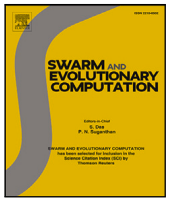
Swarm and Evolutionary Computation 88 (2024) 101579

Contents lists available at ScienceDirect

Swarm and Evolutionary Computation

journal homepage: www.elsevier.com/locate/swevo



A knowledge-guided Estimation of Distribution Algorithm for energy-efficient Joint Robotic Assembly Line Balancing and Feeding Problem

Chu-ge Wu *, Ruochen Zhang, Yuanqing Xia

School of Automation, Beijing Institute of Technology, Beijing, 100081, China

ARTICLE INFO

Keywords:
Robotic assembly line balancing
Part feeding
Estimation of distribution algorithm
Production
Optimization

ABSTRACT

The assembly line serves as a fundamental system in discrete production. To address the challenges in balancing robotic assembly lines, timely part feeding, and the need for sustainable manufacturing, this paper studies an energy-efficient Joint Robotic Assembly Line Balancing and Feeding Problem (JRALB-FP) with the criteria of minimizing both cycle time and total fuel consumption cost. Considering the complexity of the multi-problem and multi-objective optimization, a knowledge-guided Estimation of Distribution Algorithm (KEDA) is proposed to solve energy-efficient JRALB-FP. First, a probability model of EDA for task-workstation allocation paired with a heuristic method-based sampling mechanism is created. Using this probability model, a specific encoding mechanism is designed for part-trailer allocation, and good initial solutions are produced. Second, several properties of the bi-objective problem are analyzed to guide the design of local search operators for both objectives optimization. Third, the updating mechanism of the probability model is designed to learn from the elite solutions. Fourth, two knowledge-guided local search operators are designed and implemented to exploit better non-dominated solutions sufficiently. A design of experiment is carried out to determine the parameters. Extensive computational tests and comparisons with the state-of-the-art multi-objective algorithms are carried out, which verify the effectiveness of the knowledge-guided local search operators, the problem-oriented heuristic-based sampling mechanism, and the special designs of the KEDA in solving the energy-efficient JRALB-FP.

1. Introduction

The assembly line is a typical manufacturing system, as the flow-oriented assembly processes play a crucial role for producers across various discrete manufacturing industries [1], notably the automotive and electronic industries [2]. Recent years have seen an increment in the consumption of electronic products, such as smartphones, which has greatly amplified the demand for the high productivity of the corresponding assembly lines. In addition, in response to the climate change crisis, many countries have initiated low-carbon projects. Energy conservation and energy harvesting [3,4] have gained significant importance across various domains. Especially, it is important to implement energy-efficient manufacturing strategies as approximately one-fifth of greenhouse gas emissions originate from the manufacturing industry [5].

To gain the efficiency of the assembly line, the Assembly Line Balancing Problem (ALBP) is of critical importance [6,7]. ALBP aims to effectively allocate assembly tasks with precedence constraints to workstations along the assembly line, which ensures the uninterrupted operation at each workstation and minimizes the cycle time [8]. The use of industrial assembly robots is widespread in efforts to increase assembly line productivity. Considering this situation, the ALB problem is connected to a robot selection problem as Robotic ALBP [9,10], where each workstation is equipped with a single robot selected from a set of different robot types. A robotic two-sided assembly line balancing problem with setup times is provided in [11]. Furthermore, timely parts feeding is crucial for the assembly process, as optimal positioning of parts containers at workstations can reduce the considerable unproductive walking involved in fetching parts [12]. To reduce the impact of unproductive walking on assembly line efficiency and consume less space, the Assembly Line Feeding Problem (ALFP) addresses the selection of appropriate part storage, feeding, and transportation strategies [13].

Considering the decisive impact of both ALBP and ALFP on the performance of the assembly system in terms of time, costs, and ergonomics [14], and in alignment with the energy efficiency objective of green manufacturing, we propose a combined problem involving RALBP and ALFP called the Joint Robotic Assembly Line Balancing and Feeding Problem (JRALB–FP). Type-II ALBP is addressed, where

* Corresponding author.
E-mail address: wucg@bit.edu.cn (C.-g. Wu).

https://doi.org/10.1016/j.swevo.2024.101579
Received 11 January 2024; Received in revised form 10 March 2024; Accepted 14 April 2024
Available online 2 May 2024

the cycle time is minimized under a fixed number of workstations. In addition, a set of in-house trailers can be enabled to fetch the parts for different assembly tasks from the supermarket. In this way, both the cycle time and fuel consumption associated with part transportation trailers are minimized to ensure efficient and effective production. The assignment of robots to workstations, tasks to workstations, and parts to trailers is considered simultaneously in this paper, to achieve a collaborative optimization solution for the part feeding and assembly line balancing problems.

Given the NP-hard nature of ALBP and ALFP, exact solving methods, construction heuristics, and evolutionary algorithms are widely employed to provide a solution within the given period. In the case of ALBP, an integer programming-based optimization approach [15] is utilized for the dynamic resource reconfiguration problem in automatic assembly lines, which is used to enable reconfigurable production, reduce costs, and promote sustainability in automatic assembly lines. Genetic Algorithms (GA) [16–18], and Simulated Annealing (SA) [19, 20] are extensively adopted to provide solutions for ALBP. A hybrid GA is presented in [16] on the assembly line balancing problem with collaborative robots. RALBP with a focus on sequence-dependent setup times and various assumptions regarding robot assignment is considered in [21], where a polynomial optimal algorithm is incorporated within a meta-heuristic framework to explore the space of giant sequences. Assembly line balancing problem involving human–robot collaboration and a diverse range of operators optimizing the cycle time is investigated in [17] and a customized GA is constructed for the problem. An improved hybrid GA is proposed in [18], incorporating penalty function adjustment and adaptive weighting mechanisms, which provide an optimization scheme for the allocation of multi-position tasks in a robot assembly line, aiming to enhance efficiency and optimize energy consumption. A neighborhood-search enhanced SA is conducted on the assembly line balancing problem with human–robot collaboration in advanced manufacturing systems [19], featuring an adaptive selection mechanism for dynamic neighborhood search.

Furthermore, a list of works explores in-house part feeding and material feeding issues to guarantee the just-in-time part delivery and the assembly process efficiency. A comprehensive survey [22] categorizes related works on the ALFP based on feeding problem design, feeding policies, assembly line, and product characteristics, as well as optimization problem objectives and constraints. Line feeding policies such as line stocking, boxed supply, sequenced parts delivery, stationery kits, and travel kits are defined. The optimization objectives encompass cost considerations, ergonomic factors, transportation costs, shop floor costs, holding costs, and logistical handling costs. Based on ALFP, several works are proposed to study the joint assembly line balancing and feeding problem. The problem is formulated in [23], taking into account both line balancing and material supply to minimize production time while considering container space, supply area constraints, and cycle time constraints. An enhanced multi-Hoffmann heuristic method is embedded with a CPLEX solver. The total operational annual costs of the system are optimized in [14], while the assembly line, supermarket, and parts handling system are synchronized to the cycle time. Its proposed joint approach is compared with the hierarchical approach in terms of the number of assembly stations, number of parts fed with different policies, assembly station volume utilization, and total operational costs, and the results dominate those of the comparison strategies.

In addition to time performance, many other metrics, including energy efficient metrics, are considered. The energy consumption for executing material delivery tasks in the mix-model assembly lines is minimized in [24]. A Taboo enhanced Particle Swarm Optimization algorithm is developed to solve the multi-objective problem. The operational cost and workload imbalance are considered in [25], where the Pareto optimal solutions are generated via the augmented ϵ-constrained method using CPLEX. The population-based evolutionary algorithms are commonly employed to address the multi-objective joint line balancing and part feeding problem due to their inherent parallelism characteristics. Production cycle time and part consumption balance are minimized in [26], which builds a multi-objective mathematical model and utilizes a multi-objective GA to obtain the solutions. Flexsim software was used for simulation. The total energy consumption is considered in [27], as well as the number of workstations and supermarkets. A modified GA with mixed encoding and repair strategy is adopted to solve the joint problem. The total energy consumption of part feeding mobile robots is considered in [28]. A multi-objective disturbance and repair strategy enhanced cohort intelligence algorithm is established to deal with the multi-objective problem.

Considering the NP-hard nature of the considered problem, the evolutionary algorithm is adopted in this paper. Estimation of Distribution Algorithm (EDA) [29] is a data-driven swarm intelligence optimization algorithm, which has been widely adopted to solve complex optimization problems [30], especially scheduling problems [31–33]. An EDA-based hyper-heuristic approach is adopted by [32] to solve the distributed assembly mixed no-idle permutation flowshop scheduling problem. An EDA enhanced by the path relinking mechanism is proposed by [33] to minimize makespan in the directed acyclic graph task scheduling problem, integrating list scheduling heuristics and a probability model for improved performance compared to existing heuristics and evolutionary algorithms. A matrix-cube-based EDA is designed in [34] to solve the distributed assembly permutation flowshop scheduling problem. Branch-and-bound guided EDA is adopted to solve RALBP by [9,10], where the cycle time and total energy consumption are minimized.

Inspired by the effectiveness of EDA in solving complex multi-objective optimization problems, a knowledge-guided EDA (KEDA) is adopted to solve the energy-efficient JRALB–FP. The main contributions of our work can be summarized as follows:

- The fuel consumption cost is reasonably quantified. A Mixed Integer Programming (MIP) model is constructed to formalize the energy-efficient JRALB–FP, considering two objectives: cycle time and trailer fuel consumption, to find the trade-off between assembly efficiency and energy costs.
- Several properties of the energy-efficient JRALB–FP are provided. The correlation between these two objectives is analyzed, and the lower bound of different variables is presented, which guides the design of local search operators.
- To enhance the effectiveness of KEDA, we designed and incorporated a specific sampling mechanism and two knowledge-guided local search operators within the basic EDA. Experimental results validate the effectiveness of KEDA in solving this problem compared to other multi-objective algorithms.

The remaining contents are organized as follows. We define the multi-objective optimization problem and introduce and formulate the JRALB–FP in Section 2. The details of the knowledge-based EDA tailored for the proposed problem are explained in Section 3, where the encoding and decoding mechanism, initialization method, problem properties, and the details of the algorithm are provided. The numerical experimental results are provided in Section 4, and the paper is concluded in Section 5.

2. Energy-efficient joint robotic assembly line balancing and feeding problem

2.1. Basic concepts of multi-objective optimization problem

A multi-objective optimization problem generally involves more than one contradictory objective simultaneously and requires the identification of a desired trade-off between such objectives. The problem can usually be defined by (1).

$$\min y = f(x) = (f_1(x), f_2(x), \ldots, f_q(x)) \tag{1}$$

where $x \in \mathcal{R}^p$ is a p-dimensional decision vector and $y \in \mathcal{R}^q$ is the objective vector that contains q individual objectives. In this paper, the non-dominated sorting method is adopted to compare and rank the solutions for optimization problems considering more than one objective. The essential knowledge of optimal Pareto solutions is provided below.

Pareto dominance: A solution x_1 is considered to dominate solution x_2 (denoted as $x_1 \succ x_2$) if and only if:

$$\begin{aligned} &f_i(x_1) \le f_i(x_2), \quad \forall i \in 1,2,\ldots,q \\ &f_i(x_1) < f_i(x_2), \quad \exists i \in 1,2,\ldots,q \end{aligned} \tag{2}$$

Optimal Pareto solution: A solution x is called an optimal Pareto solution if it is not dominated by any other solution.

Optimal Pareto set/Pareto Front: The solution set containing all optimal Pareto solutions is defined as the optimal Pareto set/Pareto Front. The optimal Pareto set obtained by a certain algorithm is defined as a non-dominated solution set/archive set (AS). Besides, C metric used to estimate multi-objective algorithms is introduced. CM is used to compare the archive sets AS_1 and AS_2 which reflects the dominance relationship between the solutions in these two sets. C metric is computed as (3).

$$C(AS_1, AS_2) = \frac{|x_2 \in AS_2 | \exists x_1 \in AS_1, x_1 \succ x_2 \textbf{ or } x_1 = x_2|}{|AS_2|} \tag{3}$$

where $C(AS_1, AS_2)$ reflects the percentage of the solutions in AS_2 that are dominated by or are the same as the solutions in AS_1. In principle, $C(AS_1, AS_2)$ and $C(AS_2, AS_1)$ are both calculated to compare the Pareto dominance of the archive sets and its algorithm. The larger $C(AS_1, AS_2)$ and smaller $C(AS_2, AS_1)$ means AS_1 is better than AS_2 under the Pareto optimal metric.

Another indicator, **Inversed Generation Distance (IGD)** [35], is also widely used to compare archive sets. IGD estimates the distance from the obtained Pareto front to the true Pareto front. For each solution, the minimum Euclidean distance between the solution and the true Pareto front is calculated. A set of distances are calculated and the average value of them is IGD, which gives a measure of both diversity and convergence.

2.2. JRALB–FP

2.2.1. Problem description

The JRALB–FP is a combination of the RALBP and ALFP. It is proposed to provide an optimal assignment for tasks and their parts to both workstations and in-house trailers, as well as an efficient assignment of manufacturing robots. This paper focus on a single-model assembly line with a one-sided straight layout as Fig. 1 shows. In addition, this paper considers Type-II assembly line balancing problem, aiming to minimize the cycle time given the number of workstations. Additionally, ALFP ensures that various parts are distributed to each workstation according to the assembly line's requirements. To achieve energy efficiency in part feeding, this paper introduces total fuel consumption of the enabled trailers as another objective.

