Int J Adv Manuf Technol (2014) 73:1665–1694
DOI 10.1007/s00170-014-5944-y

ORIGINAL ARTICLE

Literature review of assembly line balancing problems

P. Sivasankaran · P. Shahabudeen

Received: 16 October 2011 / Accepted: 11 May 2014 / Published online: 27 May 2014

Abstract Mass production system design is a key for the productivity of an organization. Mass production system can be classified into production line machining a component and production line assembling a product. In this paper, the production line assembling a product, which is alternatively called as assembly line system, is considered. In this system, balancing the assembly line as per a desired volume of production per shift is a challenging task. The main objectives of the assembly line design are to minimize the number of workstations for a given cycle time (type 1), to minimize the maximum of the times of workstations for a given number of workstations (type 2), and so forth. Because this problem comes under combinatorial category, the use of heuristics is inevitable. Development of a mathematical model may also be attempted, which will help researchers to compare the solutions of the heuristics with that of the model.

In this paper, an attempt is made to present a comprehensive review of literature on the assembly line balancing. The assembly line balancing problems are classified into eight types based on three parameters, viz. the number of models (single-model and multi-model), the nature of task times (deterministic and probabilistic), and the type of assembly line (straight-type and U-type). The review of literature is organized as per the above classification. Further, directions for future research are also presented.

P. Sivasankaran (✉)
Department of Mechanical Engineering, ARS College of Engineering, Maraimali Nagar, Chennai 603209, India
e-mail: sivasankaran.panneerselvam@yahoo.com

P. Shahabudeen
Department of Industrial Engineering, College of Engineering, Anna University, Chennai 600 025, India

Keywords Assembly line balancing · Single model · Multi-model · One-sided · Two-sided · Balancing efficiency · U-type · Cycle time

1 Introduction

The production system is classified into mass production system and batch production system [73]. In mass production system, manufacturing facility is geared up to produce the products of interest in large volume, whereas in batch production system, manufacturing facility is geared up to produce the products in much smaller volumes. A flow line may be used for the mass production system, which produces the same product over a long period of time. The batch production is realized through job shop implementation.

Generally, product layout is used for mass production system [73]. The layout for assembling washing machine and that for machining connecting rod used in internal combustion engine constitute examples of the product layout. In the case of assembling a product, it will contain a set of tasks which are to be processed as per the relationship defined in the form of a precedence network. As per the network relationship, some of the tasks will be processed in serial order and some of them will be processed in parallel. These tasks are to be grouped into different workstations without violating the precedence relationships such that the assumed measure(s) of performance is/are optimized, which is known as design of assembly line system. The design of the assembly line system to assemble a product mainly involves minimization of the number of workstations, balancing workload between workstations, etc., which helps a company to better utilize its facilities and produce exact number of units of a product to meet the demand of that product in the case of custom made product. If the product is made to stock in anticipation of future demand of that product, then the design of the assembly line helps the company to better utilize its facilities. This problem comes under combinatorial category, which requires more computational time to solve large-size problems. When a component is machined using a product layout, the necessary machines are to be arranged as per the process sequence of that component, whose layout design does not warrant a complex analysis. In this paper, the design of the assembly line system is considered.

The inputs for the design of the assembly line system are as listed below:

- Precedence network of tasks [73]
- Task times, which may be either deterministic or probabilistic
- Cycle time or number of workstations

The precedence network defines the immediate precedence relationships among the tasks of assembling a product. The execution of each task requires certain time, which is known as task time. This may be deterministic or probabilistic. The cycle time is the time between consecutive releases of the assemblies at the end of the line or the total time (maximum time) allocated to each workstation in the assembly line. All workstations have the same cycle time.

The formula for the cycle time (CT) [73] is as given below:

$$\mathrm{CT} = \frac{\text{Effective time available per shift}}{\text{Production volume/shift}}$$

The cycle time and the number of workstations are expected to be inversely proportional. If the cycle time is more, the number of workstations is expected to be less and vice versa. If the objective is to minimize the number of workstations for a given production rate, it is usually referred in literature as SALB-1 (type 1) problem. If the goal is to maximize the production rate by minimizing the maximum of the sum of the task times of the workstations for a given number of workstations, then it is referred as SALB-2 (type 2) problem.

The formula to compute the balancing efficiency [73] is given below:

$$\text{Balancing efficiency} = \frac{\text{Sum of all task times}}{\text{Number of workstations} \times \text{Cycle time}} \times 100$$

In the formula for the balancing efficiency, the sum of all task times and cycle time is given as input. The cycle is computed based on a desired production volume of a product that is to be assembled in the line. The balancing efficiency is the ratio between the sum of the task times and the total time that is provided to execute all the tasks (Number of workstations ×Cycle time). From the formula, it is clear that the lesser the number of workstations, the more is the balancing efficiency, and the lesser is the requirement of resources (Operators).

The prime objectives of the assembly line balancing (ALB) problem are as listed below:

- To subdivide the tasks in a given precedence network into a number of workstations for a given cycle time subject to the following two conditions such that the balancing efficiency is maximized (SALB-1 problem):
  - non-violation of the precedence constraints among the tasks
  - sum of the processing times of the tasks assigned to each workstation is less than or equal to the given cycle time
- To subdivide the tasks in a given precedence network into a given number of workstations without violating the precedence constraints among the tasks such that the cycle time is minimized (SALB-2 problem)

In this paper, a comprehensive review of literature of the assembly line balancing problem is carried out. The contributions of various researchers are presented in eight categories and an in-depth review of literature in each category is presented along with weaknesses, strengths, and directions for future researches. This review will help researchers to know the state-of-the-art assembly line balancing researches and to select topics for future researches.

In this paper, first the classification of the assembly line balancing problems is presented. It is followed by a major section on the review of literature of single-model deterministic assembly line balancing problems, which, in turn, consists of two subsections, viz. single-model deterministic straight-type problem and single-model deterministic U-type problem. The next section gives the review of literature of single-model probabilistic assembly line balancing problems, which consists of subsections on single-model probabilistic straight-type problem and single-model probabilistic U-type problem. It is followed by a section on multi-model deterministic assembly line balancing problems. This, in turn, consists of subsections on multi-model deterministic straight-type problem and multi-model deterministic U-type problem. The last section of the review is on multi-model probabilistic assembly line balancing problems, which, in turn, consists of subsections on multi-model probabilistic straight-type problem and multi-model probabilistic U-type problem. Within each of these eight subsections of the review, contributions of the researchers are further classified as per the methods used to solve the respective assembly line balancing problems. After the review of literature, a section on three different types of analysis of the assembly line balancing literature is presented. At the end, a quick recap of the review and directions for future researches are discussed in conclusion.

2 Classification of assembly line balancing problems

The assembly line balancing (ALB) problems can be classified based on the number of models produced in the line, the nature of task times (deterministic or probabilistic) and the nature of flow (straight-type or U-type). In the same assembly line, one or more models of a product may be assembled. If only a single model is assembled in the line, then the production system is defined as a single-model assembly system; otherwise, it is called a multi-model assembly system. The processing times of the tasks may be either deterministic or probabilistic. If the tasks are performed using all sophisticated tools and fixtures by highly skilled labors, then the processing times of the tasks may be approximated to deterministic quantity, because the variability in the processing times may be less under such situation. This is because of the facilitating nature of tools and availability of operators with required skills. But, normally, in assembly-type operations, the processing times will vary, which can be characterized in the form of some probability distribution. The arrangement of the workstations of the assembly line may be in a straight-line layout or in a U-shape layout [1, 3]. In the U-shape layout, an operator may manage more than one workstation.

In this paper, the hierarchy of classification based on these parameters of the *ALB* problems is shown in Fig. 1. The resultant categories based on the above parameters are listed below:

1. Single-model deterministic straight-type (SM_D_S) problem
2. Single-model deterministic U-type (SM_D_U) problem
3. Single-model probabilistic straight-type (SM_P_S) problem
4. Single-model probabilistic U-type (SM_P_U) problem
5. Multi-model deterministic straight-type (MM_D_S) problem
6. Multi-model deterministic U-type (MM_D_U) problem
7. Multi-model probabilistic straight-type (MM_P_S) problem
8. Multi-model probabilistic U-type (MM_P_U) problem

As stated in Section 1, the basic classifications of the assembly line balancing problems are SALB-1 problem and SALB-2 problem. Some authors have considered both the classifications, i.e., SALB-1 problem and SALB-2 problem. So further subcategories are formed in each of the eight categories shown in Fig. 1 by nesting SALB-1, SALB-2, and SALB-1 & SALB-2. For example, SM_D_S-1 means SM_D_S category for SALB-1 problem, SM_D_S-2 means SM_D_S category for SALB-2 problem. The resultant extended categories are presented below:

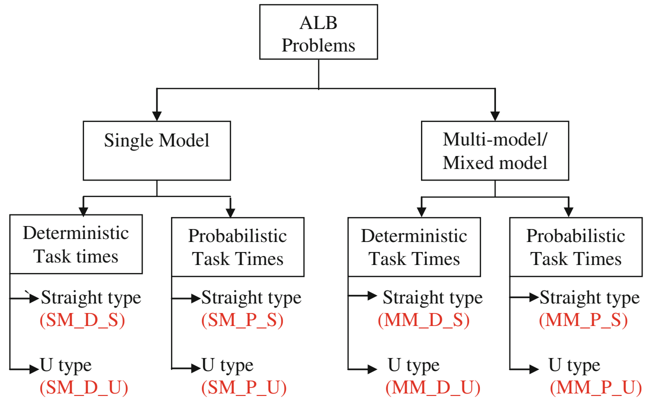


Fig. 1 Classification of ALB problems

- SM_D_S-1, SM_D_S-2, and SM_D_S-1 & SM_D_S-2
- SM_D_U-1, SM_D_U-2, and SM_D_U-1 & SM_D_U-2
- SM_P_S-1, SM_P_S-2, and SM_P_S-1 & SM_P_S-2
- SM_P_U-1, SM_P_U-2, and SM_P_U-1 & SM_P_U-2
- MM_D_S-1, MM_D_S-2, and MM_D_S-1 & MM_D_S-2
- MM_D_U-1, MM_D_U-2, and MM_D_U-1 & MM_D_U-2
- MM_P_S-1, MM_P_S-2, and MM_P_S-1 & MM_P_S-2
- MM_P_U-1, MM_P_U-2, and MM_P_U-1 & MM_P_U-2

In each of the following eight subsections (Sections 2.1.1 and 2.1.2, Sections 2.2.1 and 2.2.2, Sections 2.3.1 and 2.3.2, and Sections 2.4.1 and 2.4.2.), the review of literature of the eight categories shown in Fig. 1 is presented. At the end of each category, a review of extended categories by nesting SALB-1, SALB-2, and SALB-1 & SALB-2 is presented.

Boysen et al. [14] studied the classification of assembly line balancing problems. Rashid et al. [77] reviewed on assembly sequence planning and ALB optimization using soft computing approaches. Tasan and Tunali [89] carried out a review of literature of genetic algorithms (GAs) applied to the assembly line balancing problems.

2.1 Single-model (simple) deterministic ALB problems

This section presents a review of literature of single-model deterministic assembly line balancing (ALB) problems. This type of problems deals with assembly of only one model in a line, in which the task times are deterministic. These problems are further classified into two types based on the type of layout used to assembly the model, viz. straight-type layout and U-type layout. Hence, the classification under this category is as listed below:

- Single-model deterministic straight-type (SM_D_S) problem
- Single-model deterministic U-type (SM_D_U) problem

2.1.1 Single-model deterministic straight-type (SM_D_S) problem

The single-model deterministic straight-type (SM_D_S) assembly line balancing problem is the simplest among all the ALB problems. This problem considers single model which is to be assembled in the assembly line in which the workstations are arranged in a straight-line form. The execution time of each task of the single model (product) that is assembled in the assembly line is deterministic. This section presents a review of literature of the SM_D_S assembly line balancing problem. Based on the methods used to solve this problem, the literature under this category is further classified as listed below:

- Mathematical models
- Petri net algorithms
- Heuristics
- Genetic algorithms
- Simulated annealing algorithms
- Tabu search algorithms
- Ant colony optimization (ACO) algorithms
- Shortest path algorithm
- Memetic algorithm
- Bee algorithms

Mathematical models Thangavelu and Shetty [90] showed that certain steps in Geoffrion's 0–1 integer programming algorithm can be simplified or eliminated to solve the simple assembly line balancing problem such that the number of workstations is minimized for a given cycle time. Deckro and Rangachari [25] developed a goal programming model for the simple assembly line balancing problem with the objective of minimizing the number of workstations by simultaneously considering various operational requirements such as zoning, sequencing, idle time, cycle time, and costs. The research carried out by Ozcan and Toklu [72] presents a mathematical model for pre-emptive goal and a fuzzy goal programming model for imprecise goals for two-sided assembly line balancing problem. The mathematical model minimizes the number of mated workstations as the primary objective function, and it minimizes the number of workstations as a secondary objective for a given cycle time. The zoning constraints are also considered in this model. The goals considered in this model are the number of mated stations, cycle time, and load of workstations. Future research may be directed to extend this model for multi-model two-sided assembly lines.

Sivasankaran and Shahabudeen [85] developed a hybrid mathematical model for the single-model assembly line balancing problem. In the first stage, the objective of minimizing the number of workstations for a given cycle time is achieved, and for the number of workstations that is determined in the first stage, the objective of minimizing the cycle time is achieved in the second stage. The extra production that is realized using the second-stage optimization can be used to meet demand if there is any fluctuation in it in future. As an extension to this research, meta-heuristics, viz. simulated annealing (SA) algorithm, genetic algorithm, ACO algorithm, particle swarm optimization algorithm, etc., may be developed for this hybrid assembly line balancing problem.

Petri-net algorithms Kilincci and Bayhan [46] considered the SALB-1 problem, in which the objective is to maximize the balancing efficiency for a given cycle time. They developed a Petri net approach for this problem, which searches enabled transitions (or assignable tasks) in the Petri net model of precedence relations between tasks, and then the task which minimizes the idle time is assigned to the solution under consideration. The Petri net is a mathematical and graphical tool to model and analyze discrete event systems. Future work may be directed to compare this algorithm with a branch and bound algorithm. Further, this algorithm may be tried for SALB-2 problem, in which the objective is to minimize the cycle time for a given number of workstations.

Kilincci [45] considered the SALB-2 problem in which the objective is to minimize the variations in workloads among the workstations for a given number of workstations. The author developed a Petri net heuristic for this problem. The heuristic determines available tasks and assign them to current workstation by using reachability analysis, one of the main properties of Petri nets and token movement. To improve the solution, a binary search procedure is implemented between the first feasible solution and the last infeasible solution. They developed three versions of the heuristic by integrating with forward, backward, and bidirectional procedures. This work may be used as a seed generation algorithm in simulated annealing algorithm. Further, this algorithm may be combined with meta-heuristics like tabu search, genetic algorithm, etc. to give hybrid algorithms. Further, a new objective like workload smoothness may be added.

