Ant Colony Optimisation for vehicle routing problems: from theory to applications

A.E. Rizzoli[1], F. Oliverio[2], R. Montemanni[1], L.M. Gambardella[1,*]

[1] *Istituto Dalle Molle di Studi sull'Intelligenza Artificiale (IDSIA)*
Galleria 2, CH-6928 Manno, Switzerland
{andrea, roberto, luca}@idsia.ch
[2] *AntOptima, via Fusoni 4, CH-6900 Lugano, Switzerland*
fabrizio.oliverio@antoptima.ch

Abstract

Ant Colony Optimisation is a metaheuristic for combinatorial optimisation problems. In this paper we show its successful application to the Vehicle Routing Problem (VRP). First, we introduce VRP and its many variants, such as VRP with Time Windows, Time Dependent VRP, Dynamic VRP, VRP with Pickup and Delivery. These variants have been formulated in order to bring the VRP as close as possible to the kind of situations encountered in real-world distribution processes. Two case studies are presented: the application of Ant Colony Optimisation to the solution of the Time Dependent VRP, where the travel times depend on the time of the day, and Ant Colony Optimisation for Dynamic VRP, where customers' orders arrive during the delivery process. Finally, two real-world, industrial-scale applications are presented. The former is an application solving a VRP with Time Windows for a major supermarket chain in Switzerland; the latter is an application solving a VRP with Pickup and Delivery for a leading distribution

[*] Corresponding author. Tel: +41 91 610 8663. Fax: +41 91 610 8661. Email: luca@idsia.ch.

company in Italy. The results for these two real-world cases, in particular the increase in the vehicle routes performances and their potential use as strategic planning tools, are presented and discussed.

Keywords: Metaheuristics, Ant Colony Optimisation, Vehicle Routing Problem.

1 Introduction

A traditional business model is articulated in three stages: production, distribution, and sales. Each one of these activities is usually managed by a different company, or by a different branch of the same company. Research has been trying to integrate these activities since the 60s when multi-echelon inventory systems were first investigated [11], but, in the late 70s, the discipline which is now widely known as Supply Chain Management was not delivering what was expected, since the integration of data and management procedures was too hard to achieve, given the lack of real integration between the Enterprise Resource Planning (ERP) and the Enterprise Data Processing (EDP) systems [58]. Only in the early 1990s did ERP vendors start to deploy products able to exploit the pervasive expansion of EDP systems at all levels of the supply chain. The moment was ripe for a new breed of companies to put data to work and start to implement and commercialise advanced logistics systems, whose aim is to optimise the supply chain seen as a unique process from the start to the end (a software review can be found in [1]). Such systems were originally the preserve of big companies, who could afford the investment in research and development required to study their case and to customise the application to interact with the existing EDP systems. Moreover, the available optimisation algorithms required massive computational resources, especially for hard, combinatorial, problems such as Vehicle Routing. Finding the most cost efficient way to distribute goods across the logistic network is the keystone of supply-chain systems. Even a tiny improvement in the efficiency in the Vehicle Routing process is transformed in a sensible monetary gain, due to the fact that the distribution process is repeated every day of the year and gains are easily cumulated over time.

While these advanced systems were first deployed, researchers in the field of Operational Research were first investigating new *metaheuristics* [35], heuristic methods that can be applied to a wide class of problems, such as Ant Colony Optimisation – ACO [4], [21], [19]. It is now more than a decade since ants, industrious insects living in colonies, inspired operations researchers to design innovative algorithms to solve optimisation problems on graphs. Algorithms based on ACO draw their inspiration from the behaviour of real ants, which always find the shortest path between their nest and a food source, thanks to local message exchange via the deposition of pheromone trails. The first applications were a direct transposition of the observed ant behaviour and allowed to solve the Shortest Path Problem and the Travelling Salesman Problem [21]. Soon researchers discovered the considerable potential of ant-inspired algorithms and started to apply them to diverse problems, from scheduling [13] to water distribution [48]. The term Ant Colony Optimization was coined and a very successful metaheuristic was born. The advantage of ACO based algorithm over traditional optimisation algorithms is the ability to produce a good suboptimal solution in a very quick time, as it has been shown in experimental cases for the Travelling Salesman Problem [26], the VRP [30], and the Sequential Ordering Problem [27]).

In this paper we focus on the application of the Ant Colony Optimization (ACO) metaheuristic to the Vehicle Routing Problem (VRP) and some of its variants which are common in many real-world problems. The aim of the paper is to introduce the reader to ACO for VRP and to demonstrate its application in some case studies and real-world situations. First, case studies for the Time Dependent and the Dynamic case are presented; these are two variants of VRP that are currently attracting a lot of research efforts, thanks to their closeness to the real-world traffic and distribution models, where travel times are uncertain and not all distribution orders are known at planning time. Finally, two real-world applications are presented, to demonstrate how ACO can be successfully applied in the day-to-day operations of large distribution processes.

2 Vehicle Routing Problems

The Vehicle Routing Problem concerns the transport of items between depots and customers by means of a fleet of vehicles. The VRP can be instantiated to many real-world domains, examples are the milk float, mail delivery, school bus routing, solid waste collection, heating oil distribution, parcel pick-up and delivery, dial-a-ride systems, and many others. In general, solving a VRP means to find the best route to service all customers using a fleet of vehicles. The solution must ensure that all customers are served, respecting the operational constraints, such as vehicle capacity and the driver's maximum working time, and minimising the total transportation cost.

A VRP can be formulated as a mathematical programming problem, defined by an objective function, and a set of constraints. Exploiting the characteristics of the mathematical formulation of the problem, we want to design an algorithm able to efficiently find a solution.

2.1 Objectives, constraints, and solutions

Objectives measure the fitness of a solution. They can be multiple and often they are also conflicting. The most common objective is the minimisation of transportation costs as a function of the travelled distance or of the travel time; fixed costs associated with vehicles and drivers can be considered, and therefore the number of vehicles can also be minimised. Another objective can take into account vehicle efficiency, expressed as the percentage of load capacity (the higher, the better). The objective function can also be used to represent "soft" constraints, which are those constraints which can be violated paying a penalty. For instance, if a customer is not served according to the agreed time schedule a penalty is to be paid. Road pricing schemes can also be mirrored in the objective function, attributing a higher cost to routes through city centres.

The objective function contains both independent variables (decision variables), under the control of the planner, and dependent variables, which are a consequence of the assumed decisions. The solution of the problem is given by the decision variables returning the best evaluation of the objective function. In the VRP case, the decisions to be made define the order of the sequence of visits to the customers; they are a set of routes. A route departs from the depot and it is an ordered sequence of visits made by a vehicle to the customers, fulfilling their orders. A solution must be verified to be feasible, checking that it does not violate any constraint, such as the one stating that the sum of the demands of the visited vertices shall not exceed the vehicle capacity.

To find the values for the decision variables, we need a model of the vehicle routing system. Such a model is defined by the constrains that establish the relationships among independent and dependent variable and set limits of variables' values. The elements, which define and constrain the model, are: the road network, describing the connectivity among customers and depots; the vehicles, transporting goods between customers and depots on the road network; the customers, which place orders and receive goods.

2.1.1 The road network

The road network is represented as a graph, where depots and customers are placed on nodes and the edges represent the distance, in space and/or time, between two nodes. The road network graph can be obtained from a detailed map of the distribution area on which the depots and the customers must be geo-referenced. Standard algorithms can then be used to find all the shortest routes, with respect to time and distance, between all couple of nodes, in order to build the distance matrix. According to the adopted metric, different VRP instances may arise. For instance, if the travel time on edges depends on the time of the day (quite common in most highly congested cities) then we encounter the Time Dependent VRP.

2.1.2 The vehicles

The vehicles and their characteristics also impose constraints on the vehicle routing model. The fleet can be homogeneous, if all vehicles are equal in all their characteristics, otherwise it is said to be non-homogeneous. Most real-world fleets are non-homogeneous. Mechanical features (length, weight, width, number of axles) and configuration (trailer, semi-trailer, van, etc.) define the access constraints for a vehicle. For instance, a vehicle cannot travel on some arcs of a road network, because of excessive weight or dimensions. On-board equipment, such as loading/unloading devices, may also impose access constraints that depend on the type of customer to be served. For instance, a customer could be served only by trucks with an hydraulic lift. Capacity constraints, stating the maximum load to be transported by a vehicle, are also relative to the mechanical features of a vehicle. These are expressed in a unit of measure determined by the transported goods (e.g. litres for fluids, pallets for boxed goods, and also kilograms, cubic metres).

2.1.3 The customers

Each customer requests a given amount of goods (an order), which must be delivered or collected (picked-up) at the customer location. Time intervals during which the customer can be served (time windows) can be specified. These *time windows* can be single (only one continuous interval) or multiple (disjoint intervals, eg., delivery is possible only from 10 am to 11 am and from 3 pm to 4 pm). Time windows can be "hard", when a vehicle cannot arrive later than a given time, but it can wait if arriving early. In such a case, the objective function tries to minimise the distance and the waiting time. On the other hand, when a penalty is paid in case of violation, time windows are said to be "soft". Soft time windows can be incorporated in the objective function, by means of an appropriate cost function. Finally, the vehicle routing model can also include an estimation of the loading and unloading times at the customer (service time). These times depend on the customer facilities, on the ordered quantity, and on bureaucratic requirements. They are used to compute the delivery time, which is needed to compute the time at which the vehicle is ready to leave for the next customer in its tour.

