Model and migrating birds optimization algorithm for two-sided assembly line worker assignment and balancing problem

Mukund Nilakantan Janardhanan[1*], Zixiang Li[2,3], Peter Nielsen,[4]

[1]Department of Engineering, University of Leicester, United Kingdom.

Email: mukund.janardhanan@leicester.ac.uk, peter@m-tech.aau.dk

[2]Key Laboratory of Metallurgical Equipment and Control Technology, Wuhan University of Science and Technology, Wuhan, Hubei, China.

[3]Hubei Key Laboratory of Mechanical Transmission and Manufacturing Engineering, Wuhan University of Science and Technology, Wuhan, Hubei, China.

Email: zixiangliwust@gmail.com

[4]Department of Materials and Production, Aalborg University, Denmark.

Email: peter@m-tech.aau.dk

Abstract: Worker assignment is a relatively new problem in assembly lines that typically is encountered in situations in which the workforce is heterogeneous. The optimal assignment of a heterogeneous workforce is known as the assembly line worker assignment and balancing problem (ALWABP). This problem is different from the well-known simple assembly line balancing problem concerning the task execution times, and it varies according to the assigned worker. Minimal work has been reported in worker assignment in two-sided assembly lines. This research studies worker assignment and line balancing in two-sided assembly lines with an objective of minimizing the cycle time (TALWABP). A mixed-integer programming model is developed, and CPLEX solver is used to solve the small-size problems. An improved migrating birds optimization (MBO) algorithm is employed to deal with the large-size problems due to the NP-hard nature of the problem. The proposed algorithm utilizes a restart mechanism to avoid being trapped in the local optima. The solutions obtained using the proposed algorithms are compared with well-known metaheuristic algorithms such as artificial bee colony and simulated annealing. Comparative study and statistical analysis indicate that the proposed algorithm can achieve the optimal solutions for small-size problems, and it shows superior performance over benchmark algorithms for large-size problems.

Keywords: Assembly line balancing; Two-sided assembly line; Worker assignment; Migrating birds optimization; Metaheuristics

1. Introduction

Assembly lines are utilized extensively in manufacturing to produce standardized products in a process in which a set of tasks is divided among workstations, and each workstation is assigned workers to perform the allocated tasks (Oksuz et al. 2017). Assembly line balancing problems primarily aim at allocating the tasks to workstations in a balanced manner such that one or several objective functions are optimized. The basic version of the assembly line balancing problem is simple assembly line balancing problem (ALBP) with workstation minimization criterion, referred to as type I ALBP (Boysen et al. 2007; Chica et al. 2015). Another widely researched objective function is minimizing cycle time, leading to a more complex ALBP referred to as type II ALBP (Zhong and Ai 2017). Type I ALBP has been criticized for lacking real-life application and being too theoretical since the environment in real industry is much more complex. Hence, variants of simple type I ALBP are studied to tackle problems found in industry (Purnomo et al. 2013). The literature has reported that type I ALBPs are NP-hard (Scholl and Becker 2006), hence variants of ALBP are always more complex due to having more constraints.

A variant of ALBP includes the consideration of worker assignment (Moreira et al. 2015a). This is an essential problem that is found in sheltered work centers for the disabled (Miralles et al. 2007). Certain workers might need more time to perform/operate certain tasks or are incapable of operating some tasks. Due to increased concern and respect for the disabled, many industries are employing them, and companies always are concerned about worker assignment in assembly lines (Blum and Miralles 2011). A task allocation that ignores this situation might be ineffective or even infeasible. Worker assignment and task allocation result in a new integrated assembly line worker assignment and balancing problem which contains two interacting sub-problems. Worker assignment determines the assignment of workers to workstations, and assembly line balancing aims at distributing tasks to workstations in a balanced manner while satisfying different constraints such as precedence constraint and cycle time constraint.

The two-sided assembly line balancing problem (TALBP) is a variant of simple ALBP where workers operate tasks on two-faced workstations that are placed parallel to each other, referred to as mated-stations (Tang et al. 2017). This type of assembly line is widely used in industry since it has several advantages over the one-sided assembly line such as shorter line length, reduced material handling and more tool-sharing. When simultaneously taking the worker assignment and assembly line balancing into account for a two-sided assembly line, a new integrated two-sided assembly line worker-assignment and balancing problem (TALWABP) arises and is solved in this paper. The proposed problem belongs to the category of NP-hard problems. TALBP is proven to be NP-hard in the recent literature reported by Tang et al. (2017). If the worker assignment problem were fixed, it would be possible to utilize the proposed TALWABP to solve TALBP. Considered TALWABP has many worker assignments and it falls under the NP-hard category due to this. Figure 1 depicts an example of worker assignment and task allocation on a two-sided assembly line. TALWABP comprises two sub-problems: worker assignment and assembly line balancing. The assembly line balancing problem determines the detailed assignment of tasks on each mated station, while the worker assignment problem assigns the best-fit worker to each workstation. In the case of TALWABP, there is a special constraint on the preferred directions of tasks referred to as direction constraint (L-type tasks, R-type tasks, and E-type tasks). L-type tasks are allocated to the left side; R-type tasks are assigned to the right side, and E-type tasks can be assigned to the left or right side (Tuncel and Aydin 2014). Also, there is a special condition for tasks on mated stations: sequence-dependent idle time (Tang et al. 2016). Sequence-dependent idle time is caused by the precedence constraint and utilization of two sides, and it can be reduced by optimizing the task sequence on workstations. Taking the sequence-dependent idle time into account, TALWABP should aim to optimize worker assignment, task allocation to workstations, and task sequence on each workstation.

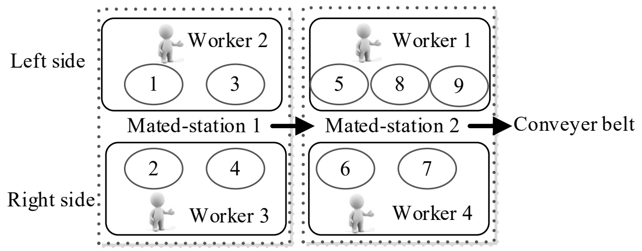

Figure 1. Worker assignment and task allocation on a two-sided assembly line

To the authors' best knowledge, there is limited work reported on TALWABP with the objective of minimizing cycle time, except for Janardhanan et al. (2018) who introduced the problem for the first time. To tackle this new problem, this research presents the following two major contributions. (1) The considered problem is studied for the first time, and a new mixed-integer programming model is formulated and solved by CPLEX solver for optimality to solve small-sized problems with less than 16 tasks. (2) This research develops an efficient metaheuristic algorithm to solve the new problem. A migrating birds optimization (MBO) algorithm is developed to tackle large-size problems in an acceptable CPU time. The considered problem is differs significantly from the traditional two-sided assembly line balancing problem (TALBP) and the algorithms utilized to solve TALBP are not directly applicable to the considered problem and this is one of the major motivation behind the work. For example, in literature the best reported algorithm to solve TALBP is iterated greedy algorithm and this cannot be utilized for a worker assignment problem which is the major component in the considered problem. Hence, this research focuses on utilizing new efficient algorithm rather than the existing algorithms utilized for TALBP. For solving complex problems in different applications with uncertainty and vagueness like the one considered here researchers have utilized fuzzy logic and metaheuristic approaches to solve them in an acceptable computation time and using these algorithms it helps in decision making which is very critical in such systems (Fahmi et al. 2018a; Fahmi et al. 2018c; Fahmi et al. 2017b; Fahmi et al. 2018e). MBO is a relatively new metaheuristic algorithm that has shown superior performance in solving similar types of optimization problems (Duman et al. 2012; Gao and Pan 2016; Zhang et al. 2017). The main properties of the MBO are a set of individuals searching in parallel for a good solution. Additionally, the proposed method is improved by replacing the incumbent individual immediately rather than evaluating it after all the neighbor solutions are obtained; this is done with the aim of increasing search speed. A restart mechanism is also employed to assist MBO to escape from local optima. A comprehensive comparative study is carried out to test the performance of the improved MBO, where MBO is compared with three artificial bee colony algorithms and a simulated annealing algorithm to demonstrate the superiority of the proposed algorithm. A computational study is conducted on 156 instances introduced in Janardhanan et al. (2018), and the performance of the proposed method is demonstrated. There is a need of developing a decision support system to help production managers in their decision making (Al_Janabi 2018; Fahmi et al. 2018b). The work done in this paper will help them to efficiently allocate workers and balance the workstation with an objective of minimizing the cycle time. Different researchers have used different computing techniques for decision making (Amin et al. 2018; Fahmi et al. 2018d) and this paper utilizes metaheuristic algorithms to solve complex real life problems like the one considered in this paper.

