Ant Colony Optimization and Its Application to the Vehicle Routing Problem with Pickups and Deliveries

Bülent Çatay

Abstract. Ant Colony Optimization (ACO) is a population-based metaheuristic that can be used to find approximate solutions to difficult optimization problems. It was first introduced for solving the Traveling Salesperson Problem. Since then many implementations of ACO have been proposed for a variety of combinatorial optimization problems. In this chapter, ACO is applied to the Vehicle Routing Problem with Pickups and Deliveries (VRPPD). VRPPD determines a set of vehicle routes originating and ending at a single depot and visiting all customers exactly once. The vehicles are not only required to deliver goods but also to pick up some goods from the customers. The objective is to minimize the total distance traversed. The chapter first provides an overview of the ACO approach. Next, VRPPD is described and the related literature is reviewed. Then, an ACO approach for VRPPD is presented. The approach proposes a new visibility function which attempts to capture the "delivery" and "pickup" nature of the problem. The performance of the approach is tested using well-known benchmark problems from the literature.

1 Introduction

Ant Colony Optimization (ACO) is a population-based metaheuristic that can be used to find approximate solutions to difficult optimization problems [16]. It was first introduced for solving the Traveling Salesperson Problem (TSP) [15, 18]. Since then many implementations of ACO have been proposed for a variety of combinatorial optimization problems such as Quadratic Assignment Problem [34], Scheduling Problems [11], Sequential Ordering Problem [21], and various Vehicle Routing Problems [5, 6, 14, 22, 30, 31].

The approach is based on the observation of the behavior of real ant colonies searching for food sources. Real ants deposit an aromatic essence, called pheromone,

Bülent Çatay
Sabanci University, Faculty of Engineering and Natural Sciences,
Tuzla, 34956 Istanbul, Turkey
e-mail: catay@sabanciuniv.edu

on the path they walk. Other ants searching for food sense that pheromone and use this information in selecting their path. The quantity of pheromone deposited on a path is based on the length of the path and the quality of the food source. As more ants follow a path the level of pheromone on that path will increase, thus increasing its selection probability by other ants. In ACO, artificial ants are used for searching good solutions to an optimization problem by taking advantage of this cooperative learning process.

In this chapter, we apply the ACO approach to the well-known Vehicle Routing Problem with Pickups and Deliveries (VRPPD). The classical Vehicle Routing Problem (VRP) involves a set of delivery customers to be serviced by a fleet of vehicles housed at a central depot. The objective of the problem is to develop a set of vehicle routes originating and terminating at the depot such that all customers are serviced, the demands of the customers assigned to each route do not violate the capacity of the vehicle that services the route, and the total distance traveled by all vehicles is minimized. VRPPD is a variant of the VRP where the vehicles are not only required to deliver goods to customers but also to pick up some goods from the customers. Customers receiving goods are called linehauls and customers sending goods are called backhauls. VRPPD may be classified into three categories: (i) *Deliveries First, Pickups Second*: the vehicles pick up goods only after they have delivered their goods; (ii) *Mixed Pickups and Deliveries*: the vehicles deliver and pick up goods in any sequence along their routes; and (iii) *Simultaneous Pickups and Deliveries*: the vehicles simultaneously deliver and pick up goods [28].

VRP with delivery first, pickup second is the first VRPPD problem introduced in the literature and is known as the VRP with Backhauls (VRPB). The reason why the vehicles have to finish delivering their load before they start picking up items may be due to the difficulty of rearranging the delivery and pickup items on the vehicles, e.g. rear loaded vehicles. However, it is also possible to perform both tasks in any order or simultaneously when the vehicle is nearly empty or is designed for both rear and side loading and unloading. Hence, several variants of this problem have been proposed over time relaxing the restriction of servicing backhaul customers after the linehauls as well as introducing multiple-depot cases. In this chapter, we consider two of these variants: Mixed VRP with Backhauls (MVRPB) and VRP with Simultaneous Pickups and Deliveries (VRPSPD). In MVRPB and VRPSPD the objective and constraints are the same as in VRPB except the servicing order of the customers, which makes the former two problems more complicated because of the fluctuating loads on the vehicle along the route.

VRPB has been extensively studied in the literature. However, the research on MVRPB is scant and VRPSPD has only recently received some attention. These two problems are more realistic and applicable to real-world situations. This chapter attacks these problems using an ant algorithm and is organized as follows: Section 2 depicts the mechanism of the ACO metaheuristic and summarizes some of the variants proposed in the literature. Section 3 is devoted to the description of VRPPD and the overview of various approaches proposed for solving the problem. Section 4 introduces an ACO approach by proposing a new visibility function and Section 5 presents the computational experiments and numerical results. Finally, concluding remarks are given in the last section.

2 Ant Colony Optimization

ACO is a metaheuristic approach designed for solving hard combinatorial optimization problems. Real ant colonies deposit pheromone on the paths they walk while searching for food sources. If other ants searching for food sense the pheromone on a path, they are likely to follow it rather than traveling at random, thus reinforcing the path. As more and more ants follow a path the level of pheromone on that path will enhance, which in turn will increase its selection probability by other ants. On the other hand, the pheromone evaporates over time, reducing the chance of other ants following the path. The longer the path between the nest and the food source the more the pheromone evaporates. Thus, the pheromone levels remain higher on the shorter paths. As a consequence, the level of pheromone laid is basically based on the path length and the quality of the food source.

The experimental setting given in Figure 1 illustrates the above described behavior of the real ants. Figure 1.(a) shows a path that has been formed by ants walking between the food source A and the nest E. When the path is cut off with an obstacle as shown in Figure 1.(b) the ants located at point B walking from A to E and those located at point D walking from E to A have to choose either the path passing through point C or the path passing through point H. Since there is no previous pheromone trail on any of the two alternative paths, the selection of either path by the first ants reaching these points is equally likely. Since the path BCD is shorter than the path BHD the ant that has selected the path through point C will arrive at point D before the ant that has selected the path through point H. Hence, an ant returning from E to A and located at point D will find a stronger trail on path DCB

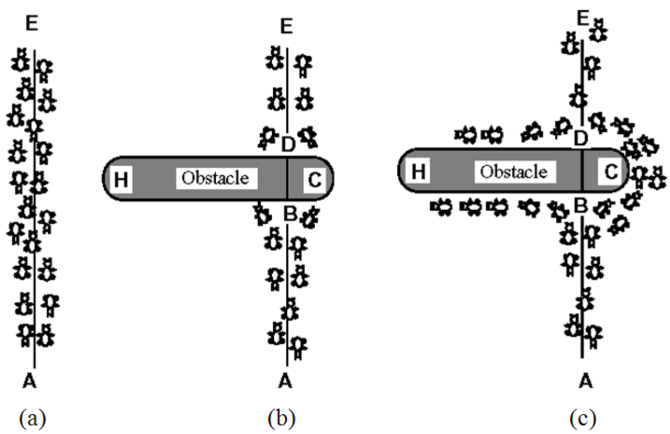

Fig. 1 The natural behavior of real ants [18]

due to the ants that have already selected that path by chance and those walking through BCD. Therefore, the selection probability of path DCB will be larger than that of path DHB. Consequently, the amount of pheromone on path BCD will increase faster than the pheromone on path BHD because of the larger number of ants following path BCD per unit time and the evaporation factor. In time, all ants will select the shorter path [18].

ACO simulates this natural behavior of real ants to solve combinatorial optimization problems by using artificial ants. To apply ACO, the optimization problem is transformed into the problem of finding the best path on a weighted graph. The artificial ants incrementally build solutions by moving on the graph using a stochastic construction process guided by artificial pheromone and heuristic information known as visibility [16]. The amount of pheromone deposited on arcs is proportional to the quality of the solution generated and increases at run-time during the computation.

The Ant System (AS) is the first ACO algorithm which was applied for solving the TSP [15, 18]. Given a number of cities and the costs of traveling from any city to any other city, TSP aims at finding the least-cost round-trip route that visits each city exactly once and then returns to the starting city. In AS, each ant probabilistically chooses the next city to visit based on a heuristic combining the distance to that city and the amount of virtual pheromone deposited on the arc to that city. The ants explore, depositing pheromone on each arc they cross, until they have all completed a tour. At this point the ant which has completed the shortest tour deposits virtual pheromone along its complete tour. The amount of pheromone deposited is inversely proportional to the tour length; i.e., the shorter the tour, the more amount of pheromone the ant deposits on the arcs of the corresponding tour.

Although AS provided competitive results its performance was still inferior in large instances compared to other algorithms specifically designed for the TSP [19]. However, its successful application has led to many extensions for various combinatorial optimization problems utilizing a similar construction mechanism. Some early applications include the elitist strategy for Ant System (EAS) [15, 18], rank-based version of Ant System (ASrank) [6], *MAX-MIN* Ant System (*MM*AS) [35], Ant Colony System (ACS) [17], and Multiple Ant Colony System (MACS) [22].