2.1.4 Uncertainties in the model

The VRP formulation introduced so far made no assumptions on the role of uncertainties in the various elements of the problem. It must be remarked that the evaluation of the objective function depends on the computation of many quantities, which depend on data that are often uncertain and subject to a high degree of variability. In those cases, where some elements of problem are uncertain, we deal with the class of Stochastic VRPs. The uncertainty can be in the presence or absence of the customers, in the quantity of the their orders, and in the travel and service times [46].

Stochastic customers and demands are typical when the planning horizon is longer than the horizon of the currently available data. A company might be interested in planning routes a few months ahead, in order to anticipate future and potential demand. These pre-planned routes will be later adapted, when the actual demand will be known. These kind of problems have been extensively studied (see for instance [33]) and in Bianchi et al. [2] a study on the role of various metaheuristics for VRP with Stochastic Demands is presented.

Stochastic travel and service times are very frequent in urban environment. Especially with respect to travel times, the variability can be very high and considerably affect the solution. Later in this paper, we describe how this variability can be reduced if we assume that travel times are nearly constant within time periods in a day. This is quite true for peak and off-peak traffic conditions, which are observed in most cities. Such a characterisation of travel times leads to the Time Dependent VRP variant.

Finally, in some cases, the uncertainty is not in the real world, but in the model, and it can be considered as a measurement error. Cost can be a limiting factor in computing an accurate distance matrix, since geo-referencing a large customer database can become very expensive, especially for small companies. In such cases, an approximate model of the distance matrix can be elicited from the drivers' experience. The solution of the VRP problem will be obtained assuming that those values really represent the travel times among customer visits, and therefore it might over or under-estimate the time required to actually perform a route. While travel times on standard routes, which are travelled over and over by the same drivers, tend to be 'learnt' by the driver over time, if the set of customers is very varied and keeps changing, rough estimates of travel times can be a problem. Then, it becomes important to assess the robustness of the solution in front of variations of the travel times.

2.2 Basic problems of the vehicle routing class

Combining the various elements of the problem, we can define a whole family of different VRPs. Vigo and Toth [64] present a detailed overview of the various VRPs; here, we limit ourselves to describe the problems that have been solved by means of Ant Colony Optimisation, as described in the reminder of the paper. Thus, we briefly introduce the Capacitated Vehicle Routing Problem (CVRP), the VRP with Time Windows (VRPTW) and its Time Dependent variant (TDVRPTW), the VRP with Pickup and Delivery (VRPPD), and the Dynamic VRP (DVRP).

2.2.1 The Capacitated VRP

The Capacitated Vehicle Routing Problem (CVRP) is the basic version of the VRP. The name derives from the constraint of having vehicles with limited capacity. By removing this constraint, and imposing that all customers must be served by a single route, the CVRP collapses into a standard Travelling Salesman Problem; it can therefore be shown that CVRP is NP-hard. In the classic version of the CVRP, customer demands are deterministic and known in advance. Deliveries cannot be split, that is, an order cannot be served using two or more vehicles. The vehicle fleet is homogeneous and there is only one depot. The objective is to minimise total travel cost, usually expressed as the travelled distance required to serve all customers.

The problem can be formulated as a graph theoretic problem, where $G = (V, A)$ is a complete graph, V the vertex set (customers, and the depot, usually labelled with 0) and A is the arc set (the paths connecting all customers and the depot). A non-negative demand q_i is associated with each vertex, and a cost c_{ij} is associated with each edge in A. If the cost matrix associated with the graph, representing the distance or travel time, is asymmetric (a common situation in urban contexts), then the problem is called the asymmetric CVRP.

A detailed formulation of the problem and methods to solve it exactly are described by Vigo and Toth [63]. These methods include branch-and-bound, branch-and-cut, and set-partitioning algorithms. The size of the problems which can be solved exactly is up to 100 customers, using the branch-and-bound approach (see Fisher [23]). Bigger problems and most real-world problems can be solved using heuristic and metaheuristic methods, which provide only a suboptimal solution.

2.2.2 VRP with Time Windows

In a Vehicle Routing Problem with Time Windows (VRPTW) the capacity constraint still holds and each customer i is associated with a time interval $[a_i, b_i]$, called the time window, and with a time duration, s_i, the service time. This problem is often common in real world applications, since the assumption of complete availability over time of the customers made in CVRP is often unrealistic. Time windows can be set to any width, from days to minutes, but their width is often empirically bound to the width of the planning horizon. In other words, if we plan the distribution for the next five days, that is the planning horizon is 5 days long, and we set the time windows' width to be in the order of few minutes, it will be much harder to find a feasible solution than in the case the time windows are a few hours wide.

Note that the presence of time windows imposes a series of precedences on visits, which make the problem asymmetric, even if the distance and time matrices were originally symmetric. VRPTW is also NP-hard and even to find a feasible solution to VRPTW is an NP-hard problem [57]. The additional constraints in VRPTW call for more articulate variants of the basic methods used to obtain an exact solution for CVRP and therefore the performances tend to worsen. A good overview on the VRPTW formulation and on exact, heuristic, and metaheuristic approaches can be found in Cordeau et al. [14].

2.2.3 VRP with Pick-up and Delivery

In VRP with pick-up and delivery (VRPPD) a vehicle fleet must satisfy a set of transportation requests. This time the transport items are not originally concentrated in the depots, but they are distributed over the nodes of the road network. A transportation request consists in transferring the demand from the pick-up point to the delivery point. In the case the demand is a transport of persons the problem is commonly called "dial-a-ride". These problems always include time windows for pick-up and/or delivery and also constraints that express the user inconvenience of waiting too long at the pick-up point and impose a limit on riding time. When the demand is a transport of goods, sometimes the problem can be simplified, according to the characteristics of the transport process. For instance, in courier services, all delivered goods leave from the depot and all pick-ups return to the depot. Moreover, it can be safely assumed, in many circumstances, that all deliveries can be performed before the pick-ups, thus reducing the impact of capacity constraints. A review of various approaches to the solution of VRPPD is presented in [17].

2.2.4 Time Dependent VRP

An interesting extension of the VRPTW in urban environments is the Time Dependent VRPTW, where the arc costs on the graph depend on time. This situation is quite common in most cities, since the time taken to travel from a location to another one depends on the traffic load, which varies with the time of the day. Particular care must be taken in defining the time dependency of the cost function. If the horizon of interest is discretised into small intervals, and the travel times vary in discrete jumps from an interval to the next, then Ichoua et al. [41] note how this approach, although being quite used, does produce solutions which may go against common-sense. This happens when the FIFO property is violated, that is, a vehicle departing later may arrive earlier than an earlier departing vehicle, even following the same route. Therefore formulations where travel time and cost functions vary continuously are to be preferred. In the cited work by Ichoua *et al.*, an algorithm solving such a problem is presented. The algorithm was tested on Solomon's 100-customer Euclidean problems [59]. The travel times were calculated adjusting the travel speed in order to account for the time period (morning rush, middle of the day, evening rush). The solutions obtained in the time dependent case are compared with the ones computed assuming that travel times are static, and the results show that accounting for time dependency pays back, providing better quality routes.

2.2.5 Dynamic VRP

Another extension to standard VRP, which is also common in many real-world applications, is when the service requests are not completely known before the start of service, but they arrive during the distribution process. This variant is called Dynamic Vehicle Routing Problem (DVRP). Since new orders arrive dynamically, the routes have to be replanned at run time in order to include them. Let us assume that a communication system between the dispatcher (where the tours are calculated) and the drivers exists. The dispatcher can periodically inform the drivers about the next visit(s) assigned to them (*commitment*). According to this model of information transfer, every driver has, at each time step, a partial knowledge about the remainder of his/her tour.

Among possible applications of DVRP we find *feeder systems*, which typically are local dial-a-ride systems aimed at feeding another, wider area, transportation system at a particular transfer location (Gendreau and Potvin [34]). Another application is to courier service problems (e.g. Federal Express), where parcels are collected at customer locations and brought back to a central depot for further processing and shipping.

The DVRP has been treated in Kilby et al. [42], Gendreau et al. [31] and Ichoua et al. [40]. In [40] the algorithm described in [31] is integrated with a vehicle diversion mechanism: in practice, it is possible to divert a vehicle away from its current destination in response to a new customer request. A survey on results achieved on the different types of *DVRP*s can be found in Gendreau and Potvin [34] (see also Psaraftis [54] and [55]).