The remainder of the paper is organized as follows. Section 2 reviews the literature on TALBP and worker assignment respectively. The mathematical model is presented in Section 3. Section 4 presents detailed methodology along with an illustrated example. Section 5 presents a computational study and statistical analysis. Finally, the conclusion and future research directions are provided in Section 6.

2. Literature review

The study of assembly line design, balancing, and scheduling problems is widely reported in the literature (Dolgui et al. 2018). Although worker assignment and two-sided assembly line balancing have been thoroughly studied separately, there is limited work reported considering all the characteristics of the considered problem, namely the integrated two-sided assembly line worker assignment and balancing problem with cycle time minimization criterion. Hence, the following reviews first the recent or most cited works on two-sided assembly line balancing, and later presents the reported works on integrated worker assignment and balancing problem.

In the case of two-sided assembly line balancing problems (TALBP), Bartholdi (1993) presents the first work in this area by applying a first-fit heuristic to minimize the number of workstations. Later on, Lee et al. (2001) develop a group assignment procedure to maximize work-relatedness and slackness. Different researchers have utilized various metaheuristic algorithms as the optimization tool to minimize the number of workstations or cycle time. For workstation number minimization, the following algorithms are utilized: ant colony algorithms (Baykasoglu and Dereli 2008), tabu search algorithm (Özcan and Toklu 2008), and bee colony algorithms (Özbakır and Tapkan 2011). For cycle time minimization, the following algorithms have been reported: genetic algorithm (Kim et al. 2009), artificial bee colony algorithm (Tang et al. 2016), and iterated greedy algorithm (Li et al. 2016c). A detailed review and evaluation of these methods are presented in Tang et al. (2017).

Apart from the contributions reported on the basic TALBP, recently more contributions are reported with respect to variants of the TALBP. Different variants of TALBP reported are: TALBP (Tang et al. 2015; Yuan et al. 2015), stochastic TALBP (Özcan 2010), mixed-model TALBP (Özcan and Toklu 2009), parallel TALBP (Kucukkoc and Zhang 2014; Özcan 2010), and robotic TALBP (Li et al. 2016a; Li et al. 2016b). Among variants, robotic TALBP is similar to the problem considered in this study (Li et al. 2016a). To solve this problem, Li et al. (2016a) develop a co-evolutionary algorithm to address task allocation and robot assignment respectively. Li et al. (2016b) propose a restarted simulated annealing algorithm to solve multi-objective robotic TALBP.

Regarding worker assignment, Miralles et al. (2007) consider sheltered work centers in assembly line balancing and tested the model using a case study. Chaves et al. (2007) present two clustering search approaches and generate a set of benchmark problems. Chaves et al. (2009) present a hybrid-clustering search to solve the reported benchmark problems. Miralles et al. (2008) present a branch-and-bound approach with three search strategies. Blum and Miralles (2011) develop an iterated beam search method, and this new method outperforms the clustering search algorithm reported in Chaves et al. (2009). Moreira et al. (2012) introduce a constructive heuristic based on task and worker priority rules, and this heuristic is embedded in a hybrid genetic algorithm. Mutlu et al. (2013) develop an iterated genetic algorithm with an iterated local search for determining the worker assignment. Borba and Ritt (2014) employ a heuristic algorithm based on beam search and task-oriented branch-and-bound algorithm to solve this problem. Vilà and Pereira (2014) present a station-oriented branch-and-bound algorithm with new lower bounds, reductions and dominance rules. This new branch-and-bound algorithm yields state-of-the-art results for assembly line worker assignment and balancing problem (ALWABP) with the cycle time minimization. Moreira et al. (2015b) address ALWABP to minimize the number of workstations and present a constructive insertion heuristic. Ramezanian and Ezzatpanah (2015) extend the ALWABP into mixed-model assembly lines and employ an imperialist competitive algorithm. More recently, Sungur and Yavuz (2015) report the hierarchical worker assignment and present an integer linear programming model to address it. Ritt et al. (2016) consider uncertain worker availability in ALWABP and propose local search heuristics. Zacharia and Nearchou (2016) tackle the bi-objective ALWABP using a multi-objective evolutionary algorithm to minimize the cycle time and smoothness index. Akyol and Baykasoğlu (2016) solve ALWABP using a multiple-rule-based constructive randomized search (MRBCRS) algorithm. Thirty-nine task priority rules and four worker priority rules are defined. Performance of the proposed MRBCRS is compared with the relevant literature on benchmark data. A comparative study is conducted to show that the proposed MRBCRS is very effective. All the above-mentioned literature is related to one-sided assembly lines, and in the literature, only one work reports on worker assignment on two-sided assembly lines by Janardhanan et al. (2018); they utilize three artificial bee colony algorithms to solve the problem.

The following literature presents work which is similar to ALWABP. Fattahi et al. (2016) propose a multi-objective mixed-model two-sided assembly line balancing and worker assignment with different skills. A particle swarm optimization (PSO) algorithm is developed to solve it. They compared the performance of PSO with simulated annealing (SA) algorithm based on several benchmark problems. Recently, Roshani and Giglio (2017) addressed a new problem named Multi-manned Assembly Line Balancing Problem (MALBP) in which there is the possibility of assigning more than one operator to each workstation according to the product features with the objective of minimizing cycle time for a given number of workstations; they developed a new mathematical model for solving the proposed problem. Most of the work considered workers heterogeneity in assembly lines based on the motivation of sheltered work centers for the disabled. Few researchers had started to look at the situation faced in assembly lines in the general industrial park in the presence of worker heterogeneity. Moreira et al. (2017) use Miltenburg's regularity criterion and cycle time as metrics for integration of workers and productivity, respectively. They develop a math model and heuristics for a line balancing problem with these two goals, and results are reported from a set of computational experiments.

Although the robotic two-sided assembly line balancing problem (RTALBP) (Li et al. 2016a) is quite similar to the considered problem, the workers and robots have different features and the need to study TALWABP separately is important for the following reasons: The worker can be assigned to any workstation, and the position of a heavy robot is fixed once it is installed. There are ergonomic considerations for workers to operate on an assembly line, while in the case of robots, maintenance and service might be required. These differences make it worthwhile to study the TALWABP separately.

From the literature review, and to authors' best knowledge, there has been limited research reported on TALWABP with cycle time minimization. This research presents for the first time a mathematical model to describe the TALWABP and develops an effective meta-heuristic algorithm to solve large-sized problems.

3. Mathematical formulation

This section presents the mathematical model for the TALBWAP to minimize cycle time and the assumptions considered in this work. Mathematical models is one of the commonly used methods for real life problems to be constructed as mathematical equations. The reason to use mathematical modelling is to better understand complex real life problems (Al-Janabi and Alwan 2017). These assumptions are considered based on Miralles et al. (2007) and Blum and Miralles (2011). In the considered two-sided assembly line, it is assumed that a single product is assembled with workers having different skills. The operation times for tasks depend on the assigned workers. It is assumed that only one worker can be assigned to a workstation and worker can be allocated to any workstation. The number of workers is equal to the number of workstations. In this study, setup times between tasks, work-in-process inventory, and parallel workstation are not considered.

The following subsections introduce the mathematical model in detail and present the utilized parameters and indices.

3.1 Notation description

This section presents the indices, parameters, decision variables and indicator variables utilized in the mathematical model of the considered problem.

