A Differential Pheromone Grouping Ant Colony Optimization Algorithm for the 1-D Bin Packing Problem

Aseel Ismael Ali*
aa1225@exeter.ac.uk
University of Exeter
Exeter, United Kingdom

Edward Keedwell
e.c.keedwell@exeter.ac.uk
University of Exeter
Exeter, United Kingdom

Ayah Helal
a.helal@exeter.ac.uk
University of Exeter
Exeter, United Kingdom

ABSTRACT

The bin packing problem (BPP) is a well-researched and important NP-hard problem with many contemporary applications (e.g. stock cutting, machine scheduling), which requires a set of items with variable sizes to be packed into a set of fixed-capacity containers. Many metaheuristic approaches have been successfully trialled on this problem, including evolutionary algorithms, ant colony optimization and local search techniques. The most successful variants of these approaches use grouping techniques whereby the algorithm considers sets of items together rather than as separate decision variables. This paper presents an Ant Colony Optimization integrated with a grouping technique and a novel differential pheromone procedure for bin packing. The proposed differential pheromone grouping ACO shows state-of-the-art results for ACO approaches in BPP and approaches the performance of the best evolutionary methods.

CCS CONCEPTS

• **Theory of computation → Bio-inspired optimization**; **Packing and covering problems**.

KEYWORDS

Ant colony optimization algorithm, One-dimensional bin packing, Grouping problem, Differential pheromone

ACM Reference Format:
Aseel Ismael Ali, Edward Keedwell, and Ayah Helal. 2024. A Differential Pheromone Grouping Ant Colony Optimization Algorithm for the 1-D Bin Packing Problem . In *Genetic and Evolutionary Computation Conference (GECCO '24), July 14–18, 2024, Melbourne, VIC, Australia.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3638529.3654074

1 INTRODUCTION

The BPP is one of the most challenging NP-hard problems in combinatorial optimisation which has been investigated and studied since the 1930s [19]. It has a wide range of applications such as data storage and cutting stock [1], printed circuit board design, assembly line balancing, capacitated vehicle routing, multiprocessor scheduling [8], file allocation [13], and virtual machine load balancing [4]. Moreover, the knapsack problem and the cutting stock problem are both special categories of BPP [22]. The rationale underlying the bin packing problem is to add items of varied sizes into a minimum number of bins having a fixed capacity without exceeding these capacities [19].

The challenge comes if the problem has a large set of items, which can incur significant computation time [19], and many algorithms have been proposed to find the optimal number of bins in a reasonable time. Examples of these include approximate algorithms, heuristic algorithms, mathematical optimisation algorithms, metaheuristic algorithms, and population-based algorithms [19]. However, nature-inspired algorithms have often been found to be superior to other algorithms due to their ability to solve different types of BPP [19].

Many types of BPP have been proposed such as the one-dimensional BPP; the two-dimensional BPP (2DBPP); the multi-dimensional BPP; the real-time BPP, and the conflicting items BPP [17]. Approximation algorithms are the first method applied for solving these two types of BPP, both online and offline [19]. Offline BPPs have the number of items and their size known at the initialisation step and can be solved using heuristic approaches, such as next-fit decreasing, best-fit decreasing, and first-fit decreasing [19]. One of the main steps of those approaches starts by sorting items in decreasing order [20]. Online BPP deals with items individually without prior knowledge about the next item [1]. Online BPP restrictions include packing items that cannot be changed later, making it more challenging to find the optimal solution in a reasonable time [20]. In this paper, we are only considering the offline one-dimensional BPP.

This paper develops D_pG-ACO a Differential pheromone Grouping Ant Colony Optimization approach which can solve BPP efficiently by using a grouping and a novel differential pheromone techniques. These contributions result in a modified approaches that show significant improvement over standard and enhanced ACO for BPP, setting a new standard for ACO in this field. Furthermore, the approach meets or closely approximates the performance of the best-in-class EA methods across a range of datasets without the expensive local search initialisation processes.

This paper is organised as follows: Section 2 provides background; Section 3 will cover the proposed D_pG-ACO approach; Section 4 represents the experimental setup, and the results; Section 5 discuss the D_pG-ACO approach in depth; and finally, Section 6 summarise the findings and highlights future works.

*2nd institution: Gifted Guardianship Committee, Minster of Education, Iraq

GECCO '24, July 14–18, 2024, Melbourne, VIC, Australia

ACM ISBN 979-8-4007-0494-9/24/07.
https://doi.org/10.1145/3638529.3654074

2 BACKGROUND

2.1 Bin Packing Problem Approaches

Recently, metaheuristic approaches have been augmented with a human-designed heuristic to solve both the offline and the online BPP, these include: bounded-space algorithms, the Next-Fit heuristic (NF), the First-Fit heuristic (FF), and the Best-Fit heuristic (BF) [1]. Variable-sized bin packing problem is another class of problem which is considered a real-world problem where the sizes of bins are different. The researchers propose many algorithms to solve this problem. Such as iterative first fit decreasing, iterative best fit decreasing [14], grouped genetic algorithm [10], variable neighbourhood search [12], and minimal bin stack [18].

