Applying Partial-ACO to Large-scale Vehicle Fleet Optimisation

Darren M. Chitty, Elizabeth Wanner, Rakhi Parmar, Peter R. Lewis
Aston Lab for Intelligent Collectives Engineering (ALICE)
Aston University, Birmingham. B4 7ET UK
{d.chitty,e.wanner,r.parmar8,p.lewis}@aston.ac.uk

ABSTRACT

Optimisation of fleets of commercial vehicles with regards scheduling tasks from various locations to vehicles can result in considerably lower fleet traversal times. This has significant benefits including reduced expenses for the company and more importantly, a reduction in the degree of road use and hence vehicular emissions. Exact optimisation methods fail to scale to real commercial problem instances, thus meta-heuristics are more suitable. Ant Colony Optimisation (ACO) generally provides good solutions on small-to-medium problem sizes. However, commercial fleet optimisation problems are typically large and complex, in which ACO fails to scale well. *Partial-ACO* is a new ACO variant designed to scale to larger problem instances. Therefore this paper investigates the application of *Partial-ACO* on the problem of fleet optimisation, demonstrating the capacity of *Partial-ACO* to successfully scale to larger problems. Indeed, for real-world fleet optimisation problems supplied by a Birmingham based company with up to 298 jobs and 32 vehicles, *Partial-ACO* can improve upon their fleet traversal times by over 44%. Moreover, *Partial-ACO* demonstrates its ability to scale with considerably improved results over standard ACO and competitive results against a Genetic Algorithm.

KEYWORDS

Fleet Optimisation, Multi-Depot Vehicle Routing Problem, Ant Colony Optimisation

1 INTRODUCTION

Fleet optimisation is a common problem faced by many organisations from delivery companies to maintenance firms such as plumbers or builders to the medical profession performing care in the community. The companies in question have many tasks to perform over a geographical area and a set of vehicles with which to carry them out. The problem is two fold, which tasks to assign to each vehicle to carry out and the order by which each vehicle conducts these tasks such that the traversal time of the fleet of vehicles as a whole is minimised. Moreover, there are often challenging constraints such as vehicular capacities, capabilities and time windows within which tasks must be carried out.

The primary benefit from fleet optimisation is reduced costs both in terms of fuel and labour costs with regards driving. Optimisation can also facilitate a greater number jobs to be undertaken by the vehicle fleet. However, an added important benefit from reducing vehicular fleet traversal time is a reduction in the degree of traffic on the road network of a city. This will assist in a reduction in vehicle emission levels experienced within urban conurbations. Indeed, the World Health Organisation (WHO) reports the level of particulates such as nitrogen oxides (NO_x) in the air of major cities are increasing markedly[1]. These can cause breathing problems and have been linked to increased rates of cardiovascular disease [4, 30]. In fact, many cities have limits on permissible pollution, such as London which must maintain a level of particulates below a threshold. Consequently, clean air policies are now being pursued, such as that being considered by Birmingham City Council[2], and fleet optimisation can assist in this goal.

However, optimising fleets of vehicles with constraints is NP-hard in nature and consequently, techniques such as heuristics applied to the problem fail to scale well with problem size. The work presented in this paper will profile a recent advance in the Ant Colony Optimisation (ACO) meta-heuristic technique known as *Partial-ACO* [7] and demonstrate its ability to scale better than ACO when applied to real-world fleet optimisation problems.

The paper is laid out as follows: Section 2 describes the fleet optimisation problem and prior approaches for solving. Section 3 describes in detail the *Partial-ACO* approach. The results from applying *Partial-ACO* to a real-world fleet optimisation problem with steadily increasing complexity will be demonstrated in Section 4 alongside competing meta-heuristic methods. Finally, Section 5 sums up the benefits of *Partial-ACO* for fleet optimisation.

2 PROBLEM DEFINITION AND PRIOR ART

Optimising vehicles through a road network has been well studied. Initial work involved routing a single vehicle visiting a set of differing locations once only with the minimal distance traversed such as would be done by a salesman. Hence this type of task became known as the Travelling Salesman Problem (TSP). An extension of this problem is to consider multiple vehicles being used to visit each location known as the Vehicle Routing Problem (VRP), applicable to many real-world logistic scenarios such as parcel deliveries and the provision of maintenance services. Furthermore, an extension to this problem is to consider a capacity limit to vehicles such as packages the vehicle can hold or the amount of fuel or a working time directive. This is known as the Capacitated Vehicle Routing Problem (CVRP). Now the problem becomes minimising the distance travelled by all vehicles whilst not violating the constraints on capacity. Extending further is the consideration of multiple locations from which the vehicles operate from, essentially depots. Vehicles can return to any of these depots or have to return to the depot where hence they start from. This version of the VRP is known as the Multi Depot Vehicle Routing Problem (MDVRP) and was formulated in 1959 by Dantzig and Ramser [9]. The task is to assign customers to vehicles operating from the available depots such that all customers are serviced whilst the minimum distance is travelled by the fleet of vehicles.

[1] Air pollution levels rising in many of the worlds poorest cities. http://www.who.int/mediacentre/news/releases/2016/air-pollution-rising/en
[2] A Clean Air Zone for Birmingham https://www.birmingham.gov.uk/caz

A final aspect to consider with the VRP is time windows (VRPTW) whereby now the locations to visit must be done so within a given start and end time. For instance a customer to visit may only be available within a given time window. Consequently, the problem now becomes minimising the travel time for a fleet of vehicles whilst not violating the constraints imposed by time windows for delivery or servicing and the capacity of the vehicle.

The MDVRP can be formally defined as a complete graph $G = (V, E)$, whereby V is the vertex set and E is the set of all edges between vertices in V. The vertex set V is further partitioned into two sets , $V_c = V_1, ..., V_n$ representing customers and $V_d = V_{n+1}, ...V_{n+p}$ representing depots whereby n is the number of customers and p is the number of depots. Furthermore, each customer $v_i \in V_c$ has a service time associated with it and each vehicle $v_i \in V_d$ has a fixed capacity associated with it defining the ability to fulfill customer service. Each edge in the set E has an associated cost of traversing it represented by the matrix c_{ij}. The problem is essentially to find the set of vehicle routes such that each customer is serviced once only, each vehicle starts and finishes from the same depot, each vehicle does not exceed its capacity to service customers and the overall cost of the combined routes is minimised.

2.1 Methods Applied to Solving the MDVRP

The MDVRP is recognised as an NP-hard problem and hence difficult to solve using exact mathematical approaches. Indeed, it has been shown that for symmetric cases and a single vehicle, exact methods do not scale beyond 50 customers and only up to a few hundred for asymmetric cases [19]. However, some limited work exists in the literature. Laporte *et al.* formulated a symmetric version as an integer linear program involving three constraints and solved using branch and bound methods [20]. Laporte *et al.* also investigated asymmetric MDVRP problems by translating to constraint assignment problems and again solved using branch and bound [27]. For MDVRP instances with a heterogenous fleet of vehicles available, Dondo *et al.* used mixed-integer programming to solve this problem [10] later extending to multiple pickups and deliveries [11] and again to using time windows [12]. Branch and bound methods have also been recently applied to solving MDVRP problems with some success [2, 3].