A) Indices, and parameters:

i, h, p: A task.
j, g: A mated-station.
k, l: A side of the line; k=1 for the left side and k=2 for the right side.
w: A worker.
nt: Number of tasks.
nm: Number of mated-stations.
ns: Number of stations, $ns = 2 \times nm$.
nw: Number of workers, $nw = 2 \times nm$.
I: Set of tasks, $\mathrm{I} = \{1, 2, \cdots, i, \cdots, nt\}$.
J: Set of mated-stations, $\mathrm{J} = \{1, 2, \cdots, j, \cdots, nm\}$.
W: Set of workers, $\mathrm{W} = \{1, 2, \cdots, w, \cdots, nw\}$.
tw_{iw}: Operation time of task i by worker w.
AL: Set of tasks with left direction, $\mathrm{AL} \subseteq \mathrm{I}$.
AR: Set of tasks with right direction, $\mathrm{AR} \subseteq \mathrm{I}$.
AE: Set of tasks with either direction, $\mathrm{AE} \subseteq \mathrm{I}$.
P_0: Set of tasks which have no immediate predecessors.
$\mathrm{P}(i)$: Set of immediate predecessors of task i.
$\mathrm{P}_a(i)$: Set of all predecessors of task i.
$\mathrm{S}(i)$: Set of immediate successors of task i.
$\mathrm{S}_a(i)$: Set of all successors of task i.
M: A larger positive number, and $tw_{iw} = M$ when task i cannot be operated by worker w.
ψ: A very large positive number.
$\mathrm{C}(i)$: Set of tasks whose directions are opposite to the direction of task i;$\mathrm{C}(i) =$

$$\begin{cases} AL & \text{if } i \in AR \\ AR & \text{if } i \in AL\,. \\ \emptyset & \text{if } i \in AE \end{cases}$$

K(i): Set of integers indicating directions of the task i; $K(i) = \begin{cases} 1 & \text{if } i \in AR \\ 2 & \text{if } i \in AL. \\ 1,2 & \text{if } i \in AE \end{cases}$

B) Decision variables and indicator variables:

CT: Cycle time.

x_{ij}: 1, if task i is assigned to mated-workstation j; 0, otherwise.

w_{ik}: 1, if task i is assigned to side k; 0, otherwise.

y_{wjk}: 1, if worker r is assigned to mated-station j at side k; 0, otherwise.

t_i: Operation time of task i by one allocated worker.

t_i^s: Starting time of task i.

z_{ip}: 1, if task i is assigned earlier than task p in the same workstation; 0, otherwise.

3.2 Mathematical model for TALWABP

Based on the model presented in Borba and Ritt (2014) and Li et al. (2016a), the mathematical model for the considered TALWABP is developed with the cycle time minimization criterion and presented below:

$$\text{Minimize} \quad \text{CT} \tag{1}$$

Subject to:

$$\sum\nolimits_{j \in J} x_{ij} = 1 \quad \forall i \in I \tag{2}$$

$$\sum\nolimits_{k \in K(i)} w_{ik} = 1 \quad \forall i \in I \tag{3}$$

$$\sum\nolimits_{j \in J} j \cdot x_{ij} \geq \sum\nolimits_{g \in J} g \cdot x_{hj} \quad \forall i \in I - P_0, h \in P(i) \tag{4}$$

$$t_i^s + \psi(1 - x_{ij}) + \psi(1 - x_{hj}) \geq t_h^s + t_h \quad \forall i \in I - P_0, h \in P(i), j \in J \tag{5}$$

$$\begin{gathered} t_p^s + \psi(1 - x_{ij}) + \psi(1 - w_{ik}) + \psi(1 - x_{pj}) + \psi(1 - w_{pk}) + \psi(1 - z_{ip}) \\ \geq t_i^s + t_i \\ \forall i \in I, p \in \{c | c \in I - (P_a(i) \cup S_a(i) \cup C(i)) \text{ and } i < c\}, j \in J, k \\ \in K(i) \cap K(p) \end{gathered} \tag{6}$$

$$\begin{gathered} t_i^s + \psi(1 - x_{ij}) + \psi(1 - w_{ik}) + \psi(1 - x_{pj}) + \psi(1 - w_{pk}) + \psi \cdot z_{ip} \\ \geq t_p^s + t_p \\ \forall i \in I, p \in \{c | c \in I - (P_a(i) \cup S_a(i) \cup C(i)) \text{ and } i < c\}, j \in J, k \\ \in K(i) \cap K(p) \end{gathered} \tag{7}$$

$$t_i^s \geq 0 \quad \forall i \in I \tag{8}$$

$$t_i + \psi(1 - x_{ij}) + \psi(1 - w_{ik}) \geq \sum_{w=1}^{nw} tw_{iw} \cdot y_{wjk} \quad \forall i \in I, j \in J, k \in K(i) \tag{9}$$

$$t_i < M \tag{10}$$

$$t_i^s + t_i \leq CT \quad \forall i \in I \tag{11}$$

$$\sum_{w=1}^{nw} y_{wjk} = 1 \qquad \forall j \in J, k = 1,2 \tag{12}$$

$$\sum_{j=1}^{nm} \sum_{k=1}^{2} y_{wjk} = 1 \qquad \forall w \in W \tag{13}$$

The objective function in expression (1) optimizes cycle time. Constraint (2) and constraint (3) are the occurrence constraint that indicates that each task must be allocated to a workstation. Constraint (4) deals with precedence constraint ensuring that the predecessors of task i are allocated to the former or the same mated-station. Constraints (5-7) handle the sequence of tasks on same mated-stations. If task h is the immediate predecessor of task i and they are allocated to the same mated-station, the beginning time of task i is larger than or equal to the finishing time of task h. Constraints (6-7) take effect when a pair of tasks i and h have no relationship and they are allocated to the same side of the same mated-station. If task i is allocated earlier than task p, the beginning time of task p is larger than or equal to the finishing time of task i expressed in constraint (6). Otherwise, the beginning time of task i is larger than or equal to the finishing time of task p expressed in constraint (7) when task p is allocated earlier than task i. Expression (8) guarantees that the starting time of a task is larger than or equal to 0.0, and expression (9) calculates the operation time of task i by the corresponding worker. Constraint (10) ensures that a task is operated by another worker when task i cannot be operated by worker w. Constraint (11) is the cycle time constraint that makes sure that each task is completed within the desired cycle time. Expressions (12-13) constrains the worker assignment, where the expression (12) indicates that each workstation is assigned with only one worker and expression (13) ensures that each worker is assigned to only one workstation.

Recall that, different from the model in Li et al. (2016a), this model utilizes two new decision variables, x_{ij} and w_{ik}, to determine the task assignment. Furthermore, the proposed model also employs one decision variable, t_i, to indicate the operation time of task i by the worker to whom this task is assigned. This new model finds the optimal solutions faster than that in Li et al. (2016a); a detailed comparison is presented in Section 5.

According to the classification in Boysen et al. (2007), a one-sided assembly line with worker assignment can be classified as $[link, cum, pa|equip\ \ |c]$ (Moreira et al. 2015b), and the problem considered in this paper for a two-sided assembly line with workers assigned with the objective of minimizing cycle time is classified as $[link, cum, pa|pwork^2, equip\ \ |c]$.

4. Proposed methodology

To cope with complex decision making process in different areas such as manufacturing, web application can be utilized (Fahmi et al. 2017a; Patel et al. 2015). Assembly line balancing problem is one such complex decision making problems. Migrating bird optimization, a metaheuristic algorithm, is employed to find optimal or near-optimal solutions due to the NP-hard nature of the problem. MBO simulates birds' migration behavior of flying in V-shape and is based on local search (Duman et al. 2012). MBO is selected to solve this problem due to better performance over other algorithms in solving problems of a similar type (Duman et al. 2012; Gao and Pan 2016; Zhang et al. 2017).

In an MBO algorithm, one bird is leading the whole flock, and the remaining birds are on the left and right sides, and within each side, the birds follow in a line (Duman et al. 2012). The original MBO has four parameters: *n* as the number of individuals, *k* as the number of neighbor solutions, *x* as the number of neighbor solutions shared with the next individual, and *m* as the number of tours before replacing the leader. MBO starts with initializing *n* individuals, and subsequently, a main cycle repeats until a termination criterion is met. This cycle has three segments that are executed in sequence: leader improvement, block improvement, and leader replacement. Firstly, in the leader improvement, the leader individual tries to improve itself by producing and evaluating *k* neighbor solutions. If improvement is achieved, the incumbent leader individual is updated. Afterward, in the block improvement, the individuals try to improve themselves by evaluating *x* unused best neighbor solutions of the front solution (referred to as benefit mechanism) in the same side and its (*k*-*x*) neighbor solutions. Finally, the leader replacement is executed after carrying out leader improvement and block improvement *m* consecutive times, where the leading individual is removed to the ending of one side and one of the individuals following the leader is moved forward as the new leader. In this research, several improvements are employed to enhance the performance of the MBO. The following sections clarify the detailed segments of the proposed MBO.

