An Ant-Based Optimization Approach for Inventory Routing

Vasileios A. Tatsis[1], Konstantinos E. Parsopoulos[1], Konstantina Skouri[2], and Ioannis Konstantaras[3]

[1] Department of Computer Science, University of Ioannina, GR-45110 Ioannina, Greece
{vtatsis,kostasp}@cs.uoi.gr
[2] Department of Mathematics, University of Ioannina, GR-45110 Ioannina, Greece
kskouri@uoi.gr
[3] Department of Business Administration, University of Macedonia, GR-54006 Thessaloniki, Greece
ikonst@uom.gr

Abstract. The inventory routing problem (IRP) is a major concern in operation management of a supply chain because it integrates transportation activities with inventory management. Such problems are usually tackled by independently solving the underlying inventory and vehicle routing sub-problems. The present study introduces an ant-based solution framework by modeling the IRP problem as a vehicle routing task. In this context, a mixed-integer mathematical model for the IRP is developed, where a fleet of capacitated homogeneous vehicles transport different products from multiple suppliers to a retailer to meet the demand for each period over a finite planning horizon. In our model, shortages are allowed while unsatisfied demand is backlogged and can be met in future periods. The mathematical model is used to find the best compromise among transportation, holding, and backlogging costs. The corresponding vehicle routing problem is solved using an ant-based optimization algorithm. Preliminary results on randomly generated test problems are reported and assessed with respect to the optimal solutions found by established linear solvers such as CPLEX.

1 Introduction

The main target of Supply Chain Management (SCM) is to align the various stages of the supply chain. Integrating the decisions in planning the different activities, has shown to produce improved global performance. An example of integrating and coordinating decisions can be found in Vendor Managed Inventory (VMI), where customers make their inventory data available to their suppliers (distributors), who then take the responsibility of deciding when to replenish which customers. Thus, the supplier has to choose how often, when, and in what quantities the different customers are replenished. This integrated inventory and distribution management offers more freedom for designing efficient vehicle routes, while optimizing inventory across the supply chain.

The underlying optimization problem that has to be addressed by the supplier, namely the simultaneous decision on the replenishment quantities and the vehicle

M. Emmerich et al. (eds.), *EVOLVE - A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation IV*, Advances in Intelligent Systems and Computing 227,
DOI: 10.1007/978-3-319-01128-8_8,

routes to visit all customers, is known as the *Inventory Routing Problem* (IRP). IRP is one of the most interesting and challenging issues in the field of supply chain management and typically considers a distribution firm that operates a central vehicle station, supplying a number of geographically scattered customers by a fleet of homogeneous vehicles, over a period of time [6].

The IRP literature is extensive and includes several variants of the problem, mainly depending on the nature of the demand at customers (deterministic, stochastic etc.) as well as on the length of the planning horizon (finite, infinite etc.) We can indicatively mention several relevant works considering single-period IRPs with stochastic demand [13] or deterministic demand [9]; multi-period finite horizon IRPs with constant or dynamic demand [1, 2, 7, 17]; as well as infinite horizon IRPs with deterministic or stochastic demand [8, 14]. For recent detailed reviews of the IRP field, the reader is referred to [3, 16].

In the present work, we propose a constructive meta-heuristic approach for solving a two-echelon supply chain problem, where one retailer is served multi-products by different suppliers, using a fleet of capacitated homogeneous vehicles. This forms a multi-product, multi-period, finite horizon IRP where the retailer's demands are assumed to be known for all periods. The IRP problem is first modeled as an equivalent vehicle routing (VR) task. Then, we use an ant-based algorithm that combines elements from some of the most successful Ant Colony Optimization (ACO) variants, namely (Elitist) Ant System ((E)AS) [5, 10, 12] and Max-Min Ant System ($\mathcal{MM}$AS) [19], to solve the corresponding VR problem. Preliminary experimental results on a set of randomly generated test problems are reported. The results are compared with ones obtained using the CPLEX software, offering preliminary evidence regarding the competitiveness and weaknesses of the proposed approach.

The rest of the paper is organized as follows: Section 2 contains the mathematical formulation of the problem, while Section 3 describes the algorithm, in detail. Preliminary experimental results are reported in Section 4. Finally, the paper concludes in Section 5.

2 Problem Formulation

We consider a many-to-one, part-supply network that is similar to the one proposed in [17]. The network consists of one retailer, N suppliers, and a vehicle station. Each supplier provides a distinct product to the retailer. Henceforth, we will denote each supplier (and his product) with the corresponding index $i = 1, 2, \ldots, N$, while the index 0 will denote the station, and $N + 1$ will denote the retailer. A fleet of homogeneous capacitated vehicles housed at the vehicle station, transports products from the suppliers to meet the demand specified by the retailer over a finite horizon, while backlogging is allowed. The vehicles return to the vehicle station at the end of each trip. If the demand for more than one period is collected, the inventory is carried forward subject to product-specific holding cost. The unsatisfied demand for a specific product leads the related product-specific shortage cost.