The problem is depicted in Fig. 1, which includes the assembling of products, in-house part transportation, and part feeding. For product assembling, the **assembly task** is the smallest unit within the process, where the assembly process is divided into a set of tasks with precedence constraints. **Workstations** are the equipment used for assembling, which is located alongside the assembly line. The tasks are assigned to these workstations for their assembly in accordance with their precedence constraints. Heterogeneous **manufacturing robots** are allocated to the workstations for task assembling, resulting in varying processing times. **Trailers** are used to transport the required raw parts to specific workstations for task assembling. **Containers** are equipped for the workstations to store the required parts. Each trailer follows a fixed route, which starts from the **supermarket** where the parts are prepared, pauses at its responsible workstations to feed the parts, and then returns to the supermarket. Fig. 1 illustrates that the

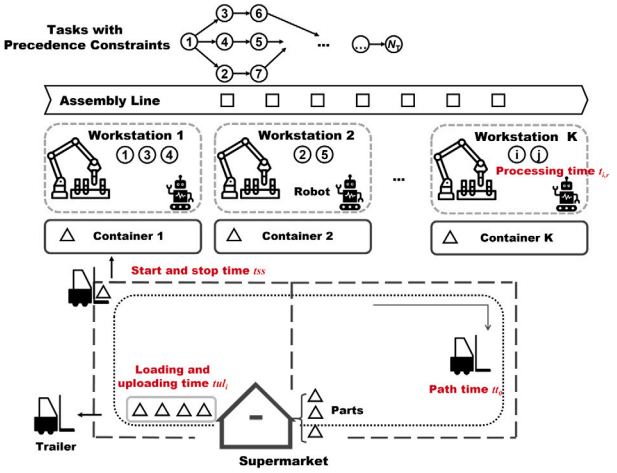


Fig. 1. Illustration of JRALB–FP.

trailer takes a fixed **path time** (tt_0), **loading and unloading time** (tul_i), and **starting and stopping time** (tss) during the each transport. tt_0 represents the time needed to travel across a fixed single route, tss represents the time needed for the trailer to stop and start at each workstation, and tul_i represents the time taken for the loading and unloading of parts for task i. **Cycle time** is defined as the interval time between the assembly of two consecutive finished products, and it is affected by both the assembly time of tasks and the transportation time of parts in this paper. **Fuel consumption cost** is defined as the total fuel consumption cost for part transportation.

2.2.2. Problem formulation

To increase the efficiency of the assembly line and reduce overall energy consumption, it is important to optimize both the cycle time and the total fuel consumption cost. A shorter cycle time allows for the completion of more products within a given period, and a smaller fuel consumption helps to save energy. To formulate the problem, the notations are explained in Table 1, and the MIP model is provided in (4)–(15).

$$\min\{c, fcost\} \tag{4}$$

subject to:

$$\sum_{k=1}^{N_W} x_{i,k} = 1, \quad \forall i \in \boldsymbol{T} \tag{5}$$

$$\sum_{r=1}^{N_R} y_{r,k} = 1, \quad \forall k \in \boldsymbol{W} \tag{6}$$

$$\sum_{g=1}^{N_G} z_{i,g} = 1, \quad \forall i \in \boldsymbol{T} \tag{7}$$

$$u_{g,k} = \max_{1 \le i \le N_T} x_{i,k} \cdot z_{i,g}, \quad \forall g \in \boldsymbol{G}, k \in \boldsymbol{W} \tag{8}$$

$$\sum_{k=1}^{N_W} k \cdot x_{i,k} - \sum_{k=1}^{N_W} k \cdot x_{j,k} \le 0, \quad \forall i \in \boldsymbol{P}_j, j \in \boldsymbol{T} \tag{9}$$

$$\sum_{i=1}^{N_T} m_i \cdot x_{i,k} \le cw, \quad \forall k \in \boldsymbol{W} \tag{10}$$

$$\sum_{i=1}^{N_T} m_i \cdot z_{i,g} \le ct, \quad \forall g \in \boldsymbol{G} \tag{11}$$

$$z_{i,g} \le zz_g, \quad \forall g \in \boldsymbol{G}, i \in \boldsymbol{T} \tag{12}$$

$$c \ge \max_{1 \le k \le N_W} \sum_{i=1}^{N_T} \sum_{r=1}^{N_R} t_{i,r} \cdot x_{i,k} \cdot y_{r,k} \tag{13}$$

$$c \ge \sum_{i=1}^{N_T} tul_i \cdot z_{i,g} + \sum_{k=1}^{N_W} tss \cdot u_{g,k} + tt_0, \quad \forall g \in \boldsymbol{G} \tag{14}$$

Table 1
Symbol definitions.

Notation	Explanation
Sets and the corresponding indices:	
T	The set of assembly tasks, where $T = \{i\}_{N_T}$;
W	The set of workstations, where $W = \{k\}_{N_W}$;
R	The set of robots, where $R = \{r\}_{N_R}$;
G	The set of trailers, where $G = \{g\}_{N_G}$;
P_i	The set of the predecessors of task i;
S_i	The set of the successors of task i;
Decision variables:	
$x_{i,k}$	$x_{i,k} \in \{0,1\}$, if task i is assigned to workstation k, $x_{i,k} = 1$; otherwise, $x_{i,k} = 0$;
$y_{r,k}$	$y_{r,k} \in \{0,1\}$, if robot r is assigned to workstation k, $y_{r,k} = 1$; otherwise, $y_{r,k} = 0$;
$z_{i,g}$	$z_{i,g} \in \{0,1\}$, if the necessary parts of task i are assigned to trailer g, $z_{i,g} = 1$; otherwise, $z_{i,g} = 0$;
zz_g	$zz_g \in \{0,1\}$, if trailer g is enabled, $zz_g = 1$; otherwise, $zz_g = 0$;
$u_{g,k}$	$u_{g,k} \in \{0,1\}$, if trailer g needs to unload parts at workstation k, $u_{g,k} = 1$; otherwise, $u_{g,k} = 0$;
Intermediate variables:	
c	Cycle time;
nt	Number of the enabled trailers;
$fcost$	Fuel consumption cost;
Problem parameters:	
m_i	The number of parts required for assembling task i;
$t_{i,r}$	The duration time of task i processed by the robot r;
tul_i	Loading and uploading time for the parts of task i;
tt_0	The fixed path time;
tss	Starting and stopping time;
cw	The capacity of the container of each workstation;
ct	The capacity of each trailer;
f_0	The fuel consumption of an unloaded trailer;
f_c	The fuel consumption per unit weight of the trailer load per unit distance traveled;
lw	The distance between the adjacent workstations;

$$fcost = \sum_{g=1}^{N_G} \left(\max_{1 \leqslant k \leqslant N_W} k \cdot u_{g,k} \cdot lw \cdot f_c + f_0 \right) \cdot zz_g \tag{15}$$

where the multi objectives of this paper are stated in (4), encompass the minimization of both the cycle time and total fuel cost. Constraint (5) indicates that each task must be assigned to a workstation once and only once, while constraint (6) indicates that the robot must be assigned to a workstation once and only once. Constraint (7) indicates for each task i, its parts must be assigned to a trailer for transportation once and only once. Constraint (8) shows the necessity for trailer g to serve workstation k is established, based on the allocation of necessary parts from workstation k to trailer g. Constraint (9) ensures the precedence constraints between the tasks are satisfied, where each task's predecessors must be processed in the prior or the same workstations. Constraint (10) and (11) enforce the capacity constraints of the containers and trailers are satisfied correspondingly. Constraint (12) indicates that if parts are allocated to a trailer g, then the trailer is enabled. Two objectives are defined in Eqs. (13), (14), and (15). Constraint (13) defines that the cycle time is the maximum workstation time. Constraint (14) ensures the parts' traveling duration time is shorter than the cycle time, guaranteeing continuous part feeding for task assembling. Eq. (15) gives the calculation of the total fuel cost, encompassing both loaded and unloaded fuel consumption. The term for loaded fuel consumption is related to the last workstation the trailer visited.

2.3. Problem encoding and decoding

For the JRALB–FP, a solution contains four parts, i.e., the task assignment to workstations, the robot assignment to workstations, the number of enabled trailers, and the part assignment to trailers of all tasks. In this paper, we adopt a heuristic approach to assign the robots to each workstation to ensure the shortest possible time to complete tasks according to the task to workstation assignment. Then, the following scheme is used to encode the solution. Let Π denote the task assignment to workstations, nt denote the number of enabled trailers, and Φ denote assignment of parts to trailers. To ensure the feasibility of a solution, the task assignment must satisfy the precedence constraints. A solution is encoded as:

$$[\Pi, nt, \Phi] = [\pi^1, \pi^2, \ldots, \pi^{N_W}; nt; \phi^1, \phi^2, \ldots, \phi^{nt}] \tag{16}$$

Table 2
Illustration of encoding mechanism.

	$k=1$	$k=2$	$k=3$	f_c	f_0
$g=1$	1,3,5	2		2	1
$g=2$		6	4,7,8		
	$r=1$	$r=2$	$r=3$	tul_i	m_i
$i=1$	1	2	4	1	1
$i=2$	3	1	5	2	1
$i=3$	1	2	6	2	1
$i=4$	6	3	1	1	1
$i=5$	1	2	4	1	1
$i=6$	3	1	5	2	1
$i=7$	4	5	1	1	1
$i=8$	3	2	1	1	1

An example with eight tasks, three workstations, two trailers is adopted to illustrate the encoding and decoding mechanism. Three types of robots, and the task processing periods are shown in Table 2. In accordance to the precedence constraints in Fig. 2, a feasible solution can be encoded as:

$$[(1,3,5;2,6;4,7,8),2,(1,2,3,5;4,6,7,8)]$$

Based on it, the fastest robot to complete the tasks within a certain workstation is allocated to the workstation following the assumption of [36]. Additionally, trailer 1 is responsible for workstation 1 and 2, trailer 2 is responsible for workstation 2 and 3 as Table 2 shows.

For decoding phase, let k_m^g represent the last workstation for which trailer g is responsible. Trailer 1 completes the delivery of all loaded parts after workstation 2 and becomes unloaded, trailer 2 completes the delivery of all loaded parts after workstation 3 and becomes unloaded, so we have $k_m^1 = 2$ and $k_m^2 = 3$. The fuel consumption of trailer 1 can be calculated as $2 \times lw \times f_c + f_0$, and for trailer 2, it can be calculated as $3 \times lw \times f_c + f_0$. We use wt_k to denote the workstation time, representing the completion time of the last task assigned to workstation k, and tt_g to denote the transportation time of trailer g. In addition, to calculate the cycle time, for the transportation time, we have $tss = 1$, $tt_0 = 1$, $lw = 1$, $cw = 5$, $ct = 5$, and then the transportation time of these two trailers can be achieved as: $tt_1 = 9$, and $tt_2 = 7$. For the workstation time, we have $wt_1 = 3$, $wt_2 = 2$, and $wt_3 = 3$. Thus, $c = 9$ and $fcost = 12$ according to (13)–(15).

3. KEDA for JRALB–FP

3.1. Properties of JRALB–FP

In this section, several properties of the JRALB–FP are analyzed, which can guide the design of optimization operators of our proposed algorithm.

Property 1. *Two objectives considered in this paper, namely c and $fcost$, are conflicting.*

Proof. The cycle time of an assembly factory is influenced by both task assembling and part transporting, expressed as $c = \max(wt_{max}, tt_{max})$,

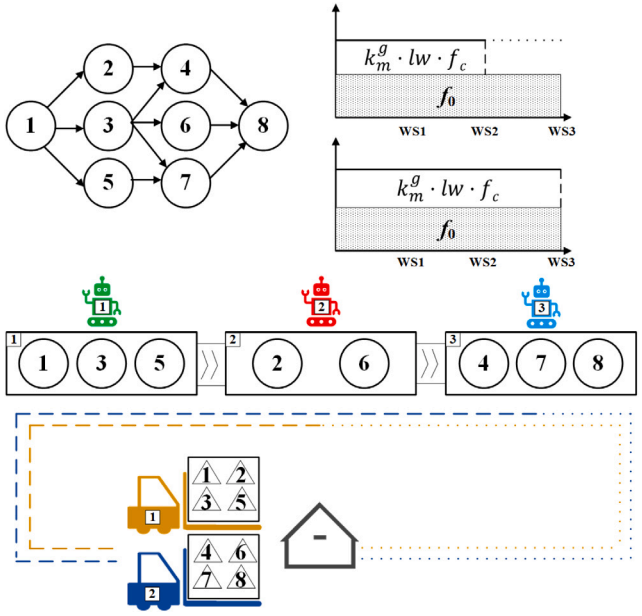


Fig. 2. Illustration of the example and its decoding mechanism.

where $wt_{max} = \max_k wt_k$, and $tt_{max} = \max_g tt_g$. We define the set of enabled trailers as $|ET|$ and $ET = nt$, the set of tasks allocated to trailer g as T_g, and the set of workstations served by the trailer g as W_g. Then, tt_g can be calculated as:

$$tt_g = \sum_{i\in T_g} tul_i + \sum_{k\in W_g} tss + tt_0, \forall g \in ET \tag{17}$$

Based on (14), we have $c \geqslant \max T_g$. Considering a given schedule, let us focus on the trailer g_1 with the highest transportation time, denoted as $g_1 = argmax_{g\in ET}\ tt_g$. If we introduce another trailer g' into ET and move one task from T_g to trailer g', then the uploading time $\sum_{i\in T_{g_1}} tul_i$ will decrease. Furthermore, the starting and stopping time $\sum_{k\in W_{g_1}} tss$ will not increase. As a result, tt_{g_1} will decrease, subsequently leading to a decrease in the lower bound of c. Enabling more trailers has the potential to reduce c.

On the contrary, the calculation of $fcost$ can be derived from (15), as shown in (18):

$$fcost = \sum_{g\in ET} \max_{k\in W_g} k \cdot lw \cdot f_c + f_0 \cdot nt \tag{18}$$

It can be seen from (18) that, enabling more trailers results in an increase in the unloading fuel consumption, represented by $f_0 \cdot nt$. Meanwhile, it is difficult to decrease the first term of (18) by enabling more trailers. Therefore, it can be inferred that having more enabled trailers leads to a higher $fcost$ and a lower c, while having fewer enabled trailers reduces $fcost$ and may increase c. These two objectives are conflicting.

Property 2. *When given the maximum workstation time (wt_{max}), the minimum number of enabled trailers (nt) can be determined using (19) without extending the cycle time. The equation for determining the lower bound for nt is:*

$$nt \geqslant \frac{\sum_{i=1}^{N_T} tul_i}{wt_{max} - tss - tt_0} \tag{19}$$

Proof. To ensure that the transportation time does not affect the cycle time c, it is necessary to maintain $tt_{max} \leqslant wt_{max}$. Additionally, the total transportation time tt_{total} under a given nt is determined by (20).

$$tt_{total} = \sum_{i=1}^{N_T} tul_i + \sum_{g=1}^{nt} |W_g| \cdot tss + tt_0 \cdot nt \tag{20}$$

where tt_0 denotes the fixed path time for each trailer, $\sum_{i=1}^{N_T} tul_i$ denotes the total uploading and loading time for the trailers to transport parts of N_T tasks, and tss is the implicit starting and stopping time for serving workstations. Since each trailer serves at least one workstation, it holds that $|W_g| \geqslant 1$. The average transportation time tt_{avg} under nt trailers can be expressed at $tt_{avg} = tt_{total}/nt$. The lower bound for tt_{avg} is given by (21) as follows:

$$tt_{avg} = \frac{\sum_{i=1}^{N_T} tul_i}{nt} + tss + tt_0 \tag{21}$$

It is a known fact that $tt_{max} \geqslant tt_{avg}$. Considering the assumption $tt_{max} \leqslant wt_{max}$, the conclusion can be drawn as shown in (19), which provides a reference for the number of necessary enabled trailers.