4.1 Encoding and decoding

Based on contributions reported in Janardhanan et al. (2018) and Li et al. (2016a), this research utilizes two vectors for encoding: worker assignment vector and task permutation vector. Worker assignment vector is a $2 \times nm$ vector corresponding to the workers assigned to workstations. Suppose the worker assignment vector is {1, 2, 4}, worker 1 is assigned to workstation (1,1), worker 2 is assigned to workstation (1, 2), and finally, worker 3 is assigned to workstation (2,2). The task permutation vector corresponds to the sequence of allocating tasks, and tasks on the former positions of the task permutation should be allocated first. Suppose that the task permutation vector is {1, 2, 4, 3, 5, 6, 8, 9, 7, 10, 11, 12}; tasks 1, 2, and 4 should be allocated first when they are assignable.

To transfer the two vectors into a feasible solution, an initial cycle time and a decoding procedure are necessary. As reported in Janardhanan et al. (2018) and Li et al. (2016a), this paper proposes the following iteration mechanism for cycle time update and decoding procedure and proposes two procedures (Procedure 1 and Procedure 2). This iteration mechanism ensures that the initial cycle time reduces gradually, and all individuals are evaluated under same initial cycle time. Decoding scheme aims to balance the workloads on two sides of a mated-station using Step 4, where the side with the larger remaining capacity is selected. The proposed decoding scheme also helps to reduce sequence-dependent idle times using Step 5, where assignable tasks can be started at the earliest starting time of the selected workstation. Section 4.3 illustrates the proposed encoding and decoding.

Procedure 1: Cycle time update procedure

***Step 1*: Set initial cycle time (CT) to a large number and best cycle time CT_N as $\mathrm{CT}_N = CT$.

***Step 2*: Decode the individuals using CT as the initial cycle time.

***Step 3*: Update CT with $\mathrm{CT} \leftarrow \mathrm{CT}_N - 1$ and re-decode the individuals using CT as the initial cycle time when a smaller best cycle time CT_N is achieved.

***Step 4*: Update the individuals and go to Step 2 until a termination criterion is satisfied.

Procedure 2: Decoding procedure

***Step 1*: Achieve assignable task sets for both sides.

***Step 2*: If both assignable task sets are empty and all tasks are allocated, terminate this procedure. If both assignable task sets are empty and some tasks remain unallocated, open a new mated-station and go to Step1.

If at least one assignable task set is not empty, go to Step 3.

***Step 3*: Select the side whose assignable task set is not empty if only one assignable task set is not empty; otherwise, go to Step 4.

***Step 4*: Select the side with a larger remaining capacity (smaller ending time) or the left side by default when the remaining capacities of both sides are equal if both assignable task sets are not empty.

***Step 5*: Remove the tasks which result in sequence-dependent idle times from the corresponding assignable task set if some assignable tasks can be started at the earliest starting time of the selected workstation.

***Step 6*: Select the assignable task, which is on the former position of the task permutation vector; allocate the task to the selected side, update the remaining capacity of the selected side, and finally, go to Step 1.

4.2 Proposed MBO method

This research introduces several improvements to enhance the performance of the proposed MBO, and the main procedure is illustrated in Figure 3 where α is the number of consecutive iterations before executing the restart mechanism. It should be noted that neighbor solution is generated by executing the neighbor operator based on the scheme followed in Janardhanan et al. (2018) and described in this section. Neighborhood structures have a great impact on the performance of neighborhood-based metaheuristic algorithms. In the reported literature for solving similar problems (Li et al. 2016a; Polat et al. 2016; Roshani and Giglio 2017), there are many neighborhood structures. Among these, insert operation and swap operation are two simple methods, but they have achieved promising results (Roshani and Giglio 2017) and are quite effective in our initial experiments. Hence, this research applies both insert operation and swap operation as presented in Figure 2. The utilization of two neighbor operations rather than one increases the search space and helps the algorithm to escape being trapped into local optima. As there are two vectors to be optimized, this research first selects one vector and then selects one of the neighbor structures randomly to modify the selected vector.

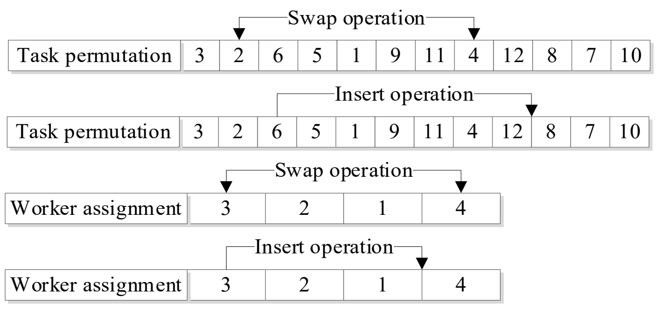

Figure 2. Neighbor operators

```
//Initialization
Initialize $n$ individuals randomly;
  While (termination criterion is not satisfied) do
    For $i$=1 to $m$ do
      //Leader improvement
      For $j=1$ to $k$ do
        Obtain a neighbor of the leader;
        Update the incumbent one with the new one when the better or the same fitness
        is achieved and the neighbor solution is different from the incumbent one.
      Endfor
      Set a large positive number as the fitness of the individual whose original
      fitness is equal to that of the incumbent one.
      // Block improvement
      For each remaining individual
        For $j$=1 to ($k$-$x$) do
          Obtain a neighbor of this individual;
          Update the incumbent one with the new one when the better or the same
          fitness is achieved and the neighbor solution is different form the incumbent
          one.
        Endfor
        Replace the incumbent individual with the best one from the ($k$-$x$) neighbors
        and $x$ unused neighbor solution of the individual in front (benefit mechanism)
        when the better or the same fitness is achieved and the neighbor solution is
        different from the incumbent one.
        Set a large positive number as the fitness of the individual whose original
        fitness is equal to that of the incumbent one.
      Endfor
      // Restart mechanism
      If (the best cycle times has not been updated for $\alpha$ iterative times)
      execute the restart mechanism;
      Endif
    Endfor
    //Leader replacement
    Move the current leader to the end and forward one of the following individuals as
    the new leader.
  Endwhile
```

Figure 3. The procedure of the improved MBO

Worker assignment vector or task permutation vector are randomly selected, and later, insert operation or swap operation is randomly selected to modify the selected vector. The procedure is like that of original MBO, but it has several advancements on leader and block improvement, and it also employs a restart mechanism. Specifically, the incumbent individual is replaced with new neighbor solutions once better fitness or the same fitness is achieved rather than evaluating all the neighbor solutions. This method increases the search speed and ensures algorithm searches around the most promising area of the search space. Additionally, this research sets a large positive number as the fitness of the individual whose original fitness is equal to that of the incumbent one. This modification tries to preserve the diversity of the population and avoid premature convergence of the proposed method.

In the case of restart mechanism, this research employs it when the best cycle time has not been updated for α iterative times, where α is a new parameter and is set to 2 based on the parameter calibration. Instead of re-initializing all individuals, this research proposes a method similar to the scout phase in artificial bee colony algorithm (Janardhanan et al. 2018) in which only one individual is re-initialized. The duplicated individual, or the individual with the worst fitness, or the individual who has survived for the longest time is selected as the abandoned individual, and this abandoned individual is replaced with the best one among a set of β neighbor solutions of it, and each neighbor solution is achieved by executing neighbor operator on the incumbent solution γ times.

4.3 An illustrated numerical example

This section illustrates a numerical example with 12 tasks, two mated-stations, and four workers to highlight the features of considered TALWABP. Table 1 illustrates preferred task directions, precedence relationships, and operation times of tasks.