Heuristics Panneerselvam and Sankar [74] considered the single-model assembly line balancing problem in which the objective is to minimize the number of workstations for a given cycle time. They considered six heuristics of research by Dar-El [23] and proposed six new heuristics. Through a carefully designed experiment, they concluded that three heuristics of the contribution of Dar-El's [23] and all the six newly proposed heuristics by them are selected as the best set of heuristics to solve the problem. Future researchers can use the best solution of this set of heuristics as the seed of a simulated annealing algorithm. Panneerselvam and Sankar [74] as well as Dar-El [23] showed that an efficient set of heuristics will provide better solution for the assembly line balancing problem. So some more recent non-meta-heuristics may be added to the efficient set of heuristics identified by Panneerselvam and Sankar [74] to finalize the best set of heuristics to minimize the number of workstations for a given cycle time.

Ponnambalam et al. [75] carried out a comparative evaluation of six assembly line balancing heuristics, viz. rank positional weight, Kilbridge and Wester, Moodie and Young, Hoffman Precedence matrix, immediate update first fit and rank, and assign heuristics based on a number of excess stations, line efficiency, smoothness index, and Central Processing Unit (CPU) time. Genikomsakis and Tourassis [33] enhanced the largest set rule for assembly line balancing using the concept of bidirectional work relatedness for the simple assembly line balancing problem. Here, the objective is to design the assembly line such that the results give comparable cycle time and improved work relatedness.

Two-sided assembly line balancing is gaining importance in which different pairs of stations will be designed. The stations in the pair of stations facing each other are referred to as a mated station. Lee et al. [53] designed a group assignment procedure, which assigns a group of tasks at a time rather than assigning a unit task while forming stations in the simple assembly line balancing problem with deterministic task times. Their approach aimed to maximize work relatedness as well as task slackness. The task slackness is a quantification of the tightness of task sequences. When this value is 1, all the related tasks are assigned to the same side. This work may be extended for multi-objective problem. The measure of work relatedness is the ratio between the number of workstations and the sum of the number of sub-networks that will be formed with respect to the activities assigned to each workstation. This measure is closely related with graph theory. Hence, future researches may be directed to develop graph theoretic based algorithms to solve this assembly line balancing problem, if maximization of work relatedness is an objective.

Jiao et al. [43] developed a Web-based interactive advisor for assembly line balancing of hard disk assembly line involving multiple criteria in which the main criterion is to minimize the number of excess stations. The Web-based advisor will compose a schedule based on various heuristics embedded in its library. It also generates simulation models for the user-specified assembly line balancing problems and presents the user with performance evaluation of the suggested assembly line balancing solution. The objectives of this research are to maximize the output per week and the average station utilization. *This Web-based interactive advisor for assembly line balancing uses primitive heuristics. So researchers may include recent efficient heuristics into the existing set of heuristics of the Web-based interactive advisor for assembly line balancing.*

Nearchou [62] considered the single-model deterministic assembly line balancing problem with bi-criteria, viz. minimizing the cycle time of the assembly line and the balance delay time of the workstations, and minimizing the cycle time and maximizing the smoothness index of the workload of the assembly line. The author developed a new population heuristic to solve the problem based on the general differential evolution (DE) method. The differential evolution method is a population heuristic for global optimization over continuous search spaces. The main characteristics of this multi-objective DE (MODE) heuristic are formulation of a cost function of each individual ALB solution as a weighted sum of multiple objective functions with self-adapted weights, maintenance of a separate population with diverse Pareto-optimal solutions, injection of the actual evolving population with some Pareto-optimal solutions, and usage of a new modified scheme to create mutation vectors. In this work, a new self-adapted method to estimate the weights w_1 and $w_{2,}$ for the objective function components is developed. If one compares the formula given in this work with that given by Gen and Cheng [31], it will be very difficult to conclude which is the best. So future researches may be directed to compare the impact of these two formulas on the objective function of the assembly line balancing problem presented in this work.

Yeh and Kao [96] developed a bidirectional heuristic to group the tasks into a minimum number of workstations such that the balancing efficiency is maximized. In this approach, the bidirectional approach is coupled with critical path method (CPM), which is used in project management. This algorithm is compared with a unidirectional CPM heuristic and an optimal method. The comparison is just based on listing the number of workstations for different literature problems without any statistical investigation. This work may be modified to further improve the effectiveness and to solve various types of assembly line balancing problems, viz. U-type, parallel, resource-constrained, etc. Sotskov et al. [86] considered the simple assembly line balancing problem in which the objective is to minimize the number of stations for a given cycle time. The activities of the assembly line are divided into two subsets (Set 1 and Set 2), where Set 1 consists of activities having deterministic task times and Set 2 consists of activities having stochastic task times. The authors studied the stability of the optimal solution with respect to the variations in the task times in Set 2.

Genetic algorithms Rubinnovitz and Levitin [79] considered the single-model assembly line balancing problem with deterministic processing time. They developed a genetic algorithm and compared its results with that of MUST algorithm suggested by Dar-El and Rubinovitch [24]. Kim et al. [48] developed a genetic algorithm for the simple assembly line balancing problem with various objectives, which are minimizing the number of workstations, minimizing the cycle time, maximizing the work load smoothness, maximizing work relatedness and multiple objective which consists of maximizing the work load smoothness and maximizing work relatedness. A repair method is newly developed so that the traditional GA approach is flexibly adapted to various types of objectives in the assembly line balancing problem. They used a single-point crossover method in the proposed genetic algorithm and compared its performance graphically with that of other crossover methods used in the genetic algorithm and showed that the genetic algorithm with the single-point crossover method performs better. The statistical significance of the difference in the performances of the algorithms will be known only through analysis of variance (ANOVA), which is not carried out. Ponnambalam et al. [76] developed a multi-objective genetic algorithm for solving the simple assembly line balancing problem for a given cycle time. The objectives include the number of workstations, line efficiency, and the smoothness index. They compared its performance with six existing heuristics. The weights for the components of the fitness function are derived dynamically using the formula given by Murata et al. [60]. The three different objectives in the fitness functions are having different magnitudes, viz. an integer number for number of workstations, percentage for line efficiency, and a number for smoothness index. So a common scalar objective preferably in percentage may be derived for all the components based on the minimum and maximum values of each of them. The formula to find the scalar value of an objective component is the ratio between the actual value and the difference between the maximum and minimum values of that objective component. Otherwise, there will be imbalance among the objective components, because of their different magnitudes. Sabuncuoglu et al. [80] developed a genetic algorithm for the simple assembly line balancing problem, in which the times are deterministic. In this research, the objective is to find the minimum number of workstations. It uses a special chromosome structure that is partitioned dynamically through the evolution process. Elitism is also implemented in the model by using some concepts of simulated annealing. As an extension to this work, the partitioning procedure developed in this study can be applied to divide large assembly line balancing problems into smaller sub-problems to improve solution time. The dynamic partitioning technique proposed in this paper may be used in future hybrid algorithm in which genetic algorithm is a component, because it helps to reduce the CPU time while solving the problem. Levintin et al. [54] developed a genetic algorithm for robotic assembly line balancing (RALB) problem. Here, the operators are replaced by robots. The robotic assembly line balancing problem is defined for robotic assembly line, where different robots may be assigned to the assembly tasks and each robot needs different assembly times to perform a given task, because of its capabilities and specialization. The solution to the RALB problem includes an attempt for optimal assignment of robots to line stations and a balanced distribution of work between different stations such that the production rate of the line is maximized. Two different procedures, viz. a recursive assignment procedure and a consecutive assignment procedure for adapting the GA to the RALB problem, are proposed. They found that the results of this GA are better than a truncated branch and bound algorithm. The statistical analysis of the results is not sound. As an extension to this research, different crossover methods may be tried and their impact on the solution may be analyzed. Chong et al. [20] considered the simple assembly line balancing problem with deterministic task times (SALB-1) with realized cycle time which is the maximum of the loads of the stations. They developed a genetic algorithm with heuristic generated initial population for this problem and compared its performance with that of the genetic algorithm with a randomly generated initial population. They found that the method proposed in their paper performs well for some problems. But the statistical significance of the difference in the results is not presented. Yu and Yin [97] developed an adaptive genetic algorithm to determine the minimum number of workstations with workload balance between the workstations for a given cycle time. In this approach, the probability of crossover and that of mutation are dynamically adjusted according to the individual's fitness value. The individuals with higher fitness values are assigned to lower probabilities of genetic operators and vice versa. The dynamic adjustment of crossover probability based on the fitness function value seems to be a new idea. Since the authors did not compare their algorithm with any of the existing heuristics, its merit is not known. So a comparative analysis may be carried out to check the merit of this new idea.

Gao et al. [30] considered the simple assembly line balancing problem with the objective of minimizing the cycle time. This study presents a robotic assembly line balancing in which the assembly tasks are to be assigned to workstations and each workstation needs to select one of the available robots to process the assigned tasks with the objective of minimizing the cycle time. An innovative genetic algorithm hybridized with local search is proposed for this problem. In this GA, partial representation technique is used in which only a part of the decision information about a candidate solution is expressed in the chromosome and the rest is computed via a heuristic method. Its performance is tested on 32 problems and compared with other methods. The idea of taking partial information about a candidate solution in the chromosome and computing the rest via a heuristic minimizes the length of its chromosome, which greatly reduces the overall time to solve the problem. The authors compared the performance of this algorithm with that of existing algorithms and made visual observations, but statistical comparison is not made to find out the true strength of the proposed algorithm in terms of significant difference of its performance from that of other algorithms.

Kim et al. [49] developed a mathematical model to minimize the cycle time for a given number of mated stations of two-sided assembly line balancing problem. Further, they provided a neighborhood GA (n-GA) for the same problem in which 3×3 neighborhood is used for the localized evolution. At the end, they compared the performance of n-GA with that of a heuristic and an existing GA and found that their approach outperforms the other methods. The objective function may be extended by adding minimization of workload deviations and maximization of work relatedness subject to positional, synchronous, and task separation constraints. As an extension, the proposed algorithm may be applied to more practical two-sided assembly line balancing problem as explained above.

Goncalves and Almeida [37] developed a hybrid genetic algorithm for the type 1 assembly line balancing problem to maximize the balancing efficiency for a given cycle time. The chromosome representation of the problem in this approach is based on random keys. They used a heuristic priority rule to assign the operations to the workstations, in which the priorities of the operations are defined by the chromosome. A local search is then used to improve the solution. The authors compared their algorithm using literature problems. Since the authors propped a hybrid algorithm, it is better to conduct a full-scale factorial ANOVA experiment by creating general problems with problem size, cycle time, mean task times, etc. as factors to bring out its true strength. Wang et al. [92] developed a hybrid genetic algorithm for multi-objective product plan selection problem with assembly sequencing planning (ASP) and assembly line balancing (ALB). Assembly sequence planning means the planning of certain priority of assembly based on individual-specific assembly rule of thumb of the organizer with the consideration of related assembly constraints. They considered both assembly sequence planning and assembly line balance simultaneously to attain greater accuracy in terms of solution accuracy. The ASP problem contains four connectors, viz. combination, assembly tools, assembly directions, and precedence relationship. They established a multi-objective mathematical model for the selection of a product plan integrating ASP and ALB with considerations of cost, time, connector homogeneity, and number of workstations. Further, they developed guided-modified weighted Pareto-based multi-objective genetic algorithm (G-WPMOGA) to solve this problem. They concluded that G-WPMOGA performs better with four objective criteria. *They compared this algorithm with two existing algorithms by taking three problems. The number of objectives for the problems 1, 2, and 3 are 2, 3, and 4, respectively. They concluded that their algorithm performs better for the problems 1 and 2. A more comprehensive statistical comparison is required by taking more problems and a minimum number of replications (at least 2) in each problem to find the real strength of the proposed algorithm.*

Cheshmehgaz et al. [16] developed a cellular rearranging of population in genetic algorithms to solve the single-model assembly line balancing problem with deterministic task times. The objective of this research is to minimize the number of workstations for a given cycle time. Application of the genetic algorithms may end up with convergence of local optimum solution. In this research, a meta-heuristic called modified cellular (grid) rearranging-population structure is developed to overcome this drawback. This is used to locate the individuals of the population on cells according to the hammimg distance value among individuals as neighbors before regenerations. They developed a family of cellular genetic algorithms (CGAs) to minimize the number of workstations. They compared these algorithms and also a non-cellular genetic algorithm using graphical analysis and found that the cellular genetic algorithms perform better. As an extension to this research, the parameters of these genetic algorithms can be optimized using Taguchi's technique, namely, *S*/*N* ratio.

Simulated annealing algorithms Hong and Cho [42] considered the problem of generation of robotic assembly sequences with considerations of line balancing using a simulated annealing algorithm. They considered the single-model deterministic assembly line balancing problem and proposed a new simulated annealing algorithm, in which an energy function is derived in consideration of the satisfaction of assembly constraints and the minimization of assembly cost and the idle time. The energy function is iteratively minimized and occasionally perturbed by a simulated annealing until no further change in energy occurs to obtain a solution of line balancing. *The problem of this research comes under hybrid problem, which simultaneously considers assembly sequence generation and line balancing with the objectives of minimizing assembly cost and idle time. Future researches may be focused on hybrid problems like the one developed in this research. Though the effectiveness of this method is experimented with a case study, it may be tested using benchmark problems and compared with existing heuristics.* Narayanan and Panneerselvam [61] developed a new efficient set of heuristics (NESHU) to design the mass production system in an effective manner. The objective of this work is to group the tasks with deterministic times into minimum number of workstations for a given cycle time, such that the balancing efficiency is maximized. In their approach, the initial solution is generated using a heuristic for assembly line balancing (HAL) and composite weight factor, and then it is improved using global search heuristic, which is similar to simulated annealing algorithm. As an extension to this method, researchers may use some other efficient heuristic for generating initial solution. Baykasoglu [9] developed a simulated annealing algorithm for the straight and U-type simple assembly line balancing problems such that the smoothness index is maximized and the number of workstations is minimized. The proposed algorithm uses task assignment rules in constructing feasible solution. Future work may be carried out to incorporate many more task assignment rules in the algorithm and experiment for checking solution accuracy. Seyed-Alagheband et al. [81] considered the general assembly line balancing problem where the simple version is enriched by taking sequence-dependent setup times between tasks (GALBPS TYPE II problem). The objective is to find the minimum cycle time for a predetermined number of workstations. They developed a mathematical model and a novel simulated annealing (SA) algorithm to solve this NP-hard problem. Further, they optimized the parameters of the SA algorithm using signal-to-noise (*S*/*N*) ratio, which is a Taguchi's method. They used L_{18} orthogonal array in place of a full factorial experiment. They tested the performance of the algorithm using four literature problems in terms of cycle time objective and CPU time. Generally, the parameters of the simulated annealing algorithm are determined through experimentation using trial and error method. Instead, in this research, the authors used Taguchi's method for the same, which is an apt way of fixing the parameters of the simulated annealing algorithm. Future researchers are advised to follow this method to fine-tune the parameters, if they develop simulated annealing algorithm.

