

The One-Commodity Traveling Salesman Problem with Selective Pickup and Delivery: an Ant Colony Approach

Rafael Falcon, Xu Li, Amiya Nayak and Ivan Stojmenovic

Abstract—We introduce a novel combinatorial optimization problem: the one-commodity traveling salesman problem with selective pickup and delivery (1-TSP-SELPD), characterized by the fact that the demand of any delivery customer can be met by a relatively large number of pickup customers. While all delivery spots are to be visited, only profitable pickup locations will be included in the tour so as to minimize its cost. The motivation for 1-TSP-SELPD stems from the carrier-based coverage repair problem in wireless sensor and robot networks, wherein a mobile robot replaces damaged sensors with spare ones. The ant colony optimization (ACO) meta-heuristic elegantly solves this problem within reasonable time and space constraints. Six ACO heuristic functions are put forward and a recently proposed exploration strategy is exploited to accelerate convergence in dense networks. Results gathered from extensive simulations confirm that our ACO-based model outperforms existing competitive approaches.

I. INTRODUCTION

Wireless sensor networks (WSNs) are collections of autonomous sensing devices that are deployed into physical environments like hospitals, forests, highways, etc. to cooperatively monitor the region. They belong to the domain of wireless ad hoc networks [27] and face many design and realization challenges, e.g., [5], [16], [24], [25]. Recently, an emergent research area coined as “wireless sensor and robot networks” (WSRNs) [20] has stemmed from the integration between WSNs and multi-robot systems. Any WSRN consists of an ensemble of sensor and robot nodes that communicate via wireless links to perform distributed sensing and actuation tasks. While sensors are highly constrained devices (i.e., they possess limited computing power, battery, memory, transmission range, etc.), robots are resource-rich, usually mobile and meant to assist, maintain and optimize sensor networks. For example, they may perform intelligent movement for data collection [3] or sensor placement [9].

We are concerned with the problem of *carrier-based coverage repair* in WSRNs, which can be defined as follows: consider a network of static sensors already deployed in some area of interest. Because of the abundance of cheap sensors scattered, not all of them are actually needed to provide reliable area coverage. Thus sensors may follow a scheduling algorithm e.g., [10] in order to decide which sensors will go to “sleep” mode and which ones will be responsible for monitoring the region. The former are called “passive” nodes and their aim is to save energy (ultimately prolonging the network lifetime) whereas the latter are referred to as

“active” nodes. Unfortunately, *network coverage* (i.e., the total area monitored by the network) will be eventually degraded because of active node failures (e.g., battery depletion, unexpected events, hazards, etc.), thus creating sensing holes in the area. Periodically, every passive sensor reports its location to a base station (by multi-hop communications) and active sensors report the coordinates of any adjacent sensing hole to the base station too. Assume that a single mobile robot (which is able to carry a limited number of sensors) is located at the base station. Its goal is to collect passive sensors all over the field and drop them into the sensing holes. This task is to be repeated every so often. We want to compute an ‘optimal’ robot trajectory so that network coverage can be repaired by replacing all damaged sensors with spare (passive) ones. By ‘optimal’ we mean a minimum-cost tour (trajectory), where the *tour cost* is here defined as the total distance traveled and the *tour length* is the number of nodes in the tour, excluding the base station. The periodicity with which the robot tends to damaged sensors depends on operational factors like the average durability of the deployed sensors, the power expenditures for robot mobility and the expected QoS of the sensor network, among others, which lays this topic beyond the scope of this paper.

The above scenario can be formulated as a combinatorial optimization problem. We represent the sensor network by a complete graph $G = (V, E)$ where $V = \{v_1, v_2, \dots, v_n\}$ is the vertex set and $E = \{e_{ij}\}$ is the edge set, with e_{ij} being the edge between sensors i and j , $\forall i, j \in \{1, 2, \dots, n\}$. Elements in V are either passive sensors (pickup customers) or sensing holes (delivery customers). Each customer has an associated unitary demand q_i (1 for passive sensors and -1 for sensing holes). Moreover, the base station is denoted as v_1 with $q_1 = 0$. There is a unique commodity (sensors) to be transported by the robot from one place to another. The robot can carry at most Q sensors and leaves the base station with an initial cargo Q_0 ($0 \leq Q_0 \leq Q$). We want to find a minimal-cost feasible tour φ_{min} that starts and ends at the base station. The cost function for any tour φ can be computed as $\sum_{e_{ij} \in \varphi} d(e_{ij})$ where $d(e_{ij})$ stands for the Euclidean length of e_{ij} . Without ambiguity, we write $d(e_{ij})$ as d_{ij} for simplify. A tour is said to be feasible if it has no repeated nodes (other than the base station as its first and last element), repairs all sensing holes (i.e., drops a passive sensor at the location of each damaged device) and never violates the robot’s capacity constraint. Figure 1 illustrates a feasible tour for a carrier-based coverage repair scenario.

A similar problem in the literature is that of “*the one-*

All authors are affiliated with SITE, University of Ottawa, Canada; email: {rfalc032, xuli, anayak, ivan}@site.uottawa.ca.

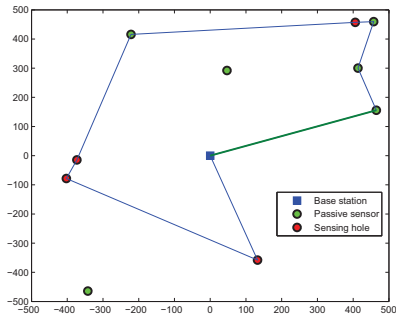


Fig. 1. A feasible tour for the carrier-based coverage repair problem with $Q_0 = 0$ and $Q = 3$. The green line symbolizes the departing direction.

commodity pickup-and-delivery traveling salesman problem” (1-PDTSP) [12]. But nevertheless, unlike 1-PDTSP, our tour is not a Hamiltonian cycle, i.e., all graph vertices need NOT to be visited. While all delivery demands are to be met, only profitable pickup spots will be included in the tour so as to minimize its cost, as per the assumption that the passive sensors throughout the field significantly outnumber the sensing holes. This distinctive feature gives rise to a novel combinatorial optimization problem, which we coin as “*the one-commodity traveling salesman problem with selective pickup and delivery*” (1-TSP-SELPD), in which the length of any feasible tour will be exactly $tl = 2 \cdot |V_{SH}| - Q_0$ nodes, where $V_{SH} = \{v_i \in V : q_i = -1\} \forall i \in \{1, 2, \dots, n\}$. It is easy to realize that 1-TSP-SELPD is a generalization of 1-PDTSP and as such, it is *NP*-hard too. Further, the quest for feasible tours in both problems is *NP*-complete [12].

