. and Iammi, J. (2003) Application of Genetic Algorithms in Mixed Model Assembly Line Balancing. *KMUTT Research & Development Journal*, **26**, 1-16.

[17] Bai, Y., Zhao, H. and Zhu, L. (2009) Mixed-Model Assembly Line Balancing Using the Hybrid Genetic Algorithm. *International Conference on Measuring Technology and Mechatronics Automation*, **3**, 242-245. http://dx.doi.org/10.1109/icmtma.2009.591

[18] Senthilkumar, P. and Shahabudeen, P. (2006) GA Based Heuristic for the Open Shop Scheduling Problem. *The International Journal of Advanced Manufacturing Technology*, **30**, 297-301. http://dx.doi.org/10.1007/s00170-005-0057-2

[19] Yagmahan, B. (2011) Mixed-Model Assembly Line Balancing Using a Multi-Objective Ant Colony Optimization Approach. *Expert Systems with Applications*, **38**, 12453-12461. http://dx.doi.org/10.1016/j.eswa.2011.04.026

[20] Akpinar, S. and Baykasoglu, A. (2014) Modelling and Solving Mixed-Model Assembly Line Balancing Problem with Setups. Part II: A Multiple Colony Hybrid Bees Algorithm. *Journal of Manufacturing Systems*, **33**, 445-461. http://dx.doi.org/10.1016/j.jmsy.2014.04.001

[21] Dar-El, E.M. and Chcuy, S. (1977) Optimal Mixed-Integer Sequencing for Balancing Assembly Lines. *Omega*, **5**, 333-342. http://dx.doi.org/10.1016/0305-0483(77)90115-3

[22] Ding, F.-Y. and Cheng, L. (1993) A Simple Sequencing Algorithm for Mixed-Model Assembly Lines in Just-in-Time Productions Systems. *Operations Research Letters*, **13**, 27-36. http://dx.doi.org/10.1016/0167-6377(93)90081-Q

[23] Leu, Y.-Y., Huang, P.Y. and Rusell, R.S. (1997) Using Beam Search Techniques for Sequencing Mixed-Model Assembly Lines. *Annals of Operations Research*, **70**, 379-397. http://dx.doi.org/10.1023/A:1018938608304

[24] Ding, F.-Y., Zhu, J. and Sun, H. (2006) Comparing Two Weighted Approaches for Sequencing Mixed-Model Assembly Lines with Multiple Objectives. *International Journal of Production Economics*, **102**, 108-131. http://dx.doi.org/10.1016/j.ijpe.2005.02.007

[25] Rabbani, M., Rahimi-Vahed, A., Javadi, B. and Tavakkoli-Moghaddam, R. (2006) A New Approach for Mixed-Model Assembly Line Sequencing. In: Waldmann, K.-H. and Stocker, U.M., Eds., *Operations Research Proceedings* 2006, Springer, Berlin, 169-174.

[26] Rahimi-Vahed, A.R., Rabbani, M., Tavakkoli-Moghaddam, R., Torabi, S.A. and Jolai, F. (2007) A Multi-Objective Scatter Search for a Mixed-Model Assembly Line Sequencing Problem. *Advanced Engineering Informatics*, **21**, 85-99. http://dx.doi.org/10.1016/j.aei.2006.09.007

[27] Erel, E., Gocgun, Y. and Sabuncuoglu, I. (2007) Mixed-Model Assembly Line Sequencing Using Beam Search. *International Journal of Production Research*, **45**, 5265-5284. http://dx.doi.org/10.1080/00207540600806497

[28] Bautista, J. and Cano, A. (2011) Solving Mixed Model Sequencing Problem in Assembly Lines with Serial Workstations with Work Overload Minimization and Interruption Rules. *European Journal of Operational Research*, **210**, 495-513. http://dx.doi.org/10.1016/j.ejor.2010.10.022

[29] Gujjula, R., Werk, S. and Giinther, H.O. (2011) A Heuristic Based on Vogel's Approximation Method for Sequencing Mixed-Model Assembly Lines. *International Journal of Production Research*, **49**, 6451-6468.

http://dx.doi.org/10.1080/00207543.2010.527384

[30] Lin, D.Y. and Chu, Y.M. (2014) A Lagrangian Relaxation Approach to the Mixed-Product Assembly Line Sequencing Problem: A Case Study of Door-Lock Company in Taiwan. *Applied Mathematical Modeling*, **38**, 4493-4511. http://dx.doi.org/10.1016/j.apm.2014.02.029

[31] Scholl, A., Klein, R. and Domschke, W. (1998) Pattern Based Vocabulary Building for Effectively Sequencing Mixed-Model Assembly Lines. *Journal of Heuristics*, **4**, 359-381. http://dx.doi.org/10.1023/A:1009613925523

[32] Kim, Y.K., Hyun, C.J. and Kim, Y. (1996) Sequencing in Mixed Model Assembly Lines: A Genetic Algorithm Approach. *Computers and Operations Research*, **23**, 1131-1145. http://dx.doi.org/10.1016/S0305-0548(96)00033-0