Roshani et al. [78] considered single-model two-sided assembely line balancing problem with deterministic task times. They developed a simulated annealing-based algorithm to minimize the total cost per product unit, which is the sum of labor cost, transportation cost, and equipment cost for implementing two-sided assembly line. The performance of this algorithm is compared with that of a mixed integer programming model applied to this problem for small-size problems and with that of a heuristic procedure, namely, First Fit Rule (FFR) for medium- and large-size problems using graphical analysis. It is found that the proposed algorithm gives optimal solution for most of the small-size problems and outperforms the heuristic for medium- and large-size problems. The parameters of the simulated annealing algorithm proposed in this research may be optimized using either simulation approach or Taguchi's technique for improved performance.

Tabu search algorithms Lapierre et al. [52] developed a new tabu search algorithm (TSA) for SALBP 1, which incorporates an intensification–diversification framework based on the neighborhood and a redefinition of the solution space and the objective function in order to allow the algorithm to visit infeasible solutions while searching for better solution. They tested their algorithm with a real industrial data set and reported the results. Though they claim that the proposed algorithm solves a real industrial data and gives better results for certain benchmark data sets, a comprehensive statistical analysis in terms of a complete factorial experiment is required to ascertain the strength of this algorithm.

Annarongsri and Limnararat [5] developed a hybrid tabu search method for the simple assembly line balancing problem with the objective of minimizing the number of workstations for a given cycle time. They combined the tabu search with the genetic algorithm to find the solution of this problem. They compared the proposed meta-heuristic with COMSOAL, which is a primitive method and reported that it performs well. It is obvious that any meta-heuristic will have superior performance over conventional algorithms. Hence, it may be compared with the recent meta-heuristics. Ozcan et al. [66] considered simple parallel assembly line balancing (PALB) problem with deterministic task times. They developed a tabu search algorithm for this problem such that the line efficiency is maximized and the variation of workload is minimized. They reported the results of the literature problems using tabu search algorithm and compared with the theoretical minimum number of workstations. Ozcan and Toklu [71] developed a tabu search algorithm for two-sided simple assembly line balancing problem with deterministic task times in which the line efficiency and smoothness index are considered as the performance measures. They found that this algorithm performs better than the other approaches. A better statistical comparison in terms of a complete factorial ANOVA may be carried out for this work at least by considering the problems P65, P148, and P205 reported by them to compare the algorithms Group Assignment Procedure (GAPR), ACO, LB, and the proposed TSA. The problem size, cycle time, and algorithms are to be treated as factors and one more replication in each of the experimental combinations is to be obtained for the analysis.

Ant colony optimization algorithms Baykasoglu and Dereli [10] considered the two-sided assembly line balancing problem with deterministic task times in which the objective is to minimize the number of stations for a given cycle time and where possible to maximize the work relatedness (relatedness is an index which represents the degree of assignment of related tasks to the same workstations). They developed an ant-colony-based algorithm to solve the problem. The problem may be solved with multi-objectives by taking into account several other criteria such as load balancing and smoothing. As an extension, a parameter analysis may be carried out for comparing the performance of several algorithms from different perspectives. Lai and Liu [51] developed an evolutionary algorithm based on ant colony optimization technique for the simple assembly line balancing problem to minimize the cycle time for a given number of workstations. Fattahi et al. [27] developed a mathematical model and ant colony algorithm for this simple assembly line balancing problem with deterministic times in which the objective is to determine the optimal number of workstations, while the total number of used workers and total idle time are same as the optimal solution of the simple assembly line configuration for a given cycle time. This may be extended with multi-objective optimization problem with load balancing and smoothing. They used a variable to represent temperature, which is same as that used in simulated annealing algorithm along with a reduction parameter and a stopping criterion. This research is the best example of hybridizing the ACO algorithm using a part of simulated annealing algorithm. Similar approach may be followed in future researches. Chica et al. [17] considered the 1/3 variant of the time and space assembly line balancing problem with the objectives of minimizing the number of stations and the station area, given a fixed value for the cycle time. They developed two algorithms, viz. multi-ant-colony-system algorithm (MACS) and multi-objective random greedy search algorithm (MORGA), to solve this problem. The results of these algorithms were compared with that of non-dominated sorting genetic algorithm II (NSGA-II) using ten well-known problem instances and found that both the proposed algorithms show good performance. Future research may be carried out to include user preferences to guide the multi-objective search procedure in the direction of expected needs. Also, a local search algorithm may be used to improve the performance of these algorithms.

Time and space assembly line balancing problem presents weight variants depending on three optimization criteria: the number of stations, the cycle time, and the area of the stations. Within these variants, there are four multi-objective problems. Chica et al. [18] tackled one of them, which is the TSALBP 1/3. It consists of minimizing the number of stations and the station area, given a fixed value of the cycle time. They developed multi-objective ant algorithm to optimize the said multi-objective function. Using the real data of Nissan pathfinder engine, a solid empirical study is carried out to obtain the most useful solutions for the decision makers in six different Nissan scenarios around the world. Ozbakir et al. [64] considered the parallel assembly line balancing problem with bi-objectives of minimizing the idle time of workstations and maximizing the line efficiency. They proposed an ACO for balancing bi-criteria parallel assembly lines. The performance of the proposed meta-heuristic is compared with that of four other existing approaches from literature and reported that the proposed ACO algorithm is very competitive. They hybridized ACO algorithm using swarm intelligent based heuristic, which is considered to be a novel approach. Future researchers may try similar approaches to improve the performance of their algorithms. McMullen and Tarasewich [59] developed a modified ant colony optimization technique for the type I assembly line balancing problem in which multi-objectives are considered in terms of crew size, system utilization, the probability of jobs being completed within a certain time frame and system design costs. These objectives are addressed simultaneously. A more detailed comprehensive comparison may be carried out to test the superiority of this technique.

Shortest path algorithm Kao et al. [44] proposed a shortest route algorithm for the assembly line balancing problem with resource constraint such that the number of workstations for a given cycle time and the sum of the number of resource types required in different stations are minimized. This may be extended to solve type 2 assembly line balancing problems. This algorithm is designed to obtain the optimal solution. It is well known that the assembly line balancing problem comes under combinatorial category. So, solving large-size problems using this approach may not be feasible. This algorithm may be used as a benchmark algorithm, while benchmarking the performance of any heuristic using small and moderate-size problems.

Memetic algorithm Chica et al. [19] considered the time and space assembly line balancing problem with deterministic task times, which is a multi-objective problem. The objectives of this research are minimizing the number of workstations and total area of the workstations for a given cycle time. They developed three memetic algorithms based on evolutionary computation, ant colony optimization, and greedy random adaptive search procedure (GRASP) and tested their performance using nine well-known problem instances. From this experiment, they found that these algorithms perform better for these problems. This is the best approach for developing a set of efficient algorithms to solve this line balancing problem, because no single algorithm can give the best solution for each of the problems in a data set.

Bee algorithms Tapkan et al. [88] considered the single-model two-sided assembly line balancing problem with deterministic task times in which the objective is to minimize the number of workstations for a given cycle time. They developed two algorithms, viz. bee algorithm (BA) and artificial bee colony algorithm (ABA). Through comparison, it is found that there is no significant difference between them in terms of solutions. They did not compare these two algorithms with the best existing algorithm. As an extension to this work, swarm particle algorithm may be developed and compared with the proposed two algorithms.

Under the single-model deterministic straight-type (SM_D_S) assembly line balancing problem, different authors researched on the development of 0–1 programming model, goal programming model, fuzzy goal programming model, simple heuristics, genetic algorithms, global search heuristic, tabu search, hybrid tabu search, ant colony optimization algorithm, Petri net algorithm, simulated annealing algorithm, shortest path algorithm, memetic algorithm, and bee algorithms. Within this category of assembly line balancing problem, the variations are simple assembly line balancing problem, parallel assembly line balancing problem, two-sided assembly line balancing with work relatedness in type 1 and type 2 assembly line balancing problems to deal with single objective as well as multi-objective situations.

In any operations research software, its model solving capability is limited by the maximum number of variables and the maximum number of constraints that can be handled by that software. So it is essential to provide formulas to compute them in terms of the number of tasks of the assembly line balancing problem. None of the above researches except the model given by Sivasankaran and Shahabudeen [85] gives such formulas to estimate the number of constraints and the number of variables for a given number of tasks. It is observed that a specific crossover method is used in each of the proposed genetic algorithms. While performing crossover operation for each pair of chromosomes in a given genetic algorithm, the crossover method may be selected randomly from the available set of crossover methods. This idea is to derive the benefit of each crossover method toward improving the fitness function value. Future researchers may focus in this direction to improve the performance of the genetic algorithm of their choice.

In most of the cases, comparisons of new approaches with existing approaches are not carried out using extensive statistical analysis like design of experiment. Future researchers may focus on developing hybrid meta-heuristics to solve single as well as multi-objective assembly line balancing problems under the SM_D_S problem.

The summary of important literature under the SM_D_S problem for different algorithms is presented in Table 1. In this table, SALB-1 nested under SM_D_S problem is represented by SM_D_S-1, and SALB-2 nested under SM_D_S problem is represented by SM_D_S-2. From Table 1, it is clear that for the one-sided SALB-1 problem, either the ant colony optimization algorithm developed by Ozbakir et al. [64] or the memetic algorithm developed by Chica et al. [19] may be used. The simulated annealing algorithm developed by Seyed-Alagheband et al. [81] may be used for the one-sided SALB-2 problem. It is evident that there is no organized way of comparing the algorithms. If one develops an algorithm for a problem today, then it should be compared with the best algorithm reported in literature to establish its superiority in terms of performance. Some of the researchers compared their contributions with some primitive heuristics, which may not give the real strength of those algorithms. So in the future, all important meta-heuristics listed in Table 1 may be compared using a common benchmark data set to select the best among them in terms of solution accuracy, mainly to identify a benchmark algorithm.

2.1.2 Single-model deterministic U-type (SM_D_U) problem

In this section, a review of literature of the single-model deterministic U-type assembly line balancing problem is presented. This problem considers single model which is to be assembled in the assembly line in which the workstations are arranged in U form. The execution time of each task of the single model (product) that is assembled in the assembly line is deterministic. The literature under this category is further classified based on the methods used to solve the problem and the corresponding sub-classification is as listed below:

- Mathematical models
- Heuristic
- Simulated annealing algorithm
- ACO algorithms
- Shortest path algorithm
- CPM method
- Imperialistic competitive algorithm

Mathematical models Gokcen and Agpak [34] developed a goal programming model to the simple U-line balancing problem to optimize different objectives by taking assignment constraints, cycle time constraints, workstation constraints, and task load constraints. The goals are with respect to the number of workstations, cycle time, and the number of tasks per workstations. They illustrated the application of the model using numerical examples. One should note that in goal programming, the objective function will be constructed using the deviational variables of the constraints of the goals with stated priorities. Widyadana [93] developed a multi-objective model with two goals for balancing the simple U-type assembly line with permanent and temporary workers such that the goals, viz. the number of maximal tasks in one workstation for a permanent or a temporary worker, the number of cycle time in each workstation, and the number of temporary workers, are optimized within the range permitted. They compared the solution of the straight line and the U-line using ten different cases generated by randomizing the task times of temporary workers for the example problem reported in this research and showed that the U-line gives better results. In future research, one can include operator walking time in this problem.

Heuristic Yegul et al. [95] considered the U-shaped two-sided simple assembly line balancing problem in which the objective is to find the special design (U-shaped and two-sided) such that the total number of stations is minimized for a given cycle time using a new algorithm. One side of the line is arranged in U-shape allowing stations with crossover, and the other side of the line is balanced like a traditional straight flow. They proposed a multi-pass random assignment algorithm to find the minimum number of stations required to

Table 1 Summary of important literature under SM_D_S-type problem

Method	Literature	ALB type T-1/T-2	Computational effort	Solution performance	One-sided/two-sided (OS/TS)	Goals						
						No. of WS	No. of mated WS	Idle time	Cycle time	Work load	Cost	Area
Math model	[25]	T-1	High	Optimal	OS	X		X	X		X	
	[72]	T-1	High	Optimal	TS	X	X		X	X		
	[85]	T-2	High	Optimal	OS	X			X			
Petri nets	[46]	T-1	Medium	Near optimal	OS	X		X				
	[45]	T-2	Medium	Near optimal	OS				X	X		
Heuristics	[74]	T-1	Less	Near optimal	OS	X						
	[33]	T-2	Less	Near optimal	OS				X			
	[62]	T-2	Less	Near optimal	OS			X	X			
Genetic algorithm	[48]	T-1/T-2	Medium	Near optimal	OS	X			X	X		
	[76]	T-1/T-2	Medium	Near optimal	OS	X			X			
	[80]	T-1	Medium	Near optimal	OS	X						
	[97]	T-1	Medium	Near optimal	OS	X				X		
	[30]	T-2	Medium	Near optimal	OS			X				
	[49]	T-2	Medium	Near optimal	TS			X				
	[37]	T-1	Medium	Near optimal	OS	X						
	[16]	T-1	Medium	Very near optimal	OS	X						
Simulated annealing	[42]	T-1	Medium	Near optimal	OS	X		X			X	
	[61]	T-1	Medium	Near optimal	OS	X						
	[9]	T-1	Medium	Near optimal	OS	X				X		
	[81]	T-2	Medium	Very near optimal	OS				X			
Tabu search	[66]	T-1	Medium	Near optimal	OS	X				X		
	[71]	T-1	Medium	Very near optimal	OS	X				X		
ACO	[17]	T-1	Medium	Near optimal	OS	X						X
	[64]	T-1	Medium	Very near optimal	OS	X		X				
Shortest path algorithm	[44]	T-1	High	Optimal	OS	X						
Memetic algorithm	[19]	T-1	Medium	Very near optimal	OS	X						X
Bee algorithm	[88]	T-1	Medium	Near optimal	OS	X						

T-1 SM_D_S-1 problem, *T-2* SM_D_S-2 problem, *WS* workstations.

assemble a product. They compared the performance of their algorithm with other existing algorithms using limited literature problems and concluded that it performs well. But there is no statistical evidence to support their claim. Since only one research is carried out on the development of heuristic for the SM_D_U-type assembly line balancing problem, further researchers may focus to develop many more efficient heuristics for this problem.

Simulated annealing algorithm Ozcan and Toklu [70] developed a hybrid improvement heuristic to the simple straight and U-type assembly line balancing problem. The heuristic is based on the idea of adaptive learning approach and simulated annealing to maximize the balancing efficiency and minimize the variation of the workloads among workstations. Determination of better learning strategies or better priority rules to incorporate in this algorithm may be treated as an extension to this research. The adaptive learning approach that is used in this algorithm may be used in other meta-heuristics.