3 Solving the VRP with metaheuristics

Since all the VRP variants are NP-hard, their combinatorial complexity makes them intractable as soon as the search space becomes too large, and in vehicle routing this happens in practice when there are a few dozens of customers to serve. Thus, heuristic and metaheuristic methods are the only feasible way to provide solutions for industrial scale problems. The integration of optimisation algorithms based on innovative metaheuristics, such as Ant Colony Optimisation, Tabu Search [36], Iterated Local Search [60], Simulated Annealing [45], with advanced logistic systems for Supply Chain Management opens new perspectives for OR applications in industry. Not only big companies can afford these softwares, but also small and medium enterprises can use state-of-the-art algorithms, which run quickly enough to be adopted for online decision making.

Among heuristic methods we find *constructive* methods, such as the seminal Clarke and Wright Savings algorithm [12], which finds an initial feasible solution to the problem trying to cluster customers, and *improvement* methods, which start from an existing solution and try to improve it (see for instance the work by Kinderwater and Savelsbergh [44]). As noted by Laporte and Semet [47], heuristics perform a limited exploration of the search space, in a relatively short computation time, while metaheuristics follow the principles of *intensification* and *diversification* in their search. Intensification pushes towards a much more detailed exploration, in the most promising areas of the search space, while diversification aims to avoid being locked up in local minima, but at the expense of computational speed.

Despite the common principles of intensification and diversification, metaheuristics can be profoundly different and also their applicability to a given problem can produce varied results (a comparison among various metaheuristics can be found in Blum and Roli [3]). Moreover, it was also remarked by Martin, Otto, and Felten [49] that metaheuristics tend to perform very well when *hybridized* with local search methods, combining specific problem knowledge in the improvement of the solutions. ACO is particularly apt for hybridization, since it is rarely able to build a solution that is good enough, but on the other hand it produces good candidate solutions which can be further improved using various local search techniques, as shown by Gambardella and Dorigo for the Sequential Ordering Problem in [27].

Various metaheuristics have been applied to the VRP and its variants: Simulated Annealing [52], Tabu Search [32], [62], Granular Tabu Search [65], Genetic Algorithms [6], Guided Local Search [43], Variable Neighbourhood Search [5], and Ant Colony Optimisation [7], [30]. In the next paragraphs, we focus our attention on Ant Colony Optimisation and then we present its application to CVRP, VRPTW, TDVRP and DVRP.

3.1 The ACO metaheuristic

Ant Colony Optimisation [19] has been inspired by the foraging behaviour of real ants. Ants randomly explore the surroundings of the anthill; when they find food, they return to the nest depositing a *pheromone trail*, a trace of a chemical substance that can be smelled by other ants. Ants can follow various paths to the food source and back, but it has been observed [16] that, thanks to the reinforcement of the pheromone trail by successive passages, only the shortest path remains in use, since ants prefer to follow stronger pheromone concentrations. Pheromone reinforcement is autocatalytic, since the shortest the path, the least time will be taken to travel back and forth, and therefore, while ants on longer paths are still in transit, the ants on the shortest path can restart the route again, reinforcing the pheromone trail on the shortest path. Over time, the majority of the ants will travel on that path, while a minority will still choose alternative paths. The behaviour of this minority is important, since it allows to explore the environment to find even better solutions, which initially were not considered. The choice of the path is therefore probabilistic and, while it is strongly influenced by the pheromone intensity, it still allows for random deviations from the current best solution.

The ACO algorithm replicates this behaviour, adding some features to make it more efficient in the computer implementation. Ants are implemented as a set of concurrent and asynchronous agents. They construct a solution visiting a series of nodes on a graph. They select the move along an edge to the next node to visit according to two parameters: trails and attractiveness. As real ants, also artificial ants will prefer in most cases a deterministic choice of the path, based on the selection of the path with the strongest pheromone and on the highest attractiveness. Yet, in a fraction of cases, the choice will be made probabilistically, though guided by attractiveness and trails.

The attractiveness η_{ij} of a move from node i to j is computed according to an heuristic that expresses the *a priori* desirability of the move. In a shortest path problem, the desirability can be the inverse of the distance; in other VRP variants, the desirability can also depend on other parameters, beside the distance, for instance in VRPTW it also depends on the current time and the time window limits of the customers to be visited.

The trail level τ_{ij} of a move depends on the pheromone level, and it represents a dynamic indication *a posteriori* of its goodness. In other words, if the artificial ant smells a strong pheromone trail leading to a node, it knows that it is a promising direction to explore. When the constructive procedure has finished and artificial ants have computed a set of solutions, the pheromone information associated to some of the edges i–j are updated according to the following equation:

$$\tau_{ij} = (1-\rho)\cdot\tau_{ij} + \Delta\tau_{ij} \tag{1}$$

where both the set of updated edges i–j and the $\Delta\tau_{ij}$ values depend on the specific ACO implementation. Pheromone also *evaporates* in order to avoid locking into local minima. Trail evaporation reduces pheromone over all trails iteration by iteration, usually by exponential decay.

There are several ACO metaheuristic implementations that differ in the way the artificial pheromone is used and updated. The first prototype implementation is Ant System [20]: when an artificial ant is at node i, the next node j is selected probabilistically according to a *random-proportional* rule:

$$p_{ij} = \begin{cases} \frac{[\tau_{ij}]^\alpha[\eta_{ij}]^\beta}{\sum_{h\in\Omega}[\tau_{ih}]^\alpha[\eta_{ih}]^\beta} & \text{if } j\in\Omega \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

Where p_{ij} is the probability of moving to j from i, and Ω is the set of nodes which are feasible to be visited from i. The parameters α and β weight the influence of trails and visibility.

Ants construct their solutions in parallel. At the end of each constructive phase (iteration) the entire set of computed solutions is used to update the pheromone trail. In this case $\Delta\tau_{ij}$ of equation (1) is computed according to the following formula:

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

where m is the number of ants and $\Delta\tau_{ij}^k$ is the amount of trail laid on edge i–j by ant k, which can be computed as $\tau_{ij}^k = Q/J_k$ if ant k uses edge i–j in its tour, and Q is a constant.

Ant System was initially applied to the solution of the Travelling Salesman Problem but was not able to compete against the state-of-the art algorithms in the field. To improve Ant System, Gambardella and Dorigo proposed in 1995 Ant-Q algorithm [25], and in 1996 Ant Colony System (ACS) [26],[20] a simplified version of Ant-Q that maintained approximately the same level of performance, measured by algorithm complexity and by computational results. ACS introduced the concepts of *local* and *global update*, and of the *pseudo-random-proportional* transition rule, which are at the basis of any successful ACO implementation.

In ACS, at the end of each iteration, global update uses only the best solution, computed so far, to update the pheromone trail. The only edges that are modified are those edges i^*–j^* belonging to the best solution ψ^*. The objective function evaluates the best solution ψ^* to the value J^* . This strategy reduces the time to convergence by directly concentrating the search in a neighborhood of the best solution. The formula for global update is:

$$\tau_{i^*j^*} = (1-\rho) \cdot \tau_{i^*j^*} + \rho \cdot \frac{1}{J^*} \quad \forall (i^*, j^*) \in \psi^* \tag{4}$$

Local update is performed much more frequently: every time an ant decides to use edge i–j in its solution, the pheromone τ_{ij} is modified:

$$\tau_{ij} = (1-\rho) \cdot \tau_{ij} + \rho \cdot \tau_0 \tag{5}$$

where τ_0 is the initial pheromone value defined as $\tau_0 = 1/(n \cdot J_{nn})$, with n, the number of customers in the solution, and J_{nn}, the tour length produced by the execution of one

ACS iteration without the pheromone component (this is equivalent to a probabilistic nearest neighbor heuristic).

During the construction of a new solution the *pseudo-random-proportional* transition rule is introduced. The pseudo-random-proportional rule is a compromise between the pseudo-random state choice rule typically used in Q-learning [66] and the random-proportional action choice rule typically used in Ant System, described in equation (2) . With the pseudo-random rule, if q is a random variable uniformly distributed over $[0, 1]$, with probability q_0 (*exploitation*) the next chosen node is the one with the best $\tau_{ij} \cdot \eta_{ij}$, while with probability $1 - q_0$ (*exploration*) the node is chosen using the Ant System random-proportional choice rule. The pseudo-random rule can be therefore expressed as:

$$\begin{cases} \arg\max \tau_{ij} \cdot \eta_{ij} & \text{if } q \leq q_0 \\ \text{equation (2)} & \text{otherwise} \end{cases} \tag{6}$$

where q_0 is a parameter $0 \leq q_0 \leq 1$ usually equal to 0.9. The pseudo-random rule provides a straightforward way to balance between exploration of new states and exploitation of a priori and accumulated knowledge.

ACS has been proved to be very efficient in solving many graph routing problems. Like in the ACO framework also ACS uses *daemon actions* to perform global actions, which are not possible for artificial ants, which can act only locally. A typical daemon action is the launch of a local search, to improve the current best solution found so far by the ACO.

Different algorithms implement different versions of ACO and the kind of VRP variant has also a considerable impact on the ACO implementation. In the next sections we outline the main features of ACO algorithms for the VRP variants we have encountered so far.

