A Two-Phase PCBA Optimization with ILP Model and Heuristic for a Beam Head Placement Machine

Guangyu Lu, *Graduate Student Member, IEEE*, Zhengkai Li, *Member, IEEE*, Hao Sun, *Member, IEEE*, Xinghu Yu, *Member, IEEE*, Jiahu Qin, Jianbin Qiu, *Fellow, IEEE*, and Huijun Gao, *Fellow, IEEE*

***Abstract*— The optimization of printed circuit board assembly (PCBA) for a beam head placement machine is a multivariable and multiconstraint combinatorial problem. Current techniques falter in solving a variety of PCBA problems since heuristic algorithms lack theoretical guarantees of optimality, and mathematical modeling methods have high computational complexity for the whole problem. This article proposes a novel two-phase optimization for PCBA, integrating the advantages of mathematical modeling with heuristic algorithms. We divide the problem into the head task assignment and the placement route schedule. For the former, an effective integer linear programming (ILP) model with component partition is proposed, encompassing key efficiency-influencing factors. A recursive heuristic-based initial solution speeds up the solving convergence, while the reduction strategies enhance model solvability. For the placement route schedule, a tailored greedy algorithm yields high-quality solutions, leveraging the results of the model, and an aggregated route relink heuristic (ARRH) does further optimization. Additionally, we propose selection criteria for the solution pool of the model to pre-evaluate the placement movement, which builds the connection between the two phases. Finally, we validate the performance of the two-phase optimization, which provides an average efficiency improvement of 8.06%∼24.32% compared to other mainstream research.**

***Index Terms*— Beam head placement machine, PCB assembly optimization, head task model, placement route schedule.**

Manuscript received X X, X; revised X X, X; accepted X X, X. Date of publication X X, X; date of current version X X, X. This work was supported in part by the National Natural Science Foundation of China under Grant U20A20188 and Grant 62203141, in part by the Major Scientific and Technological Research Project of Ningbo under Grant 2021Z040, and in part by the New Cornerstone Science Foundation through the XPLORER PRIZE. (Corresponding author: Huijun Gao.)

Guangyu Lu, Jianbin Qiu, and Huijun Gao are with the Research Institute of Intelligent Control and Systems, Harbin Institute of Technology, Harbin 150001, China (e-mail: 20b904007@stu.hit.edu.cn; jbqiu@hit.edu.cn; hjgao@hit.edu.cn).

Zhengkai Li is with the Department of Mathematics and Theories, Peng Cheng Laboratory, Shenzhen 518000, China (e-mail: lizhk@pcl.ac.cn).

Hao Sun is with the Bio-Computing Research Center, Harbin Institute of Technology, Shenzhen 518000, China, and also with the Shenzhen Key Laboratory of Visual Object Detection and Recognition, Shenzhen 518000, China (e-mail: sunhao2021@hit.edu.cn).

Xinghu Yu is with the College of Physics and Optoelectronic Engineering, Shenzhen University, Shenzhen 518060, China, and also with the Ningbo Institute of Intelligent Equipment Technology Co., Ltd., Ningbo 315201, China (e-mail: 17b304003@stu.hit.edu.cn).

Jiahu Qin is with the Department of Automation, University of Science and Technology of China, Hefei 230027, China, and also with the Institute of Artificial Intelligence, Hefei Comprehensive National Science Center, Hefei 230088, China (e-mail: jhqin@ustc.edu.cn). Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

I. Introduction

SURFACE mount technology is essential to the electronic manufacturing industry. The need for higher efficiency in production lines has become more acute in electronic industries with the expansion of the manufacturing sector. The placement machines utilized to execute automated component surface assembly operations are the most crucial equipment in integrated printed circuit board assembly (PCBA) lines [1]. Developing surface assembly equipment is a systematic project involving multiple subjects, including visual recognition and positioning, advanced motion control, scheduling techniques, etc. In this article, we study the scheduling optimization techniques of the PCBA process using mathematical programming and heuristic algorithms.

The mechanical design of the beam head placement machines comprises placement heads, feeders, nozzles, and other connected accessories. They collaborate in three steps of the assembly process: component pickup, inspection, and placement. The heads are equipped with appropriate nozzle types for various types of components and are designed for pickup and placement operations. The components are picked up from feeder slots by linear aligned heads simultaneously and placed in the predetermined PCB pads, which consist of a pick-and-place (PAP) cycle. When the nozzle on the head is incompatible with the component type picked up from the feeders, a nozzle change operation is done at the auto nozzle changer.

Early PCBA optimization research ~~focus~~focuses on modeling simple machine types, such as single-head sequential pick-and-place machines [2] and multi-heads for single component type placement machines [3]. The integrated model for PCBA optimization has ~~been~~characteristics that combine~~as a combination of~~ the models for several sub-problems. Studies in [4] formulate a model in which the multi-heads case solves component sequencing, feeder assignment, and nozzle assignment simultaneously. In contrast, studies in [2] solve the sub-problems of component sequencing and feeder arrangement as a hierarchical multi-objective optimization problem.

The high complexity of the problem makes decomposition modeling necessary. As an extension of [3] for the multi-heads and multi-component types of case, a two-stage mixed integer programming model is proposed in [5] to optimize the nozzle component assignment and assembly route schedule, respectively. In [6], the problem is decomposed into hierarchical mixed integer pickup and placement models. Studies in [7] present a problem decomposition approach for component machine allocation and PCB sequence problems, which are modeled separately. Moreover, a few of the studies model the sub-problems therein, such as the nozzle assignment model in [8], [9] and the feeder module change model in [10]. The edge-based and route-based models have been developed in [11] for placement route schedules, and the branch-and-price method with effective branch rules solves the latter.

A series of techniques are applied in the modeling process to enhance its solvability. Studies in [12] present a mathematical model based on pickup groups to reduce the scale of the model, whereas studies in [13] propose an aggregated integer programming based on batches of components. In [14], an augmented ε method is proposed to optimize multiple sub-objectives by the curve matching method.

The large space of the solutions leads to the design of improved heuristics [15], and mathematical models typically are combined with them for higher computing efficiency. Hybrid genetic [12], [16], [17], tabu search [3], [18], particle swarm [19], frog leaping [20], [21] and other intelligent optimization algorithms are integrated to the PCBA optimization. Additionally, multiobjective optimization is also integrated with intelligent optimization; for instance, studies in [14] present a multiobjective particle swarm optimization, and studies in [22] integrate intelligent optimization with curve matching techniques. A cluster-based heuristic is applied to group components based on their properties with single gantry [23] and dual gantry [24] placement machines to optimize the pick-and-place sequence.

In this article, the proposed two-phase optimization method combines integer linear programming (ILP) models and heuristic algorithms, and its framework is shown in Fig. 1. In the first phase, we extract the primary objectives of the ILP model for the head task assignment, which is related to the pickup route. ~~model, which guarantees high-quality solutions because the objective of the ILP model covers the major metrics that affecting assembly efficiency~~ A series of techniques are proposed to improve the efficiency of model solving. ~~To improve the quality of the overall solution, we further select the solutions of the model from the pool of the first phase. As there is insufficient information regarding the points and sequence of heads place, a pre-evaluation heuristic provides a selection criterion based on the estimated assembly path.~~ In the second phase, we solve the placement route schedule problem of the assembly process using heuristic methods. The combination of mathematical modeling and heuristics ensures the high-quality of the major sub-objectives while taking into account the overall solving efficiency of the algorithms.

