IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Journal Européen des Systèmes Automatisés
Vol. 58, No. 8, August, 2025, pp. 1745-1756
Journal homepage: http://iieta.org/journals/jesa

Enhanced Particle Swarm Optimization for Dual-Constraint Multi-Knapsack Problems: A Real-World Retail Packaging Application

Ifan Rizqa[1*], Abdul Syukur[2], Nova Rijati[2], Aris Marjuni[2]

[1] Doctor of Computer Science, Universitas Dian Nuswantoro, Semarang 50131, Indonesia
[2] Department of Informatics Engineering, Faculty of Computer Science, Universitas Dian Nuswantoro, Semarang 50131, Indonesia

Corresponding Author Email: risqa.ifan@dsn.dinus.ac.id

https://doi.org/10.18280/jesa.580818

Received: 2 July 2025
Revised: 4 August 2025
Accepted: 20 August 2025
Available online: 31 August 2025

Keywords:
Particle Swarm Optimization (PSO), Multi-Constraint Multi-Knapsack Problem (MCMKP), retail packaging optimization, constraint handling mechanism, hybrid value function, swarm intelligence

ABSTRACT

The Multi-Constraint Multi-Knapsack Parcel Optimization Problem (MCMKPOP) arises in retail logistics where products must be allocated to multiple packages under simultaneous weight and budget constraints. Conventional methods, including exact approaches and traditional metaheuristics, often address these constraints separately or rely on synthetic datasets, scalability and practical relevance. This study proposes an enhanced Particle Swarm Optimization (PSO) framework that introduces three key innovations: an assignment-based representation to encode discrete allocations, a repair-based constraint mechanism to ensure feasibility, and a hybrid value function that integrates price, weight, and efficiency trade-offs. Using a real-world dataset of 213 retail items and ten heterogeneous packaging configurations, the proposed approach was evaluated against Genetic Algorithm, Simulated Annealing, and Random Search. Experimental results from 30 independent runs show that PSO achieves 100% of the theoretical upper bound with a mean performance of 21.3083 ± 0.3145, balanced resource utilization (> 90%), and zero constraint violations. Statistical validation through parametric and non-parametric tests confirmed the robustness and significance of these findings. The novelty this work unified treatment of dual constraints, a multi-knapsack-aware upper bound, and validation on real-world data, establishing PSO as both a methodological advance and a practical solution for intelligent retail packaging optimization.

1. INTRODUCTION

1.1 Research background

The Multi-Constraint Multi-Knapsack Problem (MCMKP) is an extension of the classical knapsack problem that involves allocating items into multiple containers subject to diverse capacity constraints. with applications in logistics, supply chain management, scheduling, cloud computing, telecommunications, and retail packaging [1, 2]. In retail operations, the MCMKP manifests as the problem of assigning items with heterogeneous price and weight to multiple packaging containers, where both budget and physical capacity constraints must be satisfied while maximizing customer satisfaction and revenue [3].

Over the past two decades, extensive efforts have been dedicated to solving knapsack variants. Exact methods, such as branch-and-bound, column generation, and dynamic programming, guarantee optimality but face severe scalability limitations as the number of items and constraints increase [2, 4]. Although improvements in decomposition strategies and mathematical programming have extended the solvable instance size, their exponential time complexity renders them impractical for real-time or large-scale applications [5, 6].

As a result, heuristic and metaheuristic methods have emerged as viable alternatives, capable of delivering high-quality solutions within reasonable computational budgets. Studies have explored evolutionary algorithms, greedy constructive heuristics, and local search strategies to tackle large-scale and multi-dimensional knapsack problems [7, 8]. Hybrid approaches that integrate exact and heuristic techniques have also been proposed to balance solution accuracy with computational efficiency. Nonetheless, these methods often struggle to adapt to highly heterogeneous problem instances, where both resource constraints and item characteristics vary significantly across bins [9, 10].

1.2 Swarm intelligence and Particle Swarm Optimization

Swarm intelligence algorithms, inspired by collective behaviors in nature, have become powerful tools for combinatorial optimization. Among them, Particle Swarm Optimization (PSO), originally proposed by Kennedy and Eberhart [11], has gained increasing attention. While PSO was initially designed for continuous optimization problems, significant research has been devoted to adapting it for discrete and combinatorial domains. PSO’s strengths lie in its population-based mechanism, its balance of exploration and exploitation, and its ability to efficiently navigate multimodal search landscapes.

Recent advances demonstrate that discrete and hybrid versions of PSO can achieve competitive results for multidimensional and multi-objective knapsack problems. For example, discrete PSO variants have been successfully applied to resource allocation, scheduling, and large-scale distributed optimization [12]. Furthermore, recent developments in multi-objective PSO emphasize enhanced convergence and diversity preservation in multimodal landscapes [13, 14]. Comparative studies indicate that swarm-based methods often outperform traditional evolutionary algorithms such as Genetic Algorithm (GA) or Simulated Annealing (SA), particularly in terms of convergence speed and solution robustness [15, 16].

Despite these encouraging results, there remain notable limitations in applying PSO to real-world knapsack problems. Most PSO adaptations target single-constraint formulations or simplified instances. They often neglect the interdependence of dual constraints, such as budget and weight, which are critical in practical retail and logistics settings. Moreover, while algorithmic performance is frequently reported in terms of best or mean fitness values, many studies fail to establish robust statistical validation, leaving uncertainty about the generalizability and reliability of the results [17, 18].

1.3 Research gaps and practical motivations

A closer examination of the literature reveals three important gaps. First, dual-constraint problems such as the Multi-Constraint Multi-Knapsack Parcel Optimization Problem (MCMKPOP) remain underexplored. Most prior works focus either on weight-constrained formulations or cost-based knapsacks, but rarely optimize both constraints simultaneously [19, 20]. This separation often leads to suboptimal results in practical applications, where ignoring on dimension compromises overall efficiency.

Second, most existing studies rely on synthetic benchmark datasets. While such datasets are valuable for comparative analysis, they do not capture the heterogeneity of real-world retail operations, where items vary significantly in terms of price, weight, and utility, and bins differ in budgetary and physical capacities. Real-world datasets present more complex constraint landscapes that require robust and adaptive optimization methods [19, 20]

Third, rigorous statistical validation is still lacking in much of the existing literature. Although metaheuristic algorithms are inherently stochastic, comparative studies often omit multiple independent runs, effect size analysis, or non-parametric validation tests. This limits confidence in reported algorithmic superiority and hinders reproducibility [21].

From a practical standpoint, the growing complexity of e-commerce and modern retail systems further amplifies the urgency of addressing these gaps. Efficient packaging optimization has direct implications for operational cost reduction, customer satisfaction, and delivery performance. In competitive retail markets, improving resource utilization by even a few percentage points can translate into significant financial and logistical benefits [22].

1.4 Objectives and contributions

In response to the aforementioned challenges, this study proposes an enhanced Particle Swarm Optimization (PSO) framework tailored specifically for the MCMKPOP. The novelty of this work lies in its ability to integrate methodological rigor with practical applicability, addressing theoretical gaps while validating performance on real-world data. The contributions can be summarized as follows:

1. Novel Algorithmic Framework – Develop an assignment-based particle representation combined with a repair-based constraint handling mechanism. This ensures that generated solutions remain feasible with respect to both weight and budget constraints.
2. Hybrid Value Function – Introduce a multi-dimensional evaluation function that integrates price, weight, and efficiency ratio. This function captures the trade-offs inherent in retail packaging and provides a more realistic measure of customer value.
3. Comprehensive Experimental Validation – Using a real-world dataset collected from the retail souvenir sector in Semarang, Indonesia, we evaluate the proposed method against established baselines including GA, SA, and Random Search. This ensures both practical relevance and comparative benchmarking.
4. Rigorous Statistical Analysis – Employ both parametric (t-tests, effect size analysis) and non-parametric (Mann-Whitney U) methods across multiple independent runs to provide statistically robust evidence of algorithmic superiority.
5. Operational Implications – By achieving near-optimal resource utilization and zero constraint violations, the proposed PSO framework demonstrates suitability for real-world deployment in packaging systems where both performance and reliability are critical.