3.2 Capacitated VRP

CVRP is the most basic variant of VRP and it was the first to attract the attention of researchers trying to apply the ACO metaheuristics to VRP. One of the first approaches is due to Bullnheimer et al. [7], [8], where the objective function has the unique objective of minimising the total route length.

To solve the problem, artificial ants construct the solution by successively choosing customers to visit, until each customer has been served. If selecting a customer leads to the violation of a constraint on the capacity or on the total route length, the route returns to the depot. When an artificial ant is at customer i, the next customer j is selected according to a random-proportional rule (see equation (2)), in which visibility is defined using a function [53] inspired by the savings algorithm:

$$\eta_{ij} = d_{i0} + d_{0j} - g \cdot d_{ij} + f \cdot |d_{i0} - d_{0j}| \tag{7}$$

where g and f are two parameters (good settings are $g = f = 2$).

To reduce the time taken to look for the next customer to visit, not all customers are evaluated, but only those appearing in a *candidate list*. A typical way to build a candidate list is to sort customers by their distance from the current node examine only the first customers in the list.

Bullnheimer et al. implement a variant of the Ant System. The pheromone trail is updated whenever an artificial ant k computes a solution ψ^k, which is evaluated in the objective function as J_k. Equation (1) is modified in order to take into account the contribution of the σ ants computing the current best solution (ψ^* that evaluates to J^*) tha are called *elitist* ants. They contribute in a special way to the pheromone trail update, increasing each arc in the optimal solution by $\Delta\tau^*_{ij} = 1/J^*$. Given this, the updating rule is:

$$\tau_{ij} = (1-\rho)\cdot\tau_{ij} + \sum_{k=1}^{m-1}\Delta\tau^k_{ij} + \sigma\Delta\tau^*_{ij} \tag{8}$$

It was found that the number of ants m should be equal to the number of customers and that one ant is placed at each customer at the beginning of an iteration. Bullnheimer et al. also show that using the 2-opt-heuristic for the TSP [15] considerably improves the solution quality.

Following the steps of the works by Bullnheimer, Reimann et al. [56] have also implemented an ACO algorithm for CVRP, sometimes improving the original results.

3.3 VRPTW

The most efficient ACO algorithm for the VRPTW and one of the most efficient metaheuristics overall for this problem is MACS-VRPTW by Gambardella, Taillard, and Agazzi [30]. The problem is more complex that CVRP under two viewpoints: first, there are time windows on deliveries; second, the objective function is more complex, since it considers two objectives: the minimisation of the number of tours (or vehicles) and the minimisation of the total travel time. The objective function is hierarchic: tour minimisation has the precedence over time minimisation.

The central idea of the MACS-VRPTW algorithm is to use two colonies (MACS stands for Multi Ant Colony System), one colony, named ACS-VEI, minimises the vehicles and the other one, named ACS-TIME, the time. The two colonies are completely independent, since each one has its own pheromone trail, but they collaborate by sharing the variable ψ^*, which describes the best current solution. Each colony is composed by a number of ants. Every ant in the colony tries to build a feasible solution to the problem.

The algorithm is as follows: first compute a feasible initial solution ψ^*, with a number of vehicles v, using a nearest neighbour heuristic. ACS-VEI is then started: it tries to improve ψ^* using one vehicle less. ACS-TIME is also started: given v vehicles, it tries to minimise the total time required to serve all customers. When ACS-VEI finds a feasible solution with one vehicle less, ψ^* is updated and ACS-TIME is restarted with $v = v - 1$. ACS-TIME serves the purpose of refining the solutions obtained by ACS-VEI, which has no feeling for the travel time, since its objective function is independent of it.

The constructive step of both colonies is based on a procedure that, starting from the current node i, computes the set of all feasible nodes. These are the j nodes still to be visited and such that the time of arrival at node j and the load are compatible with the time window and the delivery of quantity q_j. The probabilistic choice is made according to the pseudo-random rule of equation (6). Visibility depends on a modified distance, which accounts for the effect of time windows: a node can appear 'closer' if the end of its time windows is near. The end of the construction of a solution by all ants in the colony marks the end of one algorithm iteration.

The trail pheromone is updated both *locally* and *globally*. Local updating is performed within the constructive procedure. Each ant updates the pheromone value on the arc that has just visited during the construction of the solution. The updating rule is given in equation (5). Global updating is performed after a solution has been completed, at the end of the construction phase, using the rule of equation (4).

Note that in the ACS-VEI colony, at the end of each cycle, pheromone trails are globally updated for two different solutions: $\psi^{ACS-VEI}$, the unfeasible solution with the highest number of visited customers, and ψ^*, the feasible solution with the lowest number of vehicles and the shortest travel time. This allows for updating pheromone also on arcs included in a feasible solution, guiding the search of a solution with less vehicles, but which is still possible to be made feasible.

ACS-TIME uses a local search procedure to improve the quality of the feasible solutions, which is similar to the CROSS procedure [62]. Both ACS-TIME and ACS-VEI use a local search that tries to repair unfeasible solution by inserting unvisited customers.

3.4 Time Dependent VRPTW

Donati et al. [18] introduce an ACO algorithm for the Time Dependent VRPTW. The basic idea of this algorithm is to define a pheromone trail that is time dependent. We remember from Section 2.2.4 that the time dependent model assumed that the travel times over the arcs of the graph depend on the time of the day. While the variation of travel times over time is continuous, it can be assumed that there are some distinctive time slices during one day when they are roughly constant, such as the peak and off-peak hours. The authors assume that the duration of one working day can be partitioned in l time slices, and therefore the pheromone trails can be described by $\tau_{ij}(l)$, with $l \in T_l$, where T_l is the working time horizon. The objective is to minimise the total travel time.

The algorithm then builds a solution making a probabilistic choice to select the next node j starting from i using the standard equation (2). The desirability η_{ij} of the next node is given by:

$$\eta_{ij}(t) = \frac{1}{f_{ij}(t) + w_j} \tag{9}$$

where $f_{ij}(t)$ is the travel time from i to j evaluated at time $t \in T_l$ and w_j is the waiting time at node j.

Pheromone updating is done according to:

$$\tau_{ij}(l) = (1 - \rho) \cdot \tau_{ij}(l) + \sum_{k=1}^{m-1} \Delta\tau_{ij}^k(l) \tag{10}$$

which is an adaptation of equation (8) where also the time l when the arc $i-j$ was travelled is taken into account.

3.5 Dynamic VRP

In Dynamic VRP new orders can be assigned to vehicles which have already left the depot (e.g. parcel collection, feeder systems, fuel distribution, etc.).

Montemanni et al. [51] have developed an ACO-inspired algorithm, ACS-DVRP, based on the decomposition of the DVRP into a sequence of static VRPs. There are three main elements in the architecture they propose: the event manager, the ant colony algorithm, the pheromone conservation strategy.

The event manager receives new orders and keeps track of the already served orders, of the position and residual capacity of each vehicle. This information is used to construct the sequence of static VRP-like instances. The working day is divided into time slices and for each of them a static VRP, which considers all the already received (but not yet executed) orders, is created. New orders received during a time slice are postponed until its end. At the end of each time slice, customers whose service time starts in the next time slice (according to the solution of the last static VRP) are assigned to the vehicles. They will not be taken into account in the following static VRPs.

The ACS algorithm employed in this implementation of DVRP is based on the one described in Section 3.2 for the CVRP. The single ant colony is in charge of minimizing the total travel time.

Finally, the pheromone conservation strategy is based on the fact that once a time slice is over and the relative static problem has been solved, the pheromone matrix contains information about good solutions to these problems. As each static problem is potentially very similar to the next one, this information is passed on to the next problem. This is performed efficiently, following a strategy inspired by Guntsch and Middendorf [38].

This algorithm has been put to the test on a set of benchmarks and also in a case study. We report on this in the next Sections 4.1 and 4.2.2.

4 Computational results

4.1 Benchmarks

Benchmark problems allow to evaluate and compare the performance of different VRP algorithms. These problems are experimental set-ups, which define all the data required to formulate a vehicle routing problem. Every problem variant has its own benchmark. For instance, the first set of 56 benchmark problems for the VRPTW was designed by Solomon in 1983 [59], for 25, 75, and 100 customers, and more recently Homberger [39] extended the Solomon's instances to bigger problems (200, 400, 600, 800, and 1000 customers) to put to the test the more advanced metaheuristic-based algorithms[1]. These problems hypothesise a spatial distribution of the customers (clustered, uniformly distributed, or a mix of the two) and different constraint values (narrow time windows and small vehicle capacities and wide time windows and large capacities).

It can be observed that no significative improvements in the algorithm performances took place in the past 5 years,. The majority of the best solutions found by heuristic methods for the Solomon's instances [59] were published before 2000. Recently, a paper by Bräysy [5] and a work by Mester and Bräysy [50] improved many of the known benchmarks, but the improvements for the Solomon's instances were limited, as shown in Table 8 of [5]. A possible explanation of this slowdown in the improvement of results is that no other major breakthrough has been achieved after the hybridization of metaheuristics with Local Search techniques.