2.2 ACO algorithms for BPP

Ant colony optimisation algorithms (ACO) are swarm intelligence optimisation algorithms based on the principles of ant foraging and communication. ACO has been widely applied and there have been a number of approaches to solve the BPP using ACO. For example, ACO algorithms have been used to solve conflict BPP by using the concept of graph colouring [24]. A further approach used to the multidimensional knapsack problem develops a probability selection equation by adding dynamic impact (dynamic heuristic) to improve convergence depending on the current situation, for example, a local search which improves the fitness value by 33.2% [23]. A Multi-Objective Ant Colony Optimization (MOACO) hybrid with FFD local search is used to solve the Multi-Objective Bin Packing Problem (MOBPP) by using different pheromone matrices for each objective and a variant of pheromone deposition in addition to using the Pareto dominance [15]. A further ACO technique, the Hybrid Constructive Heuristic Ant Colony Optimization (HCHACO) is used for variable-sized bin packing which hybridises ACO with mutation operator to improve the result which is gained from the ant's tour [11].

For 1-D BPP, an augmented exact method is used as local search with ACO to solve 1-D BPP to improve problem-solving efficacy [7, 16]. In this approach,$\mathcal{MAX}$-$\mathcal{MIN}$ Ant System ($\mathcal{MM}$**AS**) has been used with the Martello and Toth dominance criterion as a local search after each iteration to enhance the solution that has been produced by ants. Another ACO-based algorithm, Ant System (AS) has also been used to solve 1-D BPP, called AntPacking. In this approach, different encoding pheromones and employing a fitness function depend on the fullness of the bins in the ant's solution. The fitness function, in this approach, is the summation of only full bins for each ant. Consequently, the pheromone is deposited only on the path that leads to full bins [3].

2.3 Grouping Technique for BPP

The grouping method was first proposed to work with a genetic algorithm (GA) in 1996 by Falkenauer [9] to solve 1D BPP and was named the Hybrid Grouping Genetic Algorithm (HGGA). This method replaces a group of items in a bin with unpacked items using a special encoding scheme rather than placing items one by one [9]. Building on this idea, Quiroz et al. [21], in 2015, proposed a grouping genetic algorithm with controlled gene transmission (GGA-CGT). This new algorithm has additional features including pre-processing to generate an initial population depending on a special type of first-fit heuristic to balance exploration and exploitation. Borgulya [2], in 2021, adopted a similar approach that groups items according to the how frequently they are paired in the best solution, to create the hybrid evolutionary algorithm (HEA) which also makes use of local search to improve the quality of the solution.

3 METHODOLOGY

ACO algorithms are a good fit for solving BPP due to the dynamic construction graph which is used for representing the problems. The main phases of ACO include: initialisation, construction of ant solutions, and updating of the pheromone trail. However, to date, little work has been done to adapt the successful grouping approach seen in evolutionary algorithms to the ACO domain. Many different types of ACO exist, but the most popular are ant system (AS) and $\mathcal{MAX}$-$\mathcal{MIN}$ ant system ($\mathcal{MM}$**AS**), with the latter often outperforming the former in studies. In this section, we explain our proposed differential pheromone grouping approach which uses the $\mathcal{MM}$**AS** pheromone limits and depositing strategies as its basis. For further detail on the standard $\mathcal{MM}$**AS** approach, please see [6]. The full proposed D_pG-ACO is shown in Algorithm 1.

3.1 Problem description and formulation

A one-dimensional BPP is a problem having a set of items with different sizes that should be packed in fixed capacity bins in which the objective function is to minimise the number of used bins. The formulation of one-dimensional BPP is defined by Delorme [5] as follow: let $N = \{1, 2, ..., n\}$ be a set of items to be packed. $S = \{s_1, s_2, ..., s_n\}$ represent the sizes of these items, with s_i being the size of $item_i$. A number of possible bins from 1 to u. C is the maximum capacity of a bin. The binary decision variables can be defined as follows:

$$y_i = \begin{cases} 1 & \text{if bin } i \text{ is used in the solution;} \\ 0 & \text{otherwise} \end{cases}$$

where $i = 1, \ldots, u$

$$x_{ij} = \begin{cases} 1 & \text{if item } j \text{ is packed into bin } i; \\ 0 & \text{otherwise} \end{cases}$$

where $i = 1, \ldots, u; j = 1, \ldots, n$

According to the above, the BPP formulation is as follows:

$$min \sum_{i=1}^{u} y_i \tag{1}$$

$$\sum_{j=1}^{n} s_j X_{ij} \leq C y_i \quad (i = 1, \ldots, u) \tag{2}$$

where $x_{ij}, y_i \in \{0, 1\},\ \forall i \in u,\ \forall j \in n$

$$\sum_{i=1}^{u} x_{ij} = 1 \quad (j = 1, \ldots, n) \tag{3}$$

This formulation ensures that each item is packed into exactly one bin (Eq. 2), and the total size of items packed into each bin does not exceed the bin's capacity (Eq. 3). The objective function seeks to minimize the total number of used bins.

