Highlights

A Swarm Intelligence Based Optimization for Multi-Objective Vehicle Routing and Packing Problem

Hsin-Ping Liu[a,b], Frederick Kin Hing Phoa[b], Yun-Heh (Jessica) Chen-Burger[c]

- The Multi-Objective Vehicle Routing and Packing Problem is newly formulated.
- The proposed SIB-MOVRP leverages NSGA-II for handling multi-objective problems.
- A decoding function is proposed to generate the actual 2D placement of goods.
- The SIB-MOVRP provides multiple optimized options for further decision-making.
- The algorithm performs well across three different supply chain configurations.
- A new SIB framework that is generic to address real-world supply chain problems in a more comprehensive manner.

A Swarm Intelligence Based Optimization for Multi-Objective Vehicle Routing and Packing Problem

Hsin-Ping Liu[a,b], Frederick Kin Hing Phoa[b], Yun-Heh (Jessica) Chen-Burger[c]

Data Science Degree Program, National Taiwan University, No.1, Sec. 4, Roosevelt Road, Taipei, 106319, Taiwan

Institute of Statistical Science, Academia Sinica, 128 Section 2, Academia Road, Nankang, Taipei, 115201, Taiwan

Electrical Engineering and Computer Science, Heriot-Watt University, Edinburgh, EH14 4AS, Scotland, United Kingdom

---

Abstract

Efficient logistics execution is critical for a supply chain, however, it frequently needs to wrestle with conflictive goals that often renders traditional optimization approaches intractable. This study introduces an innovative swarm intelligence based algorithm to address the Multi-Objective Vehicle Routing and Packing Problem (MOVRPP), an extension of the 2L-CVRP that incorporates realistic two-dimensional packing constraints and explicit unloading overheads. This Swarm Intelligence-Based Multi-Objective Vehicle Routing and Packing (SIB-MOVRP) algorithm, which integrates NSGA-II into the SIB framework and employs a priority and heuristic-based decoding function. Evaluations across diverse scenarios demonstrate that SIB-MOVRP consistently outperforms a deterministic baseline. When coupled with Pareto Front, its ranked solutions with optimization of different goals provide decision-makers with valuable flexible and trade-off information for real-world applications.

Keywords:
Swarm intelligence, Logistic planning, Optimization, Two-dimensional packing problem, Multiple objective

---

1. Introduction

In today's highly competitive global market, the ability to generate efficient logistics plans has become a critical determinant for a company's success. While advances in manufacturing have led to increasingly comparable production costs across firms, optimized logistics strategies can provide substantial competitive advantages by significantly reducing operational costs and enhancing customer satisfaction, particularly in large-scale operations such as e-commerce, retail distribution, and furniture delivery.

However, the inherently complex decision-making processes in logistics optimization pose significant computational challenges. These problems are typically classified as NP-hard due to their combinatorial nature, large-scale instances, and numerous intertwined conflicting constraints and objectives. The Vehicle Routing Problem (VRP) has been extensively studied in the literature, with solutions ranging from exact algorithms [1] suitable for small- to medium-sized instances to a variety of metaheuristics, such as Genetic Algorithms (GA) [2, 3] and Particle Swarm Optimization (PSO) [4, 5], for addressing larger and more complex scenarios. Despite their utility, many of these approaches are still too simplistic, as they omit real-world constraints, such as truck storage space dimensions and capacity, goods loading layouts, the necessity and cost of goods loading, unloading, re-arrangement, drivers and truck costs, and meeting customer order contractual terms, etc.

To take into account some of the loading problems, researchers investigated the Capacitated Vehicle Routing Problem (CVRP), which incorporates basic vehicle capacity limits under the assumption that goods can be loaded as long as the total weight or the volume of goods does not exceed truck capacity. Exact methods have been proposed to solve CVRP, such as the branch-and-cut-and-price algorithm developed by Fukasawa et al. [6] and its extensions in subsequent studies [7, 8]. However, these methods are limited to small- and medium-scale supply chain problems. In addition, heuristic and metaheuristic approaches have been widely explored and compared across diverse transportation and routing applications [9]. Two-stage solutions have also been studied, in which customers orders are first clustered, and routes are then optimized within each cluster [10, 11]. Nevertheless, such CVRP models still fail to account for the physical dimensions or spatial placement of goods within vehicles, limiting their practical applicability in industrial contexts.

A major step toward operational realism was made in 2004 with the introduction of the CVRP with two-dimensional loading constraints (2L-CVRP) [12]. This variant, along with its three-dimensional counterpart (3L-CVRP), explicitly takes into account the physical feasibilities of truck storage dimensions and positions of items in vehicles. It also takes care of handling goods loading sequences and placements. Examples of handled constraints include orthogonal loading (items are placed to align with vehicles to save space), sequential loading (items are placed Fist In Last Out (FILO) based on delivery order so that goods are not blocked when unloading), and orientation restrictions (items are placed in fixed orientations). The complexity of these models has spurred the development of a new wave of diverse optimization strategies. For example, Iori et al. [12] proposed an exact branch-and-cut method for 2L-CVRP. Metaheuristics, such as savings-based Ant Colony Optimization (originally developed for CVRP [13, 14]) combined with simple but efficient packing heuristics, have also been adapted for 2L-CVRP and 3L-CVRP [15, 16]. Nonetheless, many assumptions in these studies still fall short of addressing many real-world complex requirements that may be contradictory to each other.

Building on the foundation of 2L-CVRP, this work extends the problem domain by moving beyond strict sequential loading constraints to improve the overall efficiency of the delivery process. In specific, we consider factors such as routing, delivery time, (un)loading sequences, optimized layout of items in vehicles, and the number of vehicles required. This leads to the formulation of the Multi-Objective Vehicle Routing and Packing Problem (MOVRPP). Our objectives include minimizing the total number of vehicles deployed, reducing overall delivery time, and lowering handling effort during loading and unloading while fulfilling each order.

To solve the MOVRPP, we propose the Swarm Intelligence-Based Multi-Objective Vehicle Routing and Packing (SIB-MOVRP) algorithm. This method integrates the well-established Non-Dominated Sorting Genetic Algorithm II (NSGA-II) [17] within a swarm intelligence framework [19]. The resulting multi-objective framework produces a Pareto front of optimized solutions, allowing firms to balance trade-offs between resource utilization and service speed. Furthermore, we develop a decoding function based on the Priority Heuristic (PH) algorithm [20], which translates abstract loading sequences into practical two-dimensional placement plans, ensuring operational feasibility.