The main contributions of this article are summarized as follows:

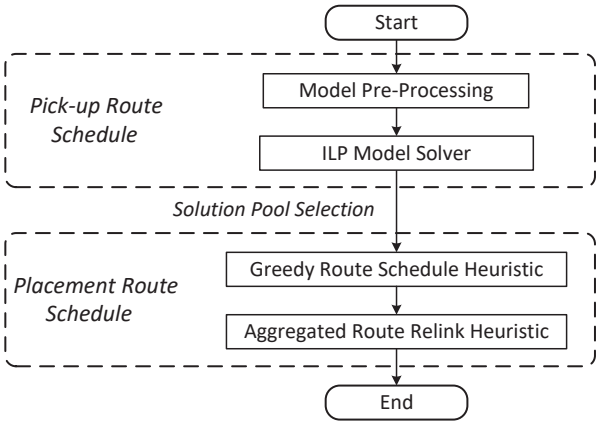

Fig. 1. The framework of two-phase optimization with ILP model and heuristic algorithms.

1) An effective integer linear model for the PCB assembly process is proposed to optimize the primary sub-objectives of the assembly process. The model pre-processing techniques are studied to improve the search efficiency.
2) A placement greedy route schedule the linearly aligned heads is proposed for with the constraint of the head task assignment, and the solution is further optimized by a route relink heuristic, which outperforms mainstream methods.
3) A pre-evaluation selection criterion is present for the one from the solution pool, which overcomes the drawbacks that modeling without movement terms may degrade the quality of the solution.

The rest of this article is organized as follows. In Sections II and III, respectively, each phase of the proposed framework is discussed. An ILP model based on the analysis results of the assembly process and its solving techniques is proposed in Section II. ~~Section II discusses the assembly problem, and presents an integer linear model with solving techniques.~~ The placement route schedule heuristics with determined greedy and random relink heuristic algorithms are present in Section III. In Section IV, we give the experimental comparative results ~~of the proposed two-phase optimization~~with a commercial optimizer Gurobi [25]. Section V concludes this article.

II. Head Task ~~ILP~~ Model Formulation

A. PCB Assembly Problem

The PCBA process comprises several aspects. The pick-and-place operations, nozzle change operations, and movements are the most critical aspects that affect its efficiency. The mechanism of beam heads is specially designed for simultaneous pickup operations to improve efficiency, whereas the placement operation time is determined by the PCB data. The heads can assemble different components by changing a compatible nozzle type, but it is time consuming and often discouraged. Beam head movements consist of pickup, placement, and round-trip movements between the feeder base and PCB. The number of PAP cycles affects the round-trip

movements, and the slots where pickup operations take place affects the pickup movements.

The nozzle types, component types, and pickup slots are the three basic compositions of the head task assignment. We call the consecutive PAP cycles with the same head task assignment as the cycle group. The objective of the model entails the primary sub-objectives, except for the movements of the gantry, which is optimized by the route schedule method. The PCBA process can be regarded as a capacitated vehicle route schedule problem [12], with restriction of a head-accessible point set, which proves it is nevertheless a NP-hard problem, and the extra constraints rather increase the difficulty of solving the problem.

The assumptions for the PCBA process are listed blowbelow.

- The compatibility between the nozzle and component types is predetermined.
- The assembly time of the different types of components is the same, and the capacity of the feeder base is much larger than the requirement.
- The interval between adjacent heads is an integer time of the interval between adjacent slots for simultaneous pickup.
- The time spent moving to the ANC for nozzle change is included in the nozzle change time, and the number of nozzle types is less than the number of heads.

B. Integer Linear Programming Model

An integer model for the head task assignment is derived based on [6], where the components are partitioned into different cycle groups. The notations of the integer model are summarized in Table I. The objective (1) of the model is the sum of the number of PAP cycles, nozzle changes, and pickup operations with different weights.

$$\min T_1 \cdot \sum_{l\in L} w_l + T_2 \cdot \sum_{h\in H}\sum_{l\in L} n_{lh} + T_3 \cdot \sum_{s\in S_e}\sum_{l\in L} w_l \cdot p_{sl} \tag{1}$$

The nonlinear term $w_l \cdot p_{sl}$ in the objective can be substituted by an intermediate variable λ_{sl}, which represents the number of pickups from slot s in cycle group l and can be linearized with big M method as

$$\begin{cases} \lambda_{sl} \le M \cdot p_{sl}, \\ \lambda_{sl} \le w_l, \\ \lambda_{sl} \ge w_l - M \cdot (1 - p_{sl}), \end{cases} \quad \forall s \in S_e, l \in L. \tag{2}$$

Constraint (3) ensures that the sum of placement points of component type i on all cycle groups equals the number of points on the PCB.

$$\sum_{h\in H}\sum_{l\in L} w_l \cdot u_{ihl} = \phi_i \quad \forall i \in I \tag{3}$$

The nonlinear term of constraint (3) can also be linearized, similar asto the linearization of the nonlinear term in the objective function.

Constraints (4)–(5) convert the pickup slot to the leftmost head-aligned one, so that the number of pickup operations in a cycle group can be computed directly.

$$p_{sl} \ge v_{[s+(h-1)\cdot r]hl} \quad \forall h \in H, s \in S_e, l \in L \tag{4}$$

TABLE I
NOTATIONS SUMMARY OF THE MATHEMATICAL MODEL

Indices & Sets	
$i \in I$	index of component type, $I = \{1, 2, \cdots\}$
$j \in J$	index of nozzle type, $J = \{1, 2, \cdots\}$
$h \in H$	index of head, $H = \{1, 2, \cdots\}$
$p \in P$	index of placement point, $P = \{1, 2, \cdots\}$
$l \in L$	index of cycle group, $L = \{1, 2, \cdots\}$
$s \in S, S_e$[1]	index of feeder slot, $S = \{1, 2, \cdots\}$, and $S_e = \{-r \cdot (\lvert H\rvert - 1) + 1, \ldots, 0, 1, 2, \ldots, \lvert S\rvert\}$
Parameters	
T_1	the average moving time of round trip between PCB and feeder base
T_2	the average time of nozzle change operation
T_3	the average time of pickup operation
ζ_{ip}	= 1 if component type i is compatible with placement point p, otherwise, ζ_{ip} = 0
ϕ_i	the number of placement points of component type i
r	the ratio between the interval of adjacent heads and slots
τ	the interval distance between adjacent heads
M	a sufficiently large positive number.
Decision Variables	
u_{ihl}	$= 1$ if and only if head h picks up the component type i in cycle group l
z_{jhl}	$= 1$ if and only if head h is equipped with nozzle type j in cycle group l
v_{shl}	$= 1$ if and only if head h picks up component from slot s in cycle group l
f_{si}	$= 1$ if and only if component type i is arranged on slot s
p_{sl}	$= 1$ if and only there are at least one head h picking up components from slot $s+(h-1)\cdot r$ whose equivalent slot is s.
n_{lh}	$= 1$ if and only if head h changes its equipped nozzle between cycle group l and $l+1$
w_l	the number of PAP cycle in cycle group l

[1] The subset S_e refers to the equivalent slots set of aligned slots of the leftmost head when one head pickups component.

$$\sum_{h\in H} v_{[s+(h-1)\cdot r]hl} \ge p_{sl} \quad \forall s \in S_e, l \in L \tag{5}$$

The number of nozzle changes between cycle group l and $l+1$ is determined by constraint (6). Since the boards take over during the assembly process, we can regard the $(|L|+1)$st cycle as the first cycle of the next board.