However, these exact methods can only solve problems with fewer than 50 customers for symmetric instances. Consequently, heuristic methods are more likely to be applied to larger problems. Heuristic methods are rule of thumb approaches not guaranteed to find optimal solutions to problems but can find solutions close to the optimal in significantly less computational time than exact methods. One the earliest methods uses the Clarke and Wright savings criterion [8] whereby customers are assigned to their nearest deport and routes between depots and customers created which are then gradually merged into larger routes using this savings criterion [40]. Tillman and Hering extended this further to consider looking ahead to consider the effect of assignments on future assignment decisions [41].

An alternative heuristic is a sweep method employed by Wren and Holliday whereby each customer is assigned to their nearest depot and the polar angle to this depot calculated [43]. Customers are sorted in ascending order of angle and iteratively assigned to routes with least additional distance. Gillette and Johnson used a similar approach but instead assign customers to depots to form compact clusters [21]. Each depot takes a turn to be an attracting centre whereby customers are possibly reassigned when lying between two depots. Golden *et al.* superimposed a grid over the problem and then only join vertices in adjoining cells [22]. The authors used a second approach whereby first customers are assigned to depots and then the VRP solved for each individual depot followed by an improvement phase. Tests were performed on problems of up to 256 customers with reasonable results. Raft used an approach whereby the best number of vehicles to use is calculated [28]. Once this has been estimated the customers are clustered into a matching number of groups which are each assigned to a vehicle. 2-opt local search is then used to find the optimal cluster routes. Chao *et al.* assigned customers to their closest depot and then solve the VRPs for each individual depot using a modified savings algorithm [6]. Salhi and Sari presented a "multi-level composite" heuristic which could identify solutions similar to those found in the literature at a fraction of the computational cost [32]. Salhi and Nagy later use an insertion heuristic to minimise the routing cost [31].

2.2 Meta-Heuristic Approaches

An alternative to solving MDVRP problems is to use meta-heuristics, which are essentially a search-based heuristic and are problem independent. The first use of a meta-heuristic approach to the MDVRP was implemented by Gendreau *et al.* involving tabu search solution exploring the search space by potentially moving to a neighbouring solution even with degradation in the quality [18]. Solutions recently examined are declared *tabu* for a number of iterations such that cycling is avoided. Renaud *et al.* also used tabu search with the additional constraint that a vehicle route cannot exceed a certain amount to adhere to working time directives [29].

The most commonly used meta-heuristic approach to solving VRP-type problems is a Genetic Algorithm (GA) [25]. A population based approach which uses the principles of Darwinian evolution such as natural selection, crossover and mutation to successively improve the quality of a population of solutions. The first use of a GA to solve the MDVRP was proposed by Filipec *et al.* for non-fixed destination instances [15]. Salhi et al. also used a GA to solve MDVRP instances [33]. Skok *et al.* used a GA and compared six differing crossover operators [34] later applying the approach to a real-world problem with 248 customers and three depots [35]. An alternative combining clustering with a GA was proposed by Thangiah and Salhi whereby the GA defines clusters of customers and then routes were found by solving the resulting TSP [39].

Ho *et al.* proposed a hybrid approach whereby the initial population of solutions is generated using the Clarke and Wright saving algorithm with the nearest neighbour heuristic proving better than random generation [24]. Surekha and Sumathi used a similar implementation obtaining comparable results to the literature using a set of classical problems [38]. Alba and Dorronsoro used a cellular GA approach with the population organised into a grid and genetic operations only occur within small neighbourhoods successfully improving upon the best known solutions for nine classical instances of the MDVRP [1]. A wide survey of the use of GAs for solving MDVRPs can be found in Karakatič and Podgorelec [26].

The largest competing meta-heuristic approach to a GA applied to the MDVRP is based on the foraging behaviour of ants known

as Ant Colony Optimisation (ACO) [13]. Essentially, ants operate as a collective in finding food by depositing a pheromone as they move. This pheromone builds up as ants successively traverse the same area helping guide ants to promising areas containing food. Yalian implemented a version of ACO which combined a scanning algorithm to initially assign jobs to nearest depots and then combined with GA inspired mutation operators and local search [44]. Results demonstrated an improvement over standard ACO for six problems. Yu *et al.* also achieved better results from a parallel implementation of an improved ACO algorithm whereby a virtual depot is used with mutation operators applied to problems with up to 360 customers and 6 depots [46]. Yao *et al.* also used a single depot approach to solve a real-world MDVRP finding multiple routes from this central depot and then assigning the individual routes to depots [45]. Calvete *et al.* contrasted two ACO approaches, one using a single *super depot* and the other a GA-ACO approach whereby customers are assigned to depots using a GA and then the routing solving by ACO for problems of up to 288 customers and 6 depots [5]. The GA-ACO proved the better algorithm for solving the MDVRP. Stodola uses ACO augmented with an additional route optimisation step to solve a modification of the MDVRP whereby the goal is to minimise the longest route experienced by a vehicle [36].

A final meta-heuristic of note for solving the MDVRP is based on the behaviours of swarms of insects and flocks of birds known as Particle Swarm Optimisation (PSO) [14]. This algorithm searches the fitness landscape using a collection of particles that fly around the landscape with each particle consisting of a position and velocity both described as a set of multi-dimensional coordinates. Each particle's position is updated according to its velocity and then the velocity is updated according to a particles best found solution and the globally best found. Surekha and Sumathi applied PSO to the MDVRP grouping customers to nearest depots and initially routing using the Clark and Wright saving method and these routes were then improved using PSO [38]. Wenjing and Ye achieved better results with improved inertia weights and mutation operators [42]. Geethra *et al.* apply a modified PSO algorithm whereby initial particles are generated using k-means and nearest neighbour heuristics [17] improving further with a nested PSO approach whereby a master swarm assigns customers to groups serviced by a depot and slave swarms then find the optimal route for each group applied to problems with 160 customers and 4 depots [16].

3 THE PARTIAL-ACO APPROACH

Partial-ACO is a derivative of the Ant Colony Optimisation (ACO) technique. ACO will be profiled first followed by a description of the *Partial-ACO* approach.

3.1 Ant Colony Optimisation

The predominate alternative meta-heuristic approach to using a GA to solve fleet optimisation/MDVRP problems is to consider an algorithm based on the study of how ants forage for food known as Ant Colony Optimisation (ACO) [13]. Ants deposit pheromone on the paths they take which enables them to successfully find food and bring it back to their nest. Pheromone levels will build up on edges leading to food enabling further ants from the colony to discover the food source. The ACO algorithm applied to fleet optimisation and MDVRP type problems involves simulated ants moving through a graph G probabilistically visiting every customer once only and depositing pheromone as they move. The level of pheromone an ant deposits on the edges E of graph G is defined by the quality of the solution the given ant has generated. Ants probabilistically decide which location or customer to visit next using this pheromone level on the edges of graph G plus heuristic information based upon the distance between an ant's current position and customers left to visit. An *evaporation* effect is used to prevent pheromone levels building up too much reaching a state of local optima. Therefore, the ACO algorithm consists of two stages, the first *solution construction* and the second stage *pheromone update.*

The solution construction stage involves m ants constructing complete solutions to the fleet optimisation / MDVRP type problems. Ants start from depot of a random vehicle from the fleet and iteratively make probabilistic choices using the *random proportional rule* as to which customer to visit next or to return to the depot whereby the probability of ant k at point i visiting point $j \in N^k$ is defined as:

$$p_{ij}^k = \frac{[\tau_{ij}]^\alpha [\eta_{ij}]^\beta}{\sum_{l \in N^k} [\tau_{il}]^\alpha [\eta_{il}]^\beta} \tag{1}$$

where $[\tau_{il}]$ is the pheromone level deposited on the edge leading from location i to location l; $[\eta_{il}]$ is the heuristic information consisting of the distance between customer or depot i and customer or depot l set at $1/d_{il}$; α and β are tuning parameters controlling the relative influence of the pheromone deposit $[\tau_{il}]$ and the heuristic information $[\eta_{il}]$.