The main contributions of this paper are as follows. First, we formulate the novel Multi-Objective Vehicle Routing and Packing Problem (MOVRPP), extending conventional VRP variants. Second, we propose the SIB-MOVRP algorithm, which integrates NSGA-II within a canonical swarm intelligence framework. Third, we design an advanced decoding function that converts high-quality packing sequences into feasible two-dimensional placement layouts. Finally, we conduct comprehensive experiments across diverse logistical configurations, comparing the proposed method against relevant benchmarks. The results demonstrate the superior performance and flexibility of SIB-MOVRP in providing optimized solutions under different supply chain settings.

The remainder of this paper is organized as follows. Section 2 reviews related work, including the PH algorithm, NSGA-II, and the SIB method. Section 3 formally defines the MOVRPP and its associated constraints and assumptions. Section 4 describes the proposed algorithm, detailing the integration of NSGA-II and the specific operations within the SIB framework. Section 5 presents the experimental setup and results, evaluating the performance of the proposed method against benchmarks. Finally, Section 6 concludes the paper and discusses potential future research directions.

2. Related Works

2.1. Priority Heuristic Algorithm

Zhang originally proposed a recursive heuristic algorithm for the strip rectangular packing problem in 2006 [21]. This algorithm employs a divide-and-conquer approach by partitioning the available space into multiple subspaces for individual packing. When a package is placed within a space, the remaining area is divided into two smaller subspaces—unless the package fully occupies the space. These subspaces are then recursively processed until each is either completely filled by a package or no additional packages can be accommodated. During this process, packages are prioritized using a maximum-area-first rule, which selects the largest remaining package for placement.

In 2016, Zhang refined this approach and introduced the Priority Heuristic algorithm [20]. Instead of relying solely on the maximum-area-first strategy, this method assigns a priority level to each package based on five distinct placement scenarios. The highest priority is given when a package exactly fills the available space, matching both its width and length. Packages that align with only one edge of the space are assigned the second or third priority, depending on the chosen sorting strategy (e.g., non-increasing height or non-increasing width). The fourth priority is given to packages that can be loaded without matching any edge, while the lowest priority applies when no package can be accommodated in the current space.

2.2. Swarm Intelligence Based (SIB) Method

The Swarm Intelligence Based (SIB) method, introduced by Phoa [19], is inspired by two well-established metaheuristic paradigms: the Genetic Algorithm (GA) and Particle Swarm Optimization (PSO). GA simulates biological evolution, generating new solutions through selection, crossover, and mutation of parent solutions in line with Darwin's principle of survival of the fittest [22]. By contrast, PSO models the social behavior of bird flocks, where each particle iteratively updates its position based on its own best-known solution (local best) and the best-known solution identified by the swarm (global best) [23, 24].

Many studies have attempted to combine the global exploration capability of GA with the local refinement strength of PSO [25, 26, 27]. In the SIB method, a tailored MIX operation replaces the traditional GA-style crossover. This operator blends a particle with either its local best or the global best, preserving beneficial search history while leveraging PSO's learning-driven dynamics. Additionally, a Random Jump operation—functionally similar to mutation or reinitialization—is introduced to prevent premature convergence by injecting controlled randomness into the search process.

The algorithm begins with an initial swarm, where each particle represents a feasible solution and maintains its own local best. Simultaneously, the swarm tracks the global best solution across all particles. At each iteration, new particles are generated via MIX operations, followed by a MOVE operation that determines their subsequent positions. If the MIX operation fails to produce an improved solution, the particle undergoes a Random Jump. This iterative process continues until a termination criterion, such as a maximum number of iterations or time limit, is reached.

2.3. Non-Dominated Sorting Genetic Algorithm II (NSGA-II)

The Non-Dominated Sorting Genetic Algorithm II (NSGA-II) [17] is a widely adopted evolutionary algorithm for multi-objective optimization. Unlike traditional single-objective methods, which often rely on weighted sums of objectives and require prior knowledge of their relative importance, NSGA-II simultaneously optimizes multiple conflicting objectives without prespecified weights. It outputs a set of trade-off solutions, collectively known as the Pareto front.

The core strength of NSGA-II lies in its efficient non-dominated sorting procedure, which organizes candidate solutions into hierarchical fronts based on Pareto dominance. A solution belongs to a higher-ranked front if it is not dominated by any solution in lower-ranked fronts; in other words, it performs better in at least one objective and no worse in all others. To preserve diversity within each front, NSGA-II employs a crowding distance metric, which ensures a well-distributed Pareto front and prevents solutions from clustering in narrow regions of the objective space.

By combining non-dominated sorting with the crowding distance mechanism, NSGA-II achieves both convergence toward the true Pareto-optimal front and diversity among solutions, making it highly effective for practical multi-objective optimization problems.

3. Problem Description

This section introduces and defines the novel Multi-Objective Vehicle Routing and Packing Problem (MOVRPP). The MOVRPP extends the 2L-CVRP by relaxing strict sequential loading constraints and introducing two optimization objectives: minimizing the total number of trucks used and minimizing the total delivery time. Importantly, the total delivery time incorporates both the trucks' travel time and the unloading time with associated overheads at each delivery stop, directly reflecting the efficiency of the physical loading arrangement. A suboptimal packing configuration, where some packages obstruct access to others, leads to additional unloading operations and thus increases the overall delivery duration.

The MOVRPP is formulated under the following assumptions:

1. All vehicles (trucks) are identical in size and capacity ; they also travel at the same constant speed.
2. An unlimited number of trucks is available when needed.
3. Loading or unloading a single package requires one time unit (e.g., minute).

In addition, the MOVRPP inherits certain constraints from the 2L-CVRP:

1. **Orthogonal loading:** Packages must be placed with their sides parallel to the truck's container walls.
2. **Item clustering:** All packages belonging to a single customer must be assigned to the same truck.