Property 3. *To guarantee that all the parts of the tasks are transported to the workstation during the assembly cycle time, the lower bound of nt can be calculated using (22).*

$$nt \geqslant \left\lceil \frac{\sum_{i=1}^{N_T} m_i}{ct} \right\rceil \tag{22}$$

Proof. If all the parts are transported by the trailers on average, the number of trailers will be minimized and the lower bound is held.

Property 4. *The upper bound of nt is N_T. When $nt = N_T$, the transportation time tt_g of each trailer g can be calculated as (23):*

$$tt_g = tul_i \cdot z_{i,g} + tss + tt_0 \tag{23}$$

Proof. Since the parts of a particular task cannot be divided and transported by different trailers, the upper bound of nt is N_T. Under this scenario, tt_{min} is achieved, which can be expressed as $\min_i tul_i + tss + tt_0$.

Property 5. *For all enabled trailers g, if $|W_g| \geqslant 2$, the fuel cost $fcost$ can be reduced by transferring the parts prepared for k_2 from trailer g to other trailers, provided that its transportation time does not exceed the current cycle time c, whereas k_2 satisfies:*

$$k_2 = \arg\max_{k\in W_g} k, \quad and \quad \exists g^* \neq g, k_2 \in W_{g^*} \tag{24}$$

Proof. Let $T(g,k)$ be the task set of tasks assigned to the workstation k and their parts are transported by trailer g. Simultaneously, we have $g^* \neq g, k_2 \in W_{g^*}$. If $tt_{g^*} + \sum_{i\in T(g,k_2)} tul_i \leqslant c$ holds, then inserting $T(g,k_2)$ into T_{g^*}. After the adjustment, the updated transportation time tt' and the updated fuel cost $fcost'$ can be calculated as:

$$T'_{g^*} = T_{g^*} \cup T(g,k_2) \tag{25}$$

$$T'_g = T_g \backslash T(g,k_2) \tag{26}$$

$$tt'_{g^*} = tt_{g^*} + \sum_{i\in T(g,k_2)} tul_i \leqslant c \tag{27}$$

$$tt'_g = tt_g - \sum_{i\in T(g,k_2)} tul_i < tt_g \leqslant c \tag{28}$$

$$fcost' = fcost - (k_2 - k_1) \cdot lw \cdot f_c,\ k_1 = \arg\max_{k\in W'_g} k \tag{29}$$

where it is clear that $k_1 < k_2$, leading to $fcost' < fcost$. This property enables us to develop an operator optimizing the assignment of parts to trailers based on the allocation of tasks to workstations.

3.2. The probability model, and its sampling and updating schemes

3.2.1. The probability model

In this paper, a probability model is adopted to generate task-workstation allocations, wherein each value $p_{i,k}$ represents the probability of a particular task being allocated to a workstation. This

probability matrix is sized $N_T \times N_W$, where N_T refers to the number of tasks, and N_W to the number of workstations. Obviously, the sum of the elements in each row of the matrix is 1. The probability model can be represented as (30), where g represents the gth iteration of evolution:

$$P(g) = \begin{bmatrix} p_{1,1}(g) & p_{1,2}(g) & \cdots & p_{1,N_W}(g) \\ p_{2,1}(g) & p_{2,2}(g) & \cdots & p_{2,N_W}(g) \\ \vdots & \vdots & \ddots & \vdots \\ p_{N_T,1}(g) & p_{N_T,2}(g) & \cdots & p_{N_T,N_W}(g) \end{bmatrix} \tag{30}$$

3.2.2. Initialization and sampling

To initialize individuals with diversity, the probability model is initialized uniformly, i.e., $p_{i,k}(0) = \frac{1}{N_W}$. During the gth iteration, task-to-workstation assignment: Π is sampled based on $P(g)$. First, the number of tasks allocated to a certain workstation k is estimated. Let $\overline{\omega}_k(g)$ represent the estimated number of tasks assigned to a specific workstation k at the gth iteration, which can be calculated according to the probability values within the probability model as follows:

$$\overline{\omega}_k(g) = \begin{cases} \lfloor \sum_l p_{l,k}(g) \rfloor + 1, \text{if } \ q \le \sum_l p_{l,k}(g) - \lfloor \sum_l p_{l,k}(g) \rfloor \\ \lfloor \sum_l p_{l,k}(g) \rfloor, \text{otherwise} \end{cases} \tag{31}$$

where $q \in (0,1)$ is a random number. For example, the sum of the first column of the probability model is 3.4, indicating that there is a 0.4 probability that 4 tasks will be allocated to the first workstation and a 0.6 probability of allocating 3 tasks. Additionally, the estimated number of tasks allocated to the last workstation is as follows:

$$\overline{\omega}_{N_W}(g) = N_T - \sum_{k=1}^{N_W-1} \overline{\omega}_k(g) \tag{32}$$

Second, task-to-workstation assignment, denoted as Π, is generated in accordance with $P(g)$ and $\overline{\omega}(g)$. A task is defined as available if all its predecessor tasks have already been allocated to workstations. An available task set AT a is maintained and updated during the sampling of the probability model, where $AT := \{i\}, P_i \in \pi^*, i \notin \pi^*$, and π^* represents the set of tasks that have been allocated. A tabu set $Tabu$ is used to collect the tasks violating capacity constraints during the phase of allocation. For each workstation k, a task i is chosen from AT based on a roulette-wheel method [37] with the probability array $p_{:,k}(g)$. If the assigned task i exceeds the capacity of the workstation k, it will be pushed back to AT. The selection continues until a certain task is assigned to the workstation k, or all tasks in AT are traversed. Finally, the fastest robot is assigned to each workstation.

Based on Π, the number of enabled trailers, denoted as nt, and the corresponding task-to-trailer allocation, denoted as Φ, are generated. The minimum value of nt is given by Property 3. To optimize trailer capacity utilization, the average load of each trailer is set to exceed 60% of its maximum capacity in this paper. Consequently, the permissible range of nt is defined in Eq. (33), within which nt is randomly selected.

$$\left\lceil \frac{\sum_{i=1}^{N_T} m_i}{ct} \right\rceil \leqslant nt \leqslant \left\lceil \frac{\sum_{i=1}^{N_T} m_i}{0.6 \times ct} \right\rceil \tag{33}$$

The next consideration is the part-to-trailer assignment, denoted as Φ. To minimize both transportation time (tss) and energy consumption, each trailer should serve fewer workstations near one another following Property 5. To avoid disperse part feeding, the task-to-workstation allocation scheme is adopted to guide the part-to-trailer assignment. A load ratio (act) is generated randomly in the range of 60% to 100%. For each newly enabled trailer g, its load limit can be calculated as $act \times ct$, then the parts are allocated to the trailer g following the task sequence until the load limit is reached. If the capacity constraint is not satisfied, a new trailer is enabled and the remained parts are assigned to the new trailer. The pseudo-code can be referred to as Algorithm 1, where wm_k is defined as the sum of the part quantities corresponding to the tasks that have been assigned to the kth workstation, and tm_g is defined as the sum of the part quantities corresponding to the tasks that have been assigned to the gth trailer.

Thus, $[\Pi, nt, \Phi]$ is sampled based on the heuristic-oriented sampling mechanism as Algorithm 1 shows. To make the pseudo code clearer, some functions are defined as follows.

- Update(X): Update the set X.
- x = Select(Y, z): Select task x from set Y based on the probability matrix z.
- Calculate(x): Calculate the value of x.
- Remove(x, y): Remove task x from workstation y.
- Swap(x, y): Swap task x and task y.
- Insert(x, y): Insert task x to workstation y.
- Determine(x): Determine the value of x based on the corresponding information.

Algorithm 1: Sampling

Input: $P(g)$
Output: $indi : [\Pi, nt, \Phi]$
Calculate $\overline{\Omega}(g) = \{\overline{\omega}_1(g), \overline{\omega}_2(g), \ldots, \overline{\omega}_{N_W}(g)\}$ from ((31), (32))
for $k \in [1, N_W]$ **do**
  $\omega_k(g) = 0$;
  while $\omega_k(g) < \overline{\omega}_k(g)$ **do**
    Update(AT);
    Update($Tabu$);
    i =Select($AT \backslash Tabu$, $P(g)$);
    π^k.append(i);
    Assign the fastest robot r to workstation k;
    Calculate (wt_k);
    Update (wm_k);
    $\omega_k(g)$+ = 1;
  end
end
$\Pi = \{\pi^1, \pi^2, .., \pi^k, ..\}$;
Initialize $nt = 1$, $g = 1$, and $tm_1 = 0$;
for $\pi^k \in \Pi$ **do**
  for $i \in \pi^k$ **do**
    if $tm_g + m_i \leqslant act \times ct$ **then**
      ϕ^g.append(i);
      Update (tm_g);
    end
    else
      nt+ = 1;
      $g \rightarrow next(g)$;
    end
  end
end
$\Phi = \{\phi^1, \phi^2, .., \phi^g, ..\}$;
return $indi = [\Pi, nt, \Phi]$

3.2.3. Fitness calculation and probability matrix updating

After the population is initialized, each individual is evaluated according to (14) and (15). The corresponding pseudo-code is given in Algorithm 2. Then, the individuals are non-dominated sorted, and the first $N_P \times \beta$ individuals are chosen as the elite population, where N_P denotes the size of the population and $\beta \in (0,1)$ denotes the ratio of the elite population. The probability values are updated with the elite population based on the incremental learning method [38]. Specifically,

$$p_{i,k}(g+1) = (1-\alpha) \times p_{i,k}(g) + \alpha \times \Sigma_q I^q_{i,k}(g)/|Q| \tag{34}$$

where $\alpha \in (0,1)$ represents the learning rate, Q represents the elite population, whereas $|Q| = N_P \times \beta$ denotes the number of elite individuals,

and $I_{i,k}^{q}(g)$ is the indicator corresponding to the qth elite individual.

$$I_{i,k}^{q}(g)=\begin{cases}1, \text{Task } i \text{ is assigned to workstation } k\\ \text{in } indi\ d \text{ during the } g\text{th generation}\\ 0, \text{otherwise}\end{cases} \tag{35}$$

3.3. Knowledge-guided local search operators

To enhance the exploitation capability, two local search operators are designed under the guidance of the problem properties, which are used to optimize the cycle time and fuel consumption cost respectively. Local search operations are performed in turn on each elite individual in each iteration. The pseudo-code of both local search operators are shown in Algorithm 3–6.

3.3.1. Local search operator for cycle time

As cycle time is influenced by both workstation time and transportation time, task-to-workstation and part-to-trailer both make sense.

Algorithm 2: Evaluation

Input: $indi$
Output: $fitness$
for $k \in N_W$ **do**
  Calculate(wt_k);
end
for $g \in N_G$ **do**
  Calculate(tt_g);
end
$c = max(max(tt_g), max(wt_k))$;
Calculate($fcost$);
Fitness transformation.
return $fitness$

Algorithm 3: Local search for cycle time

Input: $indi$
Output: $indi\_optimized$
Calculate(wt_{max}, tt_{max});
if $wt_{max} \geqslant tt_{max}$ **then**
  Algorithm 4;
end
else
  Algorithm 5;
end
return $indi\_optimized$

First, if $wt_{max} > tt_{max}$, it indicates that the allocation of tasks to the workstation limits the cycle time. The workstation with wt_{max} is noted as the bottleneck workstation WSB, and we aim to insert the tasks in WSB to other workstations. If there are no predecessors of task i allocated in WSB, then it can be inserted into the previous workstation (WSP). If there are no successors of task i in WSB, then it can be inserted into the next workstation (WSN). If wt_{max} is reduced and the capacity constraints are satisfied, then the insertion operator is implemented.

In addition, swap operators are implemented to reduce wt_{max} as well. The tasks in WSB are swapped to the previous or next workstations, if the precedence constraints and container capacity are satisfied and wt_{max} is reduced. The operators are illustrated in Fig. 3.

On the other hand, if the $wt_{max} < tt_{max}$, then transportation time is needed to be optimized. First, the trailer corresponding to the tt_{max} is identified, referred to as the bottleneck trailer TB. The first task TB_f in the TB is inserted into the previous trailer TBP or the last task TB_l in the TB is inserted into the next trailer TBN as the part inserted into adjacent trailers can minimize the impact on trailer fuel consumption. The operator is implemented if container capacity is satisfied and tt_{max} is reduced. The operators are illustrated in Fig. 4.