The process is repeated for each vehicle. Once all ants have completed the solution construction stage, pheromone levels on the edges E of graph G are updated. First, evaporation of pheromone levels upon every edge of graph G occurs whereby the level is reduced by a value ρ relative to the pheromone upon that edge:

$$\tau_{ij} \leftarrow (1 - \rho)\tau_{ij} \tag{2}$$

where ρ is the *evaporation rate* typically set between 0 and 1. Once this evaporation is completed each ant k will then deposit pheromone on the edges it has traversed based on the quality of the solution found:

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^k \tag{3}$$

where the pheromone ant k deposits, $\Delta\tau_{ij}^k$ is defined by:

$$\Delta\tau_{ij}^k = \begin{cases} 1/C^k, & \text{if edge } (i,j) \text{ belongs to } T^k \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

where $1/C^k$ is the quality of ant k's solution T^k. This methodology ensures that better solutions found by an ant result in greater levels of pheromone being deposited on those edges.

3.2 Partial-ACO

Although ACO has been a highly successful meta-heuristic approach there are two main issues with the technique. The first is the requirement for a pheromone matrix to be held in memory of size

n^2. Hence, if n becomes extremely large there will not be enough memory in a modern computer to hold this matrix. For example, for a 100,000 customer problem and using a float datatype which is four bytes in memory, approximately 37GB of memory will be required to hold this matrix.

A second scalability factor with ACO is the length of the required solutions as the size of the problem increases. At each step an ant needs to probabilistically decide which unvisited location to visit next by collating the pheromone on all unvisited edges in graph G. The ant then moves to the next location using a random value combined with the pheromone levels. Thus it can be postulated that an ant will probabilistically make a poor decision over which location to visit next at some point when constructing a solution even with high concentrations of pheromone on the correct edge. However, even if the occurrence of a poor decision being made by ant at a decision point is of a low probability, as the problem sizes increases the greater number of decisions an ant needs to make and hence the chance an ant will make a poor choice. But to construct an optimal solution every one of an ant's decisions will need to be the optimal choice. Therefore, it can be hypothesized that ACO by its probabilistic nature will be less likely to find the optimal solution the larger in size the required solution becomes.

Moreover, the computational cost of probabilistically making a decision at each step of constructing a solution increases quadratically. At each step an ant needs to compare the pheromone levels and a random probability for every unvisited edge. Thus for a five customer TSP instance, at the first step four comparisons are needed with the ant selecting the best edge. At the next step three comparisons are required and so forth. In total, nine comparisons are required to construct a complete solution. This is similar to the triangular number sequence described as $(n(n-1)/2)$. Thus, for a 100,000 customer problem, nearly five billion pheromone edge comparisons will be required to construct a complete solution. Indeed, the original author of ACO noted this computational complexity proposing a variant known as Ant Colony System (ACS) [13] whereby the neighbourhood of unvisited cities is restricted. A *candidate list* approach is used whereby at each decision point made by an ant, only the closest cities are considered. If these have already been visited then normal ACO used. This approach significantly reduces the computational complexity but is reliant on accurate heuristic information to define the neighbourhood.

To address these issues a derivative of ACO known as *Partial-ACO* has been proposed which enables ACO to be successfully applied to TSP instances of up to 200,000 cities [7]. *Partial-ACO* operates in a similar manner to ACO but does not use a pheromone matrix addressing the first scalability problem with ACO. Instead, pheromone is calculated from a population of ants and their respective solutions and their associated quality. This is effectively population based ACO (P-ACO) [23] whereby a population of ant solutions is maintained in a First In First Out manner. If a new best solution is found then it is inserted into the population and the oldest one removed. However, *Partial-ACO* maintains the population of ant solutions differently in that each ant has a *local memory* (l_{best}) of the best solution it has personally found during the search process operating essentially as a *steady state* process. This is similar in effect to Particle Swarm Optimisation (PSO) [14] whereby particles use both their local best solution and a global best to update their position.

Pheromone deposit also operates differently to ACO as pheromone cannot *build up* on the edges of graph G. To account for this issue, pheromone deposit of ant k on an edge E of graph G is related to the quality of the solution l^k_{best} related to the best solution found so far, g_{best}. Hence, the amount of pheromone ant k deposits, $\Delta\tau^k_{ij}$ is defined by:

$$\Delta\tau^k_{ij} = \begin{cases} (g_{best}/l^k_{best})^{\alpha}, & \text{if edge } (i,j) \text{ belongs to } T^k \\ 0, & \text{otherwise} \end{cases} \quad (5)$$

where (g_{best}/l^k_{best}) is the quality of ant k's locally best found solution in relation to the globally best found solution and α is a parameter controlling the influence of pheromone. This mechanism ensures that pheromone level deposits do not alter as improved solutions are found and also do not suffer from any scalability issues.

Since there is no pheromone matrix, an ant k when building a solution has to reconstruct the pheromone levels on the edges from its current location to all unvisited locations. This involves iterating through all the l_{best} solutions of the ant population and finding the edges taken from the current location and depositing the pheromone if that location has not yet been visited. Moreover, for each location in an ant's solution there is an edge taken from that given location but there is also an edge taken to arrive at the given location. Consequently, these edges are also taken into consideration when constructing the pheromone levels on edges.

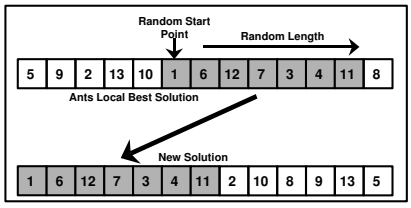

Figure 1: An illustration of the *Partial-ACO* methodology.

The second component of *Partial-ACO* addresses the error probability during solution construction and the computational complexity of ACO. Recall that for a hundred thousand job problem, the number of decisions that need to be made are of an equal number whereby there is a probability of making a poor decision. Secondly, the number of pheromone comparisons required to build a complete solution for a problem consisting of a hundred thousand jobs requires nearly five billion comparisons. Consequently, *Partial-ACO* proposes that an ant when constructing a new solution takes its current l_{best} tour and retains part of this tour and then completes the rest in the same probabilistic manner as described earlier. This is to some degree similar to crossover in a GA whereby a complete section of a parent solution is copied into the child solution. To achieve this *partial* reconstruction of a solution firstly a random point is selected in the l_{best} tour. Secondly, a random length of the tour to be retained is decided and this section is copied into the new solution. The remaining part of the solution is then constructed as normal. This process is illustrated in Figure 1.