3.1. Problem Definition and Notation

The MOVRPP is formally defined as follows:

- Let N denote the total number of customers and M the total number of packages.
- The location of customer i is denoted by $\mathbf{c}_i$, $i = 1, \ldots, N$.
- The distance between two locations α and β is $d_{\alpha,\beta}$.
- The number of packages ordered by customer i is m_i, such that $M = \sum_{i=1}^{N} m_i$.
- The set of packages for customer i is $P_i = \{p_{i,j} \mid j = 1, \ldots, m_i\}$, where $p_{i,j}$ represents the j-th package of customer i.
- Each package $p_{i,j}$ is defined by its dimensions: width $w_{i,j}$ (aligned with the truck's width), length $h_{i,j}$ (aligned with the truck's length/driving direction), and bottom-left corner coordinates $(x_{i,j}, y_{i,j})$ within the truck loading space.
- The truck loading space is modeled as an xy-coordinate system, with the bottom-left corner at $(0, 0)$. The truck container dimensions are W (width) and H (length).
- The constant speed of all trucks is s.
- Let K denote the total number of trucks used in a logistics plan.
- For each truck $k = 1, \ldots, K$, its route is denoted by $R_k = \{r_{k,1}, r_{k,2}, \ldots, r_{k,l_k}\}$, where l_k is the number of stops in route k, and $r_{k,i}$ is the i-th stop (customer location) in route k.
- PT_k denotes the set of all packages assigned to truck k.

To account for unloading overheads, a blocking relationship is defined. Package p_a blocks package p_b (denoted as $p_a \# p_b$) if the following three conditions are satisfied:

1. $x_b < x_a < x_b + w_b$ (package p_a horizontally overlaps with p_b),
2. $y_a > y_b + h_a$ (package p_a is positioned "above" p_b, obstructing its unloading path),
3. p_a is scheduled for delivery after p_b.

3.2. Objective Functions

The MOVRPP seeks to optimize two conflicting objectives. The first objective is to minimize the total number of trucks used (K):

$$Minimise\ Z_1 = K \tag{1}$$

The second objective is to minimize the total delivery time, composed of the total travel time (T_{travel}) and the total unloading time with overheads ($T_{unloading}$).

The total travel time is the sum of travel times for all route segments:

$$T_{travel} = \sum_{k=1}^{K} \left(\frac{1}{s} \sum_{i=1}^{l_k-1} d_{r_{k,i}, r_{k,i+1}} \right) \tag{2}$$

The total unloading time incorporates both the base time for unloading each package and additional time for handling blocking packages. Each blocking package requires two minutes (one to unload and one to reload). Thus, for each package $p_{ij} \in PT_k$, its unloading time is:

$$1 + 2 \times \sum_{p_a \in PT_k} I\left(p_a \# p_{ij}\right) \tag{3}$$

where $I(\cdot)$ is an indicator function equal to 1 if the condition holds and 0 otherwise. The total unloading time across all trucks is therefore:

$$T_{unloading} = \sum_{k=1}^{K} \sum_{i=1}^{l_k-1} \sum_{j=1}^{m_{r_i}} \left(1 + 2 \times \sum_{p_a \in PT_k} I\left(p_a \# p_{r_i,j}\right) \right) \tag{4}$$

The second objective function is expressed as:

$$Minimise\ Z_2 = T_{travel} + T_{unloading} \tag{5}$$

3.3. Constraints

The MOVRPP is subject to the following constraints:

1. **Package Placement:** Every package must be placed entirely within the truck's loading space (Equation (6)).

$$\forall a \in \bigcup_{i=1}^{N} P_i, \quad 0 < x_a < W - w_a, \quad 0 < y_a < H - h_a \tag{6}$$

2. **Non-overlapping Packages:** Packages within the same truck must not overlap.
3. **Customer Assignment and Coverage:** All packages belonging to a single customer must be assigned to the same truck. Furthermore, each customer must be visited exactly once across all routes (Equation (7)).

$$\bigcup_{k=1}^{K} R_k = \{i = 1, \ldots, N\} \tag{7}$$

4. Method

Integrating the canonical framework of SIB and the NSGA-II algorithm, we propose the Swarm Intelligence-Based Method for Multi-Objective Vehicle Routing and Packing Problems (SIB-MOVRP). Each particle in the algorithm represents a valid logistics plan, including detailed packing and routing plans. Algorithm 1 outlines the SIB-MOVRP algorithm. First, a swarm consisting of several particles is initialized and evaluated. In this two-objective problem, each particle has two Local Bests (LBs) corresponding to the two objectives; similarly, the swarm has two Global Bests (GBs). Moreover, the initial Pareto front repository, abbreviated as the repository, is determined through non-dominated sorting among the initial particles. If the number of non-dominated particles exceeds the user-defined maximum repository size, the particles are selected using the grid selection method to maintain diversity among optimized solutions. This procedure is applied in every repository update of our algorithm.

This algorithm is iterative, with each iteration consisting of one MIX and one MOVE operation. By mixing each particle with its LBs and GBs, four new modified particles are generated. A rearranging function is then applied to these modified and original particles to optimize their internally

Algorithm 1 Outline of SIB-MOVRP

Input: N (Size of Population), M (Size of Repository), Maxgen (Number of Generation)

Output: REP (Pareto Front Repository)

$POP \leftarrow$ Generate Initial Population (N)
$LB_{obj1}, LB_{obj2} \leftarrow POP$
$GB_{obj1} \leftarrow$ The best particle among POP based on objective 1
$GB_{obj2} \leftarrow$ The best particle among POP based on objective 2
$REP \leftarrow$ Determine_repository(POP)
$gen = 1$
while $gen < Maxgen$ **do**
    for each particle P in POP **do**
        $m_1 \leftarrow$ Rearranging(P)
        $m_2 \leftarrow$ Rearranging(MIX(P, LB_{obj1}))
        $m_3 \leftarrow$ Rearranging(MIX(P, LB_{obj2}))
        $m_4 \leftarrow$ Rearranging(MIX(P, GB_{obj1}))
        $m_5 \leftarrow$ Rearranging(MIX(P, GB_{obj2}))
        $Next \leftarrow$ Choose the best among $m_{1\sim5}$
        if $Next$ is better than P (determined based on rank) **then**
            $P \leftarrow Next$
        else
            $P \leftarrow$ Random Jump($Next$)
        end if
    end for
    $REP \leftarrow$ Determine_repository($POP + REP$)
    $LBs, GBs \leftarrow$ Update_best(POP)
    $gen = gen + 1$
end while
return (REP)

specified packing plans. These five rearranged particles are evaluated using the non-dominated sorting algorithm to determine the best particle as the next position of the current particle. In cases where the rearranged original particle is the best among the five, a Random Jump operation is executed to prevent the algorithm from prematurely converging to a local optimum. This operation partially and randomly modifies the original particle to generate the next position, regardless of whether it is better than the original position.

