Evaluating Alternative Continuous Metaheuristic Solution Encodings for Routing Problems with Multiple Drones with Interceptions

Sarah Dillon and Jacomine Grobler(✉)

Department of Industrial Engineering, Stellenbosch University, Stellenbosch, South Africa
sarahdillon999@gmail.com, jacomine.grobler@gmail.com

Abstract. The integration of trucks and drones has emerged as a promising approach for improving the efficiency of last-mile delivery operations. This paper studies the travelling salesman problem with multiple drones and interception, in which a truck carries multiple drones that may be launched from the truck and retrieved while the truck is in motion. Recent work has shown that continuous metaheuristic algorithms are effective for this problem class; however, the impact of solution encoding on computational performance has not yet been systematically examined. Two alternative solution encodings are compared in this paper: an assignment strategy that explicitly allocates each customer to a specific vehicle, and a more general encoding that assigns customers to either the truck or the next available drone without prespecifying which drone. Both encodings are embedded within identical continuous metaheuristic frameworks and evaluated using two algorithms across 15 benchmark instances with varying spatial distributions and sizes. Computational results demonstrate that the more general encoding consistently achieves substantial reductions in runtime while maintaining comparable solution quality. The findings highlight the critical role of encoding design in continuous optimisation for hybrid vehicle routing problems.

Keywords: Drone delivery · TSP with multiple drones with interceptions · Evolutionary algorithms · Swarm intelligence · Last-mile

1 Introduction

The optimisation of last mile delivery logistics has become increasingly relevant due to the rapid growth of e-commerce in recent years. E-commerce, otherwise known as online shopping, refers to the use of the internet for commercial transactions [1]. With widespread internet access and the growth of smartphone and computer usage, retail companies leverage e-commerce to reach consumers more

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2027
O. Gervasi et al. (Eds.): ICCSA 2026, LNCS 16769, pp. 538–555, 2027.
https://doi.org/10.1007/978-3-032-30494-0_34

effectively. The surge in online shopping has led to an increased volume in the last mile, and this surge is accompanied by heightened customer expectations. These pressures have intensified interest in delivery systems that make use of drones, or *unmanned aerial vehicles* (UAV), in order to improve operational efficiency beyond what is achievable with truck-only fleets.

Drones offer high travel speeds and the ability to bypass road congestion; however, their limited payload capacity and flight endurance prevent them from fully replacing ground vehicles [2]. Consequently, research has increasingly focused on integrated truckâĂŞdrone systems, which have been shown to reduce delivery times and operational costs when compared to conventional truck-based routing [3]. The growing practical relevance of such systems is reflected in recent industry deployments. Since the first commercial autonomous drone delivery in 2016 [4], adoption has accelerated steadily, with thousands of deliveries performed daily by 2022 and hundreds of thousands annually by 2023 [5,6]. Major logistics providers, including Amazon, Wing, Flytrex, Alibaba, JD Logistics, and Walmart, have incorporated drone technology into their delivery strategies [7].

From an optimisation perspective, many coordinated truckâĂŞdrone delivery problems have been formalised as extensions of the *travelling salesman problem with a drone* (TSPD). The original TSPD was introduced by Agatz *et al.* [8], and extended to allow drones to intercept the moving truck between customer locations [9]. Allowing for multiple drones onboard the truck to intercept the truck en route yields the *travelling salesman problem with multiple drones and interception* (TSPmDi) [10].

Existing continuous metaheuristic approaches to the TSPmDi rely on a solution encoding in which each customer is explicitly assigned either to the truck or to a specific drone [10]. This encoding, referred to as *Solution Encoding 1* (SE1), requires repeated discretisation across all drones and customers, introducing significant computational overhead. This paper investigates an alternative modelling approach, termed *Solution Encoding 2* (SE2), in which customers are assigned either to the truck or to the next available drone, without pre-selecting a specific drone. The impact of this encoding on algorithmic performance is evaluated using 15 benchmark datasets, with varying sizes and spatial distributions. A *self adaptive neighbourhood search differential evolution* (SaNSDE) algorithm and a *guaranteed convergence particle swarm optimisation* (GCPSO) algorithm, both combined with a *nearest neighbour heuristic for initial solutions* (NNHis) operator, is applied to the TSPmDi for both SE1 and SE2. By modifying only the assignment encoding, the study isolates the performance and computational effect of encoding design.

To the authors' knowledge, this is the first study to explicitly quantify how solution encoding influences the performance and computational efficiency of continuous metaheuristics for the TSPmDi. This paper demonstrates that encoding-level design choices can dominate runtime performance without affecting solution quality.

The paper is organised as follows: Sect. 2 reviews related work, and Sect. 3 outlines the metaheuristic algorithms used in this paper. The problem is

described in Sect. 4. The proposed continuous metaheuristic algorithm is introduced in Sect. 5. The empirical evaluation of the metaheuristic algorithms is documented in Sect. 6. A conclusion of the paper is presented in Sect. 7.

2 Literature Review

Recent research has increasingly focused on cooperative truck-drone systems for last-mile delivery. Several comprehensive survey papers review developments in this area, covering modelling variants, solution approaches, and emerging applications [7,11–17].

Two well-studied single-truck and single-drone variants are the *travelling salesman problem with a drone* (TSPD) [8] and the *flying sidekick travelling salesman problem* (FSTSP) [18,19]. Moremi [9] extended the TSPD to define the *TSPD with interception* (TSPDi), which allows the drone to intercept the truck while the truck is in motion between customer nodes. An *ant colony system* (ACS) algorithm was developed to solve this problem. Optimal solutions for the TSPDi were later obtained by Knoetze [20], although the computational time limit of eight hours resulted in no optimal solutions for instances containing 50 or more nodes. Ernst [21] subsequently proposed continuous metaheuristic algorithms for the TSPDi, which outperformed both the ACS approach and the exact method, indicating the significant potential of continuous metaheuristics for this problem.