If this partially reconstructed solution has greater quality then the current l_{best} for the given ant then this new solution replaces

the ant's l_{best} solution. To highlight the computational advantage of this technique, consider retaining 50% of solutions for a 100,000 job problem. In this instance only 50,000 probabilistic decisions now need to be made and only 1.25 billion pheromone comparisons would be required, a reduction of 75%. In fact, the potential computational cost savings are quadratic in nature. An overview of the *Partial-ACO* technique is described in Algorithm 1.

Algorithm 1 *Partial-ACO*

```
for each ant do
    Generate an initial solution probabilistically
end for
for number of iterations do
    for each ant k do
        Select uniform random start point from l^k_best solution
        Select uniform random length of l^k_best to preserve
        Copy l^k_best points from start for specified length
        Complete remaining aspect probabilistically
        If new solution better than l^k_best then update l^k_best
    end for
end for
Output best l_best solution (the g_best solution)
```

3.3 Partial-ACO Applied to Fleet Optimisation

Fleet optimisation is in effect the MDVRPTW problem whereby there are a range of vehicles operating from a number of depots that need to be assigned jobs to complete such that all jobs are completed within their time windows and the total traversal time of the fleet of vehicles is minimised. To represent this problem a solution will consist of a set of vehicles and the list of jobs to be completed in the order they need to be completed. This representation is depicted in Figure 2 whereby V relates to a vehicle and J relates to a job. It can be observed that the first vehicle in the fleet will undertake jobs 6, 5 and 9, the second vehicle jobs 3, 7, and 2 and so forth. Note that vehicle 3 is not assigned any jobs.

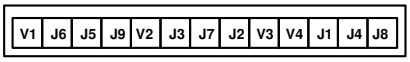

Figure 2: Example solution representation.

An ant will build a solution in a similar manner to a TSP instance in that the number of locations to visit equates to the number of jobs and the number of vehicles. Each of these locations must be visited once only. To start a solution a vehicle must be selected first. From this point ants probabilistically move to unfulfilled jobs or another vehicle at which point the current vehicle returns to its depot and a new sequence of jobs to be assigned to this new vehicle commences.

Once a new solution has been generated by *Partial-ACO* its quality needs to be assessed. The quality is measured using two objectives, the first of which is to maximise the number of jobs correctly performed within their given time window. Any missed jobs are assigned a penalty. The second objective is the total traversal time of the fleet of vehicles. The goal is to minimise both the number of missed jobs and the traversal time of the vehicular fleet. Reducing the number of missed jobs is considered the primary objective.

Pheromone deposit is calculated using the two objectives to be optimised, the number of customers serviced and the length of the traversal time between customers. A penalty based function will be utilised for the first objective whereby any customers that have not been serviced due to capacity limitations or missing the time window will be penalised by the predicted job time. The secondary objective objective is to minimise the time the fleet of vehicles spend traversing the road network between jobs. Therefore, the quality of a tour can be described as:

$$C^k = (S - s^k + 1) * L^k \tag{6}$$

where S is the total amount of time of jobs to be serviced, s^k is the amount of job service time achieved by ant k's solution and L^k is the total traversal time of the fleet of vehicles of ant k's solution. Clearly, if ant k has achieved the primary objective of fulfilling all customer demand S then C^k becomes merely the total traversal time of the fleet of vehicles which needs to be minimised.

Table 1: Real-world problem scenarios as supplied by a Birmingham maintenance company. Scenarios described in terms of the vehicles available, the number of customers to service, the total predicted service time required and the total travel time using the company's current scheduling.

Problem	Number of Vehicles	Number of Jobs	Total Job Servicing (hh:mm)	Total Fleet Traversal Time (hh:mm)
Week_1	8	77	47:09	31:12
Week_2	8	79	48:24	22:49
Week_3	8	81	48:33	19:54
Week_4	8	61	54:52	25:55
Fortnight_1	16	156	95:33	54:01
Fortnight_2	16	138	102:01	57:07
Fortnight_3	16	160	96:57	42:43
Fortnight_4	16	142	103:25	45:49
ThreeWeek_1	24	237	144:06	73:55
ThreeWeek_2	24	217	150:25	79:56
ThreeWeek_3	24	219	150:34	77:01
ThreeWeek_4	24	221	151:49	68:38
Month_1	32	298	198:58	99:50

4 RESULTS

4.1 Problem Description and Setup

To evaluate the application of *Partial-ACO* to fleet optimisation a real-world problem will be considered taking data from a maintenance company based in the Birmingham area which has multiple vehicles and multiple customers requiring periodic external maintenance of properties. Each vehicle starts from a given latitude and longitude, essentially representing its depot, and must return to this location when it has finished servicing customers. Each customer is also defined by a location consisting of a latitude and longitude coordinate and a job duration predicting the length of time the job will take take and in some cases, a time window for when the jobs must be completed. The speed of travel of a van

Table 2: Parameters used throughout for the GA, ACO, *Partial-ACO* and *Partial-ACO*^{PH} approaches unless otherwise stated.

Genetic Algorithm		Max Min Ant System		Partial-ACO		Partial-ACO^{PH}	
Population Size	192	Number of Ants	192	Number of Ants	32	Number of Ants	192
Max Generations	1,000,000	Max Iterations	1,000,000	Max Iterations	6,000,000	Max Iterations	1,000,000
Tournament Size	5	α	1.0	α	3.0	α	1.0
Crossover Probability	0.5	β	1.0	β	1.0	β	1.0
Mutation Probability	0.5	ρ	0.02			ρ	0.5

Table 3: The results from each meta-heuristic approach regards optimising the schedules of the real-world scenarios in terms of the service time demand achieved and the reduction in traversal time of the vehicle fleet over the original scheduling.

Problem	Genetic Algorithm		Max Min Ant System		Partial-ACO		Partial-ACO^{PH}	
	Job Time Serviced (%)	Traversal Time Reduction (%)	Job Time Serviced (%)	Traversal Time Reduction (%)	Job Time Serviced (%)	Traversal Time Reduction (%)	Job Time Serviced (%)	Traversal Time Reduction (%)
Week_1	100.00 ± 0.00	28.10 ± 7.15	100.00 ± 0.00	33.62 ± 3.39	100.00 ± 0.00	32.29 ± 3.77	100.00 ± 0.00	37.14 ± 2.15
Week_2	100.00 ± 0.00	28.06 ± 10.60	100.00 ± 0.00	30.70 ± 4.85	100.00 ± 0.00	22.39 ± 7.84	100.00 ± 0.00	38.20 ± 2.88
Week_3	100.00 ± 0.00	27.47 ± 5.17	100.00 ± 0.00	31.48 ± 4.68	100.00 ± 0.00	28.75 ± 5.55	100.00 ± 0.00	40.43 ± 3.36
Week_4	100.00 ± 0.00	19.19 ± 6.67	100.00 ± 0.00	26.27 ± 4.56	100.00 ± 0.00	24.28 ± 6.84	100.00 ± 0.00	12.11 ± 10.26
Fortnight_1	100.00 ± 0.00	25.38 ± 5.93	100.00 ± 0.00	23.84 ± 7.46	99.98 ± 0.10	23.12 ± 6.30	100.00 ± 0.00	36.43 ± 1.37
Fortnight_2	100.00 ± 0.00	27.33 ± 4.32	100.00 ± 0.00	22.55 ± 5.01	99.98 ± 0.09	25.81 ± 4.72	100.00 ± 0.00	30.56 ± 1.91
Fortnight_3	100.00 ± 0.00	26.58 ± 5.50	100.00 ± 0.00	28.64 ± 4.99	100.00 ± 0.00	27.27 ± 5.49	100.00 ± 0.00	36.99 ± 2.16
Fortnight_4	100.00 ± 0.00	24.83 ± 4.53	100.00 ± 0.00	25.02 ± 4.49	100.00 ± 0.00	29.70 ± 6.64	100.00 ± 0.00	36.14 ± 1.99
ThreeWeek_1	100.00 ± 0.00	28.94 ± 4.01	99.81 ± 0.18	−11.43 ± 7.62	100.00 ± 0.00	17.13 ± 4.47	99.98 ± 0.07	7.37 ± 4.94
ThreeWeek_2	100.00 ± 0.00	26.52 ± 4.35	99.89 ± 0.20	−4.04 ± 3.89	99.80 ± 0.20	16.48 ± 5.18	99.95 ± 0.10	8.44 ± 8.38
ThreeWeek_3	100.00 ± 0.00	29.62 ± 4.21	99.95 ± 0.11	7.33 ± 6.56	99.91 ± 0.21	20.64 ± 2.36	100.00 ± 0.00	20.01 ± 2.71
ThreeWeek_4	100.00 ± 0.00	26.26 ± 4.13	99.86 ± 0.18	−2.36 ± 5.92	99.84 ± 0.27	18.00 ± 7.84	100.00 ± 0.00	13.74 ± 3.84
Month_1	100.00 ± 0.00	29.23 ± 4.18	99.76 ± 0.18	−17.85 ± 3.75	99.82 ± 0.18	19.60 ± 4.09	98.04 ± 0.49	−26.59 ± 8.48

