Journal of Mathematical Extension
Vol. 19, No. 5 (2025) (1) 1-42
ISSN: 1735-8299
URL: http://doi.org/10.30495/JME.2025.3391
Original Research Paper

Hybrid and Hyper Meta-Heuristics Algorithms for Cost-Based U-Shaped Assembly Line Balancing Problem with Multi-Mode Equipment Possibility

S. Zhagharian
Shiraz University of Technology

H. R. Maleki[*]
Shiraz University of Technology

S. Niroomand
Firouzabad Higher Education Center, Shiraz University of Technology

Abstract. A U-shaped assembly line (UAL) is an assembly line (AL) effective type that offers many advantages over straight AL. It can be installed in a smaller physical space with fewer number of stations. This study investigates a bi-objective model that incorporates station installation costs, variable operating costs, and equipment purchasing costs into a typical cost-based UAL balancing problem. For the first time, the possibility of selecting equipment types is considered in such AL, where task operating duration is determined by the sort of equipment needed for that task. As the problem has a high degree of complexity, we employ several classical meta-heuristics and hyper-heuristics, enhance them with new logic, and hybridize them to form our solution approach. The Taguchi approach is used to tune the parameters of each algorithm. According to the obtained outcomes, the GD, VNS, and their hybrid version obtain better results than other algorithms.

Received: June 2025; Accepted: September 2025
[*]Corresponding Author

AMS Subject Classification: 90B30; 90C27; 68T20
Keywords and Phrases: Hybrid meta-heuristic algorithm; assembly line balancing; U-shaped assembly line; NP-hard; equipment selection.

1 Introduction

Assembly lines (ALs) are continuous systems with flow line production, extensively utilized in the industrial manufacturing of large quantities of standardized products. In recent years, these systems have become important in the high valence production of the products that are customized. Because of the high level of automation, the (re)-configuration assembly systems are related to significant financial costs and critical challenges. Therefore, it is essential for manufacturers to develop effective strategies and utilize advanced technologies to optimize their AL systems. Many researches have focused on creating optimization models, such as assembly line balancing problems (ALBP), for backing real-world configuration schematization and reducing investment costs. An ALBP in general consists of all sequencing of duties and decisions that are associated with the allocation of resources and aligning the productive units for a specific production procedure before the real assembly can start in order to optimize some criteria. This includes setting the number of stations, the capacities of system, i.e. cycle time, station equipment, and also allocation of work content to production units, i.e. determining assignments and considering limitations of precedence of work contents. The cycle time of a line means the time among sequential permissions of the outputs at the end of the line or the maximum time dedicated to each station in the AL. A precedence diagram is formed of grids representing the tasks of the production procedure and a set of arcs representing the precedence constraints between the tasks. Bryton initially introduced the ALBP [7], and the first scientific investigation was conducted by Salveson [34]. Subsequently, numerous studies have been published on this subject. ALBPs are categorized into various types according to the features of ALs, for instance, the essence of task times (probabilistic and deterministic), the quantity of models (single and multi-model), and the form of AL (U- and straight-type) [39]. According to the classification by Baybars, ALBPs can be grouped into two broad categories, namely, basic and comprehensive ALBPs (SALBP-Simple assembly line balancing; GALBP-Generalized assembly line balancing, respectively) [4]. The GALBP considers all subjects and have more characteristics than the SALBPs, including mixed-model ALBPs, parallel ALBPs, two-sided ALBPs, U-shaped assembly line balancing problems (UALBPs), etc. [13]. It is important to mention that in terms of the objective function (OF), the ALBPs can be categorized into four well-known types. Type-I focuses on reducing the number of stations for a given cycle time, while Type-II aims to minimize the cycle time for a specified number of stations. Type-E seeks to maximize line efficiency when both cycle time and the number of stations are variable. Type-F finds a practical solution when both the cycle time and the number of stations are predetermined [31]. Adaptability is a crucial factor in contemporary manufacturing systems to meet unique demands efficiently and economically [20]. U-shaped lines with increasing number of crossover stations not only provide flexibility but also by permitting the interaction among tasks and stations, enhancing line efficiency, balancing workloads among stations (operators), optimizing space usage, improving communication among human operators [27].

There are different approaches to solve mathematical models of ALBPs. They are categorized into two major classes: exact methods and approximate methods. The branch and bound algorithm is among the most widely used exact solution techniques, which can be utilized with an optimization software. When the dimensions of the problem increase, a lot of time is required for calculations. Since, ALBPs belong to the category of NP-hard optimization problems (OPs) [17], it is essential to employ meta-heuristics or approximate solution methods. Ponnambalam et al conducted a study on type-I in a SALBP [30]. Hamta et al. conducted effective research to minimize the cycle time [14]. Salehi et al. conducted a study on SALBP, incorporating three minimization-type OFs that account for the total cost of the AL, including equipment and labor costs, within a fuzzy environment. A hybrid fuzzy interactive approach was proposed to address this problem [32]. Azizoğlu and Imat developed a branch and bound approach utilizing a combined integer linear programming model to smooth the workload in SALBPs, given a predetermined number of stations and cycle time [3]. Çil et al. developed a mixed-model ALBP that integrates human labor and robots to minimize model cycle time, which can be optimally addressed using a mixed-integer linear programming model for small-sized problems. In addition, the bee and artificial bee colony algorithms are introduced for handling larger instances [10]. Salehi et al. formalized a new cost-oriented ALBP. They designed several meta-heuristic algorithms for it [33]. Niroomand for the first time formulated a straight ALBP with the possibility of availability of equipment with various types for tasks. To solve it efficiently, the artificial electric field and simulated annealing algorithms are considered [26]. Özbakır and Seçme considered the stochastic parallel ALBP with a cost-based OF include the cost of incompletion, equipment, and station construction and to solve it apply a hyper-heuristic algorithm based on simulated annealing [28]. The UALBP is a classic problem that was first introduced by Miltenburg and Wijngaard [22]. After that, various research have been focused on handling large size UALBPs. For instance see [37, 15, 19, 24, 5]. Li et al. have recently utilized an innovative beam search heuristic algorithm to unravel both types-I and II of UALBPs [21]. Zhang et al. devised a human operator assignment strategy that dynamically balances cycle time and ergonomic risk within a unified approach to UALBPs. To optimize simultaneously both objectives, a restarted iterated pareto greedy algorithm was employed [43]. Pınarbaşı introduced new mathematical programming and constraint programming models for type-II of UALBPs with assignment restrictions, line proficiency and CPU time as other implementation measurements [29]. Khorram et al. optimized simultaneously equipment cost, stations number, and level of duty performing quality with assignment duty and worker to station in a UALBP. To consider it, a multi-objective nonlinear formulation was presented and also used it linearized version [18]. Jiao et al. introduced a multi-objective model for balancing a two-sided UALBP, aiming to minimize a smoothing index, and the number of stations. The enhanced ant colony optimization algorithm is applied to address this balancing problem [16].

So far, in the UALBPs, the fixed-wage of the human operators, the station construction, and purchasing of the equipment have been considered as costs. Since in the real world, human operators have abilities with different qualitative levels, and each task has a certain difficulty, and several tasks may be assigned at each station, therefore the variable human operator' wage could be based on the most difficult and expensive task at the station. Furthermore, each model of equipment such as manual, automated, etc. has a direct effect on the operating time of the task and a direct effect on the cost of purchasing equipment. Mainly, more expensive models have higher operating speed to perform a task, which shortens the time to perform the task and vice versa.

In this work, for the first time, we consider some new characteristics, addressing the UALBP. Some contributions of the UALBP of this study are given below.

- Minimizing the variable costs of human operators' wages based on their abilities to carry out the task in a UALBP.
- Minimizing the cost of using different models of equipment in each station in a UALBP.
- Considering the desired assumptions in formulating the UALBP.
- Solving the problem model efficiently using several meta-heuristic methods and proposing their hybrid versions.

The simultaneous investigation of these items in the basic model of UALBP creates a novel and complex model.

The paper's framework is as follows: In Section 2, the UALBP of this study is described, the assumptions are explained in more detail, and its mathematical formulation is constructed. In Section 3, various traditional and hybrid meta-heuristic methods are suggested to address the established mathematical model. Section 4 conducts a comprehensive computational analysis on several test problems by the proposed algorithms. The paper's conclusion is provided in Section 5.

2 Explanation and Formulation of The Problem

Installing an AL is a long-term decision and requires a lot of investment. Thus, it is crucial to meticulously design and balance such a system to ensure its optimal efficiency in the future. The flexible nature of the U-shaped lines in the structure causes the integration of the material flow in the industrial unit, facilitates the relationship between the worker and the supervisor and increases the efficiency of the space.

In any business, production costs play a significant role. A key component of making sure the company's goods are profitable and cost-effective is knowing how to estimate, record, and manage manufacturing costs. The cost of production arises from raw material, production equipment, worker compensation and commissions, office supplies, resources, supervisor or support staff salaries, renting the company's place, and maintenance expenditures. In the following, we discuss the details of the assumptions of the problem examined in this research.

This study's ALBP investigates how tasks are allocated to stations, taking into account the following factors:

- The AL is U-shaped.
- Every task is allocated to a single station.
- Only one human operator is allowed to perform tasks at each station.
- Cycle time of the line is known.
- Tasks should be assigned to stations according to the precedence diagram.
- There are various models of required tools for completing each task due to the technological advancements but only one type of each required tool is assigned to a station for the tasks of that station.
- If two tasks in a station require a common tool, only one unit of that tool is allocated to that station.
- The processing time of each task depends on the type of the tool placed in that station and is a distinctive value.
- The maximum time dedicated to each station should not exceed the cycle time.
- The cost of establishing each station is fixed, determined and the same for all stations.
- Variable human operator' wage of each station is based on the most difficult and expensive task at the station.