4.1.1 The benchmarked performance of ACO for CVRP and VRPTW

ACO inspired algorithm for CVRP and VRPTW, originally presented towards the end of the 1990s, still perform quite well, also when compared with the most recent algorithms. Bullnhemier et al. [7] tested their ant system for CVRP on fourteen benchmark problems described in [10] and they show how the hybridization with local search leads to sensibly better results. Bullnheimer et al. also highlight how Tabu Search was better for the CVRP, but the potential for growth of ACO was high. This potential was exploited by Gambardella et al. [30] with the MACS-VRPTW algorithm. This algorithms has improved many instances

[1] A summary of the solutions found for these problems is available at http://www.sintef.no/static/am/opti/projects/top/vrp/benchmarks.html

of the Solomon's problems and, it is still among the best. Gambardella et al. tested their algorithm on the classical set of 56 Solomon's problems and MACS-VRPTW was shown to be the best method for C1 and RC2 types and it was always among the best for other problem sets. Moreover, MACS-VRPTW was able to find the solutions in a very short time, when compared to other algorithms, and it was also run on the CVRP problem producing excellent results [24].

4.1.2 A new benchmark for DVRP

Given the relatively slow advancement in more traditional VRP variants, it becomes more and more interesting to concentrate our attention on novel problems, which present new challenges, such as the VRPTD and the DVRP variants. For these problems new benchmarks are required. In particular, in Montemanni et al. [51] a benchmark for the Dynamic VRP is described. The authors adapted a set of problems originally proposed by Kilby et al. [42] (the problems are available in [37]), which, in turn, are derived from some very popular static VRP benchmark datasets, namely 12 problems are taken from Taillard [61], 7 problems are from Christofides and Beasley [9] and 2 problems are from Fisher et al. [22]. These problems range from 50 to 199 customers.

Kilby et al. added to these problem the concept of *length of the working day*, and to each customer an *appearance time*, namely the time when the order becomes available (during the working day) and a *duration*, namely the time required to perform an order once reached the customer. They also fixed the number of available vehicles to 50 for each problem. Montemanni *et al.* added to these problems two parameters: the *advanced commitment time*, T_{ac}, and the *cutoff time* T_{co}. In DVRP, orders arrive dynamically during the day. The cutoff time defines the time within which all received orders will be processed durring the same day. The advanced commitment time represents how long in advance the order must be committed to drivers. For example, if an order has to be executed at 3 pm and T_{ac}=10 minutes, then the order must be communicated to the driver no later than 2:50 pm.

The set of benchmarks is defined by setting the time after which the orders are postponed to the day after, T_{co}, to $T \times 0.5$, while the advance commitment time, T_{ac}, is fixed to $T \times 0.01$. A set of widely available set of problems on which to compare the algorithms of different authors has therefore been proposed.

Montemanni et al. investigated how the results of ACS-DVRP are connected with pheromone conservation; then they compared ACS-DVRP with a method based on a *Multi-Start Local Search* algorithm (MSLS). In order to carry out the tests, the working day of each problem was compressed into 1500 seconds of CPU time of an Intel Pentium 4 1.5 GHz processor. Five runs for each of the 21 problems were considered.

It was shown that the average results of ACS-DVRP are 4.37% better then those of MSLS. The best (worst) results are 4.86% (2.70%) better in average. The best solutions were always found by ACS-DVRP.

4.2 Case and feasibility studies

We briefly introduce two case studies, in which ACO-based algorithms have been applied on real world data. These case studies demonstrate the applicability of the proposed algorithms.

4.2.1 Time Dependent VRPTW in the city of Padua

The city of Padua, Italy, is interested in setting up a logistic platform to collect all incoming goods to be distributed to a number of shops in the city centre, which is affected by traffic congestion problems and also loading and unloading space is scarcely available. A better organisation of the flow of goods into the city centre, served by a centralised planning of vehicle routes can sensibly reduce pollution and traffic problems due to commercial transport. For this purpose, Donati et al. [18] have studied an application of the TDVRPTW algorithm presented in Section 3.4 to the city of Padua. In the test study, the central depot was open from 8 am till 6 pm and the traffic data during that period on Padua road network was collected. Four time intervals, with similar traffic patterns and the relative travel speeds on network arcs, were identified. A set of 30 customers was considered.

The authors compared the solution of the VRP using the time dependent variant with a solution of the same problem where the travel times on the road arcs were constant, depending only on the distance. In a series of nine tests, where customers were picked randomly out of a set of real customers, it turned out that the TD variant performs 7% better than the standard VRPTW algorithm.

4.2.2 Dynamic VRP for fuel distribution

Montemanni et al. [51] tested the Dynamic VRP algorithm, described in section 3.5, on a realistic case study, set up in the city of Lugano, Canton Ticino, Switzerland. The road network and the customer data were provided by a leading Swiss fuel oil distribution company, which serves its customers from its main depot located near Lugano. During the peak season, in Winter, when customers tend to run out of fuel and they place urgent orders, the delivery pattern of the vehicles becomes very 'dynamic', in the sense that a noticeable percentage of orders must be fulfilled after the vehicles have already left the depot. From the customers data base, 50 customers have been randomly chosen, and travel times among them have been calculated. A working day of 8 hours (28'800 seconds) is considered, while a service time of 10 minutes (600 seconds) is set up for each customer. Customers randomly appear during the working day with random requests for a quantity of fuel to be delivered. The vehicle fleet was composed by 10 vehicles, the cut-off time was set to $T_{co} = T \times 0.5$ (i.e. $T_{co} = 14'400$ seconds) and the advanced commitment time $T_{ac} = T \times 0.01$ (i.e. $T_{ac} = 288$ seconds).

In Table 1 we presents the results obtained by the ACS-DVRP algorithm on different experiments, where the number of time slots (namely parameter n_{ts}) was varied during the experiments. Also the time allocated to executing the Ant Colony System (t_{acs}) and the time dedicated to the local search improving the solution (t_{ls}) were adjusted according to the values of n_{ts}. In particular the ratio between t_{acs} and t_{ls} is, in this case, always around 10.

In Table 1 the first three rows define the setting of the experiments, i.e. the values of parameters n_{ts}, t_{acs} and t_{ls}. The fourth row shows the total travel time of the solutions found by the ACS-DVRP algorithm.

Table 1: Experimental results on the case study of Lugano.

n_{ts}	200	100	50	25	10	5
t_{acs}	144	288	576	1152	2880	5760
t_{ls}	15	30	60	120	240	480
Travel time	12702	12422	10399	9744	10733	11201

Table 1 suggests that, for the case study analyzed, good values for n_{ts} are between 10 or 50. In particular, 25 seems to be the best choice. Large values of n_{ts} did not lead to satisfactory results because optimisation was restarted too often, before a good local minima could be reached. On the other hand, when n_{ts} was too small, the system was not able to take advantage of information on new incoming orders.

5 Applications

Sales and distribution processes require the ability to forecast customer demand and to optimally plan the fine distribution of the products to the consumers. These two strategic activities, forecast and optimisation, must be tightly interconnected in order to improve the performance of the system as a whole [29].

Insert Figure 1 about here

In Figure 1 the workflow process of a distribution-centred company is sketched. The sales department generates new orders by contacting the customers (old and new ones) to check whether they need a new delivery. The effectiveness of this operation can be increased thanks to inventory management modules, which estimates the demand of every customer, indicating the best re-order time for each of them. New orders are then processed by the planning department, which, according to the quantity requested, the location of the customers, the time windows for the delivery, decides how many vehicles to employ and computes the best routes for the delivery, in order to minimise the total travel time and space. This task is assisted by a vehicle routing algorithm, represented by the OPTIMISE block. The vehicle tours are then assigned to the fleet, which is monitored by the fleet operational control station, which monitors the evolution of deliveries in real time. This process is assisted by the SIMULATE/MONITOR/RE-PLAN module, which allows re-planning online in face of new urgent orders, which were not yet available during the previous off line planning phase. Finally, after vehicles have returned to the depot, delivery data are off-loaded and transferred back to the company database.

In the next paragraphs, we describe two real-world applications, where ACO has been used in the implementation of the vehicle routing algorithms that are implemented in the OPTIMISE block of Figure 1.

5.1 A VRPTW application

In this application the client is one of the major supermarket chains in Switzerland. The problem is to distribute palletized goods to more than 600 stores, all over Switzerland. The stores are the customers of the vehicle routing problem, since they order daily quantities of goods to replenish their local stocks. The stores want the goods to be delivered within time windows, in order to plan in advance the daily availability of their personnel, allocating a fraction of their time to inventory management tasks. The supermarket chain has recently reorganised its logistic process, since it concentrated the distribution process from nine inventories, distributed in various locations, to a central inventory.

There are three types of vehicles: trucks (capacity: 17 pallets), trucks with trailers (35 pallets), and tractor unit with semi-trailer (33 pallets). According to the store location, only some vehicles can access it. In some cases the truck with trailer can leave the trailer at a previous store and then continue alone to other less accessible locations. The number of vehicles in the fleet is assumed infinite, since transport services can be purchased on the market according to the needs.