between maintenance jobs is defined as an average of 13kph to account for city traffic etc. There is a also a hard start time and end time to a given working day defined as 08:00 and 19:00 hours. Customer jobs cannot be started or finished before or after these start and end times. However, a vehicle can depart its depot prior to the start time in order to be able to commence a scheduled task at 08:00 hours. Equally a vehicle can finish a job at 19:00 and return to its depot after this time irrespective of how long it takes to get back.

The data supplied by the company is split into a number time periods of jobs and vehicle availability. For each problem, the company has supplied the actual division of jobs between vehicles used and the order that the jobs were conducted. This facilitates a *ground truth* to be established of vehicle usage for the company and consequently real-world reductions to be ascertained from using a meta-heuristical optimisation approach. In this instance, the company divided jobs between vehicles in a geographical manner which they considered correct and then derived the route for each vehicle to complete its assigned tasks by directing it to do the job furthest from its depot and then work its way back only deviating from this if there are jobs with time windows. The details of each of the individual problems supplied by the company are shown in Table 1. Approximately one in ten of the jobs has a defined time window whereby the job must be completed.

To evaluate the effectiveness of *Partial-ACO* it will be compared to both ACO and GA meta-heuristic approaches. In this instance the ACO approach will be based upon the max min ant system (MMAS) [37] which only updates the pheromone deposits for the best found so far solution at each iteration rather than using the whole population. Moreover, the pheromone is restricted to minimum and maximum levels. The GA approach will use the crossover operators cyclic crossover (CX), order crossover (OX) and partially mapped crossover (PMX). Mutation will consist of several operators, swapping two solution points, reversing the order between two points and inserting a solution point elsewhere into the solution. The GA will operate in a *steady state* manner whereby child solutions only replace parents if their quality is better.

Additionally, a second implementation of *Partial-ACO* will be considered which does use a pheromone matrix to be used by ants to probabilistically make decisions. In all other respects the approach operates in the same manner as standard *Partial-ACO* with each ant maintaining a memory of its locally best found solution l_{best} and at every iteration these locally best solutions update the pheromone according to the quality of their individual solutions. This approach will be termed *Partial-ACO*^{PH}.

The parameters governing the operation of each of the four meta-heuristic approaches are described in Table 2. A population size or number of ants of 192 is used as each approach is parallelised and is executed on a processor with 16 threads so the size needs to be a multiple of 16. Furthermore, *Partial-ACO* uses only 32 ants as this is similar to the number used in the original work [7] and was found to be highly effective. To ensure the same number of solutions are evaluated, *Partial-ACO* is executed for six times the number of iterations as the other meta-heuristics. Experiments throughout were conducted using an AMD Ryzen 2700 processor with each approach using 16 parallel threads of execution in order to utilise all the available processor cores of the processor. The algorithms were compiled using Microsoft C++. Experiments are averaged over 25 individual execution runs for each problem with a differing random seed used in each instance.

4.2 Initial Results

To begin with, both approaches will be tested on each of the single day problems to test their ability to find solutions better than those currently used by the Birmingham company that supplied the data. These are described in terms of percentage of total customer service demand achieved, the percentage reduction in time spent traversing the road network for the fleet of vehicles when compared the company's original solution. These results are shown in Table 3. From these results it can be firstly observed that both the GA and ACO approaches have improved upon the approach utilised by the given company to schedule jobs to its vehicles for the smaller problem instances. However, ACO was unable to scale to the larger problem instances as postulated being unable to consistently find solutions which service all the customer tasks within their time windows and additionally find solutions with larger traversal times than the company's original scheduling. The GA proved better at scaling by achieving better results than ACO for the larger problem instances and being able to service all customer tasks for all the problem instances.

With regards the *Partial-ACO* results, both variants were able to improve upon standard ACO demonstrating their ability to scale better. However, both techniques were unable to consistently service all customers within their time windows. *Partial-ACO*^{PH} proved better than the non-pheromone matrix implementation for the smaller problem instances but proved less capable of scaling up to the larger problem instances. The best reduction in total fleet traversal time over the original company scheduling is achieved by *Partial-ACO*^{PH} for the Week_3 problem instance with a 40% reduction.

4.3 Reducing the Degree of Modification

With regards the prior results, only the GA approach was able to find solutions that always serviced all the scheduled tasks. The proposed *Partial-ACO* approaches both failed to scale up to the larger problem instances. However, it was observed in the experimental runs that diversity amongst the population became an issue with ants traversing the same edges. Consequently, solutions converged prematurely. A method that could be used to counter this effect is to ensure that the degree to which an ant can modify its l_{best} solution is limited to a maximum percentage of the solution. This should have the effect of preventing an ant from copying the globally best tour. In fact, in the original *Partial-ACO* work, reducing the degree of modification of solutions improved results and secondly, increased the speed of the approach by reducing the probabilistic decision making of the ants [7].

Table 4 demonstrates the results from both the *Partial-ACO* approaches when the degree of modification of an ant's l_{best} solution is restricted to a maximum of 50%. However, to ensure there is scope to escape local optima this restriction is removed with a probability of 0.001 during the search process. From these results it can be observed that there is a considerable improvement in the quality of the solutions achieved by *Partial-ACO* and it now scales to the larger problem instances. Indeed, the technique now achieves results considerably better than the GA approach for the week long and two week long scenarios. However, for the larger problems the GA meta-heuristic approach still achieves slightly better reductions in fleet traversal times. With regards the pheromone matrix based implementation, *Partial-ACO*^{PH}, slightly better solutions are achieved for the smaller problem instances compared to the no pheromone matrix version. Reductions of up to 44% are achieved regards the fleet traversal times of the original scheduling of the company that supplied the problem data.