Table 1: Indices, parameters, and decision variables utilized in this paper.

Indicator	Type	Explanation
$i(j)(I)$	Parameter	Indicator utilized for task (indicator utilized for task) (number of tasks)
$k(K)$	Parameter	Indicator utilized for station (number of stations)
$l(L)$	Parameter	Indicator utilized for equipment (equipment number)
$m(M)$	Parameter	Indicator utilized for accessible equipment types (accessible equipment number types)
t_{il_m}	Parameter	Processing time for task i using equipment l with model m
tl_i	Parameter	Set of necessary equipment for carry out task i
p_i	Parameter	precedence relationships task i
ec_{l_m}	Parameter	Purchasing cost of model m of equipment l
ct	Parameter	Cycle time
d	Parameter	Demand quantity
stb	Parameter	Fixed cost of station construction
sal	Parameter	Fixed wage for operator hired at each station
c_i	Parameter	Cost rate of task i
C_k	Positive variable	Cost rate of station k based on its most costly task
X_{ik}, Y_{jk}	Binary variable	1, in the case of task i is allocated to station k, 0, else
Z_{l_mk}	Binary variable	1, for the case that model m of equipment l is allocated to station k 0, else
W_k	Binary variable	1, whenever station k is opened 0, else

- The problem should minimize total stations' construction cost, total human operators' wage, and total equipment procurement cost.

Based on the explained assumptions, the outline of the mathematical model for the proposed UALBP with the possibility of choosing the equipment model according to the notations described in Table 1 is as follows:

$$OF_1 = min(stb + sal)\sum_{k=1}^{K} W_k + ct \times d\sum_{k=1}^{K} C_k \tag{1}$$

$$OF_2 = min\sum_{l=1}^{L}\sum_{m=1}^{M}\sum_{k=1}^{K} ec_{l_m} Z_{l_mk} \tag{2}$$

subject to

$$\sum_{k=1}^{K} X_{ik} + Y_{ik} = 1 \qquad \forall i \tag{3}$$

$$\sum_{k=1}^{K} (K-k+1)(X_{ik} - X_{hk}) \geq 0 \qquad \forall (i,h) \in p_i \tag{4}$$

$$\sum_{k=1}^{K} (K-k+1)(Y_{hk} - Y_{ik}) \geq 0 \qquad \forall (i,h) \in p_i \tag{5}$$

$$\sum_{i=1}^{I} \sum_{l \in tl_i} \sum_{m=1}^{M} t_{il_m} Z_{l_m k} (X_{ik} + Y_{ik}) \leq ct \times W_k \qquad \forall k = 1, ..., K \tag{6}$$

$$X_{ik} + Y_{jk} \leq W_k \qquad \forall i, j, k \tag{7}$$

$$\sum_{m=1}^{M} Z_{l_m k} \leq 1 \qquad \forall k, l \tag{8}$$

$$X_{ik} + Y_{ik} \leq \frac{\sum_{l \in tl_i} \sum_{m=1}^{M} Z_{l_m k}}{|tl_i|} \qquad \forall i, k \tag{9}$$

$$c_i (X_{ik} + Y_{ik}) \leq C_k \qquad \forall i, k \tag{10}$$

$$X_{ik}, Y_{jk}, Z_{l_m k}, W_k \in \{0, 1\} \qquad \forall i, j, k, l, m \tag{11}$$

$$C_k \geq 0 \qquad \forall k \tag{12}$$

Constraint (1) reduces both the construction costs of the stations and the total wages of the human operators. Constraint (2) minimizes the equipment procurement cost. Constraint (3) satisfies that each task is given to only one station either in a reverse or forward direction. Constraints set (4) and (5) ensure compliance during the forward and backward assignments, respectively, the precedence relationships (P_i) among the tasks are not violated. Constraint (6) is a non-linear expression which estimates the sum of tasks times assigned to each station must not exceed the cycle time. Constraint (7) defines the value of the variable W_k. Constraint (8) warrants maximum of one type of each equipment can only be allocated to each station. Constraint (9) confirms that when a task is devoted to a station, each required type of equipment must be assigned to that station only once. Constraint (10) determines the human operators' variable wage rate in station k based on the problem's assumptions. Constraint set (11)-(12) describe the type of variables.

In the following, we are going to linearize the above model. For this purpose the model can be linearized by introducing new binary variables V_{il_mk} and Q_{jl_mk} where $V_{il_mk} = Z_{l_mk}X_{ik}$ and $Q_{jl_mk} = Z_{l_mk}Y_{jk}$ with the condition V_{il_mk} and $Q_{jl_mk} \in \{0,1\}$. The product of X_{ik} and Z_{l_mk} can only be non-zero if both of them take value of one, thus $X_{ik} = 0$ or $Z_{l_mk} = 0$ implies that V_{il_mk} must equal zero. Therefore, the following constraints are considered.

$$\begin{aligned} V_{il_mk} &\leq X_{ik}, \\ V_{il_mk} &\leq Z_{l_mk}. \end{aligned}$$

That implies, $2V_{il_mk} \leqslant X_{ik} + Z_{l_mk}$. On the other hand, V_{il_mk} take value of 1 if the product of X_{ik} and Z_{l_mk} is one, which only happens if both of them are one. Therefore, the following constraint is considered.

$$X_{ik} + Z_{l_mk} \leqslant 1 + V_{il_mk}.$$

So the non-linear constraint (6) is converted to the below set of constraints.

$$\begin{aligned} &\sum_{i=1}^{I}\sum_{l\in tl_i}\sum_{m=1}^{M} t_{il_m}(V_{il_mk} + Q_{il_mk}) \leq ct \times W_k && \forall k \\ &2V_{il_mk} \leq Z_{l_mk} + X_{ik} \leq 1 + V_{il_mk} && \forall i,k,m,l \in tl_i \\ &2Q_{il_mk} \leq Z_{l_mk} + Y_{ik} \leq 1 + Q_{il_mk} && \forall i,k,m,l \in tl_i. \end{aligned}$$

2.1 Incorporation of the OFs

Most of the multi-objective optimization approaches are presented as a problem that aims to determine the "best" solution regarding to optimizing the value of a single OF that aggregates all the different goals into a single unit. The formulation presented above is a bi-objective model and both of its OF values are cost based with different scales. Therefore, for integrating them, the OFs may not be algebraically added directly together as the OF with smaller scale not be considered in solving the model [35]. To overcome these limitations, we first convert the value of each OF to the range $[0, 1]$ to OFs be of approximately the same scale. For this purpose, each OF is divided by an upper bound and then are combined to form a single OF. The recommended values for the upper bounds are detailed below:

- For the case, the maximum number of stations (all potential stations) are established, an upper bound value of the first OF is obtained. So,

$$UOF_1 = (stb + sal)K + ct \times d \sum_{i=1}^{I} c_i$$

- By considering all stations as active and using all available equipment of their most expensive type at each station, an upper bound related to the second OF is achieved. So

$$UOF_2 = K \sum_{l=1}^{L} max_{m \in \{1,\ldots,M\}} ec_{l_m}$$

Therefore, the single-objective and linearized form of bi-objective the model of the proposed UALBP is written as below,

$$OF = min \left(\frac{(stb + sal) \sum\limits_{k=1}^{K} W_k + ct \times d \sum\limits_{k=1}^{K} C_k}{UOF_1} + \frac{\sum\limits_{l=1}^{L} \sum\limits_{m=1}^{M} \sum\limits_{k=1}^{K} ec_{l_m} Z_{l_m k}}{UOF_2} \right) \tag{13}$$

subject to

$$\sum_{k=1}^{K} X_{ik} + Y_{ik} = 1 \qquad \forall i \tag{14}$$

$$\sum_{k=1}^{K} (K-k+1)(X_{ik} - X_{hk}) \geq 0 \qquad \forall (i,h) \in p_i \tag{15}$$

$$\sum_{k=1}^{K} (K-k+1)(Y_{hk} - Y_{ik}) \geq 0 \qquad \forall (i,h) \in p_i \tag{16}$$

$$\sum_{i=1}^{I} \sum_{l \in tl_i} \sum_{m=1}^{M} t_{il_m}(V_{il_mk} + Q_{il_mk}) \leq ct \times W_k \qquad \forall k \tag{17}$$

$$2V_{il_mk} \leq Z_{l_mk} + X_{ik} \leq 1 + V_{il_mk} \qquad \forall i,k,m,l \in tl_i \tag{18}$$

$$2Q_{il_mk} \leq Z_{l_mk} + Y_{ik} \leq 1 + Q_{il_mk} \qquad \forall i,k,m,l \in tl_i \tag{19}$$

$$X_{ik} + Y_{jk} \leq W_k \qquad \forall i,j,k \tag{20}$$

$$\sum_{m=1}^{M} Z_{l_mk} \leq 1 \qquad \forall k,l \tag{21}$$

$$X_{ik} + Y_{ik} \leq \frac{\sum_{l \in tl_i} \sum_{m=1}^{M} Z_{l_mk}}{|tl_i|} \qquad \forall i,k \tag{22}$$

$$c_i(X_{ik} + Y_{ik}) \leq C_k \qquad \forall i,k \tag{23}$$

$$X_{ik}, Y_{jk}, Z_{l_mk}, W_k \in \{0,1\} \qquad \forall i,j,k,l,m \tag{24}$$

$$C_k \geq 0 \qquad \forall k \tag{25}$$

The model (13)-(25) is feasible (has a solution) only if there is at least one combination of task assignment and equipment selection that simultaneously satisfies the following three main conditions:

- Precedence relationships must be respected. Constraints (15)-(16) of this model ensure that all tasks are assigned without violating the precedence relationships.