Because of limited research contribution on the development of simulated annealing algorithm applied to the SM_D_U-type assembly line balancing problem, more researches may be carried out in this direction.

ACO algorithms Zhang et al. [100] proposed a new design of ant colony optimization for solving the simple U-shaped assembly line balancing problem in which the number of stations is minimized. The proposed algorithm uses the trial information which is deposited between the task and the task selected position and adapted pheromone summation rules. This ant colony optimization algorithm may be augmented with meta-heuristics to design a hybrid algorithm. Cuoglu et al. [22] considered the single-model U-type assembly line balancing problem with the objective of minimizing the number of stations for a given cycle time. They developed an ant-colony algorithm to solve this problem. They compared this algorithm with a simulated annealing algorithm and found that this ACO outperforms the simulated annealing algorithm. Also, they benchmarked this algorithm with the state-of-the-art ULINO algorithm and found that this algorithm is competitive. As an extension to this algorithm, some other meta-heuristic may be augmented with this ACO algorithm for improved performance. The parameters of this algorithm may be optimized using Taguchi's technique so that the solution reaches ultimate value. Baykasoglu and Dereli [11] developed an ant-colony-based algorithm for the simple U-type assembly line balancing problem to minimize the number of workstations for a given cycle time. The proposed method integrates COMSOAL algorithm and Rank Positional Weight heuristic with ACO-based algorithm. Since COMSOAL algorithm and rank positional weight heuristic, which are integrated with the ACO algorithm, are primitive heuristics, better heuristics may be integrated with ACO-based algorithm in future researches.

Since few researches are carried out on the development of ACO algorithm for the SM_D_U-type assembly line balancing problem, more work may be carried out in this direction.

Shortest path algorithm Gokcen et al. [35] modeled the simple U-type assembly line balancing problem with deterministic task times as a shortest route model based on the shortest route model developed by Gutjahr and Nemhauser [38] to minimize the number of workstations for a given cycle time. This shortest route model can be used as a framework to develop efficient heuristic to solve simple U-type assembly line balancing problem.

Critical path method Avikal et al. [6] developed a heuristic-based critical path method for U-shaped assembly line balancing problem to improve labor productivity in terms of reduced number of workstations. In this research, first the problem is solved by assuming the network as a project network using CPM method to identify the set of critical and non-critical activities. Then new temporary workstations are created by assigning the tasks by giving priority for the critical activities. Then a temporary workstation will be converted into a permanent workstation, if its slack time is very minimal. This process continues until all the tasks are assigned to some workstations. They did not compare this method with any of the best existing method. It is a graph-based method. So it can be used as a local search heuristic in ant colony optimization algorithm, simulated annealing algorithm, etc. to improve their performance.

Imperialistic competitive algorithm Nourmohammadi et al. [63] considered the single-model U-shaped assembly line balancing problem with deterministic task times. The objectives of this research are minimizing the number of workstations and the variations in workload of the workstations. They developed an imperialistic competitive algorithm (ICA) to solve this problem. This algorithm is based on socio-political evolution. They compared this algorithm with genetic algorithm (GA) and found that ICA performs better than GA. The parameters, namely, assimilation rate (AR) affects the solution quality significantly. Hence, some new schemes for this parameter may be developed to further improve the solution of ICA. The parameters of this algorithm may be optimized using Taguchi's method.

Under the single-model deterministic U-type (SM_D_U) assembly line balancing problem, the authors developed shortest route formulation, goal programming model, simple heuristic, ant colony optimization algorithm, critical path method, imperialistic competitive algorithm, and hybrid heuristic. The two-sided assembly line balancing problem is also attempted in this category. In most of the researches, multiple objectives are used. Because this problem comes under combinatorial category, heuristics are used to obtain near-optimal solution for any large-size problem. While benchmarking the heuristics in terms of their closeness toward optimal solution, the use of goal-programming modes is highly inevitable. So future research may be carried out in this direction. The development of any new algorithm is to be compared with existing best algorithm using a complete factorial experiment instead of a simple comparison.

The summary of the literature of the SM_D_U problem is presented in Table 2. In this table, SALB-1 nested under SM_D_U problem is represented by SM_D_U-1, and SALB-2 nested under SM_D_U problem is represented by SM_D_U-2. From Table 2, it is clear that either ant colony optimization algorithm developed by Cuoglu et al. [22] or imperialistic competitive algorithm developed by Nourmohammadi et al. [63] may be used to solve the one-sided SALB-1 problem. It is observed that no heuristic/meta-heuristic is developed for the SALB-2 problem. Hence, future researches may be focused in this direction.

Table 2 Summary of important literature under SM_D_U problem

Method	Literature	SM_D_U-1/ SM_D_U-2	Computational effort	Solution performance	One-sided/two-sided (OS/TS)	Goals		
						No. of WS	Cycle time	Work load
Model	[34]	SM_D_U-1/ SM_D_U-2	High	Optimal	OS	X	X	X
	[93]	SM_D_U-2	High	Optimal	OS		X	X
Heuristic	[95]	SM_D_U-1	Less	Near optimal	TS	X		
Simulated annealing	[70]	SM_D_U-1	Medium	Near optimal	OS	X		X
Ant colony algorithm	[100]	SM_D_U-1	Medium	Near optimal	OS	X		
	[22]	SM_D_U-1	Medium	Very near optimal	OS	X		
	[11]	SM_D_U-1	Medium	Near optimal	OS	X		
Shortest path algorithm	[35]	SM_D_U-1	High	Optimal	OS	X		
Critical path method	[6]	SM_D_U-1	Medium	Near optimal	OS	X		
Imperialistic competitive algorithm	[63]	SM_D_U-1	Medium	Very near optimal	OS	X		X

2.2 Single-model probabilistic ALB problems

This section presents a review of literature of single-model probabilistic assembly line balancing problems. This type of problems deals with assembly of only one model in a line, in which the task times are probabilistic. These problems are further classified into two types based on the type of layout used to assembly the model, viz. straight-type layout and U-type layout. Hence, the sub-classification under this category is as listed below:

- Single-model probabilistic straight-type (SM_P_S) problem
- Single-model probabilistic U-type (SM_P_U) problem

2.2.1 Single-model probabilistic straight-type (SM_P_S) problem

This section presents a review of literature of the single-model probabilistic straight-type (SM_P_S) problem. This problem considers single model which is to be assembled in the assembly line in which the workstations are arranged in a straight-line form. The execution time of each task of the single model (product) that is assembled in the assembly line is probabilistic. Based on the methods used to solve the problem, the literature under this category is further classified as given below:

- Mathematical models
- Heuristics
- Genetic algorithms
- Simulated annealing algorithms
- Shortest path algorithm
- Particle swarm optimization algorithm

Mathematical model Agpak and Gokcen [2] developed chance-constrained 0–1 integer programming models for the probabilistic traditional (straight-type) and U-type line balancing problems to minimize the number of workstations. Several test problems were solved using these models. A goal programming approach is also presented for these problems. These models may serve as validation tools for the heuristics developed to solve the SM_P_S type assembly line balancing problem. In this research, the task times are assumed to follow normal distribution. Future research may be directed to identify the probable distributions that are used for the task time and they may be introduced in these models.

Hazir and Dolgui [41] developed an exact algorithm for the single-model assembly line balancing problem with probabilistic task times. The objective is to minimize the cycle time for a given number of workstations. In this research, a mathematical model is developed and then a robust algorithm based on decomposition is developed to solve the problem optimally. This algorithm can form a part of a decision support system for the assembly line balancing problem. They did not compare the proposed algorithm with the best existing algorithm in terms of CPU time. As an extension to this research, quantitative robustness metrics may be formulated and their efficiency and effectiveness may be tested using a simulation.

Heuristics Liu et al. [55] designed a bidirectional heuristic for the probabilistic assembly line balancing problem under SALB-2, in which the objective is to minimize the cycle time for a given number of workstations and pre-specified assembly reliability, which is the probability of the work load not exceeding the cycle time for the whole assembly line. In this work, the tasks are assigned to workstations in forward directions, alternatively. In the second stage, the tasks are swapped among workstations until the cycle time is reduced such that the pre-specified assembly line reliability is satisfied. This algorithm is compared with the modified version of Moodie and Young's algorithm and found that the proposed heuristic is more efficient. Instead of comparing the heuristic with primitive heuristic, it may be compared with recent efficient heuristic to prove its solution accuracy. In future research, for this problem with large coefficient of variation of task time, the use of unpaced assembly line system can be considered. Gamberini et al. [29] considered the assembly re-balancing problem with probabilistic task times. They developed a new heuristic based on the integration of a multi-attribute decision-making procedure, named "Technique for order preference by similarity to ideal solution" (TOPSIS) and the well-known heuristic approach of Kottas and Lau [50]. The frequent changes in products features and sales volume (mutable scenarios) are considered in this research. The algorithm deals with the assembly line (re)balancing problem by considering the minimization of two performance criteria, viz. unit labor and expected unit incompletion costs, and tasks re-assignments. The later objective addresses the problem of keeping a high degree of similarity between previous and new balancing, in order to avoid costs related to tasks movement, operators training, product quality assurance, equipment installation and moving. This work is a valuable starting point for future studies centered on solving assembly line balancing problem in which mutable sales scenarios constitutes an important factor.

Genetic algorithms Tsujimura et al. [91] developed a genetic algorithm to design the single-model assembly line balancing problem with fuzzy processing times to minimize the balancing delay and the number of workstations. They used a repair algorithm to rearrange the chromosomes to maintain the precedence constraints among the tasks. They demonstrated the results of the algorithm using a numerical example. But they did not compare the performance of this algorithm with that of any of the existing algorithms. Gen et al. [32] considered the assembly line balancing problem with fuzzy processing times. The objectives of this research are to minimize the number of workstations and total idle time under precedence constraints with fuzzy processing times of jobs. They used genetic algorithms to solve this problem. For infeasible chromosomes that do not satisfy with precedence constraints, repairing techniques are used. The authors did not compare the performance of their method with that of the best existing method. Zacharia and Nearchou [98] considered the assembly line balancing problem of type 2 with fuzzy job processing times. The job processing times are formulated by triangular fuzzy membership function. They developed a new multi-objective genetic algorithm for this problem. The total fuzzy cost function is formulated as the weighted sum of two bi-criteria fuzzy objectives, viz. minimizing fuzzy cycle time and fuzzy smoothness index of work load of the line and minimizing fuzzy cycle time of the line and fuzzy balance delay time of workstations. This work may be extended for multi-model assembly line balancing problem with fuzzy processing times. They carried out an extensive experimentation, reported the results, and made inferences only through graphs.

Zacharia and Nearchou [99] considered the assembly line balancing problem with fuzzy task times. They developed a meta-heuristic to find a combination of number of workstations and cycle time such that the balancing efficiency is maximized. The meta-heuristic is based on a genetic algorithm. As an extension to this algorithm, a fast local search procedure like variable neighborhood search (VNS) can be incorporated to improve the performance of the proposed algorithm.

Simulated annealing algorithms Ozcan [65] considered the single-model assembly line balancing problem with probabilistic times for tasks and two-sided assembly line. In the first phase, the author developed a chance constrained piecewise linear mixed integer programming model (CPMIP) to minimize the primary measure of the number of mated stations with respect to some constraints and the secondary measure of the number of stations. They developed a simulated annealing algorithm and a heuristic based on COMSOAL algorithm. Finally, the solutions of these algorithms are compared with that of CPMIP model using a set of test problems and found that these algorithms perform well. Since the mathematical model can solve only small- or moderate-size problems, the performance of the algorithms proposed in this research should be compared with that of any other meta-heuristic for large-size problems. Cakir et al. [15] developed a hybrid simulated annealing algorithm for multi-objective optimization of a probabilistic assembly line balancing with parallel stations. The objectives are the minimization of the smoothing index and the design cost. They proposed m-SAA based on simulated annealing algorithm which implements a multimodal probability mass function approach, tabu list, two repair algorithms, and a diversification strategy. They compared the results of this approach with that of another simulated annealing algorithm using a weight-sum approach for 24 test problems and found that their approach is more effective in terms of the quality of Pareto-optimal solutions. The complexity of this algorithm increases by the problem size and the number of objectives. Hence, an interactive solution procedure to guide the direction of search for Pareto-optimal solutions in only potential areas of solution space may be developed for the simulated annealing algorithm, which will lead to steep reduction in CPU time.

Shortest path algorithm Boysen and Fliedner [13] proposed a versatile assembly line balancing algorithm which is able to solve instances of the single-model assembly line balancing problem (SMALBP) as well as several generalized assembly line balancing problems (GALBP) with relevant constraints such as parallel workstations and tasks, cost strategies, processing alternatives, zoning constraints, probabilistic processing times, or U-shaped assembly lines. They decomposed the original problem into two stages which can be solved independently. In the first stage, the precedence network is utilized in order to construct a valid sequence of tasks. This sequence is hence processed in the second stage, as the tasks are assigned to workstations by solving a shortest-path problem. They demonstrated the application of this algorithm using the single-model probabilistic straight-type problem as well as the single-model probabilistic U-type problem. As an extension to this research, efforts may be made to derive bounds for GALBP instances, since so far very little effort is made in this direction.

Particle swarm optimization algorithm Hamta et al. [39] developed a hybrid particle swarm optimization algorithm (H-PSO) with variable neighborhood search (VNS) as its local search algorithm for the single-model assembly line balancing problem with the objective of minimizing the cycle time, total equipment cost, and smoothness index. In this research, the task time is dependent on worker(s)/machine(s) learning. Further, the task time is sequence-dependent. They compared this algorithm with an existing evolutionary algorithm using a graphical analysis and found that the proposed algorithm performs better. As an extension to this research, a branch and bound algorithm may be developed to obtain optimal solutions for small- and moderate-size problems, so that it can act as a benchmark algorithm in future. The parameters of H-PSO may be optimally set using Taguchi's technique.

From the literature on the SM_P_S assembly line balancing problem, it is observed that only limited researches are carried out. Different authors developed mathematical models, heuristics, genetic algorithms, simulated annealing algorithms, shortest path algorithm, and particle swarm optimization algorithm for this problem. As a future extension to the simulated annealing, ACO algorithm, etc., the parameters may be optimized using Taguchi's method, instead of fixing them by trial and error method. Future research may be directed to develop hybrid algorithms for this problem.

The summary of the literature of the SM_P_S problem is given in Table 3. In this table, SALB-1 nested under SM_P_S problem is represented by SM_P_S-1, and SALB-2 nested under SM_P_S problem is represented by SM_P_S-2. Based on the summary of this table, it is recommended to use the simulated annealing algorithm developed by Cakir et al. [15] to solve the one-sided SALB-1 problem. The genetic algorithm developed by Zacharia and Nearchou [99] may be used to solve the one-sided SALB-2 problem, because it considers the SALB-1 problem simultaneously. The simulated annealing algorithm developed by Ozcan [65] may be used to solve the two-sided SALB-1 problem.