In the next section we provide a more detailed explanation of the mechanisms of AS approach and its extensions applied to the TSP.

2.1 Ant System

In AS , K artificial ants probabilistically construct tours in parallel exploiting a given pheromone model. Initially, all ants are placed on randomly chosen cities. At each iteration, each ant moves from one city to another, keeping track of the partial solution it has constructed so far. The algorithm has two fundamental components:

- The amount of pheromone on arc (i, j), τ_{ij}
- Desirability of arc (i, j), η_{ij}

where arc (i, j) denotes the connection between city i and city j.

At the start of the algorithm an initial amount of pheromone τ_0 is deposited on each arc: $\tau_{ij} = \tau_0 = K/L_0$, where L_0 is the length of an initial feasible tour and K is the number of ants. In AS, the initial tour is constructed using the nearest-neighbor algorithm; however, another TSP heuristic may be utilized as well. The desirability value (also referred to as visibility or heuristic information) between a pair of cities is the inverse of their distance $\eta_{ij} = 1/d_{ij}$, where d_{ij} is the distance between cities i and j. So, if the distance on the arc (i, j) is long, visiting city j after city i (or vice-versa) will be less desirable.

Each ant constructs its own tour utilizing a transition probability: an ant k positioned at a city i selects the next city j to visit with a probability given by

$$p_{ij}^k = \begin{cases} \dfrac{\tau_{ij}^\alpha \eta_{ij}^\beta}{\sum\limits_{l \in N_i^k} \tau_{il}^\alpha \eta_{il}^\beta} & \text{, if } j \in N_i^k \\ 0 & \text{, otherwise} \end{cases} \tag{1}$$

Here, N_i^k denotes the set of not yet visited cities; α and β are positive parameters to control the relative weight of pheromone information τ_{ij} and heuristic information η_{ij}. Note that $\tau_{ij}^\alpha \eta_{ij}^\beta$ is also referred to as the attractiveness and is denoted as φ_{ij}.

After each ant has completed its tour, the pheromone levels are updated. The pheromone update consists of the pheromone evaporation and pheromone reinforcement. The pheromone evaporation refers to uniformly decreasing the pheromone values on all arcs. The aim is to prevent the rapid convergence of the algorithm to a local optimal solution by reducing the probability of repeatedly selecting certain cities. The pheromone reinforcement process, on the other hand, allows each ant to deposit a certain amount of pheromone on the arcs belonging to its tour. The aim is to increase the probability of selecting the arcs frequently used by the ants that construct short tours. The pheromone update rule is the following:

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \sum_{k=1}^{K} \Delta\tau_{ij}^k, \qquad \forall (i, j) \tag{2}$$

In this formulation, ρ $(0 < \rho \leq 1)$ is the pheromone evaporation parameter and $\Delta\tau_{ij}^k$ is the amount of pheromone deposited on arc (i, j) by ant k and is computed as follows:

$$\Delta\tau_{ij}^k = \begin{cases} \frac{1}{L^k} & \text{, if ant } k \text{ uses arc } (i, j) \text{ on its tour} \\ 0 & \text{, otherwise} \end{cases} \tag{3}$$

where L^k is the length of tour constructed by ant k.

Prior to the pheromone update a local search procedure may be applied on the tours constructed by the ants to reduce the distance traversed. It has been observed that such a procedure enhances the performance of the AS algorithm. In Figure 2 an overview of the steps of the algorithm is provided.

```
compute visibility
initialize pheromone levels
while (max number of iterations is not reached)
   for each ant
     while (not all cities are visited)
       select a not yet visited city
       update tour length and list of not yet visited cities
     end while
     perform local search
   end for
   perform pheromone update
   save the best-so-far solution
end while
```

Fig. 2 Description of AS

2.2 The Extensions of AS

In the EAS [15, 18] an elitist strategy is implemented by further increasing the pheromone levels on the arcs belonging to the best tour achieved since the initiation of the algorithm. That best-so-far tour is referred to as the "global-best" tour. The pheromone update rule is as follows:

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \sum_{k=1}^{K} \Delta\tau_{ij}^{k} + w\Delta\tau_{ij}^{gb}, \qquad \forall (i,j) \tag{4}$$

Here, w denotes the weight associated with the global-best tour and $\Delta\tau_{ij}^{gb}$ is the amount of pheromone deposited on arc (i, j) by the global-best ant and calculated by the following formula:

$$\Delta\tau_{ij}^{gb} = \begin{cases} \frac{1}{L^{gb}} & \text{, if global best ant uses arc } (i,j) \text{ on its tour} \\ 0 & \text{, otherwise} \end{cases} \tag{5}$$

where L^{gb} is the length of global-best tour.

In the AS_{rank} [6] a rank-based elitist strategy is adopted in an attempt to prevent the algorithm from being trapped in a local minimum. In this strategy, w best-ranked ants are used to update the pheromone levels and the amount of pheromone deposited by each ant decreases with its rank. Furthermore, at each iteration, the global-best ant is allowed to deposit the largest amount of pheromone. The update rule is the following:

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \sum_{r=1}^{w-1} (w-r)\Delta\tau_{ij}^{r} + w\Delta\tau_{ij}^{gb}, \qquad \forall (i,j) \tag{6}$$

The ACS presented in [17] attempts to improve AS by increasing the importance of exploitation versus exploration of the search space. This is achieved by employing a strong elitist strategy to update pheromone levels and a pseudo-random proportional rule in selecting the next node to visit. The strong elitist strategy is applied by using the global-best ant only to increase the pheromone levels on the arcs that belong to the global-best tour:

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \rho\Delta\tau_{ij}^{gb}, \qquad \forall(i,j) \tag{7}$$

The mechanism of the pseudo-random proportional rule is as follows: an ant k located at customer i may either visit its most favorable customer or randomly select a customer. The selection rule is the following:

$$j^k = \begin{cases} \arg\max\limits_{j \in N_i^k} \tau_{ij}^{\alpha}\eta_{ij}^{\beta} & \text{, if } z \leq z_0 \\ J^k & \text{, otherwise} \end{cases} \tag{8}$$

where z is a random variable drawn from a uniform distribution $U[0,1]$ and z_0 ($0 \leq z_0 \leq 1$) is a parameter to control exploitation versus exploration. J^k is selected according to the probability distribution (1). ACS also uses local pheromone updating while building solutions: as soon as an ant moves from city i to city j the pheromone level on arc (i, j) is reduced in an attempt to promote the exploration of other arcs by other ants. The local pheromone update is performed as follows:

$$\tau_{ij} \leftarrow (1-\xi)\tau_{ij} + \xi\tau_0 \tag{9}$$

where ξ is a positive parameter less than 1.

Similar to ACS, *MM*AS [35] uses either the global-best ant or the iteration-best ant alone to reinforce the pheromone. It has been observed that using iteration-best ant at the start of the algorithm and then gradually increasing the frequency of using the global-best ant for the pheromone update improves the performance. However, this strategy may cause a rapid convergence to a sub-optimal solution. Thus, maximum and minimum limits on the pheromone levels are imposed to avoid stagnation. The interval in which the pheromone may vary is set to $[\tau_{min}, \tau_{max}]$. The pheromone levels are initialized at τ_{max} to allow the exploration of the search space at the beginning. In addition, the pheromone levels are reinitialized whenever the system approaches stagnation or no improvement has been achieved after a number of consecutive iterations.

Gambardella et al. [22] developed a multiple ACS (MACS) for solving the VRP with Time Windows (VRPTW). VRPTW has two objectives: to minimize the number of vehicles used and the total tour time. The former is considered to be the primary objective, i.e. a solution with less number of vehicles but longer travel time is preferred over a solution with more vehicles but shorter travel time. MACS attempts to minimize both objectives simultaneously by using two parallel ant colonies. The first colony, named as ACS-VEI, reduces the number of vehicles while the second, named as ACS-TIME, minimizes the total tour time by using the number of vehicles provided by ACS-VEI. Although the two ant colonies run in parallel they use independent pheromone trails.

The interested reader is referred to [19] for more details on ACO metaheuristic and its variants.

3 Vehicle Routing Problem with Pickups and Deliveries

In this section, we first describe VRPPD and present a 0-1 mixed integer linear programming model following the formulation of [13]. We next review the existing literature on MVRPB and VRPSPD.

3.1 Problem Description

VRPPD deals with a single depot distribution/collection system servicing a set of customers by means of a homogeneous fleet of vehicles, i.e. all vehicles have the same capacity. The customers may require two types of service: a delivery and/or a pickup. Products to be delivered are loaded at the depot and products picked up are transported back to the depot. The objective is to find the set of vehicle routes servicing all the customers with the minimum total distance. A maximum route length restriction may be imposed on the vehicles.