Most *TSP with multiple drones* (TSPmD) problems in the literature require the truck and drones to rendezvous at customer locations after completing their respective deliveries with the first-arriving vehicle waiting for the other. However, increasing research is being conducted on models that allow drones to intercept the truck at non-customer locations. Table 1 summarises existing interception-based systems that consider multiple drones.

These studies can be broadly divided into two categories. The first assumes predefined launch and rendezvous locations. Betti *et al.* [22,23] allowed each customer location to be associated with corresponding launch and rendezvous points that may replace customer nodes for drone operations, while Salama [24] and Han *et al.* [25] considered a variant with predefined truck stop locations from which drones may be launched or retrieved. The second category permits drone launches and retrievals anywhere along the truck's route between two customer nodes. Within this group, some studies restrict the truck to a road network. Lin *et al.* [26,27] and Narayan [28] adopted this assumption. Other studies removed the road network restriction entirely [10,29,30]. The Euclidean distance is the most common distance metric used in the literature [10,23–27,30].

No existing work investigates alternative solution encodings for continuous metaheuristics or investigates the effect of adding more drones to the system when solving the TSPmDi with continuous metaheuristics. In particular, no existing work evaluates how different continuous solution encodings affect feasibility enforcement, repair behaviour, and runtime efficiency, when solving the TSPmDi.

Table 1. Summary of relevant work on truck and multiple drone interception systems

	Reference	Drones	Nodes	Method	Objective
Predefined locations	Betti *et al.* [22]	1, 3, 5	25–100	Greedy heuristics	Maximise reward
	Betti *et al.* [23]	1, 3, 5	25–100	Knapsack approximation	Minimise time
	Han *et al.* [25]	2	10–50	Variable Neighbourhood Search (VNS)	Minimise waiting
	Salama *et al.* [24]	4	8–59	Simulated annealing, VNS, rule-based heuristics	Minimise time
Interception Allowed	Lin *et al.* [27]	2	10–100	Genetic Algorithm (GA)	Minimise time
	Lin *et al.* [26]	1–5	20–100	Hybrid GA/PSO	Minimise time
	Narayan [28]	1–11	28	Exact methods	Minimise time
	Li [29]	2	5–30	Adaptive Large Neighbourhood Search (ALNS)	Minimise cost
	Thomas [30]	3, 6	10–100	Heuristics	Minimise time
	Dillon *et al.* [10]	2	10–250	Continuous metaheuristics	Minimise time
	This Paper	**2–4**	**10–250**	**Continuous metaheuristics**	**Minimise time**

3 Algorithm Background

The SaNSDE and GCPSO algorithms are explained in this section, along with the *nearest neighbour heuristic for initial solutions* (NNHis) initialisation strategy.

3.1 Self Adaptive Neighbourhood Search Differential Evolution

Differential evolution (DE) is a population-based, stochastic optimisation algorithm that relies on difference vectors to determine step sizes between individuals in the population. These differences drive the search toward improved solutions. The difference mutation operator enables step sizes to adapt automatically to the objective function landscape [31]. In the early stages of the search, when individuals are widely dispersed across the search space, larger step sizes promote exploration and help avoid local optima. As the population converges, smaller step sizes emerge, allowing for refined exploitation around promising regions. The SaNSDE algorithm applied in this paper was introduced by Yang *et al.* [32]. The algorithm combines the best features of the *Neighbourhood Search DE* (NSDE) [32] and the *Self-adaptive DE* (SaDE) [33] algorithms.

3.2 Guaranteed Convergence Particle Swarm Optimisation

Particle swarm optimisation (PSO) is a swarm intelligence optimisation algorithm developed by Kennedy and Eberhart [34], inspired by the collective move-

ment of bird flocks. The algorithm iteratively improves candidate solutions by moving particles through a multi-dimensional hyperspace, where each particle's position represents a candidate solution. Velocity and displacement updates are applied to guide particles to a new position, thereby obtaining new solutions in the search space [35]. At any time step t, a particle's velocity is determined by its previous velocity, its personal best position, and the global best position found by the swarm. Convergence occurs as particles collectively move towards these best-known positions.

To prevent stagnation, the *guaranteed convergence PSO* (GCPSO) introduces a random search by the global best particle around the global best particle position [36]. The area of the search adapts according to the number of consecutive successful or unsuccessful attempts to improve the objective function value.

3.3 The Nearest Neighbour Heuristic for Initial Solutions

The NNHis initialisation strategy introduces six high-quality initial solutions to the initial population based on nearest-neighbour construction using truck-only and alternating truck-drone assignments [21,37]. The NNHis operator is adapted from [21] for the TSPmDi by introducing the following six initial solutions into the initial population of the algorithm:

- The first solution is generated using a *nearest neighbour heuristic* (NNH). The depot is considered the first node. The node with the smallest Euclidean distance from the depot is chosen as the second node to visit. Thereafter, the node closest to the second node is allocated as the next stop. This process is continued until all nodes have been allocated and the final node is again assigned as the depot. All nodes are to be visited by the truck, and the drone is stationary.
- The second solution is generated using two parallel NNHs. Two different routes are generated, both starting from the depot. The nodes are added to the two routes in alternating order, first to the one route and then to the other. After allocating all nodes, the second route is reversed and connected onto the end of the first half of the route to ensure that the depot is visited first and last. Again, all nodes are visited by a truck.
- The third solution is generated using two parallel NNHs, as in the second solution. However, in the third solution, nodes are continuously added to either route depending on which endpoint is nearest to any unallocated node. The second route is again reversed after all nodes are allocated and connected to the first route to construct one full route where the depot is visited at the start and end of the route with truck-only nodes.
- The fourth solution is generated using an *alternating nearest neighbour heuristic* (ANNH). The first node, which is the depot, is allocated to the truck. Thereafter, the node with the smallest distance to the first node is allocated to a drone. The nearest node to the drone node is then chosen and allocated to the next free drone. This procedure is continued until all the drones have been assigned, after which a node is allocated to the truck again and the drone

assignments restart. This process repeats until all the nodes are assigned and the last node is again allocated as the depot.
- The fifth solution is generated using two parallel ANNHs, combining solutions two and four, by allocating nodes to the truck and thereafter, to all the drones before again allocating nodes to the truck. This process is continued until all nodes have been added. The second half of the route is again reversed and added onto the end of the first half of the route to ensure that the depot is visited first and last.
- The sixth solution is generated similarly to the third solution, except that nodes are now allocated to the truck and thereafter, to all the drones before again allocating a node to the truck (thus combining solutions three and four).