$$n_{lh} = \frac{1}{2} \cdot \sum_{j\in J} \left| z_{jhl} - z_{jh(l+1)} \right| \quad \forall h \in H, l \in L \tag{6}$$

The nonlinear term of absolute value can be further linearized as present in [13], which is replaced by the sum of two positive terms n^+_{jhl} and n^-_{jhl} as

$$\begin{cases} n_{lh} = \frac{1}{2}\sum_{j\in J}\left(n^+_{jhl} + n^-_{jhl}\right), \\ z_{jhl} - z_{jh(l+1)} = n^+_{jhl} - n^-_{jhl}, \\ n^+_{jhl} \ge 0, n^-_{jhl} \ge 0 \end{cases} \quad \forall j \in J, h \in H, l \in L. \tag{7}$$

There is a coupling between the two decision variables u_{ihl} and v_{shl}, and the product of the two γ_{ishl} determines the feeder assignment as

$$f_{si} \ge \gamma_{ishl} \quad \forall i \in I, s \in S, h \in H, l \in L \tag{8}$$

$$\sum_{h\in H}\sum_{l\in L} \gamma_{ishl} \ge f_{si} \quad \forall s \in S, i \in I \tag{9}$$

with the nonlinear term $\gamma_{ishl} = u_{ihl} \cdot v_{shl}$, which represents whether the head h picks up components i from slot s in cycle

group l, is rewritten as

$$\begin{cases} \gamma_{ishl} \le u_{ihl}, \\ \gamma_{ishl} \le v_{shl}, \\ \gamma_{ishl} \ge u_{ihl} + v_{shl} - 1, \end{cases} \quad \begin{array}{l} \forall i \in I, s \in S, h \in H, \\ l \in L. \end{array} \tag{10}$$

Component assignment determines the pickup slots, and Constraint (11) specifies the relationship between the result of the pickup operation and component assignment.

$$\sum_{s \in S} v_{shl} \ge \sum_{i \in I} u_{ihl} \quad \forall h \in H, l \in L \tag{11}$$

In addition to the above improved constraints, the constraints on tool consistency and compatibility are given in [6].

C. Initial Solution with Heuristic Algorithm

The proposed model solving is a complex computing process in the branch-and-cut framework, and a high-quality initial solution could eliminate the blindness search and speed up convergence to the optimal solution. In the modeling process, the number of cycle groups $|L|$ is still an uncertain hyperparameter, which has a significant impact on the model complexity and solution quality. An initialized heuristic is proposed to determine both the hyperparameter and initial solutions of the model.

Algorithm 1: Initialized Heuristic for the ILP Model

1 **function** *model_initialize_solution*(ϕ, ξ)
2 Initialize $L \leftarrow \{1\}$ and $\mathcal{H}_j \leftarrow 1$ for $j \in J$;
3 **while** $\sum_{j \in J} \mathcal{H}_j \neq |H|$ **do**
4 $j' \leftarrow \text{argmax}_{j \in J} \{\sum_{i \in I} \xi_{ij} \cdot \phi_i / \mathcal{H}_j\}$;
5 $\mathcal{H}_{j'} \leftarrow \mathcal{H}_{j'} + 1$;
6 **end**
7 **while** ***true*** **do**
8 Let $\mathcal{C}$ be a $|L| \times |H|$ matrix, $\mathcal{W}$ be a $|L| \times 1$ matrix;
9 $res \leftarrow recursive\,(\max_{i \in I} \phi_i, \phi, 1, L, \mathcal{H}, \mathcal{C}, \mathcal{W})$;
10 **if** *res = success* **then**
11 **break**;
12 **end**
13 $L \leftarrow L \cup \{|L| + 1\}$;
14 **end**
15 **return** $\mathcal{C}, \mathcal{W}, L$
16 **end**

The pseudo-code of the initialized heuristic presented in Algorithm 1 consists of two parts. The head nozzle assignment result is determined in the first part (*line* 2∼6), i.e., the number of available heads $\mathcal{H}_j$ of nozzle type j under the condition that minimizing the number of cycles without nozzle change. After that, the algorithm recursively searches for a feasible solution by adding the placement points of the cycle group set L (*line* 7∼14). The heuristic findings workload results $\mathcal{W}_l$ and component assignment result $\mathcal{C}_{lh}$ offer the initial solution of the model, i.e., Equation (12).

$$w_l = \mathcal{W}_l, \quad u_{\mathcal{C}_{lh}hl} = 1 \quad l \in L, h \in H. \tag{12}$$

The *recursive* function is implemented as shown in Algorithm 2, which is to iteratively distribute components in a non-decreasing order of points, following the cycle group index. There are three possible cases for the return of the recursive process. Except for *success*, which indicates an initial solution has been found, *fail* indicates that the model is infeasible for the given cycle group L, while *backtrack* indicates that the current workload d for cycle group l is unsolvable and another try is executed to distribute a new workload $d - 1$.

Algorithm 2: Implementation of Function *recursive*

1 **function** *recursive*$(d,\ \phi,\ l,\ L,\ \mathcal{H}, \mathcal{C}, \mathcal{W})$
2 **if** $l > |L|$ *and* $\sum_{i \in I} \phi_i = 0$ **then**
3 **return** *success*;
4 **else if** $d \le 0$ *and* $l = 1$ **then**
5 **return** *fail*;
6 **else if** $d \le 0$ *or* $l > |L|$ **then**
7 **return** *backtrack*;
8 **end**
9 $\phi' \leftarrow \phi$, $\mathcal{H}' \leftarrow \mathcal{H}$, $\mathcal{W}_l \leftarrow d$, $h \leftarrow 0$;
10 **for** $j \in J$ **do**
11 **while** $h \leftarrow h + 1; \mathcal{H}'_j > 0$ **do**
12 $i' \leftarrow \text{argmin}_{i \in I} \{\phi_i \mid \xi_{ij} \cdot \phi_i \ge d\}$;
13 $\mathcal{C}_{lh} \leftarrow i'$, $\phi_{i'} \leftarrow \phi_{i'} - d$, $\mathcal{H}'_j \leftarrow \mathcal{H}'_j - 1$;
14 **end**
15 **end**
16 $res \leftarrow recursive(\max_{i \in I} \phi_i, \phi, l + 1, L, \mathcal{H}, \mathcal{C}, \mathcal{W})$;
17 **if** *res = success* **then**
18 **return** *success*;
19 **else if** *res = backtrack* **then**
20 **return** $recursive(d - 1, \phi', l, L, \mathcal{H}, \mathcal{C}, \mathcal{W})$;
21 **end**
22 **end**

D. Complexity Reduction Strategies for the Model

When dealing with actual production data, the high complexity of the model makes it difficult to obtain a high-quality solution in a reasonable time. It's necessary to appropriately reduce the complexity of the model in accordance with the features of PCBA, which focus on two aspects.

1) Limit the values of decision variables: As the feeders are densely arranged in an area of the feeder base, slots farther away from the PCB are always ignored. Only consecutive slots with an equal number of feeders are valid, and we define the leftmost valid slot as the reference slot, which is decided by the component assignment and consists of the following steps.

Step I average a weighted sum of the assembly heads for different types of components i with its workload.

$$\overline{h}_i \leftarrow \sum_{l \in L} \sum_{h \in H} \frac{u_{ihl} \cdot h \cdot w_l}{w_l}. \tag{13}$$

Step II convert the x coordinate of all the placement points to the position of the leftmost head and average the value.

$$\overline{x} \leftarrow \sum_{p \in P} \frac{x_p - \sum_{i \in I} \zeta_{ip} \cdot \overline{h}_i \cdot \tau}{|P|}. \tag{14}$$

where x_p and y_p are the x coordinate and the y coordinate of placement point p, respectively.