The road network graph has been computed using an approximation based on the distance in kilometres between couples of stores that has been rescaled using a speed model, which depends on the distance: longer distances allow an higher average speed. For instance, if the distance is less than 5 km, the average speed is 20 km/h; if the distance is more than 90 km, the speed is 60 km/h; in between there is a fine range of other speed values. The data have been collected over many years and they have been validated by the drivers' experience. The time to set-up the vehicle for unloading is constant, as the time required to hook/unhook a trailer. The service time is variable and it depends on the number of pallets to unload.

All the routes must be performed in one day, and the client imposes an extra constraint stating that a vehicle must perform its latest delivery as far as possible from the inventory, since it could be used to perform extra services on its way back. These extra services were not included in the planning by explicit request of the client.

5.1.1 Formalisation of the problem

This problem was formalised as a VRPTW, and ANTROUTE, a modified version of the MACS-VRPTW algorithm [30], was implemented. The algorithm was adapted to the problem in order to handle the choice of the vehicle type, thus, at the start of each tour the ant agent chooses a vehicle. Two ant colonies were used, one minimising the number of vehicles, and the other one the length of the tours. A waiting cost was introduced in order to prevent vehicles arriving too early at the stores. Local search moves allow to improve the quality of the solutions, exchanging stores between routes or reversing the visit order.

5.1.2 The solution: conquering the user acceptance

In most real-world application of Operational Research techniques, part of the effort goes into the formalization of the problem in order to solve it in the most efficient way, but even a greater effort goes into obtaining solutions that are accepted by the user. Sometimes, OR practitioners clearly see the advantage of a solution, which is groundbreaking and therefore upsets the traditional way managers see the problem. Sometimes, convincing the managers in actually adopting this innovative solution is very hard. In our previous research [28] we showed how building a detailed simulation model of the managed system could convince managers of the increase in performance gained by adopting the new management policy. Managers and decision makers are often suspicious about the assumptions made during the modelling phase. They are often right, since models to be used in combinatorial optimisation tend to be as simple as possible, in order to lighten the computational burden. A simulation model can nevertheless take into account uncertainties and processes and reproduce the process in a greater detail. Model calibration and validation and scenario analyses can slowly build the trust of managers in the simulation model. The final step is therefore to show that the optimised policy works well in the detailed simulation model and it can trustworthy be transferred to the real-world. In other cases, a groundbreaking policy would not even be considered, since the hurdles to its implementation, such as changing management practices, are too high. In this application, we faced such a situation. The tours produced by running an adapted version of the MACS-VRPTW algorithm were not accepted as feasible by the human tour planners, even if the performance was quite higher and no explicit constraint was violated. Thus, a further modelling iteration was required, to let 'hidden' constraints emerge. The human planners still adopted a regional planning strategy, that led to petal shaped tours. Rather than trying to impose our viewpoint, we preferred to incorporate this preference, but at the same time we tried to loosen the constraint a bit. We attributed stores to catchment areas, but at the same time we allowed stores near the border of the distribution region to also belong to the neighbouring region. This allowed us to generate tours which are slightly worse than the unconstrained solution, but nevertheless better than the solutions found by the human planners. In Table 2 we present the results obtained by the ANTROUTE compared with the results of the human planners. ANTROUTE was run under two configurations: AR-RegTW, with regional planning and 1-hour time windows; AR-Free, where the regional and the time windows constraints were relaxed. The problem was to distribute 52'000 pallets to 6'800 customers over a period of 20 days. Every day ANTROUTE was run on the available set of orders and it took about 5 minutes to find a solution. At the same time, the planners were at work and it took them at least 3 hours to find a solution. At the end of the testing period, the performances of the algorithm and of the planners have been compared using the same objective function.

Table 2: Comparison of the computer-generated vs. man-made tours in the VRPTW application

	Planner	AR-RegTW	AR-Free	AR-RegTW vs Planner	AR-Free vs Planner
Total number of tours	2'056	1'807	1'614	12.11%	21.50%
Total km	147'271	143'983	126'258	2.23%	14.27%
Average truck loading	76.91%	87.35%	97.81%	10.44%	20.9%

The advantage of an algorithm able to find the solution to an otherwise very hard problem in such a short time is the possibility of using it as a *strategic planning tool*. In Figure 2 it is shown how running the algorithm with wider time-windows at the stores returns a smaller number of tours, which can be translated in a substantial reduction of transportation costs. The logistic manager can therefore use the optimisation algorithm as a tool to investigate how to re-design the time-windows in the stores.

Insert Figure 2 about here

5.2 A real-world VRPPD application

In this second application the client was a major logistic operator in Italy. The distribution process involves moving palletized goods from factories to inventory stores, before they are finally distributed to shops. A customer in this vehicle routing problem is either a pick-up or a delivery point. There is no central depot, and approximately 1'000 – 1'500 customers per day are served. Routes can be performed within the same day, over two days, or over three days, since the Italian peninsula is quite long and there's a strict constraint on the maximum number of hours per day that a driver can travel. All pick-ups of a tour must take place before deliveries. Orders cannot be split among tours. Time windows are associated with each store.

There is only one type of truck: tractor with semi trailer. The load is measured in pallets, in kilograms, and in cubic metres. There are capacity constraints on each one of these measurement units, and the first one that saturates implies the violation of the constraint. Vehicles are assumed to be infinite, since they are provided by flexible sub-contractors. Subcontractors are distributed all over Italy, and therefore vehicles can start their routes from the first assigned customer, and no cost is incurred in travelling to the first customer in the route.

The road network graph has been elicited from digital road maps, computing the shortest path between each couple of stores. The travel times are computed according to the travelled distance, given the average speed that can be sustained on each road segment, according to its type (highway, extraurban road, urban road). Loading and unloading times are assumed to be constant. This is a rough approximation imposed by the client, since they have been insofar unable to provide better estimates, accounting for waiting times at the store, which are quite variable and unpredictable. It is a conservative and risk-averse approach. The client also imposed another constraint, related to the same problem, setting a maximum number of cities to visit per tour (usually less than six). Note that more than one customer can reside in a city. Moreover, the client requested that the distance between successive deliveries should be limited by a parameter.

5.2.1 Formalisation of the problem

The problem can be formalised as a VRPPDTW (VRP with Pickup and Delivery and Time Windows). The objective function measures the efficiency of a tour, expressed as the occupancy ratio of a vehicle over the travelled distance within the tour. The efficiency of tour i is $e_i = \frac{\sum_{j=1}^{M_i} q_j l_j}{Q_i L_i}$, where: M_i is the number of orders in the i-th tour; q_j are the pallets in the j-th order; l_j is the distance between source and destination points of the j-th order; Q_i is the capacity of the vehicle serving the i-th tour; L_i is the total length of the i-th tour. The total efficiency is the sum of the tour efficiencies over the number N of tours: $f = \frac{\sum_{j=1}^{N} e_j}{N}$.

5.2.2 The solution and evaluation of the results

The ANTROUTE algorithm has also been used in this context, but since there is a single objective – to maximise the efficiency – it has been adapted removing the ant colony minimising the number of vehicles. The first step of an ant agent is to select the starting city. Since it is a pickup and delivery problem, each source node must be paired with the corresponding destination node, and the search space is therefore harder to explore than in a delivery problem. The algorithm tries to simplify exploration using an approximation of the delivery phase, assuming that all deliveries will be performed in the reverse order with respect to pickups. Thus, a first level local search exchanges orders between tours, preserving the order of deliveries; later, in a second level local search, nodes are exchanged within the same tour.

Table 3 summarises the comparison between man-made and computer-generated tours over a testing period of two weeks. A noticeable improvement in the efficiency is shown.

Table 3: Comparison of the computer-generated vs. man-made tours in the VRPPDTW application

	Planner	Ants	Absolute difference	Relative difference
Total number of tours	471.5	460.8	-10.7	-2.63%
Total km	175'441	173'623	-1'818.2	-1.32%
Efficiency	84.08%	88.27%	+4.19%	-

It is also interesting to remark how the algorithm performance is correlated with the difficulty of the problem, which is related to the number of orders to satisfy. In Figure 3 we plot on the x-axis the efficiency of the man-made tours, and on the y-axis the efficiency improvement obtained using the computer-generated tours. When the problem is easy, because it involves a limited number of orders, and the human planner performs well, the computer is not able to provide a remarkable improvement, but when the planner starts to fail coping with the problem complexity, and the performance decreases, the gain in using the algorithm sensibly increases.

Insert Figure 3 about here

5.3 Conclusions

In this paper we described how the Ant Colony Optimisation metaheuristic can be successfully used to solve a number of variants of the basic Vehicle Routing Problem. We focused our attention on two important variants, the Time Dependent VRP and the Dynamic VRP, which are receiving increasing attention due to their relevance to real world problems, in particular for distribution in urban environments. Finally, we presented two industrial-scale applications of ACO: the first to a VRPTW problem and the second to a VRPPD problem. In conclusion, after more than ten years of research, ACO has proven to be one of the most successful metaheuristics and its application to real world problems demonstrates that it has now become a fundamental tool in applied Operational Research and Management Science.