After the MOVE operation, a new set of optimized particles is generated. These particles are then added to the current repository, where the non-dominated sorting function and grid selection are applied to select the particles to be preserved. At the end of each iteration, the LBs and GBs are updated according to the particles' new positions and objective values. The algorithm terminates when predefined stopping criteria are fulfilled. At completion, the particles stored in the repository represent the desirable optimized logistics plans based on the multiple objectives.

4.1. Particle Definition

A particle represents a feasible logistics plan, consisting of the number of trucks used, routing information, and a valid packing plan for each truck. The packing plan specifies the loading sequence of assigned packages, which determines the spatial placement of each item within the truck. The routing plan indicates the set of customer orders assigned to each truck and the corresponding delivery route. Once the customer orders (and their delivery locations) are assigned, the routes are automatically generated based on a nearest-first rule, where each truck departs from the depot and sequentially visits the closest unvisited delivery location.

4.2. Decoding function

While the packing plan only records the sequence of packages, a decoding function is required to transform it into a placement plan specifying the position of each package within the truck. This decoding function is adapted from the recursive packing algorithm proposed by Zhang [21], with a key difference: instead of following the maximum-area-first rule, packages are processed strictly according to the loading sequence encoded in each particle. During decoding, a list of empty spaces is maintained and continuously updated. This list is sorted by the width of each space, where width refers to the left-to-right direction, and length refers to the front-to-back direction of the truck. For each package, the algorithm scans the list to find the first space large enough to accommodate it.

Inspired by Zhang's PH algorithm [20], this function considers eight placement scenarios (Figure 9). Packages are placed horizontally in scenarios (a), (b), (e), and (g), and vertically in the others. These scenarios are applied in a predefined order, moving to the next only if the current scenario is infeasible. If no suitable space is available, the package is placed at a new level, i.e., behind all previously placed packages. Unless the package fits perfectly into the space (as in scenarios (a) and (b)), each placement generates one or two new empty spaces, which are then added to the list.

After each placement, the list of empty spaces is updated according to two rules:

1. Two adjacent spaces are merged if the newly created space (e.g., space D in Figure 12) is at least as wide as its neighboring space (e.g., space C).
2. If an existing space is entirely blocked by the newly placed package, it is removed from the list.

Figure 15 illustrates the placement process of four packages. In Figure 10, the first package is placed, generating space A. The second package is placed into space A according to scenario (g), which replaces A with two smaller empty spaces, B and C (Figure 11). Since no existing space is large enough for the third package, it is placed at a new level, generating space D (Figure 12). According to the first updating rule, spaces C and D are merged into a larger space E (Figure 14). The final package spans the entire width of the truck, thereby blocking all remaining spaces, which are removed according to the second updating rule.

4.3. Initialization

The initial swarm is generated with a predetermined number of particles. To initialize a particle, orders are sequentially assigned to a truck at random until it reaches its capacity. Once a truck is full, the process continues with the next unused truck, repeating until all orders are allocated. During this process, the PH algorithm [2, 20] is employed to ensure that the package assignment of each truck is valid, i.e., the assigned packages do not exceed the truck's allowable space. For each truck, given its assigned orders and their destinations, a delivery route is constructed by prioritizing destinations based on proximity, starting from the depot. The performance of each particle is

Figure 1: (a) $X = W$ and $Y = L$

Figure 2: (b) $X < W$ and $Y = L$

Figure 3: (c) $X = L$ and $Y = W$

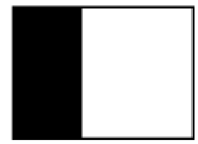

Figure 4: (d) $X = L$ and $Y < W$

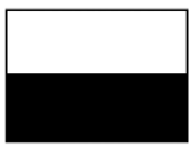

Figure 5: (e) $X = W$ and $Y < L$

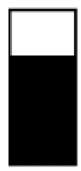

Figure 6: (f) $X < L$ and $Y = W$

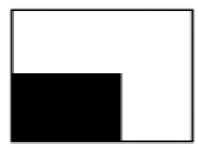

Figure 7: (g) $X < W$ and $Y < L$

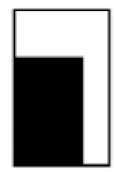

Figure 8: (h) $X < W$ and $Y < L$

Figure 9: Eight scenarios for placing a package into an empty space. X denotes the longer edge of the package, and Y denotes the shorter edge. W and L represent the width and length of the empty space, respectively.

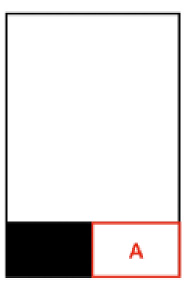

Figure 10: (a)

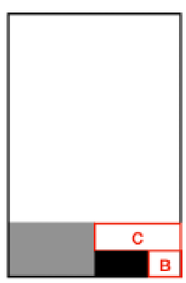

Figure 11: (b)

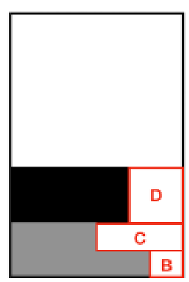

Figure 12: (c)

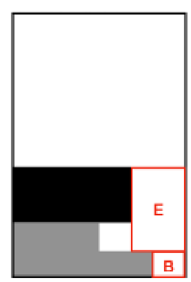

Figure 13: (d)

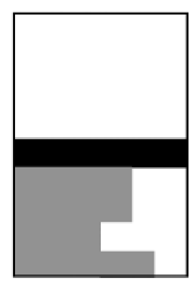

Figure 14: (e)

Figure 15: Illustration of the placement process and the two rules for updating empty spaces. Black rectangles represent the package placed in the current step, while grey ones represent previously placed packages. Red-bordered areas indicate empty spaces. Rule (1) is applied in (d), where adjacent spaces C and D are merged into a larger space E. Rule (2) is applied in (e), where all empty spaces are removed because they are completely blocked.

evaluated according to the number of trucks used and the total delivery time. With these initial particles and their objective values, the initial repository, the LBs, and the GBs are determined.

4.4. MIX Operation

A newly designed MIX operation is proposed to leverage the information stored in the LBs and GBs to generate improved particle positions. This operation consists of two components: MIX-1, which optimizes the routing plan (i.e., package assignment), and Rearranging, which enhances the packing plans of the involved trucks. In each MIX operation, the current particle is mixed with its two LBs and the two GBs, yielding four modified particles. Each modified particle is generated through three consecutive MIX-1 operations followed by one Rearranging operation. In addition, the current particle undergoes a Rearranging operation before the MOVE operation to improve its packing efficiency.