In VRPB, each customer has either a delivery or a pickup demand to be satisfied and the vehicle services the linehaul customers first. The main reasoning behind visiting linehaul customers before backhaul customers is the fact that linehaul customers have precedence over backhaul customers in many real world cases and vehicles are often rear loaded. The latter causes problems when rearranging the items on the vehicle, thus preventing the mixed routes and simultaneous pickup and delivery. However, the improved design of vehicles allows side loadings, making the mixed routes a more practical option since that would provide shorter routes. Thus, in MVRPB, each customer has *either a delivery or a pickup* demand and backhaul and linehaul customers may be visited in any order. On the other hand, servicing the customers in any order but not allowing simultaneous pickup and delivery is not practical and realistic in many real world situations. As a result, VRPSPD was proposed where each customer has *both a delivery and a pickup* demand and both services are performed simultaneously. Although the customers can be visited in any order along the route in both problems they must be serviced exactly once.

From a practical point of view VRPPD models situations such as distribution of bottled drinks, chemicals, LPG tanks, laundry service of hotels, etc. where the customers are typically visited for a double service. In the case of the distribution of the bottled drinks for instance, full bottles are delivered to customers and empty ones are brought back either for re-use or for recycling. In the distribution of chemicals case, some hazardous materials may need to be returned for safe disposal. Regulations or environmental issues may also force companies to take responsibility for their products throughout their lifetime and to collect them.

In VRPB, the loads of linehaul customers and backhaul customers can be checked separately during the delivery route and pickup route, respectively, to ensure that the vehicle capacity is not exceeded. In MVRPB, however, the decrease or increase on the vehicle load at each customer must be checked depending on whether the customer is a linehaul or backhaul customer, respectively. Similarly, in VRPSPD, the net change (decrease or increase) on the vehicle load at each customer must be monitored. Therefore, in these two problems the vehicle capacity must not be exceeded at any arc along the route.

Since VRPB is out of the scope of this study we omit further discussion on the problem and refer the interested reader to [4, 23, 36] for details.

3.2 Problem Formulation

Mathematically, VRPSPD is described by a set of homogenous vehicles V, a set of customers J, and a complete undirected graph $G(N, A)$. The graph consists of n+1 vertices where the customers are denoted by 1, 2, ..., n and the depot is represented by the vertex 0. $A = \{(i, j): i, j \in N, i \neq j\}$ denotes the set of arcs that represents connections between the depot and the customers and among the customers. A cost (time, distance) c_{ij} is associated with each arc (i, j). Each vehicle has capacity Q and each customer (node) i is characterized by its geographical location and its delivery and pickup requests D_i and P_i, respectively. Finally, Q, D_i, P_i, and c_{ij} are assumed to be non-negative integers. The VRPSPD determines a set of paths (routes) such that:

1. each vehicle travels exactly one route;
2. each customer is visited only once by one of the vehicles completely satisfying its demand and supply;
3. the load carried by a vehicle between any pair of adjacent customers on the route must not exceed its capacity; and
4. total distance given by the sum of the arcs belonging to these routes is minimal.

In addition, a maximum route length (maximum time) restriction may be imposed on the vehicles.

Following the model in [13] the 0-1 mixed integer linear programming model of VRPSPD can be formulated as follows:

Decision Variables

L_j load of vehicle after having serviced customer $j \in J$

z_j subtour elimination variable

$$x_{ijv} = \begin{cases} 1 & \text{, if vehicle } v \text{ travels directly from customer } i \text{ to } j \\ 0 & \text{, otherwise} \end{cases}$$

Mathematical Model

$$\text{Minimize} \quad z = \sum_{i \in N} \sum_{j \in N} \sum_{v \in V} c_{ij} x_{ijv} \tag{10}$$

Subject to

$$\sum_{j \in N} \sum_{v \in V} x_{ijv} = 1 \qquad i \in J \tag{11}$$

$$\sum_{i \in N} \sum_{v \in V} x_{ijv} = 1 \qquad j \in J \tag{12}$$

$$\sum_{i \in N} x_{ikv} - \sum_{j \in N} x_{kjv} = 0 \qquad k \in J, v \in V \tag{13}$$

$$L_j \geq \sum_{i \in N} \sum_{j \in J} D_j x_{ijv} - D_j + P_j - M\left(1 - x_{0jv}\right) \quad j \in J, v \in V \tag{14}$$

$$L_j \geq L_i - D_j + P_j - M\left(1 - \sum_{v \in V} x_{ijv}\right) \qquad i, j \in J, i \neq j \tag{15}$$

$$\sum_{i \in N} \sum_{j \in J} D_j x_{ijv} \leq Q \qquad v \in V \tag{16}$$

$$L_j \leq Q \qquad j \in J \tag{17}$$

$$z_j \geq z_i + 1 - n\left(1 - \sum_{v \in V} x_{ijv}\right) \qquad i, j \in J, i \neq j \tag{18}$$

$$z_j \geq 0 \qquad j \in J \tag{19}$$

$$x_{ijv} \in \{0, 1\} \qquad i, j \in N, v \in V \tag{20}$$

where M is a sufficiently large number (e.g. $M = \max\left\{\sum_{j \in N_c} (D_j + P_j), \sum_{i \in N} \sum_{\substack{j \in N \\ j \neq i}} C_{ij}\right\}$).

The objective function (10) minimizes the total distance traveled. Constraint sets (11) and (12) assure servicing each customer exactly once. Constraint (13) makes sure that if a vehicle arrives at a customer, then the same vehicle departs from it. The load after servicing the first customer is defined with constraint (14) while the load "en route" is limited with constraint (15). Constraint sets (16) and (17) ensure that the load when leaving the depot and "en route", respectively, does not exceed the vehicle capacity. Constraint (18) is the subtour elimination constraint. Constraint (19) is the non-negativity constraint and constraint (20) defines the binary variables.

MVRPB may be considered as a special case of VRPSPD in which some of the customers require only delivery service while the remaining customers require only pickup service. In other words, we define J_L as the set of linehaul customers, $J_L = \{j: j \in J, D_j > 0, P_j = 0\}$, J_B as the set of backhaul customers, $J_B = \{j: j \in J, D_j = 0, P_j > 0\}$, and $J = J_L \cup J_B$. Therefore, the above VRPSPD model also formulates MVRPB where $P_j = 0$ for $j \in J_L$ and $D_j = 0$ for $j \in J_B$.

We can prove that VRPSPD and MVRPB are *NP*-hard in the following way: Let $J_B = \oslash$. Then MVRPB reduces to VRP, which is known to be *NP*-hard. Hence, MVRPB is also *NP*-hard. Since MVRPB is a special case of VRPSPD, VRPSPD is *NP*-hard as well.

3.3 Literature Review

Although research on VRPSPD has recently gained momentum, there are only a few papers attacking MVRPB. Golden et al. [24] developed an algorithm which inserts backhaul customers into the routes formed by the linehaul customers. The algorithm utilizes a penalty factor which considers the number of linehaul customers left on the route after the insertion point.

Casco et al. [7] proposed a load-based insertion procedure where the insertion cost for backhaul customers is determined based on the remaining load to be delivered along the route of the vehicle. Salhi and Nagy [33] modified this method by proposing the cluster insertion of backhauls to solve both MVRPB and VRPSPD. In their problem structure, nodes are represented as disjoint delivery or pickup nodes; thus repetitive servicing is allowed. Salhi and Nagy also investigated the case with multiple depots. Nagy and Salhi [28] improved their previous approach using several heuristics in which they first find a solution to the VRP by allowing infeasibilities then modify this solution to make it feasible for the MVRPB and VRPSPD. The proposed approach is capable of solving both single- and multi-depot problems.

Wade and Salhi [38] proposed an ant algorithm which uses the ACS approach of Dorigo and Gambardella [17]. However, the computational results were rather poor compared to those in the literature. Wade and Salhi [39] further enhanced their ant algorithm by using different mechanisms, and hence, improved their earlier results. Recently, Ropke and Pisinger [32] developed a unified heuristic for a large class of VRPPD based on a large neighborhood search (LNS). The proposed heuristic provides competitive results for both MVRPB and VRPSPD.

VRPSPD was first introduced by Min [26] as a book distribution and collection problem between a central library and 22 remote libraries in Ohio using two vehicles. Min utilizes a cluster-first route-second approach and solves the TSP to optimality as sub-problems. Halse [25] proposed a cluster-first route-second approach for VRPSPD as well as for several other variants of VRPPD. In this approach the nodes are first distributed to vehicles and then the problem is solved using the 3-opt algorithm. Halse also utilizes Lagrangean relaxation and column generation techniques and discusses the results for single depot instances with 22 to 150 customers.

Angelelli and Mansini [2] addressed the VRPSPD with time windows constraints. They developed a branch-and-price approach based on a set covering formulation for the master problem. A relaxation of the elementary shortest path problem with time windows and capacity constraints is used as the pricing problem. Branch-and-bound is applied to obtain integer solutions.