Table 3 Summary of literature of SM_P_S problem

Method	Literature	ALB type T-1/T-2	Computational effort	Solution performance	One-sided/two-sided (OS/TS)	Goals				
						No. of WS	No. of mated WS	Idle time	Cycle time	Work load
Model	[2]	T-1	High	Optimal	OS	X				
	[41]	T-2	High	Optimal	OS				X	
Heuristic	[55]	T-2	Less	Far less from optimality	OS				X	
Genetic algorithm	[91]	T-1	Medium	Near optimal	OS	X				X
	[32]	T-1	Medium	Near optimal	OS	X		X		
	[98]	T-2	Medium	Near optimal	OS				X	X
	[99]	T-1 T-2	Medium	Near optimal	OS	X			X	
Simulated annealing	[65]	T-1	Medium	Near optimal	TS	X	X			
	[15]	T-1	Medium	Very Near optimal	OS	X				X
Shortest path algorithm	[13]	T-1	Medium	Near optimal	OS	X				
PSO algorithm	[39]	T-1	Medium	Near optimal	OS				X	X

T-1 SM_P_S-1 problem, *T-2* SM_P_S-2 problem, *WS* workstations.

2.2.2 Single-model probabilistic U-type (SM_P_U) problem

In this section, a review of literature of the single-model probabilistic U-type (SM_P_U) problem is presented. This problem considers single model which is to be assembled in the assembly line in which the workstations are arranged in U form. The execution time of each task of the single model (product) that is assembled in the assembly line is probabilistic. The contributions of this problem are presented under genetic algorithm, tabu search algorithm and imperialistic competitive algorithm.

Genetic algorithm Baykasoglu and Ozbakir [12] considered the probabilistic U-line balancing problem with the objective of minimizing the number of stations for a given cycle time. They developed a multiple-rule-based genetic algorithm (MRGA-SUALB) for this problem. The proposed algorithm integrates COMSOAL method, task assignment rules, and genetic algorithm. They compared the results of their algorithm with the optimal solutions and found that it gives optimal solution for all the problems except one problem with reduced CPU time.

Tabu search algorithm Erel et al. [26] considered the probabilistic assembly line balancing problem in U-lines using beam search, which is a special type of tabu search algorithm. This method minimizes total expected cost, which consists of total labor cost and total expected incompletion cost for a given cycle time. They assumed three levels for the cycle time. Since it is of its first kind, they did experiments with benchmark problems and reported their results. In this study, the task times are independent of the skills of the workers. So this assumption may be relaxed by assuming dependency of the task times on the skills of the workers, and accordingly, the performance of the algorithm may be checked. Further, the incomplete tasks may be completed off the line to speed up the assembly process.

Imperialistic competitive algorithm Bagher et al. [7] considered the probabilistic U-type simple assembly line and they developed an imperialistic competitive algorithm to form the U-type assembly line with the aim of minimizing the number of workstations, idle time at each station, and non-completion probabilities of each station. The proposed algorithm is a combination of COMSOAL, task assignment rules heuristic and newly introduced imperialistic competitive algorithm (ICA). ICA is inspired from socio-political evolution processes. The design parameters of the algorithm are optimized by means of Taguchi method. It was found that this algorithm performs better than the best algorithm proposed before. As an extension to this research, multiple objectives may be tried.

In the single-model probabilistic U-type (SM_P_U) assembly line balancing problem, genetic algorithm, cost oriented heuristic and imperialistic competitive algorithm (ICA) are developed. The meta-heuristics such as simulated annealing algorithm and ACO algorithm may be developed for this problem. In all the researches of this problem, the task times are assumed to follow normal distribution. But, in reality, they may follow some non-standard distributions. Hence, an attempt may be made to incorporate such distributions while sampling task time. The summary of the literature of the SM_P_U problem is shown in Table. 4. In this table, SALB-1 nested under SM_P_U problem is represented by SM_P_U-1, and SALB-2 nested under SM_P_U problem is represented by SM_P_U-2. Based on the summary in this table, it is recommended to use the imperialistic competitive algorithm presented by Bagher et al. [7] to solve the one-sided SALB-1 problem. Since no work is carried out for the SALB-2 problem, future research may be carried out to develop meta-heuristics for this problem.

2.3 Multi-model deterministic ALB problems

This section presents a review of literature of multi-model deterministic assembly line balancing problems. This type of problems deals with assembly of many models in the same line, in which the task times of the models are deterministic. These problems are further classified into two types based on the type of layout used to assembly the model, viz. straight-type layout and U-type layout. Hence, the classification under this category is as listed below:

Table 4 Summary of literature of SM_P_U problem

Method	Literature	*SM_P_U*-1/ *SM_P_U-2*	Computational effort	Solution performance	One-sided/two-sided (OS/TS)	Goals	
						No. of workstations	Labor cost
Genetic algorithm	[12]	*SM_P_U-1*	Medium	Near optimal	OS	X	
Tabu search algorithm	[26]	*SM_P_U-1*	Medium	Near optimal	OS	X	X
Imperialistic competitive algorithm	[7]	*SM_P_U-1*	Medium	Very near optimal	OS	X	

- Multi-model deterministic straight-type (MM_D_S) problem
- Multi-model deterministic U-type (MM_D_U) problem

2.3.1 Multi-model deterministic straight-type (MM_D_S) problem

The multi-model deterministic straight-type (MM_D_S) assembly line balancing problem is relatively complex when compared to the SM_D_S problem. This problem considers multiple models which are to be assembled in the assembly line in which the workstations are arranged in a straight-line form. The execution time of each task of each of the models (products) that are assembled in the assembly line is deterministic. This section discusses a review of literature of the multi-model deterministic straight-type (MM_D_S) problem. The sub-classification of this category is as listed below:

- Mathematical model
- Genetic algorithms
- Simulated annealing algorithms
- Tabu search
- ACO algorithms

Mathematical model Gokcen and Erel [36] 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. In this method, the number of stations is same for all models and parallel stations are not allowed. This model may be used as a validation tool for heuristics. The flexibility ratio (*F*) presented in this work helps researchers to know the required size of primary memory and secondary memory of computer to solve a problem. While comparing the performance of any heuristic with that of this model, this ratio will be useful for researchers to generate problems within the permitted problem size.

Genetic algorithms Haq et al. [40] developed a hybrid genetic algorithm to the mixed-model assembly line balancing problem in which the objective is to minimize the number of workstations for a given cycle time. They used a modified ranked positional weight method to generate the initial solution for the genetic algorithm. They compared the performance of the hybrid genetic algorithm with that of a traditional genetic algorithm and reported the results in terms of number of workstations and CPU time. A detailed statistical analysis (ANOVA) will help to discriminate the power of the proposed algorithm from that of the traditional genetic algorithm.

Bai et al. [8] considered the mixed-model assembly line balancing problem for which, they proposed a mathematical model based on two factors integrated including the workstation number and the assembly line efficiency. Then a new hybrid genetic algorithm is developed for finding the optimal solution of the problems. In this research, to prevent the premature convergence of solution and enhance global optimization capability, a genetic algorithm is combined with a simulated annealing algorithm. This research sets an example of combining two meta-heuristics to improve the performance of the proposed hybrid genetic algorithm. Future researchers can make similar attempts, which will lead to establishment of new competitive algorithms to solve the assembly line balancing problems.

Simaria and Vilarinho [82] considered the mixed model assembly line balancing problem (MALBP) with parallel workstations in which the objective is to design the assembly line to maximize the production rate of the line (minimize the cycle time) for a pre-determined number of operators. Finally, they considered zoning constraints and workload balancing. They developed a genetic algorithm to solve the problem. Su and Lu [87] considered the mixed-model assembly line balancing problem in which the objective is to design the assembly line to smooth workload balance within each workstation. They developed a genetic algorithm to find the sequence of models which will minimize the cycle time. They carried out a simulation experiment. This algorithm consists of two phases. In the first phase, the best feasible solution is obtained by assuming an upper limit for the cycle time and then it is decreased by 1 continuously until there is no feasible solution found. In the second phase, the best feasible solution is fine-tuned such that the load balancing among the workstations is maximized. This is an innovative iterative approach built in the genetic algorithm. In this direction, many more variations of genetic algorithms may be tried by future researchers.

Mamun et al. [56] developed a genetic algorithm to solve mixed model assembly line balancing problem with the objective of minimizing the number of workstations. The task times are assumed to be deterministic. Also, they developed a local search heuristic to avoid convergence of optimum solution. This algorithm is applied to a company manufacturing plastic bag and its results are reported. They did not compare the performance of this algorithm with that of the best existing algorithm.

Sivasankaran and Shahabudeen [84] developed a genetic algorithm for concurrent balancing of mixed-model assembly lines with original task times of the models. The objective is to minimize the number of workstations, which, in turn, amounts maximizing balancing efficiency. Mostly in previous researches, average time is assumed for each task while forming workstations. Instead of this approach, in this research, the original task times are used while constructing the workstations for the mixed models. The superiorly of this approach over existing practice in previous researches is proved using an example problem. *As an extension to this research, different types of crossover methods may be proposed and compared for their effects over solution accuracy.*

Simulated annealing algorithm Fattahi and Salehi [28] considered the mixed-model assembly line balancing problem to minimize the total utility and idle costs with variable launching interval for sequencing the models with deterministic task times. They aimed to determine the cycle time, the number, and the sequence of stations, to balance the assembly line, and also to determine the sequence of products in the assembly line with variable launching interval between the products. They developed a mathematical model as well as a simulated annealing for this problem. If the operator in a station does not complete all the tasks in it, then rest will be done by a utility worker. They compared the performance of the simulated annealing algorithm with that of existing algorithms using a percentage analysis, which will not discriminate the power of the proposed algorithm from that of other algorithms. As an extension to this work, a mathematical model with multi-objective may be developed. Ozcan et al. [67] considered the balancing and sequencing of parallel mixed assembly lines. They developed a simulated annealing algorithm for this problem to minimize the number of workstations and attain equalization of workstations among workstations. They reported a percentage analysis of comparison of their proposed algorithm with existing algorithms, which will not discriminate its power from other algorithms. So a detailed statistical analysis is needed in this research.

Tabu search algorithm Ozcan et al. [68] considered the problem of two or more two-sided assembly lines located in parallel to each other to assemble one or more product lines. They proposed a tabu search algorithm which combines the advantages of both types of production lines. In this work, each two-sided assembly line may be having different cycle times. So a common cycle time should be used which is based on the least common multiple (LCM) of the cycle times for different cycle time situations. They did not compare the performance of the proposed algorithm with that of any other work.

ACO algorithm Simaria and Vilarinho [83] considered the two-sided mixed model assembly line balancing problem with the objective of minimizing the number of workstations for a given cycle time. They presented a mathematical model to describe the problem. Then an ant colony optimization algorithm is proposed to solve the problem. In this algorithm, two ants are introduced simultaneously, one at each side of the line to build a balancing solution which verifies the precedence, zoning, capacity, side, and synchronization constraints of the assembly process. Hence, this algorithm is called 2-ANTBAL. In this research, they considered many constraints of the reality. So future researches may be focused in this direction.

Akpinar et al. [4] considered the mixed-model assembly line balancing problem with sequence-dependent setup times between tasks. In this research, parallel stations and zoning constraints are considered. The objective is to minimize the

Table 5 Summary of literature of MM_D_S problem

Method	Literature	ALB type T-1/T-2	Computational effort	Solution performance	One-sided/two-sided (OS/TS)	Goals		
						No. of WS	Cycle time	Workload
Model	[36]	T-1	High	Optimal	OS	X		
Genetic algorithm	[40]	T-1	Medium	Near optimal	OS	X		
	[8]	T-1	Medium	Very near optimal	OS	X		
	[82]	T-2	Medium	Near optimal	OS		X	X
	[87]	T-2	Medium	Very near optimal	OS		X	X
	[56]	T-1	Medium	No comparison done		X		
	[84]	T-1	Medium	Near optimal	OS	X		
Simulated annealing algorithm	[28]	T-2	Medium	Near optimal	OS		X	
	[67]	T-1	Medium	Near optimal	OS	X		X
Tabu search algorithm	[68]	T-1	Medium	No comparison done	TS	X		
ACO algorithm	[83]	T-1	Medium	Near optimal	TS	X		
	[4]	T-1	medium	Very near optimal	OS	X		

T-1 MM_D_S-1 problem, *T-2* MM_D_S-2 problem, *WS* workstations.

number of workstations. They developed a hybrid ant colony optimization (H-ACO) algorithm in which genetic algorithm is used as a local search heuristic. The ant colony optimization algorithm provides diversification and the genetic algorithm provides intensification. They compared the hybrid ACO with ACO and GA using 20 test problems covering all sizes and found that the hybrid ACO performs better. As an extension to this research, Taguchi's method may be used to optimize the parameters of the hybrid ACO to further enhance its performance.

Under the multi-model deterministic straight-type (MM_D_S) problem, various authors researched the applications of mathematical model, genetic algorithms, hybrid genetic algorithms, simulated annealing algorithm, tabu search algorithm, and ant colony optimization algorithm for this ALB problem. The two-sided assembly and parallel assembly stations are also attempted in this category. In future researches, attempts may be made to optimize the parameters of meta-heuristics and hybrid algorithms using Taguchi's technique for better performance in terms of their solution accuracy.

The summary of the literature of the MM_D_S problem is presented in Table 5. *In this table*, *SALB-1 nested under MM_D_S problem is represented by MM_D_S-1*, *and SALB-2 nested under MM_D_S problem is represented by MM_D_S-2*. From this table, one can verify the fact that either the ACO algorithm developed by Akpinar et al. [4] or the genetic algorithm developed by Bai et al. [8] may be used to solve the one-sided SALB-1 problem. The ACO algorithm developed by Simaria and Vilarinho [83] may be used to solve the two-sided SALB-1 problem. Practitioners can use the genetic algorithm developed by Su and Lu [87] to solve the one-sided SALB-2 problem.

2.3.2 Multi-model deterministic U-type (MM_D_U) problem

The multi-model deterministic U-type (MM_D_U) assembly line balancing problem is relatively complex when compared to the SM_D_U problem. This problem considers multiple models which are to be assembled in the assembly line in which the workstations are arranged in U form. The execution time of each task of each of the models (products) that are assembled in the assembly line is deterministic. This section presents a review of literature of the multi-model deterministic U-type (MM_D_U) assembly line balancing problem. The literature on genetic algorithms of this problem is presented below.