[33] Leu, Y.-Y., Matheson, L.A. and Rees, L.P. (1996) Sequencing Mixed-Model Assembly Lines with Genetic Algorithms. *Computers and Industrial Engineering*, **30**, 1027-1036. http://dx.doi.org/10.1016/0360-8352(96)00050-2

[34] Hyun, C.J., Kim, Y. and Kim, Y.K. (1998) A Genetic Algorithm for Multiple Objective Sequencing Problems in Mixed Model Assembly Lines. *Computers & Operations Research*, **25**, 675-690. http://dx.doi.org/10.1016/S0305-0548(98)00026-4

[35] Ponnambalam, S.G., Aravindan, P. and Subbu Rao, M. (2003) Genetic Algorithms for Sequencing Problems in Mixed Model Assembly Lines. *Computers & Industrial Engineering*, **45**, 669-690. http://dx.doi.org/10.1016/j.cie.2003.09.001

[36] Su, P. and Lu, Y. (2007) Combining Genetic Algorithm and Simulation for the Mixed-Model Assembly Line Balancing Problem. 3*rd International Conference on Natural Computation* (*ICNC* 2007), **4**, 314-318. http://dx.doi.org/10.1109/ICNC.2007.306

[37] Zhao, X.B. and Ohno, K. (1997) Algorithms for Sequencing Mixed Models on an Assembly Line in a JIT Production System. *Computers & Industrial Engineering*, **32**, 47-56. http://dx.doi.org/10.1016/S0360-8352(96)00193-3

[38] Fattahi, P. and Salehi, M. (2009) Sequencing the Mixed-Model Assembly Line to Minimize the Total Utility and Idle Costs with Variable Launching Interval. *International Journal of Advanced Manufacturing Technology*, **45**, 987-998. http://dx.doi.org/10.1007/s00170-009-2020-0

[39] Amlashi, Z.Z. and Zandieh, M. (2011) Sequencing Mixed Model Assembly Line Problem to Minimize Line Stoppages Cost by a Modified Simulated Annealing Algorithm Based on Cloud Theory. *Journal of optimization in Industrial Engineering*, **8**, 9-18.

[40] Rahimi-Vahed, A. and Mirzaei, A.H. (2007) A Hybrid Multi-Objective Shuffled Frog-Leaping Algorithm for a Mixed-Model Assembly Line Sequencing Problem. *Computers & Industrial Engineering*, **53**, 642-666. http://dx.doi.org/10.1016/j.cie.2007.06.007

[41] Thomopoulos, N.T. (1967) Line Balancing-Sequencing for Mixed-Model Assembly. *Management Science*, **14**, B59-B75. http://dx.doi.org/10.1287/mnsc.14.2.B59

[42] Merengo, C., Nava, F. and Pozzetti, A. (1999) Balancing and Sequencing Manual Mixed-Model Assembly Lines. *International Journal of Production Research*, **37**, 2835-2860. http://dx.doi.org/10.1080/002075499190545

[43] Kim, Y.K. and Kim, J.Y. (2000) A Co-Evolutionary Algorithm for Balancing and Sequencing in Mixed Model Assembly Lines. *Applied Intelligence*, **13**, 247-258. http://dx.doi.org/10.1023/A:1026568011013

[44] Lovgren, R.H. and Racer, M. (2000) Algorithms for Mixed-Model Sequencing with Due Date Restrictions. *European Journal of Operational Research*, **120**, 408-422. http://dx.doi.org/10.1016/S0377-2217(99)00091-0

[45] Hwang, R. and Katayaa, H. (2010) Integrated Procedure of Balancing and Sequencing for Mixed-Model Assembly Lines: A Multi-Objective Evolutionary Approach. *International Journal of Production Research*, **48**, 6417-6441. http://dx.doi.org/10.1080/00207540903289755

[46] Ozcan, U., Cercioglu, H., Gokcen, H. and Toklu, B. (2010) Balancing and Sequencing of Parallel Mixed-Model Assembly Lines. *International Journal of Production Research*, **48**, 5089-5113. http://dx.doi.org/10.1080/00207540903055735

[47] Mosadegh, H., Zandieh, M. and Ghomi, S.M.T.F. (2012) Simultaneous Solving of Balancing and Sequencing Problems with Station-Dependant Assembly Times for Mixed-Model Assembly Lines. *Applied Soft Computing*, **12**, 1359-1370. http://dx.doi.org/10.1016/j.asoc.2011.11.027

[48] Panneerselvam, R. (2016) Design and Analysis of Algorithms. 2nd Edition, PHI Learning Private Ltd., New Delhi.

[49] Panneerselvam, R. (2012) Research Methodology. 2nd Edition, PHI Learning Private Ltd., New Delhi

[50] Panneerselvam, R. (2012) Design and Analysis of Experiments. PHI Learning Private Limited, New Delhi.