Dethloff [13] presented insertion-based heuristics using four different criteria. He developed 40 instances to test his algorithm. He also compared his results with those of Salhi and Nagy [33] and reported an improvement on Min's problem. Vural et al. [37] reported improvements on the results of Dethloff problems employing a dual genetic algorithm approach. In this approach first tours are created and partitioned into sub-tours, then a local search is performed, and finally crossover and mutation operations are executed.

Crispim and Brandão [12] proposed a hybrid tabu search (TS)-variable neighborhood search approach. The approach uses the sweep algorithm to construct the initial solution and then performs arc exchanges in the TS procedure. If it is faced with any overloads it exchanges the order of the customers on the route until it achieves feasibility. For the improvement phase it uses "insert" and "swap" moves by penalizing the overloads. Other TS approaches for VRPSPD include Chen and Wu [8] which presented an insertion-based procedure followed by a hybrid heuristic based on the record-to-record travel, tabu lists, and improvement procedures; Montané and Galvão [27] which developed a TS algorithm using "insert", "exchange", "split and splice" (on two routes), and 2-opt; Bianchessi and Righini [3] which proposed constructive algorithms, local search algorithms, and TS algorithms to obtain approximate solutions fast; and Wassan et al. [40] which designed a reactive tabu search metaheuristic. The last approach provides competitive results.

In what follows we propose an ACO algorithm for efficiently solving MVRPB and VRPSPD and test its performance against the approaches presented in the above mentioned papers.

4 An Ant Algorithm for VRPPD

An initial solution is first obtained using the nearest-neighbor heuristic: start at the depot and then select the not yet visited closest feasible customer as the next customer to be visited. A customer is feasible if visiting her next does not violate the capacity constraint (and the maximum route length restriction, if any). If no feasible customer is available then the route is terminated at the depot and a new route is initiated. The procedure is repeated until all customers are serviced. This solution is used to initialize the pheromone trails on the arcs as follows: $\tau_0 = n/L_0$, where L_0 is the length of the nearest-neighbor heuristic solution. Note that this heuristic does not guarantee feasibility if a limit on the number of vehicles is imposed.

4.1 Heuristic Information

In the classical ant approaches developed for solving TSP and VRP the visibility value between a pair of customers is the inverse of their distance. On the other hand, [6, 14] employed the savings function as the visibility function for solving the VRP. While the latter utilized the classical Clarke and Wright savings function ([10]) the former used the Paessens' parametrical savings function ([29]). The classical savings function calculates the savings in distance achieved by serving two customers i and j on the same route instead of serving them on different routes using the following formula:

$$S_{ij} = (c_{0i} + c_{i0} + c_{0j} + c_{j0}) - (c_{0i} + c_{ij} + c_{j0}) = c_{i0} + c_{0j} - c_{ij} \tag{21}$$

where c_{i0} (c_{0j}) is the distance of customer i (j) to the depot and c_{ij} represents the distance between the customer i and j. Since a high value of savings indicates that visiting customer j after customer i is a desired choice the tour length is expected to be shorter if the probability of moving from customer i to customer j increases with the savings value.

Paessens' formulation aims at collecting more information about the distribution of the customers in an attempt to avoid the circumferenced formation of routes. The proposed parameterized formulation is as follows:

$$S_{ij} = c_{i0} + c_{0j} - \lambda c_{ij} + \mu \left| c_{i0} - c_{0j} \right| \tag{22}$$

where λ and μ are non-negative constants.

In our approach, the visibility function consists of two components. The first component is an enhanced savings function developed by [20]:

$$S_{ij} = c_{i0} + c_{0j} - \lambda c_{ij} + \mu cos(\theta_{ij}) \left| c_{0i} - c_{j0} \right| \tag{23}$$

where θ_{ij} is the angle formed by the two rays originating from the depot and crossing the customers i and j, and λ and μ are non-negative parameters. The proposed savings function (23) can be regarded as a more general enhancement to Paessens' savings formulation and was shown to perform better on various problem sets.

The second component takes into consideration the load of the vehicle on its route. This component is equal to the largest of the ratio of delivery to customer j to the average value of all deliveries and the ratio of pickup from customer j to the average value of all pickups if total deliveries or total pickups so far have exceeded half of the vehicle capacity; and is equal to 1 otherwise. The idea is to basically give more chance of selection to customers requiring larger delivery or pickup quantities. Our motivation in doing so stems from the "put first larger items" approach used in [1]. The reason why we start employing this approach after half of the vehicle capacity is used up is to let the first component determine the selection of the customers at the early stages of the route construction in an attempt to not adversely affect the influence of this heuristic information on building a shorter route. In other words, the first component acts as a primary heuristic information whereas the second component starts playing a role after we have already constructed a partial tour using only the distance criterion. The computation of the second visibility value is as follows:

$$R_j = \begin{cases} \max\left(\frac{P_j}{\overline{P}}, \frac{D_j}{\overline{D}}\right) & \text{, if } \min\left(\sum\limits_{k \in V_q} P_k, \sum\limits_{k \in V_q} D_k\right) > \frac{Q}{2} \\ 1 & \text{, otherwise} \end{cases} \tag{24}$$

Here, $\overline{D}$ ($\overline{P}$) is the average delivery (pickup) and V_q is the set of customers already visited by the associated vehicle q. Note that the first component is static whereas the second depends on the current load of the vehicle. The visibility function is then the following:

$$\eta_{ij} = S_{ij} \times R_j \tag{25}$$

4.2 Route Construction

The route construction process uses the pseudo-random proportional rule of ACS depicted in Section 2.2. In addition, a candidate list is used in selecting a customer to visit, i.e. N_{ik} consists of a number of customers that have not been visited yet and have the largest attractiveness values.

After an ant has constructed its tour, a local search is performed in an attempt to further improve the solution. In our algorithm we use the "swap" and "move" procedures sequentially. In swap two customers are exchanged whereas in move a customer is removed and inserted into another arc. These procedures are applied both within routes and between different routes.

4.3 Pheromone Update

Our pheromone update consists of a rank-based *MM*AS strategy. In this strategy, w best-ranked ants of each iteration along with the best-so-far ant are used to update the pheromone trails. The pheromone reinforcement of each ant is proportional to its rank. Our pheromone update rule is as follows:

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \sum_{r=1}^{w}(w-r)\Delta\tau_{ij}^{r} + w\Delta\tau_{ij}^{gb} \tag{26}$$

Here, $\Delta\tau_{ij}^{r}$=$1/L^r$ for all arcs $(i,\ j)$ belonging to the tour built by the r^{th} best ant where L^r is the length of the corresponding tour. gb denotes the global-best ant. If the pheromone level on any arc drops below an explicit lower limit or exceeds an explicit upper limit it is set equal to that limit. In other words, if any $\tau_{ij} < \tau_{min}$ ($\tau_{ij} > \tau_{max}$) then $\tau_{ij} = \tau_{min}$ ($\tau_{ij} = \tau_{max}$). The aim in using this *MM*AS approach is to reduce the risk of a premature convergence.

5 Experimental Study

The proposed algorithm is coded using C++ and executed on an Intel Pentium T2130 1.86 GHz processor with 1 Gb RAM. The parameters in the savings function are $\lambda = \mu = 1$. The parameters of the ACO algorithm are set according to initial experimental runs as: $z_0 = 0.7, \alpha = 1, \beta = 4, \rho = 0.1, \tau_{max} = n/\rho L^{gb}$, and $\tau_{min} = \rho\tau_{max}/50$. For consistency, the same parameter values are used for solving both VRPSPD and MVRPB. The number of best ants used for the pheromone reinforcement and the size of the candidate list used in the selection of the next customer to be visited are proportional to the number of ants and the number of customers, respectively, and their values are set to $w = n/10$ and $s = n/5$, respectively. Since setting the number of ants equal to the number of customers has been observed to perform well in the literature (see e.g. [19]) we also adopted the same strategy. For each problem instance we performed 10 runs, each carried on for 100 iterations.