Genetic algorithms (evolutionary algorithms) The integration of balancing and sequencing in mixed U-lines is a challenging problem. Kim et al. [47] developed an endosymbiotic evolutionary algorithm (EEA) for this problem. They aimed both balancing and sequencing problems in which workload smoothness is achieved. EEA provides a proper balance between parallel search with partial solution and integrated search with entire solution. A set of experiments is carried out to compare EEA with hierarchical genetic algorithm (HGA), separated and population-based symbiotic evolutionary algorithm (SPA), and separated and neighbor-based symbiotic evolutionary algorithm (SNA), and the results are reported. They compared the performance of the proposed EEA with that of the other algorithms using percentage analysis and graphical analysis and concluded that the proposed algorithm outperforms the other algorithms. An important strength of the evolutionary algorithm is that it can easily handle variations in objectives and constraints. So this endosymbiotic evolutionary algorithm (EEA) may be extended to other types of line balancing problem. Chutima and Olanviwatchai [21] considered the mixed-model U-shaped assembly line balancing problem with the objective of minimizing the number of workstations, work relatedness, and the variation of workload for a given cycle time. They developed a new evolutionary algorithm called combinatorial optimization with coincidence algorithm (COIN). The variances of COIN are also proposed, which are CN SGA II and COIN-MA. COIN, and these variances are tested against a well-known algorithm, namely, non-dominated sorting genetic algorithm II (NSGA II) and MNSGA II and found that COIN outperforms NSGA II. Since the algorithm COIN-MA outperforms the algorithm CNSGA II, it can be used to search for an optimal Pareto front for the MM_D_U-type assembly line balancing problem. In this research, the parameters of the algorithm are set by trial

Table 6 Summary of literature of MM_D_U problem

Method	Literature	MM_D_U-1/ MM_D_U-2	Computational effort	Solution performance	One-sided/two-sided (OS/ TS)	Goals	
						No. of WS	Work load
Genetic algorithm	[47]	MM_D_U-1	Medium	Vey near optimal	OS	X	X
	[21]	MM_D_U-1	Medium	Very Near optimal	OS	X	X

WS workstations.

Table 7 Summary of literature of MM_P_S problem

Method	Literature	MM_P_S-1/ MM_P_S-2	Computational effort	Solution performance	One-sided/two-sided (OS/TS)	Goals		
						No. of WS	Cycle time	Work load
Genetic algorithm	[94]	MM_P_S-1	Medium	Near optimal	OS	X		X
Simulated annealing algorithm	[57]	MM_P_S-2	Medium	Very Near optimal	OS		X	
ACO algorithm	[58]	MM_P_S-1	Medium	Very Near optimal	OS	X		

and error method. Instead, Taguchi's technique may be used to set optimal parameters for the algorithm before actually executing the steps of this algorithm.

Under this category of multi-model deterministic U-type (MM_D_U) assembly line balancing problem, endosymbiotic evolutionary algorithm and evolutionary algorithm called coincidence (COIN) algorithm are developed. In this category of work, only less work is carried out.

The summary of the literature of the MM_D_U problem is given in Table 6. *In this table, SALB-1 nested under MM_D_U problem is represented by MM_D_U-1, and SALB-2 nested under MM_D_U problem is represented by MM_D_U-2.* Based on the summary provided in this table, the practitioners are suggested to use any one of the genetic algorithms presented in this section to solve the one-sided SALB-1 problem. Future research may be directed to develop meta-heuristics for the SALB-2 problem, because there is no work reported for this problem.

2.4 Multi-model probabilistic ALB problems

This section presents a review of literature of multi-model probabilistic assembly line balancing problems. This type of problems deals with assembly of many models in the same line, in which the task times of the models are probabilistic. These problems are further classified into two types based on the type of layout used to assembly the model, viz. straight-type layout and U-type layout. Hence, the classification under this category is as listed below:

- Multi-model probabilistic straight-type (MM_P_S) problem
- Multi-model probabilistic U-type (MM_P_U) problem

2.4.1 Multi-model probabilistic straight-type (MM_P_S) problem

The multi-model probabilistic straight-type (MM_P_S) assembly line balancing problem is relatively complex when compared to the SM_P_S problem. This problem considers multiple models which are to be assembled in the assembly line in which the workstations are arranged in a straight-line form. The execution time of each task of each of the models (products) that are assembled in the assembly line is probabilistic. In this section, a literature review of the multi-model probabilistic straight-type (MM_P_S) assembly line balancing problem is presented. Based on the methods used to solve the problem, the literature under this category is further classified into genetic algorithm, simulated annealing algorithm, and ACO algorithms.

Genetic algorithm In the research carried out by Xu and Xiao [94], a special type of assembly line balancing problem (mixed models, fuzzy operations times, and drifting operations) with station lengths longer than the distance for which the conveyer moves within one cycle time is investigated in fuzzy environments, where operation times are assumed to be fuzzy variables. The objective is to minimize the total work overload time during the decision horizon. They proposed a fuzzy α total workload time minimization model. In order to

Table 8 Summary of literature of MM_P_U problem

Method	Literature	MM_P_U-1/ MM_P_U-2	Computational effort	Solution performance	One-sided/two-sided (OS/TS)	Goals	
						No. of WS	Cycle time
Genetic algorithm	[69]	MM_P_U-2	Medium	Near optimal	OS		X
ACO algorithm	[3]	MM_P_U-1	Medium	Near optimal	OS	X	

Table 9 Summary of ALB literature

Method	Reference	No. of objectives	Problem type							
			SM_D_S	SM_D_U	SM_P_S	SM_P_U	MM_D_S	MM_D_U	MM_P_S	MM_P_U
Mathematical models	[90]	1	X							
	[25]	5	X							
	[72]	3	X							
	[85]	1	X							
	[49]	1	X							
	[27]	1	X							
	[81]	1	X							
	[34]	3		X						
	[93]	3		X						
	[2]	1			X					
	[41]	1			X					
	[65]	2			X					
	[36]	1					X			
	[8]	2					X			
	[28]	5					X			
	[83]	2					X			
	[3]	1								X
Total: 17										
Petri net	[46]	1	X							
	[45]	1	X							
Total: 2										
Heuristics	[23]	1	X							
	[24]	1	X							
	[75]	3	X							
	[33]	2	X							
	[61]	1	X							
	[74]	1	X							
	[53]	2	X							
	[43]	2	X							
	[62]	2	X							
	[31]	2	X							
	[86]	1	X							
	[96]	1	X							
	[37]	2	X							
	[95]	1		X						
	[70]	2		X						
	[55]	1			X					
	[29]	2			X					
	[13]	2			X					
	[26]	2				X				
	[47]	1						X		
	[21]	3						X		
Total: 21										
Genetic algorithms	[79]	1	X							
	[48]	5	X							
	[80]	1	X							
	[37]	1	X							

Table 9 (continued)

Method	Reference	No. of objectives	Problem type							
			SM_D_S	SM_D_U	SM_P_S	SM_P_U	MM_D_S	MM_D_U	MM_P_S	MM_P_U
	[54]	2	X							
	[20]	1	X							
	[46]	2	X							
	[30]	1	X							
	[49]	1	X							
	[17]	2	X							
	[97]	2	X							
	[92]	4	X							
	[76]	3	X							
	[16]	1	X							
	[32]	2			X					
	[91]	1			X					
	[98]	2			X					
	[99]	2			X					
	[12]	1				X				
	[82]	3					X			
	[40]	1					X			
	[87]	1					X			
	[8]	2					X			
	[56]	1					X			
	[84]	1					X			
	[47]	1						X		
	[21]	3						X		
	[94]	1							X	
	[69]	1								X
Total: 29										
Simulated annealing algorithms	[42]	2	X							
	[9]	2	X							
	[81]	1	X							
	[61]	1	X							
	[78]	3	X							
	[70]	2		X						
	[65]	2			X					
	[15]	2			X					
	[28]	5					X			
	[67]	2					X			
	[57]	2							X	
Total: 11										
Tabu search	[52]	1	X							
	[5]	1	X							
	[66]	2	X							
	[71]	2	X							
	[26]	2				X				
	[68]	1					X			
Total: 6										
Ant colony optimization algorithms (ACO algorithm)	[10]	2	X							
	[51]	1	X							

Table 9 (continued)

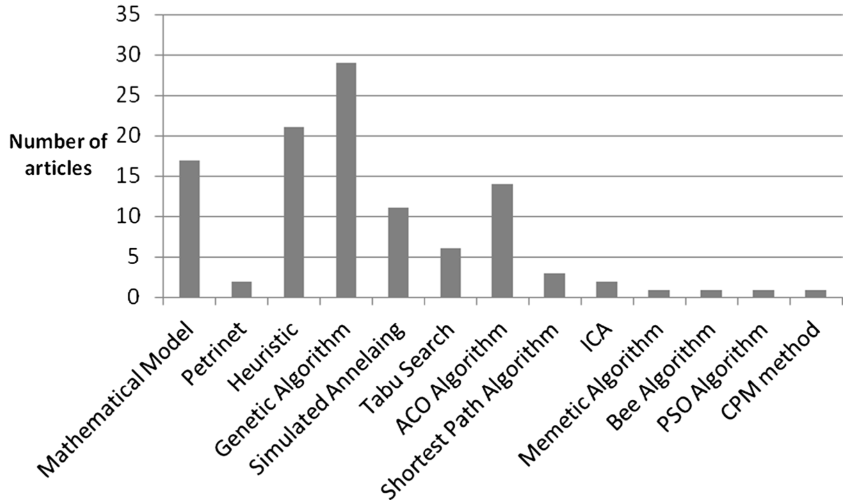
Method	Reference	No. of objectives	Problem type							
			SM_D_S	SM_D_U	SM_P_S	SM_P_U	MM_D_S	MM_D_U	MM_P_S	MM_P_U
	[27]	1	X							
	[17]	2	X							
	[18]	3	X							
	[64]	2	X							
	[59]	4	X							
	[100]	1		X						
	[22]	1		X						
	[11]	1		X						
	[83]	2					X			
	[4]	1					X			
	[58]	2							X	
	[3]	1								X
Total: 14										
Shortest path algorithm	[44]	2	X							
	[35]	1		X						
	[13]	1			X					
Total: 3										
Imperialistic competitive algorithm (ICA)	[63]	2		X						
	[7]	3				X				
Total: 2										
Memetic algorithm	[19]	2	X							
Total: 1										
Bees algorithms	[88]	1	X							
Total: 1										
Critical path method	[6]	1		X						
Total: 1										
Particle swarm optimization algorithm	[39]	3			X					
Total: 1										
Total in problem type			55	11	14	4	15	4	3	3

Some papers are included in more than one method (like mathematical + other method).

solve this model efficiently, a fuzzy simulation is designed and embedded into genetic algorithm to produce a hybrid intelligent algorithm. *They compared this hybrid algorithm with other heuristics using mean and standard deviation of the measures and found that it performs better. Embedding the simulated annealing algorithm into genetic algorithm to solve this problem is a good approach, which improved the performance of the resultant hybrid algorithm. Similarly, one may try to embed the genetic algorithm within simulated annealing algorithm to solve this problem. In such approach, the genetic algorithm may be used to provide the initial seed for the simulated annealing algorithm. Then its performance may be compared with that of the hybrid algorithm proposed in this research.*

Simulated annealing algorithm McMullen and Frazier [57] considered the assembly line balancing problem in which multiple product types scheduled in mixed-model fashion with probabilistic task times and parallel workstations. The objectives are minimization of the total cost and maximization of the degree to which the desired cycle time is achieved. They designed a simulated annealing algorithm for this problem. Based on an experiment, they found that the simulated annealing algorithm yields significantly better solution on cycle time performance, but only average solution was on cost performance. When equal weights are assumed for both the objectives, the proposed simulated annealing algorithm performs better than other approaches. As an extension to this research, several other heuristics may be used as seed

Fig. 3 Method vs. no. of articles graph



generation algorithms to supply initial seed to the simulated annealing algorithm, and their performance may be compared to select the best hybrid algorithm for solving this problem.

ACO algorithms McMullen and Tarasewich [58] considered the mixed model probabilistic assembly line balancing problem with parallel workstations in which the objective is to design the assembly line such that the number of workstations or the cost is minimized for a given cycle time. They used ant techniques to solve this problem and found that if efficient cycle time performance is the goal, then three of the four ant-related heuristics (all but ANT-4) outperforms the other 23 comparison heuristics tested. If the minimization of the cost is the objective, then only one ant-related heuristic (ANT-4) outperforms the comparison heuristics. In terms of on-time completion, two ant-heuristics (ANT-2 and ANT-3) perform well. They used simulation to fix the parameters of the proposed ACO algorithm. As a substitute to this attempt, Taguchi's technique, namely, *S*/*N* ratio may be used to optimize the parameters of the proposed ACO algorithms and the corresponding results of ACO algorithms may be compared with the results presented in this research.

Under the category of multi-model probabilistic straight-type (MM_P_S) assembling line balancing problem, fuzzy simulation embedded genetic algorithm, simulated annealing algorithm, and ant-related heuristics are developed. In this category, a limited work is done. Though a limited literature is available for this problem, the trend shows that development of hybrid heuristics will emerge for this problem.

The summary of the literature of the MM_P_S problem is given in Table 7. In this table, SALB-1 nested under MM_P_S problem is represented by MM_P_S-1, and SALB-2 nested under MM_P_S problem is represented by MM_P_S-2. Based on the summary provided in this table, the practitioners are suggested to use the ACO algorithm developed by McMullen and Tarasewich [58] to solve the one-sided SALB-1 problem. The one-sided SALB-2 problem may be solved using the simulated annealing algorithm developed by McMullen and Frazier [57].

2.4.2 Multi-model probabilistic U-type (MM_P_U) problem

The multi-model probabilistic U-type (MM_P_U) assembly line balancing problem is relatively complex when compared to the SM_P_U problem. This problem considers multiple models which are to be assembled in the assembly line in which the workstations are arranged in U form. The execution time of each task of each of the models (products) that are assembled in the assembly line is probabilistic. This section presents a review of literature of the multi-model probabilistic U-type (MM_P_U) assembly line balancing problem. The literature of this problem is classified under genetic algorithm and ACO algorithm, and they are presented below.