The main objective of the problem is the minimization of the total transportation, inventory, and shortages costs over the planning horizon. Putting it formally, let the sets of indices:

$$\begin{array}{ll} \text{Suppliers:} & I_s = \{1, 2, \ldots, N\}, \\ \text{Vehicles:} & I_v = \{1, 2, \ldots, M\}, \\ \text{Time periods:} & I_p = \{1, 2, \ldots, T\}, \end{array} \tag{1}$$

and $I'_s = I_s \cup \{N+1\}$. We use the following notation, which is similar to [17], although our model assumes a finite fleet size instead of the unlimited number of vehicles in [17]:

C: capacity of each vehicle.
F: fixed vehicle cost per trip (same for all periods).
V: travel cost per unit distance.
M: size of the vehicle fleet.
d_{it}: retailer's demand for product from supplier i in period t.
c_{ij}: travel distance between supplier i and j where $c_{ij} = c_{ji}$ and the triangle inequality, $c_{ik} + c_{kj} \geqslant c_{ij}$, holds for all i, j, k with $i \neq j$, $k \neq i$, and $k \neq j$.
h_i: holding cost at the retailer for product i per unit product per unit time.
s_i: backlogging cost at the retailer for product i per unit product per unit time.
I_{i0}: inventory level of product i at the retailer, at the beginning of period 1.
a_{it}: total amount to be picked up at supplier i in period t.
I_{it}: inventory level of product from supplier i at the retailer, at end of period t.
q_{ijt}: quantity transported through the directed arc (i, j) in period t.
x_{ijt}: number of times that the directed arc (i, j) is visited by vehicles in period t.

Then, the mathematical formulation of the problem is defined as follows:

$$\text{minimize} \quad Z_1 + Z_2 + Z_3 + Z_4 + Z_5, \tag{2}$$

where:

$$\begin{aligned} Z_1 &= \sum_{i=1}^{N} h_i \sum_{t=1}^{T} I_{it}^{+}, \\ Z_2 &= \sum_{i=1}^{N} s_i \sum_{t=1}^{T} (-I_{it})^{+}, \\ Z_3 &= V \left(\sum_{\substack{j=1 \\ j \neq i}}^{N} \sum_{i=0}^{N} c_{ij} \left(\sum_{t=1}^{T} x_{ijt} \right) \right), \\ Z_4 &= V \left(\sum_{i=1}^{N} c_{i,N+1} \left(\sum_{t=1}^{T} x_{i,N+1,t} \right) \right), \\ Z_5 &= (F + c_{N+1,0}) \sum_{i=1}^{N} \sum_{t=1}^{T} x_{0it}, \end{aligned} \tag{3}$$

where $x^+ = \max\{x, 0\}$, subject to the constraints:

$$\text{(C1):} \quad I_{it} = I_{it-1} + a_{it} - d_{it}, \quad i \in I_s, t \in I_p, \tag{4}$$

$$\text{(C2):} \quad \sum_{\substack{i=0 \\ i \neq j}}^{N} q_{ijt} + a_{jt} = \sum_{\substack{i=1 \\ i \neq j}}^{N+1} q_{jit}, \quad j \in I_s, t \in I_p, \tag{5}$$

$$\text{(C3):} \quad \sum_{i=1}^{N} q_{i,N+1,t} = \sum_{i=1}^{N} a_{it}, \quad t \in I_p, \tag{6}$$

$$\text{(C4):} \quad \sum_{\substack{i=0 \\ i \neq j}}^{N} x_{ijt} = \sum_{\substack{i=1 \\ i \neq j}}^{N+1} x_{jit}, \quad j \in I_s, t \in I_p, \tag{7}$$

$$\text{(C5):} \quad \sum_{j=1}^{N} x_{0jt} = \sum_{j=1}^{N} x_{j,N+1,t}, \quad t \in I_p, \tag{8}$$

$$\text{(C6):} \quad q_{ijt} \leqslant C\, x_{ijt}, \quad i \in I_s, j \in I'_s, i \neq j, t \in I_p, \tag{9}$$

$$\text{(C7):} \quad \sum_{i=1}^{N} x_{0it} \leqslant M, \quad t \in I_p, \tag{10}$$

$$\text{(C8):} \quad \sum_{t=1}^{T} a_{it} = \sum_{t=1}^{T} d_{it}, \quad i \in I_s, \tag{11}$$

$$\text{(C9):} \quad C\, x_{ijt} - q_{ijt} \leqslant C - 1, \quad i \in I_s, j \in I'_s, t \in I_p, \tag{12}$$

$$\text{(C10):} \quad a_{jt} \leqslant \sum_{\substack{i=1 \\ i \neq j}}^{N} C\, x_{ijt}, \quad j \in I_s, t \in I_p, \tag{13}$$

$$\text{(C11):} \quad x_{ijt} \leqslant \sum_{\substack{k=0 \\ k \neq i,j}}^{N} x_{kit} \quad i \in I_s, j \in I'_s, t \in I_p \tag{14}$$