Table 1 Results for the VRPSPD data set of Dethloff [13]

Problem	MG	ACO Avg	ACO Best	%Gap
SCA3-0	640.55	640.47	636.06	**-0.71**
SCA3-1	697.84	708.59	700.50	0.38
SCA3-2	659.34	662.72	659.86	0.08
SCA3-3	680.04	685.13	680.04	0.00
SCA3-4	690.50	691.26	690.50	0.00
SCA3-5	659.90	673.27	662.75	0.43
SCA3-6	653.81	661.08	653.69	**-0.02**
SCA3-7	659.17	668.83	659.17	0.00
SCA3-8	719.70	724.62	720.06	0.05
SCA3-9	681.00	687.91	682.33	0.20
SCA8-0	981.47	992.60	978.10	**-0.34**
SCA8-1	1077.44	1085.45	1079.92	0.23
SCA8-2	1050.98	1058.43	1046.20	**-0.46**
SCA8-3	983.34	1027.30	1006.59	2.31
SCA8-4	1073.46	1098.11	1077.01	0.33
SCA8-5	1047.24	1081.44	1067.29	1.88
SCA8-6	995.59	1004.30	990.44	**-0.52**
SCA8-7	1068.56	1088.35	1079.20	0.99
SCA8-8	1080.58	1100.73	1086.72	0.57
SCA8-9	1084.80	1091.19	1074.40	**-0.97**
CON3-0	631.39	619.30	617.98	**-2.17**
CON3-1	554.47	562.89	558.69	0.76
CON3-2	522.86	522.55	519.11	**-0.72**
CON3-3	591.19	591.63	591.19	0.00
CON3-4	591.12	597.08	590.49	**-0.11**
CON3-5	563.70	572.44	564.88	0.21
CON3-6	506.19	504.51	501.34	**-0.97**
CON3-7	577.68	591.48	585.51	1.34
CON3-8	523.05	524.51	523.14	0.02
CON3-9	580.05	591.57	588.84	1.49
CON8-0	860.48	873.67	861.40	0.11
CON8-1	740.85	772.33	753.81	1.72
CON8-2	723.32	729.46	725.54	0.31
CON8-3	811.23	852.86	835.77	2.94
CON8-4	772.25	795.59	775.67	0.44
CON8-5	756.91	781.35	769.20	1.60
CON8-6	678.92	701.69	695.43	2.37
CON8-7	814.50	819.04	811.96	**-0.31**
CON8-8	775.59	795.40	775.56	0.00
CON8-9	809.00	834.63	826.95	2.17
Average	764.25	776.64	767.58	0.39

5.1 Benchmark Problems and Results for the VRPSPD

The performance of the algorithm for VRPSPD is tested using two well-known benchmark problem sets from the literature. The first problem set was proposed by Salhi and Nagy [33] based on the 14 VRP problems proposed in [9]. The number of customers in this data varies from 50 to 199 and problems CMT6-10 and CMT13-14 impose a maximum route length restriction for the vehicles. For each VRP instance in [9], Salhi and Nagy generated a VRPSPD problem by splitting the original demand between demand and pickup loads. Another instance was obtained by exchanging these demand and pickup loads of every other customer. Thus, two classes of problems were generated and referred to as problem classes X and Y.

The second problem set was presented by Dethloff [13] where random instances with 50 customers were generated considering two different geographical scenarios: In scenario SCA, the coordinates of the customers are uniformly distributed over the interval [0,100]. In scenario CON, half of the customers are distributed in the same way as in SCA while the coordinates of the other half are uniformly distributed over the interval [100/3,200/3]. The delivery demand D_j of the customers is uniformly distributed over the interval [0,100]. The pickup demand P_j are computed by using a random number r_j that is uniformly distributed over the interval [0,1] such that $P_j = (0.5 + r_j)D_j$. Instances with different vehicle capacities were generated by choosing the minimal number of vehicles μ. Then, the corresponding capacity was set to $C = \sum_{s \in J} D_s/\mu$ where μ was chosen to be 3 or 8.

Table 1[1] compares the best-so-far distances for Dethloff's problems to the average and best distance values found using ACO. Avg and Best columns denote the average distance and best distance, respectively. %Gap column shows the percentage difference between the best-so-far distances and ACO distances and calculated as (Best-so-far/ACO Best)-1. We note that only Ropke and Pisinger [32] and Montané and Galvão [27] utilize this data set in their experiments; however, the former only reports the average results. Thus, we compare our results to those of [27] which are better than those of [13] in all instances. We observe that the ACO improves the solutions of 10 problems and matches 5. Although the overall performance of [27] is better the average gap is only 0.39%. On the other hand, ACO requires more computational effort: 18 seconds versus 3.7 seconds (using Athlon 2.0 GHz processor).

Table 2 provides a comparison of the best solutions published by Dethloff [13], Chen and Wu [8], Montané and Galvão [27], Wassan et al. [40] and those obtained by ACO for Salhi and Nagy problems. In this table column n denotes the number of customers and Veh is the number of vehicles. Note that Nagy and Salhi [28] and Ropke and Pisinger [32] reported only their average results (the former for all problems and the latter for class X only). Hence, we cannot make a detailed comparison.

However, we include their results in Table 3 where we make a comparison of the average solutions. Note also that none of the results in [12] is any better than those published in the literature. Thus, we do not use them as benchmarks. Furthermore,

[1] In all the tables presented in this section, the first letter of the names of the corresponding authors are used for referencing.

Table 2 Comparison of results for the VRPSPD data set of Salhi and Nagy [33]

Problem	*n*	D		CV		MG		WWN		ACO	
		Veh	**Best**	**Veh**	**Best**	**Veh**	**Best**	**Veh**	**Best**	**Veh**	**Best**
CMT1X	50	3	501	3	478.59	3	472	3	**468.30**	3	479.94
CMT2X	75	7	782	6	688.51	7	695	6	**668.77**	6	707.87
CMT3X	100	5	847	5	744.77	5	**721**	4	729.63	5	729.27
CMT4X	150	7	1050	7	887.00	7	880	7	**876.50**	7	914.65
CMT5X	199	11	1348	10	1089.22	11	1098	9	**1044.51**	11	1133.92
CMT6X	50	6	584	-	-	-	-	6	**556.06**	6	556.68
CMT7X	75	11	961	-	-	-	-	11	903.05	11	**901.22**
CMT8X	100	9	928	-	-	-	-	9	879.60	9	**865.51**
CMT9X	150	15	1299	-	-	-	-	15	1220.00	**14**	**1184.34**
CMT10X	199	19	1571	-	-	-	-	19	1464.58	**18**	**1444.90**
CMT11X	120	4	959	4	**858.57**	4	900	4	861.97	4	898.35
CMT12X	100	6	804	6	678.46	6	675	5	**644.70**	6	678.08
CMT13X	120	11	**1576**	-	-	-	-	12	1647.51	11	1596.01
CMT14X	100	10	871	-	-	-	-	10	823.95	10	**821.75**
CMT1Y	50	3	501	3	480.78	3	470	3	**458.96**	3	475.37
CMT2Y	75	7	782	6	679.44	7	700	6	**663.25**	6	699.89
CMT3Y	100	5	847	5	723.88	5	**719**	4	745.46	5	733.82
CMT4Y	150	7	1050	7	**852.35**	7	878	7	870.44	7	894.11
CMT5Y	199	11	1348	10	1084.27	10	1083	9	**1054.46**	10	1112.66
CMT6Y	50	6	584	-	-	-	-	6	558.17	6	**555.43**
CMT7Y	75	11	961	-	-	-	-	11	903.36	11	**901.22**
CMT8Y	100	9	936	-	-	-	-	10	917.42	9	**865.50**
CMT9Y	150	15	1299	-	-	-	-	15	1213.11	**14**	**1184.05**
CMT10Y	199	19	1571	-	-	-	-	18	**1419.79**	18	1437.07
CMT11Y	120	4	1070	5	859.77	5	910	4	**830.39**	4	933.59
CMT12Y	100	5	825	6	676.23	6	689	6	**659.52**	5	**676.21**
CMT13Y	120	11	**1576**	-	-	-	-	11	1647.04	11	1597.03
CMT14Y	100	10	871	-	-	-	-	10	823.34	10	**821.75**
Average			1010.79						912.64		921.43

Table 3 Comparison of the average results for the VRPSPD data set of Salhi and Nagy [33]

Problem	**Type**	**D**	**NS**	**CV**	**RP**	**MG**	**WWN**	**ACO**
CMT-X	**no distance restriction**	899	-	775	-	777	792	756
	distance restriction	1113	-	-	-	-	1053	1071
	All	1006	991	-	919	-	922	914
CMT-Y	**no distance restriction**	918	-	765	-	778	789	755
	distance restriction	1114	-	-	-	-	1052	1069
	All	1016	989	-	-	-	921	912

since the maximum route length constraints were not considered in [8, 27] we only compare the unrestricted problems. The results show that ACO improves 10 best-so-far distances. In addition, ACO finds best-so-far distance and number of vehicles in

Table 4 Improvements on the best-so-far solutions for the VRPSPD data set of Salhi and Nagy [33]

	Best-so-far			ACO	
Problem	**Reference**	**Veh**	**Dist**	**Veh**	**Dist**
CMT7X	WWN	11	903.05	11	**901.22**
CMT8X	WWN	9	879.60	9	**865.51**
CMT9X	WWN	15	1220.00	**14**	**1184.34**
CMT10X	WWN	19	1464.58	**18**	**1444.90**
CMT14X	WWN	10	823.95	10	**821.75**
CMT6Y	WWN	6	558.17	6	**555.43**
CMT7Y	WWN	11	903.36	11	**901.22**
CMT8Y	WWN	10	917.42	10	**865.50**
CMT9Y	WWN	15	1213.11	**14**	**1184.05**
CMT12Y	WWN	6	659.52	**5**	676.21
CMT14Y	WWN	10	823.34	10	**821.75**

problems CMT9X, CMT10X, and CMT9Y. Although the distance is not improved in CMT12Y, the best-so-far distance is obtained using 5 vehicles. The improvements on the best-so-far solutions are summarized in Table 4. We also observe in Table 2 that the average gap in the performance of ACO is inferior to that of Wassan et al. [40] but the difference is less than 1%. On the other hand, we note that ACO requires more computational effort: an average of 615 seconds versus 221 seconds in [40] using UltraSPARC-IIIi 1062 MHz Solaris 9.