Within ACO, BPP can be formulated into an undirected weighted graph structure $G = (V, E)$. This graph is used to represent the relationship between items where vertices V are the items, and the edges E are the relationships between these items where $(i, j) = 1$ if the $item_i$ and $item_j$ are in the same bin. The pheromone value $\tau_{i,j}$ is used to weight the edge that connects the $item_i$ with $item_j$.

At the beginning of each ant's tour, all items belong to the 'Unpacked Items' list. While the ant packs items, those items are sequentially removed from this list. Simultaneously, a 'feasible item set' list, which is a subset of the 'Unpacked Items' list, is utilised. This subset only includes those unpacked items that can fit in the current bin, thus guiding the ants to choose from a feasible item set.

Each ant has a different starting point in the construction graph moving through it by picking up items one by one from the feasible item set to its solution according to the transition rule. After finishing its tour, the fitness function evaluates the solution. Following this, the pheromone trail in the standard algorithm is uniformly updated according to fitness, by depositing the pheromone on the ant's path in accordance with the optimality of the solution in addition to the evaporation of all the pheromones.

Algorithm 1 D_pG-ACO algorithm

```
procedure ACO ALGORITHM
    function INITIALIZATION
        Setting parameters
        Initialize the pheromone matrix.
        Creating a construction graph
    end function
    while i <= number of iterations or find global optima do
        for each ant in ants do
            Construct Ant Solutions
            Put ants randomly on the vertex.
            Apply the transition rule.
            if vertex ∈ group then
                Add combinations randomly.
            end if
            Calculate fitness function.
            for each edge in a construction graph do
                Update the pheromone trails according to Eq. 4
            end for
        end for
    end while
end procedure
```

3.2 G-ACO

The adaptation of the grouping technique in the ACO approach is highlighted in this section. After all ants finish their tour, any part of a solution that is generated by an ant that can fill a bin 100%, is considered as a group. Accordingly, each item will belong to one of three possible statuses: an item in a single group, an item in multiple groups or an item with no associated group. To adapt these changes to the construction graph, the following procedure is applied when an item is selected according to the transition rule:

(1) Item with no associated groups: the item is selected as before, probabilistically according to pheromone.
(2) Item with one group: the group will be selected as a singular item. The ant position will move to the last item in that group. All items in this group will be removed from the unpacked items list.
(3) Item with multiple different groups: the ant randomly chooses from the list of possible groups. The ant position will move to the last item in the selected group. The grouped items in the selected group will be removed from the unpacked items list.

At the end of the procedure, the ant next move using the transition rule to the available qualified items list from the current position. This approach exploits grouping to speed up the convergence to optimal or near-optimal solutions by ensuring that collections of items that completely fill a bin are preserved whilst reducing the size of the space required to be searched by the algorithm.

3.3 Dynamic Group Management

The construction graph in the proposed algorithm is dynamic due to the group strategy as this strategy adds and removes vertices. Thus, it produces a dynamic graph which needs to be managed through a group management system explained here.

An essential mechanism of this approach is the ability to dynamically form groups. This formation is primarily triggered when an ant creates a solution involving a solution section that leads to a full bin, indicating the potential for group formation. However, it's essential to ensure redundancy is avoided. To achieve this, a 'Group Existence Check' is initiated, searching for any section that matches the existing groups. If there is a match, the approach prevents the creation of a new group; if not, it proceeds with the formation of a new group and adds to the construction graph.

3.4 D_pG-ACO

Once an ant completes a tour, the pheromone trails τ_{ij}^t are updated through the evaporation and depositing process, as shown in Eq.(4).

$$\tau_{ij}^t = (1 - p)\tau_{ij}^{t-1} + \Delta\tau_{ij}^t \tag{4}$$

Where $0 < p \leq 1$ is the pheromone evaporating rate. $\Delta\tau_{ij}^t$ is depositing pheromone amount between $item_i$ and $item_j$ calculated according to Eq. (5).