Table 4: The results from both *Partial-ACO* and *Partial-ACO*^{PH} when applying a maximum of 50% modification of ant's locally best solution. Results are shown in terms of the percentage of service time demand achieved and the percentage reduction in traversal time of the vehicle fleet over the original company scheduling.

Problem	Partial-ACO		Partial-ACO^{PH}	
	Job Time Serviced (%)	Traversal Reduction (%)	Job Time Serviced (%)	Traversal Reduction (%)
Week_1	100.00 ± 0.00	41.17 ± 0.49	100.00 ± 0.00	41.94 ± 1.37
Week_2	100.00 ± 0.00	42.39 ± 1.11	100.00 ± 0.00	43.48 ± 0.44
Week_3	100.00 ± 0.00	44.43 ± 0.77	100.00 ± 0.00	44.84 ± 0.43
Week_4	100.00 ± 0.00	38.55 ± 1.58	100.00 ± 0.00	40.78 ± 0.41
Fortnight_1	100.00 ± 0.00	23.89 ± 6.54	100.00 ± 0.00	34.30 ± 2.39
Fortnight_2	100.00 ± 0.00	24.41 ± 5.13	100.00 ± 0.00	34.08 ± 1.12
Fortnight_3	100.00 ± 0.00	35.55 ± 9.35	100.00 ± 0.00	41.32 ± 1.89
Fortnight_4	100.00 ± 0.00	31.07 ± 11.24	100.00 ± 0.00	37.51 ± 1.57
ThreeWeek_1	100.00 ± 0.00	28.16 ± 5.05	100.00 ± 0.00	21.78 ± 2.46
ThreeWeek_2	100.00 ± 0.00	23.54 ± 4.48	100.00 ± 0.00	21.00 ± 2.40
ThreeWeek_3	100.00 ± 0.00	27.12 ± 4.17	100.00 ± 0.00	25.94 ± 1.87
ThreeWeek_4	100.00 ± 0.00	22.52 ± 4.05	100.00 ± 0.00	22.29 ± 1.53
Month_1	100.00 ± 0.00	28.19 ± 5.90	98.67 ± 0.28	−5.98 ± 3.54

Table 5: The results from both *Partial-ACO* and *Partial-ACO*^{PH} when applying a maximum of 25% modification of ant's locally best solution. Results are shown in terms of the percentage of service time demand achieved and the percentage reduction in traversal time of the vehicle fleet over the original company scheduling.

Problem	Partial-ACO		Partial-ACO^{PH}	
	Job Time Serviced (%)	Traversal Reduction (%)	Job Time Serviced (%)	Traversal Reduction (%)
Week_1	100.00 ± 0.00	32.93 ± 13.07	100.00 ± 0.00	30.12 ± 2.39
Week_2	100.00 ± 0.00	33.91 ± 5.24	100.00 ± 0.00	30.84 ± 4.91
Week_3	100.00 ± 0.00	33.95 ± 3.10	100.00 ± 0.00	33.20 ± 1.64
Week_4	100.00 ± 0.00	35.06 ± 3.28	100.00 ± 0.00	30.16 ± 3.49
Fortnight_1	100.00 ± 0.00	28.08 ± 7.53	100.00 ± 0.00	31.38 ± 2.37
Fortnight_2	100.00 ± 0.00	30.66 ± 6.14	100.00 ± 0.00	32.15 ± 2.67
Fortnight_3	100.00 ± 0.00	27.27 ± 9.15	100.00 ± 0.00	35.77 ± 2.33
Fortnight_4	100.00 ± 0.00	33.50 ± 7.27	100.00 ± 0.00	30.24 ± 2.42
ThreeWeek_1	100.00 ± 0.00	31.32 ± 6.41	100.00 ± 0.00	23.45 ± 3.21
ThreeWeek_2	100.00 ± 0.00	30.23 ± 5.22	100.00 ± 0.00	19.84 ± 2.03
ThreeWeek_3	100.00 ± 0.00	33.27 ± 6.85	100.00 ± 0.00	26.18 ± 2.89
ThreeWeek_4	100.00 ± 0.00	31.66 ± 6.74	100.00 ± 0.00	22.78 ± 2.42
Month_1	100.00 ± 0.00	32.91 ± 3.73	99.41 ± 0.32	5.35 ± 5.65

This concept can be extended by further reducing the maximum permissible degree of modification of an ant's l_{best} solution to just 25%. Again though, to ensure there is scope to escape local optima this restriction is removed with a probability of 0.001 during the search process. These results are shown in Table 5 whereby from the results regards *Partial-ACO* two effects can be observed. For the larger problem instances the such as Month_1 improved reductions in traversal times are achieved of 30-33% which when now compared the to the GA approach are several percent better. However, with the smaller one week problems the results are

Table 6: The execution timings in minutes across all of the meta-heuristics tested.

Problem	GA	MMAS	Partial-ACO	Partial-ACO^{PH}
Week_1	1.32 ± 0.04	2.23 ± 0.10	4.69 ± 0.49	3.61 ± 0.02
Week_2	1.35 ± 0.04	2.33 ± 0.10	4.76 ± 0.45	3.69 ± 0.03
Week_3	1.35 ± 0.04	2.49 ± 0.10	4.90 ± 0.50	3.82 ± 0.03
Week_4	1.09 ± 0.03	1.64 ± 0.05	3.86 ± 0.51	2.85 ± 0.02
Fortnight_1	2.52 ± 0.03	6.56 ± 0.13	9.87 ± 0.43	8.92 ± 0.04
Fortnight_2	2.20 ± 0.02	5.52 ± 0.07	8.81 ± 0.50	7.72 ± 0.06
Fortnight_3	2.59 ± 0.02	6.84 ± 0.10	10.18 ± 0.38	9.26 ± 0.07
Fortnight_4	2.26 ± 0.02	5.76 ± 0.11	9.09 ± 0.48	7.96 ± 0.05
ThreeWeek_1	4.30 ± 0.03	13.09 ± 0.11	16.42 ± 0.71	16.34 ± 0.03
ThreeWeek_2	3.95 ± 0.02	11.79 ± 0.08	14.81 ± 0.68	14.51 ± 0.06
ThreeWeek_3	3.97 ± 0.02	11.57 ± 0.15	14.54 ± 0.78	14.65 ± 0.05
ThreeWeek_4	4.06 ± 0.03	12.09 ± 0.15	14.95 ± 0.49	14.85 ± 0.06
Month_1	5.96 ± 0.04	19.75 ± 0.13	23.59 ± 0.83	23.18 ± 0.05

poorer. This is because with the larger problems there is still quite a few vehicles and jobs that can be reorganised even with only a 25% maximum modification restriction. But for the smaller problems there is not much capacity for change hence the worsening results. Therefore, it can be postulated that restricting the maximum modification is beneficial to the performance of *Partial-ACO* the size of the problem needs to be taken into account such there is sufficient scope for sufficient solution modification. With regards *Partial-ACO*^{PH} only minor improvements are noted for the larger problems and a similar degradation in performance for the smaller problems. It should be noted that all meta-heuristics did not consistently find an optimal solution. However, all results were generated without any local search operator such as 2-opt which would likely significantly improve the results.