References

[1] Y. Aksoy and A. Derbez. Software survey: Supply chain management. *OR/MS Today*, 30(3), June 2003.

[2] L. Bianchi, M. Birattari, M. Chiarandini, M. Manfrin, M. Mastrolilli, L. Paquete, O Rossi-Doria, and T. Schiavinotto. Metaheuristics for the vehicle routing problem with stochastic demands. Technical Report TR-12-04, IDSIA, Galleria 2, Manno, 6928, Switzerland, 2004.

[3] C. Blum and A. Roli. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Computing Surveys*, 35(3):268–308, 2003.

[4] E. Bonabeau, M. Dorigo, and G. Theraulaz. Inspiration for optimization from social insect behaviour. *Nature*, 406:39–42, July 2000.

[5] O. Bräysy. A reactive variable neighborhood search for the vehicle routing problem with time windows. *INFORMS Journal on Computing*, 15(4):347–368, 2003.

[6] A. Van Breedam. An analysis of the effect of local improvement operators in genetic algorithms and simulated annealing for the vehicle routing problem. RUCA Working Paper 96/14, University of Antwerp, 1996.

[7] B. Bullnheimer, R. F. Hartl, and C. Strauss. Applying the ant system to the vehicle routing problem. In *Proceedings of the 2nd International Conference on Metaheuristics - MIC97*. INRA Sophia-Antipolis & PRiSM Versailles, 1997.

[8] B. Bullnheimer, R.F. Hartl, and C. Struss. An improved ant system algorithm for the vehicle routing problem. *Annals of Operations Research*, 89:319–328, 1999.

[9] N. Christofides and J. Beasley. The period routing problem. *Networks*, 14:237–256, 1984.

[10] N. Christofides, A. Mingozzi, and P. Toth. *Combinatorial Optimization*, chapter The Vehicle Routing problem, pages 315–338. Wiley, 1979.

[11] A.J. Clark and H. Scarf. Optimal policies for a multi-echelon inventory problem. *Management Science*, 6:475–490, 1960.

[12] G. Clarke and J.W. Wright. Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*, 12(4), 1964.

[13] A. Colorni, M. Dorigo, V. Maniezzo, and M. Trubian. Ant system for job-shop scheduling. *JORBEL - Belgian Journal of Operations Research, Statistics and Computer Science*, 34(1):39–53, 1994.

[14] J.-F. Cordeau, G. Desaulniers, J. Desrosiers, M.M. Solomon, and F. Soumis. *The Vehicle Routing Problem*, chapter VRP with Time Windows, pages 157–193. SIAM, Society for Industrial and Applied Mathematics, Philadelphia, USA, 2000.

[15] G.A. Croes. A method for solving traveling-salesman problems. *Operations Research*, 6:791–812, 1958.

[16] J.-L. Deneubourg, S. Aron, and S. Gossand J.-M. Pasteels. The self-organizing exploratory pattern of the Argentine ant. *Journal of Insect Behavuiour*, 3:159–168, 1990.

[17] G. Desaulniers, J. Desrosiers, A. Erdmann, M.M. Solomon, and F. Soumis. *The Vehicle Routing Problem*, chapter VRP with Pickup and Delivery, pages 225–242. SIAM, Society for Industrial and Applied Mathematics, Philadelphia, USA, 2000.

[18] A.V. Donati, R. Montemanni, N. Casagrande, A. E. Rizzoli, and L.M. Gambardella. Time dependent vehicle routing problem with a multi ant colony system. Technical Report TR-17-03, IDSIA, Galleria 2, Manno, 6928, Switzerland, 2003.

[19] M. Dorigo, G. Di Caro, and L.M. Gambardella. Ant algorithms for discrete optimization. *Artificial Life*, 5:137–172, 1999.

[20] M. Dorigo and L.M. Gambardella. Ant colony system: A cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, 1(1):53–66, 1997.

[21] M. Dorigo, V. Maniezzo, and A. Colorni. The ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man and Cybernetics - Part B: Cybernetics*, 26(1):29–41, 1996.

[22] M. Fisher, R. Jakumar, and L. van Wassenhove. A generalized assignment heuristic for vehicle routing. *Networks*, 11:109–124, 1981.

[23] M.L. Fisher. Optimal solution of vehicle routing problems using minimum k-trees. *Operations Research*, 42(4):626–642, 1994.

[24] L.M. Gambardella. Numerical results for the solomon problems, http://www.idsia.ch/~luca/macs-vrptw/solutions/welcome.htm, Last accessed: May 2004.

[25] L.M. Gambardella and M. Dorigo. Ant-Q: a reinforcement learning approach to the travelling salesman problem. In *Proceedings of the Twelfth International Conference on Machine Learning, ML-95*. Morgan Kaufmann, Palo Alto, California, USA, 1995.

[26] L.M. Gambardella and M. Dorigo. Solving symmetric and asymmetric TSPs by ant colonies. In *IEEE Conference on Evolutionary Computation (ICEC96)*, pages 622–627, 1996.

[27] L.M. Gambardella and M. Dorigo. An ant colony system hybridized with a new local search for the sequential ordering problem. *INFORMS Journal on Computing*, 12(3):237–255, 2000.

[28] L.M. Gambardella, M. Mastrolilli, A.E. Rizzoli, and M. Zaffalon. An integrated approach to the optimisation of an intermodal terminal based on efficient resource allocation and scheduling. *Journal of Intelligent Manufacturing*, 12(5/6):521–534, October 2001.

[29] L.M. Gambardella, A.E. Rizzoli, and M. Zaffalon. Simulation and planning of an intermodal container terminal. *Simulation*, 71(2):107–116, 1998.

[30] L.M. Gambardella, É. Taillard, and G. Agazzi. MACS-VRPTW: a multiple ant colony system for vehicle routing problems with time windows. in *New ideas in optimization*. D. Corne et al. editors. Pages 63-76, 1999.

[31] M. Gendreau, F. Guertin, J.-Y. Potvin, and É. Taillard. Parallel tabu search for real-time vehicle routing and dispatching. *Transportation Science*, 33(4):381–390, November 1999.

[32] M. Gendreau, A. Hertz, and G. Laporte. A tabu search heuristic for the vehicle routing problem. *Management Science*, 40(10):1276–1290, October 1994.

[33] M. Gendreau, G. Laporte, and R. Séguin. Stochastic vehicle routing. *European Journal of Operational Research*, 88(1):3–12, 1996.

[34] M. Gendreau and J.-Y. Potvin. Dynamic vehicle routing and dispatching. in *Fleet management and logistic*. T.G. Crainic and G. Laporte editors. Pages 115-226, 1998.

[35] F. Glover. Future paths for integer programming and links to artificial intelligence. *Computers & Operations Research*, 13(5):533–549, 1986.

[36] F. Glover and M. Laguna. *Tabu Search*. Kluwer Academic Publishers, Boston, Massachusetts, USA, 1997.

[37] APES Group. Apes data – algorithms, problems, empirical studies, http://www.dcs.st-and.ac.uk/˜apes/apedata.html, Last accessed: April 2004.

[38] M. Guntsch and M. Middendorf. Pheromone modification strategies for ant algorithms applied to dynamic TSP. In E.J.W. Boers et al., editor, *Application of evolutionary com-*

puting: Proceedings of EcoWorkshops 2001, volume Lecture Notes in Computer Science 2037, pages 213–222, 2001.

[39] J. Homberger. Extended Solomon's VRPTW instances. http://www.fernuni-hagen.de/winf/touren/inhalte/probinst.htm, Last accessed: April 2004.

[40] S. Ichoua, M. Gendreau, and J.-Y. Potvin. Diversion issues in real-time vehicle dispatching. *Transportation Science*, 34(4):426–438, November 2000.

[41] S. Ichoua, M. Gendreau, and J.-Y. Potvin. Vehicle dispatching with time-dependent travel times. *European Journal of Operational Research*, 144(2):379–396, 2003.

[42] P. Kilby, P. Prosser, and P. Shaw. Dynamic VRPs: a study of scenarios. Technical Report APES-06-1998, University of Strathclyde, September 1998.

[43] P. Kilby, P. Prosser, and P. Shaw. *Meta-heuristics: Advances and Trends in Local Search Paradigms for Optimization*, chapter Guided local search for the vehicle routing problem, pages 473–486. Kluwer Academic Publishers Boston, 1999.

[44] G.A.P. Kindervater and M.W.P. Savelsbergh. *Local Search in Combinatorial Optimization*, chapter Vehicle Routing: Handling Edge Exchanges, pages 337–360. Wiiley, Chichester, UK, 1997.

[45] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[46] G. Laporte and F.V. Louveaux. *Fleet Management and Logistics*, chapter Solving stochastic routing problems with the integer L-shaped method, pages 159–167. Kluwer Academic Publishers, Boston/Dordrecht/London, 1998.

[47] G. Laporte and F. Semet. *The Vehicle Routing Problem*, chapter Classical Heuristics for the Capacitated VRP, pages 109–128. SIAM, Society for Industrial and Applied Mathematiccs, Philadelphia, USA, 2000.