To enhance the pheromone deposition process, we introduce a differential procedure. In contrast to the uniform deposition described above, this new method rewards sections of the solution differently based on their efficiency. A section of a solution leading to a completely filled bin receives a full pheromone deposit, whereas any other section that partially fills the bin receives a reduced pheromone deposit, as shown in Eq. (5).

$$\Delta\tau_{ij}^t = \begin{cases} f(x) & \text{if } i, j \in FB \\ f(x) \times p & \text{if } i, j \notin FB \\ 0 & \text{Otherwise} \end{cases} \tag{5}$$

Where FB is 100% fullness of the bin. $0 < p \leq 1$ is an evaporation rate. $f(x)$ is the fitness function of the best ant, calculated based on Falkenauer's fitness function [9].

Through this refined deposition methodology, the approach encourages solutions that prioritise full bins. By distinctly recognising and rewarding efficient packing, this method significantly limits the number of iterations required to identify the most promising regions within the search space.

3.4.1 Duplicated Items Handling. The items with duplicated sizes are managed individually to ensure they're accurately grouped. Such precision ensures that each group is unique.

3.4.2 New Group Creation and Management. As the algorithm progresses, there may arise situations where a particular set of items doesn't align with any existing group. In such instances, a new group is formed, and the algorithm establishes relationships between these items and the newly created group.

4 EXPERIMENTAL SETUP

The D_pG-ACO algorithms were coded in C# and run on a PC under Windows 10 Pro with a 2.40 GHz Core i5-1135G7 and 16.0 GB RAM. In the experimentation, 20 independent runs were conducted to ensure the robustness and reliability of the findings. The termination criteria for each run were either when the algorithm achieved a predetermined number of maximum iterations or discovered the global optima, whichever occurred sooner.

4.1 Benchmark Problem Sets

The proposed algorithms were evaluated using wide types of benchmark problem sets ranging from easy to challenging categories which are publicly available on http://or.dei.unibo.it/library/bpplib. Across these problem sets, a total of 1587 instances were examined. The details of these problem sets are shown in Table 1 which is adapted from [2].

4.2 Parameter Settings

To identify the best parameter settings for $\mathcal{MM}$**AS** and the proposed algorithms, several tests were performed to tune the parameters to find the best results. Various settings were applied to control parameters concerning the different methods for the proposed algorithm, as shown in Table 2. These parameters include: the number of ants, number of iterations, and evaporation rate.

Table 2: Control-parameter settings of proposed algorithms

Control parameter settings	Value
Number of ants	50
Number of iterations	10000
Evaporation rate	0.01

4.3 Results

In the results section, the first aim is to assess the performance of the proposed algorithm in comparison to standard ACO and the potential contribution of the new components to improve its performance. The second aim is to then compare the performance with other ACO algorithms and state-of-the-art algorithms.

4.4 Statistical analysis

To highlight the impact of new components on the standard ACO across different problem sets, different metrics have been used. Table 3 provides a comprehensive performance comparison between the standard ACO, the proposed G-ACO and D_pG-ACO capturing mean success rate (the mean percentage of runs that achieved the global optima from 20 independent runs of each instance in the problem sets), Root Mean Square Error (RMSE) calculated from the global optimum, and associated Standard Deviation (STD) across the problem sets.

As illustrated in Table 3, G-ACO and D_pG-ACO demonstrated significant performance improvements over $\mathcal{MM}$**AS** across all problem sets. Considering the mean success rate, a metric indicating the frequency at which the algorithms reached the global optima in 20 independent runs for each problem set instance, both the D_pG-ACO and G-ACO achieved a significantly high success rate of 78% and 77%, respectively, compared to the 47% of ACO, showing consist performance enhancements. Particularly, $\mathcal{MM}$**AS** struggled with 0% and 9% success in the Triplet and Scholl3 problem sets, respectively. In contrast, D_pG-ACO and G-ACO achieved 41% and 43% for the corresponding problem sets. Additionally, in Schwerin1, Schwerin2 and Scholl2 sets, D_pG-ACO and G-ACO outperformed $\mathcal{MM}$**AS** by achieving a 100% success rate, while $\mathcal{MM}$**AS** lagged with success rates of 45%, 87% and 81% for the same sets, respectively. In the Scholl1 problem set, all the algorithms perform comparably well, with success rates closely matched ($\mathcal{MM}$**AS** 95%, D_pG-ACO 96% and G-ACO 99%). While ACO had a low success rate at 12% in Wäscher, G-ACO and D_pG-ACO improved to 59% and 64%, respectively. Moreover, in the Uniform problem set, G-ACO and D_pG-ACO outperformed $\mathcal{MM}$**AS** , with a success rate of 75% and 76%, compared to the 46%.