$$\text{(C12):} \quad I_{it} \in R, \quad i \in I_s, t \in I_p, \tag{15}$$

$$\text{(C13):} \quad a_{it} \geqslant 0, \quad i \in I_s, t \in I_p, \tag{16}$$

$$\text{(C14):} \quad x_{ijt} \in \{0, 1\}, \quad i, j \in I_s, t \in I_p, \tag{17}$$

$$\text{(C15):} \quad x_{0jt} \in \mathbb{N}, \quad j \in I_s, t \in I_p, \tag{18}$$

$$\text{(C16):} \quad x_{i,N+1,t} \in \mathbb{N}, \quad i \in I_s, t \in I_p, \tag{19}$$

$$\text{(C17):} \quad x_{0,N+1,t} = 0, \quad t \in I_p, \tag{20}$$

$$\text{(C18):} \quad x_{i0t} = 0, \quad i \in I_s, t \in I_p, \tag{21}$$

$$\text{(C19):} \quad x_{N+1,j,t} = 0, \quad j \in I_s, t \in I_p, \tag{22}$$

$$\text{(C20):} \quad q_{ijt} \geqslant 0, \quad i \in I_s, j \in I'_s, t \in I_p, \tag{23}$$

$$\text{(C21):} \quad q_{0jt} = 0, \quad j \in I_s, t \in I_p. \tag{24}$$

The objective function defined by Eqs. (2) and (3) comprises both inventory costs (holding and backlogging) and transportation costs (variable travel costs and vehicle fixed cost). We note that the fixed transportation cost consists of the fixed cost incurred per trip and the constant cost of vehicles returning to the station from the retailer.

Constraint (C1) is the inventory balance equation for each product, while (C2) is the product flow conservation equations, assuring the flow balance at each supplier and eliminating all subtours. Constraint (C3) assures the accumulative picked up quantities at the retailer and (C4) and (C5) ensure that the number of vehicles leaving a supplier, the retailer or the station is equal to the number of its arrival vehicles. We note that constraint (C5) is introduced because each vehicle has to visit the retailer before returning to the station. Constraint (C6) guarantees that the vehicle capacity is respected and gives the logical relationship between q_{ijt} and x_{ijt}, which allows for split pick ups.

Constraint (C7) is introduced due to the limited fleet size. Constraint (C8) ensures that the cumulative demand for every product will be satisfied, while (C9) is imposed to ensure that either $q_{ijt} = 0$ with $x_{ijt} = 0$ or $q_{ijt} \geqslant 1$ with $x_{ijt} \geqslant 1$. Moreover, (C10) ensures that the pick up quantities are limited by the number of vehicles and their capacities. Constraint (C11) ensures that if there is a vehicle to travel from one supplier to another, then this vehicle should previously arrive to the first supplier from another one (or the station). This is necessary to avoid fake closed loops of vehicles that may appear in the VR formulation of the problem. Finally, (C12) implies that the demand can be backlogged. The rest are non-negativity constraints imposed on the variables. We note that (C17), (C18), and (C19) ensure the absence of direct links from the station to the retailer, from a supplier to the station, and from the retailer to a supplier, respectively.

3 Proposed Approach

3.1 Ant Colony Optimization

Ant Colony Optimization (ACO) constitutes a general metaheuristic methodology for solving combinatorial optimization problems [12]. It draws inspiration from the collective behavior of termites during social activities such as foraging. The emergent behavior of such primitive entities is based on a mechanism called stigmergy, which allows them to coordinate their actions based on stimulation from the environment. The stimulation is their response to pheromone's chemical traces left in the environment by their mates and them.

ACO can be elegantly introduced in the framework of the Traveling Salesman Problem (TSP) as a group (swarm) of agents that individually construct a route from a start city to an end city visiting all other cities just once. At each stage of the route construction, the agent makes a decision of its next move based on a probabilistic scheme that gives higher probability to the alternatives that have more frequently been visited by the rest of the swarm and, thus, they possess higher pheromone levels. The general procedure flow of ACO approaches can be summarized in the following actions:

```
// Procedure ACO
WHILE (termination condition is false)
    Construct_Solutions()
    Further_Actions()
    Update_Pheromones()
END WHILE
```

Due to space limitation the reader is referred to devoted texts such as [5,12] for a thorough introduction of the general framework of ACO and its most popular variants.

3.2 Solution Representation

We will now try to put the considered IRP problem in a form that can be handled from the considered ant-based approaches that will be later described. Let the problem consist of N suppliers that employ M vehicles to transport their products over a time horizon of T periods. Following the notation presented in Section 2, we consider the sets of indices, I_s, I_v, and I_p, defined in Eq. (1). Our approach is vehicle-centric, i.e., each vehicle constructs its order for visiting the suppliers at each time period. The decision of not visiting a specific supplier implies that the supplier is absent in the constructed visiting order. This formulation is adequate to transform the IRP into an equivalent VR problem as described below.