5.2 Benchmark Problems and Results for the MVRPB

To test the performance of our algorithm for MVRPB we used the benchmark problems generated by Salhi and Nagy [33] based on 14 VRP problems proposed in [9] and the problems proposed by Goetschalckx and Jacobs-Blecha [23]. For each VRP instance Salhi and Nagy generated three MVRPB problems replacing every second, forth, and tenth delivery customer with a backhaul customer and assigning a pickup quantity equal to its original delivery quantity. Thus, three classes of problems were generated for 50%, 25%, and 10% of backhauls, respectively, and were referred to as problem classes H, Q, and T, respectively. The data set of [23] consists of 63 instances with the number of customers varying from 25 to 150.

Since only Salhi and Nagy [33] reported the individual results we base our comparisons on those results in Table 5. The results show that the performance of ACO is significantly better. However, the average computation time in [33] is less than 4 seconds on a VAX 4000-500 computer whereas our average is 672 seconds. If we investigate the average results of each type of data given in Table 6, we see that ACO outperforms the more recent results of Nagy and Salhi [28]; however, the average results of Ropke and Pisinger [32] are better.

Table 5 Results for the MVRPB data set of Salhi and Nagy [33]

Problem	SN	ACO Avg	ACO Best	ACO Veh	%Gap
CMT1T	541	522.66	520.93	5	-3.71
CMT2T	839	817.50	802.75	9	-4.32
CMT3T	903	831.36	826.20	7	-8.51
CMT4T	1111	1045.28	1023.20	11	-7.90
CMT5T	1423	1306.66	1295.34	15	-8.97
CMT6T	571	556.66	555.43	6	-2.73
CMT7T	-	906.92	903.05	11	-
CMT8T	911	869.81	865.54	9	-4.99
CMT9T	1164	1201.22	1181.34	14	1.49
CMT10T	1418	1469.37	1447.59	18	2.09
CMT11T	1075	1060.00	1042.46	7	-3.03
CMT12T	827	818.65	804.89	9	-2.67
CMT13T	1600	1608.66	1587.17	11	-0.80
CMT14T	866	840.18	829.76	10	-4.18
CMT1Q	557	497.64	492.79	4	-11.53
CMT2Q	860	756.82	745.64	8	-13.30
CMT3Q	918	778.05	764.88	6	-16.68
CMT4Q	1164	969.08	947.19	9	-18.63
CMT5Q	1477	1228.70	1213.41	13	-17.85
CMT6Q	594	558.86	556.68	6	-6.28
CMT7Q	-	903.37	900.69	11	-
CMT8Q	918	871.30	865.50	9	-5.72
CMT9Q	1178	1202.60	1187.16	14	0.78
CMT10Q	1477	1461.64	1452.52	18	-1.66
CMT11Q	1075	990.50	976.04	6	-9.21
CMT12Q	843	755.57	747.94	7	-11.28
CMT13Q	1613	1610.48	1588.68	11	-1.51
CMT14Q	873	827.75	823.11	10	-5.71
CMT1H	594	469.17	465.02	3	-21.71
CMT2H	873	672.76	664.82	6	-23.85
CMT3H	915	748.75	735.43	4	-19.63
CMT4H	1164	896.41	875.15	7	-24.82
CMT5H	1509	1095.15	1080.69	9	-28.38
CMT6H	594	558.40	556.68	6	-6.28
CMT7H	-	901.28	901.22	11	-
CMT8H	915	867.82	865.51	9	-5.41
CMT9H	1164	1200.56	1191.53	14	2.37
CMT10H	1509	1458.58	1446.11	18	-4.17
CMT11H	1120	868.44	852.62	4	-23.87
CMT12H	850	686.47	671.51	5	-21.00
CMT13H	1546	1616.59	1598.75	11	3.41
CMT14H	866	827.69	821.75	10	-5.11
Average	1036.28	955.60	944.63		-8.85

Table 6 Comparison of the average results for the MVRPB data set of Salhi and Nagy [33]

Problem Set	SN	NS	RP	ACO
10% (T)	2008	1011	955	978
25% (Q)	2050	1034	922	947
50% (H)	2088	1045	881	909

Table 7 compares our average distances to those of Wade and Salhi [39]. In this table, n_1 and n_2 columns denote the number of linehaul and backhaul customers, respectively, and column Q denotes the vehicle capacity. The results show that our ACO approach outperforms the ant system algorithm of [39] in 28 instances out of 46. The average improvement is 0.22.

In Table 8, we report our best distances and compare them to the best distances reported by Halse [25] and Wade and Salhi [39]. The results reveal that ACO improves 17 best-so-far solutions. In addition, ACO finds best-so-far distance and number of vehicles in problems f1, l4, and h1. We also observe that ACO provides very competitive results: the average improvement of ACO on Wade and Salhi's results is 0.03% and on Halse's results is 0.65%. On the other hand, ACO's performance is surprisingly poor on problem instance j4.

5.3 Sensitivity of the Results to the Parameter Setting

In the initial experimental study that we conducted for determining suitable parameter values we observed that the algorithm was robust in the sense that fairly good solutions could be obtained for varying parameter values. This is mostly due to the contribution of the local search strategy to the overall solution quality. As mentioned in Section 4.2, we apply the local search at each iteration after each ant has constructed its tour. This exhaustive local search procedure has a significant contribution in achieving relatively short distances fast. To illustrate the effect of the local search mechanism we provide in Figure 3 the results for 5 sample runs of a medium size problem instance with 100 customers, namely CMT3X. This figure also shows the convergence pattern with regard to the 100 iterations. Note that all parameter values are kept same as in the earlier experimental study. We see that the local search procedure provides a significant improvement in the total distance traversed.

To investigate the robustness of the proposed method with respect to the parameter set we show the variations in the solutions of problem CMT3X for various parameter values in Figure 4. The charts (a)-(d) in this figure report the average distance of 10 runs for different values of the (a) heuristic information parameter β, (b) evaporation rate ρ, (c) pseudo-random proportional rule parameter z_0, and (d) number of best ants used for the pheromone update w, respectively — ceteris paribus. We observe that z_0 and w have more effect on the solution quality compared to β and ρ: the maximum deviations from the best distance are 2.6% and 3.4% in the case of

Table 7 Comparison of the average distances of MVRPB data set of Goetschalckx and Jacobs-Blecha [23]

Problem	n_1	n_2	Q	**WS Avg**	**ACO Avg**	**%Gap**
a1	20	5	1550	223374	225002	0.73
a2	20	5	2550	169797	169500	**-0.17**
a3	20	5	4050	142126	142034	**-0.06**
b1	20	10	1600	234514	231667	**-1.21**
b2	20	10	2600	179760	181391	0.91
b3	20	10	4000	145702	145702	0.00
c1	20	20	1800	240126	237979	**-0.89**
c2	20	20	2600	197287	199002	0.87
c3	20	20	4150	165710	166858	0.69
d1	30	8	1700	307383	309454	0.67
d3	30	8	2750	224598	223616	**-0.44**
d4	30	8	4075	-	184171	-
e1	30	15	2650	225927	223172	**-1.22**
e2	30	15	4300	191342	191240	**-0.05**
e3	30	15	5225	184621	184136	**-0.26**
f1	30	30	3000	251364	244671	**-2.66**
f3	30	30	4400	218818	212939	**-2.69**
f4	30	30	5500	202280	199993	**-1.13**
g1	45	12	2700	311057	304590	**-2.08**
g2	45	12	4300	236623	235416	**-0.51**
g3	45	12	5300	215133	213154	**-0.92**
g5	45	12	6400	203777	204068	0.14
g6	45	12	8000	190572	190128	**-0.23**
h1	45	23	4000	241850	239730	**-0.88**
h2	45	23	5100	220532	215634	**-2.22**
h4	45	23	6100	208693	206046	**-1.27**
h5	45	23	7100	203647	199589	**-1.99**
i1	45	45	3000	336501	332321	**-1.24**
i2	45	45	4000	281864	283803	0.69
i3	45	45	5700	241102	245197	1.70
j1	75	19	4400	341272	337791	**-1.02**
j2	75	19	5600	301779	300712	**-0.35**
j3	75	19	8200	261264	265131	1.48
j4	75	19	6600	278866	284264	1.94
k1	75	38	4100	370134	369437	**-0.19**
k2	75	38	5200	327324	326044	**-0.39**
k4	75	38	6200	308065	302954	**-1.66**
l1	75	75	4000	418644	412875	**-1.38**
l2	75	75	5000	377962	374397	**-0.94**
l4	75	75	6000	339020	350558	3.40
m1	100	25	5200	381006	382016	0.26
m3	100	25	6200	354797	345619	**-2.59**
m4	100	25	8000	311410	312310	0.29
n1	100	50	5700	387926	392479	1.17
n3	100	50	6600	349555	366509	4.85
n5	100	50	8500	332754	336372	1.09
Average				263064	260906	**-0.22**