Regarding the RMSE metric, both D_pG-ACO and G-ACO showed significant improvements over $\mathcal{MM}$**AS** . On average, ACO recorded an RMSE of 0.58, indicating larger errors. Conversely, G-ACO and D_pG-ACO achieved near global optima results with lower RMSE values of 0.30 and 0.29, respectively. Notably, in Schwerin1, Schwerin2 and Scholl2 problem sets, G-ACO and D_pG-ACO achieved an RMSE of 0, reflecting hitting the global optima in all 20 independent runs for all instances for these problem sets, while ACO recorded higher RMSE values (0.56 for Schwerin1, 0.15 for Schwerin2, and 0.39 for Scholl2). Concerning STD, which measures the consistency of the algorithm across runs, $\mathcal{MM}$**AS** generally had the lowest variability, with an average STD of 0.05. However, G-ACO and D_pG-ACO , despite showing more variability with the average STD metric values of 0.15 and 0.16, respectively, showed more stable and consistent behaviour in problem sets Schwerin1, Schwerin2 and Scholl2, where both achieved an STD of 0, contrasting with $\mathcal{MM}$**AS** 's performance.

Overall, The comparative analysis across the metrics indicates the superior performance of G-ACO and D_pG-ACO over $\mathcal{MM}$**AS** . These results show a valuable contribution of both grouping and differentiation of pheromone components in enhancing the $\mathcal{MM}$**AS** formulation. The results also show that the grouping is responsible for the majority of the performance improvements seen, but that

Table 1: Overview of Benchmark Problem Sets Used for Evaluation

Problem Sets	Number of Instances	Difficulty Level	Range of Number of Items	Capacity
Falkenauer Uniform	80	Moderate	[120,1000]	150
Falkenauer Triplet	80	Challenge	[60 – 501]	1000
Scholl-1	720	Easy	[50 - 500]	[100 - 150]
Scholl-2	480	Moderate	[50 - 500]	1000
Scholl-3	10	Challenge	200	100000
Schwerin1	100	Easy	100	1000
Schwerin2	100	Moderate	120	1000
Wäscher	17	Challenge	[57 - 239]	10000

Table 3: The table provides a comprehensive performance comparison between the standard ACO, the proposed G-ACO and D_pG-ACO capturing mean success rate (the mean percentage of runs that achieved the global optima from 20 independent runs of each instance in the problem sets), Root Mean Square Error (RMSE), and Standard Deviation (STD) across various problem sets.

Problem sets	$\mathcal{MM}$**AS**			G-ACO			D_pG-ACO		
	Mean Success rate	RMSE	STD	Mean Success rate	RMSE	STD	Mean Success rate	RMSE	STD
Triplet	0%	1	**0**	**41%**	**0.76**	0.47	**41%**	0.77	0.47
Uniform	46%	0.58	**0.09**	75%	0.37	0.24	**76%**	**0.36**	0.24
Schwerin1	45%	0.56	0.04	**100%**	**0**	**0**	**100%**	**0**	**0**
Schwerin2	87%	0.15	0.05	**100%**	**0**	**0**	**100%**	**0**	**0**
Wäscher	12%	0.88	**0.01**	59%	0.51	0.17	**64%**	**0.45**	0.15
Scholl1	95%	0.05	**0**	**99%**	**0.01**	0.05	96%	0.04	0.08
Scholl2	81%	0.39	0.01	**100%**	**0**	**0**	**100%**	**0**	**0**
Scholl3	9%	1.02	**0.18**	41%	0.72	0.25	**43%**	**0.71**	0.31
Average	47%	0.58	**0.05**	77%	0.30	0.15	**78%**	**0.29**	0.16

the differential pheromone does lead to significant performance improvements in the Wascher and Scholl3 problem sets and a slightly improved average overall.

Table 4 shows the statistical significance of the improvement of D_pG-ACO on the standard ACO using the Wilcoxon signed-rank test based on 20 runs across 149 instances. As expected, the results indicate that D_pG-ACO has a better average rank compared to $\mathcal{MM}$**AS** , illustrating its superiority across the problem sets. The computed Z-value of -6.71 signifies a substantial difference between the two algorithms. A negative Z-value points to D_pG-ACO having a ranking that is significantly lower and thus better performance than ACO. The p-value of 0.00001 strongly rejects the null hypothesis of no difference in performance between the two algorithms. This indicates that the improvement of D_pG-ACO over $\mathcal{MM}$**AS** is statistically significant.

Table 4: Table showing the Wilcoxon signed-rank test on the average of 20 runs over the 149 instances set.

	D_pG-ACO	$\mathcal{MM}$**AS**
Average Rank	**1.318**	1.681
Z-value	-6.71	
p-value	**0.00001**	

4.5 Comparative Analysis of Leading Algorithms with D_pG-ACO

Table 5 shows the comparative performance between D_pG-ACO and state-of-art algorithms: HACO, AntPacking, GGA-CGT and HEA, across a variety of problem sets. The evaluation used is taken from [21] based on hit rate (measuring the ability of finding the global optima of each instance in the problem sets at least once) and the maximum number of fitness evaluations (FEs) to get the global optima.