We model the carrier-based coverage repair problem as a 1-TSP-SELPD with unitary pickup and delivery and apply the ant colony optimization (ACO) meta-heuristic [7] to solve it. In particular, we devise six heuristic functions (see Sec. IV) that embrace diverse optimization criteria to guide the search undertaken by the Max-Min Ant System (MMAS) algorithm [26]. To make our MMAS-based approach more robust and scalable, we exploit a recently proposed exploration strategy [22] which finds good-quality solutions in about 40% - 60% the running time required by conventional MMAS. Not only we are formally introducing and solving 1-TSP-SELPD via artificial ant colonies, but our approach can be slightly modified to become the first ACO application, to the best of our knowledge, to 1-PDTSP as well.

The paper is structured as follows. Section II is related work. An overview of ACO techniques is presented in Sec. III. We propose the ACO-based model for solving 1-TSP-SELPD in Sec. IV. Section V describes the implemented two-step exploration strategy to overcome scalability problems and guarantee faster convergence. The empirical results outlined in Sec. VI precede the final remarks in Sec. VII.

II. RELATED WORK

The traveling salesman problem (TSP) [23] is a paradigmatic *NP*-hard combinatorial optimization problem. It has been drawing significant research attention, probably because it arises practically in many areas, is very easy to understand

and serves as a standard test bed for further algorithmic developments. This is especially true for the ACO research community. Nearly all ACO algorithms, ranging from the early Ant System (AS) to the more elaborate Ant Colony System (ACS), Max-Min Ant System (MMAS) or Rank-Based Ant System (AS-rank) have been tried on several TSP instances [6][7], mainly because the search metaphor of the artificial ants is easily adapted to the problem.

Among the many variations of standard TSP, the one closest to our problem is 1-PDTSP that was formally introduced in [12]. Its formulation differs from the general TSP with pickups and deliveries (TSPPD) [11] in that the commodity gathered at a pickup customer can be used to furnish a delivery customer, as opposed to TSPPD where all commodities have to depart from (end up at) the depot. Anily and Bramel studied the unitary 1-PDTSP [2] in the context of inventory repositioning whereas Chalasani and Motwani [4] called it “*Q-delivery TSP*”. Hernández-Pérez et al proposed a branch-and-cut algorithm for handling small 1-PDTSP instances [14] which was subsequently enriched by adding local search and incomplete optimization techniques [15]. This latter approach was recently outperformed by a hybridization of greedy randomized adaptive search procedure (GRASP) and variable neighborhood descent (VND) [13]. Simulated annealing also proved successful in targeting 1-PDTSP, as reported in [19].

However, none of the aforementioned techniques fall under the umbrella of population-based meta-heuristic algorithms. This sort of methods (among which ant colony optimization has earned a sound reputation) have become very popular among the optimization community for their proved ability to overcome local optima through a parallel exploration of the search space and the use of social communication mechanisms to drive the population toward promising regions.

We therefore solve the carrier-based coverage repair problem (modeled as a unitary 1-TSP-SELPD) through an ACO approach. The rationale behind the selection of artificial ant colonies as the underlying optimization engine lies in: (1) the increasing empirical evidence [7] that ACO together with local search techniques is among the best-ranked algorithms for working out a wide array of difficult combinatorial optimization problems; (2) the incremental construction of the solution, which allows us to manage the problem constraints in a straightforward fashion [1] as opposed to building the entire tour and discarding it afterwards; (3) the late research findings on the superiority of ACO parallel implementations over peer meta-heuristics for TSP posed in [17] and on enhanced exploration strategies in [22], which are of great importance for scalability purposes.

III. ANT COLONY OPTIMIZATION

In this section, we briefly describe the main ideas behind the ACO meta-heuristic and highlight the distinctive features of the MMAS algorithm.

A. Overview

Ant colony optimization is a stochastic search technique to find shortest paths in discrete (graph-representable) optimiza-

tion problems. It owes its inspiration to the foraging behavior of natural ants. The high degree of coordination among simple individual agents like natural ants makes it possible to globally attain good suboptimal solutions (i.e., short paths from the nest to the food source) at the colony level. Communication is performed indirectly, i.e., mediated by the environment (*stigmergy*) through the deposit of a chemical substance named pheromone. The higher the intensity of the pheromone deposited on an edge, the more likely it is to be selected by future ants as part of their journey.

Ants are initialized in different graph nodes, either at random or following a specific rule. Their purpose is to concurrently and incrementally construct tours (i.e., candidate solutions). The next node in the tour is selected in a stochastic manner according to a probabilistic transition rule from the current node to every other node in each ant's neighborhood (set of nodes to which the ant can move from the current node). Such rule is posed as a function of the pheromone trails already present in the graph and the problem-dependent heuristic desirability of going from one node to another. In the Ant System (AS), the first ACO algorithm [6], the transition rule takes the form:

$$p_{ij}^k(t) = \frac{[\tau_{ij}(t)]^\alpha \cdot [\eta_{ij}]^\beta}{\sum_{l \in N_i^k} [\tau_{il}(t)]^\alpha \cdot [\eta_{il}]^\beta}, \quad (1)$$

where p_{ij}^k denotes the probability of the k -th ant (currently at node i) selecting j as the next node in the tour among all candidates in its neighborhood N_i^k , $\tau_{ij}(t)$ is the amount of pheromone deposited on e_{ij} at iteration t , η_{ij} is the heuristic desirability of traversing e_{ij} whereas α and β are two sensitivity coefficients associated with the pheromone (collective knowledge) and heuristic (problem-specific knowledge) components of the ACO model, respectively.