- For each task i, all equipment l in the set tl_i must be available at the station to which the task is assigned. This requirement is included in constraint (22). This constraint guarantees that if a task is assigned to a station ($X_{ik} + Y_{ik} = 1$), at least one model (m) of each required tool ($l \in tl_i$) must be assigned to that station.

- The cycle time must not be violated. Constraint (17) must hold for each station k used ($W_k = 1$). This means that the sum of the times of all tasks assigned to a station, given the equipment model chosen for that station, must not exceed the cycle time.

If cycle time is so small that even by choosing the fastest model for each equipment, the sum of the times of at least one task (or a combination of tasks whose prerequisites allow them to be in the same station) cannot be included, the model becomes infeasible.

Infeasibility management solutions are presented based on the domain standards and implications:

- Increase cycle time: This is the simplest and most common solution. If the model is infeasible because cycle time is too small, the decision maker (production manager) should reconsider increasing the cycle time and decreasing the production output rate. This provides the necessary space to include tasks in the stations.

- Invest in faster equipment: If increasing the cycle time is not a viable option, another solution is to invest in faster and more expensive equipment models (i.e., choosing a m model that has a lower t_{il_m}). This will reduce the operation time and may allow the previous cycle time to be respected. This trade-off between investment cost and production rate is exactly what the model is trying to optimize.

- Revision of product/process design: In extreme cases, it may be necessary to revise the task prerequisite diagram (assembly process) to enable better balancing. This is usually outside the scope of the line balancing problem and within the scope of product design.

The single-objective model of the UALBP of this section presented by the formulation (13)-(25) is adjusted and computed for some instances via the General Algebraic Modeling Software (GAMS). This is done to study the complexity of the proposed formulation. The obtained results are shown by Table 2. According to the obtained optimality gap values, we observe that by increasing the problem size, in the given time of 6 hours the optimality gap is increased. For example for the problem with size of 45 tasks there is 66.97% optimality gap after 6 hours. The optimality gap significantly widens as the problem size increases. Therefore, use meta-heuristic or approximate solution methods to solve large-sized instances of this problem may be necessary. In the continuation, numerous meta-heuristic algorithms are established to solve the formulation (13)-(25).

Table 2: Results of implementation model (13)-(25) using GAMS.

				Output	
SP	Tasks number	Equipment number	Equipment models number	CPU time	Optimality gap
1	8	2	2	00:00:03	0%
2	8	3	2	00:00:04	0%
3	21	5	4	6:00:04	3.3%
4	21	6	4	6:00:04	0.02%
5	30	6	5	6:00:03	43.27%
6	30	7	5	6:05:34	42.16%
7	45	7	6	6:00:03	68.45%
8	45	8	6	6:00:03	66.97%

3 Established Approaches

Here several meta-heuristic strategies such as genetic algorithm (GA), great deluge (GD), variable neighbor search (VNS), simulated annealing (SA), and hyper-heuristic algorithms are proposed to solve the formulation (13)-(25) of Section 2. As some basic requirements, we need to define solution representation, its evaluation method, and some neighborhood generation mechanisms. In this section, the proposed encoding-decoding strategy, the suggested neighborhood search operators, and the suggested meta-heuristic solution approaches are presented respectively to be used in the established algorithms.

3.1 Encoding decoding scheme

To create and represent a solution, first a vector is considered with the length of n tasks which consists of a random permutation of integer value n (called vector A). So, the initial solution is created as a random permutation vector. Second, it is supposed that the number of potential stations matches the number of tasks. Because each equipment may be available in different models, a matrix of L rows and K columns is considered (called matrix B), and each value of this matrix represents a randomly selected model of the equipment of its row in the station of its column. Therefore, at the beginning, randomly one model of each equipment is considered for each station. Now, the tasks are allocated to the stations according to the random order of vector A according to the below steps. In these steps a solution is generated, and its OF value is calculated.

Step 1: In the order of tasks given by vector A, the first task with no predecessor or no successor is selected and assigned to the first station (the first eligible station). The station time according to the required equipment of this task and the model of equipment of the station (based on matrix B) is calculated after this assignment. The assigned task is deleted from vector A and also from the predecessor and successor sets of other remaining tasks.

Step 2: Step 1 is repeated for other tasks till vector A becomes empty. The eligible stations for assigning a selected task of matrix A begins at the last station that includes its predecessors and/or successors and has enough time for the task. Based on the equipment model of the stations and the required equipment of the considered task, the task is allocated to the first eligible station that can accept it according to the cycle time. If no such eligible station exists for this task, the considered task is allocated to the first empty station.

Step 3: The empty stations are removed. In each of the remained stations, according to the assigned tasks, some pre-assigned equipment may not be needed. These equipment also are removed from the stations. Then the solution OF is evaluated according to the integrated OF provided in Section 2.

3.2 Neighborhood search operators

To examine the solution space and search for a solution achieving an improved OF value, during the procedure of meta-heuristic solution approaches, a local search operator should be performed on each initial solution. For this purpose, the neighborhood search operators like swap, insertion, reversion, mutation, and single-point crossover are used here. These operators are explained below.

- **Swap**: Two tasks from vector A in the solution representation are selected randomly and exchanged.
- **Insertion**: A pair of tasks from vector A is randomly selected and one of them is inserted to the right of other one.
- **Reversion**: A pair of tasks from vector A is randomly selected and the order of tasks between them is reversed.
- **Mutation**: This operator performs the permutation operation with the probability of p_m (mutation probability) on the sequence of vector A.
- **Single-point crossover**: The process crosses two parents P_1 and P_2 to generate two offspring O_1 and O_2. Two parents are selected to pair the chromosomes of one parent with the chromosomes of the other parent. These two parents are cut at a random cut point (say c). In P_1, the genes to the left of c are copied directly into O_1 at the same position, and the same happens for P_2 and O_2. The genes to the right of the cut point into O_1 are then filled with any elements from P_2 that were not already copied from P_1. The listed elements are copied into O_1, preserving the order they had in P_2. The same procedure is applied to produce O_2.

3.3 Meta-heuristic solution approaches

Meta-heuristic algorithms are probabilistic approximation techniques that permit to solve OPs effectively. These algorithms usually compute sub-optimal and good-quality within a reasonable implementation time-frame for a wide class of OPs, even for problems that are very difficult to be solved by classical exact approaches [25]. In this section, eight meta-heuristic algorithms including classical algorithms and their hybridizations are introduced. In the following, we will discuss their details.

3.3.1 Simulated annealing (SA) algorithm

The simulated algorithm is one of the most effective meta-heuristic methods based on a single solution for solving OPs in large scales. In general, the SA is an iterative method according to variable temperature parameter which that imitate an annealing procedure [41]. In every step, the SA considers a neighbor state from the system and decides possibly between moving to a new state or staying in the previous state. Eventually, these probabilities lead the system to a suitable state [1]. For solving an OP, the SA algorithm begins with an initial solution and then moves to neighbor solutions in an iterative loop. If the new state is better than the current state, the algorithm confirms it as the current state and moves towards it, otherwise algorithm confirms it as the current solution only with the probability of $\exp(\frac{-\Delta E}{T})$. ΔE is the difference of the OF value of the current solution and the new solution and T is the temperature parameter. At each temperature several repetitions are performed then the temperature is slowly decreased to a final temperature as a stopping criterion.

3.3.2 Genetic algorithm (GA)

For tackling a complex OP, population-based meta-heuristic solution approaches gather much attention as several solutions can be considered and evaluated simultaneously [2]. The genetic algorithm (GA) is one of the most famous algorithms which works with a population of solutions. The initial principles of the GA are taken from the science of genetic and inspired from Darwin's evolution theory [38]. To apply this approach for a problem each chromosome corresponds to a solution to the OP and is formed of several genes. Each gene simulates the variables of the OP. In the GA algorithm, a population of candidates or random solutions is considered. Each solution is evaluated with a fitness value calculated from its OF value. Based on the roulette wheel technique and stochastically selection, the parents are chosen from the current population considering the suitability of the solutions for generation. Then the crossover and mutation operators considering the probability of crossover (p_c), the probability of mutation (p_m), and other probable operators are employed to the parents in order to improve the chromosomes and form a novel population. Finally, the current population is reformed and the process is continued for a number of iterations to reach a termination criterion. The solution with the highest fitness value in the final population is reported as the best solution obtained by the GA.

3.3.3 Variable neighbor search (VNS) algorithm

The VNS algorithm was designed for the first time by Brimberg [6]. This algorithm systematically searches the solution space by changing the neighbors around a solution to escape from a local optimum [23]. The VNS algorithm begins from an initial solution (x_0) with even a poor approximation and then moves towards a better solution until the stopping condition (for example, the maximum number of iterations or the maximum CPU time) is fulfilled. Then selects a neighbor through the operator $N_1()$ and if the achieved solution is better than the previous solution, the local search operator seeks for a more suitable solution in the neighborhood of the current solution, which is created based on the determined method until its local optimum is obtained. The mentioned steps are repeated until we reach the neighboring structures with the highest order $N_{max}()$. $N_1()$,..., $N_3()$ are respectively swap, reversion and insertion operators, which are the neighboring operators of the slution. In this algorithm changing of the neighborhood search operator creates more diversity in the local search.

3.3.4 Great deluge (GD) algorithm

The GD algorithm is a general algorithm based on a single solution developed by Dueck [12]. It is similar to the SA algorithm from many technical points of view so that the GD algorithm accepts improved solutions and even worse solutions under certain conditions. First, an initial solution (x_0) (current solution) is generated and from this solution, a neighboring solution (x_1) is generated. The OF value of the initial solution is considered as the threshold value. One of the following two conditions will happen:

- If the value of OF at the neighboring solution is better than the value of OF at the current solution, the novel solution considered as the current solution.
- If the OF value at the neighboring solution is not better than the OF value at the current solution, first, the threshold value is reduced by ϵ which is obtained by below formula. Then if the OF value at the neighboring solution is lower than the threshold value, the new solution is considered as the current solution.

  $$\epsilon = (f(x_1) - f(best_{sol}))/NIteraion$$

  where $NIteration$ is the number of iterations.