In this subsection, we display the comparative performance of leading algorithms, including HACO [7, 16], HGGA [10], GGA-CGT [21], HEA [2], AntPacking [3], and the proposed algorithm D_pG-ACO on different benchmarks. The primary metrics of focus hit rate and the number of fitness evaluations.

Both HGGA [10] and GGA-CGT [21] employ grouping techniques, like D_pG-ACO but are based on a genetic algorithm. HACO [7, 16] and AntPacking [3] are included in the comparison as contemporary ACO approaches that tackle these problem sets.

In the Triplet problem sets, GGA-CGT and D_pG-ACO both achieved a perfect hit rate of 100% with 50,000 FEs as well as HEA had a 100% hit rate. In contrast, HACO did not manage to achieve the global optima in all instances, with 420,000 FEs. Additionally, in the Uniform problem set, Both GGA-CGT and D_pG-ACO showed outstanding performance with a 100% hit rate, though D_pG-ACO

Table 5: a comparative performance between D_pG-ACO and state-of-art algorithms: HACO, AntPacking, GGA-CGT and HEA, across a variety of problem sets, evaluated based on hit rate (measuring the ability of finding the global optima of each instance in the problem sets at least once) and the maximum number of fitness evaluations (FEs) to get the global optima.

Problem sets	GGA-CGT [21]		HEA [2]		HACO [7, 16]		AntPacking [3]		D_pG-ACO	
	Hit Rate	FE	Hit Rate	FE	Hit Rate	FE	Hit Rate	FE	Hit Rate	FE
Triplet	**100%**	**50,000**	**100%**	N/A	0%	420,000	N/A	N/A	**100%**	**50,000**
Uniform	**100%**	50,000	**100%**	N/A	97%	**20,000**	**100%**	N/A	**100%**	40,000
Scholl1	**100%**	50,000	99.70%	N/A	N/A	N/A	N/A	N/A	**100%**	**30,000**
Scholl2	**100%**	5,000	**100%**	N/A	N/A	N/A	N/A	N/A	**100%**	**2,000**
Scholl3	**100%**	10,000	80%	N/A	N/A	N/A	N/A	N/A	88%	**11,000**
Schwerin1	**100%**	5,000	**100%**	N/A	N/A	N/A	N/A	N/A	**100%**	**4,000**
Schwerin2	**100%**	5,000	**100%**	N/A	N/A	N/A	N/A	N/A	**100%**	**3,500**
Wäscher	94.10%	500,000	**100%**	N/A	N/A	N/A	N/A	N/A	71%	**350,000**
Average	**99%**	84,375	97%	N/A	49%	220,000	100%	N/A	95%	**61,313**

N/A denotes that the metric value was not provided or the algorithm was not executed on this instance.

was more efficient, requiring only 40,000 FEs compared to GGA-CGT's 50,000. HEA also achieved 100% and HACO managed a 97% hit rate with 20,000 FEs.

In both Schwerin1 and Schwerin2 problem sets, GGA-CGT and D_pG-ACO achieved a 100% hit rate. Although D_pG-ACO was more efficient, requiring fewer function evaluations.

Regarding Wäscher, GGA-CGT had a 94.10% hit rate with a high FE count of 500,000. D_pG-ACO had a lower hit rate of 71% but required fewer FEs (350,000). HEA achieved a perfect hit rate.

For Scholl1 and Scholl2 problem sets, both GGA-CGT and D_pG-ACO achieved a perfect hit rate of 100%, though D_pG-ACO demonstrated greater efficiency, requiring only 30,000 and 2,000 FEs compared to GGA-CGT's 50,000 and 5000 for Scholl1 and Scholl2, respectively. However, despite HEA being slightly lower with a 99.70% hit rate in Scholl1, but performed worse in Scholl2 with an 80% hit rate. Moving to Scholl3, GGA-CGT upheld its 100% hit rate with 10,000 FEs, whereas D_pG-ACO achieved a lower performance of 88% hit rate, with a higher FE count of 11,000 compared to GGA-CGT. However, HEA's performance in Scholl3 dropped to an 80% hit rate, illustrating the increasing complexity and challenges presented by these problem sets. This analysis underscores D_pG-ACO 's consistent efficiency, particularly in the Scholl2 set, while highlighting the robustness of GGA-CGT across all Scholl problem sets.

On average, GGA-CGT led with a 99% hit rate but required a higher number of FEs (84,375). GGA-CGT employs local search and pre-processing procedures. HEA showed a promising 97% average hit rate, but FE data was not available. HEA also have integrated local search procedures. D_pG-ACO closely followed with a 95% hit rate and was more efficient with an average of 61,313 FEs. D_pG-ACO is competitive to both approaches without the use of any local search or pre-processing procedures. HACO, with the lowest average hit rate of 49%, required a significant number of FEs (220,000).