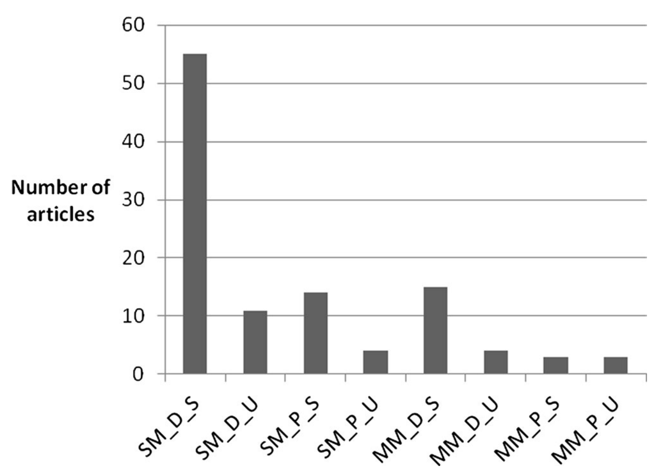


Fig. 2 Problem type vs. no. of articles graph

Table 10 Classification of literature after nesting SALB-1, SALB-2, and SALB-1 and SALB-2 in each classification shown in Fig. 1

Category	Article numbers	Total
SM_D_S-1	5, 9, 10, 16, 17, 18, 19, 23, 25, 27, 37, 42, 43, 44, 46, 52, 53, 54, 59, 61, 64, 66, 71, 72, 74, 75, 76, 78, 79, 80, 86, 88, 90, 92, 96, 97	36
SM_D_S-2	30, 33, 45, 49, 51, 62, 81	7
SM_D_S-1&2	48, 85	2
SM_D_U-1	6, 11, 22, 35, 63, 70, 95, 100	8
SM_D_U-2	Nil	0
SM_D_U-1&2	34, 93	2
SM_P_S-1	2, 13, 26, 32, 65, 91, 15	7
SM_P_S-2	39, 41, 55, 98	4
SM_P_S-1&2	99	1
SM_P_U-1	7, 12, 26	3
SM_P_U-2	Nil	0
SM_P_U-1&2	Nil	0
MM_D_S-1	4, 8, 36, 40, 56, 67, 68, 83, 84	9
MM_D_S-2	82, 87	2
MM_D_S-1&2	28	1
MM_D_U-1	21, 47	2
MM_D_U-2	Nil	0
MM_D_U-1&2	Nil	0
MM_P_S-1	94, 58	2
MM_P_S-2	57	1
MM_P_S-1&2	Nil	0
MM_P_U-1	3	1
MM_P_U-2	69	1
MM_P_U-1&2	Nil	0

Genetic algorithm In the mixed-model U-line assembly line balancing problem, there are two interrelated problems called line balancing and model sequencing. Ozcan et al. [69] considered this problem with probabilistic task times and proposed a genetic algorithm to solve it such that the cycle time is minimized for a given number of workstations.

ACO algorithm Agrawal and Tiwari [3] developed a collaborative ant colony optimization algorithm for the probabilistic multi-model U-shaped disassembly line balancing and sequencing. Disassembly is a process of systematic extraction of useful materials from discarded or worn-out products. This approach maintains bilateral colonies of ants which independently identifies the two sequences but utilizes the information obtained by their collaboration to guide the future path. They proved that the collaborative ant colony optimization algorithm performs better than the traditional ant colony optimization algorithm through DOE techniques. As a future work, an improved procedure for allocation of tasks to workstations may be developed.

In this category, the researchers concentrated in developing genetic algorithm and ant colony optimization algorithm. For each of these algorithms, Taguchi's *S/N* ratio may be used to optimize its parameters before executing its steps and the corresponding results may be compared with the results of the respective algorithm reported in this section. Since very little work is carried out in this category, there is more scope to develop algorithms using other methods such as simulated annealing algorithm, tabu search, etc. Further, an attempt may be made to develop hybrid algorithms for this problem. Also, mathematical models may be developed to validate the heuristics under this category using small- and moderate-size problems.

The summary of the literature of the MM_P_U problem is given in Table 8. *In this table, SALB-1 nested under MM_P_U problem is represented by MM_P_U-1, and SALB-2 nested under MM_P_U problem is represented by MM_P_U-2.* Based on the summary provided in this table, the practitioners are suggested to use the genetic algorithm developed by Ozcan et al. [69] to solve the one-sided SALB-2 problem and the ACO algorithm developed by Agrawal and Tiwari [3] to solve the one-sided SALB-1.

3 Analysis of ALB literature

This section analyzes the assembly line balancing literature in terms of eight problem types, methods used to solve assembly line balancing problems, and primary classification of ALB problems (SALB-1 problem, SALB-2 problem, and combination of both problems). The summary of assembly line balancing (ALB) literature as per the first two classifications is given in Table 9. The graph for "Problem Type" vs. "No. of Articles" is shown in Fig. 2. The graph for "Method vs. Number of Articles" is shown in Fig. 3. From Fig. 2, it is clear that more researchers contributed to SM_D_S problem and it is followed by MM_D_S, SM_P_S, and SM_D_U. The number of articles in each other problem type is very less. From Fig. 3, the methods in the decreasing order of number of articles are shown below:

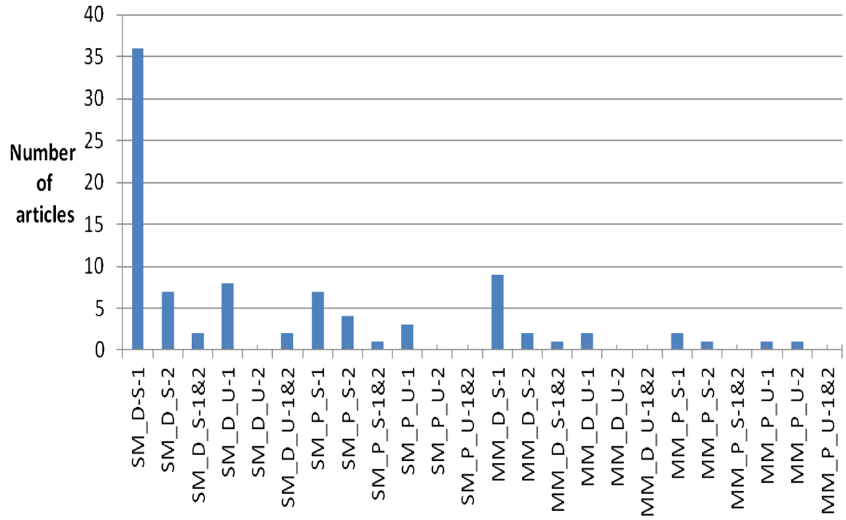


Fig. 4 Graph for no. of articles vs. ALB problem categories after nesting SALB-1, SALB-2, and SALB-1 and SALB-2 in the classification shown in Fig. 1

- Genetic algorithms
- Heuristics
- Mathematical models
- ACO algorithms
- Simulated annealing algorithms
- Tabu search
- Shortest path algorithms
- Petri net
- Imperialistic competitive algorithm (ICA)
- Memetic algorithm
- Bees algorithms
- Particle swarm optimization algorithm (PSO algorithm)
- Critical path method algorithm

This type of classification gives an idea about the contributions of researchers in different methods used for the assembly balancing problems as well as in different types of assembly line balancing problem.

Next, the classification of literature after nesting SALB-1 problem, SALB-2 problem, and combination of both (SALB-1 and SALB-2) problems under each of the classifications shown in Fig. 1 is given in Table 10. The corresponding graphical view is shown in Fig. 4. From Fig. 4, it is clear that maximum number of researchers concentrated on the SM_D_S-1 and it is followed by MM_D_S-1,SM_D_U-1, SM_D_S-2, and SM_P_S-1. The contributions of researchers in the remaining categories are insignificant. So future researches may be carried out in these the areas where the contributions are insignificant.

4 Conclusion

The research on the assembly line balancing problems is highly crucial for the productivity of mass production systems with the aim of maximizing the throughput rate. The assembly line balancing problems are classified into the following categories:

- Single-model deterministic straight-type (SM_D_S) problem
- Single-model deterministic U-type (SM_D_U) problem
- Single-model probabilistic straight-type (SM_P_S) problem
- Single-model probabilistic U-type (SM_P_U) problem
- Multi-model deterministic straight-type (MM_D_S) problem
- Multi-model deterministic U-type (MM_D_U) problem
- Multi-model probabilistic straight-type (MM_P_S) problem
- Multi-model probabilistic U-type (MM_P_U) problem

In this paper, a comprehensive review of literature under the above eight categories is presented. Further, within each category, the literature is classified on the basis of the methods used to solve the *ALB* problems and inferences are drawn.

At the end of each of the eight categories, the literature is further divided into three subcategories by nesting SALB-1, SALB-2, and SALB-1 & SALB-2 in it, as listed below:

- SM_D_S-1, SM_D_S-2, and SM_D_S-1 & SM_D_S-2
- SM_D_U-1, SM_D_U-2, and SM_D_U-1 & SM_D_U-2
- SM_P_S-1, SM_P_S-2, and SM_P_S-1 & SM_P_S-2
- SM_P_U-1, SM_P_U-2, and SM_P_U-1 & SM_P_U-2
- MM_D_S-1, MM_D_S-2, and MM_D_S-1 & MM_D_S-2
- MM_D_U-1, MM_D_U-2, and MM_D_U-1 & MM_D_U-2
- MM_P_S-1, MM_P_S-2, and MM_P_S-1 & MM_P_S-2
- MM_P_U-1, MM_P_U-2, and MM_P_U-1 & MM_P_U-2

Under the SM_D_S assembly line balancing problem, various authors researched on the development of models, simple heuristics, meta-heuristics, viz. genetic algorithm, global search heuristic, tabu search, hybrid tabu search, ant colony optimization algorithm, simulated annealing, shortest path algorithm, memetic algorithm, and bee algorithms. Within this category of problem, the variations are simple assembly line balancing problem, parallel assembly line balancing problem, two-sided assembly line balancing with work relatedness in type 1 and type 2 assembly line balancing problems. Under the SM_D_U assembly line balancing problem, the authors developed shortest route formulation, goal programming model, simple heuristic, ant colony optimization algorithm, CPM method, and imperialistic competitive algorithm. Under the SM_P_S assembly line balancing problem, the authors designed mathematical models, a heuristic known as TOPSIS, and meta-heuristics, viz. genetic algorithm, simulated annealing algorithm, particle swarm optimization algorithm, and hybrid simulated annealing algorithm. The simple as well as parallel version of the assembly line balancing problem is dealt in this category for single and multi-objective situations. In the category “SM_P_U assembly line balancing problem,” cost-oriented heuristic and imperialistic competitive algorithm are developed.

Under the MM_D_S assembly line balancing problem, the authors developed mathematical model, meta-heuristics, viz. genetic algorithms, hybrid genetic algorithms, simulated annealing algorithm, ant colony algorithm, and tabu search algorithm. It is observed that many authors developed genetic as well as hybrid genetic algorithms for this problem. Under the MM_D_U assembly line balancing problem, endosymbiotic evolutionary algorithm and evolutionary algorithm called coincidence (COIN) algorithm are developed. Under the MM_P_S assembling line balancing problem, the authors developed simulated annealing algorithm, ant-related heuristics, and fuzzy simulation-embedded genetic algorithm. In the MM_P_U assembly line balancing problem, only ACO algorithm is developed. An overall observation is that more researches have been carried out on SM_D_S problem, and further, SM_D_S-1 problem has been researched by many researchers.

At the end of each section of the literature classification, the summary of literature is presented to enable an integrated comparison of the literature of the respective assembly line balancing problem. Further, directions are given to select appropriate methods to solve different problems within each section based on the goals of practitioners.

4.1 Suggestions and future research

The critical review of literature of assembly line balancing problems gives in-depth understanding of different categories of assembly line balancing problems and state-of-the-art research in each category.

Researchers may develop more meta-heuristics, viz. genetic algorithm, simulated annealing algorithm, tabu search algorithm, ACO algorithm, PSO algorithm, GRASP, etc., for multi-model ALB problems. While developing genetic algorithm, the researchers may propose different crossover methods and compare them for their effectiveness in terms solution accuracy. Similarly, while developing simulated annealing algorithm, an efficient seed generation algorithm is to be designed in its first phase, which will result with greater effect on the solution accuracy in the second phase of the simulated annealing algorithm. The development of hybrid approach is encouraged. In this approach, two or more basic heuristic(s)/ meta- heuristic(s) are to be combined to bring synergetic effect on the solution accuracy. For a selected problem, the researchers may conduct a study type research to find the best combination of the basic heuristic(s)/ meta-heuristic(s) in terms of greater solution accuracy to serve as a hybrid algorithm.

In genetic algorithms, researchers use a specific crossover method. Instead, they may select a crossover method randomly for each pair of chromosome from a pool of crossover methods, while performing the crossover operation in each of the iterations. This will give an opportunity to form offspring with improved fitness function values.

In most of the meta-heuristics, viz. simulated annealing algorithm, genetic algorithm, ant colony optimization algorithm, particle swarm optimization algorithm, etc., the parameters are either fixed through trial-and-error method using an experiment or based on past published works, which may not guarantee the best performance. Hence, before executing each of these algorithms, researchers should optimize its parameters using Taguchi’s technique, which will enhance solution accuracy of the problems that are solved using that algorithm.

In all the categories, statistical comparisons of the performance of new approaches with that of existing approaches are not comprehensive in terms of applying design of experiment to test the effects of the factors/parameters of the problems on the response variable(s). So the comparison of the algorithms must be done using a complete factorial experiment by considering relevant factors with interaction effects, in which “Algorithm” is a factor to check the superiority of the proposed algorithm.

It is observed that very less work is done on multi-model assembly line balancing categories and the categories with U-type assembly line balancing problems. Future researches may be carried out in these categories for developing meta-heuristics with single/multi-objectives. Also, it is found that less work is carried out on SALB-2 problem. Since it is also an important problem in assembly line balancing, efforts may be made to research more on this problem.

Acknowledgments The authors are very much thankful to all referees for their constructive criticisms and suggestions that helped them to improve this paper.