The pheromone trails are dynamically altered by the ants during the algorithm's execution, in contrast to the heuristic information that remains unchanged. An ACO iteration comprises ant initialization, solution construction and pheromone update. Multiple iterations take place until some stop condition is met. Defining when, how and by whom the pheromone trails are updated is a pivotal issue for any ACO implementation. For instance, in the elitist version of AS, the ant yielding the best solution (i.e., shortest tour) at the end of every iteration is the only one entitled to perform the pheromone update as follows:

$$\tau_{ij}(t+1) = \rho \cdot \tau_{ij}(t) + \Delta\tau_{ij}, \quad (2)$$

where $\tau_{ij}(t+1)$ is the amount of pheromone on e_{ij} at the next iteration, $0 \leq \rho \leq 1$ is the pheromone persistence rate (implying that some amount will evaporate as a mechanism to prevent stagnation in local optima) and $\Delta\tau_{ij}$ is the change in pheromone concentration on e_{ij} . In this paper, we modify the pheromone values as shown below:

$$\Delta\tau_{ij} = \begin{cases} \frac{R}{f(\varphi)}, & \text{if } e_{ij} \in \varphi \\ 0, & \text{otherwise,} \end{cases} \quad (3)$$

where φ is the best tour produced in the current iteration, $f(\varphi)$ its associated cost, R is called "reward factor" and set

to the cost of the best solution found in the first iteration. The shorter the tour computed in subsequent iterations, the larger the amount of pheromone added to its edges. This update rule helps drive the search towards good solutions.

The termination of ACO algorithms is usually envisioned in terms of a maximum number of iterations carried out, although reaching a threshold on runtime, number of generated tours or solution quality may also become a stop criterion.

B. Max-Min Ant System (MMAS)

Early ACO versions exhibit a degraded performance for problems of large dimensions owing to the stagnation phenomenon which arises from the long-term accumulation of pheromone on the best edges (especially in elitist approaches, i.e., those which reward the best solutions alone). This undesirable effect strongly biases the ants' selection and narrows their overall exploration capabilities. Premature convergence to suboptimal solutions has been dealt with the Max-Min Ant System (MMAS) introduced by Stützle and Hoos [26]. MMAS can be regarded as a direct successor of AS (actually, both use the same transition rule (1)) but introduces innovative modifications to the AS machinery.

MMAS is an elitist approach, i.e., it highly exploits the most promising (either iteration-best or global-best) tour discovered, whose edges receive the highest pheromone concentrations. Yet MMAS strongly encourages the exploration of the entire space by placing a cap on the amount of pheromone on each edge such that $\tau_{ij} \in [\tau_{min}; \tau_{max}]$. The obvious effect is that there will also be a cap on the probability of an edge being selected as next node. Pheromone trails are initialized to τ_{max} which increases the exploration of tours at the beginning of the search. Furthermore, the trails are reinitialized each time the system undergoes stagnation or lack of improvement of the best tour generated after a certain number of consecutive iterations. A remarkable MMAS feature is the dynamic nature of the pheromone bounds, which change every time a global best solution is encountered. In our implementation, the pheromone bounds are computed according to (4) and (5).

$$\tau_{max} = \frac{1}{\rho} \cdot \frac{R}{f(\varphi_{opt})}, \quad (4)$$

$$\tau_{min} = \frac{\tau_{max} \cdot (1 - \sqrt[t]{p_{best}})}{J_{avg} \cdot \sqrt[t]{p_{best}}}, \quad (5)$$

where φ_{opt} is the best-ever-found solution, p_{best} stands for the probability that the best solution is constructed again, J_{avg} is the average number of options (decisions) available at the graph nodes and tl the tour length. MMAS behavior was thoroughly examined for several TSP instances [26] and it was proved to excel competitive ACO algorithms.

IV. ACO FOR CARRIER-BASED COVERAGE REPAIR

The application of ACO to real-life problems is governed by the following design guidelines:

- 1) Represent the problem as a graph whose vertices are the problem states and whose edges are the transitions

between the states. The graph will be “walked” by the ants to iteratively build the solutions.

- 2) Define the meaning of the pheromone trails τ .
- 3) Define the meaning of the heuristic preference η (a.k.a. visibility function).
- 4) Use the ants’ local memory as well as the heuristic function to meet the problem’s constraints, if possible.
- 5) Choose the ACO implementation that best fits the problem at hand.
- 6) Combine ACO with local search methods to improve the quality of the solutions found.
- 7) Tune the algorithm parameters.

Recall that carrier-based coverage repair is modeled here as a unitary 1-TSP-SELPD, i.e., every delivery or pickup quantity is exactly one unit. In light of that, our graph-based representation for 1-TSP-SELPD was already outlined in Sec. I. Guidelines 2 and 3 are heavily problem-dependent and have a crucial bearing on the algorithm’s ultimate performance. When we are dealing with ordering problems like 1-TSP-SELPD (i.e., the order in which the nodes are visited does matter), the pheromone trails are associated with the graph edges in order to reward good visiting sequences. The ACO heuristic function traditionally used for TSP is $\eta_{ij} = \frac{1}{w(e_{ij})}$ with $w(e_{ij})$ being the weight of the edge connecting nodes i and j . For TSP, the weight is the Euclidean distance between them. Unfortunately, in 1-TSP-SELPD the desirability of e_{ij} is not only a function of its weight, but it has to take into account the current cargo of the search agent (ant) in order to avoid infeasible solutions (e.g., a fully-laden ant must not select any passive sensor as next node in the tour). Little bit more elaborate heuristic functions for our problem are to be devised shortly.

As explained in Sec. II, one of the reasons behind the election of ACO meta-heuristic to solve 1-TSP-SELPD is that constraint satisfaction is carried out in an elegant way by exploiting the incremental tour construction feature of ACO models. Most meta-heuristic approaches (e.g., evolutionary computation, particle swarm optimization, simulated annealing, tabu search, etc. [7]) will rather generate an entire solution and then check its feasibility, either discarding it completely or modifying it in an attempt to reach feasibility. With ACO, only feasible nodes are added to the tour.