4 Problem Description

The TSPmDi considers a delivery system in which a single truck carries multiple drones to serve a set of customer locations. Drones can rejoin the truck either at a customer node or by intercepting the truck while the truck is en-route [10]. An example network and feasible route for the TSPmDi with two drones is shown in Fig. 1.

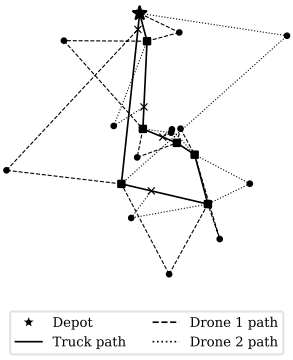

Fig. 1. TSPmDi example.

4.1 Network

The problem is defined on a complete, undirected graph, where the nodes represent the depot and customer locations, and edges represent feasible travel paths for both the truck and the drones. Distances between customer nodes are calculated using Euclidean distance. Drones are assumed to fly in straight lines between locations. The truck operates without road network constraints.

4.2 Vehicles

The system consists of one truck carrying a fixed number of identical drones. Each drone can carry a single package and is assumed to have unlimited range. After completing a delivery, a drone must return to the truck. Both the truck and drones travel at constant speeds, with drones assumed to move at twice the speed of the truck.

4.3 Customers

Each customer must be served exactly once, either by the truck or by a drone. A truck node is defined as a customer node that is assigned to be serviced by the truck. A drone node is defined as a customer node that is assigned to be serviced by a drone.

4.4 Routes

The delivery path consists of a truck route and a set of drone sub-tours. The truck route departs from the depot, visits all the assigned truck nodes, and returns to the depot. Drone sub-tours occur when a drone is launched from the truck to service an assigned drone node before returning to the truck. The vehicle routes, including the truck route and all drone sub-tours, and the assignment of customer nodes to vehicles are determined as part of the optimisation process. Figure 2 illustrates a drone sub-tour with an interception.

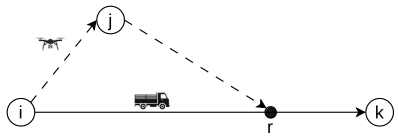

Fig. 2. Drone sub-tour with interception [10].

Multiple drones can be launched from the same truck node, and the truck may retrieve multiple drones either at a node or while travelling between nodes. After servicing a customer, a drone may either return directly to the next truck node or intercept the truck along its route.

Interception occurs when a drone's travel time from node i to j to k, denoted T_{ijk}, is less than the truck's travel time from node i to k, denoted T_{ik}. In this case, the interception location is calculated such that the drone rejoins the truck along the truck route at a point, r, on the edge (i, k). If interception is not feasible, the drone flies directly to node k. The total drone sub-tour time for servicing node j while the truck travels from node i to k, T^D_{ijk}, is given by

$$T^D_{ijk} = T_{ij} + s + \min\left(T_{jk},\ t^{\text{int}}_{ijk}\right), \tag{1}$$

where s denotes the service time at a customer node, T_{jk} denotes the time for the drone to travel directly from node j to the truck node k, and t_{ijk}^{int} denotes the feasible interception time.

Waiting time occurs if the truck arrives at the customer node k before all the drones assigned to the edge (i, k) have completed their sub-tours. In this case, the truck is required to wait for the remaining drone(s) to arrive at node k. The waiting time per edge (i, k), w_{ik}, is defined as

$$w_{ik} = \max\left(0,\ \max_{j \in \mathcal{D}_{ik}} \left(T_{ijk}^{D} - (T_{ik} + s)\right)\right), \tag{2}$$

where $\mathcal{D}_{ik}$ is the set of customer nodes served by drones during truck edge (i, k).

4.5 Objective

The objective is to minimise the total completion time of the delivery route. This total time includes the truck's travel time, the service times at the customer nodes, and the waiting time incurred. The overall objective is defined as

$$\min Z = \sum_{(i,k) \in \mathcal{E}_T} T_{ik} + \sum_{k \in \mathcal{N}_T \setminus \{0\}} s + \sum_{(i,k) \in \mathcal{E}_T} w_{ik}, \tag{3}$$

where $\mathcal{E}_T$ is the set of edges (i, k) traversed by the truck route, $\mathcal{N}_T$ is the set of nodes visited by the truck, and w_{ik} is the maximum delay introduced by any drone operating on edge (i, k), as defined in Eq. 2.

5 Proposed Algorithm

The proposed algorithm integrates the NNHis initialisation strategy, a TSPmDi problem representation, and an interchangeable metaheuristic framework. An activity diagram of the algorithm is shown in Fig. 3.

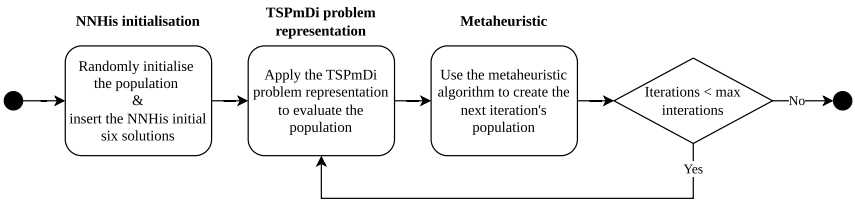

Fig. 3. Activity diagram of the proposed algorithm.