4.4 Execution Timings

One final aspect of analysis left to consider is the execution timings of the various approaches evaluated and these are shown in Table 6. The GA approach is clearly the fastest approach over three times faster than the ACO MMAS approach and nearly four times faster than the *Partial-ACO* approaches. This is because the GA approach does not need to construct solutions by probabilistically investigating each every option at each step. Furthermore, *Partial-ACO* is required to reconstruct pheromone on the unvisited edges from the population. *Partial-ACO*^{PH} is equally slow as this approach needs to update the pheromone matrix each time all ants have constructed a new solution. However, Table 7 shows the execution timings for the *Partial-ACO* approaches when restricting the degree of modification. For *Partial-ACO* execution timings have reduced considerably, although less than expected, through being able to make fewer pheromone reconstructions as less decisions are required to construct a solution. Execution timings are now on a par with a GA approach. With regards *Partial-ACO*^{PH} reductions in execution times are also observed but to a lower degree since the approach still has to update the pheromone matrix.

5 CONCLUSIONS

The work presented in this paper has investigated the use of meta-heuristic approaches for tackling larger scale fleet optimisation problems. Specifically, the use of the novel *Partial-ACO* approach has been compared to the main competing meta-heuristic approaches, GA and ACO. *Partial-ACO* is similar to ACO but involves ants

Table 7: The execution timings in minutes for the *Partial-ACO* approaches and restrictions on solution modification.

Problem	Max. Modification 50%		Max. Modification 25%	
	Partial-ACO	Partial-ACO^{PH}	Partial-ACO	Partial-ACO^{PH}
Week_1	2.96 ± 0.64	3.33 ± 0.02	2.03 ± 0.32	3.25 ± 0.05
Week_2	2.96 ± 0.73	3.38 ± 0.02	2.02 ± 0.17	3.25 ± 0.05
Week_3	3.07 ± 0.59	3.52 ± 0.03	2.05 ± 0.05	3.38 ± 0.05
Week_4	2.91 ± 0.84	2.66 ± 0.05	1.67 ± 0.04	2.53 ± 0.05
Fortnight_1	5.56 ± 0.36	7.81 ± 0.04	3.36 ± 0.40	7.21 ± 0.05
Fortnight_2	4.88 ± 0.42	6.76 ± 0.03	3.08 ± 0.12	6.31 ± 0.06
Fortnight_3	6.03 ± 0.29	8.04 ± 0.05	3.50 ± 0.08	7.45 ± 0.07
Fortnight_4	5.15 ± 0.38	6.97 ± 0.03	3.20 ± 0.06	6.50 ± 0.06
ThreeWeek_1	8.20 ± 0.49	13.80 ± 0.06	4.93 ± 0.35	12.60 ± 0.07
ThreeWeek_2	7.38 ± 0.48	12.41 ± 0.06	4.62 ± 0.38	11.35 ± 0.07
ThreeWeek_3	7.45 ± 0.50	12.79 ± 0.09	4.58 ± 0.54	11.67 ± 0.05
ThreeWeek_4	7.69 ± 0.44	12.75 ± 0.11	4.60 ± 0.34	11.59 ± 0.06
Month_1	10.64 ± 0.52	19.40 ± 0.08	6.18 ± 0.24	17.64 ± 0.12

maintaining a memory of their best found solution and only partially modifying this solution at each iteration. The premise behind this technique was that ACO will struggle to scale well to larger problems because as the size of the problem increases the potential for an ant to probabilistically make an error also increases. By reducing the number of decisions an ant makes when creating a new solution, the probability of an ant making a poor choice is similarly reduced enabling ACO to scale to larger problems better.

The *Partial-ACO* approach with and without a pheromone matrix was tested on a number of real-world problems of increasing complexity and size and compared with the more familiar GA and ACO meta-heuristics. Overall, *Partial-ACO* proved competitive with a GA approach and significantly better than the ACO approach both with and without a pheromone matrix. Indeed, these results reinforced the hypothesis that the more decisions an ant has to make when constructing a solution the greater the probability of making an error occurs reducing the capability for ACO to scale to larger problems. However, it was evidenced that *Partial-ACO* can bypass this potential problem. By ants being able to make fewer decisions as a result of only partially modifying an existing good solution, the probability of making a poor decision is reduced leading to higher quality solutions. Moreover, through a lower degree of decision making the computational speed of *Partial-ACO* is much improved. Overall, *Partial-ACO* was able to improve upon the schedules provided by the Birmingham based company which provided the dataset by up to 44% for smaller problems and 28% for the largest problem containing 32 vehicles and 298 jobs.

It can be stated that fleet optimisation using *Partial-ACO* is capable of providing significant cost savings to commercial companies. Moreover, reducing vehicle traversal to this degree provides significant assistance to reducing vehicular pollution in cities at little to no cost. Future work will consist of developing better strategies with regards the solution preservation aspect of *Partial-ACO* and a complete investigation of parameter settings with a view to scaling *Partial-ACO* to even larger fleet optimisation problems.

6 ACKNOWLEDGEMENT

This work was carried out under the System Analytics for Innovation project, which is part-funded by the European Regional Development Fund (ERDF).

REFERENCES

[1] Enrique Alba and Bernabé Dorronsoro. 2006. Computing nine new best-so-far solutions for capacitated VRP with a cellular genetic algorithm. *Inform. Process. Lett.* 98, 6 (2006), 225–230.

[2] Enrique Benavent and Antonio Martínez. 2013. Multi-depot multiple TSP: a polyhedral study and computational results. *Annals of Operations Research* 207, 1 (2013), 7–25.

[3] Kris Braekers, An Caris, and Gerrit K Janssens. 2014. Exact and meta-heuristic approach for a general heterogeneous dial-a-ride problem with multiple depots. *Transportation Research Part B: Methodological* 67 (2014), 166–186.

[4] L Calderón-Garcidueñas, E Leray, P Heydarpour, R Torres-Jardón, and J Reis. 2016. Air pollution, a rising environmental risk factor for cognition, neuroinflammation and neurodegeneration: the clinical impact on children and beyond. *Revue neurologique* 172, 1 (2016), 69–80.

[5] Herminia I Calvete, Carmen Galé, and María-José Oliveros. 2011. Evolutive and ACO Strategies for Solving the Multi-depot Vehicle Routing Problem.. In *IJCCI (ECTA-FCTA)*. 73–79.

[6] I-Ming Chao, Bruce L Golden, and Edward Wasil. 1993. A new heuristic for the multi-depot vehicle routing problem that improves upon best-known solutions. *American Journal of Mathematical and Management Sciences* 13, 3-4 (1993), 371–406.

[7] Darren M Chitty. 2017. Applying ACO to Large Scale TSP Instances. In *UK Workshop on Computational Intelligence*. Springer, 104–118.

[8] Geoff Clarke and John W Wright. 1964. Scheduling of vehicles from a central depot to a number of delivery points. *Operations research* 12, 4 (1964), 568–581.

[9] George B Dantzig and John H Ramser. 1959. The truck dispatching problem. *Management science* 6, 1 (1959), 80–91.

[10] Rodolfo Dondo, Carlos Alberto Mendez, and Jaime Cerdá. 2003. An optimal approach to the multiple-depot heterogeneous vehicle routing problem with time window and capacity constraints. *Latin American applied research* 33, 2 (2003), 129–134.

[11] Rodolfo Dondo, Carlos A Méndez, and Jaime Cerdá. 2008. Optimal management of logistic activities in multi-site environments. *Computers & Chemical Engineering* 32, 11 (2008), 2547–2569.