Table 1. Preferred directions, precedence relationships, and operation times of tasks

Tasks	Direction	Successors	Operation times			
			Worker 1	Worker 2	Worker 3	Worker 4
1	L	4	2	6	-	-
2	R	5	3	4	3	7
3	E	6	2	2	3	4
4	L	7	3	4	9	6
5	E	7, 8, 9	1	-	1	-
6	L	9	1	-	1	-
7	E	10	3	6	3	-
8	R	10	3	2	3	-
9	E	11	2	-	3	3
10	E	-	2	-	1	5
11	E	12	2	5	1	5
12	R	-	1	2	3	-

Suppose that the initial cycle time is 8 units and the worker assignment and task permutation vectors are those presented in Figure 4; achieved worker assignment and task assignment are presented in Figure 4. It is observed that worker assignment vector is 1, 2, 4, and 3, and thus, workers 1, 2, 4, and 3 are assigned to workstations (1, 1), (1, 2), (2, 1), and (2, 2). Tasks 1, 2, 4, and 3 are in former positions of task permutation, they are first allocated and thus they are all allocated to mated-station 1. Based on operation times by workers as reported in Table 1, the operation times of tasks 1, 4, 5, and 6 by worker 1 is 2, 3, 1, and 1, and completion time of workstation (1, 1) is 7. Subsequently, the completion times of other workstations are determined: 8 for workstation (1, 2), 8 for workstation (2, 1), and 7 for workstation (2, 2). Clearly, achieved cycle time is 8 which is the largest value of the completion times of workstations.

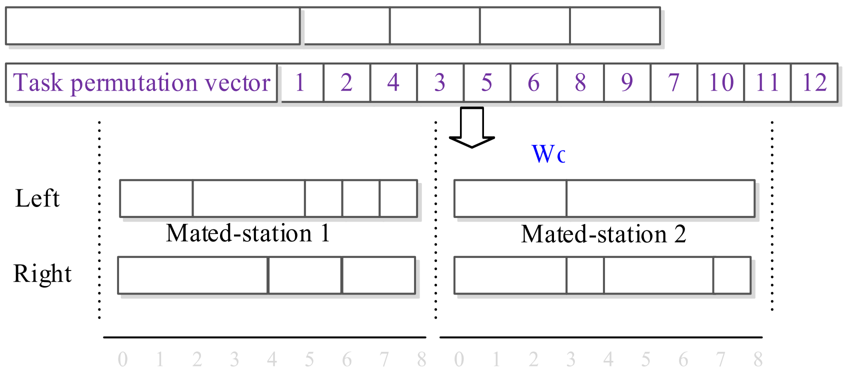

Figure 4. Worker assignment and task assignment for the numerical example

5. Computational study

This section presents detailed experiments conducted to test the performance of the proposed methodology. Experimental design is first introduced to explain tested problems and compared methods. The conducted computational study is presented along with statistical analysis.

5.1 Experimental design

This research utilizes two datasets: the dataset in Janardhanan et al. (2018) and a newly generated data set in this paper. For the dataset in Janardhanan et al. (2018), precedence relations and preferred directions are taken from Kim et al. (2009) and Li et al. (2016c), and operation times of tasks by workers are produced following the method used in Chaves et al. (2007). Table 2 summarizes these instances. There are two variabilities of the operation times and two percentages of task-worker incompatibilities. There are 39 benchmark instances for TALBP (Li et al. 2016c), and due to additional features for this problem, there are 156 instances for the considered TALWABP. The new instances are generated by repeating the aforementioned method 10 times, and thus, there are 1,560 new instances.

Table 2. Summary of the tested benchmarks

Problem	nt	Num. of cases	nm	Time variabilities	Incompatibilities
P9	9	8	2,3	Low, high	Low, high
P12	12	16	2,3,4,5	Low, high	Low, high
P16	16	16	2,3,4,5	Low, high	Low, high
P24	24	16	2,3,4,5	Low, high	Low, high
P65	65	20	4,5,6,7,8	Low, high	Low, high
P148	148	36	4,5,6,7,8,9,10,11,12	Low, high	Low, high
P205	205	44	4,5,6,7,8,9,10,11,12,13,14	Low, high	Low, high

To evaluate the performance of the proposed MBO, this research carries out two comparative studies on both datasets. For the first dataset, MBO is compared with six algorithms: partial swarm optimization (PSO) (Al_Janabi et al. 2018), genetic algorithm (GA), three artificial bee colony algorithms (ABC1, ABC2, and ABC3) (Janardhanan et al. 2018), and a simulated annealing algorithm (SA) (Özcan and Toklu 2009). It should be noted that these six methods are selected as they are the only available applied methods for the considered problems. For the second dataset, MBO is compared with three algorithms: co-evolutionary particle swarm optimization (CoPSO), discrete cuckoo search algorithm (DCS), and co-evolutionary cuckoo search (CoCS). These three methodologies are selected as they have been applied to solve robotic TALBP (similar to the considered problem), and they have achieved state-of-the-art results by outperforming SA, GA, and PSO. In short, in this comparative study, the proposed MBO is compared with nine metaheuristics, including most of the methods in TALWABP and the three best methodologies in robotic TALBP. To observe the performance of tested method under different termination criteria, this research utilizes the termination criterion of $\mathrm{nt} \times \mathrm{nt} \times \tau$ milliseconds, similar to ones reported in Janardhanan et al. (2018) and Li et al. (2016a) in which τ is set to 10, 20, and 30 respectively. It is done in this manner so that this expression ensures that large-size instances have more CPU time for execution.

5.2 Computational study

This section first presents computational results obtained using CPLEX solver in General Algebraic Modeling System 23.0 and a set of meta-heuristic algorithms when solving the dataset in Janardhanan et al. (2018). As parameter values have a great impact on the performance of algorithms, this paper first calibrates the tested algorithms using full factor design and multifactor analysis of variance (ANOVA) technique following Li et al. (2016a) and Roshani and Giglio (2017), and many others. Specifically, a largest problem instance (P205 with 6 mated-stations, low time variabilities, and low percentages of task-worker incompatibility) is solved using all the combinations of parameters with termination criterion of $\mathrm{nt} \times \mathrm{nt} \times 10$ milliseconds. For simplicity, the calibration process of ABC3 is provided as an example, and this algorithm has one parameter: population size. The initial levels of this parameter are set to 10, 20, 30, and 40 determined by initial experiments. Then, the algorithm runs these instances for 10 repeated times, and the achieved cycle times are transferred into relative percentage deviation (RPD) using expression (11), where $CTsome$ is the cycle time by one combination and $CTBest$ is cycle time achieved by all combinations. Subsequently, ANOVA technique is utilized to analyze these RPD values and determine the selected parameter's values. Regarding ANOVA results, one-parameter values have a statistically important impact on the performance of the algorithm if the corresponding P-value is smaller than a pre-determined number (0.05). For simplicity, Figure 5 illustrates the mean plot of the population size with 95% Tukey's honestly significant difference (HSD) confidence intervals. It is observed that the value of 40 has the best performance, and it is selected as the population size. As for the algorithms with several parameters, a multi-factor ANOVA test is carried out and the values of the parameters are determined in the increasing order of P-values. The detailed calibration process and selected parameters are omitted due to space considerations, but they are available upon request.

$$RPD = 100 \cdot (CT_{some} - CT_{Best})/CT_{Best} \tag{11}$$

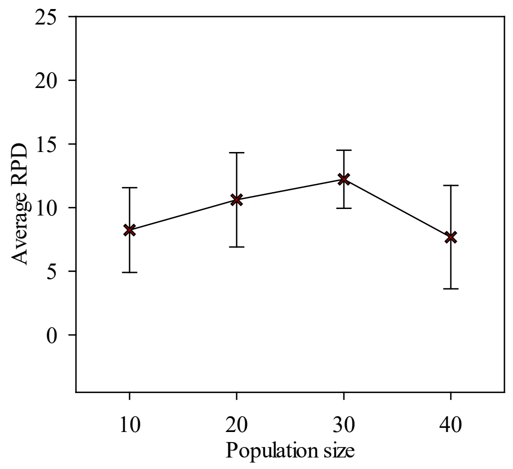

Figure 5. Mean plot and 95% Tukey's HSD confidence intervals of population size

After determining parameter values, all algorithms are solved for the first dataset instances 10 iterative times. As CPLEX could not solve large-size problems optimally in acceptable CPU time, this research first presents the computational results for small-size problems (P9, P12, and P16) in Table 3 where the results of only three of the five methods are presented. In Table 3, there are four cases for each mated-station number. *Num of OPT* refers to the number of cases solved optimally by the proposed model and the model in Li et al. (2016a); numbers under the algorithms are the number of cases solved to optimality within 10 times by the algorithm, and CPU(s) is the average computational time in seconds. It is very clear that all tested cases could be solved to optimality by CPLEX and by algorithms, although CPLEX needed more CPU time for P16 with four and five mated-stations. And the proposed model achieves the same results with less running time for most instances, indicating that the proposed model outperforms the published one in search speed. This study verifies the superiority of the algorithms over CPLEX solver in solving large-size instances.

