An Ant Colony System for the Skilled, Multi-depot VRP with Due Dates and Time Windows

Marine Dubillard, Xavier Lorca, Matthieu Lauras

Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 56-2 (2023) 11129–11134

An Ant Colony System for the Skilled, Multi-depot VRP with Due Dates and Time Windows

Marine Dubillard * **Xavier Lorca** * **Matthieu Lauras** *

* *University of Toulouse – IMT Mines Albi, Industrial Engineering Department, Albi, France (e-mail: firstName.lastName@mines-albi.fr).*

Abstract: This article introduces a real-world maintenance scheduling problem that can be defined as a Skilled Multi-Depot Vehicle Routing Problem with Due Dates and Time Windows, or Skill-MDVRPDDTW, and addresses two methods to solve it. One is a greedy heuristic inspired from the real-world planning processes used in a water service management context, and the other is a version of the Ant Colony System algorithm, widely used in the literature for the Vehicle Routing Problem and its variants and adapted to fit the features of the real-world maintenance problem. Both the problem and the algorithms are positioned in the literature and mathematically formulated, then experiments and results are discussed and compared through a set of various indicators.

Keywords: Optimization, Simulation, Vehicle Routing Problem, Ant Colony System, Maintenance Scheduling

1. INTRODUCTION

The Vehicle Routing Problem (VRP) is one of the most well-known and studied optimization problems (Toth (2002)). It is an extension of the even better-known Traveling Salesman Problem (TSP) in which *the distance covered by a salesman to visit a set of given cities is minimized*. In the Vehicle Routing Problem, the salesman is replaced by a set of vehicles (with each a capacity for the Capacitated VRP or CVRP) and the cities are "customers" with various needs, such as quantities of products for the CVRP, time duration or time windows for the VRP with Time Windows or VRPTW, skills for the Skill VRP, release and due dates for the Multi-period VRP with Release and Due dates or VRPRD... Moreover, there can be one or several depots from which the vehicles start and end their routes, and the constraint to visit every customer can either be strict or encouraged by a system of rewards or penalties for each visited or missed customer. All variants of the VRP can be rendered dynamic by the addition of customers or the update of travel and service duration during real time execution.

In this study, we focus on a real-world application of a VRP: the scheduling of maintenance operations required by a water distribution network. The problem has many specifications, for it includes time windows, multiple depots, a multiple-day-horizon along with due dates, and skill requirements. It could hence be classified as a Skill Multi-Depot VRP with Due Dates and Time Windows or Skill-MDVRPDDTW.

As well as the TSP or the classic VRP, this very specific optimization problem belongs to the NP-Complete family, and can not be optimally solved within a reasonable time on the whole set of interventions. Therefore, water distribution management companies have to establish heuristic planning processes to be efficient in terms of service quality as well as traveled kilometers.

The resulting research problem is the following: how to assess the efficiency and the robustness to hazards of such a real-world heuristic? And how to solve this complex and specific VRP with algorithms derived from literature?

In this paper, we propose to compare a simulation of the heuristic used in a real-world context of water network maintenance planning with an *Ant Colony System* (ACS) *algorithm*. Section 2 develops a better description of the real-life problem and looks further into literature for ACS algorithms adapted to Vehicle Routing Problems, with various specifications, section 3 describes the present solving heuristic processes, section 4 describes the ACS model proposed, section 5 explains the experimental protocol used and questions the results, and section 6 opens avenues for future work.

2. PROBLEM STATEMENT AND ASSOCIATED BACKGROUND

This section gives a more precise definition of the real-world application problem (a Skill Multi-Depot VRP with Due dates and Time Windows) and provides a succinct overview of the methods used to solve the VRP variants in the literature.