For example, the restrictions that all sensing holes have to be visited and that each node in the tour must not be repeated can be enforced by storing in the local memory of each ant two lists: ‘*mustVisit*’ and ‘*tour*’, the former containing the still-unvisited sensing holes and the latter recording the previously visited vertices. An ant is said to have completed a solution when the former list is empty. So, the feasible neighborhood of ant k at decision point i will be $N_i^k = V - (\text{tour} \cup \{v_1\})$. As such, we also guarantee that the robot departs from (arrives to) the base station. Notice that we are purposefully not excluding here those nodes which make the tour infeasible due to the capacity violation constraint. They could also be added to an “infeasible” list, but in order to prevent it from growing too much, we would

rather rule them out mathematically by the computation of the heuristic function, as shown in Sec. IV-B.

The selection of the ACO model that best targets the problem at hand is also a crucial design step. We have no knowledge of any ant algorithm being applied to 1-PDTSP, but we have abundant empirical evidence that MMAS outperformed rival ant approaches when tested against a wide assortment of TSP instances, outperforming AS and its improved versions. This outcome is due to the strong exploitation of the best solutions discovered and the effective search space exploration mechanism enforced by the dynamic pheromone bounds, which make MMAS a very appealing choice for our current research efforts.

Although still some improvements in the quality of the tours can be attained by adding local search procedures and dynamic tuning of the MMAS parameters (as stated in guidelines 6 and 7), for the sake of simplicity we will not incorporate any of these mechanisms into our ACO-based proposal for 1-TSP-SELPD.

A. Initialization

At the beginning of every iteration, ants must be initialized on different graph nodes from which they will start adding components to their tours. This step is done entirely at random for the TSP. Yet in our problem, we must place an ant at a feasible node depending on its initial cargo Q_0 . The feasible initial neighborhood N_k for the k -th ant, from which a node will be randomly chosen as its starting point, is defined as

$$N_k = \begin{cases} V - V_{SH} - \{v_1\}, & \text{if } Q_0 = 0 \\ V_{SH}, & \text{if } Q_0 = Q_{max} \\ V - \{v_1\}, & \text{otherwise,} \end{cases}$$

where n_F is the number of remaining sensing holes in the field (all, at the beginning) and $Q_{max} = \min(n_F, Q)$.

B. Heuristic Rules

The heuristic component in any ant algorithm describes the unchanging, problem-specific knowledge, as opposed to the pheromone trails which convey the dynamic, collective knowledge achieved by means of self-organization. In 1-TSP-SELPD, the heuristic desirability η_{ij} of transitioning from the current problem state i to any neighbor state j is posed as a function of their distance, the type of the current node (pickup or delivery) and the current cargo of the search agent. Several visibility rules are envisioned and presented next, each trying to capture a desirable facet of the overall tour construction.

1) “*DROP-OFF FIRST*” *RULE*: If robot cargo is empty, then choose the closest passive sensor. Otherwise, prioritize the drop-off operation by selecting the nearest sensing hole as in (6).

$$\eta_{ij} = (-1)^{\text{sgn}(Q_k)} \times \frac{q_i}{d_{ij}}, \quad (6)$$

where $\text{sgn}(\cdot)$ is the sign function, Q_k is the current cargo of the k -th ant, q_i is the type of the current node (either passive sensor or sensing hole, see Sec. I) and d_{ij} is the Euclidean distance between nodes i and j .

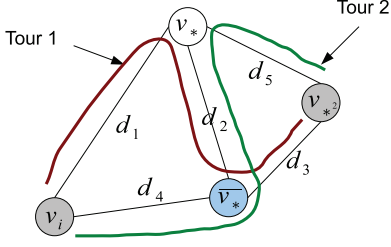


Fig. 2. The LOOK AHEAD heuristic rule

2) “*PICKUP FIRST*” RULE: If robot is full, drop a sensor at the closest hole. Otherwise, prioritize the pickup operation by choosing the nearest passive sensor as in (7).

$$\eta_{ij} = (-1) \lfloor \frac{Q_k}{Q_{max}} \rfloor \times \frac{q_i}{d_{ij}}, \quad (7)$$

where $\lfloor \cdot \rfloor$ is the floor function. We assume that $Q_k \leq Q_{max}$, i.e., the current cargo must never exceed the number of sensing holes left in the terrain (extra load) or the maximum cargo (capacity violation).

3) “*NEAREST NEIGHBOR*” RULE: Whenever both a drop-off and a pickup are possible, always choose the nearest spot according to (8).

$$\eta_{ij} = (-1) \text{sgn}(Q_{max} - Q_k) + 1 \times \frac{q_i \lfloor 1 - \text{frac}(\frac{Q_k}{Q_{max}}) \rfloor}{d_{ij}}, \quad (8)$$

where $\text{frac}(\cdot)$ returns the fractional part of its argument. Expression (8) boils down to $-\frac{q_i}{d_{ij}}$ when the robot is fully loaded, to $\frac{q_i}{d_{ij}}$ when it is empty and to $\frac{1}{d_{ij}}$ when both operations are possible, thus always leading to a suitable decision making.

4) “*LOOK AHEAD*” RULE: This rule is an extension of the previous one but from a “look-ahead” perspective. Let v_i be the current node and v_H and v_P the nearest sensing hole and passive sensor to v_i , respectively. Let us denote by v_* the location (either pickup or drop-off) that the robot will go to by the NEAREST NEIGHBOR rule. Let \bar{v}_* be v_P if $v_* = v_H$, and v_H otherwise. Finally, denote by v_{*2} the location that the robot will advance to, by the NEAREST NEIGHBOR rule, after it arrives at v_* , as depicted in Fig. 2.

We are now ready to enunciate the look-ahead rule as follows. If $v_{*2} = \bar{v}_*$, the ant will simply follow the NEAREST NEIGHBOR rule for next node selection. Otherwise, it will compute two tours for v_* , \bar{v}_* and v_{*2} , $\text{tour}_1 = v_i \rightarrow v_* \rightarrow \bar{v}_* \rightarrow v_{*2}$ and $\text{tour}_2 = v_i \rightarrow \bar{v}_* \rightarrow v_* \rightarrow v_{*2}$, with different next node but the same destination v_{*2} . The ant will select the next node that leads to a shorter tour. In case of tie, preference is given to the delivery customer.