This algorithm is repeated till reaching the stopping criterion which can be a number of iteration or a limited running time.

3.3.5 Hyper SA algorithm

A hyper-heuristic algorithm is a general problem-solving technique which is designed to be autonomous of the domain of the problem. That is described as “A hyper-heuristic is an automated scheme to choice or produce heuristics for solving huge computational search problems” [9]. The hyper-heuristic algorithms proposed in this study are of the heuristic selection type. Their primary mechanism is to intelligently select and sequence from a predefined set of low-level heuristics (LLHs) during the search process to construct a solution for the complex UALBP. The low-level heuristics are simple, domain-specific rules used to decide the order in which tasks are assigned to stations. Ten different LLHs were employed in our framework, each based on a distinct priority rule. They are listed below with their abbreviations and descriptions:

- SPT (Shortest Processing Time): Prioritizes tasks with the shortest processing time.
- LPT (Longest Processing Time): Prioritizes tasks with the longest processing time.

- MiTNST (Minimum Total Number of Successor Tasks): Prioritizes tasks with the fewest total successor tasks.
- MaTNST (Maximum Total Number of Successor Tasks): Prioritizes tasks with the most total successor tasks.
- MiCT (Minimum Cost of Tools): Prioritizes tasks requiring tools with the lowest cost.
- MaCT (Maximum Cost of Tools): Prioritizes tasks requiring tools with the highest cost.
- MaTNPT (Maximum Total Number of Predecessor Tasks): Prioritizes tasks with the most total predecessor tasks.
- MiTNPT (Minimum Total Number of Predecessor Tasks): Prioritizes tasks with the fewest total predecessor tasks.
- MaTNET (Maximum Total Number of Equipment Tasks): Prioritizes tasks that require the most number of different tools.
- MiTNET (Minimum Total Number of Equipment Tasks): Prioritizes tasks that require the fewest number of different tools.

The core mechanism for selecting LLHs is an Adaptive Operator Selection (AOS) strategy, which uses a learning process to identify the best sequence of heuristics. This process is managed through an Adaptive Heuristic Selection (AHS) matrix (ratings table) with rows corresponding to LLHs and columns to task positions [8]. A sequence of LLHs (an LLH vector) is generated for a solution, where each element dictates the rule for assigning a task. The selection of i-th element of the LLH vector to pick out an assignable task for each assignable position is obtained based on the roulette wheel procedure on column i of the AHS matrix. The performance of an LLH vector is evaluated by the quality of the solution it generates. The acceptance of a new solution (generated by a new LLH vector) is governed by the underlying metaheuristic framework (Simulated Annealing for HyperSA), which includes an aspiration criterion. If the new solution has a better OF value, it is immediately considered as the current solution. For HyperSA, a worse solution can be accepted with a probability based on the current temperature ($exp(-\Delta E/T)$). Furthermore, the AHS matrix ratings are updated based on acceptance: Reward ($\alpha\%$ increase): If the new solution is accepted and is an improvement. Small Reward ($\beta\%$ increase, where $\beta << \alpha$): If the new solution is accepted but is not an improvement (uphill move). Penalty ($\sigma\%$ decrease): If the new solution is rejected. The information (including iteration, temperature and the number of unimproved iteration parameters) are updated, and the main loop procedure is repeated to identify a solution to the problem such that the termination criterion is met.

3.3.6 Hyper GD algorithm

Design and implementation of the hyper GD algorithm is similar to the hyper SA algorithm but with some changes. The first layer works according to the principles of the GD algorithm. The initial level for OF value based on the initial LLH vector of heuristics is determined. A new LLH neighbor vector is created as described above, and produces a neighbor solution which is compared to the solution at hand. For HyperGD, a worse solution is accepted if its objective value is below a dynamically decreasing level (threshold). If the OF value for the solution is better or if it is lower than the level, the new LLH vector is immediately utilized as the current LLH vector, and all parameters and the AHS matrix rating of the LLH vector are updated positively. Otherwise, the AHS matrix rating is updated negatively and the new LLH vector is produced and returned to the main loop.

3.3.7 GA-GD algorithm

Any search algorithm must explore and exploit the space of search. Exploration means the process of visiting entirely novel areas of a search space, while exploitation means the procedure of visiting those areas of the search space in the neighborhood of previously visited points. To be successful, a search algorithm must strike a good balance between exploration and exploitation. One way to achieve this case is to combine meta-heuristic algorithms with local search [11].

This hybrid algorithm links the exploration ability of the genetic algorithm with the impressive local search of the GD algorithm. Hybridization of these algorithms may ameliorate their efficiency. This is related to the way offsprings are produced. The GA algorithm focuses on crossover operator and the GD algorithm focuses on mutation operator. The pseudo-code of this proposed hybrid GA-GD algorithm is illustrated by Algorithm 1.

3.3.8 VNS-GD algorithm

As another hybrid algorithm, we combine the VNS and GD algorithms. This combination of the algorithms may provide better results and avoid local optima. The combination of the VNS with the GD starts from the first neighborhood search operator in the VNS algorithm, and the GD algorithm is utilized to compare neighboring solutions. This process is repeated for all considered search operators. Eventually, the best-found result is revealed. The pseudo-code of this hybrid VNS-GD algorithm is shown by Algorithm 2.

4 Computational Study

Some computational experiments are performed in the sequel to evaluate the proposed algorithms performance given in Section 3 for solving the problem of Section 2. To do this, some SPs of the literature are modified and utilized here. The Taguchi design scheme is applied to tune the parameters of the established algorithms. According to the best values extracted for the parameters, the algorithms are executed for all SPs and the final outcomes are found. The continuation of this section represents the details of this numerical study. This is notable to mention that, all algorithms are coded in MATLAB and are run on a PC with Intel(R) Core(TM) i7-7700 CPU 3.60 GHz processor and 32.0 GB RAM.

4.1 Test problems

Several SPs are examined to evaluate the effectiveness of the algorithms. In this study, 9 test problems, each of them with two sets of equipment are used. The number of tools needed to carry out the production process in the mentioned sets of equipment are approximately 15% and 20%

Algorithm 1 The hybrid GA-GD algorithm proposed in this study.

```
Require: $It_{max}$ (number of iterations), It (iterations for mutate), $n_{pop}$ (population size), $p_c$ (crossover rate), $p_m$ (mutation rate), $r_m$ (mutation probability)
 1: Initialization: $pop$ (an initial random population), $Cost(pop)$ (evaluate fitness), $level$ (initial level)
 2: Sort $pop$ in ascending order based on OF value
 3: while termination criterion is not met do
 4:     for $i = 1$ to $round(\frac{p_c * n_{pop}}{2})$ do
 5:         Choose two chromosomes $x_1$ and $x_2$ from $pop$
 6:         Generate two new chromosomes by single-point crossover, save to $pop_c$
 7:         $pop = pop + popc$
 8:     end for
 9:     if $rand \leq r_m$ then
10:         for $j = 1$ to $round(p_m * n_{pop})$ do
11:             Randomly select a chromosome $x_i$ from $pop$
12:             Mutate $x_i$ with probability $r_m$ to generate $x_i^{'}$
13:             if $Cost(x_i^{'}) \leq Cost(x_i)$ or $Cost(x_i^{'}) \leq level$ then
14:                 Save $x_i^{'}$ to $pop_m$
15:             else
16:                 for $k = 1$ to $It$ do
17:                     Generate $x_i^{''}$ as neighbor of $x_i^{'}$
18:                     if $Cost(x_i^{''}) \leq Cost(x_i^{'})$ or $Cost(x_i^{''}) \leq level$ then
19:                         Save $x_i^{''}$ to $pop_m$
20:                     end if
21:                     $\delta = \frac{Cost(x_i^{'}) - Cost(best_{sol})}{It_{max}}$
22:                     $level = level - \delta$
23:                 end for
24:             end if
25:         end for
26:         $pop = pop + pop_m$
27:     end if
28:     Select best $n_{pop}$ chromosomes from the merged population
29:     Sort population in ascending order
30:     $best_{sol}$ is the best chromosome in population
31: end while
32: Output: the best chromosome
```

Algorithm 2 The hybrid VNS-GD algorithm proposed in this study.

```
Require: It (the number of iterations for each neighborhood operator), $It_{max}$ (maximum iterations), $k_{max}$ (number of neighborhood operators), $NE_k$ (set of neighborhood operators)
 1: Initialization: $level$ (the initial threshold), $I_{pos}$ (an initial random solution), $Cost(I_{pos})$ (compute fitness value), $best_{sol} \leftarrow I_{pos}$
 2: while termination criterion is not met do
 3:     for $k = 1$ to $k_{max}$ do
 4:         $I'_{pos} \leftarrow NE_k(I_{pos})$                ▷ Apply neighborhood operator $k$
 5:         for $j = 1$ to $It$ do
 6:             $I''_{pos} \leftarrow NE_k(I'_{pos})$              ▷ Generate new neighbor
 7:             if $Cost(I''_{pos}) \leq Cost(I_{pos})$ or $Cost(I''_{pos}) \leq level$ then
 8:                 $I_{pos} \leftarrow I''_{pos}$
 9:                 if $Cost(I_{pos}) \leq Cost(best_{sol})$ then
10:                     $best_{sol} \leftarrow I_{pos}$
11:                 end if
12:             end if
13:             $\delta \leftarrow \frac{Cost(I'_{pos}) - Cost(best_{sol})}{It_{max}}$
14:             $level \leftarrow level - \delta$
15:         end for
16:     end for
17: end while
18: Output: $best_{sol}$
```

of the number of tasks respectively. The precedence diagrams of the SPs are taken from Scholl [36]. The task times, and other required costs are generated randomly. The details of the SPs are presented in Table 3.