2.1 Problem description and formulation

The real-world specifications are the following: the *location*, *duration* and *nature* of every maintenance intervention (the equivalent of the classical VRP "customers" described above) can be known several days, weeks or months in advance, as well as the *skills* needed to perform them. These interventions can be for instance new buildings to

10.1016/j.ifacol.2023.10.828

hook up, sample collections for water quality measurements, or proactive leak search. Some of the interventions are planned in advance with customers (for leaks or water shortage fixing for example), and are also attached to time constraints: they have a specific *due date*, and a *time window*, precise to the minute, corresponding to the hour of the appointment. The vehicles are each associated to an agent with specific *skills*, *working hours* and a *departure and arrival location*. Moreover, the problem is solved *over several days*, meaning that each day, only a fraction of the whole set of interventions needs to be executed (a fraction which necessarily includes the customer appointments). Finally, reactive maintenance activities add a *dynamic dimension* to the problem, as emergency interventions can appear at any moment of the day.

To model the problem, we first have to define: A the set of agents with their skills and working hours, I the set of interventions with their characteristics (duration, location, skills needed), $I_{app} \subseteq I$ set of appointments (with their due dates and time windows), T the set of days included in the planning horizon. Let us also define B the time of beginning of the working days, and E the time of end. What we want to obtain at the end of the scheduling process is a set of non-intersecting lists of interventions $route_{a,d} \subseteq I$ assigned to each $a \in A$ for each day $d \in T$, as well as the time of beginning of execution $beginTime_i$ for all $i \in route_{a,d}$, for all $a, d \in A \times T$.

2.2 The variants of Vehicle Routing Problems in the literature

This very specific problem is not defined as is in the literature, but each of its specificities has been studied separately. Laporte (1992) defines the most commonly used mathematical model of the simple Vehicle Routing Problem (VRP), using a linear programming formulation. From here, Cordeau et al. (1999) define a model including *time windows and time durations* for each intervention, the VRP with Time Windows (VRPTW). They also consider the *multi-depot variant* of their model, with a specification of each vehicle departure and arrival location (MDVRP). On the other hand, Cappanera et al. (2011) introduce the Skill Vehicle Routing Problem (Skill VRP), with skills required for each task, while Archetti et al (2015) introduce the Multi-period Vehicle Routing Problem with Due dates (MVRPD), adding a temporal dimension with multiple days to execute the tasks. Finally, uncertainties and emergency interventions are modeled in dynamic versions of VRP, called Dynamic VRP and widely developed in Larsen et al (2000).

All these variants have been studied and solved in many ways. Global overviews such as Laporte (1992) or Osaba et al. (2020) for the simple VRP, and El-Sherbeny (2010) for VRP with Time Windows, distinguish *exact methods* and *approximate methods*.

Exact methods include (i) Mixed Integer Linear Programming (MILP) used since Dantzig et al. (1954) for the Traveling Salesman Problem, and adapted to all variants of VRP for both mathematical modeling and solving purposes, (ii) direct tree search and (iii) dynamic programming, developed in Laporte (1992) for the simple VRP, or (iv) lagrangean relaxation and (v) column generation developed in Cordeau et al. (1999) for the VRP with Time windows.

Among *approximate methods*, we find (i) heuristics, such as 2-opt or 3-opt algorithms, more developed in Baker et al. (1986), (ii) metaheuristics, both single-solution based such as Tabu Search or Simulated Annealing developed in Laporte (1992), or large neighborhood search in Mancini (2016) and population-based algorithms such as genetic algorithms, particle swarm optimization, ant colony optimization and many others nature-inspired described in Osaba et al. (2020). A review of metaheuristics for the VRPTW can be found in Dixit et al. (2019). Finally and more recently, (iii) learning-based optimization algorithm, detailed in Li et al. (2022) for their application to the VRP.

Among the methods providing good results to large instances of the VRP and all the variants that constitutes the Skill Multi-Depot VRP with Due dates and Time Windows, we chose to investigate the *Ant Colony System Optimization algorithm*. Exact methods are more time consuming while searching for an optimal result, which is not an absolute necessity in this specific case. On the contrary, simple constructive heuristics might not be efficient enough, and most importantly, a simple heuristic is what is already presently used to solve the problem, and what we are trying to evaluate. We found no example of learning-based optimization for a VRP on several days or with due date and finally, among all metaheuristics, if many showed good results, none seems to stand out from the others in term of time or cost efficiency. We chose address Ant Colony Optimization for its immediate similarity to the problem's modeling: ants exploring paths to find the shortest routes to visiting a set tasks can directly be put in parallel with agents spending the less time possible in transportation while carrying out the interventions needed. The mathematical models of Ant Colony Optimization's variants are described in Cordón et al. (2002). A successful ACS algorithm for the VRP with Time Windows can be found in Gong et al. (2007), and in Yu et al. (2011) or Wang et al. (2020) for the Multi-Period VRP with Time Windows.