[48] H.R. Maier, A.R. Simpson, A.C. Zecchin, W.K. Foong, K.Y. Phang, H.Y. Seah, and C.L.Tan. Ant colony optimization for design of water distribution systems. *Journal of Water Resources Planning and Management*, 129(3):200–209, May/June 2003.

[49] O. Martin, S.W. Otto, and E.W. Felten. Large-step markov chains for the TSP incorporating local search heruristics. *Operations Research Letters*, 11:219–224, May 1992.

[50] D. Mester and O. Bräysy. Active guided evolution strategies for the large scale vehicle routing problem with time windows. *Computers & Operations Research*, pages xx–xx, To appear.

[51] R. Montemanni, L.M. Gambardella, A.E. Rizzoli, and A.V. Donati. A new algorithm for a dynamic vehicle routing problem based on ant colony system. Technical Report TR-23-02, IDSIA, Galleria 2, Manno, 6928, Switzerland, 2004.

[52] I.H. Osman. Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem. *Annals of Operations Research*, 41:421–451, 1993.

[53] H. Paessens. The savings algorithm for the vehicle routing problem. *European Journal of Operations Research*, 34(3):336–344, 1988.

[54] H. Psaraftis. Dynamic vehicle routing problems. in *Vehicle Routing: methods and Studies*. B.L. Golden and A.A. Assad editors. Pages 223-248, 1988.

[55] H. Psaraftis. Dynamic vehicle routing: status and prospects. *Annals of Operations Research*, 61:143–164, 1995.

[56] M. Reimenn, K. Doerner, and R.F. Hartl. A savings based ant system for the vehicle routing problem. In W.B. Langdon et al., editor, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2002)*, pages 1317–1325. Morgan Kaufmann Publishers, San Francisco, CA, 2002.

[57] M.W.P. Savelsbergh. Local search in routing problems with time windows. *Annals of Operations Research*, 4:285–305, 1985.

[58] M.S. Sodhi. Applications and opportunities for operations research in internet-enabled supply chains and electronic marketplaces. *Interfaces*, 31(2):56–69, 2001.

[59] M. Solomon. Best known solutions identified by heuristics. http://web.cba.neu.edu/˜msolomon/heuristi.htm, Last accessed: April 2004.

[60] T. Stützle and H. Hoos. Analyzing the run-time behaviour of iterated local search for the tsp. In P. Hansen and C.C. Ribeiro, editors, *Proceedings of III Metaheuristics International Conference (MIC99)*, pages 449–453. PUC-Rio - Catholic University of Rio de Janeiro, 1999.

[61] È.D. Taillard. Parallel iterative search methods for vehicle-routing problems. *Networks*, 23(8):661–673, 1994.

[62] È.D. Taillard, E.P. Badeau, M. Gendreau, F. Guertin, and J. Y. Potvin. A tabu search heuristic for the vehicle routing problem with soft time windows,. *Transportation Science*, 31(2):170–186, 1997.

[63] P. Toth and D. Vigo. *Fleet Management and Logistics*, chapter Exact solution of the vehicle routing problem, pages 1–31. Kluwer Academic Publishers: Boston/Dordrecht/London, 1998.

[64] P. Toth and D. Vigo. *The Vehicle Routing Problem*, chapter An Overview of Vehicle Routing Problems, pages 1–26. SIAM, Society for Industrial and Applied Mathematics, Philadelphia, USA, 2000.

[65] P. Toth and D. Vigo. The granular tabu search and its application to the vehicle routing problem. *INFORMS Journal on Computing*, 15(4):333–346, Fall 2003.

[66] C.J. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8:279–292, 1992.

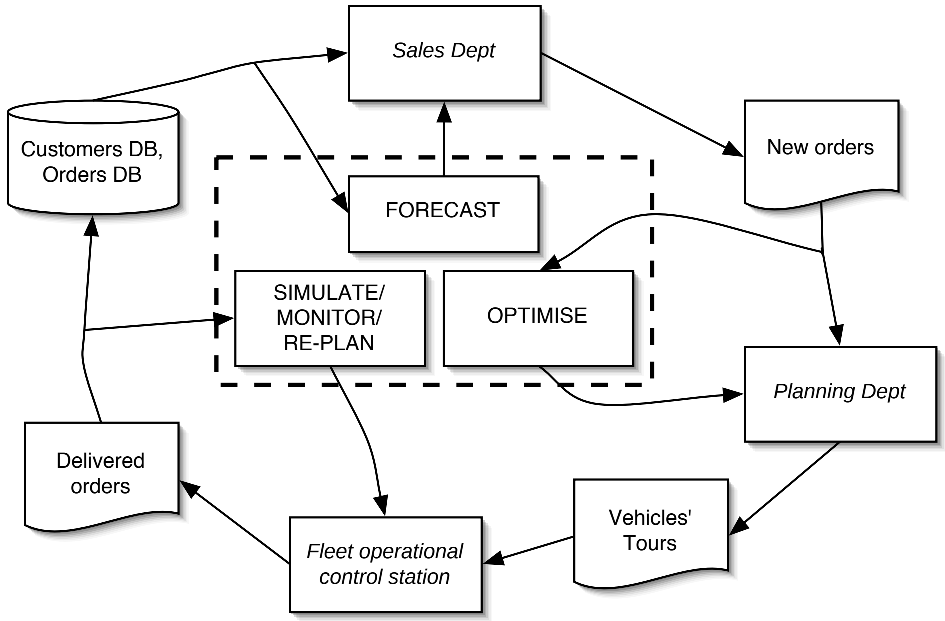

Figure 1: The Forecast-Optimise-Simulate loop: the role of optimisation in the efficient management of a distribution process.

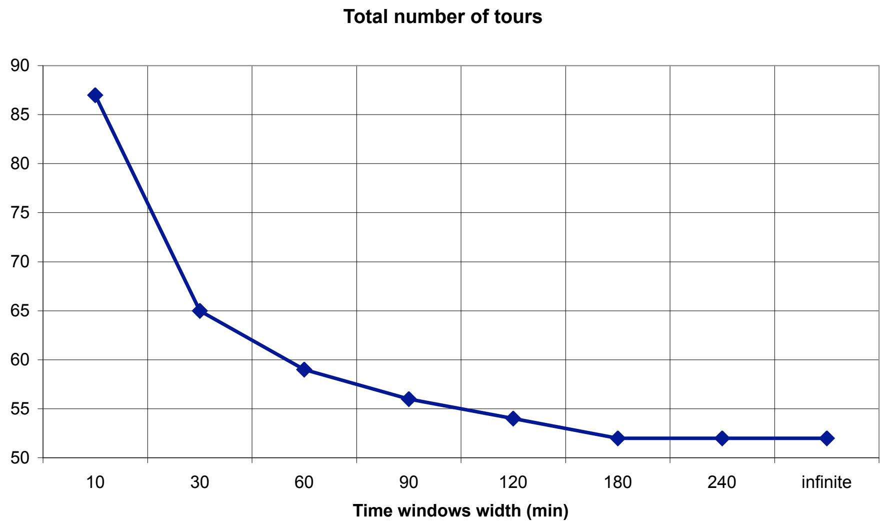

Figure 2: The relationship between the time window width and the number of vehicle routes.

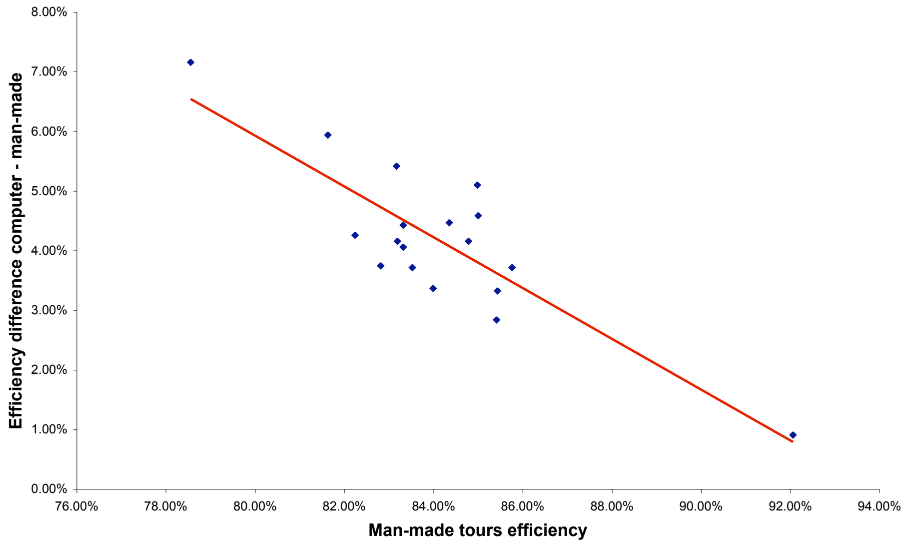

Figure 3: Comparing man-made and computer-generated tours. Higher efficiency improvements are observed when the human planner performance is lower. The dots are experimental values, and the solid line is a regression on those values.