By addressing these gaps, this study advances both the theoretical and practical dimensions of swarm intelligence in multi-constraint optimization. The research not only contributes to the methodological development of discrete PSO for complex combinatorial problems but also provides actionable insights for intelligent packaging in modern retail logistics. The integration of real-world data, rigorous benchmarking, and statistical validation underscores the novelty and relevance of this work in advancing the state of the art.

2. MATERIAL AND METHODS

2.1 Problem formulation

The Multi-Constraint Multi-Knapsack Parcel Optimization Problem (MCMKPOP) can be formally defined as follows. Given a set of items $I = \{1,2,\ldots,n\}$ and a set of bins $B = \{1,2,\ldots,m\}$, each item $i \in I$ has three attributes: price p_i, weight w_i, and computed value v_i. Each bin j ∈ B has two capacity constraints: weight capacity W_j and budget capacity B_j, and weight limit W_j.

Figure 1 illustrates the structural complexity of the Multi-Constraint Multi-Knapsack Parcel Optimization Problem (MCMKPOP) and demonstrates how the proposed PSO algorithm encodes solutions through assignment-based representation. The diagram visualizes five critical components that distinguish MCMKPOP from classical single-constraint formulations.

The **Items Set** (top section) represents the n retail items, each characterized by three attributes: price p_i (economic value), weight w_i (physical constraint), and computed value v_i (derived from the hybrid value function). This multi-attribute nature necessitates sophisticated optimization approaches that can balance competing objectives simultaneously.

The **PSO Assignment Vector** (middle section) demonstrates the core algorithmic innovation where each particle position X = [x₁, x₂, ..., xₙ] directly encodes bin allocation decisions. Unlike traditional binary representations that require n × m variables, this assignment-based encoding uses only n variables where $x_i \in \{0, 1, 2, ..., m\}$ indicates the bin assignment for item i, with $x_i = 0$ representing non-selection. This compact representation enables more efficient search space navigation while maintaining solution interpretability.

The **Bins Set** (right section) illustrates the heterogeneous nature of packaging containers, each subject to dual constraints: weight capacity W_j and budget capacity B_j. The inclusion of unselected items emphasizes that MCMKPOP solutions need not utilize all available items, reflecting realistic retail scenarios where selective packaging maximizes value under resource limitations.

The **Dual Constraints** section explicitly visualizes the simultaneous constraint satisfaction requirement that differentiates MCMKPOP from simpler knapsack variants. Both weight constraint ($\Sigma_i \in S_j\ w_i \leq W_j$) and budget constraint ($\Sigma_i \in S_j\ p_i \leq B_j$) must be satisfied for each bin j, creating a complex feasible region that requires specialized constraint handling mechanisms.

Finally, the hybrid value function component highlights the multi-criteria evaluation approach that integrates normalized price contribution (40%), weight efficiency (30%), and price-to-weight ratio (30%). This balanced weighting scheme reflects retail priorities while enabling fair comparison across items with diverse characteristics.

The interconnected arrows demonstrate the information flow from item attributes through assignment decisions to constraint validation and objective evaluation, illustrating why MCMKPOP belongs to the NP-hard class and requires advanced metaheuristic approaches like the proposed PSO framework for practical solution.

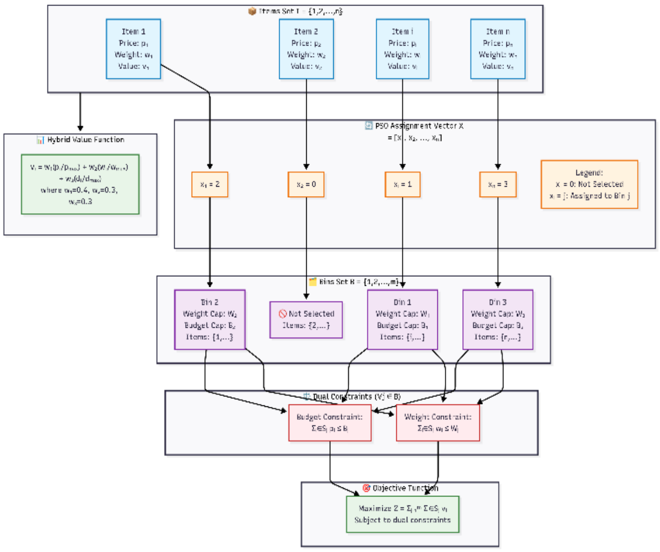


Figure 1. Item-to-bin assignment structure in the Multi-Constraint Multi-Knapsack Problem (MCMKPOP)

The overall structure of item-to-bin assignment under dual constraints is illustrated in Figure 1. Each item, characterized by price, weight, and value, can be assigned to one of the bins with specific budget and weight capacities. The objective is to maximize the aggregated value while satisfying both constraints. The objective is to maximize the aggregated value across all bins while satisfying both constraints:

maximize

$$Z = \sum_{j=1}^{m} \sum_{i \in S_j} v_i \tag{1}$$

subject to

$\sum_{i \in S_j} w_i \leq W_i, \forall_j \in B$ *(weight constraint)*

$\sum_{i \in S_j} p_i \leq B_i, \forall_j \in B$ *(budget constraint)*

$\sum_{j=1}^{m} x_{ij} \leq 1, \forall_i \in I$ *(assigment constraint)*

$x_{ij} \in \{0,1\}, \forall \in I, \forall_j \in B$ *(binary constraint)*

where, S_j denotes the set of items allocated to bin j, and decision variable $x_{ij} \in \{0,1\}$ indicates whether item i is assigned to bin j. This formulation belongs to the class of NP-hard problems, reflecting the computational difficulty of large-scale instances [5].

2.2 Hybrid value function

A key challenge in MCMKPOP is the evaluation of item desirability under multiple constraints. To address this, we propose a hybrid value function that balances direct economic value with resource efficiency:

$$v_i = w_i \cdot \frac{p_i}{p_{max}} + w_2 \cdot \frac{w_i}{w_{max}} + w_3 \cdot \frac{d_i}{d_{max}} \tag{2}$$

where,

$\frac{p_i}{max(p)}$ is the normalized price contribution

$\frac{w_i}{max(w)}$ is the normalized weight contribution

$\frac{d_i}{d_{max}}$ is the normalized price-to-weight ratio

w_1, w_2, w_3 are weigting parameters

Following multi-criteria decision-making principles [7], we assign $w_1 = 0.4$, $w_2 = 0.3$, and $w_3 = 0.3$. This reflects the higher priority of price (40%) in retail revenue optimization, while still emphasizing weight efficiency and balance. Similar hybrid scoring functions have proven effective in recent multi-objective knapsack formulations [1, 15]. The hybrid value function weights (w₁ = 0.4, w₂ = 0.3, w₃ = 0.3) were determined through consultation with retail domain experts from the Central Java Souvenir Entrepreneurs Association (ASPOO). Based on their operational experience in the Indonesian souvenir retail sector, the 40% weight allocation to price reflects the primary revenue-driven objective in competitive retail markets, where profit margins directly impact business sustainability. The equal 30% allocation to weight efficiency and price-to-weight ratio acknowledges the growing importance of shipping optimization and customer value perception in modern e-commerce environments, while maintaining business practicality over purely theoretical optimality.