3. REAL-WORLD HEURISTIC DESCRIPTION

This section describes the organizational constraints used in a water service management company, and gives a model of their scheduling heuristic.

In order to reduce the size of the scheduling problem, the initial set of interventions is separated in smaller sets with *geographic boundaries*, within which local service managers are independently in charge of the scheduling, with a *planning horizon of one week*. Moreover, the scheduling process uses a *resource-oriented* approach, meaning that it aims to optimize the filling of the schedule of each agent (as opposed to *task-oriented*, where the positioning of each task is the first decision criteria). It also uses *stricter skill constraints*, with agents executing only one type of intervention over the day or the week. The different steps of the scheduling process are described in pseudo-code in Algorithms 1 and 2, and schematized in Fig. 1.

Algorithm 1 shows a two-stage scheduling heuristic. First, interventions with strict time windows are assigned to

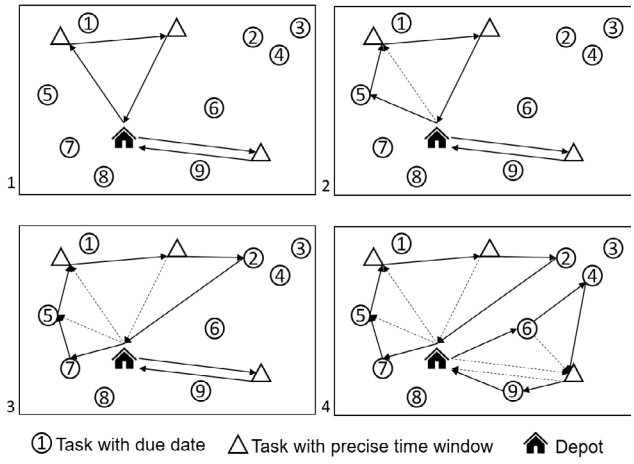

Fig. 1. The different steps of the scheduling heuristic

the first available agent with the corresponding skills. This is what is represented in the first part of Fig. 1: 3 appointments, represented with triangles, are affected to two agents. As shown in Algorithm 2, at each iteration, all interventions are considered for every slot in the currently constructed planning, and the feasibility (appropriate skills, and fitting intervention and traveling times) and cost of their addition at this precise position is calculated. The most advantageous insertion is then kept in the route, as schematically represented in part 2 of Fig. 1; here, task number 5 is the closest feasible addition to the first agent's route. This loop continues until no intervention can be added in the schedule, and the same procedure is applied to the next agent with the remaining interventions. Part 3 of Fig. 1 shows the complete route of the first agent (no other intervention can be added after the addition of tasks 5, 7 and 2), and part 4 the complete routes of all agents (tasks 6, 4 and 9 have been added to the second agent's route). We can notice that once an intervention has been assigned to an agent $a \in A$ and on a day $d \in T$, its affectation is never reconsidered, which makes this heuristic a member of the *greedy algorithms* family.

Algorithm 1 $solveWithExistingHeuristic()$

for $d \in T$ **do**
  for $i \in I_{app}$ **do**
    while not $isPlanned_i$ **do**
      $a \leftarrow$ first agent of A
      if a has skills and disponibility for i **then**
        insert$(i, beginTime_i)$ in $route_{a,d}$
        $isPlanned_i \leftarrow True$
      else
        $a \leftarrow$ next agent in A
  for $a \in A$ **do**
    while not $isFull_{a,d}$ **do**
      $i, beginTime_i = bestInsertion(a, d)$
      if $i = \oslash$ **then**
        $isFull_{a,d} \leftarrow True$
      else
        insert$(i, beginTime_i)$ in $route_{a,d}$

Finally, emergencies are added to the route of the agent who is the closest to its location when they occur, and the rest of his planned interventions are simply postponed as described in Algorithm 3, where E represents the time of end of the working day. The postponed interventions that do not longer fit in the working schedule are removed from the route.