References

1. Aase GR, Olson JR, Schniederjans MJ (2004) U-shaped assembly line layouts and their impact on labor productivity: an experimental study. Eur J Oper Res 156:698–711
2. Agpak K, Gokcen H (2007) A chance-constrained approach to stochastic line balancing problem. Eur J Oper Res 180:1098–1115
3. Agrawal S, Tiwari MK (2008) A collaborative ant colony algorithm to stochastic mixed-model U-shaped disassembly line balancing and sequencing problem. Int J Prod Res 46(6):1405–1429
4. Akpinar S, Bayhan GM, Baykasoglu A (2013) Hybridizing ant colony optimization via genetic algorithm for mixed-model assembly line balancing problem with sequence dependent setup times between tasks. Appl Soft Comput 13:574–589
5. Annarongsri S, Limnararat S (2007) A hybrid tabu search method for assembly line balancing. Proceedings of the 7th international conference on simulation (modelling and optimization) China
6. Avikal S, Jain R, Mishra PK, Yadav HC (2013) A heuristic approach for U-shaped assembly line balancing to improve labour productivity. Comput Ind Eng 64:895–901
7. Bagher M, Zandieh M, Farsijani H (2010) Balancing of stochastic U-type assembly lines: an imperialist competitive algorithm. Int J Adv Manuf Technol. doi:10.1007/s00170-010-2937-3
8. Bai Y, Zhao H, Zhu L (2009) Mixed-model assembly line balancing using the hybrid genetic algorithm. ICMTMA 3:242–245
9. Baykasoglu A (2006) Multi-rule multi-objective simulated annealing algorithm for straight and U type assembly line balancing problems. J Intell Manuf 17:217–232
10. Baykasoglu A, Dereli T (2008) Two-sided assembly line balancing using an ant-colony-based heuristic. Int J Adv Manuf Technol 36: 582–588
11. Baykasoglu A, Dereli T (2009) Simple and U-type assembly line balancing by using an ant colony based algorithm. Math Comput Appl 14(1):1–12
12. Baykasoglu A, Ozbakir L (2007) Stochastic U-line balancing using genetic algorithms. Int J Adv Manuf Technol 32:139–147
13. Boysen N, Fliedner M (2008) A versatile algorithm for assembly line balancing. Eur J Oper Res 184:39–56
14. Boysen N, Fliedner M, Scholl A (2007) A classification of assembly line balancing problems. Eur J Oper Res 183:674–693
15. Cakir B, Altiparmak F, Dengiz B (2011) Multi-objective optimization of a stochastic assembly line balancing: a hybrid simulated annealing algorithm. Comput Ind Eng 60:376–384
16. Cheshmehgaz HR, Desa MI, Kazemipour F (2012) A cellular-rearranging of population in genetic algorithms to solve assembly-line balancing problem. J Mech Eng Autom 2:25–35
17. Chica M, Cordon O, Damas S, Bautista J (2010) Multi-objective constructive heuristics for the1/3 variant of the time and space assembly line balancing problem: ACO and random greedy search. Inf Sci 180:3465–3487
18. Chica M, Cordon O, Damas S, Bautista J (2011) Including different kinds of preferences in a multi-objective ant algorithm for time and space assembly line balancing on different Nissan scenarios. Expert Syst Appl 38:709–720
19. Chica M, Cordon O, Damas S, Bautista J (2012) Multiobjective memetic algorithms for time and space assembly line balancing. Eng Appl Artif Intell 25:254–273
20. Chong KE, Omar, MK, Baker NA (2008) Solving assembly line balancing problem using genetic algorithm with heuristic treated initial population. Proceedings of the World Congress on Engineering ISBN:978-988-17012-3-7
21. Chutima P, Olanviwatchai P (2010) Mixed model U shaped assembly line balancing problem with coincidence memetic algorithm. J Softw Eng Appl 3:347–363
22. Cuoglu IS, Erel E, Alp A (2009) Ant colony optimization for the single model U-type assembly line balancing problem. Int J Prod Econ 120:287–300
23. Dar-El EM (1975) Solving large single-model assembly line balancing problems: a comparative study. AIIE Trans 7(3):302–310
24. Dar-El EM, Rubinovitch Y (1979) MUST— a multiple solutions technique for balancing single model assembly lines. Manag Sci 25: 1105–1113
25. Deckro RF, Rangachari S (1990) A goal approach to assembly line balancing. Comput Oper Res 17(5):509–521
26. Erel E, Sabuncuoglu I, Sekerci H (2005) Stochastic assembly line balancing using beam search. Int J Prod Res 43(7):1411–1426
27. Fattahi P, Roshani A, Roshani A (2011) A mathematical model and ant colony algorithm for multi-manned assembly line balancing problem. Int J Adv Manuf Technol 53:363–378
28. Fattahi P, Salehi M (2009) Sequencing the mixed-model assembly line to minimize the total utility and idle costs with variable launching interval. Int J Adv Manuf Technol 45:987–998
29. Gamberini R, Grassi A, Rimini B (2006) A new multi-objective heuristic algorithm for solving the stochastic line re-balancing problem. Int J Prod Econ 102:226–243
30. Gao J, Sun L, Wang L, Gen M (2009) An efficient approach for type II robotic assembly line balancing problems. Comput Ind Eng 56: 1065–1080
31. Gen M, Cheng R (2000) Genetic algorithms and engineering optimization. Wiley Interscience, New York
32. Gen M, Tsujimura Y, Li Y (1996) Fuzzy assembly line balancing using genetic algorithms. Comput Ind Eng 31(3/4):631–634
33. Genikomsakis KN, Tourassis VD (2010) Enhancing the largest set rule for assembly line balancing through the concept of bi-directional work relatedness. J Adv Comput Intell Inf 14(4):353–363
34. Gokcen H, Agpak K (2006) A goal programming approach to simple U-line balancing problem. Eur J Oper Res 171:577–585
35. Gokcen H, Agpak K, Gencer C, Kizilkaya E (2005) A shortest route formulation of simple U-type assembly line balancing problem. Appl Math Model 29:373–380
36. Gokcen H, Erel E (1998) Binary integer formulation for mixed-model assembly line balancing problem. Comput Ind Eng 34(2): 451–461
37. Goncalves JF, Almeida JRD (2002) A hybrid genetic algorithm for assembly line balancing. J Heuristics 8:629–642
38. Gutjahr AL, Nemhauser GL (1964) An algorithm for the line balancing problem. Manag Sci 11(2):308–315
39. Hamta N, Ghomi SMTF, Jolai F, Shirazi MA (2013) A hybrid PSO algorithm for a multi-objective assembly line balancing problem with flexible operation times, sequence-dependent setup times and learning effect. Int J Prod Econ 141:99–111
40. Haq AN, Zayaprakash J, Rengarajan K (2006) A hybrid genetic algorithm approach to mixed-model assembly line balancing. Int J Adv Manuf Technol 28:337–341
41. Hazir O, Dolgui A (2013) Assembly line balancing under uncertainty:robust optimization models and exact solution method. Comput Ind Eng 65:261–267
42. Hong DS, Cho HS (1997) Generation of robotic assembly sequences with consideration of line balancing using simulated annealing. Robotica 15:663–674
43. Jiano J, Kumar A, Martin W (2006) A web-based interactive advisor for assembly line balancing. Int J Adv Manuf Technol 27:1192–1201
44. Kao HH et al (2010) An optimal algorithm for type-I assembly line balancing problem with resource constraint. Afr J Bus Manag 4(10): 2051–2058

45. Kilincci O (2010) A Petri net-based heuristic for simple assembly line balancing problem of type 2. Int J Adv Manuf Technol 46:329–338
46. Kilincci O, Bayhan GM (2006) A Petri net approach for simple assembly line balancing problems. Int J Adv Manuf Technol 30: 1165–1173
47. Kim YK, Kim JY, Kim Y (2006) An endosymbiotic evolutionary algorithm for the integration of balancing and sequencing in mixed-model U-lines. Eur J Oper Res 168:838–852
48. Kim YK, Kim YJU, Kim Y (1996) Genetic algorithms for assembly line balancing with various objectives. Comput Ind Eng 30(3):397–409
49. Kim YK, Song WS, Kim JH (2009) A mathematical model and a genetic algorithm for two-sided assembly line balancing. Comput Oper Res 36:853–865
50. Kottas JF, Lau HS (1973) A cost oriented approach to stochastic line balancing. AIIE Trans 5:164–171
51. Lai, LKC, Liu JNK (2009) ALBO: an assembly line balance optimization model using ant colony optimization. Fifth International Conference on Natural Computation Tianjian, China
52. Lapierre SD, Ruiz A, Soriano P (2006) Balancing assembly lines with tabu search. Eur J Oper Res 168:826–837
53. Lee TO, Kim Y, Kim YK (2001) Two-sided assembly line balancing to maximize work relatedness and slackness. Comput Ind Eng 40: 273–292
54. Levintin G, Rubinovitz J, Shnits B (2006) A genetic algorithm for robotic assembly line balancing. Eur J Oper Res 168:811–825
55. Liu SB, Ong HL, Huang HC (2005) A bidirectional heuristic for stochastic assembly line balancing type-II problem. Int J Adv Manuf Technol 25:71–77
56. Mamun AA, Khaled AA, Ali SM, Chowdhury MM (2012) A heuristic approach for balancing mixed-model assembly line of type I using genetic algorithm. Int J Prod Res 50:5106–5116
57. McMullen PR, Frazier GV (1998) Using simulated annealing to solve a multi-objective assembly line balancing problem with parallel workstations. Int J Prod Res 36(10):2717–2741
58. McMullen PR, Tarasewich P (2003) Using ant techniques to solve the assembly line balancing problem. IIE Trans 35:605–617
59. McMullen PR, Tarasewich P (2006) Multi-objective assembly line balancing via a modified ant colony technique. Int J Prod Res 44(1): 27–42
60. Murata T, Ishibuchi H, Tanaka H (1996) Multi-objective genetic algorithm and its application to flowshop scheduling. Comput Ind Eng 30:957–968
61. Narayanan S, Panneerselvam R (2000) New efficient set of heuristics for assembly line balancing. Int J Manag Syst 16(3):287–300
62. Nearchou AC (2008) Multi-objective balancing of assembly lines by population heuristics. Int J Prod Res 46(8):2275–2297
63. Nourmohammadi A, Zandieh M, Tavakkoli-Moghaddam R (2013) An imperialist competitive algorithm for multi-objective U-type assembly line design. J Comput Sci 4:393–400
64. Ozbakir L, Baykasoglu A, Gorkemli B, Gorkemli L (2011) Multiple-colony ant algorithm for parallel line balancing problem. Appl Soft Comput. doi:10.1016/j.asoc.2010.12.021
65. Ozcan U (2010) Balancing stochastic two-sided assembly lines: a chance-constrained piecewise-linear, mixed integer program and a simulated annealing algorithm. Eur J Oper Res 205:81–97
66. Ozcan U, Cercioglu H, Gokcen H, Toklu B (2009) A tabu search algorithm for the parallel assembly line balancing problem. G.U. J Sci 22(4):313–323
67. Ozcan U, Cercioglu H, Gokcen H, Toklu B (2010) Balancing and sequencing of parallel mixed-model assembly lines. Int J Prod Res 48(17):5089–5113
68. Ozcan U, Gokcen H, Toklu B (2010) Balancing parallel two-sided assembly lines. Int J Prod Res 48(16):4767–4784
69. Ozcan U, Kellegoz T, Toklu B (2011) A genetic algorithm for the stochastic mixed-model U-line balancing and sequencing problem. Int J Prod Res 49(6):1605–1626
70. Ozcan U, Toklu B (2009) A new hybrid improvement heuristic approach to simple straight and U-type assembly line balancing problems. J Intell Manuf 20:123–136
71. Ozcan U, Toklu B (2009) A tabu search algorithm for two-sided assembly line balancing. Int J Adv Manuf Technol 43:822–829
72. Ozcan U, Toklu B (2009) Multiple-criteria decision-making in two-sided assembly line balancing: a goal programming and a fuzzy goal programming models. Comput Oper Res 36:1955–1965
73. Panneerselvam R (2005) Production and operations management, 2nd edn. PHI Learning, New Delhi
74. Panneerselvam R, Sankar CO (1993) New heuristics for assembly line balancing problem. Int J Manag Syst 9(1):25–36
75. Ponnambalam SG, Aravindan P, Naidu GM (1999) A comparative evaluation of assembly line balancing heuristics. Ad Manuf Technol 15:577–586
76. Ponnambalam SG, Aravindan P, Naidu GM (2000) A multi-objective genetic algorithm for solving assembly line balancing problem. Int J Adv Manuf Technol 16:341–352
77. Rashid MFF, Hutabarat W, Tiwari A (2011) A review on assembly sequence planning and assembly line balancing optimization using soft computing approaches. Int J Adv Manuf Technol 59(1–4):335–349
78. Roshani A, Fattahi P, Roshani A, Salehi M, Roshani A (2012) Cost-oriented two-sided assembly line balancing problem: a simulated annealing approach. Int J Comput Integr Manuf 25:680–715
79. Rubinnovitz J, Levitin G (1995) Genetic algorithm for assembly line balancing. Int J Prod Econ 41(1):343–354
80. Sabuncuoglu I, Erel R, Tanyer M (2000) Assembly line balancing using genetic algorithm. J Intell Manuf 11:295–310
81. Seyed-Alagheband SA, Ghomi SMTF, Zandieh M (2011) A simulated annealing algorithm for balancing the assembly line type II problem with sequence-dependent setup times between tasks. Int J Prod Res 49(3):805–825
82. Simaria AS, Vilarinho PM (2004) A genetic algorithm based approach to the mixed-model assembly line balancing problem of type II. Comput Ind Eng 47(4):391–407
83. Simaria AS, Vilarinho PM (2009) 2-ANTBAL: an ant colony optimization for balancing two-sided assembly lines. Comput Ind Eng 56(2):489–506. doi:10.1016/j.cie.2007.10.007
84. Sivasankaran P, Shahabudeen P (2013) Genetic algorithm for concurrent balancing of mixed-model assembly lines with original task times of models. Intell Inf Manag 5:84–92
85. Sivasankaran P, Shahabudeen P (2013) Modelling hybrid single model assembly line balancing problem. Udyog Pragati 37:26–36
86. Sotskov YN, Dolgui A, Portmann MC (2006) Stability analysis of an optimal balance for an assembly line with fixed cycle time. Eur J Oper Res 168:783–797
87. Su P, Lu Y (2007) Combining genetic algorithm and simulation for the mixed-model assembly line balancing problem. ICNC 4:314–318, Third International Conference on Natural Computation (ICNC 2007)
88. Tapkan P, Ozbakir L, Baykasoglu A (2012) Modeling and solving constrained two-sided assembly line balancing problem via bee algorithms. Appl Soft Comput 12:3343–3355
89. Tasan SO, Tunali S (2008) A review of the current applications of genetic algorithms in assembly line balancing. J Intell Manuf 19: 49–69
90. Thangavelu SR, Shetty CM (1971) Assembly line balancing by zero-one integer programming. IIE Trans 3(1):61–68
91. Tsujimura Y, Gen M, Kubota E (1995) Solving fuzzy assembly-line balancing problem with genetic algorithm. Comput Ind Eng 29(1–4):543–547

92. Wang HS, Che ZH, Chiang CJ (2012) A hybrid genetic algorithm for multi-objective product plan selection problem with ASP and ALB. Expert Syst Appl 39:5440–5450
93. Widyadana GA (2009) Multi-objective model for balancing U-type assembly line with permanent and temporary workers. Jurnal Tecnik Industri 11(1):33–42
94. Xu W, Xiao T (2008) Mixed model assembly line balancing problem with fuzzy operation times and drifting operations. Proceedings of the 2008 Winter Simulation Conference (IEEE) 1752–1760
95. Yegul MF, Agpak K, Yavuz M (2010) A new algorithm for U-shaped two-sided assembly line balancing. Trans Can Soc Mech Eng 34(2):225–241
96. Yeh DH, Kao HH (2009) A new bidirectional heuristic for the assembly line balancing problem. Comput Ind Eng 57:1155–1160
97. Yu J, Yin Y (2010) Assembly line balancing based on an adaptive genetic algorithm. Int J Adv Manuf Technol 48:347–354
98. Zacharia PT, Nearchou AC (2012) Multi-objective fuzzy assembly line balancing using genetic algorithm. J Intell Manuf. doi:10.1007/s10845-010-0400-9
99. Zacharia PT, Nearchou AC (2013) A meta-heuristic algorithm for the fuzzy assembly line balancing type-E problem. Comput Oper Res 40:3033–3044
100. Zhang Z, Cheng W, Cheng Y and Liang J (2008) A novel ant colony optimization algorithm for U-shaped line balancing problem. ICNC 7:455–459, Fourth International Conference on Natural Computation