Step III calculate the average number of slots that the heads crossed by for the pickup process in one cycle on the feeder

base.

$$\Delta s \leftarrow \sum_{l \in L} \frac{\mathcal{R}\left\{v_{shl} \cdot (s - h \cdot r + r) \mid v_{shl} \neq 0, s \in S, h \in H\right\}}{w_l} \tag{15}$$

where $\mathcal{R}\{\cdot\}$ denotes the range of the set.

Step IV determine the reference slot s^{REF} based on the head pickup range (slots crossed by) and the average placement position of the head.

$$s^{\mathrm{REF}} \leftarrow \lfloor \frac{\overline{x} - s^{\mathrm{F1}}}{\tau} \cdot r + \frac{\Delta s + 1}{2} \rfloor + 1 \tag{16}$$

where s^{F1} is the x coordinate of the leftmost slot on the feeder base. The feeder slot for component type i is computed from the solution of the model and the reference slot position, i.e., $s^{\mathrm{REF}} + r \cdot \sum_{s \in S} s \cdot f_{si}$.

2) Reduce the range of feasible domains: The solution space of the model is cut by adding constraints to further improve the solving efficiency. Constraints (17)–(20) are not the necessary condition for model solving and are utilized to reduce the range of feasible domains further and round out inappropriate solutions ahead of time.

~~Constraint (17) ensures the lower cycle group has no less workload than the higher cycle group.~~ Constraint (17) ensures that the lower cycle group has a higher priority in picking up components with more PAP cycles.

$$w_l \geq w_{l+1} \quad \forall l \in L \backslash \{|L|\} \tag{17}$$

~~Constraint (18) gives the lower bound and upper bound of the number of PAP cycles, where the upper one is given based on the heuristic initialize algorithm.~~ The heuristic solution $\mathcal{W}_l$ gives the worst case for the number of total PAP cycles without nozzle change, and an optimal case is that all heads divide components equally; two of these cases give the upper bound and lower bound of cycle groups in Constraint (18).

$$\lceil \sum_{i \in I} \phi_i / |H| \rceil \leq \sum_{l \in L} w_l \leq \sum_{l \in L} \mathcal{W}_l \tag{18}$$

~~Constraint (19) presumes that all placement heads have nozzles, even if they do not pick up and position components, which helps to eliminate the unsatisfactory result.~~ A general case in Constraint (19) is that all heads are not empty, even if they do not pick up any components.

$$\sum_{h \in H} \sum_{j \in J} z_{jhl} = |H| \quad \forall l \in L \tag{19}$$

~~Constraint (20) limits workload with component assignment results to improve the search speeds further.~~

~~$$M \cdot \sum_{i \in I} \sum_{h \in H} u_{ihl} \geq w_l \quad \forall l \in L \tag{20}$$~~

E. Selection Criteria of Solution Pool

In general, the solution of the model is not unique, and ~~different solutions indirectly affect the movements of the gantry.~~ standard solvers can systematically search for a solution pool- a collection of multiple optimal solutions. The model determines both the component assignment and feeder arrangement. However, its objective function does not incorporate the pickup movement, which leads to different pickup paths that have the same objective values. It also does not take into account the layout of the placement points, but its solution limits the set of points that each head can access.

As there is insufficient information regarding the points and sequence of heads placed, we propose a fast pre-evaluation heuristic algorithm for selecting one result from the solution pool. The assignment of the head task determines the path of the pickup process as

$$E_1 = \frac{\tau}{r} \sum_{l \in L} w_l \cdot \mathcal{R}\left\{v_{shl} \cdot (s - h \cdot r + r) \mid v_{shl} \neq 0, s \in S, h \in H\right\} \tag{21}$$

~~Regarding the placement movement,~~ The placement points set for each head is constrained by the component assignment of the model. For the placement process, the first w_l points of the component type $\sum_{i \in I} i \cdot u_{ihl}$ are assigned to the head h, followed by the subsequent w_l points, and so forth. For each head in the cycle group, we implement a route schedule for the centroids of the assigned points, and the length of placement route movement is denoted by E_2. Out of all the solutions in the pool, the one with the minimal $E_1 + E_2$ is selected for the next phase of optimization.

III. Route Schedule Heuristic

The placement route scheduling problem has a wide solution space, and heuristic algorithms based on expertise or rules are appropriate and generally yield satisfactory results. On the basis of the mechanical structure of beam-heads, we propose greedy-based and random route relink heuristics for the placement route schedule.

A. Greedy-Based Route Schedule Heuristic

The greedy-based route schedule heuristic consists of the following steps.

Step I compute the x coordinate of left boundary α and right boundary β of the PCB and repeat through the *Step II* to *Step VII* with the search step $\delta = (\beta - \alpha) / (2 \cdot |H|)$ and three distinct search directions: from left to right (L→R), from right to left (R→L), from center to edge (C→E).

Step II generate the starting point list $\hat{\mathcal{S}}$ and head list $\hat{\mathcal{H}}$ - linear sequences based on the search direction.

L→R: $\hat{\mathcal{S}} = \{\alpha + (h-1) \cdot \delta \mid h \in H\}$, $\hat{\mathcal{H}} = H$.

R→L: $\hat{\mathcal{S}} = \{\beta - (h-1) \cdot \delta \mid h \in H\}$, $\hat{\mathcal{H}} = \{|H| + 1 - h \mid h \in H\}$.

C→E: $\hat{\mathcal{S}} = \{(3 \cdot \alpha + \beta)/4 + (h-1) \cdot 2/\delta \mid h \in H\}$, $\hat{\mathcal{H}} = \left\{\lceil |H| + 1/2 \rceil - (-1)^h \cdot (\lceil h/2 \rceil - 1/2) - 7/2 \mid h \in H\right\}$.

The head list $\hat{\mathcal{H}}$ represents the order in which the different heads are assigned to the search direction.

Step III repeat through the cycle index $k \in K$, where $K = \left\{1, 2, \cdots, \sum_{l \in L} w_l\right\}$ and initialize $\mathcal{P}_k$ as a $1 \times |H|$ array with elements of -1, which represents the placement result.

Step IV repeat through search direction L→R, R→L, C→E with starting point $\Theta \in \hat{\mathcal{S}}$ and head list $\hat{\mathcal{H}}$.

Step V iterate through all the heads $h \in \hat{\mathcal{H}}$. If h is the first one, find the point nearest to the starting point in the horizontal direction,

$$p \leftarrow \operatorname{argmin}_{p' \in \{p'' | \iota(p'')=\mathcal{C}_{kh}, p'' \in P\}} \left| x_{p''} - \Delta\tau_h - \Theta \right| \tag{22}$$

otherwise, sort the assigned placement points and calculate the moving distance, where $\Delta\tau_h = (h-1)\cdot\tau$ and $\iota(p)$ is the component type of placement point p.

$$\mathcal{X}_p \leftarrow \left\{ x_{\mathcal{P}_{kh'}} - \Delta\tau_h \mid \mathcal{P}_{kh'} \neq 1, h' \in H \right\} \cup \{x_p\} \tag{23}$$

$$\mathcal{Y}_p \leftarrow \left\{ y_{\mathcal{P}_{kh'}} \mid \mathcal{P}_{kh'} \neq 1, h' \in H \right\} \cup \{y_p\} \tag{24}$$

Note q is the index of $\mathcal{X}$ with the qth smallest coordinate of x axis, and

$$p \leftarrow \operatorname{argmin}_{p' \in \{p'' | \iota(p'')=\mathcal{C}_{kh}, p'' \in P\}} \sum_{q=1}^{\mathcal{X}_{p'}-1} \max\left(\left| \mathcal{X}_{p'q} - \mathcal{X}_{p'(q+1)} \right|, \left| \mathcal{Y}_{p'q} - \mathcal{Y}_{p'(q+1)} \right| \right) \tag{25}$$

Step VI update the placement assignment result $\mathcal{P}_{kh} \leftarrow p$, $P \leftarrow P \backslash \{p\}$, go to *Step V* until $\mathcal{P}_{kh} \neq -1, \forall h \in H$.