Algorithm 2 $bestInsertion(Agent\ a, Day\ d)$

$bestIntervention, bestTime = \oslash, B$
$cost_{best} = +\infty$
for $i \in I$ **do**
  if not $isPlanned_i$ **then**
    for $i2 \in route_{a,d}$ **do**
      if i is feasible after $i2$ **then**
        $cost_i =$ cost of adding i after $i2$
        if $cost_i \leq cost_{best}$ **then**
          $bestIntervention \leftarrow i$
          $bestTime = endTime_{i2} + distance_{i2,i}$
          $cost_{best} \leftarrow cost_i$
return $bestIntervention, bestTime$

Algorithm 3 $insertEmergency(Emergency\ e)$

$a =$ closest agent to the emergency
insert$(e, beginTime_e)$ in $route_{a,emergency\_d}$
for $i \in route_{a,emergency\_d}$ **do**
  if $endTime_i \geq beginTime_e$ **then**
    $\delta = duration_e +$ transportation time for e
    $beginTime_i \leftarrow beginTime_i + \delta$
    $endTime_i \leftarrow endTime_i + \delta$
    if $endTime_i + distance_{i,depot} \geq E$ **then**
      remove(i, $route_{a,emergency\_d}$)

4. ANT COLONY OPTIMIZATION PROPOSAL

This section describes the Ant Colony System (ACS) algorithm model we developed, and how it was adapted to fit the specific VRP. The dynamic aspect and the addition of emergency interventions is for the moment put aside.

4.1 Ant Colony Systems

In Ant Colony Optimization (ACO), "artificial ants" construct solutions by following probabilistic rules inspired by the natural behavior of ant populations. Every path $x_{i,j}$ between two locations $i, j \in I^2$ has an associated amount of pheromone deposit $\tau_{i,j}$ that evolves during the algorithm's progression. The probability for a path to be chosen by an artificial ant is a function of its quantity of pheromone deposit. The update of pheromone during the algorithm progression depends on which ACO method is chosen. Here, we apply a variant of the *Ant Colony System model* or *ACS*, as it is described in Cordón et al. (2002). At each iteration of the ACS algorithm, each ant provides a solution, and the pheromone deposits are updated in two ways: (i) *online evaporation* is applied on every path visited by an ant, which means their pheromone deposits are decreased during the solution construction, in order to encourage *diversification* (a path has less chances to be chosen during an iteration if it has already been chosen before); (ii) At the end of every iteration, the global-best solution is rewarded with an increase of pheromones on all the paths it contains, in order to encourage *intensification* (the more promising paths have more chances to be selected by the next ants). On top of pheromone deposits,

each path $x_{i,j}$ has an associated *heuristic information* $\eta_{i,j}$, that is set in the initialization phase of the algorithm, and does not change during its execution. This heuristic information will be used in the same way as pheromones to guide the construction of routes, and contains structural information about the problem, such as the distance between i and j, so as to increase the probability of choosing in the same route interventions that are close to each other.

Furthermore, as indicated in Yu et al. (2011) and Wang et al. (2020), the problem is not symmetric regarding the time horizon, so we use a multi-dimension pheromone matrix, with different pheromone deposit associated to each path (i, j) for each day d.

4.2 The specific model

Let us define *nIterations* and *nAnts* the numbers of iterations and ants. In this specific case, as specified earlier, a solution s is a non-intersecting set of lists of interventions $route_{a,d}$ assigned to each agent $a \in A$ for each day $d \in T$, with the time of beginning of execution $beginTime_i$ for all intervention $i \in route_{a,d}$. In each iteration, every ant will thus construct a solution composed of $|A| \times |T|$ routes. The algorithm structure is described in Fig. 2. In this structure, four steps need to be defined more precisely: the *initialization phase*, the *route construction phase*, the *addition of appointments to daily schedules* and the *pheromone update in each iteration*.