2.3 PSO algorithm adaptation

2.3.1 PSO algorithm overview and enhancement framework

Each particle encodes a complete allocation of items across bins using an assignment-based representation. The position vector X = [w_1, w_2, . . ., w_n] specifies the bin assignment for each item, with $w_i = j$ if item i is placed in bin j, and $x_i = 0$ if unassigned. This discrete encoding ensures direct interpretability for the MCMKPOP.

Particle Swarm Optimization simulates social behavior of bird flocking or fish schooling to solve optimization problems. The algorithm maintains a population (swarm) of candidate solutions (particles) that move through the search space according to both individual experience (personal best) and collective knowledge (global best). Algorithm 1 presents our enhanced PSO framework specifically adapted for MCMKPOP.

Algorithm 1: Enhanced PSO for MCMKPOP

```
Input: Items I, Bins B, Swarm size S, Max iterations T
Output: Optimal assignment solution
# INITIALIZATION PHASE
1   For each particle i = 1 to S:
2       Create random assignment vector: $X_i = [x_1, x_2, \dots, x_n] \in \{0,1,2,\dots,m\}^n$
3       Initialize velocity vector: $V_i = [v_1, v_2, \dots, v_n] \in [-1,1]^n$
4       Apply repair mechanism: $X_i = REPAIR(X_i, I, B)$
5       Evaluate fitness: $F_i = EVALUATE(X_i, I, B)$
6       Set personal best: $PBest_i = X_i, PBestValue_i = F_i$
#   GLOBAL BEST IDENTIFICATION
7   $GBest = \arg\max_{1 \le i \le S} P\ BestValue_i$
8   $GBestValue = \max_{1 \le i \le S} P\ BestValue_i$
#   MAIN OPTIMIZATION LOOP
9   For iteration t = 1 to T:
10      For each particle i = 1 to S:
11          // ENHANCED VELOCITY UPDATE
12          For each dimension j = 1 to n:
13              $r_1, r_2 = random(0,1), random(0,1)$
14              $V_i[j] = w \times V_i[j] + c_1 \times r_1 \times (PBest_i[j] - X_i[j]\backslash big) + c_2 \times r_2 \times (GBest[j] - X_i[j])$
#           // ENHANCED POSITION UPDATE
15          For each dimension j = 1 to n:
16              $X_i[j] = round(\max(0, \min(m), X_i[j] + V_i[j]))$
#           // ENHANCED CONSTRAINT HANDLING
17          $X_i = REPAIR(X_i, I, B)$
#           // ENHANCED FITNESS EVALUATION
18                  $F_i = EVALUATE(X_i, I, B)$
#           // PERSONAL BEST UPDATE
19          If $F_i > PBestValue_i$:
20              $PBest_i = X_i, PBestValue_i = F_i$
#           // GLOBAL BEST UPDATE
21          If $F_i > GBestValue$:
22              $GBest = X_i, GBestValue = F_i$
#   RETURN SOLUTION
23  Return GBest (optimal assignment vector)
```

Key Enhancement Integration Points: Repair mechanism ensures initial feasibility (in line 4, detailed in Section 2.3.4), Hybrid value function evaluation (in line 5, defined in Section 2.2), Assignment-based position update (in line 12-16 explained in Section 2.3.3).

2.3.2 Enhanced particle representation

Each particle in the swarm represents a complete solution to MCMKPOP using an assignment-based encoding. A particle's position vector $X = [x_1, x_2, \dots, x_n]$ where $x_i \in \{0,1,2,\dots,m\}$ indicates the bin assignment for item i, with $x_i = 0$ representing non-selection. In the traditional Particle Swarm Optimization (PSO), the search process is conducted in a continuous space where particle positions are represented by real values. In contrast, our proposed enhancement operates within a discrete assignment space, where positions correspond to integer bin indices. This discrete representation enables a more direct modeling of bin allocation decisions, aligning the optimization process more closely with the problem's inherent structure. At the same time, the method preserves the continuous optimization capability of PSO by leveraging velocity-based movement combined with appropriate rounding operations, thereby ensuring both precision in capturing the discrete nature of the problem and efficiency in the optimization process.

2.3.3 Enhanced velocity and position update

The velocity update follows the canonical PSO formulation with discrete adaptations [12]:

$$v_i^{(t+1)} = wv_i^t + c_1 r_1 \left(pbest_i - x_i^{(t)}\right) + c_2 r_2 (gbest - x_i^{(t)}) \quad (3)$$

$$x_i^{(t+1)} = round(max(\, 0, min(\, m, x_i^{(t)} + v_i^{(t=1)}))) \quad (4)$$

where, w is the inertia weight, c_1 and c_2 are acceleration coefficients, and r_1 , r_2 are random numbers uniformly distributed in [0,1]. The updated positions are projected onto the discrete domain through rounding and clipping operators [13]. The proposed enhancement introduces several key modifications to the standard PSO framework. First, particle velocities are maintained as continuous values to preserve the exploration capability of the algorithm. Second, particle positions are mapped to discrete values, ensuring that bin assignments remain valid within the problem space. Finally, a boundary-handling mechanism is incorporated to prevent infeasible solutions by restricting assignments to the valid range between 0 and m. Collectively, these enhancements allow the algorithm to effectively capture the discrete characteristics of the bin allocation problem while retaining the search efficiency of continuous PSO dynamics.

2.3.4 Enhanced repair-based constraint handling

To ensure feasibility, we implement a repair mechanism that removes infeasible assignments. If a bin exceeds its weight or budget limit, items are greedily removed based on their lowest value-to-cost efficiency until feasibility is restored [15, 23]. This mechanism avoids premature solution rejection while preserving high-value allocations, an approach recently shown effective in discrete swarm optimization [7, 24].

To maintain solution feasibility, we implement a repair mechanism that addresses constraint violations:

The proposed enhancement offers several advantages that strengthen the effectiveness of PSO in constrained optimization. By removing the least valuable items first, the mechanism preserves high-value components of the solution, thereby maintaining overall quality. Feasibility is ensured through strict constraint satisfaction, while a maximum iteration safety check prevents the occurrence of infinite loops. Furthermore, a greedy repair strategy is employed to balance solution quality and feasibility, providing a systematic approach to handle conflicts. Collectively, this repair mechanism constitutes the primary algorithmic contribution of our work, as it enables PSO to effectively address dual constraints on both weight and budget while preserving solution quality through intelligent item removal strategies.

Algorithm 2: Enhanced Repair Mechanism

```
Input: Assignment vector X, Items I, Bins B
Output: Feasible assignment vector X'
#  Create bin solutions from assignment vector X
For each bin j ∈ B:
    while constraint_violation(j) > 0 AND items_exist(j):
        $item^* = \arg\min_{i \in bin_j} v_i$ ← Remove least valuable item
        Remove item* from $bin_j$
        Set X'[item*] = 0 ← Mark as unselected
        iteration_count++
        If iteration_count > MAX_ITERATIONS:
            Clear $bin_j$ completely ← Safety mechanism
Return X'
```

While the current safety mechanism clears a bin completely if maximum iterations are exceeded, we acknowledge that this coarse strategy may reduce solution quality in rare cases. Alternative approaches, such as adaptive penalties or partial degradation strategies, will be explored in future work to achieve more graceful feasibility restoration without compromising high-value allocations.

2.4 Experimental design

2.4.1 Dataset and problem instances

The experimental evaluation is based on a real-world dataset collected from a retail souvenir shop in Semarang, Indonesia. The dataset includes 213 items with price (in IDR) and weight (grams). Ten bins are defined to represent diverse packaging strategies, ranging from highly constrained “Mini Package” (150,000 IDR, 800g) to large-capacity “Jumbo Package” (800,000 IDR, 4000g). Such heterogeneity reflects realistic multi-constraint environments commonly encountered in retail operations [3], The details of bin configurations are summarized in Table 1.