Table 3. Computational results for small-size problems

Problem	nm	Num. of cases	New model		Model in Li et al. (2016a)		Implemented algorithms			
			Num of OPT	CPU(s)	Num of OPT	CPU(s)	ABC2	ABC3	MBO	CPU(s)
P9	2	4	4	0.13	4	0.19	4	4	4	2.43
P9	3	4	4	0.17	4	0.21	4	4	4	2.43
P12	2	4	4	0.29	4	0.38	4	4	4	4.32
P12	3	4	4	0.99	4	1.11	4	4	4	4.32
P12	4	4	4	2.75	4	2.97	4	4	4	4.32
P12	5	4	4	1.64	4	3.57	4	4	4	4.32
P16	2	4	4	1.03	4	1.57	4	4	4	7.68
P16	3	4	4	5.70	4	19.93	4	4	4	7.68
P16	4	4	4	93.12	4	62.21	4	4	4	7.68
P16	5	4	4	72.31	4	300.68	4	4	4	7.68

A more comprehensive study is presented in Table 4 where average RPD values by tested algorithm are exhibited under three termination criteria. Recall that Cbest is the best cycle time achieved by all the algorithms when calculating RPD values. Each cell contains the average value of the RPD values for several cases for 10 repetitions. From this table, it is observed that

MBO has smallest overall average RPD of 8.70, 6.99, and 6.13 when $\tau = 10, 20,$ and 30 respectively. In other words, MBO is the best performer among the seven tested algorithms under three termination criteria. Among other methods, ABC3, ABC2, SA, and ABC1 rank second, third, fourth, and fifth when $\tau = 10$ and $\tau = 20$. These results coincide with those of Janardhanan et al. (2018). SA ranks fifth and ABC1 ranks fourth when $\tau = 30$. This computational study suggests that the proposed MBO is the most effective methodology among the tested methods for considered TALWABP.

Table 4. Average RPD values by implemented algorithms

Problem	Num. of cases	Average relative percentage deviation							CPU time(s)
		SA	PSO	GA	ABC1	ABC2	ABC3	MBO	
$\tau = 10$									
P9	8	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.81
P12	16	0.31	1.88	2.66	1.09	0.63	0.78	0.00	1.44
P16	16	0.78	1.74	1.08	0.38	0.76	0.43	0.36	2.56
P24	16	2.92	7.35	5.30	2.44	3.30	2.65	1.65	5.76
P65	20	14.58	42.35	18.93	13.22	13.61	12.98	10.04	42.25
P148	36	20.87	51.81	28.26	23.32	15.99	15.84	14.37	219.04
P205	44	18.41	55.00	23.32	19.85	15.69	15.67	13.78	420.25
Average RPI		12.29	34.02	16.45	13.08	10.34	10.14	**8.70**	-
$\tau = 20$									
P9	8	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.62
P12	16	0.31	0.52	2.27	0.78	0.47	0.63	0.00	2.88
P16	16	0.78	0.97	0.98	0.13	0.38	0.14	0.00	5.12
P24	16	2.92	6.10	4.93	2.17	3.05	2.18	1.39	11.52
P65	20	13.59	38.92	16.89	10.57	12.16	11.89	8.50	84.50
P148	36	16.88	48.81	25.53	18.92	12.54	12.56	11.62	438.08
P205	44	15.53	51.41	20.90	15.85	13.24	12.92	10.92	840.50
Average RPI		10.43	31.53	14.79	10.51	8.59	8.37	**6.99**	-
$\tau = 30$									
P9	8	0.00	0.00	0.00	0.00	0.00	0.00	0.00	2.43
P12	16	0.31	0.21	2.03	0.63	0.31	0.63	0.00	4.32
P16	16	0.78	0.65	0.69	0.13	0.22	0.14	0.00	7.68
P24	16	2.92	5.58	4.38	1.99	2.70	2.00	1.31	17.28
P65	20	13.21	36.98	15.87	9.47	11.66	11.16	7.89	126.75
P148	36	14.83	47.10	24.32	16.54	11.01	10.93	10.10	657.12
P205	44	14.25	49.29	19.43	13.83	11.84	11.49	9.39	1260.75
Average RPI		9.55	30.17	13.86	9.21	7.71	7.48	**6.13**	-

To ascertain that proposed MBO is statistically better than others, this research carries out a multifactorial ANOVA test in which algorithms and termination criteria ($\tau = 10, 20, 30$) are regarded as factors. As algorithms have diverse performances on different instances, this research utilizes average RPD of all tested instances in one run as the response variable, similar to the work reported by Tang et al. (2017). There are 10 average RPD values for each combination as algorithms solve each case 10 times. However, homogeneity of variance is slightly violated as seen by results of the initial ANOVA test. Hence, this research carries out both an ANOVA test and a Friedman test, where the ANOVA test is utilized to analyze interactions between the algorithms and computational times and the Friedman test is conducted to ascertain the results by ANOVA test. ANOVA shows that there are statistically significant differences between tested algorithms, three termination criteria, and interaction of two above factors. The Friedman test suggests that there are statistically significant differences between tested algorithms under three termination criteria. This paper illustrates the means plot of the interaction of two factors in Figure 6, where only the best five methods are plotted for a better vision.

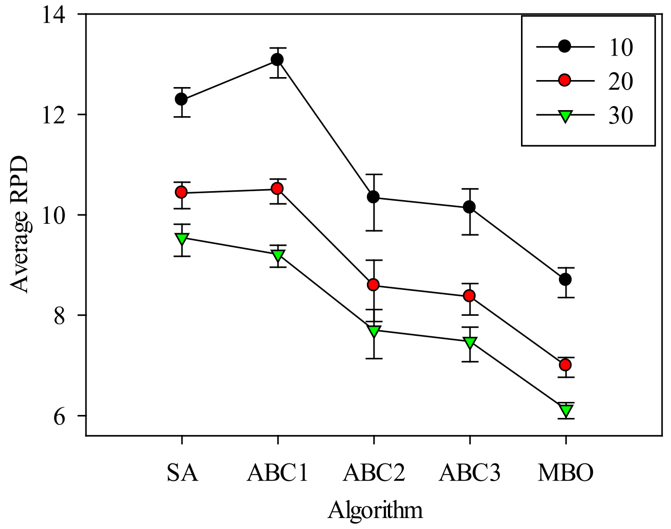

Figure 6. Means plot and 95% Tukey HSD confidence intervals for interactions between algorithms and elapsed CPU times

Figure 6 provides a direct and clear observation of the performance of tested algorithms. MBO is the best performer when $\tau = 10, 20, 30$ and ABC3 is the second-best performer. It is also observed that the values of average RPD decline as elapsed CPU time increases. As there is no overlap between the confidence interval of MBO and other methods, it is sufficient to state that the proposed MBO is statistically better.

For the second dataset, each instance is solved once, and RPD is again applied to transfer the achieved cycle times. This section also carries out the multifactorial ANOVA test to evaluate the performances of the tested algorithms. As this new dataset contains 10 groups of instances by repeating the method in Janardhanan et al. (2018) 10 times, the ANOVA test utilizes the average RPD of the instances in one group in one run as the response variable, resulting in 10 average RPD values. Again, algorithms and termination criteria ($\tau = 10, 20, 30$) are regarded as factors, and Figure 6 depicts the means plot of interaction of two factors. This figure suggests that MBO is the best performer when $\tau = 10, 20, 30$, DCS is the second-best performer, and CoPSO is the worst performer in solving this new dataset. It is also observed that no overlap exists between the confidence interval of MBO and other algorithms. In summary, MBO outperforms other methods statistically and is the best performer among the tested algorithms.

The algorithms and proposed approach for solving a two-sided assembly line worker assignment and balancing problem (TALWABP) are very beneficial for real-time decision-making and can be used by production managers in their production planning. In most cases, it is not possible to carry out real-time experiments on the existing system due to the cost factor involved. It helps managers to estimate the resource required for the considered assembly line based on their performance. The study can also help managers to take decisions for fully utilizing resources.