Putting it formally, let $p_{it}^{[j]}$ denote the position of supplier i in the visiting order of vehicle j at time period t. Then, it holds that:

$$p_{it}^{[j]} \in \{0, 1, 2, \ldots, N\}, \quad \text{for all} \quad i \in I_s, j \in I_v, t \in I_p, \tag{25}$$

where $p_{it}^{[j]} = 0$ simply denotes that supplier i is not visited by vehicle j at time period t. The quantities $p_{it}^{[j]}$ for all i, j, and t, are adequate to provide the visiting frequencies x_{ijt} in our IRP model described in Section 2. Indeed, the set of indices of the vehicles that contain the directed arc (i, j) with $i, j \in I_s$, $i \neq j$, in their routes at time period t, is defined as:

$$K_{(i,j,t)} = \Big\{ k \in I_v \ \text{ such that } \ p_{it}^{[k]} = l - 1, \ p_{jt}^{[k]} = l, \ l \in I_s \setminus \{1\} \Big\}.$$

Moreover, let the sets:

$$K_{(0,i,t)} = \Big\{ k \in I_v \ \text{ such that } \ p_{it}^{[k]} = 1 \Big\},$$
$$K_{(i,N+1,t)} = \Big\{ k \in I_v \ \text{ such that } \ p_{it}^{[k]} > p_{jt}^{[k]}, \ \forall j \in I_s \setminus \{i\} \Big\},$$

which define the vehicles that visit supplier i first, and the vehicles that visit supplier i last, just before completing their route at the retailer, respectively. Then, if $|K_{(i,j,t)}|$ denotes the cardinality of $K_{(i,j,t)}$, we can easily infer that:

$$x_{ijt} = \begin{cases} |K_{(i,j,t)}|, & \text{if } K_{(i,j,t)} \neq \emptyset, \\ 0, & \text{otherwise,} \end{cases} \qquad \text{for all} \quad i, j, t. \tag{26}$$

This equation determines all the visiting frequencies in the IRP, solely using the constructed visiting orders of the vehicles.

Apart from the visiting order, each vehicle also follows a policy regarding the quantities that are picked up from each supplier. The policy is based on the reasonable assumption that a vehicle shall satisfy all demand and backlogging requirements as long as it is permitted by its capacity. In other words, if $L_{it}^{[k]}$ denotes the k-th vehicle's load when visiting supplier i at time t, and $a_{it}^{[k]}$ is the quantity of products that it will pick up from the supplier, it shall hold that $a_{it}^{[k]} = \min\left\{C - L_{it}^{[k]},\, d_{it} - I_{i,t-1}\right\}$, where C is the vehicle's capacity, while d_{it} and $I_{i,t-1}$ stand for the demand and the current inventory, respectively. Obviously, if $p_{it}^{[k]} = 0$ (i.e., the vehicle does not visit supplier i) then the picked-up quantity will be $a_{it}^{[k]} = 0$. Then, the total quantity picked up by supplier i at time t by all vehicles, is given by:

$$a_{it} = \sum_{k=1}^{M} a_{it}^{[k]}.$$

We can easily notice that, according to this equation, it is possible that the total amount picked up by supplier i becomes larger than the quantity dictated by the system's demands. This may be observed in the case where a number of vehicles with adequate free capacity visit the same supplier, each one picking an amount equal to $d_{it} - I_{i,t-1}$. However, such a solution will be infeasible due to the constraint of Eq. (11) and, eventually, it will be rejected by the algorithm.

The rest of the model's parameters, i.e., the inventory levels I_{it} and the transported quantities q_{ijt}, can be straightforwardly determined by taking into consideration the constraints of the IRP model. Specifically, I_{it} is given directly from Eq. (4), while the transported quantities are given as:

$$q_{ijt} = \sum_{k \in K_{(i,j,t)}} \left(L_{it}^{[k]} + a_{it}^{[k]}\right), \qquad i, j \in I_s,\ i \neq j.$$

Thus, the vehicles' visiting orders can offer all the necessary information to describe the whole system's operation, rendering the VR problem an equivalent formulation of the original IRP.

Based on this formulation, we considered a solution representation scheme that consists of the visiting orders of all vehicles for all time periods, as follows:

$$\Big(\quad \cdots \quad \underbrace{\underbrace{p_{1t}^{[1]}, \ldots, p_{Nt}^{[1]},}_{\text{vehicle 1}} \quad \ldots, \quad \underbrace{p_{1t}^{[M]}, \ldots, p_{Nt}^{[M]},}_{\text{vehicle M}}}_{\text{time period t}} \quad \cdots \quad \Big), \tag{27}$$

where $p_{it}^{[j]}$ is defined as in Eq. (25). Obviously, for a problem with N suppliers, M vehicles, and T time periods, this scheme requires a fixed-size vector of $N \times M \times T$ components to represent a candidate solution.