Table 3: Some features of the SPs used for the computational experiment.

SP	Tasks number	Min. of the task times	Max. of the task times	Equipment number	Models number in each equipment
1	8	10	40	2	2
2	8	10	40	3	2
3	21	10	40	4	2
4	21	10	40	5	2
5	30	10	40	5	2
6	30	10	40	6	2
7	45	10	40	7	3
8	45	10	40	9	3
9	70	10	40	11	3
10	70	10	40	14	3
11	89	10	40	13	3
12	89	10	40	18	3
13	111	10	40	17	4
14	111	10	40	23	4
15	148	10	40	23	4
16	148	10	40	29	4
17	160	10	40	24	4
18	160	10	40	32	4

4.2 Parameter setting

The Taguchi experimental design method was created via Taguchi [40]. It is an effective tool to design the high-quality systems. This approach produces an efficient, simple and systematic scheme to optimize the designs for cost, performance and quality [42]. A perfect investigation of an algorithm with 6 variables at 3 levels needs 729 experiments to find the best level of each variable which is actually time consuming. Using the Taguchi method, the number of required experiments is decreased significantly and depending to the selected orthogonal array it may be even decreased from 729 to 27 experiments in the given example. In this method, the factors of an algorithm are divided into two types of uncontrollable (noise) and controllable factors. Therefore, this method tries to find the best value of the controllable factors in such a way to minimize the effect of the noise factors. The key parameters for each algorithm were selected based on their established importance in the meta-heuristics literature and their direct influence on the algorithm's behavior. The proposed values (levels) for each parameter were chosen based on initial testing on a subset of small-to-medium test problems to identify a reasonable range where each parameter showed an effect and values commonly used and recommended in selected based on empirical knowledge and previous studies for similar optimization problems. To apply the Taguchi method on a meta-heuristic algorithm, the below steps are followed.

Step 1: The potential levels (values) of each parameter of the algorithm are determined.

Step 2: A suitable orthogonal array for determining the set of required experiments for tuning the algorithm is selected. A suitable orthogonal array should have some experiments equal to or greater than the total degree of freedom value which is obtained by summation of the levels of all parameters minus the number of parameters plus 1. In each experiment the level of each parameter is fixed.

Step 3: The algorithm is executed for each experiment for a number of times and in all experiments, the marginal mean of OF values is recorded as the performance measure and shown by C. The signal-to-noise ratio (S/N) for each level of the parameters is obtained by the below formulation.

$$\frac{S}{N} = -10 \log C^2$$

Step 4: The goal of problem is to minimize the OF. A higher S/N ratio indicates a more robust parameter setting that minimizes variance and leads to better (lower) OF values. The level with the highest average S/N ratio for each parameter is selected as the best level.

In this work, for the algorithms GD, VNS, SA, hyperSA, hyperGD, and VNS-GD, the $L9$ orthogonal array and for the algorithms GA and GA-GD, the $L27$ orthogonal array is considered. The level of parameters are determined based on empirical knowledge and previous studies. The test problem involving 70 tasks as a representative medium-sized instance is considered, and each experiment is conducted 5 times and the mean of the OF values from these 5 runs was recorded as the performance measure for that experiment. After calculations of the Taguchi

Table 4: The proposed values and the best level elicited for the parameters used in the established meta-heuristic algorithms.

Algorithm	Parameter	Proposed values	Number of necessary trials by simple combination	Taguchi experimental design		
				Degree of freedom	Required orthogonal array	Best value obtained for the parameters
GD	It_{max}	50,100,200	9	5	$L9$	200
	It	100,200,300				300
VNS	It_{max}	50,100,200	9	5	$L9$	200
	It	100,200,300				300
SA	It	100,200,300	81	9	$L9$	200
	T_0	50,200,500				500
	T_f	0.001,0.01,1				0.001
	α	0.9,0.95,0.99				0.95
VNS-GD	It_{max}	50,100,200	9	5	$L9$	200
	It	100,200,300				200
HyperGD	It_{max}	50,100,200	9	5	$L9$	100
	It	100,200,300				200
HyperSA	It	100,200,300	81	9	$L9$	300
	T_0	50,200,500				50
	T_f	0.001,0.01,1				0.01
	α	0.9,0.95,0.99				0.99
GA	It	100,200,300	243	11	$L27$	200
	n_{pop}	100,200,300				300
	p_c	0.6,0.75,0.9				0.6
	p_m	0.1,0.25,0.4				0.25
	μ	0.3,0.55,0.8				0.3
GA-GD	It_{max}	100,200,300	243	11	$L27$	300
	It	100,150,200				200
	n_{pop}	100,200,300				200
	p_c	0.6,0.75,0.9				0.6
	p_m	0.1,0.25,0.4				0.25
	μ	0.3,0.55,0.8				0.8

method on the elicited OF values, the suggested values of each parameter and the best obtained levels for them are determined and provided in Table 4. The result of the Taguchi method for the SA algorithm of this study is shown in Figure 1.

5 Findings and Conclusions

5.1 Computational Experiments Results

To conduct the last tests on the sample problems (SPs) and obtain the final results, the following issues are considered.

- The parameters of each algorithm are considered to the best value reported by Table 4.
- To make an accurate comparison among the algorithms, in all the meta-heuristic algorithms, a common run time for each SP is taken into account. For this aim, the run time of the VNS algorithm (as

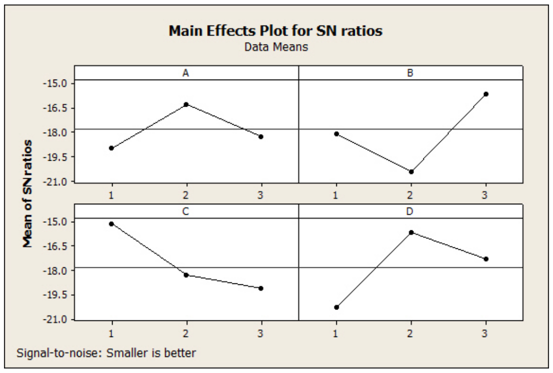

Figure 1: S/N ratio for the obtained numbers of the SA algorithm.

it has the longest run time among the algorithms) for each test problem is considered as the common run time of all algorithms. Therefore, execution of other algorithms is repeated to reach the common run time.

- To achieve more reliable results, each algorithm is implemented 10 times for each SP and the results are compared according to the minimum, maximum, and mean of the extracted OF values of the 10 runs.
- For more comparisons, the CPLEX solver in GAMS software is used for each test problem with at most 6 hours allowed run time. The obtained optimality gaps and the values of the OF are used for comparison purpose.

As mentioned, the final results are expressed in terms of maximum, minimum, and the mean of the found OF values for each SP in each algorithm. These outcomes are summarized in Tables 5-7.

As it is clear from the derived minimum OF values of the utilized algorithms for each SP in Table 5, the best minimum value in 55.56% of the test problems are obtained by the GD algorithm. In all cases, except for the small size of the first SP to the fourth SP, the hybrid VNS-GD algorithm is superior to classical VNS algorithm. Also, the minimum values extracted via the VNS algorithm are superior to the minimum values extracted via the GA and SA algorithms. The results of the minimum values of the hybrid GA-GD algorithm are not better than the ones derived by the classical GA algorithm. The ranking of the performances of the established algorithms for the minimum obtained values (among the 10 runs) are GD, VNS-GD, VNS, GA, GA-GD, SA, HyperSA, HyperGD.

Table 6 shows the highest OF values achieved from 10 runs of each designed algorithm for each SP.

Regarding the results of this table, in 61% of the cases, the GD algorithm performance is better than other algorithms. The hybrid VNS-GD algorithm works better than the VNS algorithm in more than half of the test problems.

In the comparison among the VNS, SA, and GA algorithms, the VNS algorithm works better than the SA and GA, and the GA and SA algorithms obtain the same number of minimum values.

The mean of the identified OF values by each of the designed approaches for each SP is represented by Table 7. According to these results, in 66.66% of the test problems, the GD algorithm provides better results. Subsequently, the combination of the GD and VNS algorithms ameliorates the classic VNS algorithm performance, but the combination of the GD and GA algorithms cannot ameliorate the GA algorithm performance. In comparison of the classical algorithms, the VNS algorithm operates better than the GA algorithm, and the GA algorithm operates better than the SA algorithm. The ranking of the algorithms created by the mean OF values from the 10 runs is shown as GD, VNS-GD, VNS, GA, GA-GD, SA, HyperSA, HyperGA.

According to the results of Table 5-7 which are also depicted schematically by Figure 2-10, the following conclusion can be drawn:

- The GD algorithm, as a local search algorithm, by accepting inferior solutions in each iteration, demonstrates a higher exploration ability and can work effectively for the problem of this study.
- According to the previous point, it can be concluded that by influencing the exploration performance of the GD algorithm, the combination of GD and VNS algorithms shows a better performance than the traditional VNS algorithm for searching the solution space.

Table 5: Minimum of the values extracted via the OF with 10 runs for any test problem.