The algorithm is initialised by generating N real-valued vectors, each representing a candidate solution with $2(n)$ decision variables bounded between l_b and u_b, where n is the number of nodes excluding the depot. These decision variables

are iteratively updated throughout the optimisation process. To enhance initial solution quality, the NNHis strategy introduces six high-quality initial solutions to the initial population, constructed using nearest-neighbour heuristics based on truck-only routes and alternating truck-drone assignments.

Fitness evaluation is performed by mapping the continuous values to routes using priority-based mapping [38]. The first n values of each vector are sorted in descending order to determine the visiting sequence of the nodes. The remaining n values determine which vehicle services each node by applying either Solution Encoding 1 or Solution Encoding 2. Repair mechanisms are applied to ensure feasible solutions. The specific mapping and repair procedures for Solution Encoding 1 and Solution Encoding 2 are outlined in Sect. 5.1 and 5.2. This paper evaluates two metaheuristic algorithms: SaNSDE NNHis and GCPSO NNHis.

5.1 Solution Encoding 1

In Solution Encoding 1, the interval $u_b - l_b$ is divided into $d+1$ equal segments, where d is the number of drones in the system. Decision variable values in the first segment assign nodes to the truck, the second to drone 1, the third to drone 2, and so on. This encoding explicitly encodes the assignment of each node to a specific vehicle.

An infeasible solution occurs when a drone is assigned to more than one node between two consecutive truck nodes, as each drone can only serve one node in this interval. In such cases, the node remains a drone node but is reassigned to another available drone. If there are no available drones, then the node is reassigned to the truck. Importantly, repair operations involve simple reassignment and do not require re-sorting or re-discretisation of decision variables.

5.2 Solution Encoding 2

In Solution Encoding 2, the interval $u_b - l_b$ is divided into two equal segments. Values in the first segment assign nodes to the truck, while values in the second segment assign nodes to a drone. Unlike Solution Encoding 1, this encoding does not specify a specific drone at the assignment stage. A customer node that has been assigned to a drone may be assigned to any available drone.

An infeasible solution occurs when the number of consecutive drone-assigned nodes between two truck nodes exceeds the number of drones in the system, as each drone can serve only one node in this interval. In such cases, the $d+1^{th}$ node is reassigned to the truck to restore feasibility. This encoding separates node assignment from drone identity, significantly simplifying the mapping process and reducing encoding-induced infeasibility.

6 Empirical Evaluation

SE1 and SE2 are evaluated using 15 datasets from Bouman and Schmidt [39], featuring varying sizes and coordinate spreads. Single-centred datasets consist of

a central node around which most nodes are scattered; double-centred datasets contain two central locations with nodes distributed around both locations; and uniform datasets contain nodes that are spread out uniformly with no centre. Each distribution has five datasets with the following sizes: 10, 20, 50, 100, and 250 nodes.

Results for Solution Encoding 1 are taken directly from [10], while Solution Encoding 2 is introduced and evaluated in this paper. Both algorithms were run on each dataset for 30 independent runs of 1000 iterations each. The datasets are labelled as follows: '*distribution-reference-size*'. The uniform distributions are referred to as U, the single-centred distributions are referred to as SC, and the double-centred distributions are referred to as DC. For example, the label '*U-51-n10*' refers to the dataset with a uniform distribution, a reference number of 51, and a problem size of 10. The experiment parameters are provided in Table 2.

Table 2. Experiment parameters

Parameter	Value
Truck travel speed	10 distance units / time unit
Drone travel speed	20 distance units / time unit
Service time	0.1 time units
Population size (N)	100
Decision variable lower bound (l_b)	0
Decision variable upper bound (u_b)	200
Number of drones (d)	$2, 3, 4$

For Solution Encoding 2, the hyperparameters for the SaNSDE NNHis and GCPSO NNHis algorithms were tuned using Optuna, an automatic hyperparameter optimisation software framework [40]. The objective function was defined as the mean performance across all 15 datasets. A budget of 30 trials was allocated, and the *tree-structured parzen estimator* (TPE) sampler was employed to minimise the average cost function. The tuning was performed on the 2-drone configuration to establish a baseline set of robust parameters which were further evaluated on the 3 and 4-drone configurations to test the scalability of the algorithms. The hyperparameters of the SaNSDE NNHis and GCPSO NNHis algorithms are provided in Table 3 and Table 4.

6.1 SE1 vs. SE2 for the 2-Drone Configuration

The SaNSDE NNHis and the GCPSO NNHis algorithms were applied to both SE1 and SE2 respectively, for the 2-drone configuration. The means of the resulting delivery times and standard deviation across the 30 runs are presented in

Table 3. Hyperparameter search space and optimized values for SaNSDE NNHis

Parameter	Description	Search Range	Final Value
F	Differential weight	[0.1, 1.0]	0.991
C	Crossover probability	[0.1, 1.0]	0.801
p_{wr}	Mutation strategy probability	[0.1, 0.9]	0.162

Table 4. Hyperparameter search space and optimized values for GCPSO NNHis

Parameter	Description	Search Range	Final Value
c_1	Cognitive acceleration coefficient	[0.5, 3.0]	0.558
c_2	Social acceleration coefficient	[0.5, 3.0]	0.683
$v_{\max}$	Velocity clamping	[1.0, 10.0]	1.023

Table 5. The fastest delivery time per dataset is bolded. To assess the performance of the encodings across the datasets, a Wilcoxon signed-rank test was conducted. The test assessed whether there is a statistically significant difference between the aggregated results for SE1 and SE2 across all 15 datasets. As shown in Table 6, the test yielded a p-value of 0.5995, which is greater than the significance level of 0.05. Therefore, the results indicate that there is not a statistically significant difference between SE1 and SE2 for the two-drone configuration. With only two drones, the assignment space is relatively constrained, limiting the impact of encoding flexibility on solution quality. Figure 4 presents the final SE2 routes generated by the SaNSDE NNHis algorithm for the datasets with n = 20 to validate the legitimacy of the routes.