Step VII dynamic programming for route schedule in each cycle and storing the Chebyshev moving distance. The x coordinate of the center point Φ equals $\sum_{h\in H} x_{\mathcal{P}_{kh}} / |H|$ and its y coordinate equals the pickup position of the feeder slot. The transfer equation is written as

$$\mathcal{F}(\Phi, \{\Phi\}) \leftarrow 0 \tag{26}$$

$$\mathcal{F}\left(h, \hat{\mathcal{H}}' + \{h\}\right) \leftarrow \min_{h' \in \hat{\mathcal{H}}'} \left\{ \mathcal{F}\left(h', \hat{\mathcal{H}}'\right) + g(h, h') \right\}, \quad \hat{\mathcal{H}}' \subseteq \hat{\mathcal{H}} = H \cup \{\Phi\}, h \in H \tag{27}$$

if $h \neq \Phi$ and $h' \neq \Phi$,

$$g(h, h') = \max\left(\left| x_{\mathcal{P}_{kh}} - x_{\mathcal{P}_{kh'}} - \Delta\tau_{h-h'} \right|, \left| y_{\mathcal{P}_{kh}} - y_{\mathcal{P}_{kh'}} \right| \right) \tag{28}$$

otherwise,

$$g(h, \Phi) = \max\left(\left| x_{\mathcal{P}_{kh}} - \Phi_x - \Delta\tau_h \right|, \left| y_{\mathcal{P}_{kh}} - \Phi_y \right| \right) \tag{29}$$

with final result equals $\min_{h\in\hat{\mathcal{H}}} \left\{ \mathcal{F}\left(h, \hat{\mathcal{H}}\right) + g(h, \Phi) \right\}$.

Each head is associated with one placement position, and the sequence in which the heads are placed is solved. The placement sequence pair $\mathcal{Q}$ is formed by arranging the two heads consecutively.

Step VIII compare the total moving distance and get the placement assignment result with the minimal one.

B. Aggregated Route Relink Heuristic

An aggregated route relink heuristic (ARRH) is proposed for the placement route improvement, and its flow is shown in Algorithm 3. The primary principle of the algorithm is to reallocate the off-center points in each cycle. The design of the algorithm is based on the average position and moving distance in each cycle (*line* 1). The cycle and its corresponding off-center point are determined based on the moving distance and offset, respectively (*line* 4). The swapping cycle, which is nearest to the former off-center point, and the swapping point are further determined (*line* 5∼11). After performing the relink operation (*line* 12), the distribution of the cycle can be more concentrated. The proposed *cycle_schedule* relinks the placement routes with a plain idea for search faster: sorting the placement points non-decreasingly w.r.t. coordinate of x axis and allocating them on the head from left to right.

Algorithm 3: The Flow of ARRH Algorithm

Input : component assignment $\mathcal{C}$, placement assignment $\mathcal{P}$, placement sequence $\mathcal{Q}$
Output: reschedule placement assignment $\widetilde{\mathcal{P}}$ and placement sequence $\widetilde{\mathcal{Q}}$

1 calculate average position $\overline{x}_k$, $\overline{y}_k$ and moving distance D_k, $\overline{x}_k \leftarrow \sum_{h\in H} x_{\mathcal{P}_{kh}} / |H|$, $\overline{y}_k \leftarrow \sum_{h\in H} y_{\mathcal{P}_{kh}} / |H|$, $D_k \leftarrow \sum_{(q_1,q_2)\in\mathcal{Q}_k} \max\left(\left| x_{\mathcal{P}_{kq_1}} - x_{\mathcal{P}_{kq_2}} \right|, \left| y_{\mathcal{P}_{kq_1}} - y_{\mathcal{P}_{kq_2}} \right| \right)$ in each cycle k, $k \in K = \{1, 2, \cdots, \sum_{l\in L} w_l\}$;
2 $\widetilde{\mathcal{P}} \leftarrow \mathcal{P}$, $\widetilde{\mathcal{Q}} \leftarrow \mathcal{Q}$;
3 **while** *the terminated time has not been reached* **do**
4 $p_r \leftarrow \mathcal{P}_{k_r h_r}$ where $k_r \leftarrow$ *random*$_{k\in K}(D_k)$, $h_r \leftarrow$ *random*$_{h\in H}\left(\max\left(\left| x_{\mathcal{P}_{k_r h}} - \overline{x}_{k_r} \right|, \left| y_{\mathcal{P}_{k_r h}} - \overline{y}_{k_r} \right| \right) \right)$;
5 $k_c \leftarrow \operatorname{argmin}_{k'\in K, k'\neq k_r} \max\left(|x_{p_r} - \overline{x}_{k'}|, |y_{p_r} - \overline{y}_{k'}| \right)$;
6 **for** $h \in H$ **do**
7 $\overline{x} \leftarrow \frac{x_{p_r} - x_{\mathcal{P}_{k_r h}}}{|H|} + \overline{x}_k$, $\overline{y} \leftarrow \frac{y_{p_r} - y_{\mathcal{P}_{k_r h}}}{|H|} + \overline{y}_k$;
8 $u_h \leftarrow \max\left(|x_{p_r} - \overline{x}|, |y_{p_r} - \overline{y}| \right)$;
9 **foreach** $h' \in H \backslash \{h\}$ **do**
$u_h \leftarrow u_h + \max\left(\left| x_{\mathcal{P}_{k_r h'}} - \overline{x} \right|, \left| y_{\mathcal{P}_{k_r h'}} - \overline{y} \right| \right)$;
10 **end**
11 $h_c \leftarrow \operatorname{argmin}_{h\in\left\{h' | \iota(p_r)=\iota\left(\mathcal{P}_{k_c h'}\right), h'\in H\right\}} u_h$, $p_c \leftarrow \mathcal{P}_{k_c h_c}$;
12 $\mathcal{P}_{k_c h_c} \leftarrow p_r$, $\mathcal{P}_{k_r h_r} \leftarrow p_c$;
13 $D'_{k_c}, \mathcal{Q}_{k_c} \leftarrow$ *cycle_schedule* $\left(\mathcal{P}_{k_c}\right)$, $D'_{k_r}, \mathcal{Q}_{k_r} \leftarrow$ *cycle_schedule* $\left(\mathcal{P}_{k_r}\right)$;
14 **if** $D_{k_r} + D_{k_c} > D'_{k_r} + D'_{k_c}$ **then**
15 $\widetilde{\mathcal{P}} \leftarrow \mathcal{P}$, $\widetilde{\mathcal{Q}} \leftarrow \mathcal{Q}$, $D_{k_r} \leftarrow D'_{k_r}$, $D_{k_c} \leftarrow D'_{k_c}$;
16 $\overline{x}_{k_c} \leftarrow \frac{x_{p_c} - x_{\mathcal{P}_{k_c h_c}}}{|H|} + \overline{x}_{k_c}$, $\overline{y}_{k_c} \leftarrow \frac{y_{p_c} - y_{\mathcal{P}_{k_c h_c}}}{|H|} + \overline{y}_{k_c}$,
17 $\overline{x}_{k_r} \leftarrow \frac{x_{p_r} - x_{\mathcal{P}_{k_r h_r}}}{|H|} + \overline{x}_{k_r}$, $\overline{y}_{k_r} \leftarrow \frac{y_{p_r} - y_{\mathcal{P}_{k_r h_r}}}{|H|} + \overline{y}_{k_r}$
18 **else**
19 $\mathcal{P} \leftarrow \widetilde{\mathcal{P}}$, $\mathcal{Q} \leftarrow \widetilde{\mathcal{Q}}$;
20 **end**
21 **end**