Initialization In the initialization phase, a value has to be set for the initial pheromone rate τ_0 for every path $(i, j) \in I^2$ and day $d \in T$, as well as for the heuristic information $\eta_{i,j}$ for every path $(i, j) \in I^2$. The inverse of the distance between i and j is chosen to encourage the selection of short paths.

$$\tau_{i,j,d} = \tau_0, \ \forall i, j, d \in I^2 \times T \tag{1}$$

$$\eta_{i,j} = 1/distance_{i,j}, \ \forall i, j \in I^2 \tag{2}$$

Route construction A solution consists in a set of feasible routes, one for each day $d \in T$ and each available agent $a \in A$. Ants start from their associated depot node at departure time B. At every stage of the route construction, let us call $\mathcal{N}_{k,s} \subseteq I$ the feasible neighborhood of ant k at the state s, that is to say the set of interventions that can be done by agent a on day d, that has not already been planned in a previous state, and that fits in the schedule timing. For each state s of the route construction, if i is the last intervention made, the next intervention j is chosen among all interventions with the following probability function, where α and β are two constant parameters that weight the balance between the influence of pheromones and of heuristic information on the choice of the next intervention:

$$p^s_{i,j,d} = \begin{cases} \dfrac{\tau^\alpha_{i,j,d} \cdot \eta^\beta_{i,j}}{\sum_{l \in \mathcal{N}_{k,s}} \tau^\alpha_{i,l,d} \cdot \eta^\beta_{i,l}} & \text{if } j \in \mathcal{N}_{k,s} \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

Once the next intervention j has been chosen, online evaporation is applied on $\tau_{i,j,d}$ following this rule, where $\varphi \in (0, 1]$ is a *decay parameter*:

$$\tau_{i,j,d} \leftarrow (1 - \varphi) \cdot \tau_{i,j,d} + \varphi \cdot \tau_0 \tag{4}$$

Appointments addition In ACS algorithms, additional steps called *daemon actions* can be added at the end of iterations to modify solutions (most of the time to improve them with local search methods). Here, some interventions $I_{app} \subseteq I$ are *appointments*, which means they have time windows precise to the minute, and have to be selected on their *due date* dd_i. To make sure they appear in every solution, an additional step is executed on each day d after every agent's route has been constructed. One by one, every intervention in $i \in \{I_{app}, \ dd_i = d\}$ is inserted in the route of the agent a that it degrades the less. Every intervention of agent a after the insertion of the appointment i is shifted in time, and interventions that exceeds now the end time of the schedule are removed and have to be scheduled again in the following days.

Pheromone update A solution s is evaluated on its *added value indicator*, calculated as shown in Equation 5, where d_i is the time duration of intervention i and $t_{i,j}$ is the transportation time between the locations of interventions i and j. The added value indicator corresponds to the rate between the total time spent in intervention and the total time spend in both transportation and intervention.

$$C(s) = \frac{\sum_{(i,j) \in s} d_i}{\sum_{(i,j) \in s} d_i + t_{i,j}} \tag{5}$$

The pheromone update is applied at the end of each iteration on the global-best solution s^*, following the rule defined by Equation 6, where $\rho \in (0, 1]$ is a pheromone deposit rate, and . For all day $d \in T$ and all path $(i, j) \in I^2$ selected in the solution s^* on d:

$$\tau_{i,j,d} \leftarrow (1 - \rho) \cdot \tau_{i,j,d} + \rho \cdot 10 \cdot C(s^*) \tag{6}$$

5. EXPERIMENTAL PROTOCOL AND DISCUSSION

Both the present field heuristic and the ACS model have been experimented on several instances of the Skill-VRPDDTW, constructed so as to represent at best the water service management field reality. Even though it can be handled by both algorithms, the multi-depot aspect has been set aside from the experimental protocol to fit the field data used to calibrate the instances, which is for the moment concentrated on only one depot. For the existing heuristic, as it fixes geographic boundaries between the different depots, the results should not differ. Every intervention is only part of one geographic area and is attached to one specific depot. For the ACS algorithm however, more solutions could be considered because mixing geographic areas is possible. One could therefore expect slightly better results when adding multiple depots.