Table 8 Comparison of results for the MVRPB for the data set Goetschalckx and Jacobs-Blecha [23]

Problem	ACO		WS			H		
	Best	**Veh**	**Best**	**Veh**	**%Gap**	**Best**	**Veh**	**%Gap**
a1	223088	8	223088	8	0.00	227725	8	-2.04
a2	169500	5	169500	5	0.00	169497	5	0
a3	142034	3	142034	3	0.00	142032	3	0
b1	229403	7	233001	7	-1.54	233950	7	-1.94
b2	179194	4	179258	4	-0.04	182326	4	-1.72
b3	145702	3	145702	3	0.00	145699	3	0
c1	233087	7	239192	7	-2.55	242931	7	-4.05
c2	197278	5	196883	5	0.20	197276	5	0
c3	164891	3	164891	3	0.00	167663	4	-1.65
d1	307110	11	307110	11	0.00	307875	11	-0.25
d3	220751	7	224598	7	-1.71	222195	7	-0.65
d4	182496	5	-	-	-	-	-	-
e1	220742	7	223774	7	-1.35	222518	7	-0.80
e2	191135	4	190559	4	0.30	190048	4	0.57
e3	183197	4	182804	4	0.21	187793	4	-2.45
f1	243496	6	248333	7	-1.95	254977	6	-4.50
f3	210629	4	217317	5	-3.08	215575	5	-2.29
f4	198709	4	200964	4	-1.12	203448	5	-2.33
g1	298882	10	301235	10	-0.78	304106	10	-1.72
g2	234242	6	235920	6	-0.71	235220	6	-0.42
g3	212841	5	214534	5	-0.79	213757	5	-0.43
g5	202282	4	203233	4	-0.47	202610	4	-0.16
g6	188696	3	189922	3	-0.65	201875	4	-6.53
h1	237631	6	241619	6	-1.65	235269	6	1.00
h2	213756	5	220305	5	-2.97	215649	5	-0.88
h4	203441	4	208412	4	-2.39	202971	4	0.23
h5	197875	3	203193	4	-2.62	201896	4	-1.99
i1	326706	10	327168	10	-0.14	329237	10	-0.77
i2	279396	7	278727	7	0.24	289501	7	-3.49
i3	240974	5	238626	5	0.98	244782	5	-1.56
j1	331175	10	332471	10	-0.39	337800	10	-1.96
j2	296918	8	292698	8	1.44	298432	8	-0.51
j3	260619	6	259243	6	0.53	280070	7	-6.95
j4	280140	7	261066	7	7.31	257895	6	8.63
k1	365664	10	360954	10	1.30	361287	10	1.21
k2	322150	8	323979	8	-0.56	320012	8	0.67
k4	297570	7	298518	7	-0.32	296766	7	0.27
l1	405412	10	416167	11	-2.58	412278	11	-1.67
l2	371012	8	360018	8	3.05	362399	8	2.38
l4	343574	7	337620	7	1.76	341304	7	0.67
m1	377846	10	370920	10	1.87	372840	11	1.34
m3	343133	9	335486	9	2.28	336011	9	2.12
m4	307990	7	310567	7	-0.83	305118	7	0.94
n1	386780	10	370690	10	4.34	385978	10	0.21
n3	357552	9	349516	9	2.30	352992	9	1.29
n5	329462	7	323698	7	1.78	319811	7	3.02
Average	257743		259011		**-0.03**	260698		**-0.65**

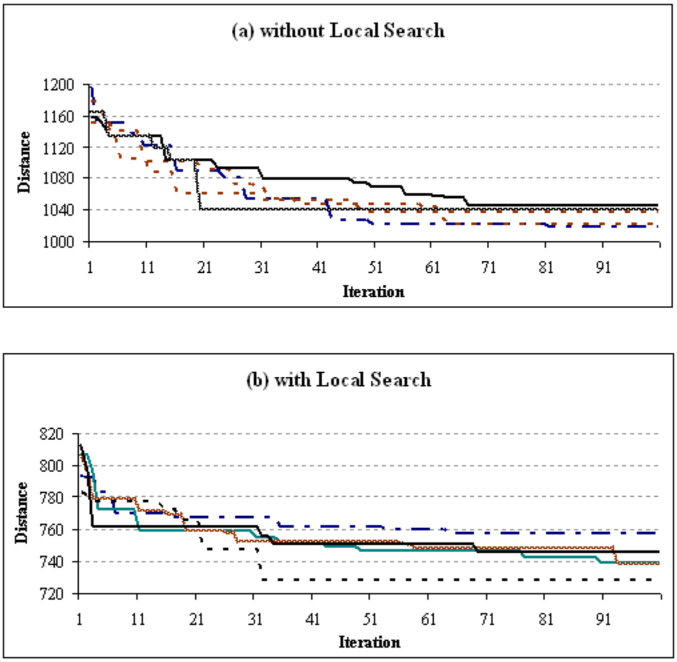

Fig. 3 Sample results of 5 runs for problem CMT3X (a) without local search, (b) with local search

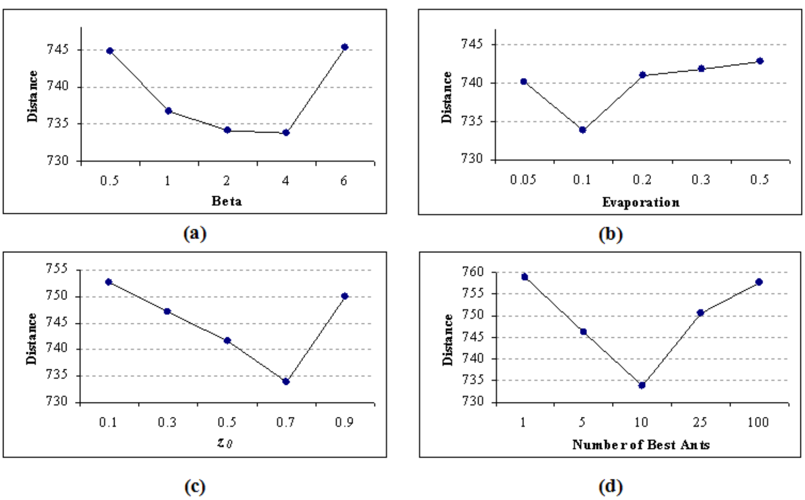

Fig. 4 Average distance of 10 runs of CMT3X for different parameter values of the (a) heuristic information parameter β, (b) evaporation rate ρ, (c) pseudo-random proportional rule parameter z_0, and (d) number of best ants used for the pheromone update w

z_0 and w, respectively, whereas the maximum deviations are 1.2% and 1.6% in the case of β and ρ. Note that these solutions are obtained by varying the value of one parameter only and keeping the remaining parameters at the already determined values. As expected, changing the values of multiple parameters simultaneously would lead to inferior solutions. These results reveal that a different selection of parameter values would not deteriorate the solution quality much due to the contribution of the local search. Nevertheless, the best distances are achieved by using the values we employed in our experimental study: $\beta = 4, \rho = 0.1, z_0 = 0.7$, and w = 100/10=10.

6 Conclusions

In this chapter, we addressed two types of VRPPD, namely MVRPB and VRPSPD which have a growing practical relevance in the reverse logistics literature. The computational complexity of these problems necessitates good heuristic solution procedures. We developed an ant colony algorithm equipped with a new visibility function in an attempt to capture the delivery and pickup nature of these two problems. The experimental analysis reveals promising results compared to the results published in the literature. Furthermore, improvements on some of the best-so-far solutions are obtained as well.

Although a fair comparison of the computational efforts cannot be made because of the use of different processors we observed that our computation times are fairly long compared to other heuristics presented in the literature. We observed that the local search procedure takes a significant amount of time. Thus, further research may focus on reducing the computation times, for instance by decreasing the ant colony size, using a candidate list, and/or by carrying out an elitist local search using only some best-performing ants. More efficient pheromone trail update procedures and visibility functions specific to the problem type may also be investigated to improve the solution quality.