The analytical expression corresponding to the look-ahead rule is given below:

$$\eta_{ij} = \text{sgn}(q_j) \text{sgn}(\lfloor \frac{d_5}{d_1 + d_3 - d_4} \rfloor) \times \frac{q_i}{d_{ij}}, \quad (9)$$

where $d_1 = d(v_i, v_*)$, $d_3 = d(\bar{v}_*, v_{*2})$, $d_4 = d(v_i, \bar{v}_*)$ and $d_5 = d(v_*, v_{*2})$. Notice that Eqn. (9) boils down to $\frac{q_i}{d_{ij}}$ when it is worth picking up the passive sensor and to $-\frac{q_i}{d_{ij}}$ otherwise.

5) *MAJORITY RULE*: Choose as next node the most voted one among the set of outcomes provided by all previous heuristic rules. Arbitrarily break ties.

6) *STOCHASTIC RULE*: Randomly select any of the previous rules at each step of the tour construction.

Notice that all heuristic rules can take negative values for those nodes that will make the tour infeasible due to the violation of the capacity constraint. Correspondingly, expression (1) has been slightly modified by dividing by the absolute value of the current denominator (and using only odd values for the heuristic sensitivity coefficient β) in an effort to keep the sign yielded by the heuristic function. This mechanism prunes all such invalid nodes, since negative probabilities are not eligible for the subsequent stochastic roulette wheel selection of the next node in the tour.

V. ADDRESSING SCALABILITY CONCERNS

As the size of the region of interest grows, so it does the number of sensors required for reliably monitoring the area. This fact implies that the increased dimensionality of the 1-TSP-SELDP instances to be dealt with might strike the relative ability of the algorithm to render good solutions within reasonable time and space boundaries unless it scales well to large systems. Scalability is a highly sought-after property in optimization methods.

Fortunately, population-based meta-heuristics share an intrinsic distributed nature that makes them good candidates for fast development of parallel implementations. Parallelization techniques bear two goals in mind: to diversify the search and thus concurrently explore multiple subregions of the space and to speed up the execution time needed for finding good-quality solutions. Recent research findings [17] highlight the superior performance obtained through parallel ACO implementations when tested against challenging TSP instances.

A. Two-Step Ant Colony Optimization (TS-ACO)

Acceleration can be accomplished if we approach the initial search state to the final (goal) state as much as possible. This is the chief thought behind the novel “two-step” exploration strategy for ant algorithms (TS-ACO) [21] [22]. Briefly described, the search process undertaken by the ants is split into two stages. In the first stage, a subset of the entire colony walk throughout the graph in a subset of the total iterations trying to find optimal subtours, i.e., tours with lower dimensionality than the actual tours. The results of the exploration are actively exploited to speed up the algorithm’s convergence, either by utilizing the best subtours found as initial subsolutions for the remaining ants in the second stage or by reusing the corporate knowledge expressed in the pheromone matrix computed during the first stage. The former version is referred to as “solution-driven” (S-TS-ACO); whereas, the latter was baptized as “pheromone-driven” (P-TS-ACO).

A key parameter of the enhanced exploration strategy is the split factor $0 < r < 1$ which governs the allocation of resources for each search phase. For instance, given $r = 0.45$ it means that, during the first stage, 45% of the individuals in

the colony will look for subtours of length 45% of the desired tours in just 45% of the total number of cycles (iterations). In stage 2, the remaining ants will run for the leftover number of cycles trying to find solutions to the original problem.

In S-TS-ACO, the best ns subtours of the first search phase were recorded and now each ant in the second phase is randomly initialized on a graph node from which it will resume the search in order to complete its tour, which was randomly picked from one of the best ns partial solutions. The pheromone matrix is reinitialized. However in the P-TS-ACO version, the ants in the second phase start their search from scratch but rely on the pheromone matrix computed during the previous phase to conduct the search.

Extensive empirical and theoretical studies developed in [21] with TSP and the Quadratic Assignment Problem (QAP) concluded that TS-ACO maintains the quality of the solutions found by ACO yet shortens the computational time by roughly 40% – 60%. We take advantage of this finding to add scalability to our MMAS-based approach.

B. Applying TS-MMAS

Let us notice that, unlike standard TSP in which the tour length is dependent on the number of cities, in our real sensor-network-related scenario, the solution dimensionality is a function of the number of sensing holes reported at the base station (see Sec. I for further details). In light of that, we may split the search by following either of two criteria: the number of visited nodes (tour length, denoted as TS-TL) or the number of repaired sensing holes (TS-SH). Each criterion may prove useful under different topological circumstances.

Being m the colony size, nc the maximum number of cycles and having decided on the values of r and ns , we allocate the resources for the two search stages as follows:

- Number of ants: $m_1 = \lfloor r \cdot m \rfloor$ and $m_2 = m - m_1$
- Number of cycles: $nc_1 = \lfloor r \cdot nc \rfloor$ and $nc_2 = nc - nc_1$
- Solution dimensionality: In the first stage under TS-TL, find subtours of length $tl_1 = \lfloor r \cdot tl \rfloor$. For TS-SH, find subtours that have visited exactly $tl_1 = \lfloor r \cdot |V_{SH}| \rfloor$ sensing holes. In the second stage and irrespective of the split criterion, find tours of length $tl_2 = tl$.

Here m_1 , nc_1 and tl_1 are the resources allocated for the first stage while m_2 , nc_2 and tl_2 are those of the second stage.