4.4.1. MIX-1

In this operation, a particle is mixed with one of its LBs or one of its GBs to generate a modified particle. Let the current particle be denoted as X, and the best particle as B. The algorithm first identifies the worst truck (T1) in X, defined as the truck with the lowest space utilization, where space utilization is calculated as the total area of the packages assigned to the truck. Next, the algorithm extracts the set of orders (O) delivered by T1 and identifies the truck(s) in B that also deliver any of these orders. Among these trucks in B, the one that delivers the largest number of orders in O is selected as T2. If multiple trucks satisfy this condition equally, one is randomly chosen.

The orders in truck T2 are then randomly selected and reassigned to T1, one by one, until T1 reaches its capacity. If any truck becomes empty during this reassignment, it is removed from the particle's logistics plan. This process effectively improves the space utilization of T1 while preserving its delivery scope and leveraging information from a high-performing solution (B).

Order No.	0	1	2	3	4	5	6	7	8	9
X	1	1	2	3	3	2	3	2	1	1
B	1	2	2	1	1	3	3	1	1	2
New	1	1	2	3	3	2	3	3	3	1

Table 1: An illustrative example of the MIX-1 operation.

Given the package assignment plan in Table 1, both X and B use three trucks to deliver nine orders (0–8). In the following explanation, truck n in X is denoted as Xn, and truck m in B as Bm. Note that the truck number does not refer to a specific truck; even if n equals m, Xn and Bm do not necessarily indicate the same truck.

Assume that X3 has the lowest space utilization in X (T1), delivering orders 3, 4, and 6 (O). These orders are packed in B1 and B3, while B2 is the truck that delivers the largest number of orders in O, and is therefore selected as T2. Among the orders in B2 (T2) that are not assigned to X3 (T1) are orders 0, 7, and 8, which become the target orders to potentially move to X3. These target orders are considered one by one in a random order (e.g., 7–0–8). Assume that after moving order 7 from X2 to X3, there is insufficient space to place order 0. In this case, order 0 is skipped, and the algorithm proceeds to examine whether order 8 can be moved. After all target orders have been examined, regardless of whether each move is successful, the logistics plan of the modified particle is finalized. If order 8 can be successfully moved from X1 to X3 after moving order 7, the resulting solution corresponds to the New particle shown in Table 1.

4.4.2. Rearranging

Rearranging is a minor operation intended to refine the packing plans for a particle after the MIX operation. The algorithm evaluates the unloading time and maneuvering overheads for each package to identify the most blocked one. During each execution, the package most obstructed by others in each truck is moved to the end of the packing sequence, i.e., repositioned closest to the truck's rear doors. Figure 18 provides an illustrative example of this operation, where package 1 is moved to the end of the packing sequence.

4.5. MOVE Operation

This operation determines the next position of the current particle by comparing the performance of four modified particles and the rearranged current particle. As in repository construction, non-dominated sorting is applied to select the best candidate among the five. However, when none of the modified particles outperforms the rearranged current particle, a Random Jump operation is triggered on the original, i.e., non-rearranged, particle to avoid premature convergence.

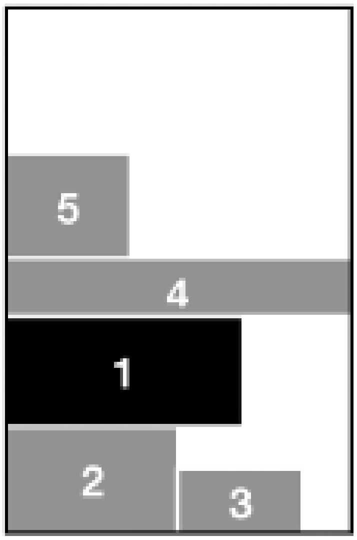

Figure 16: (a) Original packing sequence: 2-3-1-4-5

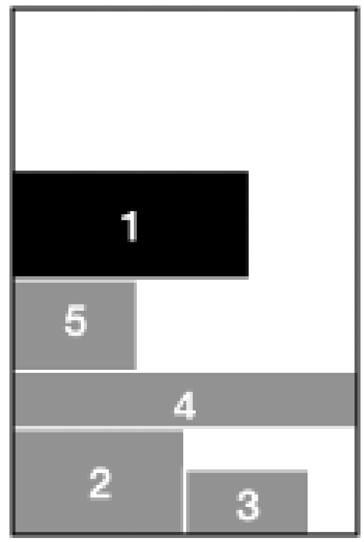

Figure 17: (b) Rearranged packing sequence: 2-3-4-5-1

Figure 18: An illustrative example of the Rearranging operation.

4.5.1. Random Jump

This operation is designed to help a particle escape from a local optimum and continue exploring the solution space. The algorithm randomly selects an order (denoted as Oj) from a randomly chosen truck and moves it to a new empty truck (Tn), thereby increasing the total number of trucks used by one. The algorithm then identifies "neighboring orders" based on geographical proximity to Oj and moves them to Tn one by one until the truck is full. Similar to the MIX-1 operation, if any truck becomes empty during this process, it is removed from the plan.

An illustrative example is provided in Table 2. Assume order 5 is selected as Oj and its neighboring orders (sorted from nearest to farthest) are 6–7–9–1–0–2–8–3–4. Order 5 is first moved to a new empty truck X4. The algorithm then attempts to move the neighboring orders sequentially. If an order cannot be placed (e.g., order 7 after placing order 6), it is skipped, and the next neighboring order is considered. In the example, the process continues until order 9 is successfully moved, filling truck X4. The result of this operation is taken as the next position of the current particle, regardless of its performance.

Order No.	0	1	2	3	4	5	6	7	8	9
X	1	1	2	3	3	2	3	2	1	1
New	1	1	2	3	3	4	4	2	1	4

Table 2: An illustrative example of the Random Jump operation.

5. Experiments

To demonstrate the robustness of the newly proposed SIB-MOVRP method, multiple datasets with varying scales and configurations were deployed across three realistic industrial scenarios. These configurations were designed to support a comprehensive evaluation of the algorithm's adaptability to diverse logistical challenges. To provide a meaningful performance comparison between our proposed approach and a commonly used human/manual-based method, we implemented a deterministic approach, referred to as the *Distance-First Heuristic*, to simulate human behaviors. This baseline human-based approach employs a rule-based planning strategy that has been widely used in real-world logistics operations. Its main objectives are to minimize routing distance and delivery cost, given requirements such as customer orders, vehicle availability, and capacity constraints.