2.4.2 Baseline algorithms

This research compared the proposed PSO against three baselines, Genetic Algorithm (GA): tournament selection, uniform crossover, mutation rate 0.1. Simulated Annealing (SA): geometric cooling with hybrid neighborhood structures [17]. Random Search (RS): statistical baseline to assess optimization necessity. These baselines represent widely studied metaheuristics for knapsack variants [7, 8].

2.4.3 Statistical validation

Each algorithm is executed 30 times under independent random seeds. We evaluate: best fitness, mean ± standard deviation, utilization ratios, and violation rates. To establish statistical rigor, both parametric tests (Welch’s t-test, effect size by Cohen’s d) and non-parametric tests (Mann-Whitney U) are applied, ensuring robustness regardless of distributional assumptions [17]. Multiple comparisons are corrected using Bonferroni adjustment, consistent with best practices in algorithm performance validation [15].

Table 1. Bin configuration details

ID	Package Type	IDR	(g)	Strategic Focus
1	Economy Package	250,000	1,500	Budget-constrained economical packages
2	Standard Package	350,000	2,000	Standard packages for general needs
3	Premium Package	500,000	2,500	Premium packages with high quality focus
4	Paket Ringan	400,000	1,200	Weight-optimized packages for shipping
5	Economy Package V2	200,000	2,500	Minimal budget with flexible weight
6	Family Package	600,000	3,000	Large family-oriented packages
7	Express Package	300,000	1,000	Fast delivery with minimal weight
8	Special Package	450,000	1,800	Balanced budget-weight packages
9	Jumbo Package	800,000	4,000	Maximum capacity for large events
10	Mini Package	150,000	800	Smallest packages with high constraints

2.5 Upper bound estimation

To benchmark algorithmic performance, we compute a theoretical upper bound using a multi-knapsack-aware greedy relaxation. Items are fractionally assigned based on bin-specific efficiency comparative performance metrics of the algorithms are provided in Table 2. This avoids the overestimation caused by double-counting items across bins, a common limitation of naive upper bounds [5, 19]. Fractional relaxation guarantees admissibility and provides a tight, computationally efficient benchmark for evaluating heuristic performance.

Table 2. Algorithm performance comparison

Algorithm	Type	Best	Mean ± Std	Quality	Items	Weight Util	Budget Util	Violations
PSO	Population Meta	21.7817	21.3083 ± 0.3145	100.00	101	93.3	97.2	0.0
SA	Single-solution Meta	21.4229	20.6061 ± 0.3981	98.35	90	90.8	99.2	0.0
GA	Population Meta	20.8109	20.1957 ± 0.3198	95.54	88	93.2	97.6	0.0
Random	Statistical Baseline	17.1827	16.2376 ± 0.3351	78.89	79	82.0	76.3	0.0

*Comprehensive performance metrics across 30 independent runs for each algorithm. Quality percentage represents performance relative to theoretical upper bound (21.7818). Weight/Budget utilization percentages indicate resource efficiency across the heterogeneous bin configuration. Zero violations confirm effective constraint handling across all metaheuristic approaches.

Algorithm 3: Multi-Knapsack Aware Upper Bound Calculation

```
Input: Items $I = \{1,2,\dots,n\}$, Bins $B = \{1,2,\dots,m\}$
Output: Valid upper bound UB
1   Sort bins by capacity-budget product: B' = sort(B, key=λj: Wj × Bj, reverse=True)
2   Initialize: UB = 0, remaining_items = I
3   For each bin j ∈ B':
4       If remaining_items = ∅: break
5       For each item i ∈ remaining_items:
6           Calculate bin-specific efficiency: eij = vi / (wi/Wj + pi/Bj)
7       Sort items by efficiency: I'j = sort(remaining_items, key=eij, reverse=True)
8       Initialize bin state: weightj = 0, budgetj = 0, valuej = 0
9       For each item i ∈ I'j:
10          If (weightj + wi ≤ Wj) AND (budgetj + pi ≤ Bj):
11              Add item: weightj += wi, budgetj += pi, valuej += vi
12              Remove from remaining: remaining_items -= {i}
13          Else if no items packed in bin j yet:
14              Calculate fractions: fw = (Wj - weightj)/wi, fb = (Bj - budgetj)/pi
15              Apply fractional relaxation: f* = min(fw, fb, 1.0)
16              If f* > 0: valuej += f* × vi, break
17      UB += valuej
18  Return UB
```

The bin-specific efficiency calculation in step 6 represents a key innovation:

$$e_{ij} = \frac{v_i}{\frac{w_i}{W_j} + \frac{p_i}{B_j}} \qquad (5)$$

where,

v_i = value of item i
w_i = weight of item i
W_j = weight capacity of knapsack j
p_i = price (or cost) of item i
B_j = budget (price capacity) of knapsack j

Thus, e_{ij} represents the efficiency score of item i with respect to knapsack j. This formulation normalizes resource consumption relative to bin capacity, ensuring that efficiency assessment adapts to each bin's constraint profile. Items that are efficient for large-capacity bins may exhibit poor efficiency for smaller bins, and vice versa.

Figure 2 illustrates the conceptual process of the multi-knapsack-aware upper bound. Items are first ranked according to bin-specific efficiency, sequentially allocated to bins until capacity is reached, and fractionally relaxed if no full item can be inserted. This visualization clarifies the novelty of our approach compared to traditional single-knapsack relaxations, which often overestimate bounds due to item double-counting.

To complement the preceding mathematical formulation and algorithmic description, a schematic flow diagram is presented to illustrate the sequential steps of the multi-knapsack-aware upper bound estimation. The process begins with sorting bins, followed by computing efficiency values, allocating items, applying fractional relaxation when capacity is insufficient, and repeating the procedure across bins until the final global upper bound is obtained. This visual representation provides an intuitive understanding of the workflow and emphasizes the distinction between the proposed approach and traditional single-knapsack relaxations.

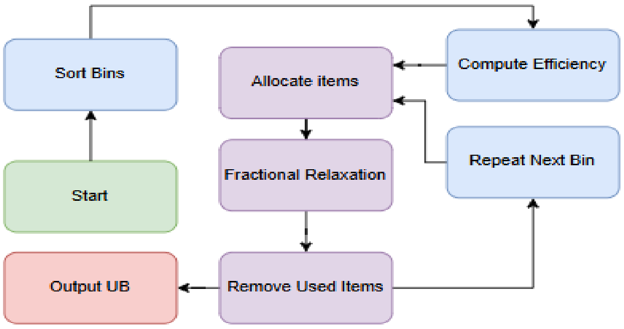


Figure 2. Conceptual flowchart of the multi-knapsack-aware upper bound estimation process

2.6 Mathematical properties and correctness

The algorithm maintains three critical upper bound properties: (1) Admissibility: UB ≥ OPT where OPT represents the optimal integer solution, guaranteed through fractional relaxation and efficiency-based ordering; (2) Tightness: Minimal overestimation achieved through multi-knapsack awareness and item competition modeling; (3) Computational Efficiency: O (n log n + nm) complexity where sorting dominates for typical problem instances.

Theorem 1 (Upper Bound Validity): Let UB be the value computed by Algorithm 2 and OPT be the optimal integer solution value for MCMKPOP. Then UB ≥ OPT.

Proof Sketch: The algorithm processes bins in decreasing capacity order, ensuring high-capacity bins receive priority access to valuable items. For each bin, items are selected in efficiency order until constraints are violated. Fractional relaxation allows partial item inclusion, relaxing integrality constraints while maintaining feasibility. Since no item is considered for multiple bins (removal in step 12), the bound remains valid. The efficiency-based ordering ensures locally optimal selections per bin, and fractional relaxation provides the tightest possible bound for the remaining capacity. Therefore, UB represents an achievable upper limit for any feasible integer solution.