IV. Experiment Result

A. Experiment Setup

This article solves the model using Gurobi 10.0 and Python 3.10 on the Intel(R) Core(TM) i5-11400 @2.60GHz with 16G RAM. Five times of runs are implemented with each PCB, and the average values are recorded as the comparative metrics. The proposed two-phase PCBA optimization (TPPO) is compared with four representative decomposition-based algorithms, including a component placer optimizer (CPO) employed in ~~an~~industrial software, hybrid genetic algorithm (HGA) [12], aggregated model (AGM) [13] and cell division genetic algorithm (CDGA) [17]. The experimental platform of a self-developed placement machine is shown in Fig. 2.

In Table II, which lists the basic parameters of the PCB data, we select ten different PCB data; among them, the first one is

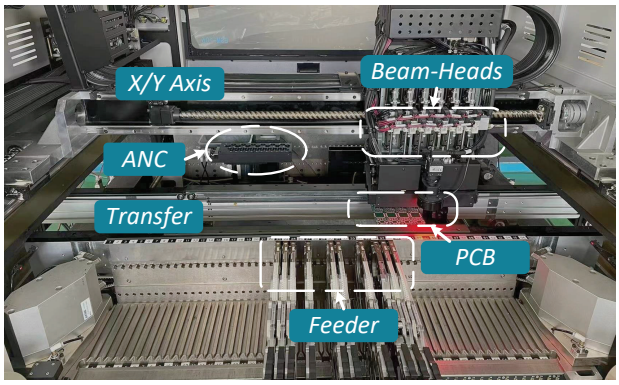

Fig. 2. The experimental platform of a placement machine.

TABLE II
BASIC PARAMETERS OF THE PCB DATA

PCB	1	2	3	4	5	6	7	8	9	10
$\|N\|$	1	1	1	2	2	3	2	3	3	4
$\|C\|$	1	2	3	4	5	5	6	7	8	10
$\|P\|$	400	216	288	352	432	384	336	198	170	196

an international standard speed test data IPC9850; the second to the fifth data with relatively ~~less~~fewer component types and randomly generated placement points are applied to test the generalization of the algorithm; the last five are selected from the actual industrial sites, to validate the application of the algorithm in practice.

The parameter settings of the proposed algorithm are listed in Table III. In the first phase, we set the pool parameters and search mode, as well as the coefficients of the model based on the metrics' impact on assembly efficiency. We specify the following as the termination condition of the model-solving process because it takes a long time to solve the model completely: the currently optimal solution has not changed for more than 30 seconds. The big M value for linearization equals the number of placement points. The search mode is set to prioritize the 30 best solutions within the gap of 10^{-4}. In the second phase, the search step is dependent on the PCB layout, and the route roulette wheel is chosen for the random search of route relink with the upper 10 seconds.

B. Comparative Experiments

The sub-objectives of the PCBA process, which include the number of cycles, nozzle changes, and pickup operations, with the comparative histogram is ~~illustrated~~shown in Fig. 3. It

TABLE III
THE PARAMETER SETTING OF THE TWO-PHASE ALGORITHM

Phase	Parameter	Setting
I	Coefficient $T_1 \mid T_2 \mid T_3$	$2 \mid 3 \mid 2$
	Big-M value	$\|P\|$
	Pool search mode	Find multiple solutions
	Pool solution	30
	Pool gap	10^{-4}
	Terminated condition	Unchanged in 30 seconds
II	Search step	$\mathcal{R}\left(\{x_p \mid p \in P\}\right) / \|H\|$
	Selection method	Roulette wheel
	Terminated time (sec)	10

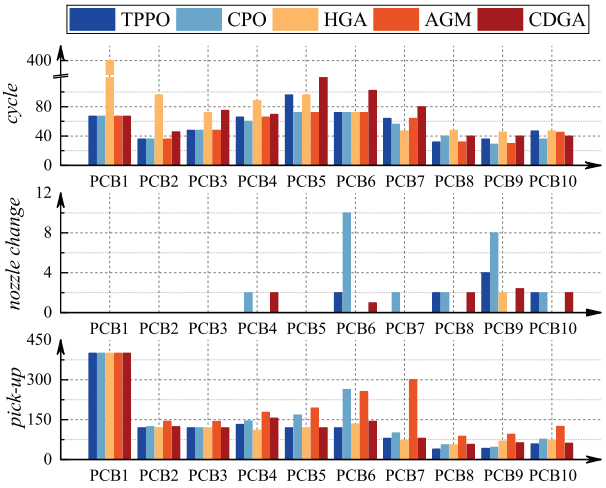

Fig. 3. The histogram of the sub-objectives comparison between the proposed model and other mainstream algorithms.

can be seen the TPPO is more comprehensive than conventional approaches. The cycle scheduling difficulties are better handled by TPPO, AGM, and CPO, whereas evolutionary-based CDGA and HGA typically have more PAP cycles. AGM and HGA forbid changing the nozzle, which prevents some of the simultaneous pickup operations from being carried out and lowers the overall efficiency. Both TPPO and AGM are model based algorithms; however, the former takes into account the mechanical characteristics and has a greater pickup efficiency. ~~In most PCBs, the cycle counts of TPPO and CPO are comparable, but TPPO performs better in terms of nozzle changes and pickups.When comparing the Z-values of weighted sub-objectives in Table IV, it becomes evident that AGM is not as effective as the methods that consider pickup efficiency.~~ Table IV shows more general and comparable results of Z-values for weighted sub-objectives that are directly related to assembly efficiency. It can be seen that when dealing with a single type of component data (PCB1), TPPO, CPO, and AGM perform equally well. As the PCB becomes more complicated with more component types, the TPPO outperforms other mainstream algorithms, and there is also a tendency to increase gaps between the proposed algorithm and other research.