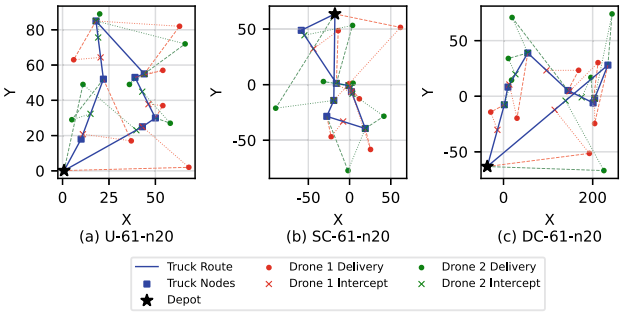

Fig. 4. SaNSDE NNHis routes for size 20 datasets.

6.2 Computational Performance: SE1 vs. SE2

This section analyses the computational performance of the two solution encodings, SE1 and SE2, across different spatial distributions and drone fleet sizes.

Table 5. Mean travel time and standard deviation for SE1 and SE2 for $d = 2$

Dataset	SE1 (Benchmarks)		SE2 (Results)	
	SaNSDE	GCPSO	SaNSDE	GCPSO
Uniform				
U-51-n10	**22.63 (0.15)**	23.32 (0.66)	22.66 (0.15)	22.86 (0.25)
U-61-n20	22.01 (0.36)	24.15 (0.48)	**21.79 (0.27)**	23.91 (1.15)
U-71-n50	52.40 (0.00)	**52.28 (0.36)**	55.72 (0.60)	53.18 (1.07)
U-91-n100	74.19 (0.00)	74.07 (0.29)	75.24 (0.85)	**73.77 (1.24)**
U-1-n250	120.12 (0.00)	**115.84 (0.24)**	119.45 (0.88)	117.09 (0.95)
Single centre				
SC-51-n10	23.10 (0.23)	24.18 (0.31)	**23.06 (0.10)**	24.48 (0.17)
SC-61-n20	30.82 (1.20)	31.74 (1.46)	**30.73 (0.80)**	35.47 (1.05)
SC-71-n50	65.24 (1.28)	58.39 (3.10)	52.93 (0.00)	**52.85 (0.46)**
SC-91-n100	123.71 (0.00)	119.43 (3.65)	**109.21 (3.17)**	111.38 (1.86)
SC-1-n250	192.73 (0.00)	185.14 (0.51)	181.28 (2.38)	**178.24 (3.65)**
Double centre				
DC-51-n10	33.63 (0.14)	34.15 (0.24)	**33.58 (0.00)**	34.67 (0.45)
DC-61-n20	59.07 (0.55)	60.46 (0.99)	**58.83 (0.93)**	62.26 (1.59)
DC-71-n50	114.43 (0.00)	111.22 (3.15)	**100.72 (0.00)**	102.49 (4.18)
DC-91-n100	153.38 (0.00)	151.29 (2.21)	140.43 (0.03)	**137.46 (2.69)**
DC-1-n250	249.76 (0.00)	**246.63 (4.00)**	258.64 (4.70)	249.97 (4.46)

The datasets with $n = 50$ are used for the computational evaluation. Table 7 reports the average CPU runtimes while Fig. 5 illustrates how runtime scales with the number of drones. Across all datasets, fleet sizes, and both metaheuristics applied, SE2 consistently outperforms SE1, with runtime reductions ranging from approximately 8% to over 17%. This performance gap is observed for both SaNSDE NNHis and GCPSO NNHis, indicating that it arises from structural differences between the encodings rather than algorithm-specific effects.

Empirical profiling results show that a dominant contributor to SE1's runtime is the discretisation step used to assign nodes to specific drones. SE1 maps decision values into $d + 1$ segments to assign specific drones, a process that is absent in SE2. For the $n = 50$ instances, this operation accounts for approximately $16-18\%$ of the total CPU runtime. This cost is incurred at every solution evaluation and remains constant across spatial distributions and fleet sizes.

SE2 eliminates this overhead entirely by replacing discretisation with a fixed interval assignment scheme, where the priority space is explicitly partitioned into one truck interval and one drone interval. As a result, SE2 avoids the repeated discretisation cost that dominates the computational profile of SE1.

To determine whether feasibility enforcement could offset this advantage, Fig. 6 reports the total number of repair operations executed by each encoding.

Table 6. Wilcoxon signed-rank test results for the aggregate performance comparison between SE1 and SE2 for $d = 2$

Metric	Value
Number of datasets (n)	15
Wilcoxon Statistic (W)	50.00
p-value	0.5995
Significance Level (α)	0.05
Result	**No Significant Difference**

Table 7. Average CPU runtime across 30 runs for $n = 50$ datasets

	2 Drones				3 Drones				4 Drones			
	SE1		SE2		SE1		SE2		SE1		SE2	
Instance	SaNSDE	GCPSO	SaNSDE	GCPSO	SaNSDE	GCPSO	SaNSDE	GCPSO	SaNSDE	GCPSO	SaNSDE	GCPSO
Uniform-71-n50	319.29	288.69	263.25	**246.59**	330.11	291.25	283.55	**273.23**	347.31	299.98	296.00	**248.48**
Singlecentre-71-n50	318.36	281.15	272.49	**255.81**	334.75	292.04	291.03	**270.53**	344.73	299.68	270.16	**256.08**
Doublecentre-71-n50	321.15	288.40	276.20	**260.11**	331.51	295.72	283.84	2 **65.57**	349.03	304.98	271.90	**261.80**

The results show that differences in feasibility repair behaviour do not explain the observed runtime differences.