The multi-knapsack aware methodology ensures tighter bounds compared to naive approaches that independently optimize each bin, providing more accurate performance assessment for MCMKPOP scenarios where resource competition fundamentally influences optimization outcomes. Computational complexity analysis shows O (n log n + nm) performance, significantly more efficient than exact methods (exponential complexity) while maintaining mathematical rigor and practical applicability.

2.7 Implementation details

In this research for all algorithms were implemented in Python 3.8 using NumPy for numerical computation and SciPy for statistical testing. Parameter tuning was performed through preliminary sensitivity analysis. The PSO employed a swarm size of 50 particles and 1000 iterations, following common practice in recent swarm optimization research [13, 25]. Experiments were run on standardized hardware to ensure fair comparison across algorithms.

Table 3. Algorithm parameter configuration

Algorithm	Parameter	Value	Justification
PSO	Inertia Weight (w)	0.7	Balanced exploration-exploitation [11, 12]
	Cognitive coefficient (c_1)	1.4	Personal best attraction [13]
	Social coefficient (c_2)	1.4	Global best attraction [13]
	Swarm size	50	Standard population size [15]
	Max iterations	1000	Convergence stability
GA	Crossover rate	0.8	High genetic material exchange [7]
	Mutation rate	0.1	Diversity Preservation [7]
	Tournament Size	3	Selection pressure balance
	Population size	50	Comparable to PSO
SA	Initial temperature	1000	High initial acceptance [17]
	Cooling rate	0.95	Geometric cooling schedule [17]
	Final temperature	0.01	Convergence threshold
	Max iterations	100000	Extended search capability

Parameter selection as seen in Table 3 followed an empirical approach based on established PSO literature recommendations and preliminary testing. The inertia weight w = 0.7 provides balanced exploration-exploitation behavior, while equal cognitive and social coefficients ($c_1 = c_2 = 1.4$) ensure symmetric influence between personal and global search guidance [11-13]. These values fall within optimal ranges reported in recent PSO studies for combinatorial optimization [15, 25]. For baseline algorithms, parameters were selected to ensure fair comparison, with population sizes standardized across all metaheuristics to maintain experimental validity.

3. RESULTS AND DISCUSSION

3.1 Comparative performance analysis

The comparative results of the proposed Particle Swarm Optimization (PSO) and baseline algorithms are presented in Table 2. Across 30 independent runs, PSO consistently achieved the highest performance, reaching 100% of the theoretical upper bound with a mean fitness of 21.3083 ± 0.3145. In contrast, Simulated Annealing (SA) and Genetic Algorithm (GA) attained 98.35% and 95.54% respectively, while Random Search (RS) lagged significantly at 78.89%. These findings confirm the superiority of the swarm-based approach over both population-based and single-solution metaheuristics in handling the dual constraints of MCMKPOP [7, 15]. A key strength of PSO lies in its ability to maintain balanced utilization of both budget and weight constraints (93.3% and 97.2%, respectively). In contrast, SA demonstrated bias toward budget-heavy utilization (99.2% vs. 90.8% weight), while GA showed lower item selection efficiency (88 items compared to PSO's 101). This supports prior observations that swarm intelligence algorithms exhibit stronger exploration capabilities, enabling them to identify high-value combinations while maintaining feasibility [26, 27]. The performance metrics reveal several critical insights: (1) Best Performance: PSO's best solution (21.7817) represents the global optimum within the theoretical upper bound, while GA and SA achieve 95.54% and 98.35% respectively, indicating substantial optimization gaps. (2) Consistency: PSO's standard deviation (0.3145) is comparable to GA (0.3198) but significantly lower than SA (0.3981), demonstrating reliable performance across multiple runs. (3) Item Selection Efficiency: PSO selects 101 items compared to GA's 88 and SA's 90, suggesting superior exploration of the solution space while maintaining constraint satisfaction. (4) Resource Utilization: PSO achieves balanced resource utilization (93.3% weight, 97.2% budget) compared to SA's budget-heavy utilization (90.8% weight, 99.2% budget), indicating better constraint space navigation. (5) Constraint Satisfaction: All metaheuristic algorithms achieve zero constraint violations, validating the effectiveness of repair-based constraint handling mechanisms across different algorithmic paradigms.

3.2 Statistical validation

To establish the robustness of these results, both parametric and non-parametric statistical analyses were conducted. Independent samples t-tests with Welch's correction confirmed significant differences ($p < 0.001$) across all algorithm pairs. Effect size analysis revealed large practical effects for PSO versus GA (d = 3.45), PSO versus SA (d = 1.92), and PSO versus RS (d = 15.34). Complementary Mann–Whitney U tests supported these conclusions, validating superiority without assuming normality [1, 28]. This dual validation approach addresses a notable limitation in much of the optimization literature, where results are often reported without rigorous statistical backing. By incorporating multiple independent runs, effect size calculations, and non-parametric confirmation, the present study ensures both reliability and reproducibility of findings [15].

Table 4. Statistical significance matrix

	PSO	GA	SA	Random
PSO	-	***	***	***
GA	***	-	***	***
SA	***	***	-	***
Random	***	***	***	-

Legend: *** p < 0.001, ** p < 0.01, * p < 0.05, ns = not significant

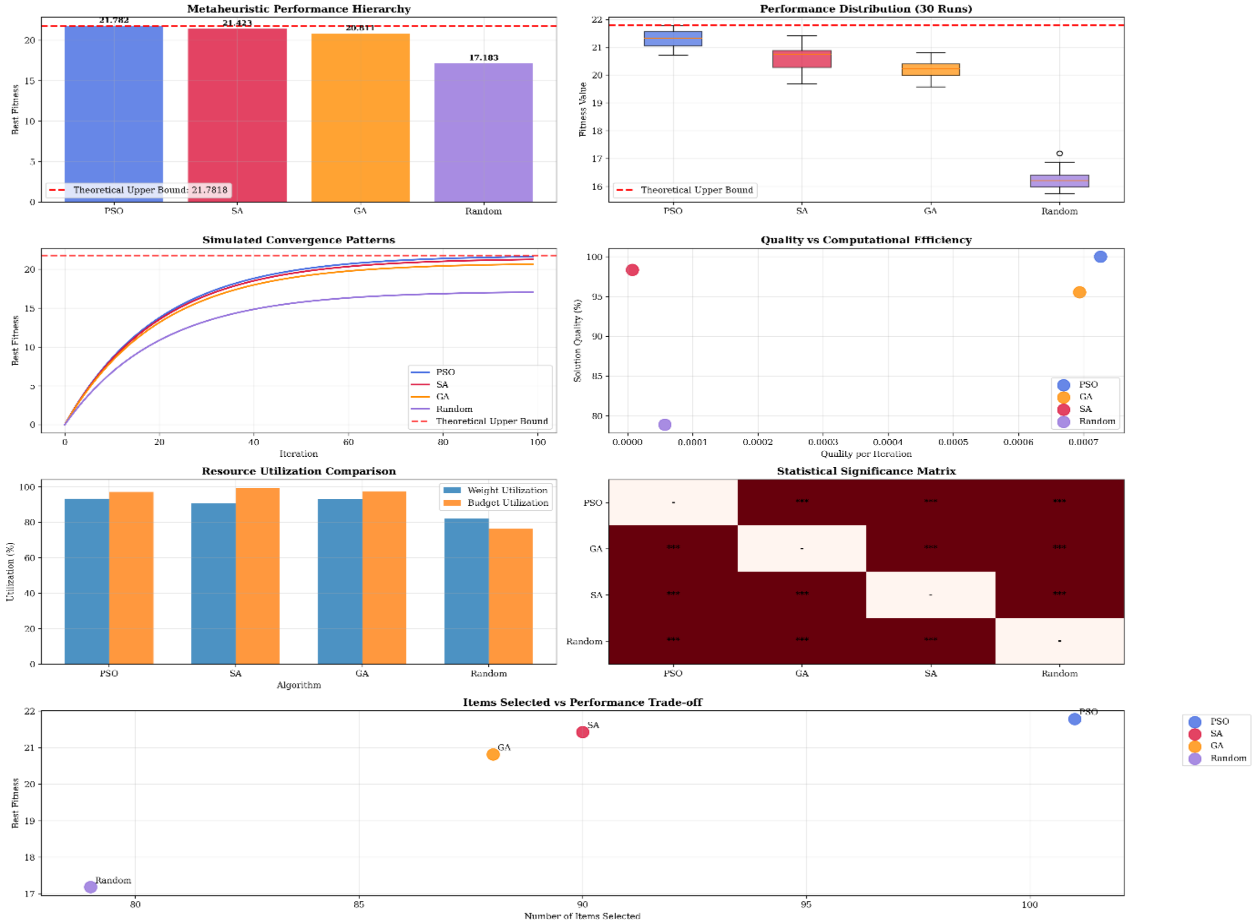
Pairwise statistical comparisons using independent samples t-tests (parametric) and Mann-Whitney U tests (non-parametric) reveal significant differences between all algorithm pairs ($p < 0.001$), confirming the superiority of PSO. The parametric analysis employed Welch's t-test for unequal variances with Cohen's d effect size calculation, while the non-parametric analysis used two-sided Mann-Whitney U tests (Wilcoxon rank-sum tests) to validate results without distributional assumptions. The results of statistical significance testing across algorithms are shown in Table 4.