VI. EXPERIMENTAL RESULTS

We have conducted a set of experiments to test the feasibility of our ACO-based proposal for 1-TSP-SELPD. Simulations were run in MATLAB with an AMD Athlon 64 X2 Dual Core at 2.6 GHz and 2 GB of memory under Windows Vista. Our benchmark instances (available from [8]) are very similar to those in [14] (for each value of n , the number of nodes in the graph, random 2D coordinates for $n - 1$ customers were generated in the square $[-500, 500]^2$) but all demands are unitary ones $\{-1, 1\}$ and the proportion of sensing holes to passive sensors was set exactly to 1:3. The depot is placed at $(0, 0)$ and has demand $q_1 = 0$. The travel cost d_{ij} was computed as the Euclidean distance between i and j . Each instance file is characterized by the number

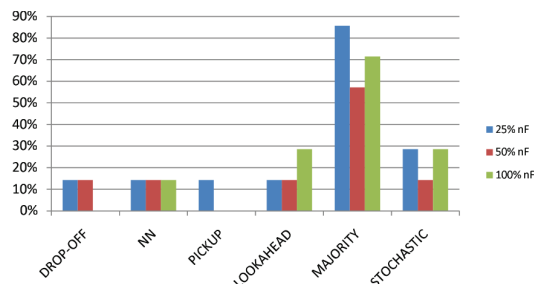


Fig. 3. % of tested files where the best overall solution was found.

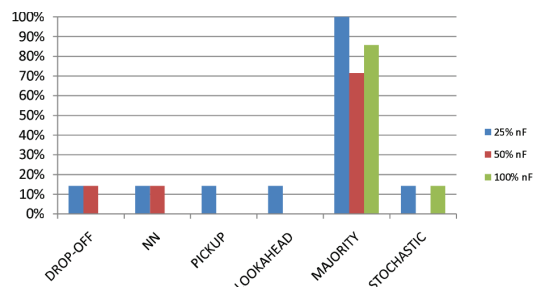


Fig. 4. % of tested files where the best average solution was found.

of sensors n and the robot capacity Q and can be split into two classes: *small* instances with $n \in \{20, 30, 40, 50, 60\}$ and *large* instances with $n \in \{100, 200, 300, 400, 500\}$. As to the robot capacity Q , we have considered three possible values: $Q \in \{25\% \cdot |V_{SH}|, 50\% \cdot |V_{SH}|, 100\% \cdot |V_{SH}|\}$ given that the robot departs from the base station with no extra load.

A. Experiment 1: The ACO Heuristic Rules

Traditionally, ACO researchers have focused on improving the collective behavior (pheromone trails) of the search agents in an attempt to reach good suboptimal solutions. Somehow, the problem-specific knowledge (heuristic component) has been neglected. In our first experiment, we are to assess the bearing of the six heuristic rules portrayed in Sec. IV-B on the overall performance of the ACO metaheuristic. To this end, we have run MMAS for 1,000 iterations over our benchmark instances by using one heuristic rule at a time. The parameter configuration is as follows: $\alpha = 2, \beta = 3, \rho = 0.9, p_{best} = 0.1$. The colony size is 15 for small instances and 30 for large instances. These values were determined empirically. The best and average tour costs over 10 independent executions per heuristic rule are reported.

Figures 3 and 4 (where ‘NN’ indicates the NEAREST NEIGHBOR RULE) reveal an interesting finding: the MAJORITY rule outperforms every other heuristic function regardless of the robot capacity or network density (number of nodes in the graph), both in terms of the best overall solution and the best average solution ever found. In the latter case, MAJORITY always reports the best average solution among peers. This fact sheds light on the need of reaching a consensus among several election criteria at the node level. The holistic view of the ant’s neighborhood seems to be much more promising than its approaching from biased standpoints. Yet the computational time of the MAJORITY rule is the longest one reported and we might not always afford such delay in the calculation of the robot trajectory,

TABLE I
PERFORMANCE OF IMSA ON 1-TSP-SELPD BENCHMARK INSTANCES
WITH MINIMUM CARRIER CAPACITY $Q = 25\%n_F$

n	Q	Best cost	Avg cost	Std dev. cost (%)	Avg time	Std dev. time (%)
20	1	2579.59	2785.23	5.59	20.62	7.86
30	2	3572.71	3842.03	4.38	25.07	6.78
40	3	4343.67	4752.25	6.43	26.77	1.42
50	3	5211.06	5747.04	5.23	33.68	8.97
60	4	6432.87	7178.91	6.22	34.43	2.03
100	6	10629.87	11637.17	5.05	98.02	0.40
200	13	23125.5	24302.86	4.25	160.89	4.92
300	19	37865.17	40627.04	4.45	222.16	5.26
400	25	54204.18	56362.71	3.33	331.51	0.09
500	31	65559.9	70719.13	3.57	397.35	4.40

especially when immediate coverage repair is demanded.

Fortunately, in such cases we may opt to apply the STOCHASTIC rule instead. Although far off the encouraging performance of MAJORITY, a stepwise randomization in the heuristic decision-making process has proved more competitive than conventional rules such as the NEAREST NEIGHBOR. The occasional application of MAJORITY within the stochastic selection yields profitable results.

Overall, LOOK AHEAD is more effective than heuristic rules 1 – 3 in Sec. IV-B, with a better demeanor in scenarios where the carrier capacity is not overly restricted. Another observation is that always prioritizing some activity (either drop-off or pickup) does not lead to good results, as displayed in the previous charts. This is also consistent with the finding in [15] that joining customers of different type increases the probability of finding a good solution for 1-PDTSP, which is a particular case of 1-TSP-SELPD.

B. Experiment 2: Performance Comparison

We empirically compare our ACO-based proposal with a recently enhanced version of the simulated annealing metaheuristic named IMSA [19]. The approach in [13] was not considered for benchmarking purposes given its heavy reliance on local search methods to improve the discovered solution. Because local search mechanisms were not contemplated in our implementation, there is no common ground for a fair comparison baseline.

For the sake of brevity, we will only outline the most restricted (and challenging) scenario in which the robot capacity Q is only a quarter of the total number of sensing holes. Tables I and II report the best cost, average cost and average execution time over 10 independent runs for IMSA and MMAS, respectively. Standard deviations (in %) of both mean indicators are also included.

Highlighted in boldface are the best tour costs per instance file, for which meaningful differences between the algorithms were uncovered by a non-parametric Iman-Davenport test running at a significance level of 5%, thus supporting the superiority of MMAS versus IMSA. However, the same test also detected a significant gap in the running time of each optimization scheme, this time in favor of IMSA. Such finding makes sense in light of the fact that IMSA performs no maintenance (update) of a global communication structure like the pheromone matrix found in all ACO algorithms, for which some computation time has to be set aside.