Problem parameters description An instance D of the problem is defined by a set of agents A, a time horizon T and and a set of interventions I. The scheduling time horizon has been set to $|T| = 20$ days to represent four weeks of maintenance, for $|A| = 4$ agents with 2 different levels of skills. Let us call $Capa_D$ the working capacities, in minutes, of all the agents of A working 420 minutes every day of the horizon T, and $Load_D$ the total time duration of the set of interventions I. The density in interventions $density_D$ is defined as following:

$$density_D = \frac{Load_D}{Capa_D} = \frac{420 \cdot |T| \cdot |A|}{\sum_{i \in I} d_i} \tag{7}$$

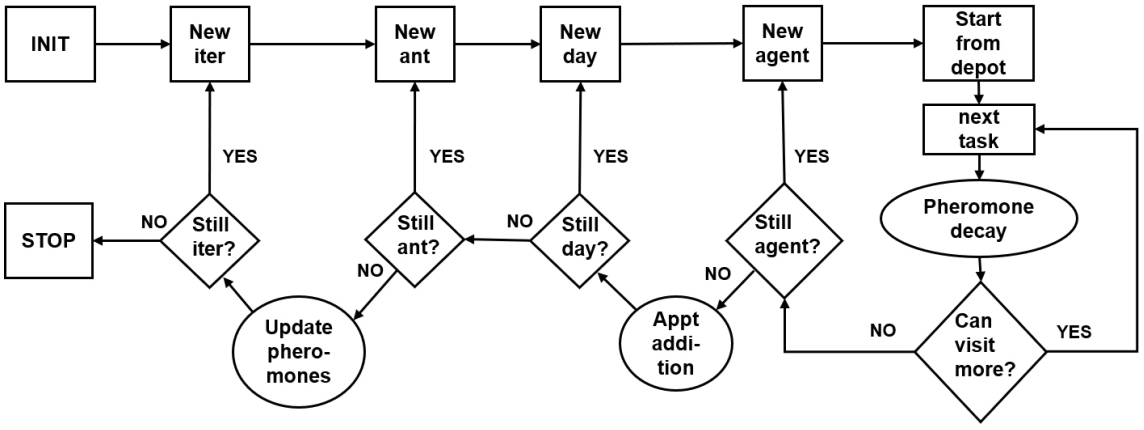

Fig. 2. The different steps of the ACS metaheuristic

Instances have been constructed so that $Load_D$ corresponds to 0.5, 1 and 1.5 times $Capa_D$, in order to evaluate the quality of the different solving methods when the density $density_D$ of the instance vary. In all three cases, the intervention durations d_i are equally distributed between 15, 30, 60 and 90 minutes, for all $i \in I$, which leads to instances of 537, 1074 and 1611 interventions. Appointments are always generated with durations of 15 minutes, and they represent 10% of the 15-minute-long interventions. Two *types of skills* are generated: half of the interventions require the skill "A" and the other half the skill "B", while among the four agents, one has only the skill "A", one the skill "B" and the two others both. *Locations of interventions* are randomly chosen in a set of 2000 addresses not distant from more than 80 minutes of transportation time of each other. Two interventions can have the same location, and the average transportation time between two locations is 35.6 minutes.

Ant Colony System parameters The parameters relative to the ACS algorithm have been set as in Yu et al. (2011), where a Multi-Period VRP with Time Windows is solved with an Improved Ant Colony Optimization algorithm. The parameters are therefore set to these values: $\tau_0 = 1$; $\eta_{i,j} = 1/distance_{i,j}\ \forall i, j \in I^2$; $\alpha = 2$; $\beta = 6$; $\varphi = \rho = 0.01$.

Results The average results for both the real-world-inspired heuristic and the ACS algorithm, on 15 instances with densities in interventions varying between 0.5, 1 and 1.5 can be seen in Table 1. The *added value indicator* $C(s)$ (as defined before in Equation 5) is measured on each solution s, as well as the number of interventions planned $nb\_p(s)$, the total time spent in intervention $w\_t(s)$ and in transportation $t\_t(s)$ in minutes, and the time of calculation in seconds $c\_t(s)$.