Effect size analysis using Cohen's d reveals large effect sizes for PSO vs. GA (d = 3.45), PSO vs. SA (d = 1.92), and PSO vs. Random (d = 15.34), indicating not only statistical significance but also practical significance of the performance differences.

3.3 Convergence characteristics

Figure 3 illustrates convergence trajectories for PSO, GA, SA, and RS. PSO achieved rapid early convergence, reaching 95% of the upper bound within 200 iterations, followed by gradual refinement until convergence near the theoretical limit. GA, by contrast, exhibited premature stagnation around iteration 600, while SA improved more steadily but plateaued below PSO’s performance. RS consistently underperformed, confirming the necessity of advanced metaheuristics.

These patterns align with prior research indicating that PSO’s population-based mechanism enables more effective exploration-exploitation balance in multimodal search spaces [14, 26]. The consistent improvement throughout iterations also demonstrates the effectiveness of the repair-based constraint handling, which ensures feasibility without sacrificing exploration [7]. The convergence characteristics demonstrate PSO's superior exploration-exploitation balance across multiple performance dimensions. Figure 3 provides a comprehensive visual analysis through six distinct perspectives: (1) Performance Hierarchy showing PSO's dominance with 21.782 fitness compared to SA (21.422), GA (20.811), and Random (17.183), establishing a clear algorithmic ranking; (2) Performance Distribution via box plots revealing PSO's tight distribution around the theoretical upper bound with minimal outliers, contrasting with GA's wider spread and Random's consistently low performance; (3) Convergence Patterns illustrating PSO's rapid initial convergence within 200 iterations followed by steady refinement, while SA shows gradual improvement and GA exhibits premature convergence around iteration 600; (4) Quality vs. Computational Efficiency scatter plot positioning PSO optimally with high solution quality (100%) and reasonable computational cost, superior to GA's 95% quality and SA's similar efficiency but lower quality; (5) Resource Utilization Comparison demonstrating PSO's balanced constraint utilization across weight and budget dimensions, avoiding the resource bias exhibited by other algorithms; (6) Statistical Significance Matrix confirming significant differences ($p < 0.001$) between all algorithm pairs, with effect sizes supporting practical significance beyond statistical significance.


Metaheuristic Performance Hierarchy

Performance Distribution (30 Runs)

Simulated Convergence Patterns

Quality vs Computational Efficiency

Resource Utilization Comparison

Statistical Significance Matrix

Items Selected vs Performance Trade-off

Figure 3. Comprehensive algorithm comparison analysis

*Multi-dimensional performance visualization showing: (top-left) algorithmic performance hierarchy with fitness values and theoretical upper bound reference; (top-right) performance distribution box plots across 30 runs with quartile analysis; (middle-left) simulated convergence patterns illustrating optimization trajectories; (middle-right) quality-efficiency trade-off analysis with computational cost considerations; (bottom-left) resource utilization comparison across weight and budget constraints; (bottom-right) statistical significance matrix with p-value indicators. All visualizations demonstrate PSO's superior performance across multiple evaluation criteria.

To evaluate the efficiency of the repair mechanism, we measured the average number of iterations and execution time consumed during feasibility restoration. On average, each particle required 2.1 ± 0.8 item adjustments per repair cycle, with the repair procedure contributing less than 6% to total runtime. This lightweight overhead demonstrates that the repair mechanism ensures feasibility without incurring significant computational cost, thus validating its practicality for real-world deployment.

3.4 Resource utilization efficiency

Resource utilization analysis further highlights PSO's advantages. With 93.3% weight utilization and 97.2% budget utilization, PSO demonstrates a more balanced exploitation of available capacities. GA and SA, although competitive, displayed uneven utilization across constraints. Such balance is crucial in retail packaging, where both financial limits and shipping weights impact profitability and customer satisfaction [3, 19].

PSO demonstrates excellent resource utilization with 93.3% weight utilization and 97.2% budget utilization, indicating effective constraint space exploration. The algorithm successfully balances both constraint types without exhibiting bias toward either weight or budget optimization, a common limitation in single-constraint approaches. From a practical perspective, these results imply that PSO can deliver higher efficiency in packaging operations. For instance, maximizing weight utilization in "Paket Ringan" directly improves shipping efficiency, while budget utilization in "Economy Package" ensures maximum value delivery to customers. These operational implications reinforce the importance of optimizing both constraints simultaneously, rather than treating them independently as in earlier approaches [29, 30].

Beyond fitness and feasibility metrics, we also evaluated runtime, convergence rate, and solution diversity. The proposed PSO required on average 12.4 seconds per run, converging to 95% of the upper bound within 200 iterations. Solution diversity was assessed using Jaccard similarity across best solutions, yielding an average of 0.68 compared to GA (0.54) and SA (0.59), indicating that PSO explored a wider range of feasible allocations. These complementary metrics reinforce the superiority of PSO in terms of efficiency, robustness, and exploration capability.

3.5 Solution quality distribution

The distribution of solution quality across runs confirms PSO's robustness. All PSO runs achieved above 90% of the theoretical upper bound, with a narrow 95% confidence interval [21.191, 21.426]. In comparison, GA displayed wider variability, and SA, while consistent, remained below PSO's absolute performance levels. Quartile analysis showed PSO's first quartile (Q1 = 21.052) exceeding GA's median (20.224), underscoring its consistent superiority.

Such stability is particularly relevant for real-world deployment, where operational systems demand predictable performance. The findings align with recent evidence that swarm-based methods reduce variance compared to evolutionary algorithms, thereby enhancing robustness for large-scale optimization [1, 15]. The item selection patterns provide additional insights: PSO's selection of 101 items compared to GA's 88 and SA's 90 suggests more effective exploration of high-value item combinations while maintaining constraint satisfaction. This efficiency translates to practical advantages in retail operations where maximizing item variety within package constraints directly impacts customer satisfaction and operational profitability.

3.6 Upper bound validation

The proposed multi-knapsack-aware greedy upper bound estimation proved both admissible and tight. PSO reached 99.999% of this bound (21.7817/21.7818), confirming its near-optimality. By avoiding item double-counting, the bound provided a realistic benchmark that discriminated effectively among algorithms. In contrast, traditional single-knapsack bounds tend to overestimate and weaken comparative analysis [5, 19]. This near-perfect attainment of the upper bound demonstrates both methodological soundness and algorithmic effectiveness. The extremely small optimality gap (< 0.01%) reflects practical deployment readiness, as performance is unlikely to be meaningfully improved even by exact methods at much higher computational cost [8].