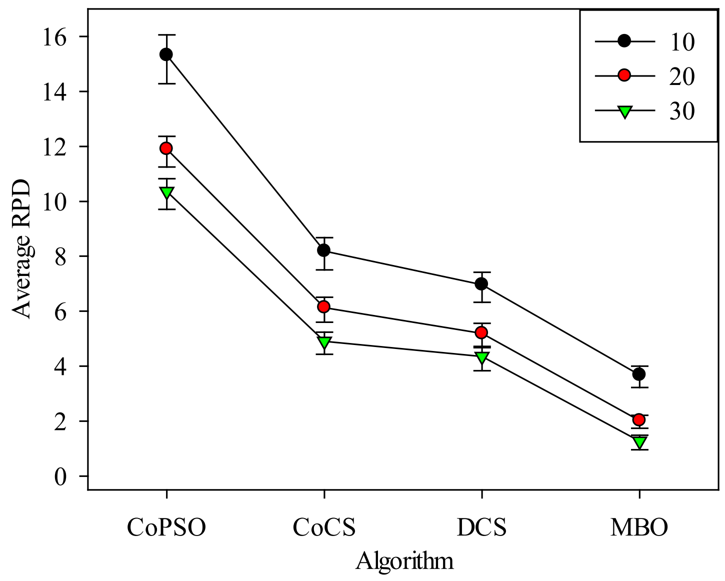

Figure 6. Means plot and 95% Tukey HSD confidence intervals for interactions between algorithms and running times

6. Conclusion and future research

Due to workers' different skills, worker assignment needs to be optimized to improve assembly line efficiency. This research tackles an integrated two-sided assembly line worker assignment and balancing problem with the objective of minimizing cycle time. A mixed-integer programming model is developed to formulate the considered problem. Migrating birds optimization algorithm is proposed to solve large-size problem instances, where a restart mechanism is developed to help the proposed MBO escape from being trapped into local optima. The proposed model is encoded into CPLEX solver, and it is capable of tackling small-size instances optimally. The proposed algorithm is tested on 156 instances and is compared with four algorithms. Computational and statistical analysis results demonstrate that MBO is the best performer and is statistically better than other methods under three termination criteria.

A developed model helps to improve the efficiency of the workers, improve the productivity of the assembly line, and reduce throughput time or cycle time, and hence, it reduces assembly cost and makes industries more competitive. In the future, the considered problem could be extended by taking multiple constraints and stochastic operation times into account. More recent and effective methods, such as co-evolutionary algorithms, are also suggested to be applied to this new problem.

Compliance with ethical standards

Conflict of interest: The authors declare that they have no conflict of interest.

Ethical approval: This article does not contain any studies with human participants or animals performed by any of the authors.

References

Akyol SD, Baykasoğlu A (2016) A multiple-rule based constructive randomized search algorithm for solving assembly line worker assignment and balancing problem Journal of Intelligent Manufacturing:1-17

Al-Janabi S, Alwan E Soft Mathematical System to Solve Black Box Problem through Development the FARB Based on Hyperbolic and Polynomial Functions. In: Developments in eSystems Engineering (DeSE), 2017 10th International Conference on, 2017. IEEE, pp 37-42

Al_Janabi S (2018) Smart system to create an optimal higher education environment using IDA and IOTs International Journal of Computers and Applications:1-16 doi:https://doi.org/10.1080/1206212X.2018.1512460

Al_Janabi S, Al_Shourbaji I, Salman MA (2018) Assessing the suitability of soft computing approaches for forest fires prediction Applied computing and informatics 14:214-224

Amin F, Fahmi A, Abdullah S (2018) Dealer using a new trapezoidal cubic hesitant fuzzy TOPSIS method and application to group decision-making program Soft Computing doi:https://doi.org/10.1007/s00500-018-3476-3

Bartholdi JJ (1993) Balancing two-sided assembly lines: a case study International Journal of Production Research 31:2447-2461 doi:10.1080/00207549308956868

Baykasoglu A, Dereli T (2008) Two-sided assembly line balancing using an ant-colony-based heuristic The International Journal of Advanced Manufacturing Technology 36:582-588 doi:10.1007/s00170-006-0861-3

Blum C, Miralles C (2011) On solving the assembly line worker assignment and balancing problem via beam search Computers & Operations Research 38:328-339 doi:http://dx.doi.org/10.1016/j.cor.2010.05.008

Borba L, Ritt M (2014) A heuristic and a branch-and-bound algorithm for the Assembly Line Worker Assignment and Balancing Problem Computers & Operations Research 45:87-96 doi:http://dx.doi.org/10.1016/j.cor.2013.12.002

Boysen N, Fliedner M, Scholl A (2007) A classification of assembly line balancing problems European Journal of Operational Research 183:674-693 doi:http://dx.doi.org/10.1016/j.ejor.2006.10.010

Chaves AA, Lorena LAN, Miralles C (2009) Hybrid Metaheuristic for the Assembly Line Worker Assignment and Balancing Problem. In: Blesa MJ, Blum C, Di Gaspero L, Roli A, Sampels M, Schaerf A (eds) Hybrid Metaheuristics: 6th International Workshop, HM 2009, Udine, Italy, October 16-17, 2009. Proceedings. Springer Berlin Heidelberg, Berlin, Heidelberg, pp 1-14. doi:10.1007/978-3-642-04918-7_1

Chaves AA, Miralles C, Lorena LAN Clustering search approach for the assembly line worker assignment and balancing problem. In: Proceedings of the 37th international conference on computers and industrial engineering, Alexandria, Egypt, 2007. pp 1469-1478

Chica M, Cordón Ó, Damas S, Bautista J (2015) Interactive preferences in multiobjective ant colony optimisation for assembly line balancing Soft Computing 19:2891-2903

Dolgui A, Kovalev S, Kovalyov MY, Malyutin S, Soukhal A (2018) Optimal workforce assignment to operations of a paced assembly line European Journal of Operational Research 264:200-211

Duman E, Uysal M, Alkaya AF (2012) Migrating Birds Optimization: A new metaheuristic approach and its performance on quadratic assignment problem Information Sciences 217:65-77 doi:http://dx.doi.org/10.1016/j.ins.2012.06.032

Fahmi A, Abdullah S, Amin F, Ahmed R, Ali A (2018a) Triangular cubic linguistic hesitant fuzzy aggregation operators and their application in group decision making Journal of Intelligent & Fuzzy Systems vol. 34:2401-2416

Fahmi A, Abdullah S, Amin F, Ali A (2017a) Precursor selection for sol–gel synthesis of titanium carbide nanopowders by a new cubic fuzzy multi-attribute group decision-making model Journal of Intelligent Systems doi:https://doi.org/10.1515/jisys-2017-0083

Fahmi A, Abdullah S, Amin F, Ali A (2018b) Weighted average rating (war) method for solving group decision making problem using triangular cubic fuzzy hybrid aggregation (tcfha) Punjab Univ J Math 50:23-34

Fahmi A, Abdullah S, Amin F, Ali A, Ahmad Khan W (2018c) Some geometric operators with triangular cubic linguistic hesitant fuzzy number and their application in group decision-making Journal of Intelligent & Fuzzy Systems vol. 35:2485-2499

Fahmi A, Abdullah S, Amin F, Khan MSA (2018d) Trapezoidal cubic fuzzy number Einstein hybrid weighted averaging operators and its application to decision making Soft Computing doi:https://doi.org/10.1007/s00500-018-3242-6

Fahmi A, Abdullah S, Amin F, Siddiqui N (2017b) Aggregation operators on triangular cubic fuzzy numbers and its application to multi-criteria decision making problems Journal of Intelligent & Fuzzy Systems:1-15

Fahmi A, Amin F, Abdullah S, Ali A (2018e) Cubic fuzzy Einstein aggregation operators and its application to decision-making International Journal of Systems Science 49:2385-2397

Fattahi P, Samoei P, Zandieh M (2016) Simultaneous Multi-Skilled Worker Assignment And Mixed-Model Two-Sided Assembly Line Balancing International Journal of Engineering-Transactions B: Applications 29:211

Gao L, Pan Q-K (2016) A shuffled multi-swarm micro-migrating birds optimizer for a multi-resource-constrained flexible job shop scheduling problem Information Sciences 372:655-676 doi:http://dx.doi.org/10.1016/j.ins.2016.08.046