		Minimum of the OF values							
SP	Number of tasks	GA	VNS	SA	GD	GA-GD	VNS-GD	HyperSA	HyperGD
1	8	0.7766	0.7766	0.7766	0.7766	0.7766	0.7766	1.0702	1.0702
2	8	0.805	0.805	0.805	0.805	0.805	0.805	1.0569	1.0569
3	21	0.5572	0.5572	0.5572	0.5572	0.5572	0.5572	0.6632	0.6632
4	21	0.4659	0.4659	0.4659	0.4659	0.4659	0.4659	0.5317	0.5317
5	30	0.4137	0.4035	0.4144	**0.3976**	0.4162	**0.3976**	0.5179	0.5179
6	30	0.4485	0.4406	0.4407	**0.4327**	0.4503	0.4381	0.5518	0.5518
7	45	0.2954	0.2811	0.2935	**0.2761**	0.3058	0.29	0.3737	0.3737
8	45	0.271	0.2648	0.271	**0.2645**	0.2751	0.2673	0.3003	0.3003
9	70	0.2047	0.1969	0.2069	**0.191**	0.2059	0.1978	0.2254	0.2254
10	70	0.1892	0.1882	0.1923	0.1826	0.1891	**0.1817**	0.2264	0.2264
11	89	0.1854	0.1814	0.1866	0.1779	0.1854	**0.1776**	0.2084	0.2084
12	89	0.1846	**0.1806**	0.1871	0.181	0.1848	0.1811	0.2102	0.2102
13	111	0.16	0.1557	0.1614	**0.151**	0.1616	0.1536	0.1756	0.1756
14	111	0.1654	0.1647	0.1685	0.1611	0.1686	**0.1591**	0.1831	0.1831
15	148	0.1438	0.1376	0.1476	**0.1356**	0.1409	0.1393	0.1573	0.1573
16	148	0.138	0.1334	0.1417	**0.1309**	0.1385	0.1338	0.1471	0.1471
17	160	0.1423	0.1413	0.1461	**0.1393**	0.1448	0.142	0.1587	0.1587
18	160	0.1352	0.132	0.14	**0.1302**	0.1378	0.1321	0.1414	0.1414

- Our computational results clearly showed that the GD algorithm consistently outperformed all other algorithms, including GA and SA, across most test problems.
- According to the obtained results, with increasing the number of problem samples, the performances of the meta-heuristic algorithms approaches closer to the results of the classical algorithms. As a suggestion, by changing the type of low-level heuristic vectors or by increasing the number of samples, perhaps better results can be obtained further.

5.2 Managerial insights

In today's rapid and competitive market, the use of ALBP is an unavoidable issue for the managers of production companies. Based on the problem, formulation, and proposed algorithms of this study, the below insights can be considered by the managers.

- The U-shaped ALBP proposed in this study could be used to balance the UALs for minimizing the establishment costs and the equipment purchasing costs.
- In the cases that different types of an operational equipment with

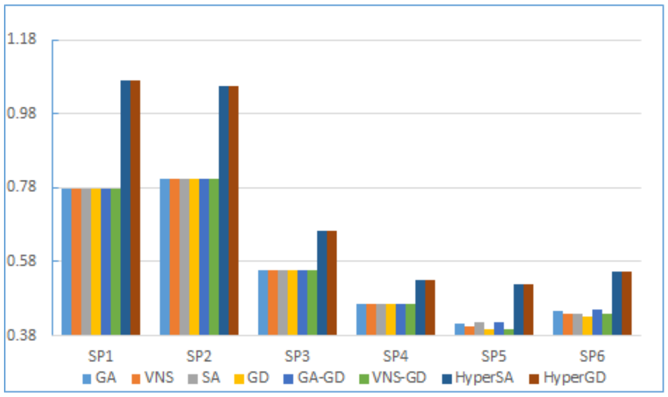

Figure 2: Diagram of the minimum values elicited via the established algorithms for SPs (1-6).

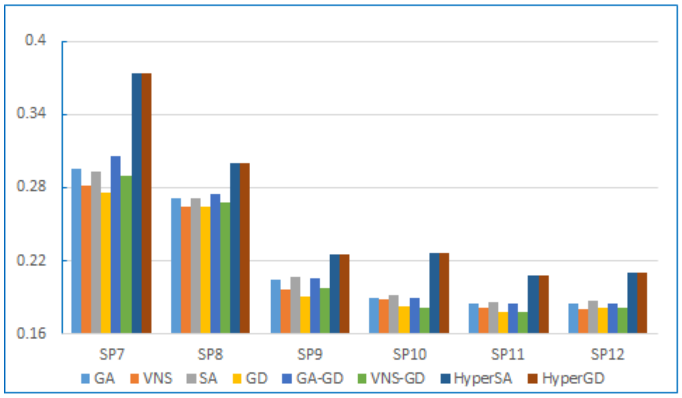

Figure 3: Diagram of the minimum values elicited via the established algorithms for SPs (7-12).

different prices and operational times, the proposed formulation can be applied by the managers.

- According to the experiments of this study, the existing exact solution approaches may not result in a good balance for the ALs following the U-shaped ALBP proposed in this study. Therefore, the exact solution approaches are not suggested for the managers

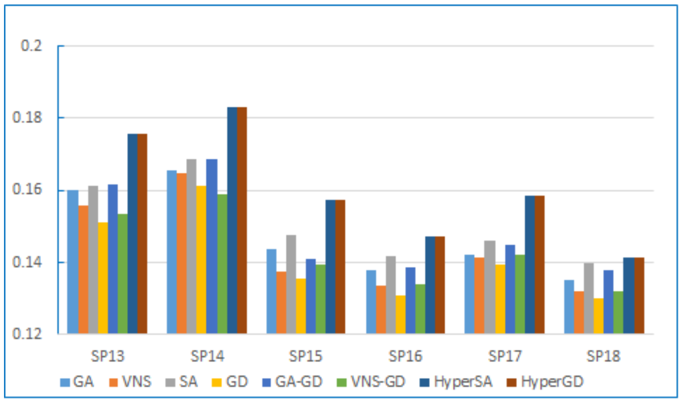

Figure 4: Diagram of the minimum values elicited via the established algorithms for SPs (13-18)

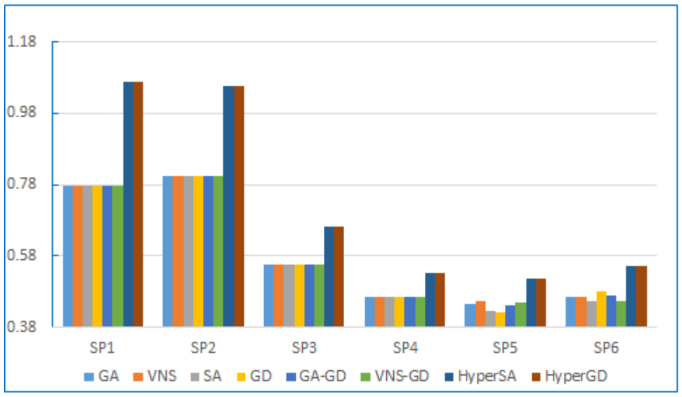

Figure 5: Diagram of the maximum values elicited via the established algorithms for SPs (1-6).

of a production company for this aim.

- The proposed meta-heuristic algorithm of this study can be used by the managers to obtain a good balance for their related AL formulated by the proposed U-shaped ALBP of this study.
- As a result of this study, the proposed hybrid meta-heuristic algorithms can provide better balance according to the proposed U-

Table 6: Maximum of the values extracted via the OF with 10 runs for any test problem.

		Minimum of the OF values							
SP	Number of tasks	GA	VNS	SA	GD	GA-GD	VNS-GD	HyperSA	HyperGD
1	8	0.7766	0.7766	0.7766	0.7766	0.7766	0.7766	1.0702	1.0702
2	8	0.805	0.805	0.805	0.805	0.805	0.805	1.0569	1.0569
3	21	0.5572	0.5572	0.5572	0.5818	0.5572	0.5572	0.6632	0.6632
4	21	0.4659	0.4659	0.4659	0.4659	0.4718	0.4659	0.5317	0.5317
5	30	0.4463	0.4547	0.4244	**0.4239**	0.4421	0.4494	0.5179	0.5179
6	30	0.4669	0.4658	0.4546	0.4799	0.4693	**0.454**	0.5518	0.5518
7	45	0.3177	0.3115	0.3116	**0.3065**	0.3185	0.3135	0.3737	0.3737
8	45	0.2819	0.2748	0.2781	**0.2731**	0.2834	0.2753	0.3003	0.3003
9	70	0.211	0.2048	0.2098	**0.2004**	0.2111	0.2071	0.2254	0.2254
10	70	0.2	0.2034	0.2016	**0.1988**	0.2021	0.2009	0.2264	0.2264
11	89	**0.1894**	0.1953	0.192	0.1961	0.1899	0.1928	0.2084	0.2084
12	89	0.1888	0.1868	0.1891	0.1867	0.188	**0.1859**	0.2102	0.2102
13	111	0.1662	0.1595	0.1661	**0.1585**	0.1663	0.1598	0.1756	0.1756
14	111	0.1733	0.1693	0.175	**0.1672**	0.1745	0.1674	0.1831	0.1831
15	148	0.1512	0.1487	0.1509	**0.1459**	0.1498	0.147	0.1573	0.1573
16	148	0.1433	0.1374	0.1437	**0.1349**	0.1428	0.137	0.1471	0.1471
17	160	0.1491	0.1462	0.1512	**0.1427**	0.1491	0.1452	0.1587	0.1587
18	160	0.1398	0.1354	0.1413	**0.1335**	0.1405	0.135	0.1414	0.1414

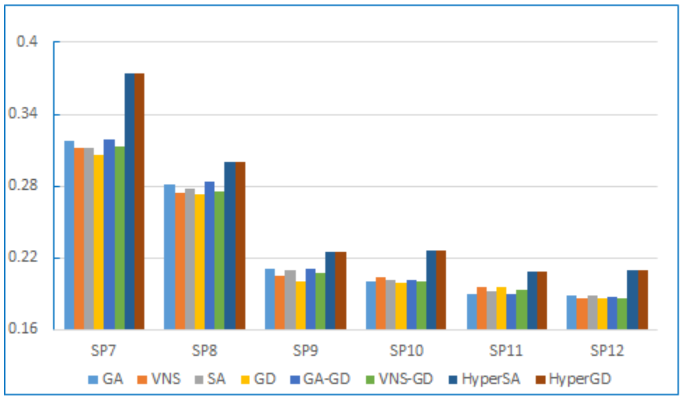

Figure 6: Diagram of the maximum values elicited via the established algorithms for SPs (7-12).

shaped ALBP for the operational managers of a production company.