For example in the case of a problem with 2 suppliers, 2 vehicles, and 2 time periods ($N = M = T = 2$), a candidate solution would be an 8-dimensional vector:

$$\Big(\underbrace{\underbrace{p_{11}^{[1]}, \; p_{21}^{[1]}}_{\text{vehicle 1}}, \; \underbrace{p_{11}^{[2]}, \; p_{21}^{[2]}}_{\text{vehicle 2}}}_{\text{time period 1}}, \; \underbrace{\underbrace{p_{12}^{[1]}, \; p_{22}^{[1]}}_{\text{vehicle 1}}, \; \underbrace{p_{12}^{[2]}, \; p_{22}^{[2]}}_{\text{vehicle 2}}}_{\text{time period 2}} \Big),$$

where $p_{it}^{[j]}$ is defined as described above. Each ant in our approach has to construct such vectors, based on the procedures described in the following section.

3.3 Algorithm Operators and Procedures

The employed algorithm is based on the general framework and operation of established ACO variants. More specifically, it considers a group of agents, called *ants*, which iteratively construct a set of potential solutions of the form described in the previous section. The solution components are stochastically selected from a set of possible values (states), similarly to the stochastic selection of a route between several cities. Thus, each component value assumes a weight that is used for the computation of its selection probability. The weights constitute the *pheromone* values that guide the ants, and they are retained in a continuously updated pheromone table.

In our approach, the components' values that are selected more frequently in the best solutions during the run of the algorithm, increase their pheromone levels and, consequently, they are more frequently selected by the ants. Pheromone restarting is also applied after a number of iterations to alleviate search stagnation. All these operations are thoroughly described in the following paragraphs.

In our case, the algorithm assumes K ants, which iteratively construct candidate solutions while retaining in memory the best one from the beginning of the run. Each ant constructs a solution vector of the form of Eq. (27), in a componentwise manner. The solution construction process is based on the probabilistic selection of each component's value, based on the table of pheromones.

More specifically, there is a pheromone value, $\tau_{it}^{[j]}(l)$, $l = 0, 1, \ldots, N$, for each one of the l possible values (states) of the variables $p_{it}^{[j]}$ defined in Eq. (25), i.e.:

$$\begin{array}{lccccccc} \text{States of } p_{it}^{[j]}: & \{ & 0, & & 1, & \ldots, & N & \} \\ & & \uparrow & & \uparrow & & \uparrow & \\ \text{Pheromones:} & & \tau_{it}^{[j]}(0) & & \tau_{it}^{[j]}(1) & \cdots & \tau_{it}^{[j]}(N) & \end{array}$$

The corresponding probability of taking $p_{it}^{[j]} = l$ is computed as:

$$\rho_{it}^{[j]}(l) = \frac{\tau_{it}^{[j]}(l)}{\sum_{k=0}^{N} \tau_{it}^{[j]}(k)}, \qquad \forall\, i, j, t.$$

It is trivial to verify the necessary condition:

$$\sum_{l=0}^{N} \rho_{it}^{[j]}(l) = 1, \qquad \forall\, i, j, t.$$

Each ant uses these probabilities to decide for the assigned component value (similarly as deciding among cities in the TSP) through the well-known *fitness proportionate selection* (also known as *roulette-wheel selection*) procedure [4].

In practice, there are some limitations in this procedure. For example, $p_{it}^{[j]}$ cannot take the same value with a previously determined $p_{kt}^{[j]}$, with $k \neq i$, since this would imply that suppliers i and k are concurrently visited by vehicle j at time t. These constraints can be handled either by penalizing the corresponding solutions in the objective function or by allowing only the proper states to participate in the selection procedure above. We followed the latter approach since such restrictions can be straightforwardly incorporated in our algorithm, while it does not add further constraints to the (already over-constrained) problem.

As soon as a candidate solution is constructed, it is evaluated with the objective function and all constraints are evaluated. If the solution is infeasible, then its objective value is penalized on the basis of the number and magnitude of constraints violations. We postpone the description of the penalty function until the next section.

Immediately after the construction and evaluation of all K candidate solutions, there is an update procedure for the best solution detected from the beginning of the run. Specifically, each constructed solution is compared to the best solution and, if superior, it replaces it. In order to avoid strict feasibility restrictions that could lead to reduced search capability of the algorithm, we allow infeasible solutions to be constructed, although adopting the following common rules for updating the best solution:

(a) Between feasible solutions, the one with the smallest objective value is selected.
(b) Between infeasible solutions, the one with the smallest total penalty is selected.
(c) Between a feasible best solution and an infeasible new candidate, the feasible best solution is always selected.
(d) Between an infeasible best solution and a feasible new candidate, the feasible one is selected.

These rules allow infeasible solutions to be accepted (which is the case mostly in the first iterations of the algorithm), while favoring solutions that lie closer or inside the feasible region. Obviously, no feasible initial solutions are required in this case.