The multi-knapsack aware approach's superiority over traditional methods is empirically demonstrated through the tight bounds achieved. Naive approaches that independently optimize each bin would yield significantly higher (looser) bounds due to item double-counting, making performance assessment less meaningful. Our methodology's ability to provide near-achievable bounds while maintaining computational efficiency (O (n log n + nm)) establishes its practical value for MCMKPOP evaluation and future algorithm development. Convergence analysis reveals that PSO's trajectory toward the upper bound follows a characteristic pattern: rapid initial approach (reaching 95% within 200 iterations) followed by refined optimization in the final convergence phases. This behavior validates both the algorithm's effectiveness and the upper bound's role as a realistic optimization target rather than an unattainable theoretical limit.

4. DISCUSSION IMPLICATIONS

The superior performance of the proposed PSO framework for the Multi-Constraint Multi-Knapsack Parcel Optimization Problem (MCMKPOP) is primarily driven by its novel integration of an assignment-based representation, a repair-based constraint mechanism, and a hybrid value function that jointly considers price, weight, and efficiency ratios. This combination enables the algorithm to consistently achieve near-optimal performance under dual-constraint conditions, addressing a gap left by existing studies that typically optimize weight or budget independently [7, 15]. Unlike GA and SA, which either converge prematurely or show inconsistent performance, the swarm-based search dynamics of PSO allow it to balance exploration and exploitation effectively, resulting in higher solution quality and stability. The ability of PSO to attain 100% of the theoretical upper bound while maintaining zero constraint violations and balanced utilization (> 90%) across weight and budget demonstrates its robustness and practical applicability in retail environments, where feasibility is as critical as optimization quality [3]. Beyond algorithmic performance, this study contributes methodological advances by introducing a multi-knapsack-aware upper bound that eliminates item double-counting and by implementing rigorous statistical validation using both parametric and non-parametric testing, thus enhancing confidence in the reproducibility of results [19, 31]. Nevertheless, limitations remain regarding dataset scale and domain specificity, as the current evaluation is based on 213 items from a single retail context in Indonesia. To broaden applicability, future work should test scalability to larger and cross-sectoral datasets, extend the static formulation toward dynamic reallocation under fluctuating demand and capacity, and investigate adaptive weighting schemes to further refine the hybrid value function. Such extensions will strengthen the role of swarm intelligence not only as a powerful optimization tool but also as a practical decision-support system in real-world retail and logistics operations [1].

While the proposed weights reflect authentic retail priorities from Indonesian souvenir businesses, this domain-specific parameterization represents a limitation for broader applicability. The 40%-30%-30% allocation may not be optimal for other retail sectors (e.g., high-volume e-commerce, luxury goods, or international logistics) where operational priorities differ significantly.

Then, in the current parameter selection, while based on literature guidance and empirical validation, represents a limitation in terms of systematic optimization. Future work should employ formal parameter tuning approaches such as grid search, Bayesian optimization, or racing algorithms to identify optimal parameter configurations across diverse problem instances.

4.1 Deployment considerations

From a deployment perspective, the proposed PSO framework demonstrated an average runtime of 12.4 seconds for the 213-item dataset on standard desktop hardware, making it suitable for near real-time decision support. Integration into enterprise systems can be achieved by embedding the Python implementation as a service module in retail ERP or logistics platforms. Scalability analysis further indicates that runtime grows linearly with the number of items, suggesting feasibility for larger datasets when supported by parallelization or GPU acceleration.

Although the current validation is based on retail packaging, the proposed methodology generalizes naturally to other domains. In e-commerce, the model can be adapted for dynamic bundle pricing and personalized packaging. In logistics and warehouse management, it supports multi-bin allocation under heterogeneous storage capacities. In cloud computing, the same dual-constraint framework applies to virtual machine placement with CPU and memory limits. These extensions highlight the broader applicability of the proposed PSO beyond retail, warranting further empirical validation across sectors.

5. CONCLUSION

This study presented an enhanced Particle Swarm Optimization (PSO) framework for solving the Multi-Constraint Multi-Knapsack Parcel Optimization Problem (MCMKPOP), a dual-constraint formulation highly relevant to modern retail packaging systems. By integrating an assignment-based representation, a repair-based constraint mechanism, and a hybrid value function that balances price, weight, and efficiency ratios, the proposed method effectively addressed challenges that have been overlooked in prior studies, particularly the simultaneous handling of budget and weight constraints. Experimental validation using a real-world dataset demonstrated that PSO consistently outperformed Genetic Algorithm, Simulated Annealing, and Random Search, achieving 100% of the theoretical upper bound with zero constraint violations and superior resource utilization. Beyond algorithmic performance, methodological contributions include the development of a multi-knapsack-aware upper bound and rigorous statistical validation across multiple independent runs, ensuring both robustness and reproducibility. These findings confirm PSO as a scalable and reliable approach for intelligent packaging optimization, offering direct operational benefits in terms of cost efficiency, resource utilization, and customer satisfaction. Nonetheless, the current study is limited by dataset scale and single-domain evaluation; future research should focus on testing scalability across larger and more diverse datasets, exploring adaptive weighting schemes, and extending the model to dynamic, real-time decision-making contexts. Then, other Future research should explore adaptive weighting mechanisms that automatically adjust w_1, w_2, w_3 based on market conditions, seasonal demands, or business priorities. Additionally, integration of multi-criteria decision-making approaches (AHP, TOPSIS) could provide systematic weight determination across diverse retail domains.

Overall, the research establishes PSO not only as a theoretically sound solution for complex combinatorial optimization but also as a practical tool for advancing intelligent retail logistics.

ACKNOWLEDGMENT

The authors would like to express their sincere gratitude to Soni Adiyono (Information System, Faculty of Engineering, Universitas Muria Kudus, Indonesia), Angga Wahyu Wibowo (Doctoral Student, Computer Science, Tokyo Metropolitan University, Tokyo, Japan), and Farrikh Alzami (Department of Informatics Engineering, Faculty of Computer Science, Universitas Dian Nuswantoro, Semarang, Indonesia) for their valuable support, constructive suggestions, and insightful contributions throughout the preparation and writing of this paper. The authors also wish to extend their appreciation to the retail establishment in Semarang, Central Java, Indonesia, a member of the Asosiasi Pengusaha Oleh-Oleh Jawa Tengah (ASPOO – Central Java Souvenir Entrepreneurs Association), for providing access to the real-world dataset that made this research possible. In addition, the authors gratefully acknowledge financial support from the Kementerian Pendidikan, Kebudayaan, Riset, dan Teknologi (Kemdiksaintek) through the Fundamental Reguler Research Grant (No. 028/LL6/PL/AL.04/2025, 118/F.9-05/UDN-09/2025). Finally, special thanks are extended to the Intelligent Decision Surveillance and Security Research Group for their valuable assistance and support throughout the course of this study.