Janardhanan MN, Li Z, Nielsen P, Tang Q (2018) Artificial bee colony algorithms for two-sided assembly line worker assignment and balancing problem. In: Omatu S, Rodríguez S, Villarrubia G, Faria P, Sitek P, Prieto J (eds) Distributed Computing and Artificial Intelligence, 14th International Conference. Springer International Publishing, Cham, pp 11-18. doi:10.1007/978-3-319-62410-5_2

Kim YK, Song WS, Kim JH (2009) A mathematical model and a genetic algorithm for two-sided assembly line balancing Computers & Operations Research 36:853-865 doi:http://dx.doi.org/10.1016/j.cor.2007.11.003

Kucukkoc I, Zhang DZ (2014) Mathematical model and agent based solution approach for the simultaneous balancing and sequencing of mixed-model parallel two-sided assembly lines International Journal of Production Economics 158:314-333

Lee TO, Kim Y, Kim YK (2001) Two-sided assembly line balancing to maximize work relatedness and slackness Computers & Industrial Engineering 40:273-292 doi:http://dx.doi.org/10.1016/S0360-8352(01)00029-8

Li Z, Janardhanan MN, Tang Q, Nielsen P (2016a) Co-evolutionary particle swarm optimization algorithm for two-sided robotic assembly line balancing problem

Advances in Mechanical Engineering 8:1-14 doi:10.1177/1687814016667907

Li Z, Tang Q, Zhang L (2016b) Minimizing energy consumption and cycle time in two-sided robotic assembly line systems using restarted simulated annealing algorithm Journal of Cleaner Production 135:508-522 doi:http://dx.doi.org/10.1016/j.jclepro.2016.06.131

Li Z, Tang Q, Zhang L (2016c) Minimizing the Cycle Time in Two-Sided Assembly Lines with Assignment Restrictions: Improvements and a Simple Algorithm Mathematical Problems in Engineering 2016:15 doi:10.1155/2016/4536426

Miralles C, García-Sabater JP, Andrés C, Cardos M (2007) Advantages of assembly lines in Sheltered Work Centres for Disabled. A case study International Journal of Production Economics 110:187-197 doi:http://dx.doi.org/10.1016/j.ijpe.2007.02.023

Miralles C, García-Sabater JP, Andrés C, Cardós M (2008) Branch and bound procedures for solving the Assembly Line Worker Assignment and Balancing Problem: Application to Sheltered Work centres for Disabled Discrete Applied Mathematics 156:352-367 doi:http://dx.doi.org/10.1016/j.dam.2005.12.012

Moreira MCO, Cordeau J-F, Costa AM, Laporte G (2015a) Robust assembly line balancing with heterogeneous workers Computers & Industrial Engineering 88:254-263

Moreira MCO, Miralles C, Costa AM (2015b) Model and heuristics for the Assembly Line Worker Integration and Balancing Problem Computers & Operations Research 54:64-73 doi:http://dx.doi.org/10.1016/j.cor.2014.08.021

Moreira MCO, Pastor R, Costa AM, Miralles C (2017) The multi-objective assembly line worker integration and balancing problem of type-2 Computers & Operations Research 82:114-125

Moreira MCO, Ritt M, Costa AM, Chaves AA (2012) Simple heuristics for the assembly line worker assignment and balancing problem J Heuristics 18:505-524 doi:10.1007/s10732-012-9195-5

Mutlu Ö, Polat O, Supciller AA (2013) An iterative genetic algorithm for the assembly line worker assignment and balancing problem of type-II Computers & Operations Research 40:418-426 doi:http://dx.doi.org/10.1016/j.cor.2012.07.010

Oksuz MK, Buyukozkan K, Satoglu SI (2017) U-shaped assembly line worker assignment and balancing problem: A mathematical model and two meta-heuristics Computers & Industrial Engineering 112:246-263

Özbakır L, Tapkan P (2011) Bee colony intelligence in zone constrained two-sided assembly line balancing problem Expert Systems with Applications 38:11947-11957 doi:http://dx.doi.org/10.1016/j.eswa.2011.03.089

Özcan U (2010) Balancing stochastic two-sided assembly lines: A chance-constrained, piecewise-linear, mixed integer program and a simulated annealing algorithm European Journal of Operational Research 205:81-97 doi:http://dx.doi.org/10.1016/j.ejor.2009.11.033

Özcan U, Toklu B (2008) A tabu search algorithm for two-sided assembly line balancing The International Journal of Advanced Manufacturing Technology 43:822-829 doi:10.1007/s00170-008-1753-5

Özcan U, Toklu B (2009) Balancing of mixed-model two-sided assembly lines Computers & Industrial Engineering 57:217-227 doi:http://dx.doi.org/10.1016/j.cie.2008.11.012

Patel A, Al-Janabi S, AlShourbaji I, Pedersen J (2015) A novel methodology towards a trusted environment in mashup web applications computers & security 49:107-122

Polat O, Kalayci CB, Mutlu Ö, Gupta SM (2016) A two-phase variable neighbourhood search algorithm for assembly line worker assignment and balancing problem type-II: an industrial case study International Journal of Production Research 54:722-741 doi:10.1080/00207543.2015.1055344

Purnomo HD, Wee H-M, Rau H (2013) Two-sided assembly lines balancing with assignment restrictions Mathematical and Computer Modelling 57:189-199

Ramezanian R, Ezzatpanah A (2015) Modeling and solving multi-objective mixed-model assembly line balancing and worker assignment problem Computers & Industrial Engineering 87:74-80 doi:http://dx.doi.org/10.1016/j.cie.2015.04.017

Ritt M, Costa AM, Miralles C (2016) The assembly line worker assignment and balancing problem with stochastic worker availability International Journal of Production Research 54:907-922 doi:10.1080/00207543.2015.1108534

Roshani A, Giglio D (2017) Simulated annealing algorithms for the multi-manned assembly line balancing problem: minimising cycle time International Journal of Production Research 55:2731-2751

Scholl A, Becker C (2006) State-of-the-art exact and heuristic solution procedures for simple assembly line balancing European Journal of Operational Research 168:666-693 doi:http://dx.doi.org/10.1016/j.ejor.2004.07.022

Sungur B, Yavuz Y (2015) Assembly line balancing with hierarchical worker assignment Journal of Manufacturing Systems 37:290-298 doi:http://dx.doi.org/10.1016/j.jmsy.2014.08.004

Tang Q, Li Z, Zhang L (2016) An effective discrete artificial bee colony algorithm with idle time reduction techniques for two-sided assembly line balancing problem of type-II Computers & Industrial Engineering 97:146-156 doi:http://dx.doi.org/10.1016/j.cie.2016.05.004

Tang Q, Li Z, Zhang L, Zhang C (2017) Balancing stochastic two-sided assembly line with multiple constraints using hybrid teaching-learning-based optimization algorithm Computers & Operations Research 82:102-113

Tang QH, Li ZX, Zhang LP, Floudas CA, Cao XJ (2015) Effective hybrid teaching-learning-based optimization algorithm for balancing two-sided assembly lines with multiple constraints Chin J Mech Eng-En 28:1067-1079 doi:10.3901/Cjme.2015.0630.084

Tuncel G, Aydin D (2014) Two-sided assembly line balancing using teaching–learning based optimization algorithm Computers & Industrial Engineering 74:291-299

Vilà M, Pereira J (2014) A branch-and-bound algorithm for assembly line worker assignment and balancing problems Computers & Operations Research 44:105-114 doi:http://dx.doi.org/10.1016/j.cor.2013.10.016

Yuan B, Zhang C, Shao X (2015) A late acceptance hill-climbing algorithm for balancing two-sided assembly lines with multiple constraints Journal of Intelligent Manufacturing 26:159-168 doi:10.1007/s10845-013-0770-x

Zacharia PT, Nearchou AC (2016) A population-based algorithm for the bi-objective assembly line worker assignment and balancing problem Engineering Applications of Artificial Intelligence 49:1-9 doi:http://dx.doi.org/10.1016/j.engappai.2015.11.007

Zhang B, Pan Q-k, Gao L, Zhang X-l, Sang H-y, Li J-q (2017) An effective modified migrating birds optimization for hybrid flowshop scheduling problem with lot streaming Applied Soft Computing 52:14-27 doi:http://dx.doi.org/10.1016/j.asoc.2016.12.021

Zhong Y-g, Ai B (2017) A modified ant colony optimization algorithm for multi-objective assembly line balancing Soft Computing 21:6881-6894