Algorithm 4: Local search for workstation time

Input: $indi$
Output: $indi\_optimized$
Calculate(wt_{max});
for $i \in WSB$ **do**
  Determine($P_i(S_i)$);
  if $P_i(S_i) \notin WSB$ **then**
    Insert(i, $WSP(WSN)$);
    Calculate($wt_{WSP}(wt_{WSN})$, $wm_{WSP}(wm_{WSN})$);
    if $wt_{WSP}(wt_{WSN}) < wt_{max}$ ***and*** $wm_{WSP}(wm_{WSN}) \leqslant cw$ **then**
      Remove(i, WSB);
    end
    else
      Remove(i, $WSP(WSN)$)
    end
  end
end
Update(WSB);
Calculate(wt_{max});
for $i \in WSB$ **do**
  Determine($P_i(S_i)$);
  if $P_i(S_i) \notin WSB$ **then**
    for $j \in WSP(WSN)$ **do**
      Determine($S_j(P_j)$);
      if $S_j(P_j) \notin WSP(WSN)$ ***and*** $j \notin P_i(S_i)$ **then**
        Swap(i, j);
        Calculate($wt_{WSP}(wt_{WSN})$, wt_{WSB}, $wm_{WSP}(wm_{WSN})$, wm_{WSB});
        if $wt_{WSP}(wt_{WSN}) > wt_{max}$ ***or*** $wt_{WSB} > wt_{max}$ ***or*** $wm_{WSP}(wm_{WSN}) > cw$ ***or*** $wm_{WSB} > cw$ **then**
          Swap(i, j);
        end
      end
    end
  end
end
return $indi\_optimized$

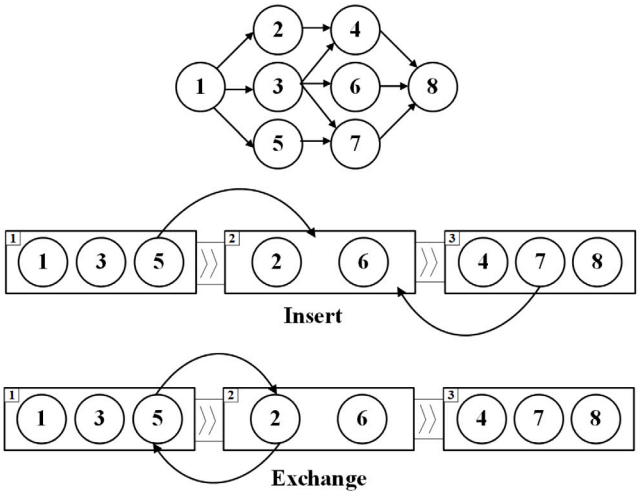


Fig. 3. Local search for workstation time.

3.3.2. Local search operator for fuel consumption cost

The fuel consumption of trailers is determined by the last workstation served, which can be lowered as Property 5. For each trailer

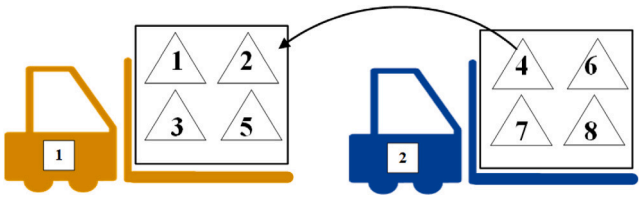


Fig. 4. Local search for trailer time.

g, the number of workstations it serves without violating the capacity constraints or raising tt_g. This strategy can effectively reduce trailer fuel consumption. The operators are depicted in Fig. 5. k_m^g represent the last workstation for which trailer g is responsible. According to the initial allocation in the figure, we have $k_m^1 = 2$, $k_m^2 = 3$. After transferring the materials of task 2 from trailer 1 to trailer 2, we have $k_m^1 = 1$, $k_m^2 = 3$. The load running distance of trailer 1 has been reduced, thereby reducing fuel consumption.

Algorithm 5: Local search for trailer time

Input: $indi$
Output: $indi\_optimized$
Calculate(tt_{TBP}, tt_{TBN});
Determine(m_f of TB's first task TB_f);
Determine(m_l of TB's last task TB_l);
if $tt_{TBP} \leqslant tt_{TBN}$ **then**
  if $tm_{TBP} + m_f \leqslant ct$ **then**
    Insert(first task of TB, TBP);
    Calculate(tt_{TBP});
    if $tt_{TBP} \leqslant tt_{max}$ **then**
      Remove(TB_f, TB);
    end
  end
  else
    Remove(TB_f, TBP)
  end
end
else
  if $tm_{TBN} + m_l \leqslant ct$ **then**
    Insert(last task in TB, TBN);
    Calculate(tt_{TBN});
    if $tt_{TBN} \leqslant tt_{max}$ **then**
      Remove(TB_l, TB);
    end
  end
  else
    Remove(TB_l, TBN)
  end
end
return $indi\_optimized$

3.4. Flowchart and complexity analysis

To solve the JRALB–FP with the criteria of minimizing both cycle time and fuel consumption cost, we develop a knowledge-guided EDA, where two knowledge-guided local search operators are embedded in the EDA. The flowchart of KEDA is shown in Fig. 6, and the main steps are concluded as follows.

- Step 1: Uniformly initialize the probability matrix. Set the Pareto set as an empty set. Then, for each evaluation generation g:
- Step 2: Generate N_P individuals by sampling the probability matrix through problem-oriented heuristic based methods.

Algorithm 6: Local search for fuel cost

Input: $indi$
Output: $indi\_optimized$
for $g \in ET$ **do**
  Calculate(W_g);
  if $W_g \geqslant 2$ **then**
    Calculate(k_2 of g);
    if $\exists g^* \neq g, k_2 \in W_{g^*}$ **then**
      if $tt_{g^*} + \sum_{i \in T(g,k_2)} tul_i \leqslant c$ **then**
        if $tm_{g^*} + m_i \leqslant ct$ **then**
          Insert(i, T_{g^*});
        end
      end
    end
  end
end
Update($T_{g^*}, T_g, tt_{g^*}, tt_g$);
Calculate($fcost'$);
return $indi\_optimized$

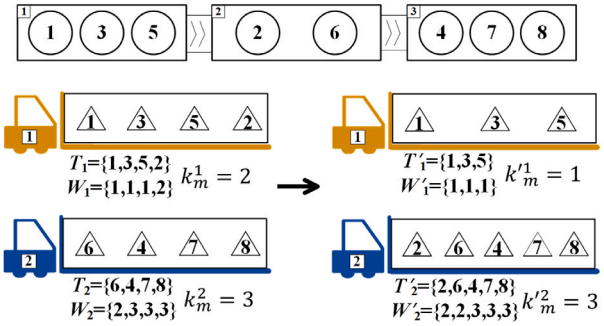


Fig. 5. Local search for fuel cost.

- Step 3: Calculate the values of cycle time and total fuel cost for each individual.
- Step 4: Perform non-dominated sorting for all individuals and select the first $\beta \times 100\%$ individuals as the elite population.
- Step 5: Perform two local search operators in turn for each elite individual.
- Step 6: Update the Pareto set and the probability matrix with a learning rate of α based on the elite individuals.
- Step 7: If the stopping condition is met, output the Pareto set; otherwise, return to Step 2.

According to the flowchart, the time complexity of our proposed algorithm is analyzed as follows:

- **Sampling**: The sampling of the 'task-workstation' assignment has an $O({N_T}^2)$ complexity, and the sampling of the 'task-trailer' assignment has an $O(N_T \times N_G)$ complexity. It can be summarized that the complexity of the sampling mechanism is of $O({N_T}^2)$ as we have $N_T > N_G$;
- **Fitness Calculation**: For each individual, the fitness calculation has an $O(N_W + N_G)$ complexity. The total complexity for the population can be concluded as $O((N_W + N_G) \times N_P)$ with the population size of N_P,;
- **Non-dominated sorting and elite solution selection**: It is known that the complexity of non-dominated sorting is $O({N_P}^2)$, and the complexity of the elite individual selection is $O(N_p)$;
- **Local search**: For each elite individual, we have: (1) For the local search operator designed to optimize the cycle time, the complexity can be summarized as $O({N_T}^2)$; (2) For the local search

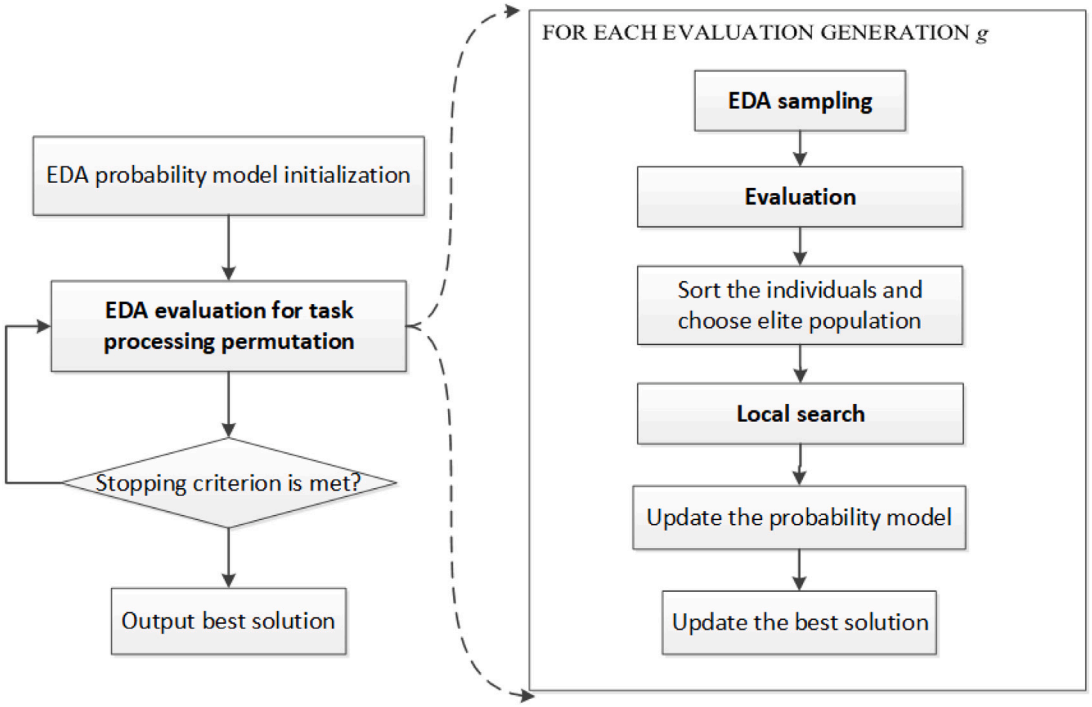


Fig. 6. Framework of KEDA.

operator designed to optimize the fuel cost, the complexity can be summarized as $O(2)$;

Thus, the complexity of the proposed algorithm is $O(iter \times (N_T{}^2 + (N_W+N_G)\times N_P+N_P{}^2+N_P\times\beta\times N_T{}^2))$, where $iter$ denotes the number of iteration, and $\beta \in (0,1)$ denotes the ratio of the elite population selected from the population. Considering $N_P \times \beta > 1$, the total algorithm complexity is of the order $O(iter \cdot N_p \cdot (N_W + N_G + N_p + \beta \cdot N_T^2))$.

4. Numerical experimental results

4.1. Experimental settings

To verify the effectiveness of the KEDA in solving the energy efficient JRALB–FP, test cases adopted from https://assembly-line-balancing.de/ is used, where $N_T = \{20, 50, 70, 83, 100, 148, 220, 297\}$. These data are used for cycle optimization of the basic assembly line scheduling problem. To verify the effectiveness of our multi-objective problem, the dataset has been improved by incorporating energy related data. The fuel costs of the trailers are added to the benchmark cases. On the other hand, different scales of workstation numbers are considered for each scale of tasks. For $N_T = 20$, N_W is set {4,5}; $N_T = \{50, 70, 83\}$, N_W is set {8,10}; $N_T = \{100, 148\}$, N_W is set {10,12}, and for $N_T = \{220, 297\}$, N_W is set {10,15}. For each scale of N_T, 10 independent cases are adopted, and each case is run 10 times respectively. The computing environment is 13th Gen Intel(R) Core(TM) i7-13790F running at 2.10 GHz with 32 GB of RAM, and the operating system is Windows 11.

Four parameters are considered in our proposed algorithm: learning rate (α), the ratio of elite population (β), the maximum number of iterations ($iter$), and the size of the population (N_P). To determine the aforementioned parameters, the Taguchi method of the design-of-experiment method (DOE) is used to investigate how the change of parameters affects the performance of the proposed algorithm. In this paper, α is chosen and set at these three levels: 0.05, 0.1, and 0.15; β is set at three levels: 5%, 10%, and 15%; $iter$ is set at three levels: 60, 80, and 100; N_P is set at three levels: 60, 80, and 100. 3^4 full-factorial experiments are employed in a randomly chosen test case.

Table 3
The response value.

α	β	$iter$	N_P	CON
0.05	0.05	60	60	0.042614
0.05	0.10	80	80	0.157440
0.05	0.15	100	100	0.038352
0.10	0.05	80	100	0.154126
0.10	0.10	100	60	0.161323
0.10	0.15	60	80	0.070887
0.15	0.05	100	80	0.184429
0.15	0.10	60	100	0.110471
0.15	0.15	80	60	0.080357

Table 4
Rank of KEDA parameters.

Level	α	β	$iter$	N_P
1	0.07947	0.12706	0.07466	0.09476
2	0.12878	0.14308	0.13064	0.13759
3	0.12509	0.06320	0.12803	0.10098
Delta	0.04931	0.07988	0.05598	0.04282
Rank	3	1	2	4

For each instance, 9 combinations of are tested independently and the obtained AS_{ci} ($ci = 1, 2, \ldots, 9$) are stored. The final AS (FAS) are obtained by integrating $AS_1, AS_2, \ldots, AS_9$. Then, the contribution of a certain combination (CON) is counted as $CON(ci) = |AS'_{ci}|/|FE|$, where $AS'_{ci} = \{X_l \in AS_{ci} | \exists X_{l'} \in FAS, X_l = X_{l'}\}$. The average CON of each combination is used as the response value (RV), and the RVs are shown in Table 3. The ranks of parameters are shown in Table 4, and the main effect plot is generated in Fig. 7. It can be seen that $N_P = 80$, $iter = 80$, $\alpha = 0.1$, and $\beta = 10\%$ are suggested as the parameters in the simulation experiments, and the parameter settings are presented in Table 5.

4.2. Comparison to Gurobi

In this section, the proposed MIP model is solved using Gurobi (version 10.0.3) to validate its correctness. The results are compared with

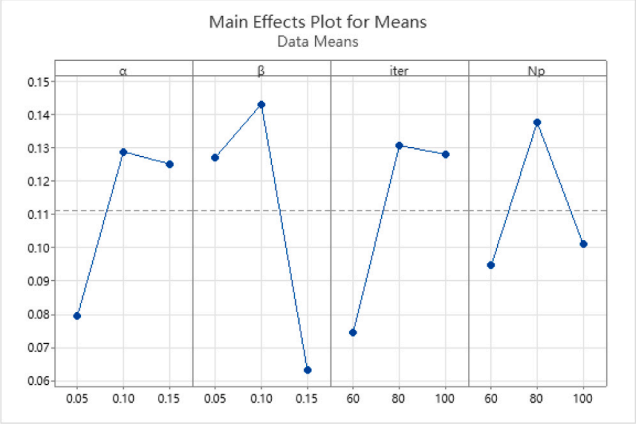


Fig. 7. Main effect plot.