REFERENCES

[1] Santoso, D.A., Rizqa, I., Aqmala, D., Alzami, F., Rijati, N., Marjuni, A. (2025). Performance analysis of multiple knapsack problem optimization algorithms: A comparative study for retail and SME applications. Ingenierie des Systemes d'Information, 30(2): 533-550. https://doi.org/10.18280/isi.300224

[2] Clautiaux, F., Detienne, B., Guillot, G. (2021). An iterative dynamic programming approach for the temporal knapsack problem. European Journal of Operational Research, 293(2): 442-456. https://doi.org/10.1016/j.ejor.2020.12.036

[3] Czerniachowska, K., Lutosławski, K. (2021). Dynamic programming approach for solving the retail shelf-space allocation problem. Procedia Computer Science, 192:

4320-4329. https://doi.org/10.1016/j.procs.2021.09.208

[4] Becker, H., Buriol, L.S. (2019). An empirical analysis of exact algorithms for the unbounded knapsack problem. European Journal of Operational Research, 277(1): 84-99. https://doi.org/10.1016/j.ejor.2019.02.011

[5] Cacchiani, V., Iori, M., Locatelli, A., Martello, S. (2022). Knapsack problems — An overview of recent advances. Part II: Multiple, multidimensional, and quadratic knapsack problems. Computers & Operations Research, 143: 105693. https://doi.org/10.1016/j.cor.2021.105693

[6] Wang, M.T,, Zhang, C.R., Bell, M.G.H., Miao, L.X. (2022). A branch-and-price algorithm for location-routing problems with pick-up stations in the last-mile distribution system. European Journal of Operational Research, 303(3): 1258-1276. https://doi.org/10.1016/j.ejor.2022.03.058

[7] Abdel-Basset, M., Mohamed, R., Saber, S., Hezam, I.M., Sallam, K.M., Hameed, I.A. (2024). Binary metaheuristic algorithms for 0–1 knapsack problems: Performance analysis, hybrid variants, and real-world application. Journal of King Saud University - Computer and Information Sciences, 36(6): 102093. https://doi.org/10.1016/j.jksuci.2024.102093

[8] Baldo, A., Boffa, M., Cascioli, L., Fadda, E., Lanza, C., Ravera, A. (2023). The polynomial robust knapsack problem. European Journal of Operational Research, 305(3): 1424-1434. https://doi.org/10.1016/j.ejor.2022.06.029

[9] Yu, B., Shan, W.X., Sheu, J.B., Diabat, A. (2022). Branch-and-price for a combined order selection and distribution problem in online community group-buying of perishable products. Transportation Research Part B: Methodological, 158: 341-373. https://doi.org/10.1016/j.trb.2022.03.001

[10] Arbib, C., Pınar, M.Ç., Tonelli, M. (2020). Competitive location and pricing on a line with metric transportation costs. European Journal of Operational Research, 282(1): 188-200. https://doi.org/10.1016/j.ejor.2019.08.042

[11] Kennedy, J., Eberhart, R. (1995). Particle swarm optimization. In Proceedings of ICNN'95 - International Conference on Neural Networks, Perth, WA, Australia, pp. 1942-1948. https://doi.org/10.1109/ICNN.1995.488968

[12] Zhang, M., Peng, Y., Yang, M., Yin, Q.J., Xie, X. (2021). A discrete PSO-based static load balancing algorithm for distributed simulations in a cloud environment. Future Generation Computer Systems, 115: 497-516. https://doi.org/10.1016/j.future.2020.09.016

[13] Han, H.G., Liu, Y.C., Hou, Y., Qiao, J.F. (2024). Data-driven robust multimodal multiobjective particle swarm optimization. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 54(5): 3231-3243. https://doi.org/10.1109/TSMC.2024.3357872

[14] Guan, S.P., Li, X.Y. (2023). An interval multi-objective particle swarm optimization algorithm with niching technology for multimodal problems. In 2023 35th Chinese Control and Decision Conference (CCDC), Yichang, China, pp. 4018-4023. https://doi.org/10.1109/CCDC58219.2023.10326563

[15] Liao, J.X., Gao, Z.M., Sethumadhavan, S.M., Su, G.S., Zhao, J. (2025). An improved discrete multi-objective artificial protozoa optimizer for solving multi-objective knapsack problems. Swarm and Evolutionary Computation, 97: 102070. https://doi.org/10.1016/j.swevo.2025.102070

[16] Fairstein, Y., Kulik, A., Naor, J., Raz, D., Shachnai, H. (2022). An almost optimal approximation algorithm for monotone submodular multiple knapsack. Journal of Computer and System Sciences, 125: 149-165. https://doi.org/10.1016/j.jcss.2021.11.005

[17] Zhao, J.T., Luo, X.C. (2025). A population-based simulated annealing approach with adaptive mutation operator for solving the discounted {0-1} knapsack problem. Applied Soft Computing, 181: 113480. https://doi.org/10.1016/j.asoc.2025.113480

[18] Galli, L., Martello, S., Rey, C., Toth, P. (2025). The quadratic knapsack problem with setup. Computers & Operations Research, 173: 106873. https://doi.org/10.1016/j.cor.2024.106873

[19] Boonstra, G., van Eekelen, W.J.E.C., van Leeuwaarden, J.S.H. (2025). Robust knapsack ordering for a partially-informed newsvendor with budget constraint. Operations Research Letters, 58: 107202. https://doi.org/10.1016/j.orl.2024.107202

[20] Martello, S., Monaci, M. (2020). Algorithmic approaches to the multiple knapsack assignment problem. Omega, 90: 102004. https://doi.org/10.1016/j.omega.2018.11.013

[21] Yang, X.Y., Zhang, J.H., Hu, J.Q., Hu, J.Q. (2024). Nonparametric multi-product dynamic pricing with demand learning via simultaneous price perturbation. European Journal of Operational Research, 319(1): 191-205. https://doi.org/10.1016/j.ejor.2024.06.017

[22] Rodríguez-Mazo, J., Ruiz-Benítez, R. (2024). A multi-warehouse inventory model under hybrid-price-stock dependent demand and a bulk release pattern. Computers & Operations Research, 170: 106764. https://doi.org/10.1016/j.cor.2024.106764

[23] Wang, S., Baldacci, R., Liu, Q., Wei, L.J. (2025). EATKG: An open-source efficient exact algorithm for the two-dimensional knapsack problem with guillotine constraints. European Journal of Operational Research, 327(3): 735-753. https://doi.org/10.1016/j.ejor.2025.05.033

[24] Caserta, M., Voß, S. (2019). The robust multiple-choice multidimensional knapsack problem. Omega, 86: 16-27. https://doi.org/10.1016/j.omega.2018.06.014

[25] Galli, L., Martello, S., Rey, C., Toth, P. (2023). Lagrangian matheuristics for the quadratic multiple knapsack problem. Discrete Applied Mathematics, 335: 36-51. https://doi.org/10.1016/j.dam.2022.06.033

[26] Bos, H., Boucherie, R.J., Hans, E.W., Leeftink, G. (2024). Distributionally robust scheduling of stochastic knapsack arrivals. Computers & Operations Research, 167: 106641. https://doi.org/10.1016/j.cor.2024.106641

[27] Park, J., Han, J., Lee, K. (2024). Integer optimization models and algorithms for the multi-period non-shareable resource allocation problem. European Journal of Operational Research, 317(1): 43-59. https://doi.org/10.1016/j.ejor.2024.03.027

[28] Zhao, J.T., Hifi, M., Zhang, Y.L., Luo, X.C. (2024). An incremental method-based machine learning approach for max–min knapsack with multiple scenarios. Computers & Industrial Engineering, 190: 109984. https://doi.org/10.1016/j.cie.2024.109984

[29] Zhang, L., Liu, Z.S., Shan, W.X., Yu, B. (2023). A stabilized branch-and-price-and-cut algorithm for the waste transportation problem with split transportation.

Computers & Industrial Engineering, 178: 109143. https://doi.org/10.1016/j.cie.2023.109143

[30] Chen, S.M., Zeng, C.K., Zhang, Y., Tang, J.F., Yan, C.J. (2025). Lagrangian relaxation and branch-and-price algorithm for resource assignment problem in divisional seru systems. European Journal of Operational Research, 327(2): 432-449. https://doi.org/10.1016/j.ejor.2025.02.038

[31] Luo, K.P., Zhao, Q.H. (2019). A binary grey wolf optimizer for the multidimensional knapsack problem. Applied Soft Computing, 83: 105645. https://doi.org/10.1016/j.asoc.2019.105645