5.1. Data Generation

The datasets used in this study are synthetically generated, and the process for each instance follows these steps:

1. **Customer Locations:** On a 100-by-100 map, customer locations are randomly assigned to simulate urban density. Specifically, half of the customers are uniformly distributed within the central box [30,70], while the remaining half are uniformly distributed across the entire map, resulting in a higher concentration of customers in the central area.
2. **Package Orders:** Given a total number of packages for an instance, the number of packages ordered by each customer is determined such that each customer orders between one and five packages.
3. **Package Size:** Package dimensions (length and width) are generated from uniform distributions, with ranges specified in Table 3 for each configuration type.

4. **Depot Location:** The depot location is manually chosen, typically situated in the city's suburbs, reflecting common distribution center placements.

Name	**Range of package size (meters)**	
	The Shorter Edge	**The Longer Edge**
Grocery (G)	[0.5, 1)	[0.5, 1.5)
Furniture (F)	[0.5, 1.5)	[1.5, 2.5)
One Large (OL) – Small package	[0.5, 1)	[0.5, 1.5)
One Large (OL) – Large package	[1, 2)	[2, 3)

Table 3: Ranges of package dimensions (length and width) used for generating instances across the three logistical configurations.

The synthetically generated datasets comprise three distinct configurations, designed to simulate real-world logistical scenarios, as shown in Table 3. The *Grocery (G)* configuration represents small-package delivery scenarios commonly observed in urban supply chains, such as those of Tesco, Walmart, and LIDL, where efficient management of frequent deliveries with numerous small items is critical. The *Furniture (F)* configuration simulates the transport of large and often irregularly shaped items, reflecting supply chains like IKEA and Wayfair, which require careful consideration of spatial constraints and variable package dimensions in truck loading and routing.

The *One Large (OL)* configuration is introduced to evaluate the algorithm's performance under extreme conditions. In this scenario, each order contains one exceptionally large package accompanied by smaller items, posing unique challenges for space utilization and order grouping. This configuration reflects specialized delivery cases such as those of Best Buy (large home appliances) or GE Healthcare (bulky medical equipment), where logistics must accommodate heterogeneous order structures.

5.2. Baseline Method – Distance-First Heuristic

Without computer assistance, companies often rely on manual planning guided by deterministic heuristics. One of the simplest and most intuitive approaches is the Distance-First Heuristic, which assigns a truck to the order nearest to its current location until the truck reaches capacity. This decision-making process is implemented as the baseline in our experiments. Similar to the initialization of a particle in SIB-MOVRP, the logistics plan is constructed by filling trucks sequentially. Each truck starts from the depot and proceeds to the nearest destination, subject to truck space constraints. As an order is assigned to the current truck, package placement is generated using the PH algorithm to ensure feasibility. Once a truck is full, i.e., no further orders can be loaded, a new truck is added to the plan. The process repeats until all orders are assigned.

5.3. Results and Discussion

Tables 4 and 5 summarize the results under three configurations, comparing both the baseline and the optimized outcomes in terms of the number of trucks and the total delivery time. The results show that SIB-MOVRP consistently outperforms the baseline method by reducing delivery time and optimizing truck utilization. These improvements hold across supply chains of varying scales, demonstrating the adaptability of the proposed method regardless of the number of orders or package sizes.

For example, under the Grocery configuration with 20 orders and 50 packages (Instance 2), SIB-MOVRP reduced delivery time by 32.1% compared to the baseline (from 545.79 to 370.71 minutes), while using the same number of trucks. In Instance 7, the method reduced the number of trucks by one and shortened delivery time from 679.99 to 615.13 minutes, showing advantages in both cost and efficiency. The method also demonstrated effectiveness in the Furniture and One Large configurations. For example, delivery time was reduced by 8.7% in Instance 11 and by 18.8% in Instance 13, with the same number of trucks used.

Overall, SIB-MOVRP achieves superior results across all tested configurations, ranging from small-item deliveries to large-scale furniture transport and mixed-size orders, highlighting its adaptability to diverse logistical scenarios. Selected optimized logistics plans are illustrated in Figure 26, providing visual insight into how route structures and loading patterns differ from the baseline. According to the packing plans (Figure 24 and 25), our method achieves shorter delivery times by distributing packages more evenly across trucks and routes.

In addition to generating solutions that outperform the baseline, the proposed method produces a set of feasible solutions rather than a single fixed plan. These solutions differ in the number of trucks used and the total delivery time, enabling more informed and adaptable decision-making. Unlike traditional approaches that impose a predefined trade-off between delivery time and truck number, the proposed method allows users to select the most

No.	Order Number	Total Package Number	Baseline		SIB-MOVRP	
			Truck Number	Delivery Time (mins)	Truck Number	Delivery Time (mins)
1	20	50	3	397.69	3	319.11
					4	318.25
2	20	80	5	545.79	5	369.50
					6	348.75
					7	340.74
3	20	100	7	404.32	7	384.44
					8	381.68
4	30	50	3	559.97	3	478.20
					4	452.30
5	30	80	5	565.45	5	536.39
					6	451.20
6	30	120	8	679.42	8	598.32
					9	586.25
					10	581.78
7	40	50	4	679.99	3	615.13
					4	538.61
					5	529.99
8	40	80	5	793.53	5	747.63
					6	747.55
9	40	120	8	721.28	8	674.08
					10	670.71

Table 4: Results of experiments on the *Grocery* configuration using Baseline method and the proposed SIB-MOVRP method.

suitable plan based on specific operational requirements. This flexibility is particularly valuable in real-world logistics, where companies may prioritize different factors such as cost, delivery speed, or resource availability. For example, when trucks are rented, minimizing the number of trucks may be the most cost-effective option. Conversely, when delivering perishable or frozen goods, a plan with shorter delivery time may be preferable, even if it requires more trucks.

No.	Type	Order Number	Total Package Number	Baseline		SIB-MOVRP	
				Truck Number	Delivery Time (mins)	Truck Number	Delivery Time (mins)
10	F	20	50	10	511.34	10	479.01
11	F	20	80	15	654.03	15	597.39
12	OL	20	50	9	527.04	9	486.37
13	OL	20	80	5	414.79	5	336.81
						6	318.91

Table 5: Results of experiments on the *Furniture* and *One Large* configuration using Baseline method and the proposed SIB-MOVRP method.

6. Conclusion

This paper introduced and formulated the novel Multi-Objective Vehicle Routing and Packing Problem (MOVRPP), extending existing Vehicle

Routing Problem (VRP) variants by incorporating realistic two-dimensional packing constraints and optimizing for two conflicting objectives: minimizing the total number of trucks used and minimizing total delivery time, including unloading overheads arising from sub-optimal package arrangements, which are often overlooked in traditional models.