Table 5
Parameter settings.

Parameter	Settings
iter	80
N_P	80
α	0.1
β	10%

Table 6
Comparison results of IGD, time, and C metric between KEDA and gurobi.

N_T	$No.$	IGD		C metric		Time(s)	
		KEDA	Gurobi	$C_{k,g}$	$C_{g,k}$	KEDA	Gurobi
20	1	16.02	0.00	0.00	1.00	58.8	3168
	2	73.19	0.00	0.00	1.00	58.3	1986
	3	38.55	0.00	0.00	1.00	58.1	5400
	4	31.08	0.00	0.00	1.00	58.6	2922
	5	18.26	0.00	0.00	0.67	59.1	3756
50	1	230.89	0.00	0.00	1.00	195.7	5400
	2	138.19	0.00	0.00	0.45	207.4	5400
	3	175.56	0.00	0.00	1.00	196.0	5400
	4	117.07	0.00	0.00	0.80	190.8	5400
	5	139.60	15.22	0.33	0.67	204.5	5400
70	1	76.82	6.73	0.00	0.00	276.2	5400
	2	0.86	0.00	0.00	0.00	278.2	5400
	3	21.82	0.00	0.00	0.20	387.3	5400
	4	4.10	35.01	0.50	0.00	279.7	5400
	5	167.32	0.00	0.00	0.88	276.6	5400
83	1	85.00	641.72	0.67	0.00	393.0	5400
	2	95.83	128.10	0.50	0.00	743.9	5400
	3	108.16	641.90	0.50	0.20	740.0	5400
	4	42.05	20.40	1.00	0.00	757.7	5400
	5	7.67	260.90	0.50	1.00	747.8	5400
100	1	302.28	0.00	0.00	1.00	487.6	10800
148	1	93.89	0.00	0.00	0.73	1081.7	10800
220	1	0.00	17071.83	1.00	0.00	3106.5	10800
297	1	37.83	40844.31	1.00	0.00	4832.6	10800

those obtained from KEDA. Both C metrics and IGD are calculated, and the CPU time are recorded and listed in Table 6, where C(KEDA,gurobi) and C(gurobi,KEDA) are denoted as $C_{k,g}$ and $C_{g,k}$ respectively for simplification.

For gurobi, we normalize the objectives and set three weights $(W_{o1}, W_{o2}) = [(0.2, 0.8), (0.5, 0.5), (0.8, 0.2)]$, (W_{o1}, W_{o2}) represents the weights of the two objectives, respectively, and each weight is run once to obtain three solutions. For KEDA, run it independently ten times to obtain the Pareto front combinations. The actual Pareto front used is the combination obtained by all the algorithms used in the experiment.

It can be seen from Table 6 that, for the small scale instances ($N_T \leqslant 70$), KEDA are dominated by the Gurobi solutions. When $N_T = 83$, the scale of decision variables increases, and our solutions dominate

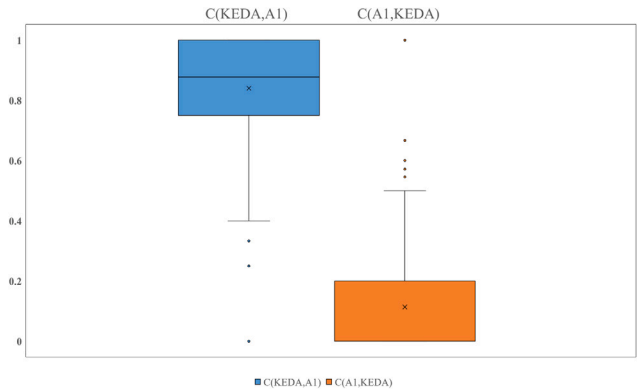


Fig. 8. Boxplot of C metric between KEDA and A1.

50%–100% of Gurobi solutions produced within 5400 s time limit. In addition, for the larger scale instances ($N_T \leqslant 70$), the time limit is extended to 10800 s to enhance the performance of Gurobi solutions. As a results, it can be seen that Gurobi solutions dominate most KEDA solutions in two random cases in the $N_T = 100, 148$ scales, while KEDA solutions dominate all Gurobi solutions in two random cases in the $N_T = 220, 297$ scales. It can be seen from IGD values that KEDA solutions are nearer to the Pareto front.

4.3. Effect of local intensification

To demonstrate the effectiveness of two knowledge-guided local search operators, we compare the KEDA to the algorithms without local search for cycle time (denoted as A1) and local search for fuel cost (denoted as A2) respectively. For each case, the average C metric of ten independent runs is summarized in Tables 7 and 8. Additionally, the boxplots of the comparison between all C metrics of KEDA and the other algorithm are shown in Figs. 8 and 9. The T-hypothesis test with a 95% confidence level is carried out, setting alternative hypothesis H1 such that the C metric achieved by KEDA is greater than the C metric achieved by the other algorithms. If H1 is implemented significantly ($p < 0.05$), then the flag is labeled as Y, otherwise it is labeled as N. C metric C(KEDA,A1) and C(A1,KEDA) are denoted as $C_{k,1}$ and $C_{1,k}$ respectively for simplification in the tables. C(KEDA,A2) and C(A2,KEDA) are denoted as $C_{k,2}$ and $C_{2,k}$.

From Table 7, it can be seen that C(KEDA, A1) is larger than C(A1, KEDA) on most instances. Especially, for the large-scale instances ($N_T \geq 200$), the difference is statistically significant with $p < 0.05$. Fig. 8 shows that KEDA is significantly better than A1. Therefore, the local search operator designed for cycle time optimization is effective in solving this problem. Similarly, it can be seen that C(KEDA, A2) is larger than C(A2, KEDA) on most instances from Table 8. Fig. 8 demonstrates that KEDA performs better than A2, confirming the effectiveness of the local search operator designed for fuel consumption cost optimization.

4.4. Effect of heuristic-based sampling mechanism

To demonstrate the effectiveness of the heuristic-based sampling mechanism, we compare it with the proposed EDA with a random sampling mechanism (denoted as A3), i.e., the assembly tasks and their parts are randomly assigned to the workstations and trailers. For each case, the average C metric of ten independent runs is summarized in Table 9. Additionally, the boxplot of the comparison between all C metrics of KEDA and A3 is shown in Fig. 10. The T-hypothesis test with a 95% confidence level is carried out, setting alternative hypothesis H1 such that the C metric achieved by KEDA is greater than the C metric achieved by the EDA with a random sampling mechanism. If H1 is implemented significantly ($p < 0.05$), then the flag is labeled as

Table 7
Comparison results of C metric between KEDA and A1.

(N_T, N_W)	(20, 4)			(50, 8)			(70, 8)			(83, 8)			(100, 10)			(148, 10)			(220, 10)			(297, 10)		
No.	$C_{k,1}$	$C_{1,k}$	Sig	$C_{k,1}$	$C_{1,k}$	Sig	$C_{k,1}$	$C_{1,k}$	Sig	$C_{k,1}$	$C_{1,k}$	Sig	$C_{k,1}$	$C_{1,k}$	Sig	$C_{k,1}$	$C_{1,k}$	Sig	$C_{k,1}$	$C_{1,k}$	Sig	$C_{k,1}$	$C_{1,k}$	Sig
1	0.50	0.33	N	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	0.86	0.20	Y	0.86	0.18	Y	0.75	0.00	Y	1.00	0.00	Y
2	0.67	0.33	Y	0.75	0.20	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	0.86	0.13	Y	0.50	0.25	N	1.00	0.00	Y
3	1.00	0.67	Y	0.50	0.60	N	1.00	0.00	Y	1.00	0.00	Y	0.86	0.00	Y	0.80	0.20	Y	0.80	0.14	Y	1.00	0.00	Y
4	0.67	0.00	Y	0.83	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	0.80	0.25	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y
5	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	0.80	0.16	Y	0.57	0.42	N	0.86	0.16	Y	1.00	0.00	Y
6	0.67	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	0.75	0.00	Y	0.88	0.13	Y	0.86	0.11	Y	1.00	0.14	Y	1.00	0.00	Y
7	0.50	1.00	N	0.88	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.14	Y	1.00	0.00	Y	0.67	0.00	Y	1.00	0.00	Y
8	1.00	0.00	Y	0.60	0.25	Y	1.00	0.00	Y	1.00	0.00	Y	0.57	0.50	N	1.00	0.00	Y	0.80	0.20	Y	1.00	0.00	Y
9	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	0.50	0.00	Y	0.75	0.25	Y	0.80	0.16	Y	1.00	0.00	Y	1.00	0.00	Y
10	0.67	0.50	N	0.80	0.25	Y	1.00	0.00	Y	0.83	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y
(N_T, N_W)	(20, 5)			(50, 10)			(70, 10)			(83, 10)			(100, 12)			(148, 12)			(220, 15)			(297, 15)		
No.	$C_{k,1}$	$C_{1,k}$	Sig	$C_{k,1}$	$C_{1,k}$	Sig	$C_{k,1}$	$C_{1,k}$	Sig	$C_{k,1}$	$C_{1,k}$	Sig	$C_{k,1}$	$C_{1,k}$	Sig	$C_{k,1}$	$C_{1,k}$	Sig	$C_{k,1}$	$C_{1,k}$	Sig	$C_{k,1}$	$C_{1,k}$	Sig
1	0.60	0.00	Y	0.88	0.11	Y	0.86	0.20	Y	0.75	0.25	Y	0.75	0.20	Y	0.70	0.27	Y	0.63	0.33	Y	1.00	0.00	Y
2	1.00	0.00	Y	0.71	0.18	Y	1.00	0.00	Y	1.00	0.00	Y	0.80	0.13	Y	0.40	0.47	N	0.67	0.08	N	1.00	0.00	Y
3	0.67	0.25	Y	1.00	0.00	Y	0.83	0.00	Y	1.00	0.00	Y	0.88	0.09	Y	0.50	0.38	N	0.67	0.29	Y	0.83	0.00	Y
4	0.50	0.25	N	0.89	0.00	Y	0.88	0.00	Y	1.00	0.00	Y	0.89	0.20	Y	0.70	0.31	Y	1.00	0.00	Y	1.00	0.00	Y
5	0.50	0.33	Y	1.00	0.00	Y	0.92	0.00	Y	0.80	0.00	Y	0.78	0.13	Y	0.46	0.55	N	0.60	0.38	N	1.00	0.00	Y
6	0.83	0.20	Y	0.71	0.00	Y	0.83	0.00	Y	0.75	0.00	Y	0.88	0.10	Y	0.92	0.13	Y	0.83	0.00	Y	1.00	0.00	Y
7	0.67	0.50	N	1.00	0.00	Y	1.00	0.00	Y	0.80	0.00	Y	0.80	0.33	Y	0.25	0.60	N	0.86	0.00	Y	1.00	0.00	Y
8	1.00	0.00	Y	0.50	0.50	N	1.00	0.00	Y	0.83	0.00	Y	0.43	0.27	N	0.89	0.08	Y	0.86	0.20	Y	0.75	0.50	N
9	1.00	0.00	Y	1.00	0.00	Y	0.83	0.25	Y	1.00	0.00	Y	0.71	0.17	Y	0.33	0.57	N	1.00	0.00	Y	1.00	0.00	Y
10	1.00	0.00	Y	0.88	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	0.86	0.00	Y	0.40	0.33	N	0.71	0.25	Y	0.50	0.00	Y

Table 8
Comparison results of C metric between KEDA and A2.

(N_T, N_W)	(20, 4)			(50, 8)			(70, 8)			(83, 8)			(100, 10)			(148, 10)			(220, 10)			(297, 10)		
No.	$C_{k,2}$	$C_{2,k}$	Sig	$C_{k,2}$	$C_{2,k}$	Sig	$C_{k,2}$	$C_{2,k}$	Sig	$C_{k,2}$	$C_{2,k}$	Sig	$C_{k,2}$	$C_{2,k}$	Sig	$C_{k,2}$	$C_{2,k}$	Sig	$C_{k,2}$	$C_{2,k}$	Sig	$C_{k,2}$	$C_{2,k}$	Sig
1	0.00	0.67	N	0.56	0.44	N	1.00	0.00	Y	1.00	0.00	Y	0.50	0.60	N	0.43	0.36	N	1.00	0.00	Y	1.00	0.33	Y
2	0.75	0.33	Y	0.78	0.00	Y	0.50	0.25	Y	0.75	0.25	Y	0.75	0.50	Y	0.50	0.62	N	0.40	0.50	N	0.50	0.33	N
3	1.00	0.00	Y	0.67	0.20	Y	0.75	0.00	Y	0.75	0.40	Y	0.33	0.50	N	1.00	0.00	Y	0.43	0.57	N	0.50	0.20	Y
4	0.00	0.50	N	0.50	0.71	N	0.33	0.75	N	1.00	0.00	Y	0.63	0.12	Y	0.75	0.37	Y	0.75	0.00	Y	0.75	0.50	Y
5	1.00	0.00	Y	0.33	0.60	N	0.33	0.00	Y	1.00	0.00	Y	0.73	0.16	Y	0.25	0.28	N	0.67	0.16	Y	0.50	0.00	Y
6	0.67	0.00	Y	0.57	0.29	Y	0.67	0.33	N	0.80	0.00	Y	0.67	0.50	Y	0.38	0.44	N	0.13	0.37	N	0.67	0.00	Y
7	0.67	0.50	N	0.80	0.17	Y	1.00	0.00	Y	0.25	0.29	N	0.58	0.42	N	0.55	0.22	Y	1.00	0.00	Y	1.00	0.00	Y
8	1.00	0.25	Y	1.00	0.00	Y	1.00	0.25	Y	0.67	0.25	Y	0.43	0.33	N	0.38	0.16	Y	0.40	0.60	N	0.67	0.60	N
9	0.75	0.00	Y	0.50	0.33	Y	1.00	0.25	Y	0.60	0.00	Y	0.60	0.50	N	0.38	0.25	N	0.85	0.00	Y	1.00	0.00	Y
10	0.75	0.25	Y	1.00	0.00	Y	0.67	0.43	Y	0.50	0.57	N	0.57	0.40	Y	0.80	0.33	Y	0.75	0.25	Y	0.75	0.20	Y
(N_T, N_W)	(20, 5)			(50, 10)			(70, 10)			(83, 10)			(100, 12)			(148, 12)			(220, 15)			(297, 15)		
No.	$C_{k,2}$	$C_{2,k}$	Sig	$C_{k,2}$	$C_{2,k}$	Sig	$C_{k,2}$	$C_{2,k}$	Sig	$C_{k,2}$	$C_{2,k}$	Sig	$C_{k,2}$	$C_{2,k}$	Sig	$C_{k,2}$	$C_{2,k}$	Sig	$C_{k,2}$	$C_{2,k}$	Sig	$C_{k,2}$	$C_{2,k}$	Sig
1	0.67	0.33	Y	0.33	0.33	N	0.38	0.40	N	1.00	0.25	Y	0.80	0.20	Y	0.73	0.08	Y	0.71	0.11	Y	0.17	0.60	N
2	1.00	0.00	Y	0.63	0.45	N	0.82	0.14	Y	0.43	0.33	N	0.90	0.13	Y	0.28	0.73	N	0.29	0.33	N	1.00	0.00	Y
3	0.60	0.25	N	0.63	0.44	N	0.67	0.20	Y	0.91	0.20	Y	0.50	0.27	Y	0.25	0.54	N	0.67	0.43	Y	1.00	0.00	Y
4	1.00	0.00	Y	0.63	0.40	Y	0.33	0.40	N	0.80	0.20	Y	0.91	0.00	Y	0.43	0.62	N	0.29	0.38	N	1.00	0.00	Y
5	1.00	0.00	Y	0.78	0.33	Y	0.50	0.37	N	0.60	0.60	N	0.43	0.63	N	0.82	0.00	Y	0.57	0.38	Y	1.00	0.00	Y
6	0.50	0.60	N	0.71	0.50	Y	0.70	0.33	Y	0.67	0.00	Y	0.50	0.60	N	0.30	0.47	N	0.43	0.29	Y	1.00	0.00	Y
7	1.00	0.00	Y	0.44	0.33	N	0.83	0.37	Y	0.67	0.33	N	0.75	0.00	Y	0.20	0.60	N	0.71	0.29	Y	1.00	0.00	Y
8	0.50	0.33	Y	0.85	0.16	Y	1.00	0.00	Y	0.80	0.33	Y	0.88	0.36	Y	0.27	0.38	N	0.83	0.20	Y	0.67	0.00	Y
9	0.83	0.00	Y	0.50	0.50	N	0.25	0.25	N	0.50	0.50	N	0.78	0.00	Y	0.18	0.64	N	0.83	0.13	Y	1.00	0.00	Y
10	0.50	0.25	N	0.25	0.40	N	0.80	0.00	Y	0.67	0.50	Y	0.88	0.00	Y	0.75	0.50	Y	0.60	0.13	Y	0.29	0.25	N