For $d = 2$, SE2 occasionally produces more repairs than SE1, particularly under SaNSDE NNHis. The additional repairs occur because the coarse binary assignment (truck vs. drone), combined with random initialisation, increases the likelihood of exceeding drone capacity between consecutive truck nodes. However, these additional repairs are computationally inexpensive and do not outweigh the savings from removing discretisation.

As the fleet size increases to $d = 3$ and $d = 4$, a clear shift occurs. SE2 consistently generates fewer repairs than SE1 across all spatial distributions and both algorithms. This behaviour arises from the differing assignment mechanisms. For SE2, increasing the number of drones expands the feasible assignment space, providing greater flexibility and reducing forced reassignments to the truck. In contrast, SE1's discretisation-based mapping becomes increasingly sensitive as

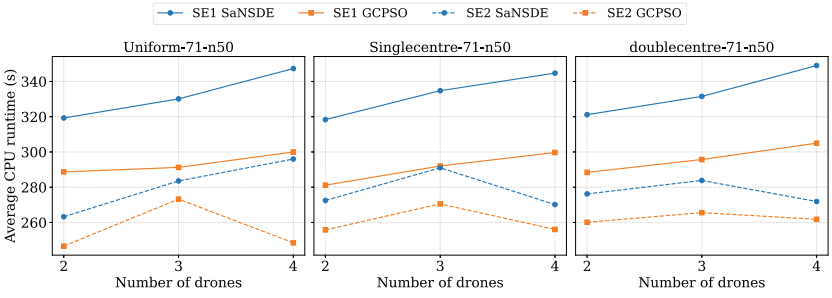

Fig. 5. Average CPU runtime across 30 runs for $n = 50$ datasets.

the number of drone intervals grows, leading to a higher probability of infeasible assignments and a corresponding increase in repair operations.

Overall, the results demonstrate that the primary driver of SE1's slower runtime is the fixed discretisation overhead, not the cost of feasibility enforcement. While repair behaviour varies with fleet size and distribution, repairs are either comparable between solution encodings or lower in SE2 for larger fleets. Consequently, SE2 remains consistently faster even in cases where SE2 performs more repairs, confirming that the elimination of discretisation results in the computational advantage.

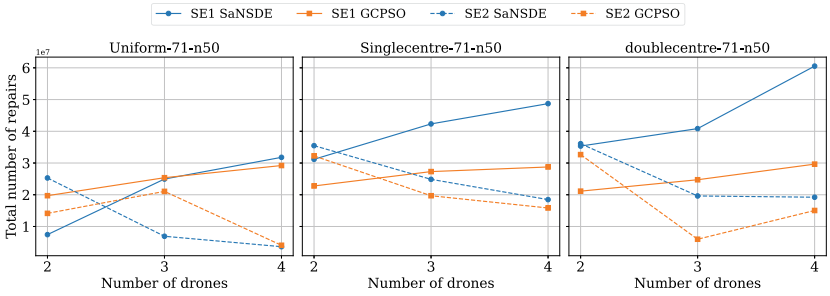

Fig. 6. Total repair mechanisms implemented across 30 runs for $n = 50$ datasets.

6.3 SE2 Scalability Evaluation

The SaNSDE NNHis and the GCPSO NNHis algorithms were applied to SE2 for fleet sizes $d = \{2, 3, 4\}$. The resulting means of the delivery times and standard deviation across the runs are presented in Table 8. An Iman-Davenport corrected Friedman test was conducted to evaluate performance across the six algorithm-fleet configurations. The results indicate a highly significant difference in performance across the six configurations ($F = 198.03, p < 0.001$), justifying the use of a post-hoc analysis.

A Nemenyi post-hoc test was performed to identify specific significant differences between algorithm-fleet pairs. The average ranks and significance groups for algorithm-fleet configurations ($d \in \{2, 3, 4\}$) are presented in Table 9. Configurations sharing a letter in the 'Group' column are not statistically different at $\alpha = 0.05$. While no significant difference was observed between SaNSDE NNHis and GCPSO NNHis at $d = 2$ ($p = 1.00$), both algorithms showed significant improvements as the fleet size increased ($p < 0.001$). Notably, SaNSDE NNHis with $d = 4$ achieved the highest overall ranking, though it was statistically indistinguishable from GCPSO NNHis with $d = 4$ and SaNSDE NNHis with $d = 3$ ($p > 0.05$).

For $d = 2$, the GCPSO NNHis is very competitive, winning in 7 out of 15 instances. For $d = \{3, 4\}$, SaNSDE NNHis has a better objective value for

Table 8. SE2 mean travel time and standard deviation for drone fleet sizes $d \in \{2, 3, 4\}$