To address the MOVRPP, the Swarm Intelligence-Based Multi-Objective Vehicle Routing and Packing (SIB-MOVRP) algorithm was proposed. This algorithm integrates the strengths of NSGA-II within a canonical SIB framework and employs a sophisticated Priority Heuristic-based decoding function to ensure practical and feasible packing solutions.

Comprehensive experimental evaluations demonstrated SIB-MOVRP's superior performance and robustness across diverse logistical scenarios, including *Grocery* (small-package deliveries), *Furniture* (large-item transport), and *One Large* (mixed-size orders). Compared with a deterministic Distance-First heuristic baseline, it consistently achieved significantly improved solutions in terms of both truck utilization and total delivery time. Moreover, the multi-objective design of SIB-MOVRP generates a Pareto front of optimized solutions, providing decision-makers with flexible trade-off options tailored to dynamic operational priorities. This capability offers a more realistic and adaptable framework for complex logistics, enabling informed, strategically effective decisions that deliver substantial cost savings and enhanced customer satisfaction in real-world operations.

Future research directions include extending the MOVRPP to incorporate three-dimensional packing constraints, thereby further enhancing real-world applicability; exploring dynamic routing scenarios where new orders or disruptions occur during delivery; and conducting larger-scale case studies using real-world industrial data to validate the practical impact of the SIB-MOVRP algorithm.

Acknowledgements

Include any individuals who provided you with help during your research, such as help with language, writing or proof reading, in the acknowledgements section. Acknowledgements should be placed in a separate section which appears directly before the reference list. Do not include acknowledgements on your title page, as a footnote to your title, or anywhere else in your article other than in the separate acknowledgements section.

Author contributions

Conceptualization – Ideas; formulation or evolution of overarching research goals and aims.

.

.

.

Funding

This is a temp

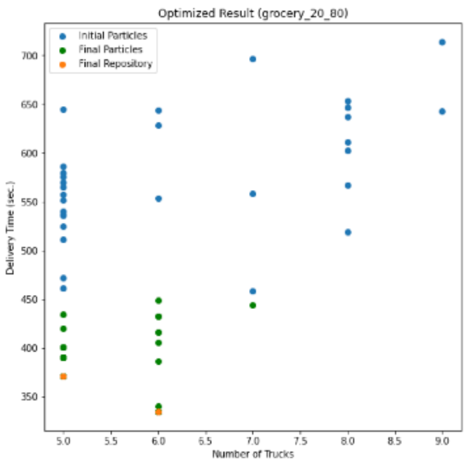

Figure 19: (a)

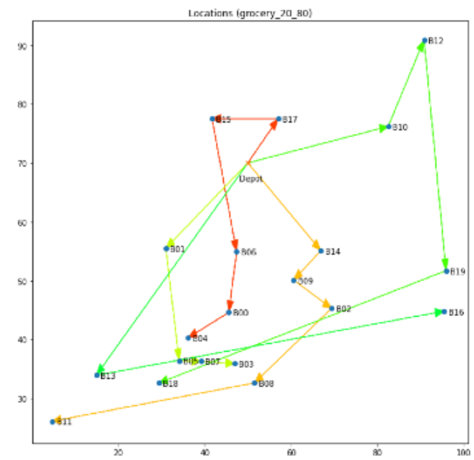

Figure 20: (b) Baseline, five trucks

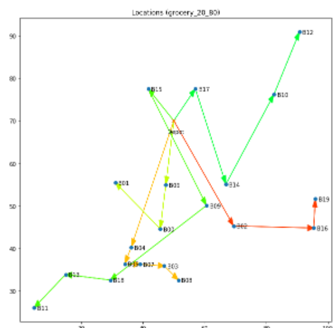

Figure 21: (c) SIB-MOVRP, five trucks

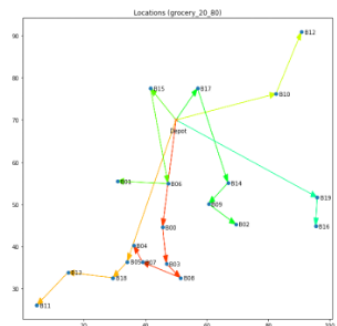

Figure 22: (d) SIB-MOVRP, six trucks

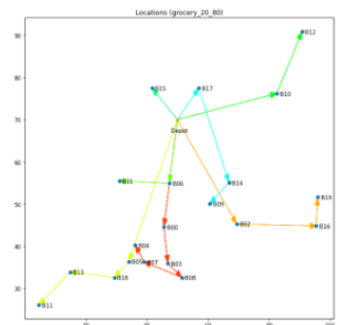

Figure 23: (e) SIB-MOVRP, seven trucks

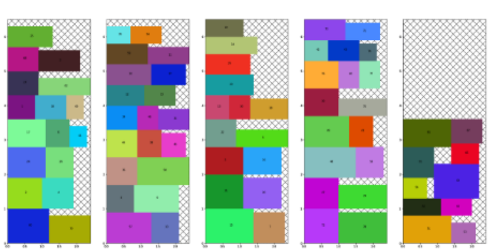

Figure 24: (f) Baseline

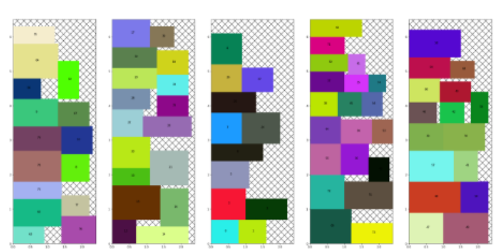

Figure 25: (g) SIB-MOVRP

Figure 26: Logistics plan visualization for instance 2 (under the *Grocery* configuration). Subfigure (a) illustrates the swarm's convergence process and the final repository. Subfigure (b) shows the routing plan generated by the baseline method, while (c), (d), and (e) depict the routing plans generated by SIB-MOVRP using five, six, and seven trucks, respectively. The corresponding packing plans for the five-truck solutions are shown in (f) for the baseline and (g) for SIB-MOVRP.

References

[1] Desrochers, M., Lenstra, J., Savelsbergh, M., Soumis, F., *Vehicle routing with time windows: Optimization and approximation*, Veh Rout Methods Stud, 16, 1988.