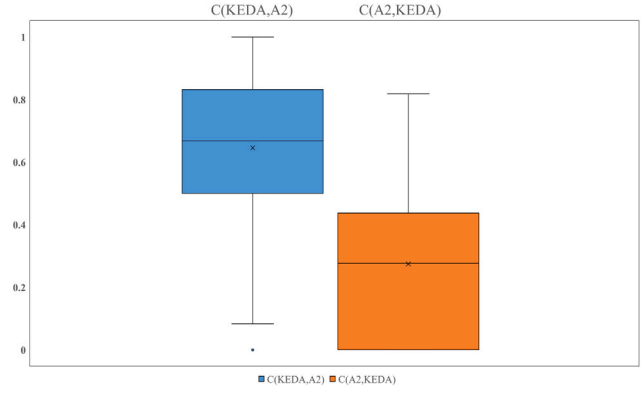


Fig. 9. Boxplot of C metric between KEDA and A2.

Y, otherwise it is labeled as N. C metrics: C(KEDA,A3) and C(A3,KEDA) are denoted as $C_{k,3}$ and $C_{3,k}$ respectively for simplification in the table.

From Table 9, it can be seen that $C_{k,3}$ is larger than $C_{3,k}$ in most instances. Fig. 10 shows that the heuristic sampling mechanism is significantly better than the random sampling mechanism in terms of the C metric. The reason is that the problem-oriented heuristic-based sampling mechanism can locate the promising solution space efficiently and then produce better solutions.

4.5. Comparison to other algorithms

In this section, two commonly used multi-objective optimization algorithms: A Multi-objective Evolutionary Algorithm Based on Decomposition (MOEA/D) [39] and Non-dominated Sorting Genetic Algorithm-II (NSGA-II) [40], are adopted as the comparison algorithms to verify the effectiveness of KEDA in solving the energy efficient JRALB–FP. For

Table 9
Comparison results of C metric between KEDA and A3.

(N_T, N_W)	(20, 4)			(50, 8)			(70, 8)			(83, 8)			(100, 10)			(148, 10)			(220, 10)			(297, 10)		
No.	$C_{k,3}$	$C_{3,k}$	Sig	$C_{k,3}$	$C_{3,k}$	Sig	$C_{k,3}$	$C_{3,k}$	Sig	$C_{k,3}$	$C_{3,k}$	Sig	$C_{k,3}$	$C_{3,k}$	Sig	$C_{k,3}$	$C_{3,k}$	Sig	$C_{k,3}$	$C_{3,k}$	Sig	$C_{k,3}$	$C_{3,k}$	Sig
1	0.40	0.33	N	0.40	0.00	Y	0.18	0.00	Y	0.35	0.00	Y	0.28	0.00	Y	0.25	0.09	Y	0.33	0.00	Y	0.13	0.00	Y
2	0.83	0.33	Y	0.56	0.00	Y	0.27	0.00	Y	0.33	0.00	Y	0.27	0.00	Y	0.27	0.00	Y	0.22	0.00	Y	0.27	0.00	Y
3	0.25	0.67	N	0.40	0.00	Y	0.30	0.00	Y	0.41	0.00	Y	0.29	0.00	Y	0.17	0.00	Y	0.20	0.00	Y	0.30	0.00	Y
4	0.00	0.50	N	0.42	0.00	Y	0.29	0.00	Y	0.35	0.00	Y	0.25	0.00	Y	0.28	0.00	Y	0.12	0.00	Y	0.13	0.00	Y
5	0.67	0.00	Y	0.30	0.00	Y	0.46	0.00	Y	0.25	0.00	Y	0.47	0.00	Y	0.24	0.00	Y	0.29	0.00	Y	0.25	0.00	Y
6	0.33	0.00	Y	0.50	0.00	Y	0.38	0.00	Y	0.33	0.00	Y	0.29	0.00	Y	0.26	0.00	Y	0.08	0.00	Y	0.00	0.00	N
7	0.57	0.25	Y	0.40	0.00	Y	0.17	0.00	Y	0.23	0.00	Y	0.20	0.00	Y	0.25	0.00	Y	0.19	0.00	Y	0.13	0.00	Y
8	0.40	0.50	N	0.33	0.13	Y	0.40	0.00	Y	0.31	0.00	Y	0.30	0.00	Y	0.21	0.00	Y	0.21	0.00	Y	0.30	0.00	Y
9	0.80	0.00	Y	0.47	0.00	Y	0.23	0.00	Y	0.20	0.00	Y	0.25	0.00	Y	0.30	0.00	Y	0.23	0.00	Y	0.08	0.00	Y
10	0.80	0.00	Y	0.43	0.00	Y	0.53	0.00	Y	0.42	0.00	Y	0.40	0.00	Y	0.07	0.00	Y	0.14	0.00	Y	0.00	0.00	N
(N_T, N_W)	(20, 5)			(50, 10)			(70, 10)			(83, 10)			(100, 12)			(148, 12)			(220, 15)			(297, 15)		
No.	$C_{k,3}$	$C_{3,k}$	Sig	$C_{k,3}$	$C_{3,k}$	Sig	$C_{k,3}$	$C_{3,k}$	Sig	$C_{k,3}$	$C_{3,k}$	Sig	$C_{k,3}$	$C_{3,k}$	Sig	$C_{k,3}$	$C_{3,k}$	Sig	$C_{k,3}$	$C_{3,k}$	Sig	$C_{k,3}$	$C_{3,k}$	Sig
1	0.29	0.00	Y	0.45	0.00	Y	0.31	0.00	Y	0.47	0.00	Y	0.25	0.00	Y	0.17	0.00	Y	0.13	0.00	Y	0.13	0.00	Y
2	0.40	0.25	Y	0.42	0.00	Y	0.40	0.00	Y	0.36	0.00	Y	0.27	0.00	Y	0.31	0.00	Y	0.29	0.00	Y	0.47	0.00	Y
3	0.75	0.00	Y	0.56	0.00	Y	0.40	0.00	Y	0.45	0.00	Y	0.31	0.00	Y	0.28	0.00	Y	0.33	0.00	Y	0.23	0.00	Y
4	0.57	0.25	Y	0.40	0.00	Y	0.50	0.00	Y	0.33	0.00	Y	0.40	0.00	Y	0.08	0.00	N	0.15	0.00	Y	0.25	0.00	Y
5	0.71	0.00	Y	0.55	0.00	Y	0.39	0.00	Y	0.25	0.00	Y	0.21	0.00	Y	0.25	0.00	Y	0.06	0.00	N	0.11	0.00	Y
6	0.57	0.20	Y	0.50	0.00	Y	0.26	0.00	Y	0.29	0.00	Y	0.28	0.00	Y	0.36	0.00	Y	0.11	0.00	Y	0.25	0.00	Y
7	0.63	0.25	Y	0.64	0.00	Y	0.42	0.00	Y	0.21	0.00	Y	0.30	0.00	Y	0.31	0.00	Y	0.19	0.00	Y	0.19	0.00	Y
8	0.20	0.50	N	0.40	0.00	Y	0.47	0.00	Y	0.33	0.00	Y	0.12	0.00	Y	0.29	0.00	Y	0.18	0.00	Y	0.11	0.00	N
9	0.50	0.00	Y	0.30	0.00	Y	0.53	0.00	Y	0.33	0.00	Y	0.17	0.00	Y	0.23	0.00	Y	0.14	0.00	Y	0.20	0.00	Y
10	0.67	0.00	Y	0.46	0.00	Y	0.25	0.00	Y	0.23	0.00	Y	0.40	0.00	Y	0.25	0.00	Y	0.41	0.00	Y	0.25	0.00	Y

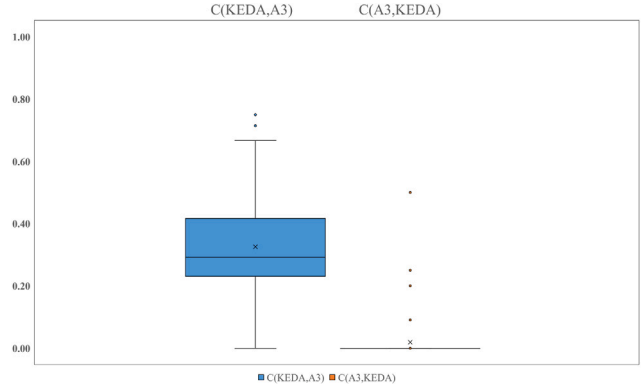


Fig. 10. Boxplot of C metric between KEDA and A3.

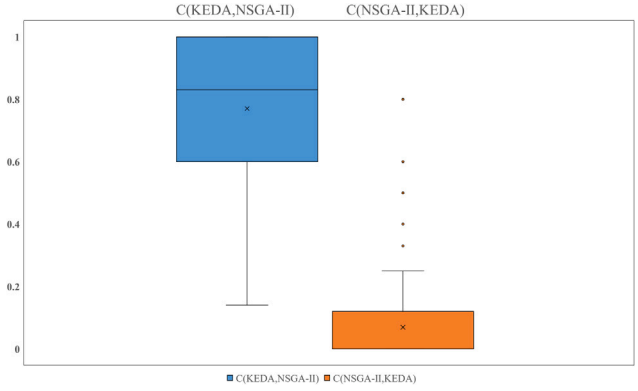


Fig. 11. Boxplot of C metric for KEDA and NSGA-II.

each case, the average C metric of ten independent runs is summarized in Tables 10 and 11. The boxplots of the comparison between all C metrics of KEDA and the comparison algorithm are shown in Figs. 11 and 12. The T-hypothesis test with a 95% confidence level is carried out, setting alternative hypothesis H1 such that the C metric achieved by KEDA is greater than the C metric achieved by the comparison algorithms. If H1 is implemented significantly ($p < 0.05$), then the flag is labeled as Y, otherwise it is labeled as N. C metric C(KEDA,NAGA-II) and C(NAGA-II,KEDA) are denoted as $C_{k,n}$ and $C_{n,k}$ respectively for simplification in the tables. C(KEDA, MOEA/D) and C(MOEA/D, KEDA) are denoted as $C_{k,m}$ and $C_{m,k}$.

In addition, the IGD of the proposed algorithm and two comparative algorithms, and presented them in the Table 12. The true Pareto front is composed of the common Pareto front obtained by all algorithms. KEDA, NSGA-II and MOEA/D are denoted as Ak, An and Am respectively for simplification. The Pareto fronts obtained by KEDA, NSGA-II, and MOEA/D are illustrated in Figs. 13–16.

It can be seen from the results that C(KEDA, NSGA-II) are larger than C(NSGA-II, KEDA) in most cases, which implies that KEDA performs better than NSGA-II in solving this problem. The average value of C(KEDA, NSGA-II) is 0.770 and the average value of C(NSGA-II, KEDA) is 0.066, which implies that about 77.0% NSGA-II solutions are dominated by the KEDA solutions, while about 6.6% KEDA solutions are dominated by NSGA-II solutions.

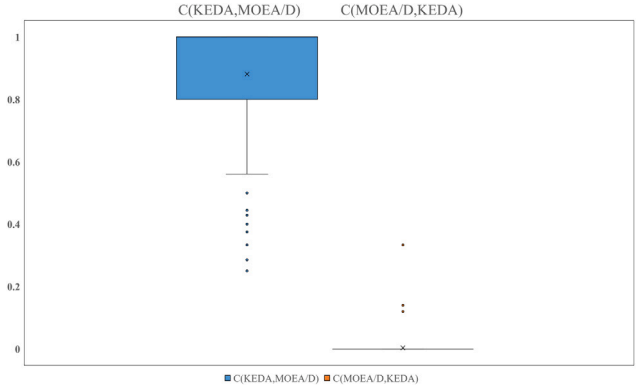


Fig. 12. Boxplot of C metric for KEDA and MOEA/D.