References

1. Altınel, İ.K., Öncan, T.: A new enhancemant of the Clarke and Wright savings heuristic for the capacitated vehicle routing problem. Journal of the Operational Research Society 56, 954–961 (2005)
2. Angelelli, E., Mansini, R.: The vehicle routing problem with time windows and simultaneous pick-up and delivery. In: Klose, A., Speranza, M.G., Van Wassenhove, L.N. (eds.) Quantitative approaches to distribution logistics and supply chain management series. Lecture Notes in Economics and Mathematical Systems, vol. 519, pp. 249–267. Springer, Berlin (2002)
3. Bianchessi, N., Righini, G.: Heuristic algorithms for the vehicle routing problem with simultaneous pick-up and delivery. Computers and Operations Research 34, 578–594 (2007)
4. Brandão, J.: A new tabu search algorithm for the vehicle routing problem with backhauls. European Journal of Operational Research 173, 540–555 (2006)

5. Bullnheimer, B., Hartl, R.F., Strauss, C.: Applying the ant system to the vehicle routing problem. In: Voss, S., et al. (eds.) Meta-heuristics: Advances and trends in local search paradigms for optimization, pp. 285–296. Kluwer Academic Publishers, Boston (1998)
6. Bullnheimer, B., Hartl, R.F., Strauss, C.: An improved ant system algorithm for the vehicle routing problem. Annals of Operations Research 89, 319–328 (1999)
7. Casco, D.O., Golden, B.L., Wasil, E.A.: Vehicle routing with backhauls: models, algorithms, and case studies. In: Golden, B.L., Assad, A.A. (eds.) Vehicle routing: Methods and studies, pp. 127–147. Elsevier, Amsterdam (1988)
8. Chen, J.F., Wu, T.H.: Vehicle routing problem with simultaneous deliveries and pickups. Journal of the Operational Research Society 57, 579–587 (2006)
9. Christofides, N., Mingozzi, A., Toth, P.: The vehicle routing problem. In: Christofides, N., et al. (eds.) Combinatorial optimization, pp. 315–338. Wiley, Chichester (1979)
10. Clarke, G., Wright, W.: Scheduling of vehicles from a central depot to a number of delivery points. Operations Research 12, 568–581 (1964)
11. Colorni, A., Dorigo, M., Maniezzo, V., Trubian, M.: Ant system for job-shop scheduling. Belgian Journal of Operations Research, Statistics and Computer Science 34, 39–45 (1994)
12. Crispim, J., Brandão, J.: Metaheuristics applied to mixed and simultaneous extensions of vehicle routing problems with backhauls. Journal of the Operational Research Society 56, 1296–1302 (2005)
13. Dethloff, J.: Vehicle routing and reverse logistics: the vehicle routing problem with simultaneous delivery and pick-up. OR Spektrum 23, 79–96 (2001)
14. Doerner, K.F., Gronalt, M., Hartl, R.F., Reimann, M., Strauss, C., Stummer, M.: SavingsAnts for the vehicle routing problem. In: Cagnoni, S., Gottlieb, J., Hart, E., Middendorf, M., Raidl, G.R. (eds.) EvoWorkshops 2002. LNCS, vol. 2279, pp. 11–20. Springer, Heidelberg (2002)
15. Dorigo, M.: Optimization, learning and natural algorithms (in Italian), PhD Thesis, Dipartimento di Elettronica, Politecnico di Milano, Italy (1992)
16. Dorigo, M.: Ant colony optimization. Scholarpedia 2(3), 1461 (2008)
17. Dorigo, M., Gambardella, L.M.: Ant colony system: A cooperative learning approach to the traveling salesman problem. IEEE Transactions on Evolutionary Computation 1, 53–66 (1997)
18. Dorigo, M., Maniezzo, V., Colorni, A.: The ant system: Optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics-Part B 26, 29–41 (1996)
19. Dorigo, M., Stützle, T.: Ant colony optimization. MIT Press, Cambridge (2004)
20. Doyuran, T., Çatay, B.: Two enhanced savings functions for the Clark-Wright algorithm. In: Blecker, T., Kersten, W., Gertz, C. (eds.) Management in logistics networks and nodes: Concepts, technology and applications. Series on Operations and Technology Management, vol. 8, pp. 245–258. Erich Schmidt Verlag, Berlin (2008)
21. Gambardella, L.M., Dorigo, M.: Ant colony system hybridized with a new local search for the sequential ordering problem. INFORMS Journal on Computing 12, 237–255 (2000)
22. Gambardella, L.M., Taillard, E., Agazzi, G.: MACS-VRPTW: a multiple ant colony system for vehicle routing problems with time windows. In: Corne, D., Dorigo, M., Glover, F. (eds.) New ideas in optimization, pp. 63–76. McGraw-Hill, London (1999)
23. Goetschalckx, M., Jacobs-Blecha, C.: The vehicle routing problem with backhauls. European Journal of Operational Research 42, 39–51 (1989)
24. Golden, B., Baker, A.J., Schaffer, J.: The vehicle routing problem with backhauling: two approaches. In: Hammesfahr, R.D. (ed.) Proceedings of the 21st Annual Meeting of S.E., TIMS, Myrtle Beach, USA, pp. 90–92 (1985)

25. Halse, K.: Modeling and solving complex vehicle routing problems. PhD Thesis, Department of Mathematical Modeling, Technical University of Denmark (1992)
26. Min, H.: The multiple vehicle routing problem with simultaneous delivery and pick-up points. Transportation Research 23A, 377–386 (1989)
27. Montané, F.A.T., Galvão, R.D.: A tabu search algorithm for the vehicle routing problem with simultaneous pick-up and delivery service. Computers and Operations Research 33, 595–619 (2006)
28. Nagy, G., Salhi, S.: Heuristic algorithms for single and multiple depot vehicle routing problems with pickups and deliveries. European Journal of Operational Research 162, 126–141 (2005)
29. Paessens, H.: The savings algorithm for the vehicle routing problem. European Journal of Operational Research 34(3), 336–344 (1988)
30. Reimann, M., Doerner, K.F., Hartl, R.F.: Insertion based ants for vehicle routing problems with backhauls and time windows. In: Dorigo, M., Di Caro, G.A., Sampels, M. (eds.) Ant Algorithms 2002. LNCS, vol. 2463, pp. 135–148. Springer, Heidelberg (2002)
31. Reimann, M., Doerner, K., Hartl, R.F.: Analyzing a unified ant system for the VRP and some of its variants. In: Raidl, G.R., Cagnoni, S., Cardalda, J.J.R., Corne, D.W., Gottlieb, J., Guillot, A., Hart, E., Johnson, C.G., Marchiori, E., Meyer, J.-A., Middendorf, M. (eds.) EvoWorkshops 2003. LNCS, vol. 2611, pp. 300–310. Springer, Heidelberg (2003)
32. Ropke, S., Pisinger, D.: A unified heuristic for vehicle routing problems with backhauls. European Journal of Operational Research 171, 750–775 (2006)
33. Salhi, S., Nagy, G.: A cluster insertion heuristic for single and multiple depot vehicle routing problems with backhauling. Journal of the Operational Research Society 50, 1034–1042 (1999)
34. Stützle, T., Dorigo, M.: ACO algorithms for the quadratic assignment problem. In: Corne, D., Dorigo, M., Glover, F. (eds.) New ideas in optimization, pp. 33–50. McGraw-Hill, London (1999)
35. Stützle, T., Hoos, H.H.: The MAX-MIN ant system and local search for the traveling salesman problem. In: Bäck, T., Michalewicz, Z., Yao, X. (eds.) Proceedings of the 1997 IEEE International Conference on Evolutionary Computation (ICEC 1997), pp. 309–314. IEEE Press, Piscataway (1997)
36. Toth, P., Vigo, D.: An exact algorithm for the vehicle routing problem with backhauls. Transportation Science 31, 372–385 (1997)
37. Vural, A.V., Çatay, B., Eksioglu, B.: A dual GA approach to capacitated vehicle routing problem with simultaneous pick-up and deliveries. In: Proceedings of the IIE Research Conference (CD-ROM), Atlanta, GA (2005)
38. Wade, A., Salhi, S.: An ant system algorithm for the vehicle routing problems with backhauls. In: De Sousa, J.P. (ed.) Proceedings of the 4th Metaheuristics International Conference (MIC 2001), Porto, Portugal, pp. 99–203 (2001)
39. Wade, A., Salhi, S.: An ant system algorithm for the mixed vehicle routing problem with backhauls. In: Resende, M.G., de Sousa, J.P. (eds.) Metaheuristics: Computer decision-making, pp. 699–719. Kluwer, New York (2003)
40. Wassan, N.A., Wassan, A.H., Nagy, G.: A reactive tabu search algorithm for the vehicle routing problem with simultaneous pickups and deliveries. Journal of Combinatorial Optimization 15, 368–386 (2008)