[2] Baker, B. M., Ayechew, M., *A genetic algorithm for the vehicle routing problem*, Computers & Operations Research, 30(5), 787–800, 2003. https://doi.org/10.1016/S0305-0548(02)00051-5

[3] Marańda, W., *Selected Genetic Algorithms for Vehicle Routing Problem Solving*, Electronics, 10(24), 3147, 2021. https://doi.org/10.3390/electronics10243147

[4] Ai, T. J., Kachitvichyanukul, V., *A particle swarm optimization for the vehicle routing problem with simultaneous pickup and delivery*, Computers & Operations Research, 36(5), 1693–1702, 2009. https://doi.org/10.1016/j.cor.2008.04.003

[5] Shen, H., Zhu, Y., Liu, T., Jin, L., *Particle Swarm Optimization in Solving Vehicle Routing Problem*, 2009 Second International Conference on Intelligent Computation Technology and Automation, Volume 1, 287–291, 2009. https://doi.org/10.1109/ICICTA.2009.77

[6] Fukasawa, R., Longo, H., Lysgaard, J., Aragão, M. P., Reis, M., Uchoa, E., Werneck, R. F., *Robust Branch-and-Cut-and-Price for the Capacitated Vehicle Routing Problem*, Mathematical Programming, 106(3), 491–511, 2006. https://doi.org/10.1007/s10107-005-0644-x

[7] Pessoa, A., de Aragão, M. P., Uchoa, E., *Robust Branch-Cut-and-Price Algorithms for Vehicle Routing Problems*, In: Golden, B., Raghavan, S., Wasil, E. (eds), The Vehicle Routing Problem, Springer, 297–325, 2008. https://doi.org/10.1007/978-0-387-77778-8_14

[8] Røpke, S., *Branching decisions in branch-and-cut-and-price algorithms for vehicle routing problems*, Presentation in Column Generation, 2012.

[9] Muriyatmoko, D., Djunaidy, A., Muklason, A., *Heuristics and Metaheuristics for Solving Capacitated Vehicle Routing Problem: An Algorithm Comparison*, Procedia Computer Science, 234, 494–501, 2024. https://doi.org/10.1016/j.procs.2024.03.032

[10] Chen, M.-H., Chang, P.-C., Chiu, C.-Y., Annadurai, S. P., *A hybrid two-stage sweep algorithm for capacitated vehicle routing problem*, 2015 International Conference on Control, Automation and Robotics, 195–199, 2015. https://doi.org/10.1109/ICCAR.2015.7166030

[11] Qi, C., Cui, S., Sun, Y., *A Two-Stage Hybrid Ant Colony Algorithm for the CVRP*, 2008 International Conference on Computational Intelligence and Security, 215–219, 2008. https://doi.org/10.1109/CIS.2008.209

[12] Iori, M., Salazar-Gonzalez, J.-J., Vigo, D., *An Exact Approach for the Vehicle Routing Problem with Two-Dimensional Loading Constraints*, Transportation Science, 41(2), 253–264, 2007.

[13] Reimann, M., Doerner, K., Hartl, R. F., *D-Ants: Savings Based Ants divide and conquer the vehicle routing problem*, Computers & Operations Research, 31(4), 563–591, 2004. https://doi.org/10.1016/S0305-0548(03)00014-5

[14] Reimann, M., Stummer, M., Doerner, K., *A Savings Based Ant System for The Vehicle Routing Problem*, Proc. Conf., 1317–1326, 2002.

[15] Fuellerer, G., Doerner, K. F., Hartl, R. F., Iori, M., *Ant colony optimization for the two-dimensional loading vehicle routing problem*, Computers & Operations Research, 36(3), 655–673, 2009. https://doi.org/10.1016/j.cor.2007.10.021

[16] Fuellerer, G., Doerner, K. F., Hartl, R. F., Iori, M., *Metaheuristics for vehicle routing problems with three-dimensional loading constraints*, European Journal of Operational Research, 201(3), 751–759, 2010. https://doi.org/10.1016/j.ejor.2009.03.046

[17] Deb, K., Pratap, A., Agarwal, S., Meyarivan, T., *A fast and elitist multiobjective genetic algorithm: NSGA-II*, IEEE Transactions on Evolutionary Computation, 6(2), 182–197, 2002. https://doi.org/10.1109/4235.996017

[18] Phoa, F.K.H., Chen, R.B., Wang, W., Wong, W.K., *Optimizing two-level supersaturated designs using swarm intelligence techniques*, Technometrics, 58(1), 43-40, 2016. https://doi.org/10.1080/00401706.2014.981346

[19] Phoa, F.K.H., *A swarm intelligence based (sib) method for optimization in designs of experiments*, Natural Comput., 16, 597–605, 2017. https://doi.org/10.1007/s11047-016-9555-4

[20] Zhang, D., *A priority heuristic for the guillotine rectangular packing problem*, Information Processing Letters, 2016.

[21] Zhang, D., Kang, Y., Deng, A., *A new heuristic recursive algorithm for the strip rectangular packing problem*, Computers & Operations Research, 33(8), 2209–2217, 2006. https://doi.org/10.1016/j.cor.2005.01.009

[22] Holland, J., *Adaptation in Natural and Artificial Systems*, MIT Press, Cambridge, MA, 1975.

[23] Kennedy, J., Eberhart, R., *Particle swarm optimization*, Proc. IEEE Int. Conf. Neural Netw., 4, 942–1948, 1995. https://doi.org/10.1109/ICNN.1995.488968

[24] Kim, T., Maruta, I., Sugie, T., *A simple and efficient constrained particle swarm optimization and its application to engineering design problems*, Proc. IMechE Part C, 224, 389–400, 2010. https://doi.org/10.1243/09544062JMES1732

[25] Roy, C., Das, D. K., *A hybrid genetic algorithm (GA)–particle swarm optimization (PSO) algorithm for demand side management in smart grid considering wind power for cost optimization*, Sādhanā, 46(2), 101, 2021. https://doi.org/10.1007/s12046-021-01626-z

[26] Kuo, R. J., Han, Y. S., *A hybrid of genetic algorithm and particle swarm optimization for solving bi-level linear programming problem – A case study on supply chain model*, Applied Mathematical Modelling, 35(8), 3905–3917, 2011. https://doi.org/10.1016/j.apm.2011.02.008

[27] Han, F., Fu, H., Wang, B., Xu, Y., Lv, B., *GP4ESP: A hybrid genetic algorithm and particle swarm optimization algorithm for edge server placement*, PeerJ Computer Science, 10, e2439, 2024. https://doi.org/10.7717/peerj-cs.2439