The core managerial dilemma captured by our bi-objective model is the trade-off between capital expenditure on equipment and operational expenditure. Our model's solutions reveal how different strategies for navigating this trade-off impact overall performance.

1. **The High-Investment, High-Efficiency Strategy:** Choosing

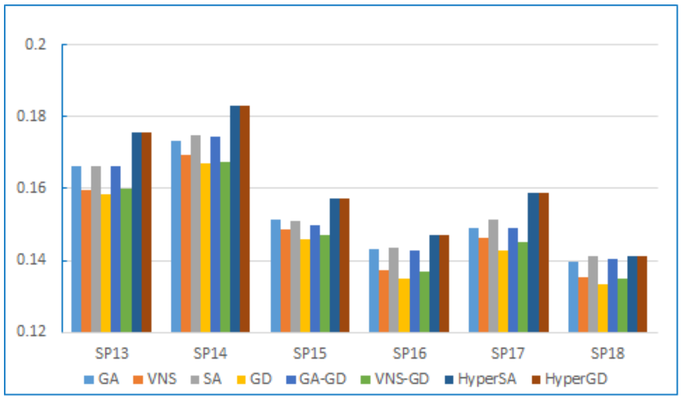

Figure 7: Diagram of the maximum values elicited via the established algorithms for SPs (13-18).

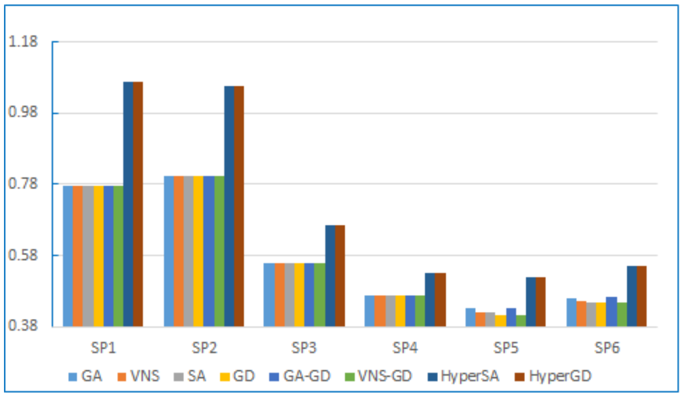

Figure 8: Diagram of the average values elicited via the established algorithms for SPs (1-6).

more expensive, automated equipment models leads to shorter task processing times. This allows:

- Fewer Stations: More tasks can be grouped into a single station without exceeding the cycle time, reducing the fixed cost of station installation.
- Lower Variable Wage Costs: The most difficult task in a sta-

Table 7: Average of the values extracted via the OF with 10 runs for any test problem.

		Minimum of the OF values							
SP	Number of tasks	GA	VNS	SA	GD	GA-GD	VNS-GD	HyperSA	HyperGD
1	8	0.7766	0.7766	0.7766	0.7766	0.7766	0.7766	1.0702	1.0702
2	8	0.805	0.805	0.805	0.805	0.805	0.805	1.0569	1.0569
3	21	0.5572	0.5572	0.5572	0.5596	0.5572	0.5572	0.6632	0.6632
4	21	0.4659	0.4659	0.4659	0.4659	0.4664	0.4659	0.5317	0.5317
5	30	0.43305	0.4208	0.41969	**0.4103**	0.43022	0.41316	0.5179	0.5179
6	30	0.45834	0.45087	0.44659	**0.44589**	0.46168	0.44651	0.5518	0.5518
7	45	0.30712	0.30014	0.30216	**0.29593**	0.31347	0.30412	0.3737	0.3737
8	45	0.27663	0.27043	0.27548	**0.26876**	0.27849	0.27065	0.3003	0.3003
9	70	0.20781	0.20144	0.20854	**0.19722**	0.20858	0.2025	0.2254	0.2254
10	70	0.19451	0.19659	0.19579	**0.18876**	0.1948	0.19364	0.2264	0.2264
11	89	0.18739	0.18747	0.18937	0.18663	0.18773	**0.18522**	0.2084	0.2084
12	89	0.18654	0.18416	0.18785	0.18427	0.18639	**0.18324**	0.2102	0.2102
13	111	0.16294	0.15745	0.16416	**0.15448**	0.16409	0.15664	0.1756	0.1756
14	111	0.1708	0.16636	0.17263	**0.16464**	0.17155	0.16483	0.1831	0.1831
15	148	0.14685	0.14515	0.14941	**0.14114**	0.14637	0.14237	0.1573	0.1573
16	148	0.14047	0.13549	0.14271	**0.13305**	0.14069	0.13528	0.1471	0.1471
17	160	0.14661	0.14367	0.1497	**0.14047**	0.14764	0.14364	0.1587	0.1587
18	160	0.13873	0.13419	0.14038	**0.13243**	0.13904	0.13387	0.1414	0.1414

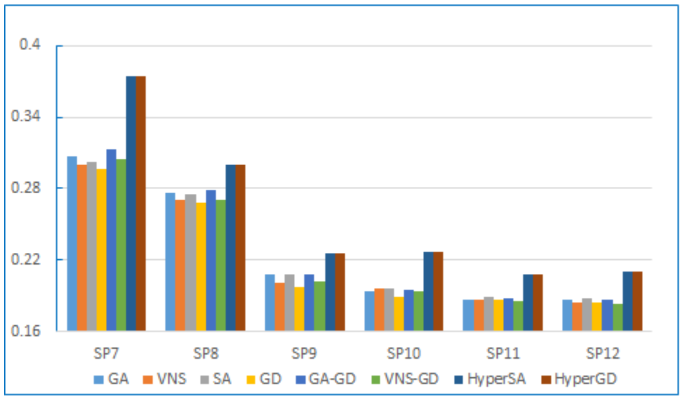

Figure 9: Diagram of the average values elicited via the established algorithms for SPs (7-12).

tion (which sets the wage rate) might be completed faster with better equipment, potentially lowering the variable wage cost. However, the primary operational expenditure saving comes from needing fewer stations and thus fewer operators.

This strategy prioritizes a lower capital expenditure on equipment over time by making a higher initial capital expenditure on equipment. It is suitable for companies with high production volumes

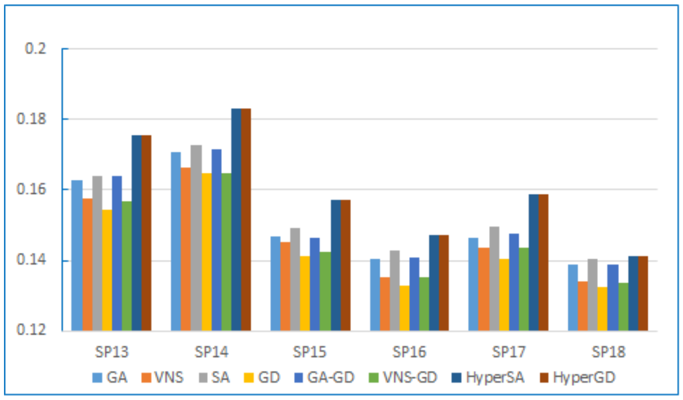

Figure 10: Diagram of the average values elicited via the established algorithms for SPs (13-18).

where the long-term savings in labor and improved throughput justify the larger upfront investment.

2. **The Low-Investment, Low-Efficiency Strategy:** Choosing cheaper, often manual equipment models reduces the immediate capital outlay. However, this leads to longer task times, which often necessitates:

   - More Stations: To meet the same cycle time with longer tasks, the line must be broken down into more stations, increasing station installation costs and the number of operators required.

   This strategy minimizes initial capital expenditure on equipment but leads to a higher operational expenditure due to higher labor and station costs. It might be a viable option for smaller manufacturers with low production volumes or capital constraints, where minimizing upfront cost is the primary concern.

3. **The Hybrid Strategy and the Pareto Frontier:** The optimal solutions found by our algorithms typically represent a hybrid strategy. They smartly mix equipment types across the line: investing in faster, more expensive equipment only for bottleneck

tasks (tasks with long inherent times or many predecessors/successors) whose acceleration allows for significantly better station consolidation. For less critical tasks, cheaper, slower equipment is used. This nuanced approach minimizes the total cost rather than minimizing one at the expense of the other.

6 Conclusion

In this research, a typical cost-based UALBP was focused. U-shaped AL has many advantages than straight AL. It needs a smaller physical space and may be installed with fewer number of stations comparing to straight ALs. In the UALBP of this study, station installation cost, variable operational cost, and equipment purchasing cost were considered simultaneously. For the first time equipment type selection possibility was considered in such line. According to this possibility, selecting a type of a required equipment for a task, results in a specific operational time for the task. The problem was modeled as a bi-objective model where the first objective minimizes installation and operational costs, and the second one minimizes the equipment purchasing costs for all stations. As the problem has high degree of complexity, as another contribution, some classical meta-heuristics were used and were hybridized as solution approach. In addition, some logic was added to their solution generation phase and some hyper-heuristic algorithms were introduced. Several problems from the literature were considered, and as the problem is new, some parameters values were generated for them. A Taguchi approach was used to tune the parameters of each algorithm. The final experiments were conducted, and based on the results achieved, the GD, VNS, and their hybrid version performed better than other algorithms. As future research directions, the equipment type selection possibility can be considered for other types of ALBPs. On the other hand, use of matheuristics (combination of mathematical programming and meta-heuristics) may be useful for obtaining more qualitative solutions.