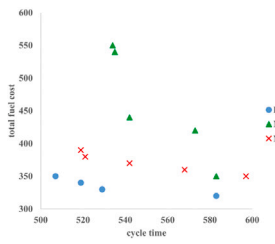
Similarly, it can be seen from the results that C(KEDA, MOEA/D) are larger than C(MOEA/D), KEDA) in most cases, which implies that KEDA performs better than MOEA/D in solving this problem. The average value of C(KEDA, NSGA-II) is 0.881 and the average value of C(NSGA-II, KEDA) is 0.002. About 88.1% MOEA/D solutions are dominated by the KEDA solutions, while about 0.2% KEDA solutions are dominated

Table 10
Comparison results of C metric between KEDA and NSGA-II.

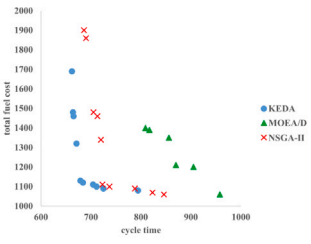
(N_T, N_W)	(20, 4)			(50, 8)			(70, 8)			(83, 8)			(100, 10)			(148, 10)			(220, 10)			(297, 10)		
No.	$C_{k,n}$	$C_{n,k}$	Sig	$C_{k,n}$	$C_{n,k}$	Sig	$C_{k,n}$	$C_{n,k}$	Sig	$C_{k,n}$	$C_{n,k}$	Sig	$C_{k,n}$	$C_{n,k}$	Sig	$C_{k,n}$	$C_{n,k}$	Sig	$C_{k,n}$	$C_{n,k}$	Sig	$C_{k,n}$	$C_{n,k}$	Sig
1	1.00	0.00	Y	1.00	0.00	Y	0.92	0.00	Y	0.78	0.00	Y	0.45	0.20	Y	0.90	0.00	Y	0.89	0.00	Y	0.43	0.33	N
2	0.75	0.17	Y	0.90	0.00	Y	1.00	0.00	Y	0.94	0.00	Y	0.83	0.00	Y	0.89	0.00	Y	1.00	0.00	Y	0.50	0.33	Y
3	0.90	0.00	Y	0.94	0.00	Y	0.83	0.00	Y	0.94	0.20	Y	0.43	0.00	Y	0.89	0.00	Y	1.00	0.00	Y	0.67	0.00	Y
4	0.71	0.00	Y	0.88	0.14	Y	0.94	0.00	Y	1.00	0.00	Y	0.43	0.12	Y	0.83	0.00	Y	0.63	0.25	Y	0.60	0.00	Y
5	0.83	0.00	Y	1.00	0.00	Y	0.78	0.00	Y	0.89	0.00	Y	0.62	0.00	Y	0.90	0.00	Y	0.73	0.00	Y	0.43	0.00	Y
6	0.71	0.00	Y	0.94	0.00	Y	0.93	0.17	Y	0.80	0.25	Y	0.60	0.00	Y	0.91	0.00	Y	1.00	0.00	Y	0.40	0.00	Y
7	0.63	0.50	N	1.00	0.00	Y	0.90	0.20	Y	1.00	0.00	Y	0.29	0.14	Y	0.73	0.00	Y	0.50	0.00	Y	0.56	0.00	Y
8	0.88	0.13	Y	0.94	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	0.75	0.16	Y	0.92	0.00	Y	0.90	0.00	Y	0.50	0.33	N
9	0.50	0.25	Y	1.00	0.00	Y	0.88	0.00	Y	0.82	0.00	Y	0.43	0.25	Y	1.00	0.00	Y	0.88	0.25	Y	0.20	0.33	N
10	0.75	0.13	Y	1.00	0.00	Y	0.95	0.07	Y	1.00	0.00	Y	0.43	0.60	N	0.44	0.33	N	0.75	0.00	Y	0.40	0.80	N
(N_T, N_W)	(20, 5)			(50, 10)			(70, 10)			(83, 10)			(100, 12)			(148, 12)			(220, 15)			(297, 15)		
No.	$C_{k,n}$	$C_{n,k}$	Sig	$C_{k,n}$	$C_{n,k}$	Sig	$C_{k,n}$	$C_{n,k}$	Sig	$C_{k,n}$	$C_{n,k}$	Sig	$C_{k,n}$	$C_{n,k}$	Sig	$C_{k,n}$	$C_{n,k}$	Sig	$C_{k,n}$	$C_{n,k}$	Sig	$C_{k,n}$	$C_{n,k}$	Sig
1	0.86	0.00	Y	0.44	0.33	Y	0.46	0.20	Y	0.55	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	0.66	0.40	N
2	1.00	0.00	Y	0.56	0.09	Y	0.75	0.00	Y	0.43	0.16	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	0.82	0.00	Y
3	1.00	0.00	Y	0.67	0.33	N	0.40	0.00	Y	0.44	0.00	Y	0.90	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	0.80	0.00	Y
4	1.00	0.00	Y	0.25	0.20	Y	0.75	0.00	Y	0.43	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	0.96	0.00	Y	0.64	0.00	Y
5	1.00	0.00	Y	0.47	0.25	Y	0.56	0.12	Y	0.36	0.00	Y	0.79	0.19	Y	1.00	0.00	Y	1.00	0.00	Y	0.82	0.00	Y
6	1.00	0.00	Y	0.63	0.00	Y	0.63	0.16	N	0.63	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	0.75	0.00	Y
7	1.00	0.00	Y	0.20	0.33	N	0.53	0.12	Y	0.43	0.33	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	0.81	0.00	Y
8	1.00	0.00	Y	0.38	0.33	Y	0.60	0.00	Y	0.67	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	0.78	0.00	Y
9	0.80	0.00	Y	0.29	0.25	Y	0.64	0.25	Y	0.14	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	0.71	0.00	Y
10	0.75	0.00	Y	0.50	0.20	Y	0.30	0.00	Y	0.50	0.00	Y	1.00	0.00	Y	0.94	0.00	Y	1.00	0.00	Y	0.69	0.25	Y

Table 11
Comparison results of C metric between KEDA and MOEA/D.

(N_T, N_W)	(20, 4)			(50, 8)			(70, 8)			(83, 8)			(100, 10)			(148, 10)			(220, 10)			(297, 10)		
No.	$C_{k,m}$	$C_{m,k}$	Sig	$C_{k,m}$	$C_{m,k}$	Sig	$C_{k,m}$	$C_{m,k}$	Sig	$C_{k,m}$	$C_{m,k}$	Sig	$C_{k,m}$	$C_{m,k}$	Sig	$C_{k,m}$	$C_{m,k}$	Sig	$C_{k,m}$	$C_{m,k}$	Sig	$C_{k,m}$	$C_{m,k}$	Sig
1	0.25	0.33	N	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	0.86	0.00	Y	0.90	0.00	Y	1.00	0.00	Y	0.57	0.00	Y
2	0.75	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	0.75	0.00	Y	0.93	0.00	Y	0.57	0.00	Y
3	0.33	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	0.50	0.14	Y	0.56	0.00	Y	0.78	0.00	Y	1.00	0.00	Y
4	0.43	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	0.89	0.00	Y	0.57	0.00	Y	0.91	0.00	Y	0.50	0.12	Y
5	0.67	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	0.56	0.00	Y	0.78	0.00	Y	0.88	0.00	Y
6	0.38	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	0.89	0.00	Y	0.75	0.00	Y	0.90	0.00	Y
7	0.40	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	0.86	0.00	Y	0.89	0.00	Y	0.73	0.00	Y	1.00	0.00	Y
8	0.44	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	0.88	0.00	Y	0.75	0.00	Y	0.89	0.00	Y	0.82	0.00	Y
9	0.29	0.12	N	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	0.88	0.00	Y	0.78	0.00	Y	0.78	0.00	Y	0.83	0.00	Y
10	0.44	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	0.93	0.00	Y	0.80	0.00	Y	0.69	0.00	Y	0.60	0.00	Y
(N_T, N_W)	(20, 5)			(50, 10)			(70, 10)			(83, 10)			(100, 12)			(148, 12)			(220, 15)			(297, 15)		
No.	$C_{k,m}$	$C_{m,k}$	Sig	$C_{k,m}$	$C_{m,k}$	Sig	$C_{k,m}$	$C_{m,k}$	Sig	$C_{k,m}$	$C_{m,k}$	Sig	$C_{k,m}$	$C_{m,k}$	Sig	$C_{k,m}$	$C_{m,k}$	Sig	$C_{k,m}$	$C_{m,k}$	Sig	$C_{k,m}$	$C_{m,k}$	Sig
1	0.75	0.00	Y	0.86	0.00	Y	0.80	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y
2	0.91	0.00	Y	0.89	0.00	Y	0.89	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y
3	1.00	0.00	Y	1.00	0.00	Y	0.82	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y
4	0.88	0.00	Y	1.00	0.00	Y	0.92	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y
5	0.71	0.00	Y	0.80	0.00	Y	0.83	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y
6	0.83	0.00	Y	1.00	0.00	Y	0.67	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y
7	0.91	0.00	Y	0.75	0.00	Y	0.56	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y
8	1.00	0.00	Y	0.89	0.00	Y	0.75	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y
9	0.80	0.00	Y	0.64	0.00	Y	0.63	0.00	Y	0.60	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y
10	0.80	0.00	Y	0.79	0.00	Y	0.75	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y	1.00	0.00	Y

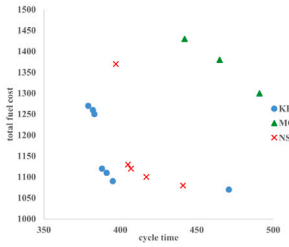


(a)$N_T = 20$

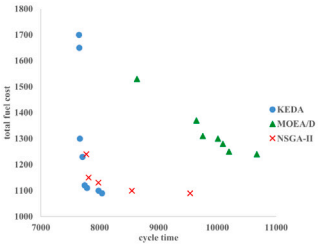


(b)$N_T = 50$

Fig. 13. Pareto fronts obtained by KEDA and the comparison algorithms for a task size of 20/50.



(a)$N_T = 70$

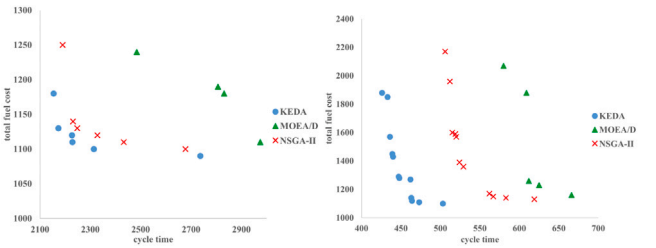


(b)$N_T = 83$

Fig. 14. Pareto fronts obtained by KEDA and the comparison algorithms for a task size of 70/83.

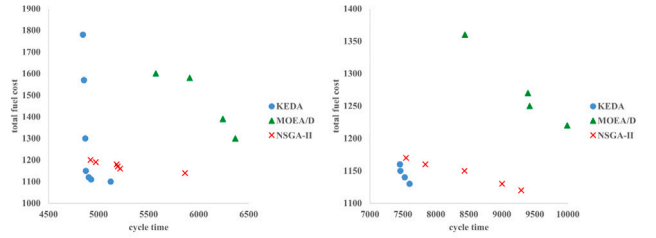
Table 12
Comparison Results of IGD between KEDA and the comparison algorithms.

(N_T, N_W)	(20, 4)			(50, 8)			(70, 8)			(83, 8)			(100, 10)			(148, 10)			(220, 10)			(297, 10)		
No.	*Ak*	*An*	*Am*	*Ak*	*An*	*Am*	*Ak*	*An*	*Am*	*Ak*	*An*	*Am*	*Ak*	*An*	*Am*	*Ak*	*An*	*Am*	*Ak*	*An*	*Am*	*Ak*	*An*	*Am*
1	31.95	72.64	37.70	2.78	216.97	328.92	0.00	384.98	483.85	0.00	494.03	781.94	154.65	97.13	478.87	2.44	118.05	127.34	0.00	1371.37	1490.70	37.84	417.78	1488.80
2	10.67	231.71	371.76	9.60	218.26	360.27	62.50	477.80	405.23	5.50	542.99	524.34	128.58	44.99	1075.92	24.04	90.97	119.66	20.48	833.71	1245.81	96.57	2750.56	1261.09
3	0.00	163.68	100.37	3.20	151.28	510.89	0.00	269.82	312.12	77.15	643.89	930.97	30.74	66.49	576.48	6.00	89.01	69.18	85.61	952.02	1586.17	0.60	1721.87	2555.79
4	2.00	57.41	112.28	45.54	74.71	460.93	9.31	425.08	498.61	0.00	521.48	812.55	9.88	27.29	569.69	32.74	227.00	143.86	28.53	806.53	1238.09	42.00	2633.91	650.08
5	0.00	59.36	136.25	54.33	128.38	479.89	0.00	219.39	599.94	0.00	361.39	721.47	8.33	43.34	209.88	5.58	99.48	99.53	16.67	1619.69	1153.67	0.00	1392.96	1876.22
6	0.00	52.54	32.89	0.86	258.48	310.65	30.02	95.26	497.36	0.00	421.65	636.40	13.69	33.52	453.71	2.25	124.00	125.04	130.80	848.68	1073.83	0.00	504.09	1024.31
7	16.67	298.19	331.61	18.37	186.64	220.23	0.00	289.71	361.45	11.14	351.38	804.90	50.60	8.06	414.90	1.35	143.32	128.78	0.00	176.25	1019.23	0.00	927.00	1720.38
8	12.54	280.96	157.37	9.25	264.34	405.56	0.00	224.67	336.52	12.75	239.20	795.08	39.41	65.42	449.94	0.33	159.34	157.90	167.23	947.51	1366.30	136.78	557.62	975.72
9	0.00	62.26	86.27	5.40	255.67	530.53	0.00	282.41	305.96	0.00	202.90	775.21	68.25	24.08	386.90	7.90	109.80	134.19	0.00	322.09	781.05	251.36	509.31	1450.50
10	10.00	17.61	161.59	5.50	176.30	431.40	24.97	292.85	239.94	95.20	374.70	732.92	183.16	145.70	784.38	93.35	11.53	191.47	2.25	670.95	642.32	219.56	824.30	1578.15
(N_T, N_W)	(20, 5)			(50, 10)			(70, 10)			(83, 10)			(100, 12)			(148, 12)			(220, 15)			(297, 15)		
No.	*Ak*	*An*	*Am*	*Ak*	*An*	*Am*	*Ak*	*An*	*Am*	*Ak*	*An*	*Am*	*Ak*	*An*	*Am*	*Ak*	*An*	*Am*	*Ak*	*An*	*Am*	*Ak*	*An*	*Am*
1	0.00	79.06	46.97	136.98	27.83	226.87	76.82	26.68	138.16	85.00	303.65	719.58	53.47	974.15	2265.40	37.08	308.53	2017.95	47.79	2966.20	5577.54	234.90	1891.29	8448.54
2	0.00	148.52	122.15	76.22	216.34	138.37	0.86	53.60	247.41	95.83	208.24	857.94	4.00	1689.73	4611.52	34.91	195.61	2214.24	7.42	2171.19	5003.86	0.00	3743.03	5723.99
3	6.00	50.37	263.39	83.41	25.35	673.77	20.02	20.33	117.94	0.00	144.92	985.56	17.88	861.67	3476.53	48.26	296.42	2174.79	100.27	1794.42	5512.02	0.00	2196.61	7572.08
4	0.75	125.27	128.05	40.75	91.92	184.42	4.10	72.45	197.09	42.05	87.40	880.44	4.65	236.51	3980.85	60.43	339.24	1851.86	5.80	1990.91	6158.15	0.00	1460.96	8816.67
5	2.67	53.77	66.08	38.42	214.21	637.62	5.88	49.66	90.49	7.67	56.46	612.35	41.56	476.28	3236.78	9.15	254.69	1824.61	32.86	2722.51	6072.32	0.00	3069.68	6293.94
6	8.80	80.01	114.79	4.13	202.50	245.14	3.33	82.05	70.85	0.00	878.14	1838.16	102.69	1378.72	4825.61	23.88	366.38	2121.24	4.60	2569.23	7424.13	0.00	1470.21	6697.42
7	19.75	93.59	100.00	7.37	27.81	200.87	19.25	20.58	68.20	22.67	25.87	977.14	0.33	1085.21	4659.14	10.11	240.91	2701.26	6.32	1090.25	5575.03	0.00	7124.40	7968.74
8	10.67	67.81	144.06	11.42	15.32	395.83	0.00	81.79	193.55	39.67	99.75	333.11	50.88	849.52	4919.11	21.86	137.78	2050.77	0.80	1443.92	5413.29	35.50	3138.27	8167.18
9	0.00	40.75	73.93	7.56	3.18	446.72	6.00	71.24	92.72	0.00	1502.17	1770.31	2.33	1855.09	4254.48	53.52	131.40	2363.44	0.00	630.30	7565.50	0.00	4485.14	8700.56
10	2.55	61.55	71.83	26.33	92.72	683.34	0.00	6.72	140.43	5.75	255.54	887.93	0.00	545.89	3404.14	6.46	326.45	2390.74	4.61	1753.14	6737.65	76.97	711.63	6312.58