[12] Rodolfo G Dondo and Jaime Cerdá. 2009. A hybrid local improvement algorithm for large-scale multi-depot vehicle routing problems with time windows. *Computers & Chemical Engineering* 33, 2 (2009), 513–530.

[13] Marco Dorigo and Luca Maria Gambardella. 1997. Ant colony system: a cooperative learning approach to the traveling salesman problem. *IEEE Transactions on evolutionary computation* 1, 1 (1997), 53–66.

[14] Russell Eberhart and James Kennedy. 1995. A new optimizer using particle swarm theory. In *Micro Machine and Human Science, 1995. MHS'95., Proceedings of the Sixth International Symposium on*. IEEE, 39–43.

[15] Minea Filipec, Davor Skrlec, and Slavko Krajcar. 1997. Darwin meets computers: New approach to multiple depot capacitated vehicle routing problem. In *Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., 1997 IEEE International Conference on*, Vol. 1. IEEE, 421–426.

[16] S Geetha, G Poonthalir, and PT Vanathi. 2013. Nested particle swarm optimisation for multi-depot vehicle routing problem. *International Journal of Operational Research* 16, 3 (2013), 329–348.

[17] Shanmugam Geetha, PT Vanathi, and Ganesan Poonthalir. 2012. Metaheuristic approach for the multi-depot vehicle routing problem. *Applied Artificial Intelligence* 26, 9 (2012), 878–901.

[18] Michel Gendreau, Alain Hertz, and Gilbert Laporte. 1994. A tabu search heuristic for the vehicle routing problem. *Management science* 40, 10 (1994), 1276–1290.

[19] Laporte Gilbert. 1992. The vehicle routing problem: An overview of exact and approximate algorithms. *European journal of operational research* 59, 3 (1992), 345–358.

[20] Laporte Gilbert, Martin Desrochers, and Yves Norbert. 1984. Two exact algorithms for the distance-constrained vehicle routing problem. *Networks* 14, 1 (1984), 161–172.

[21] Billy E Gillett and Jerry G Johnson. 1976. Multi-terminal vehicle-dispatch algorithm. *Omega* 4, 6 (1976), 711–718.

[22] Bruce L Golden, Thomas L Magnanti, and Hien Q Nguyen. 1977. Implementing vehicle routing algorithms. *Networks* 7, 2 (1977), 113–148.

[23] Michael Guntsch and Martin Middendorf. 2002. A population based approach for ACO. In *Workshops on Applications of Evolutionary Computation*. Springer, 72–81.

[24] William Ho, George TS Ho, Ping Ji, and Henry CW Lau. 2008. A hybrid genetic algorithm for the multi-depot vehicle routing problem. *Engineering Applications of Artificial Intelligence* 21, 4 (2008), 548–557.

[25] John H Holland. 1975. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. U Michigan Press.

[26] Sašo Karakatič and Vili Podgorelec. 2015. A survey of genetic algorithms for solving multi depot vehicle routing problem. *Applied Soft Computing* 27 (2015), 519–532.

[27] Gilbert Laporte, Yves Nobert, and Serge Taillefer. 1988. Solving a family of multi-depot vehicle routing and location-routing problems. *Transportation science* 22, 3 (1988), 161–172.

[28] Ole M Raft. 1982. A modular algorithm for an extended vehicle scheduling problem. *European Journal of Operational Research* 11, 1 (1982), 67–76.

[29] Jacques Renaud, Gilbert Laporte, and Fayez F Boctor. 1996. A tabu search heuristic for the multi-depot vehicle routing problem. *Computers & Operations Research* 23, 3 (1996), 229–235.

[30] Weeberb J Requia, Matthew D Adams, Altaf Arain, Stefania Papatheodorou, Petros Koutrakis, and Moataz Mahmoud. 2018. Global Association of air Pollution and Cardiorespiratory Diseases: a systematic review, meta-analysis, and investigation of modifier variables. *American journal of public health* 108, S2 (2018), S123–S130.

[31] Saïd Salhi and Gábor Nagy. 1999. A cluster insertion heuristic for single and multiple depot vehicle routing problems with backhauling. *Journal of the operational Research Society* 50, 10 (1999), 1034–1042.

[32] Said Salhi and M Sari. 1997. A multi-level composite heuristic for the multi-depot vehicle fleet mix problem. *European Journal of Operational Research* 103, 1 (1997), 95–112.

[33] Said Salhi, Sam Rabindranath Thangiah, and Fuad Rahman. 1998. A genetic clustering method for the multi-depot vehicle routing problem. In *Artificial Neural Nets and Genetic Algorithms*. Springer, 234–237.

[34] Minea Skok, Davor Skrlec, and Slavko Krajcar. 2000. The non-fixed destination multiple depot capacitated vehicle routing problem and genetic algorithms. In *Information Technology Interfaces, 2000. ITI 2000. Proceedings of the 22nd International Conference on*. IEEE, 403–408.

[35] Minea Skok, Davor Skrlec, and Slavko Krajcar. 2001. The Genetic Algorithm Scheduling of Vehicles from Multiple Depots to a Number of Delivery Points. *Arficial Intelligence* 349 (2001).

[36] Petr Stodola. 2018. Using Metaheuristics on the Multi-Depot Vehicle Routing Problem with Modified Optimization Criterion. *Algorithms* 11, 5 (2018), 74.

[37] Thomas Stützle and Holger H Hoos. 2000. MAX–MIN ant system. *Future generation computer systems* 16, 8 (2000), 889–914.

[38] P Surekha and S Sumathi. 2011. Solution to multi-depot vehicle routing problem using genetic algorithms. *World Applied Programming* 1, 3 (2011), 118–131.

[39] Sam R Thangiah and Said Salhi. 2001. Genetic clustering: an adaptive heuristic for the multidepot vehicle routing problem. *Applied Artificial Intelligence* 15, 4 (2001), 361–383.

[40] Frank A Tillman. 1969. The multiple terminal delivery problem with probabilistic demands. *Transportation Science* 3, 3 (1969), 192–204.

[41] Frank A Tillman and Robert W Hering. 1971. A study of a look-ahead procedure for solving the multiterminal delivery problem. *Transportation Research* 5, 3 (1971), 225–229.

[42] Zhang Wenjing and Jianzhong Ye. 2010. An improved particle swarm optimization for the multi-depot vehicle routing problem. In *2010 International Conference on E-Business and E-Government*. IEEE, 3188–3192.

[43] Anthony Wren and Alan Holliday. 1972. Computer scheduling of vehicles from one or more depots to a number of delivery points. *Journal of the Operational Research Society* 23, 3 (1972), 333–344.

[44] Tang Yalian. 2016. An improved ant colony optimization for multi-depot vehicle routing problem. *Int. J. Eng. Tech* 8 (2016), 385–388.

[45] Baozhen Yao, Ping Hu, Mingheng Zhang, and Xiaomei Tian. 2014. Improved ant colony optimization for seafood product delivery routing problem. *PROMET-Traffic&Transportation* 26, 1 (2014), 1–10.

[46] Bin Yu, ZZ Yang, and JX Xie. 2011. A parallel improved ant colony optimization for multi-depot vehicle routing problem. *Journal of the Operational Research Society* 62, 1 (2011), 183–188.