Acknowledgements
The authors would like to express their sincere gratitude to the anonymous reviewers for their constructive comments and valuable sugges-

tions, which greatly helped to improve the quality of this manuscript.

References

[1] F. Alkhateeb, B. H. Abed-alguni, and M. H. Al-rousan, Discrete hybrid cuckoo search and simulated annealing algorithm for solving the job shop scheduling problem, *The Journal of Supercomputing*, 78 (2022), 4799–4826.

[2] I. Asiltürk and H. Akkuş, Determining the effect of cutting parameters on surface roughness in hard turning using the Taguchi method, *Measurement*, 44 (2011), 1697–1704.

[3] M. Azizoğlu and S. Imat, Workload smoothing in simple assembly line balancing, *Computers and Operations Research*, 89 (2018), 51–57.

[4] I. Baybars, A survey of exact algorithms for the simple assembly line balancing problem, *Management Science*, 32 (1986), 909–932.

[5] A. Baykasoglu and M. E. Senol, Integrating the Bees Algorithm with WSAR for Search Direction Determination and Application to Constrained Design Optimisation Problems, *Intelligent Engineering Optimisation with the Bees Algorithm* (2024), 141–155.

[6] J. Brimberg, A variable neighborhood algorithm for solving the continuous location-allocation problem, *Studies in Locational Analysis*, 10 (1996), 1–12.

[7] B. Bryton, *Balancing of a Continuous Production Line*, PhD thesis, Northwestern University (1954).

[8] E. K. Burke, M. Gendreau, M. Hyde, G. Kendall, G. Ochoa, E. Özcan, and R. Qu, Hyper-heuristics: A survey of the state of the art, *Journal of the Operational Research Society*, 64 (2013), 1695–1724.

[9] E. K. Burke, M. Hyde, G. Kendall, G. Ochoa, E. Özcan, and J. R. Woodward, A classification of hyper-heuristic approaches, *Handbook of Metaheuristics* (2010), 449–468.

[10] Z. A. Çil, Z. Li, S. Mete, and E. Özceylan, Mathematical model and bee algorithms for mixed-model assembly line balancing problem with physical human-robot collaboration, *Applied Soft Computing*, 93 (2020), 106394.

[11] M. Črepinšek, S. H. Liu, and M. Mernik, Exploration and exploitation in evolutionary algorithms: A survey, *ACM Computing Surveys (CSUR)*, 45 (2013), 1–33.

[12] G. Dueck, New optimization heuristics: The great deluge algorithm and the record-to-record travel, *Journal of Computational Physics*, 104 (1993), 86–92.

[13] M. Eghtesadifard, M. Khalifeh, and M. Khorram, A systematic review of research themes and hot topics in assembly line balancing through the web of science within 1990–2017, *Computers and Industrial Engineering*, 139 (2020), 106182.

[14] N. Hamta, S. F. Ghomi, F. Jolai, and M. A. Shirazi, A hybrid PSO algorithm for a multi-objective assembly line balancing problem with flexible operation times, sequence-dependent setup times and learning effect, *International Journal of Production Economics*, 141 (2013), 99–111.

[15] R. K. Hwang, H. Katayama, and M. Gen, U-shaped assembly line balancing problem with genetic algorithm, *International Journal of Production Research*, 46 (2008), 4637–4649.

[16] Y. Jiao, X. Su, L. Li, and Z. Wu, An improved ant colony optimization algorithm for two-sided U-type assembly line balancing problems, *Engineering Optimization* (2024), 1–18.

[17] R. M. Karp, On the computational complexity of combinatorial problems, *Networks*, 5 (1975), 45–68.

[18] M. Khorram, M. Eghtesadifard, and S. Niroomand, Hybrid metaheuristic algorithms for U-shaped assembly line balancing problem with equipment and worker allocations, *Soft Computing*, 26 (2022), 2241–2258.

[19] I. Kucukkoc and D. Z. Zhang, Balancing of parallel U-shaped assembly lines, *Computers and Operations Research*, 64 (2015), 233–244.

[20] Y. Li, N. He, and D. Z. Zhang, Simulation based layout design and optimization for assembly line system, *Lecture Notes in Engineering and Computer Science: Proceedings of the World Congress on Engineering* (2019), 315–320.

[21] Z. Li, M. N. Janardhanan, and H. F. Rahman, Enhanced beam search heuristic for U-shaped assembly line balancing problems, *Engineering Optimization*, 53 (2021), 594–608.

[22] G. J. Miltenburg and J. Wijngaard, The U-line line balancing problem, *Management Science*, 40 (1994), 1378–1388.

[23] N. Mladenović, M. Dražić, V. Kovačevic-Vujčić, and M. Čangalović, General variable neighborhood search for the continuous optimization, *European Journal of Operational Research*, 191 (2008), 753–770.

[24] J. Mukund Nilakantan and S. Ponnambalam, Robotic U-shaped assembly line balancing using particle swarm optimization, *Engineering Optimization*, 48 (2016), 231–252.

[25] S. Nesmachnow, An overview of metaheuristics: accurate and efficient methods for optimisation, *International Journal of Metaheuristics*, 3 (2014), 320–347.

[26] S. Niroomand, Hybrid artificial electric field algorithm for assembly line balancing problem with equipment model selection possibility, *Knowledge-Based Systems*, 219 (2021), 106905.

[27] M. K. Oksuz, K. Buyukozkan, and S. I. Satoglu, U-shaped assembly line worker assignment and balancing problem: A mathematical model and two meta-heuristics, *Computers and Industrial Engineering*, 112 (2017), 246–263.

[28] L. Özbakır and G. Seçme, A hyper-heuristic approach for stochastic parallel assembly line balancing problems with equipment costs, *Operational Research* (2022), 1–38.

[29] M. Pınarbaşı, New mathematical and constraint programming models for U-type assembly line balancing problems with assignment restrictions, *Engineering Optimization*, 54 (2022), 1289–1304.

[30] S. Ponnambalam, P. Aravindan, and G. Mogileeswar Naidu, A multi-objective genetic algorithm for solving assembly line balancing problem, *The International Journal of Advanced Manufacturing Technology*, 16 (2000), 341–352.

[31] M. Rabbani, S. M. Kazemi, and N. Manavizadeh, Mixed model U-line balancing type-1 problem: A new approach, *Journal of Manufacturing Systems*, 31 (2012), 131–138.

[32] M. Salehi, H. R. Maleki, and S. Niroomand, A multi-objective assembly line balancing problem with worker's skill and qualification considerations in fuzzy environment, *Applied Intelligence*, 48 (2018), 2137–2156.

[33] M. Salehi, H. R. Maleki, and S. Niroomand, Solving a new cost-oriented assembly line balancing problem by classical and hybrid meta-heuristic algorithms, *Neural Computing and Applications*, 32 (2020), 8217–8243.

[34] M. E. Salveson, The assembly-line balancing problem, *Transactions of the American Society of Mechanical Engineers*, 77 (1955), 939–947.

[35] D. Savic, Single-objective vs. multiobjective optimisation for integrated decision support (2002).

[36] A. Scholl, *Data of Assembly Line Balancing Problems*, Techn. Hochsch., Inst. für Betriebswirtschaftslehre (1995).

[37] A. Scholl and R. Klein, Ulino: Optimally balancing U-shaped JIT assembly lines, *International Journal of Production Research*, 37 (1999), 721–736.

[38] V. K. Singh, K. C. Panda, A. Sagar, N. Al-Ansari, H.-F. Duan, P. K. Paramaguru, D. K. Vishwakarma, A. Kumar, D. Kumar, P. Kashyap, et al., Novel Genetic Algorithm (GA) based hybrid

machine learning-pedotransfer function (ML-PTF) for prediction of spatial pattern of saturated hydraulic conductivity, *Engineering Applications of Computational Fluid Mechanics*, 16 (2022), 1082–1099.

[39] P. Sivasankaran and P. Shahabudeen, Literature review of assembly line balancing problems, *The International Journal of Advanced Manufacturing Technology*, 73 (2014), 1665–1694.

[40] G. Taguchi, *Introduction to Quality Engineering: Designing Quality Into Products and Processes* (1986).

[41] P. J. Van Laarhoven and E. H. Aarts, *Simulated Annealing*, Springer Netherlands (1987).

[42] W. H. Yang and Y. Tarng, Design optimization of cutting parameters for turning operations based on the Taguchi method, *Journal of Materials Processing Technology*, 84 (1998), 122–129.

[43] Z. Zhang, Q. Tang, R. Ruiz, and L. Zhang, Ergonomic risk and cycle time minimization for the U-shaped worker assignment assembly line balancing problem: A multi-objective approach, *Computers and Operations Research*, 118 (2020), 104905.

Shabnam Zhagharian
PHD student of Applied Mathematics
Department of Mathematics
Shiraz University of Technology
Shiraz, Iran
E-mail: s.zhagharian@sutech.ac.ir

Hamid Reza Maleki
Professor of Applied Mathematics
Department of Mathematics
Shiraz University of Technology
Shiraz, Iran
E-mail: maleki@sutech.ac.ir

Sadegh Niroomand
Associate Professor of Industrial Engineering

Department of Industrial Engineering
Firouzabad Higher Education Center, Shiraz University of Technology
Shiraz, Iran
E-mail: niroomand59@gmail.com