TABLE II
PERFORMANCE OF MMAS UNDER THE LOOK-AHEAD HEURISTIC RULE
ON INSTANCES WITH MINIMUM CARRIER CAPACITY $Q = 25\%n_F$

n	Q	Best cost	Avg cost	Std dev. cost (%)	Avg time	Std dev. time (%)
20	1	2579.59	2579.59	0	37.99	4.2
30	2	3273.18	3416.31	0.45	58.96	5.23
40	3	3460.33	3593.92	1.92	89.41	3.74
50	3	3933.39	4093.7	0.85	109.12	2.06
60	4	3741.19	4053.19	3.58	137.87	0.17
100	6	6288.55	6619.84	0.94	374.8	4.01
200	13	10748.05	11277.01	2.82	942.25	3.67
300	19	15852.06	21705.81	5.22	457.6	2.91
400	25	27681	30951.95	4.25	551.08	2.13
500	31	40812.4	45562.97	4.2	614.05	3.17

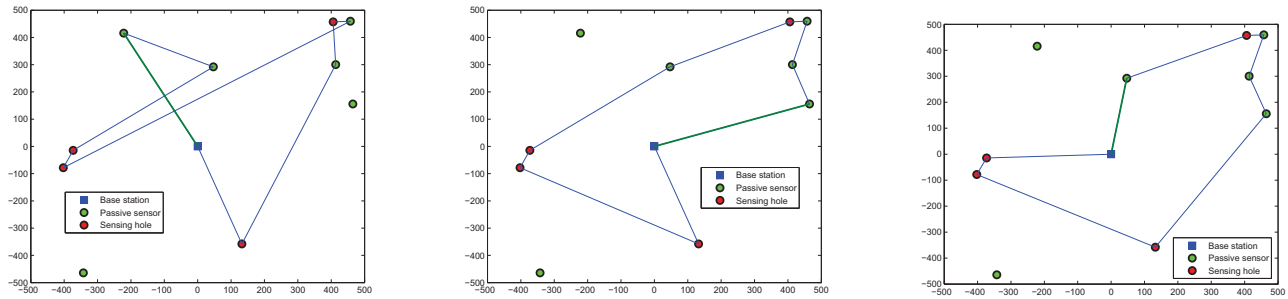
C. Experiment 3: The Two-Step Exploration Strategy

With the aim of accelerating convergence, we have implemented the four variations of TS-MMAS for our problem which were previously outlined in Sec. V-B. Several values for the split factor r were tried: 0.2, 0.25, 0.3 and 0.35 and the empirical evidence supported $r = 0.3$. As to the parameter ns , it was set to 5% of the number of subsolutions calculated in the first stage, a rule of thumb emerging in [21] after a thorough performance analysis.

All along, it seems that the pheromone-driven exploration scheme (P-TS-MMAS) outperforms the solution-driven version (S-TS-MMAS) for 75% of the data sets under consideration. In other words, the collaborative knowledge embedded in the pheromone matrix after the construction of the initial subtours seems to be a more effective means for driving the ants to good regions of the discrete search space. This observation is statistically supported by an Iman-Davenport test that uncovered significant differences among the four two-step implementations and then by a corrected Bonferroni-Dunn test where P-TS-MMAS acted as the control algorithm.

Of the two criteria used to define the subsolution dimensionality (i.e., number of sensing holes visited vs. tour length), the latter yields more encouraging outcomes across the majority (80%) of the data archives tested. The rationale behind this claim has much to do with the topological features of the network being attended. For instance, in Fig. 5(a) we illustrate the undesirable effect of building tours that visit a subset of the damaged sensors under the solution-driven exploration model, for these subtours might be far from optimality yet they will serve as starting solutions for the ants in the second stage, thus likely leading to long robot trajectories. In general, this situation can be relieved by changing the dimension of the subsolutions to be found and let them be a function of the total tour length, as in Fig. 5(b). Next, in Fig. 5(c) we show how the pheromone-driven model was able to find the optimal robot trajectory.

In a nutshell, the application of TS-MMAS to 1-TSP-SELPD produced high-quality solutions (yet not superior in most cases to those found by standard MMAS, as demonstrated by the Iman-Davenport test which yielded no significant differences between them) but in a time frame equivalent to 35% ~ 70% of MMAS' running time, greatly contributing to the robustness of the algorithm when dealing with densely deployed networks.



(a) A tour by S-TS-MMAS-nSH. Note the effect of using as starting solution for stage 2 an “optimal” subtour that visits exactly two sensing holes. (b) A tour by S-TS-MMAS-tourLen. Initial solutions for the second stage are those feasible tours that visit exactly two nodes (of any type). (c) P-TS-MMAS-nSH found the optimal network solution. Ants in stage 2 build their solution from scratch using the pheromone matrix in stage 1.

Fig. 5. Robot trajectory computed with $r = 0.3$, $Q_0 = 0$ and $Q = 3$

VII. CONCLUSIONS

We modeled carrier-based coverage repair in WSRNs as a novel combinatorial optimization problem (1-TSP-SELPD with unitary demand) and solved it by the MMAS approach belonging to the category of ant colony algorithms. Our study highlights the benefits of designing more integral heuristic rules, a component of ACO techniques traditionally neglected but with a tremendous bearing on the overall ACO performance. Moreover, our MMAS-based scheme proved superior to an extended version of simulated annealing and the incorporation of the two-step strategy allowed for considerable computational time reductions. This study can be extended in many ways. Our ongoing research efforts are directed to modeling an scenario in which a fleet of mobile robots has to tend to the network’s coverage needs. Such formulation will give rise to a novel variation of the well-known vehicle routing problem (VRP). Another possible direction is to pose our problem in terms of *focused coverage* [18], i.e., some areas of the monitoring region are more important than others and thus require immediate attention from the robot. This problem keeps a great degree of resemblance with the TSP with time windows (TSP-TW), for which every delivery customer must be served within a strict period of time.