In summary, D_pG-ACO 's performance was shown to be superior in comparison to ant colony optimization algorithms, by achieving higher hit rates in fewer function evaluations and setting new benchmarks for ACO algorithm approaches. The consistent performance across a range of problem sets showed its effectiveness in solving BPP problems compared to other ant colony variants such as HACO and AntPacking. However, despite GGA-CGT and HEA demonstrating high hit rates, D_pG-ACO stands out for its balance between efficiency and effectiveness, achieving high hit rates with comparatively fewer function evaluations across most problem sets. This highlights D_pG-ACO 's capability to deliver optimal solutions efficiently, positioning it as a competitive algorithm in this comparison.

5 DISCUSSION

The results section demonstrates that both grouping and differential pheromone significantly influence ACO's performance in solving various BPP sets, requiring fewer fitness evaluations. The effectiveness of the grouping technique is recognised by its emphasis on combining items as a singular item. This strategy not only allows bins to be filled completely but also simplifies the decision-making process by treating these items as singular items. Consequently, this approach enables the ants to concentrate more effectively on other individual items, facilitating the discovery of optimal combinations. Additionally, it effectively reduces the search space by eliminating these combined items from the dynamic construction graph.

In parallel, the differential pheromone technique contributes to the overall performance by providing accumulative experience on preferred paths for the ants. It does so by depositing an increased amount of pheromone on parts of the solution that demonstrate promising item combinations while depositing a few pheromones in other parts. This differential distribution encourages ants to explore and identify effective paths throughout these parts. Such a strategy ensures a more accurate evaluation of the ants' solutions, thereby guiding them to the regions of interest within the search space in a few fitness evaluations.

6 CONCLUSION

This paper presents a novel D_pG-ACO approach that integrated a grouping technique and a differential pheromone procedure into

the ACO algorithm to create a new ACO formulation for bin packing. The analysis confirms the superiority of D_pG-ACO over the traditional ACO across various problem sets. Specifically, the statistical significance of the performance improvements, success rates, RMSE, and computational efficiency emphasize D_pG-ACO 's potential in this domain. When compared with other Ant Colony-based algorithms, D_pG-ACO 's efficiency, both in terms of optima hits and fitness evaluations, underlines its promise in addressing the bin packing problem effectively. The D_pG-ACO is the only ACO approach that shows the performance on all the problem sets. Also, the results are not only competitive but, in many scenarios, set new benchmarks for ACO. Unlike other state-of-the-art algorithms, the D_pG-ACO does not utilise local search or pre-processing, which distinguishes our approach. Future research will focus on improving D_pG-ACO by guiding ants through a pheromone in group choice, rather than random selection, and to assess any performance improvements this might deliver.

REFERENCES

[1] János Balogh, József Békési, György Dósa, Leah Epstein, and Asaf Levin. 2021. A New Lower Bound for Classic Online Bin Packing. *Algorithmica* 83 (7 2021), 2047–2062. Issue 7. https://doi.org/10.1007/s00453-021-00818-7

[2] Istvan Borgulya. 2021. A hybrid evolutionary algorithm for the offline Bin Packing Problem. *Central European Journal of Operations Research* 29 (6 2021), 425–445. Issue 2. https://doi.org/10.1007/s10100-020-00695-5

[3] Boris Brugger, Karl F Doerner, Richard F Hartl, and Marc Reimann. 2004. AntPacking – An Ant Colony Optimization Approach for the One-Dimensional Bin Packing Problem, In Evolutionary Computation in Combinatorial Optimization. *EvoCOP 2004* Lecture Notes in Computer Science, Vol 3004, 41–50.

[4] P Joseph Charles and U Lawrence Stanislaus. 2021. Secure Virtual Machine Migration using Ant Colony Algorithm. In *2021 Fifth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC)*. IEEE, Palladam, India, 1571–1575. https://doi.org/10.1109/I-SMAC52330.2021.9640743

[5] Maxence Delorme, Manuel Iori, and Silvano Martello. 2018. BPPLIB: a library for bin packing and cutting stock problems. *Optimization Letters* 12 (3 2018), 235–250. Issue 2. https://doi.org/10.1007/s11590-017-1192-z

[6] Marco Dorigo, Mauro Birattari, and Thomas Stutzle. 2006. Ant colony optimization. *IEEE Computational Intelligence Magazine* 1 (11 2006), 28–39. Issue 4. https://doi.org/10.1109/MCI.2006.329691

[7] Frederick Ducatelle and John Levine. 2002. Ant Colony Optimisation for Bin Packing and Cutting Stock Problems. http://www.informatics.ed.ac.uk/ use item-oriented approach.