Three test cases (TCs) are constructed to compare the solving efficiency for different model settings in Table V. We call the model with component partition, complexity reduction strategies as the improved model and the model without the proposed techniques as the original model. We utilize the ~~near~~ known optimal solution as a benchmark since it is hard to find the optimal one to a NP hard problem for all PCBs. The benchmark value $\mathcal{O}_b$ of PCB1∼PCB3 are the optimal result for solving the original model. As the size of the data increases, the original model cannot find an optimal solution in an acceptable time. The solutions of PCB4∼PCB10 are obtained after solving the proposed model with a sufficient amount of time (at least 6 hours) and without the terminated conditions, which are also the best results from the proposed and comparative methods. ~~The improved model's solutions,~~

TABLE IV
COMPARISON OF THE OBJECTIVES' Z VALUE OF THE PROPOSED MODEL WITH MAINSTREAM ALGORITHMS

PCB	TPPO	CPO	HGA	AGM	CDGA
1	**-0.448**	**-0.448**	1.789	**-0.448**	-0.446
2	**-0.845**	-0.679	1.650	0.153	-0.279
3	**-1.089**	**-1.089**	0.677	0.603	0.898
4	**-0.864**	-0.318	**-0.864**	1.420	0.625
5	**-0.942**	0.211	**-0.942**	1.461	0.211
6	**-0.996**	1.208	-0.840	0.883	-0.254
7	**-0.527**	-0.370	**-0.527**	1.783	-0.360
8	**-1.470**	-0.104	0.238	1.331	0.005
9	**-1.100**	-0.936	0.763	1.147	0.127
10	**-0.715**	-0.431	-0.293	1.764	-0.325
AVG	**-0.900**	-0.295	0.165	1.010	0.020

TABLE V
COMPARISON OF THE MODEL~~SOLVING PROCESS~~ OBJECTIVE VALUE FOR ~~WITH~~ DIFFERENT TEST CASES

	PCB	1	2	3	4	5	6	7	8	9	10
BASE	$\mathcal{O}_b$	934	312	336	396	432	390	288	158	164	196
TC-1	$\mathcal{O}_1$	934	312	336	396	432	390	288	158	168	218
	$\mathcal{G}_1$	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	2.44	11.22
TC-2	$\mathcal{O}_2$	934	312	336	396	432	390	288	162	-	-
	$\mathcal{G}_2$	0.00	0.00	0.00	0.00	0.00	0.00	0.00	2.53	-	-
TC-3	$\mathcal{O}_3$	934	312	336	396	432	390	288	172	192	220
	$\mathcal{G}_3$	0.00	0.00	0.00	0.00	0.00	0.00	0.00	8.86	17.07	12.24

~~which are also the best ones found compared with another research, are selected as the benchmark for PCB4∼PCB7. The solutions of PCB8∼PCB10 are obtained by solving the improved model with sufficient search time (at least 6 hours).~~

The test cases follow~~ing~~ the settings: TC-1 represents the solution of the improved model; TC-2 represents the solution of the improved model without the initial solution; and TC-3 represents the solution of the improved model without the complexity reduction strategies. The formula for the test case t's gap is $\mathcal{G}_t = (\mathcal{O}_t/\mathcal{O}_b - 1) \cdot 100\%$, $t = 1, 2, 3$. As can be shown, the improved model's highest gap from the benchmark is 11.22%. The model-solving process can be quickly iterated with the aid of the initial solution, and under the terminated condition, the feasible solutions for PCB9 and PCB10 are not even attainable. Even though, theoretically, TC-3 could achieve better solutions, the model iterates more slowly in practice and has a larger gap than the improved model under the terminated condition.

The movement distance and assembly time are compared next, as shown in Table VI. The notation $\mathcal{D}$ and $\mathcal{T}$ represent the moving distance and assembly time, while the superscripts T, P, H, A, and C represent the TPPO, CPO, HGA, AGM, and CDGA, respectively. $\Delta\mathcal{D}$ and $\Delta\mathcal{T}$ correspond to the improvement rates of $\mathcal{D}$ and $\mathcal{T}$, respectively, relative to TPPO comparing with other research. $\mathcal{D}_1^T$ and $\mathcal{D}_2^T$ represent the moving distance without and with route relink heuristic. Since the route relink heuristic mainly adjusts the placement movement that makes up a small portion of the whole, it does not result in a high improvement in the overall movement. For the TPPO method, the assembly process can be more effective with fewer pickups and nozzle changes, even without the shortest movement distance for PCB3, PCB4, and PCB7. Compared to CPO, CDGA, AGM, and HGA, the proposed method improves by 8.06%, 13.06%, 24.32%, and 24.31% in assembly efficiency, respectively.

Lastly, we compare the solving time. CPO is not included in the comparison since the specific algorithm has not been disclosed. As shown in Table VII, compared with the TPPO, we can conclude that the component partition is an effective way to improve the search efficiency. The model without component partition can only be applied in solving small-scale data; for PCB1∼PCB3, the solving time is 21.41, 70.18, and 193.23 seconds, respectively, which is much larger than the proposed model. As a modeling method, TPPO is solved longer because of the inclusion of pickup constraints compared to AGM, but it is significantly faster than HGA excepted PCB10. Even though it requires more time for TPPO, its assembly efficiency is higher, and the time is within an acceptable amount.

V. CONCLUSION

This article presented a two-phase optimization approach for handling the head task assignment and placement route schedule after breaking the PCBA process down into two parts. By optimizing the primary sub-objectives at the modeling phase and developing heuristic algorithms at the route schedule phase, the two-phase framework combined the advantages of both mathematical model and heuristic algorithms. We compared the weighted sub-objective, which was related to the overall assembly efficiency, with both heuristic-based and model-based algorithms. The results showed that the proposed algorithms are more thorough than previous research. A series of specialized test cases validated the necessity of the pre-processing technique, including the component partition approach, initial heuristic, and reduction strategies, to solve the model. Furthermore, we compared the moving distance and assembly time with other research. Although the placement path of our proposed algorithms was not the shortest for some PCB data, it improved the assembly efficiency because of the optimization in the first phase. The solving time of the two-phase algorithm was within acceptable bounds, even though it was not faster than all the compared algorithms because we took more factors into account and searched a greater domain. Overall, the experimental results showed that the proposed two-phase optimization effectively solves PCBA problems, balancing the quality of the solution and computational cost.

REFERENCES

[1] M. Ayob and G. Kendall, "A survey of surface mount device placement machine optimisation: Machine classification," *Eur. J. Oper. Res.*, no. 3, pp. 893–914, May 2008.
[2] W. Ho and P. Ji, "An integrated scheduling problem of PCB components on sequential pick-and-place machines: Mathematical models and heuristic solutions," *Expert Syst. Appl.*, vol. 36, no. 3, pp. 7002–7010, Apr. 2009.
[3] J. Luo and J. Liu, "An MILP model and clustering heuristics for LED assembly optimisation on high-speed hybrid pick-and-place machines," *Int. J. Prod. Res.*, vol. 52, no. 4, pp. 1016–1031, Feb. 2014.
[4] H.-P. Hsu, "Solving the feeder assignment, component sequencing, and nozzle assignment problems for a multi-head gantry SMT machine using improved firefly algorithm and dynamic programming," *Adv. Eng. Inform.*, vol. 52, p. 101583, Apr. 2022.