After updating the best solution, the pheromone update takes place. This procedure is one of the features that distinguish between different ACO variants. In our approach, we combined features from different variants that were found to fit the studied problem better. Specifically, the first step in pheromone update

is the evaporation, where each pheromone value is reduced as follows:

$$\tau_{it}^{[j]}(l) \leftarrow (1 - R_{\text{ev}})\tau_{it}^{[j]}(l), \qquad \text{for all } i, j, t, l,$$

where $R_{\text{ev}} \in (0, 1)$ is the *pheromone evaporation rate* and determines the algorithm's rate of "forgetting" previous states in most ACO variants [12]. After the evaporation, the pheromones are updated again, as follows:

$$\tau_{it}^{[j]}(l) \leftarrow \tau_{it}^{[j]}(l) + \Delta_{it}^{[j]}(l), \qquad \text{for all } i, j, t, l,$$

where:

$$\Delta_{it}^{[j]}(l) = \begin{cases} \frac{\Delta\tau}{K}, & \text{if the best solution contains } p_{it}^{[j]} = l, \\ 0, & \text{otherwise,} \end{cases} \tag{28}$$

and $\Delta\tau$ is a fixed quantity that, in combination with R_{ev}, determines how strongly the algorithm promotes the best detected solution. Similarly to the $\mathcal{MM}$AS approach [19], we considered only the best solution to add pheromone instead of all ants. In the same spirit, we considered a lower bound for the pheromone values, $\tau_{it}^{[j]}(l) \geqslant 10^{-3}$, in order to avoid the complete exclusion of specific values of $p_{it}^{[j]}$ during the search. Although, an upper bound was not found to benefit the algorithm.

In addition, preliminary experiments revealed that the algorithm could stagnate if a very good solution was prematurely detected. This weakness was addressed by adopting a restarting mechanism as in $\mathcal{MM}$AS [19]. Thus, every r_{re} iterations all pheromones are randomly re-initialized, offering the necessary perturbation to unstuck the algorithm from possible local minimizers. The algorithm is terminated as soon as a user-defined stopping criterion is satisfied.

The initialization of the pheromones follows a special yet reasonable scheme. In the initial step of the algorithm, decisions shall be unbiased with respect to the selected components values. Hence, we shall assign equal probability of visiting or not a supplier. Also, if the decision is to visit a supplier, then its position in the vehicle's visiting order shall be selected with equal probability among the different states. For this reason, we assign the following initial selection probabilities of the components values:

$$\rho_{it}^{[j]}(l) = \begin{cases} 0.5, & \text{for } l = 0, \\ 0.5/N, & \text{for } l = 1, 2, \ldots, N. \end{cases}$$

Thus, the state $l = 0$ (i.e., supplier i is not visited by vehicle j at time t) is selected with probability 0.5, while the rest are selected with equal probability among them.

3.4 Penalty Function

Let $f(P)$ be the objective function under minimization, which is defined in Eq. (2), with the candidate solution vector P being defined as in Eq. (27).

Table 1. The considered problem instances and the dimensions of the corresponding optimization problems

Suppliers (N)	Vehicles (M)	Time periods (T)	Problem dimension
4	5	5	100
4	8	10	320
6	6	5	180
6	8	10	480
6	12	10	720
8	6	5	240
8	8	5	320
8	10	10	800
10	6	5	300
10	10	10	1000

All parameters of the model are determined as described in Section 3.2. As mentioned in previous sections, infeasible solutions are penalized by using a penalty function that takes into account the number and the degree of violations.

Thus, if $\mathrm{VC}(P)$ denotes the set of violated constraints for the candidate solution P, then the penalized objective function becomes:

$$PF(P) = f(P) + \sum_{i \in \mathrm{VC}(P)} |\mathrm{MV}(i)|, \tag{29}$$

where $\mathrm{MV}(i)$ is the magnitude of violation for constraint i. A constraint is considered as violated if the magnitude of violation exceeds a small, fixed tolerance value, $\varepsilon_{\mathrm{tol}} > 0$.

Moreover, we shall notice that the constraints (C12)-(C21), defined in Eqs. (15)-(24), are *de facto* satisfied by the solution representation that is used in our approach. Thus, there is no need to include them in the penalty function. All other constraints were equally considered and penalized since a solution shall satisfy all of them in order to be useful for the considered IRP model.

4 Experimental Results

The proposed algorithm was applied on a set of test problems with various numbers of suppliers, vehicles, and time periods. More specifically, the problem instances reported in Table 1 were derived from the data set[1] provided in [15]. Each problem instance was considered along with the parameter setting reported in Table 2. Notice that, in contrast to the model studied in [17] using the same

[1] Available at `http://www.mie.utoronto.ca/labs/ilr/IRP/`

Table 2. Problem and algorithm parameters

Parameter	Description	Value(s)
C	Vehicle capacity	10
F	Fixed vehicle cost per trip	20
V	Travel cost per distance unit	1
s_i	Backlogging cost (with respect to h_i)	$3 \times h_i$
K	Number of ants	5, 10, 15
$E_{\max}$	Maximum function evaluations	60×10^6
ε_{tol}	Constraints violation tolerance	10^{-8}
R_{ev}	Pheromone evaporation rate	10^{-3}
$\Delta\tau$	Pheromone increment	$10^{-2}/N$
r_{re}	Evaluations for pheromone restart	5×10^6

test problems, we considered a limited flee size instead of unlimited. This makes the problem significantly harder even in its small instances.