Table 1. Results

Algo	Density	$C(s)$	$nb\_p(s)$	$w\_t(s)$	$t\_t(s)$	$c\_t(s)$
RWH	0.5	0.89	537	16869	2053	2
ACS	0.5	**0.91**	537	16869	**1623**	123
RWH	1	0.91	1011	27870	2870	9
ACS	1	**0.93**	950	29745	**2078**	761
RWH	1.5	0.91	1322	28649	2862	20
ACS	1.5	**0.94**	968	29199	**1975**	1360

In the first case, when the work capacity is twice the total time of interventions (density=0.5), both algorithms manage to schedule every intervention. The number of interventions planned and the time spent in interventions is therefore the same for both algorithms. However, the time spent in transportation is reduced by 20% for the ACS algorithm, which represents a little more than one complete day of work of one agent. Calculation time for its part goes from 2 seconds for the heuristic to 2 minutes for the ACS.

When the work capacity is equal to the total time spent in intervention (density=1), both algorithms schedule only a part of the interventions. The balance between the time spent in intervention and in transportation, or *added value indicator* is still better for the ACS algorithm, by 2%, and time spent in transportation is reduced by $\tilde{2}7\%$ (which represents more than two complete days of work) for an increase of time in intervention of 6.7%. Calculation time goes from 20 seconds to almost 13 minutes. An interesting result is the number of intervention planned, by 6% higher for the Real-World Heuristic, while the total time spent in intervention is lower. This is because the heuristic process is only guided by distance, while the ACS algorithm also *prioritizes the planning of the longest interventions*. Indeed the length of a detour to include an intervention in a route is proportional to the length of the intervention. This also gives another advantage to the ACS algorithm compared to the Real-World Heuristic, which tends to leave for later the longest interventions (and therefore the hardest to fit in a schedule).

Finally, when the capacity is lower than the the total time spent in intervention with a density of 1.5, the results observed before (density=1) are amplified: transportation time is decreased of 31% with the ACS algorithm, intervention time increased of 1.9%, while the number of planned tasks also falls from almost 27%. The Real-World Heuristic is still very fast, with an average calculation time of 20 seconds, compared to 23 minutes for the ACS algorithm.

What can globally be learned from these experiments is that although the Real-World Heuristic has the advantage of being fast in calculation (20 seconds in average for a 1611-intervention instance), the transportation time is always reduced when using the ACS algorithm, and all the more so as the instance gets bigger. When not all interventions can fit in the schedules (density=1 or 1.5), the Real-World Heuristic tends to plan a larger number

of shorter interventions, which has for logical effect to take more time in transportation for an equivalent time in intervention. One can nevertheless see that even with the exact same interventions planned (when the density is 0.5), ACS brings an average reduction of 20% in transportation time, compared to the Real-World Heuristic. It shows that the Real-World Heuristic, choosing the closest intervention agent by agent, gives acceptable routes in a real short time, but also that 20% to 30% of transportation time can be saved, and the added value rate C can increase by 2% to 3% by using more complex algorithms that explore more widely the solution space, such as the ACS meta-heuristic.

6. CONCLUSION AND PERSPECTIVES

The Skill Multi-Depot VRP with Due Dates and Time Windows has been introduced as well as two algorithms to solve it: a real world heuristic already used for water maintenance activities, and an Ant Colony System (ACS) algorithm inspired from the literature. The results obtained in this paper for the *one-depot variant of the problem* show that the use of a metaheuristic such as the ACS could lead to a reduction in transportation time of 20% to 30% for the *forecasted scheduling*, that is to say *without taking hazards into account*. Hazards can occur in the form of (i) intervention time increase or decrease, (ii) transportation time increase or decrease, (iii) and the addition of emergency interventions, which can radically change the nature of the previously planned schedules. In future work, robust optimization will be more deeply studied so as to find strategies for rescheduling the interventions after the occurrence of said hazards.

Moreover, in the real world application, the precise time of beginning of interventions several weeks in advance is not necessary. In order to spare some useless calculation time, we consider to propose a two-stage planning, with first (i) the attribution of a day of execution for each intervention, and (ii) then a precise schedule for the first 5 days, for example.

Both the consideration of hazards and the problem separation will certainly lead to a loss in efficiency on the various key performance indicators. The results obtained in this study can be considered as *upper bounds* for the gain in efficiency when scheduling the maintenance activity.