Dataset	2 Drones		3 Drones		4 Drones	
	SaNSDE	GCPSO	SaNSDE	GCPSO	SaNSDE	GCPSO
Uniform						
U-51-n10	**22.66 (0.15)**	22.86 (0.25)	**19.36 (0.39)**	19.38 (0.35)	17.09 (0.00)	17.09 (0.00)
U-61-n20	**21.79 (0.27)**	23.91 (1.15)	**20.39 (0.56)**	21.36 (0.66)	**18.32 (0.29)**	18.65 (0.54)
U-71-n50	55.72 (0.60)	**53.18 (1.07)**	**36.29 (0.51)**	45.94 (1.27)	**33.19 (0.51)**	38.17 (0.30)
U-91-n100	75.24 (0.85)	**73.77 (1.24)**	**58.04 (0.94)**	63.94 (0.97)	**52.31 (1.05)**	59.87 (1.56)
U-1-n250	119.45 (0.88)	**117.09 (0.95)**	**103.79 (1.37)**	105.81 (2.09)	**93.36 (0.87)**	97.58 (0.9)
Single-centre						
SC-51-n10	**23.06 (0.10)**	24.48 (0.17)	21.05 (0.02)	21.05 (0.00)	**19.84 (0.02)**	19.87 (0.05)
SC-61-n20	**30.73 (0.80)**	35.47 (1.05)	**26.91 (0.31)**	28.38 (0.85)	**24.44 (0.71)**	25.49 (0.51)
SC-71-n50	52.93 (0.00)	**52.85 (0.46)**	**39.54 (0.73)**	44.35 (0.80)	**36.69 (0.70)**	43.93 (1.94)
SC-91-n100	**109.21 (3.17)**	111.38 (1.86)	**83.95 (2.09)**	95.79 (2.31)	**77.62 (4.16)**	89.45 (3.13)
SC-1-n250	181.28 (2.38)	**178.24 (3.65)**	**153.86 (3.03)**	161.51 (3.68)	**141.66 (1.88)**	153.60 (1.91)
Double-centre						
DC-51-n10	**33.58 (0.00)**	34.67 (0.45)	29.69 (0.00)	29.69 (0.00)	28.02 (0.00)	28.02 (0.00)
DC-61-n20	**58.83 (0.93)**	62.26 (1.59)	**54.60 (1.35)**	57.64 (1.73)	**48.51 (0.47)**	49.84 (1.22)
DC-71-n50	**100.72 (0.00)**	102.49 (4.18)	**84.69 (3.08)**	95.58 (1.84)	**76.18 (1.97)**	83.13 (3.47)
DC-91-n100	140.43 (0.03)	**137.46 (2.69)**	**108.39 (2.42)**	118.65 (3.30)	**95.92 (1.55)**	108.55 (2.97)
DC-1-n250	258.64 (4.70)	**249.97 (4.46)**	**208.58 (3.58)**	218.26 (2.65)	**193.92 (2.77)**	204.04 (6.49)

Table 9. Nemenyi post-hoc test results

Configuration	Average Rank	Group
SaNSDE NNHis_d4	1.33	A
GCPSO NNHis_d4	2.53	A, B
SaNSDE NNHis_d3	3.13	A, B
GCPSO NNHis_d3	4.00	C
SaNSDE NNHis_d2	5.00	D
GCPSO NNHis_d2	5.00	D

almost all datasets. However, the Nemenyi post-hoc test shows that there is no statistically significant difference between the SaNSDE NNHis and GCPSO NNHis algorithm per drone fleet size.

While GCPSO NNHis was initially identified as the best-performing algorithm for the $d = 2$ variant on large datasets in [10], SaNSDE NNHis is equally capable, and often numerically superior, when the drone fleet scales to $d = \{3, 4\}$.

7 Conclusion

This paper investigated the effect of solution encodings on the performance and the computational efficiency of continuous metaheuristic algorithms for the Travelling Salesman Problem with Multiple Drones and Interception. Two alternative

assignment strategies were evaluated within identical algorithmic frameworks, allowing the impact of encoding choice to be isolated from other algorithmic factors.

The experimental results show that SE2 consistently outperforms SE1 in terms of runtime across all tested datasets, drone fleet sizes, and optimisation algorithms, while producing solutions of comparable quality. This performance improvement is not explained by a lower number of repair operations. The dominant factor is the computational overhead introduced by the discretisation process required in SE1 to map continuous decision variables to specific drone indices. By avoiding this step, SE2 reduces per-iteration computational cost, even in cases where the number of repair mechanisms applied is similar or higher.

Increasing the drone fleet size consistently reduced the resulting travel time under SE2, although diminishing marginal returns were observed as the fleet size grew. Both metaheuristic algorithms produced statistically comparable results across the tested fleet sizes; however, SaNSDE NNHis yielded numerically superior solutions relative to GCPSO NNHis as the number of drones increased. Overall, these findings demonstrate that encoding-level design choices can have a decisive impact on algorithmic efficiency in continuous formulations of hybrid truck-and-drone routing problems.

Future research may investigate the incorporation of additional heuristic operators within the proposed algorithms. A road network can be included to investigate a real-world application.

Disclosure of Interests. The authors have no competing interests with the work discussed in this paper.

References

1. Oxford Dictionary of E-commerce. Oxford University Press (2020)
2. Chung, S.-H., Tsai, W.-H.: Drone-assisted last-mile delivery. Robot. Auton. Syst. **130**, 103–115 (2020)
3. Chang, C., Park, D.: Truck-drone collaborative delivery systems. In: Proceedings of the International Conference on Logistics, pp. 45–52 (2018)
4. 7-eleven just made the first commercial delivery by drone. https://www.theverge.com/2016/7/23/12262468/7-11-first-retailer-deliver-food-drone Accessed 11 May 2025
5. Mckinsey and Company. Drones take to the sky, potentially disrupting last-mile delivery. https://www.mckinsey.com/industries/aerospace-and-defense/our-insights/future-air-mobility-blog/drones-take-to-the-sky-potentially-disrupting-last-mile-delivery Accessed 11 May 2025
6. Mckinsey and Company. Consumer views of drone delivery: How soon can you get here? https://www.mckinsey.com/industries/aerospace-and-defense/our-insights/future-air-mobility-blog/consumer-views-of-drone-delivery-how-soon-can-you-get-here Accessed 11 May 2025
7. Moshref-Javadi, M., Winkenbach, M.: Applications and research avenues for drone-based models in logistics: a classification and review. Expert Syst. Appl. **177**, 114854 (2021)