[8] Ali Ekici. 2022. Variable-sized bin packing problem with conflicts and item fragmentation. *Computers and Industrial Engineering* 163 (1 2022), 107844. https://doi.org/10.1016/J.CIE.2021.107844

[9] Emanuel Falkenauer. 1996. A Hybrid Grouping Genetic Algorithm for Bin Packing. *Journal of Heuristics* 2 (1996), 5–30.

[10] E Falkenauer and A Delchambre. 1992. A Genetic Algorithm for Bin Packing and Line Balancing. *Proceedings of the IEEE1992 International Conference on Robotics and Automation*, 1186–1192.

[11] Yunhua Guo, Renshen Song, · Heng Zhang, · Chong Wang, and Jong Gye Shin. 2021. A hybrid algorithm for the variable-sized bin-packing problem of pipe cutting in offshore platform construction. *Journal of Marine Science and Technology* 27, 1 (2021), 1–17. https://doi.org/10.1007/s00773-021-00842-w

[12] Vera Hemmelmayr, Verena Schmid, and Christian Blum. 2012. Variable neighbourhood search for the variable sized bin packing problem. *Computers and Operations Research* 39 (5 2012), 1097–1108. Issue 5. https://doi.org/10.1016/j.cor.2011.07.003

[13] D S Johnsonf, A Demers:T:, J D Ullman, M R Gareyi, and R L Grahamii. 1974. WORST-CASE PERFORMANCE BOUNDS FOR SIMPLE ONE-DIMENSIONAL PACKING ALGORITHMS*. Issue 4.

[14] Jangha Kang and Sungsoo Park. 2003. Algorithms for the variable sized bin packing problem. *European Journal of Operational Research* 147 (6 2003), 365–372. Issue 2. https://doi.org/10.1016/S0377-2217(02)00247-3

[15] Oscar D. Lara and Miguel A. Labrador. 2010. A multiobjective ant colony-based optimization algorithm for the bin packing problem with load balancing, In IEEE Congress on Evolutionary Computation. *2010 IEEE World Congress on Computational Intelligence, WCCI 2010 - 2010 IEEE Congress on Evolutionary Computation, CEC 2010*, 1–8. https://doi.org/10.1109/CEC.2010.5586259

[16] J Levine and F Ducatelle. 2004. Ant colony optimization and local search for bin packing and cutting stock problems. *Journal of the Operational Research Society* 55 (2004), 705–716. https://doi.org/10.1057/palgrave.jors.2601771

[17] P. López. 2020. Introducing an approach based on an advanced hybrid model for the two-dimensional bin packing problem. *Annals of Electrical and Electronic Engineering* 2 (11 2020), 7–13. Issue 11. https://doi.org/10.21833/AEEE.2019.11.002

[18] Mohamed Maiza, Abdenour Labed, · Mohammed, and Said Radjef. 2013. Efficient algorithms for the offline variable sized bin-packing problem. *Journal of Global Optimization* 57 (2013), 1025–1038. https://doi.org/10.1007/s10898-012-9989-x

[19] Chanaleä Munien and Absalom E. Ezugwu. 2021. Metaheuristic algorithms for one-dimensional bin-packing problems: A survey of recent advances and applications. *Journal of Intelligent Systems* 30 (1 2021), 636–663. Issue 1. https://doi.org/10.1515/jisys-2020-0117

[20] Pooya Nikbakht. 2022. Applications and Extensions of The Bin Packing Problem. (2022).

[21] Marcela Quiroz-Castellanos, Laura Cruz-Reyes, Jose Torres-Jimenez, Claudia Gómez S., Héctor J.Fraire Huacuja, and Adriana C.F. Alvim. 2015. A grouping genetic algorithm with controlled gene transmission for the bin packing problem. *Computers and Operations Research* 55 (2015), 52–64. https://doi.org/10.1016/j.cor.2014.10.010 bin-oriented approach.

[22] A. Silva-Gálvez, E. Lara-Cárdenas, I. Amaya, J. M. Cruz-Duarte, and J. C. Ortiz-Bayliss. 2020. *A Preliminary Study on Score-Based Hyper-heuristics for Solving the Bin Packing Problem*. Vol. 12088 LNCS. Springer, 318–327. https://doi.org/10.1007/978-3-030-49076-8_30

[23] Jonas Skackauskas, Tatiana Kalganova, Ian Dear, and Mani Janakiram. 2022. Dynamic impact for ant colony optimization algorithm. *Swarm and Evolutionary Computation* 69 (3 2022). https://doi.org/10.1016/j.swevo.2021.100993 improve probability calculation equation by adding calculate the dynamic fitness called dynamic impact.

[24] Ye Yuan, Yi Jun Li, and Yan Qing Wang. 2014. An improved ACO algorithm for the bin packing problem with conflicts based on graph coloring model. *International Conference on Management Science and Engineering - Annual Conference Proceedings*, 3–9. https://doi.org/10.1109/ICMSE.2014.6930200