REFERENCES

P. Toth et D. Vigo. The vehicle routing problem. In Philadelphia: Society for Industrial and Applied Mathematics, 2002.

Gilbert Laporte. The vehicle routing problem: An overview of exact and approximate algorithms. European Journal of Operational Research 59, 345–358, 1992.

J.-F. Cordeau, Guy Desaulniers, Jacques Desrosiers, Marius M. Solomon, and François Soumis. The VRP with Time Windows. In *Les Cahiers du GERAD*, 1999.

P. Cappanera, L. Gouveia, et M. G. Scutellà. The Skill Vehicle Routing Problem. In Network Optimization, Berlin, Heidelberg, p. 354-364, 2011.

C. Archetti, O. Jabali, and M. G. Speranza. Multi-period Vehicle Routing Problem with Due dates. In Computers & Operations Research, vol. 61, p. 122-134, 2015.

Larsen, Allan and Madsen. The dynamic vehicle routing problem. In Institute of Mathematical Modelling, Technical University of Denmark, 2000.

G.B. Dantzig, D.R. Fulkerson and S.M. Johnson. Solution of a Large Scale Travelling Salesman Problem. In Operations Research 2, 393 - 410, 1954.

B. Li, G. Wu, Y. He, M. Fan, and W. Pedrycz. An Overview and Experimental Study of Learning-based Optimization Algorithms for Vehicle Routing Problem. In IEEE/CAA Journal of Automatica Sinica, vol. 9, n 7, p. 1115-1138, 2022.

Simona Mancini. A real-life Multi Depot Multi Period Vehicle Routing Problem with a Heterogeneous Fleet: Formulation and Adaptive Large Neighborhood Search based Matheuristic. In Transportation Research Part C: Emerging Technologies, vol. 70, p. 100-112, 2016.

P. Cappanera, C. Requejo, and M. G. Scutellà. Temporal constraints and device management for the Skill VRP: Mathematical model and lower bounding techniques. In Computers Operations Research, vol. 124, p. 105054, 2020.

N. A. El-Sherbeny. Vehicle routing with time windows: An overview of exact, heuristic and metaheuristic methods. In Journal of King Saud University - Science, vol. 22, n 3, p. 123-131, 2010.

O. Cordón García, F. Herrera Triguero, and T. Stützle. A review on the ant colony optimization metaheuristic: basis, models and new trends. In Mathware soft computing - 2002, Vol. IX, Núm. 2-3, 2002.

E. Osaba, X.-S. Yang, and J. Del Ser. Is the Vehicle Routing Problem Dead? An Overview Through Bioinspired Perspective and a Prospect of Opportunities. In Nature-Inspired Computation in Navigation and Routing Problems: Algorithms, Methods and Applications, X.-S. Yang et Y.-X. Zhao, p. 57-84, 2020.

E. Osaba, X.-S. Yang, et J. Del Ser. Solution Improvement Heuristics for the Vehicle Routing and Scheduling Problem with Time Window Constraints. In American Journal of Mathematical and Management Sciences, vol. 6, n 3-4, p. 261-300, 1986.

A. Dixit, A. Mishra, and A. Shukla. Vehicle Routing Problem with Time Windows Using Meta-Heuristic Algorithms: A Survey. Harmony Search and Nature Inspired Optimization Algorithms, Singapore, p. 539-546, 2019.

W. Gong, X. Liu, J. Zhang, and Z. Fu. Two-Generation Ant Colony System for Vehicle Routing Problem with Time Windows. In 2007 International Conference on Wireless Communications, Networking and Mobile Computing, Shanghai, China, p. 1917-1920, 2007.

Y. Wang, L. Wang, G. Chen, Z. Cai, Y. Zhou, and L. Xing. An Improved Ant Colony Optimization algorithm to the Periodic Vehicle Routing Problem with Time Window and Service Choice. In Swarm and Evolutionary Computation, vol. 55, p. 100675, 2020.

B. Yu and Z. Z. Yang. An ant colony optimization model: The period vehicle routing problem with time windows. In Transportation Research Part E: Logistics and Transportation Review, vol. 47, n 2, p. 166-181, 2011.