We followed the experimental setting in [17] whenever possible, except the number of experiments, which was set to 20 per problem instance, instead of 10 for the GA-based approach in [17]. The best solution detected by the proposed algorithm was recorded for each problem and compared to the solution provided by the CPLEX software (Ver. 12.2) for our model within 4 hours of execution.

Following the basic analysis in similar papers [17], the algorithm's performance was assessed in basis of the gap between the best obtained solution and the CPLEX solution. The gap is computed as follows:

$$\text{gap} = \frac{\text{solution value} - \text{CPLEX solution value}}{\text{CPLEX solution value}} \times 100\%.$$

Also, the required percentage fraction of the maximum computational budget $E_{\max}$ for the detection of each solution, was computed as follows:

$$E_{\text{FR}} = \frac{\text{required number of function evaluations}}{E_{\max}} \times 100\%,$$

and it was used as a measure of the algorithm's running-time requirements. Finally, each experiment was replicated for different numbers of ants, namely $K = 5$, 10, and 15. The experiments were conducted on a desktop computer with Intel® i7 processor and 8GB RAM. The running time for the obtained solutions was also reported.

The most successful setting proved to be that of $K = 5$, since it was able to achieve the same solution gaps with the rest, but with smaller demand in resources. The corresponding results are reported in Table 3. The results are promising, since the obtained gaps from optimality were kept in lower values, exhibiting a reasonable increasing trend with the problem's dimension. This is a well-known effect in approximation algorithms and it is tightly related to the *curse of dimensionality* [18].

Table 3. The obtained results for $K = 5$ ants.

Problem			Gap	E_{FR}	Time
N	M	T	(%)	(%)	(sec)
4	5	5	0.00	14.9	67.8
4	8	10	0.96	27.1	1218.3
6	6	5	1.84	3.1	1329.4
6	8	10	4.34	14.8	3684.6
6	12	10	5.62	17.9	3688.3
8	6	5	4.87	7.4	1842.3
8	8	5	6.64	20.6	3317.8
8	10	10	8.63	60.1	5035.3
10	6	5	4.00	45.9	2778.5
10	10	10	10.05	54.9	7346.6

Moreover, the required computational budget was slightly higher than half of the available one for the hardest cases, as we can see in the last column of the table. However, it was not proportional to the corresponding problem's dimension. This can be primarily attributed to the stochastic nature of the algorithm, as well as to the unique structure of each problem instance.

The scaling properties of the algorithm with respect to K were assessed on the basis of the *gap growing factor* (ggf), which is defined as follows:

$$\mathrm{ggf}_{K_1 \to K_2} = \frac{\text{Solution gap for } K = K_2}{\text{Solution gap for } K = K_1}.$$

The obtained values of $\mathrm{ggf}_{5\to10}$ and $\mathrm{ggf}_{5\to15}$, are illustrated in Fig. 1. Notice that there is no data for the problem instance 4-5-5 due to the zero values of the gaps reported in Table 3. In almost all cases, we observe an increasing tendency of the gap growing factor.

In a first thought, the later may seem to be a counterintuitive evidence, since the addition of ants would be expected to boost the algorithm's search capability and performance. However, it is a straightforward consequence of the pheromone update scheme. More specifically, as described in Section 3.3, only the overall best ant updates the pheromones at each iteration. Also, the amount of update is inversely proportional to the number of ants as defined in Eq. (28). Hence, larger number of ants implies smaller pheromone increments and, consequently, higher number of iterations for the domination of the best ant's components against the rest.

Therefore, under the same computational budget, smaller values of K shall be expected to converge faster to the optimal solution than the higher ones. Nevertheless, we can observe that the rate of growth of the gap is sub-linearly associated with the corresponding growth in K, since it exceeds it only in 2 cases.

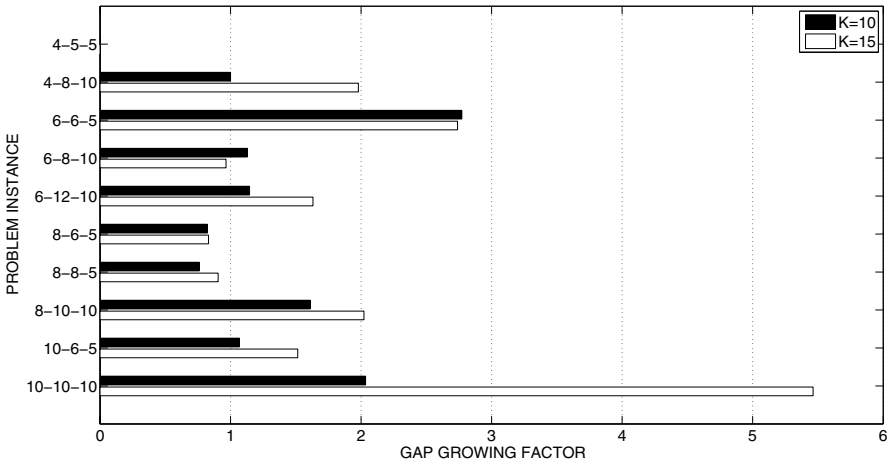

Fig. 1. Scaling properties of the algorithm with respect to the gap growing factor for $K = 10$ ($\text{ggf}_{5\to10}$) and $K = 15$ ($\text{ggf}_{5\to15}$) ants

5 Conclusions

We proposed a unified framework for solving IRPs as VR problems through an ant-based algorithm. We considered a model where a fleet of capacitated homogeneous vehicles transport different products from multiple suppliers to a retailer to meet the demand in each period over a finite planning horizon, while shortages are allowed and unsatisfied demand is backlogged.