8. Agatz, N., Bouman, P., Schmidt, M.: Optimization approaches for the traveling salesman problem with drone. Transp. Sci. **52**(4), 965–981 (2018)
9. Moremi, T.J., Grobler, J., Kaminsky, P.M.: An ant colony optimisation approach to scheduling truck and drone delivery systems, PhD thesis, Stellenbosch University (2022)
10. Dillon, S., Grobler, J.: Solving routing problems with multiple drones with interceptions. In: Fourth Decision Science Alliance International Summer Conference. Springer (2026)
11. Boccia, M., Mancuso, A., Masone, A., Murino, T., Sterle, C.: New features for customer classification in the flying sidekick traveling salesman problem. Expert Syst. Appl. **247**, 123106 (2024)
12. Chung, S.H., Sah, B., Lee, J.: Optimization for drone and drone-truck combined operations: a review of the state of the art and future directions. Comput. Operat. Res. **123**, 105004 (2020)
13. Liang, Y., Luo, Z.: A survey of truck-drone routing problem: literature review and research prospects. J. Operat. Res. Soc. China **10**(2), 343–377 (2022)
14. Madani, B., Ndiaye, M.: Hybrid truck-drone delivery systems: a systematic literature review. IEEE Access **10**, 92854–92878 (2022)
15. Otto, A., Agatz, N., Campbell, J., Golden, B., Pesch, E.: Optimization approaches for civil applications of unmanned aerial vehicles (uavs) or aerial drones: a survey. Networks **72**(4), 411–458 (2018)
16. Poikonen, S., Campbell, J.F.: Future directions in drone routing research. Networks **77**(1), 116–126 (2021)
17. Zhang, R., Dou, L., Xin, B., Chen, C., Deng, F., Chen, J.: A review on the truck and drone cooperative delivery problem. Unmanned Syst. **12**(05), 823–847 (2024)
18. Boysen, N., Briskorn, D., Fedtke, S., Schwerdfeger, S.: Drone delivery from trucks: drone scheduling for given truck routes. Networks **72**(4), 506–527 (2018)
19. Ha, Q.M., Deville, Y., Pham, Q.D., Hà, M.H.: On the min-cost traveling salesman problem with drone. Trans. Res. Part C: Emerging Technol. **86**, 597–621 (2018)
20. Knoetze, F.E.: Solving a last mile truck and drone delivery schedule to optimality. University of Stellenbosch, Final year project (2021)
21. Ernst, R.: Evolutionary algorithms for routing problems with drones with interceptions, PhD thesis, Stellenbosch University (2024)
22. Betti Sorbelli, F., Corãš, F., Das, L.P., Pinotti, C.M.: Greedy algorithms for scheduling package delivery with multiple drones. In: ACM International Conference Proceeding Series, pp. 31–39 (2022)
23. Sorbelli, F.B., Corãš, F., Das, S.K., Palazzetti, L., Pinotti, C.M.: On the scheduling of conflictual deliveries in a last-mile delivery scenario with truck-carried drones. Pervasive Mob. Comput. **87** (2022)
24. Salama, M.R., Srinivas, S.: Collaborative truck multi-drone routing and scheduling problem: package delivery with flexible launch and recovery sites. Trans. Res. Part E: Logist. Trans. Rev. **164** (2022)
25. Han, B., Chi, T., Sun, Z.: Collaborative delivery problem of trucks and multiple drones: an en-route operation with flexible launch positions. Int. Trans. Oper. Res. (2025)
26. Lin, M., Chen, Y., Han, R., Chen, Y.: Discrete optimization on truck-drone collaborative transportation system for delivering medical resources. Discret. Dyn. Nat. Soc. (2022)
27. Lin, M., Lyu, J.-Y., Gao, J.-J., Li, L.-Y.: Model and hybrid algorithm of collaborative distribution system with multiple drones and a truck. Sci. Program. (2020)

28. Narayan, S., Paulos, J., Chen, S.W., Manjanna, S., Kumar, V.: Combined routing and scheduling of heterogeneous transport and service agents. In: IEEE International Conference on Intelligent Robots and Systems, pp. 1466–1472 (2021)
29. Li, H., Chen, J., Wang, F., Zhao, Y.: Truck and drone routing problem with synchronization on arcs. Nav. Res. Logist. **69**(6), 884–901 (2022)
30. Thomas, T., Srinivas, S., Rajendran, C.: Collaborative truck multi-drone delivery system considering drone scheduling and en route operations. Ann. Oper. Res. **339**(1–2), 693–739 (2024)
31. Price, K.V., Storn, R.M., Lampinen, J.A.: Differential evolution: a practical approach to global optimization. Springer (2005)
32. Yang, Z., Tang, K., Yao, X.: Self-adaptive differential evolution with neighborhood search. In: 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence), pp. 1110–1116. IEEE (2008)
33. Qin, A.K., Suganthan, P.N.: Self-adaptive differential evolution algorithm for numerical optimization. In: 2005 IEEE congress on evolutionary computation, vol. 2, pp. 1785–1791. IEEE (2005)
34. Kennedy, J., Eberhart, R.:Particle swarm optimization. In: Proceedings of ICNN 1995-International Conference on Neural Networks, vol 4, pp. 1942–1948. IEEE (1995)
35. Sahab, M.G., Toropov, V.V., Gandomi. A.H.: A review on traditional and modern structural optimization: problems and techniques. Metaheuristic Appli. Struct. Infrastruct., 25–47 (2013)
36. van den Bergh, F., Engelbrecht, A.P.: A new locally convergent particle swarm optimiser. In: IEEE International Conference on Systems, Man and Cybernetics, vol. 3, p. 6 IEEE (2002)
37. Dillon, S., Ernst, R., Grobler, J.: Investigating alternative clustering algorithms for a cluster-first, route-second truck and drone scheduling algorithm. In: Gervasi, O., et al. ICCSA 2023. LNCS, vol. 13957. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-36808-0_10
38. Grobler, J., Engelbrecht, A.P., Kok, S., Yadavalli, S.: Metaheuristics for the multi-objective fjsp with sequence-dependent set-up times, auxiliary resources and machine down time. Ann. Oper. Res. **180**(1), 165–196 (2010)
39. Bouman, P., Agatz, N., Schmidt, M.: Instances for the tsp with drone (2015). https://zenodo.org/records/22245
40. Akiba, T., Sano, S., Yanase, T., Ohta, T., Koyama, M.: Optuna: A next-generation hyperparameter optimization framework. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2019)