TABLE VI
COMPARISON OF THE ROUTE SCHEDULE AND ASSEMBLY TIME OF THE PROPOSED HEURISTIC WITH MAINSTREAM ALGORITHMS

PCB	TPPO			CPO			HGA			AGM			CDGA		
	$\mathcal{D}_1^T$	$\mathcal{D}_2^T$	$\mathcal{T}^T$	$\mathcal{D}^P$	$\mathcal{T}^P$	$\Delta\mathcal{T}^P$	$\mathcal{D}^H$	$\mathcal{T}^H$	$\Delta\mathcal{T}^H$	$\mathcal{D}^A$	$\mathcal{T}^A$	$\Delta\mathcal{T}^A$	$\mathcal{D}^C$	$\mathcal{T}^C$	$\Delta\mathcal{T}^C$
1	34793.6	**34676.1**	114.63	35063.0	**114.25**	-0.33	131457.9	205.57	79.34	45110.9	134.82	17.62	35865.9	122.73	7.07
2	20304.0	**20059.8**	**52.99**	20207.5	53.31	-0.60	44652.9	75.33	41.29	25808.2	61.40	15.16	25711.8	59.09	10.84
3	28652.0	28390.1	66.69	**27127.4**	**65.57**	-1.68	40722.4	80.88	21.29	35627.2	76.89	15.29	39437.7	77.29	15.90
4	36825.0	36690.1	**82.02**	**35870.2**	86.51	5.47	48292.8	93.30	13.76	52397.8	101.16	23.34	43012.9	96.62	17.81
5	40952.0	**40707.8**	**95.83**	44026.4	100.20	4.56	56680.0	109.98	14.77	55825.1	114.75	19.74	58445.3	109.31	14.07
6	39096.8	**38905.2**	**90.68**	41211.0	118.00	30.12	46366.5	98.36	8.47	55493.9	117.73	29.84	54717.3	107.02	18.03
7	33676.7	33277.2	**72.97**	**32253.8**	76.56	4.92	35640.9	77.98	6.87	52810.7	124.17	56.46	42133.4	80.46	10.27
8	19799.6	**19662.2**	**45.97**	25177.6	51.31	11.62	25745.5	49.78	8.30	27170.6	49.85	8.45	24533.2	52.35	13.88
9	19938.4	**19535.4**	**41.31**	21142.5	53.81	30.26	23629.5	46.00	11.35	23376.5	48.49	17.38	23444.1	49.29	19.31
10	26024.8	25814.3	52.82	25959.3	54.03	2.29	**25563.6**	**52.74**	-0.15	30795.8	60.76	15.03	26433.5	55.28	-0.15
AVG	30006.3	**29771.8**	**71.59**	30803.9	77.36	8.06	47875.2	88.99	24.31	40441.7	89.00	24.32	37373.5	80.94	13.06

TABLE VII
COMPARISON OF THE SOLVING TIME OF THE PROPOSED MODEL WITH MAINSTREAM ALGORITHMS

PCB	TPPO	HGA	AGM	CDGA	PCB	TPPO	HGA	AGM	CDGA
1	0.4	138.2	0.3	-	6	34.7	264.2	0.5	30.1
2	4.2	218.2	0.2	41.0	7	32.0	94.2	1.1	30.1
3	15.9	373.0	0.2	35.7	8	67.6	88.0	0.9	20.1
4	31.5	134.6	0.3	36.8	9	46.4	158.9	0.4	23.0
5	31.5	172.8	0.4	33.5	10	95.3	153.9	1.2	27.0

[5] J. Luo, J. Liu, and Y. Hu, "An MILP model and a hybrid evolutionary algorithm for integrated operation optimisation of multi-head surface mounting machines in PCB assembly," *Int. J. Prod. Res.*, vol. 55, no. 1, pp. 1–16, Jun. 2016.

[6] G. Lu, X. Yu, H. Sun, Z. Li, J. Qiu, and H. Gao, "A scan-based hierarchical heuristic optimization algorithm for PCB assembly process," *IEEE Trans. Industr. Inform.*, vol. 20, no. 3, pp. 3609–3618, 2024.

[7] J. Ashayeri and W. Selen, "A planning and scheduling model for onsertion in printed circuit board assembly," *Eur. J. Oper. Res.*, pp. 909–925, Dec. 2007.

[8] C. Raduly-Baka, T. Knuutila, M. Johnsson, and O. S. Nevalainen, "Selecting the nozzle assortment for a gantry-type placement machine," *OR Spectrum*, vol. 30, no. 3, pp. 493–513, Nov. 2008.

[9] S. Guo, K. Takahashi, and K. Morikawa, "PCB assembly scheduling with alternative nozzle types for one component type," *Flex. Serv. Manuf. J.*, vol. 23, no. 3, pp. 316–345, Sep. 2022.

[10] C. Raduly-Baka, M. Johnsson, and O. S. Nevalainen, "Tool-feeder partitions for module assignment in PCB assembly," *Comput. Oper. Res.*, vol. 78, pp. 108–116, Feb. 2017.

[11] D.-S. Sun and T.-E. Lee, "A branch-and-price algorithm for placement routing for a multi-head beam-type component placement tool," *OR. Spectrum*, vol. 30, no. 3, pp. 515–534, Jun. 2008.

[12] S. Guo, F. Geng, K. Takahashi, X. Wang, and Z. Jin, "A MCVRP-based model for PCB assembly optimisation on the beam-type placement machine," *Int. J. Prod. Res.*, vol. 57, no. 18, pp. 5874–5891, Sep. 2019.

[13] J. Ashayeri, N. Ma, and R. Sotirov, "An aggregated optimization model for multi-head SMD placements," *Comput. Ind. Eng.*, vol. 60, no. 1, pp. 99–105, Jan. 2011.

[14] G.-Y. Zhu, X. Ju, and W.-B. Zhang, "Multi-objective sequence optimization of PCB component assembly with GA based on the discrete frechet distance," *Int. J. Prod. Res.*, vol. 56, pp. 1–18, Mar. 2018.

[15] M. Ayob and G. Kendall, "The optimisation of the single surface mount device placement machine in printed circuit board assembly: a survey," *Int. J. Syst. Sci*, vol. 40, no. 6, pp. 553–569, 2009.

[16] D.-S. Sun, T.-E. Lee, and K.-H. Kim, "Component allocation and feeder arrangement for a dual-gantry multi-head surface mounting placement tool," *Int. J. Prod. Econ.*, vol. 95, no. 2, pp. 245–264, Feb. 2005.

[17] Z. Li, X. Yu, J. Qiu, and H. Gao, "Cell division genetic algorithm for component allocation optimization in multi-functional placers," *IEEE Trans. Industr. Inform.*, vol. 18, no. 1, pp. 559–570, Mar. 2022.

[18] D. Li and S. W. Yoon, "PCB assembly optimization in a single gantry high-speed rotary-head collect-and-place machine," *Int. J. Adv. Manuf. Technol.*, vol. 88, pp. 2919–2834, 2017.

[19] H.-P. Hsu, "Solving feeder assignment and component sequencing problems for printed circuit board assembly using particle swarm optimization," *IEEE Trans. Autom. Sci. Eng.*, vol. 14, no. 2, pp. 881–893, Apr. 2017.

[20] H.-P. Hsu and S.-W. Yang, "Optimization of component sequencing and feeder assignment for a chip shooter machine using shuffled frog-leaping algorithm," *IEEE Trans. Autom. Sci. Eng.*, vol. 17, no. 1, pp. 56–71, June. 2020.

[21] G.-Y. Zhu and W.-B. Zhang, "An improved shuffled frog-leaping algorithm to optimize component pick-and-place sequencing optimization problem," *Expert Syst. Appl.*, vol. 41, no. 15, pp. 6818–6829, Nov. 2014.

[22] S. Torabi, M. Hamedi, and J. Ashayeri, "A new optimization approach for nozzle selection and component allocation in multi-head beam-type SMD placement machines," *J. Manuf. Syst.*, vol. 32, pp. 700–714, Oct. 2013.

[23] D. Li, T. He, and S. W. Yoon, "Clustering-based heuristic to optimize nozzle and feeder assignments for collect-and-place assembly," *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 2, pp. 755–766, Apr 2019.

[24] T. He, D. Li, and S. W. Yoon, "An adaptive clustering-based genetic algorithm for the dual-gantry pick-and-place machine optimization," *Adv. Eng. Inform.*, vol. 37, pp. 66–78, Aug 2018.

[25] L. Gurobi Optimization, "Gurobi optimizer reference manual," 2022. [Online]. Available: "https://www.gurobi.com"