The corresponding VR problem was solved with an ant-based optimization approach. Experiments on different test problems offered preliminary insight regarding the algorithm's potential. The solution gaps between the algorithm and CPLEX solutions were kept in reasonably low values, while offering perspective for further improvement by proper parameter tuning.

Also, the stochastic nature of the algorithm as well as its tolerance to new operators and representations, allows the inclusion of problem-based specialized operations. Finally, the algorithm tackles the problem without the necessity for breaking it into sub-problems and solving each one separately. Naturally, additional work is required to fully reveal the strengths and weaknesses of the algorithm. Parallel cooperative schemes could be beneficial for reducing the time complexity, and this is the primary goal of our ongoing efforts.

Acknowledgement. This research has been co-financed by the European Union (European Social Fund - ESF) and Greek national funds through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF) - Research Funding Program: ARCHIMEDES III. Investing in knowledge society through the European Social Fund.

References

1. Abdelmaguid, T.F., Dessouky, M.M.: A genetic algorithm approach to the integrated inventory–distribution problem. International Journal of Production Research 44, 4445–4464 (2006)
2. Abdelmaguid, T.F., Dessouky, M.M., Ordonez, F.: Heuristic approaches for the inventory–routing problem with backlogging. Computers and Industrial Engineering 56, 1519–1534 (2009)
3. Andersson, H., Hoff, A., Christiansen, M., Hasle, G., Lokketangen, A.: Industrial aspects and literature survey: Combined inventory management and routing. Computers & Operations Research 37, 1515–1536 (2010)
4. Bäck, T.: Evolutionary Algorithms in Theory and Practice. Oxford University Press, New York (1996)
5. Bonabeau, E., Dorigo, M., Théraulaz, G.: Swarm Intelligence: From Natural to Artificial Systems. Oxford University Press, New York (1999)
6. Campbell, A.M., Clarke, L., Kleywegt, A.J., Savelsbergh, M.W.P.: The inventory routing problem. In: Crainic, T., Laporte, G. (eds.) Fleet Management and Logistics, pp. 95–113. Kluwer Academic Publishers, Boston (1998)
7. Campbell, A.M., Savelsbergh, M.W.P.: A decomposition approach for the inventory–routing problem. Transportation Science 38(4), 488–502 (2004)
8. Chan, L., Federgruen, A., Simchi-Levi, D.: Probabilistic analyses and practical algorithms for inventory–routing models. Operations Research 46(1), 96–106 (1998)
9. Chien, T.W., Balakrishnan, A., Wong, R.T.: An integrated inventory allocation and vehicle routing problem. Transport Science 23, 67–76 (1989)
10. Dorigo, M., Di Caro, G.: The ant colony optimization meta–heuristic. In: Corne, D., Dorigo, M., Glover, F. (eds.) New Ideas in Optimization, pp. 11–32. McGraw-Hill (1999)
11. Dorigo, M., Gambardella, L.M.: Ant colony system: A cooperative learning approach to the traveling salesman problem. IEEE Transactions on Evolutionary Computation 1(1), 53–66 (1997)
12. Dorigo, M., Stützle, T.: Ant Colony Optimization. MIT Press (2004)
13. Federgruen, A., Zipkin, P.: A combined vehicle routing and inventory allocation problem. Operations Research 32(5), 1019–1036 (1984)
14. Hvattum, L.M., Lokketangen, A.: Using scenario trees and progressive hedging for stochastic inventory routing problems. Journal of Heuristics 15, 527–557 (2009)
15. Lee, C.-H., Bozer, Y.A., White III, C.C.: A heuristic approach and properties of optimal solutions to the dynamic inventory routing problem. Working Paper (2003)
16. Moin, N.H., Salhi, S.: Inventory routing problems: a logistical overview. Journal of the Operational Research Society 58, 1185–1194 (2007)
17. Moin, N.H., Salhi, S., Aziz, N.A.B.: An efficient hybrid genetic algorithm for the multi–product multi–period inventory routing problem. International Journal of Production Economics 133, 334–343 (2011)
18. Powell, W.B.: Approximate Dynamic Programming: Solving the Curses of Dimensionality. Wiley (2007)
19. Stützle, T., Hoos, H.H.: Max min ant system. Future Generation Computer Systems 16, 889–914 (2000)