(a)$N_T = 100$ (b)$N_T = 148$

Fig. 15. Pareto fronts obtained by KEDA and the comparison algorithms for a task size of 100/148.



(a)$N_T = 220$ (b)$N_T = 297$

Fig. 16. Pareto fronts obtained by KEDA and the comparison algorithms for a task size of 220/297.

by MOEA/D solutions. It can be observed that the performance of KEDA improves as the scale of workstation increases. The possible reason is that our proposed KEDA is more likely to identify better solutions within a larger solution space.

In addition, the Pareto fronts obtained by KEDA under different scales of instances are better than the fronts obtained by the other two algorithms. The IGD results show that the KEDA solutions are much nearer to the Pareto fronts achieved by all the algorithms. Specially, IGD = 0 denotes that all the KEDA solutions denominate other solutions and none of them are dominated by others.

5. Conclusion

This paper addresses the energy-efficient JRALB–FP with the minimization of both cycle time and fuel consumption cost. An knowledge-guided EDA is presented to solve the energy-efficient JRALB–FP. From numerical experimental results, it shows that our proposed algorithm performs better than the existing algorithms in terms of both time and energy consumption index. The superior performances of the algorithm mainly owe to the following aspects.

(1) The hybrid individual generation method, which combines both probability sampling and the knowledge-based heuristic method.

(2) The cooperation of search operators and probability model updating.

(3) Two problem knowledge guided local search operators, which enhances exploitation capability.

The future work will involve studying various other types of assembly line balancing and part feeding problems, including the mix-model assembly, other feeding strategies, and other objectives, such as addressing the assembly imbalance between different workstations. In-time assembly task assignment will be also considered through learning-enhanced methods [41]. We will also focus on the research about the analysis of problem and the utilization of the problem-specific knowledge to improve the effectiveness of the proposed algorithm. Moreover, it is also very important to apply the algorithms to real production cases.

CRediT authorship contribution statement

Chu-ge Wu: Validation, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Conceptualization, Writing – original draft, Writing – review & editing. **Ruochen Zhang:** Writing – review & editing, Data curation, Investigation, Methodology, Writing – original draft. **Yuanqing Xia:** Validation, Project administration.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work was funded in part by the National Key R&D Program of China (No. 2021YFB3300400) and the National Natural Science Foundation of China (No. 62203049).

References

[1] N. Boysen, P. Schulze, A. Scholl, Assembly line balancing: What happened in the last fifteen years? European J. Oper. Res. 301 (3) (2022) 797–814, http://dx.doi.org/10.1016/j.ejor.2021.11.043.

[2] J. Sternatz, Enhanced multi-Hoffmann heuristic for efficiently solving real-world assembly line balancing problems in automotive industry, European J. Oper. Res. 235 (3) (2014) 740–754, http://dx.doi.org/10.1016/j.ejor.2013.11.005.

[3] Z. Fang, J. Wang, Y. Ren, Z. Han, H.V. Poor, L. Hanzo, Age of information in energy harvesting aided massive multiple access networks, IEEE J. Sel. Areas Commun. 40 (5) (2022) 1441–1456.

[4] H. Feng, J. Wang, Z. Fang, J. Chen, D.-T. Do, Evaluating AoI-Centric HARQ protocols for UAV networks, IEEE Trans. Commun. (2023).

[5] United States Environmental Protection Agency, Sources of greenhouse gas emissions, 2017, https://www.epa.gov/ghgemissions/sources-greenhouse-gas-emissions.

[6] N. Boysen, M. Fliedner, A. Scholl, Assembly line balancing: Which model to use when? Int. J. Prod. Econ. 111 (2) (2008) 509–528.

[7] N. Boysen, M. Fliedner, A. Scholl, A classification of assembly line balancing problems, European J. Oper. Res. 183 (2) (2007) 674–693, http://dx.doi.org/10.1016/j.ejor.2006.10.010.

[8] Y.-l. Jiao, H.-q. Jin, X.-c. Xing, M.-j. Li, X.-r. Liu, Assembly line balance research methods, literature and development review, Concurr. Eng. 29 (2) (2021) 183–194, http://dx.doi.org/10.1177/1063293X20987910.

[9] B. qi Sun, L. Wang, Z. ping Peng, Bound-guided hybrid estimation of distribution algorithm for energy-efficient robotic assembly line balancing, Comput. Ind. Eng. 146 (2020) 106604, http://dx.doi.org/10.1016/j.cie.2020.106604.

[10] L.W. Binqi Sun, An estimation of distribution algorithm with branch-and-bound based knowledge for robotic assembly line balancing, Complex Intell. Syst. 7 (2021) 1125–1138.

[11] Z. Li, M.N. Janardhanan, Q. Tang, S.G. Ponnambalam, Model and metaheuristics for robotic two-sided assembly line balancing problems with setup times, Swarm Evol. Comput. 50 (2019) 100567.

[12] H.A. Sedding, Line side placement for shorter assembly line worker paths, IISE Trans. 52 (2) (2020) 181–198, http://dx.doi.org/10.1080/24725854.2018.1508929, arXiv:https://doi.org/10.1080/24725854.2018.1508929.

[13] D. Battini, M. Faccio, A. Persona, F. Sgarbossa, Design of the optimal feeding policy in an assembly system, Int. J. Prod. Econ. 121 (1) (2009) 233–254, http://dx.doi.org/10.1016/j.ijpe.2009.05.016, Modelling and Control of Productive Systems: Concepts and Applications.

[14] M. Calzavara, S. Finco, D. Battini, F. Sgarbossa, A. Persona, A joint assembly line balancing and feeding problem (JALBFP) considering direct and indirect supply strategies, Int. J. Prod. Res. 60 (19) (2022) 5727–5745, http://dx.doi.org/10.1080/00207543.2021.1968527.

[15] M. Albus, M.F. Huber, Resource reconfiguration and optimization in brownfield constrained robotic assembly line balancing problems, J. Manuf. Syst. (2023) 132–142.

[16] C. Weckenborg, K. Kieckhäfer, C. Müller, M. Grunewald, T.S. Spengler, Balancing of assembly lines with collaborative robots, Bus. Res. (2020) 93–132.

[17] A. Nourmohammadi, M. Fathi, A.H.C. Ng, E. Mahmoodi, A genetic algorithm for heterogenous human-robot collaboration assembly line balancing problems, Procedia CIRP (2022) 1444–1448.
[18] F. Haotian, W. Hongjun, Research on robot assembly line balancing considering energy consumption, Mech. Mach. Sci. (2021) 869–881.
[19] A. Nourmohammadi, M. Fathi, A.H.C. Ng, Balancing and scheduling assembly lines with human–robot collaboration tasks, Comput. Oper. Res. (2022) 105674.
[20] Z. Mao, J. Zhang, K. Fang, D. Huang, Y. Sun, Balancing U-type assembly lines with human–robot collaboration, Comput. Oper. Res. (2023) 106359.
[21] Y. Lahrichi, D. Damand, L. Deroussi, N. Grangeon, S. Norre, Investigating two variants of the sequence-dependent robotic assembly line balancing problem by means of a split-based approach, Int. J. Prod. Res. (2023) 2322–2338.
[22] N.A. Schmid, V. Limère, A classification of tactical assembly line feeding problems, Int. J. Prod. Res. 57 (24) (2019) 7586–7609, http://dx.doi.org/10.1080/00207543.2019.1581957.
[23] J. Sternatz, The joint line balancing and material supply problem, Int. J. Prod. Econ. 159 (2015) 304–318, http://dx.doi.org/10.1016/j.ijpe.2014.07.022.
[24] B.-H. Zhou, C.-Y. Shen, Multi-objective optimization of material delivery for mixed model assembly lines with energy consideration, J. Clean. Prod. 192 (2018) 293–305.
[25] Ö.F. Yılmaz, An integrated bi-objective U-shaped assembly line balancing and parts feeding problem: Optimization model and exact solution method, Ann. Math. Artif. Intell. 90 (7–9) (2022) 679–696, http://dx.doi.org/10.1007/s10472-020-09718-y.
[26] H. Zhang, X. Li, Z. Kan, X. Zhang, Z. Li, Research on optimization of assembly line based on product scheduling and just-in-time feeding of parts, Assem. Autom. 41 (5) (2021) 577–588, http://dx.doi.org/10.1108/AA-12-2020-0196.
[27] J. Chen, X. Jia, Energy-efficient integration of assembly line balancing and part feeding with a modified genetic algorithm, Int. J. Adv. Manuf. Technol. 121 (3–4) (2022) 2257–2278.
[28] B. Zhou, Z. Zhu, Multi-objective optimization of greening scheduling problems of part feeding for mixed model assembly lines based on the robotic mobile fulfillment system, Neural Comput. Appl. 33 (2021) 9913–9937.
[29] P. Larranaga, Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation (Genetic Algorithms and Evolutionary Computation), Kluwer Academic, 2002.
[30] N. Zhu, F. Zhao, L. Wang, C. Dong, An enhanced Kalman filtering and historical learning mechanism driven estimation of distribution algorithm, Swarm Evol. Comput. 86 (2024) 101502.
[31] Y. Du, J. qing Li, C. Luo, L. lei Meng, A hybrid estimation of distribution algorithm for distributed flexible job shop scheduling with crane transportations, Swarm Evol. Comput. 62 (2021) 100861.
[32] F. Zhao, B. Zhu, L. Wang, An estimation of distribution algorithm-based hyper-heuristic for the distributed assembly mixed no-idle permutation flowshop scheduling problem, IEEE Trans. Syst. Man Cybern. Syst. (2023) 1–12.
[33] C.G. Wu, L. Wang, J.J. Wang, A path relinking enhanced estimation of distribution algorithm for direct acyclic graph task scheduling problem, Knowl.-Based Syst. (2021) 107255.
[34] Z.-Q. Zhang, B. Qian, R. Hu, H.-P. Jin, L. Wang, A matrix-cube-based estimation of distribution algorithm for the distributed assembly permutation flow-shop scheduling problem, Swarm Evol. Comput. 60 (2021) 100785.
[35] M.R. Sierra, C.A. Coello Coello, Improving PSO-based multi-objective optimization using crowding, mutation and ϵ-dominance, Lecture Notes in Comput. Sci. 3410 (2005) 505–519, http://dx.doi.org/10.1007/978-3-540-31880-4_35.
[36] J. Mukund Nilakantan, S.G. Ponnambalam, N. Jawahar, G. Kanagaraj, Bio-inspired search algorithms to solve robotic assembly line balancing problems, Neural Comput. Appl. 26 (2015) 1379–1393.
[37] A. Lipowski, D. Lipowska, Roulette-wheel selection via stochastic acceptance, Phys. A 391 (6) (2012) 2193–2196.
[38] S. Baluja, Population-Based Incremental Learning. A Method for Integrating Genetic Search Based Function Optimization and Competitive Learning, Technical Report, Carnegie Mellon University, 1994.
[39] Q. Zhang, H. Li, MOEA/D: a multiobjective evolutionary algorithm based on decomposition, IEEE Trans. Evol. Comput. 11 (6) (2007) 712–731, http://dx.doi.org/10.1109/TEVC.2007.892759.
[40] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Trans. Evol. Comput. 6 (2) (2002) 182–197, http://dx.doi.org/10.1109/4235.996017.
[41] J.-F. Chen, L. Wang, H. Ren, J. Pan, S. Wang, J. Zheng, X. Wang, An imitation learning-enhanced iterated matching algorithm for on-demand food delivery, IEEE Transactions on Intelligent Transportation Systems 23 (10) (2022) 18603–18619.