ACKNOWLEDGMENT

This work was partially supported by NSERC Strategic Grant STPSC356913-2007B.

REFERENCES

- [1] M.H. Afshar, “Partially Constrained Ant Colony Optimization Algorithm for the Solution of Constrained Optimization Problems: Application to Storm Water Network Design” *Advances in Water Resources*, vol. 30(4), pp. 954–965, Apr. 2007.
- [2] S. Anily and J. Bramel, “Approximation Algorithms for the Capacitated Traveling Salesman Problem with Pickups and Deliveries” *Naval Research Logistics*, vol. 46, pp. 654–670, 1999.
- [3] L. Boukhatem and L. Friedmann, “Multi-sink Relocation with Constrained Movement in Wireless Sensor Networks” *Ad Hoc & Sensor Wireless Networks*, vol. 8(3-4), pp 211–233, 2009.
- [4] P. Chalasani and R. Motwani, “Approximating Capacitated Routing and Delivey Problems” *SIAM J. on Comp.*, vol. 28, pp. 2133–2149, 1999.
- [5] C.-H. Chang and T.Y. Chung, “A Cross-Layer Design to Achieve Stable and Efficient Parallel TCP over Next Generation Wireless Networks” *Ad Hoc & Sensor Wireless Networks* vol. 8(3-4), pp. 183–210, 2009.
- [6] M. Dorigo and L.M. Gambardella, “Ant Colonies for the Traveling Salesman Problem” *BioSystems*, vol. 43, pp. 73–81, 1997.
- [7] M. Dorigo and T. Stützle, *Ant Colony Optimization*, MIT Press, 2004.
- [8] <http://www.site.uottawa.ca/~rfalc032/benchmarks/1tspsejpd.rar>
- [9] G. Fletcher, X. Li, A. Nayak and I. Stojmenovic, “Back-Tracking based Sensor Deployment by a Robot Team” *Proc. IEEE SECON*, 2010.
- [10] A. Gallais, J. Carle, D. Simplot-Ryl and I. Stojmenovic, “Localized Sensor Area Coverage with Low Communication Overhead” *IEEE Transactions on Mobile Computing*, vol. 7(5), pp. 661–672, 2008.
- [11] M. Gendreau, G. Laporte and D. Vigo, “Heuristics for the Traveling Salesman Problem with Pickup and Delivery” *Computers & Operations Research*, vol. 26, pp. 699–714, 1999.
- [12] H. Hernández-Pérez, *Traveling salesman problems with pickups and deliveries*, Dissertation, University of La Laguna, Spain, 2004.
- [13] H. Hernández-Pérez, I. Rodríguez-Martín and J.J. Salazar-González, “A Hybrid GRASP/VND Heuristic for the One-Commodity Pickup-and-Delivery Traveling Salesman Problem” *Computers and Operations Research*, vol. 36(5), pp. 1639–1645, 2009.
- [14] H. Hernández-Pérez and J.J. Salazar-González, “A Branch-and-Cut Algorithm for a Traveling Salesman Problem with Pickup and Delivery” *Discrete Applied Mathematics*, vol. 145, pp. 126–139, 2004.
- [15] H. Hernández-Pérez and J.J. Salazar-González, “Heuristics for the One-Commodity Pickup-and-Delivery Traveling Salesman Problem” *Transportation Science*, vol. 38, pp. 245–255, 2004.
- [16] F. Ingelrest, D. Simplot-Ryl and I. Stojmenovic, “Optimal transmission radius for energy efficient broadcasting protocols in ad hoc and sensor networks” *IEEE Transactions on Parallel and Distributed Systems*, vol. 17(6), pp. 536–547, Jun. 2006.
- [17] M. Lazarova and P. Borovska, “Comparison of Parallel Metaheuristics for Solving the TSP” *Proc. 9th International Conference on Computer Systems and Technologies*, Gabrovo, Bulgaria, Jun. 2008, article 17.
- [18] X. Li, H. Frey, N. Santoro and I. Stojmenovic, “Focused Coverage by Mobile Sensor Networks” *Proc. 6th IEEE International Conf. on Mobile Ad-hoc and Sensor Systems*, Macau, China, Oct. 2009, pp. 466–475.
- [19] G. Martinovic, I. Aleksic and A. Baumgartner, “Single-Commodity Vehicle Routing Problem with Pickup and Delivery Service” *Mathematical Problems in Engineering*, vol. 2008, Article ID 697981, 17 pages, 2008.
- [20] A. Nayak and I. Stojmenovic, *Wireless Sensor and Actuator Networks: Algorithms and Protocols for Scalable Coordination and Data Communication*, John Wiley & Sons, 2010.
- [21] A. Puris, *Desarrollo de metaheurísticas poblacionales para la solución de problemas complejos*, Ph.D. Dissertation, Universidad Central de Las Villas, Santa Clara, Cuba, Jan. 2010.
- [22] A. Puris, R. Bello, Y. Trujillo, A. Nowe and Y. Martínez “Two-Stage ACO to Solve the Job Shop Scheduling Problem” *Proc. 12th Iberoamerican Congress on Pattern Recognition*, Valparaiso, Chile, Nov. 2007, LNCS 4756, pp. 447–456.
- [23] G. Reinelt, *The Traveling Salesman Problem: Computational Solutions for TSP Applications*, Berlin: Springer-Verlag, 1994.
- [24] I. Stojmenovic, “Localized network layer protocols in sensor networks based on optimizing cost over progress ratio” *IEEE Network*, vol. 20(1), pp. 21–27, Jan./Feb. 2006.
- [25] I. Stojmenovic, D. Liu, and X. Jia, “A scalable quorum based location service in ad hoc and sensor networks” *Int’l Journal of Communication Networks and Distributed Systems*, vol. 1(1), pp. 71–94, 2008.
- [26] T. Stützle and H. Hoos, “MAX-MIN Ant System” *Future Generation Computer Systems*, vol. 16(9), pp. 889–914, Jun. 2000.
- [27] J. Wu and I. Stojmenovic, “Ad Hoc Networks” *IEEE Computer*, vol. 37(2), pp. 29–31, Feb. 2004.