IFAC-PapersOnLine, Volume 50, Issue 1, 2017, Pages 8037-8044
DOI:10.1016/j.ifacol.2017.08.1229

Use of Co-operative UAVs to Support/Augment UGV Situational Awareness and/or Inter-Vehicle Communications

F. Quintin, S. Iovino, A. Savvaris, A. Tsourdos

Cranfield University, Cranfield, Bedfordshire, MK43 0AL
UK (e-mail: s.iovino@cranfield.ac.uk).

Abstract: This paper presents the development of a path-planning algorithm for Unmanned Aerial Vehicles (UAVs) in order to increase the situational awareness for platooning vehicles. The scenario considers a team of cooperative UAVs, initially docked on moving Unmanned Ground Vehicles (UGVs). In particular, the goal consists in finding the best routing plan for the UAVs in order to visit some designated targets in a wide search area to augment the UGVs' situational awareness. Taking into account the maximal endurance constraint of the UAVs, this problem becomes equivalent to a time-constrained multiple depot vehicle routing problem with moving depots. To tackle this variant of the well-known vehicle routing problem, a methodology based on a TABU search meta-heuristic is implemented. While respecting the endurance constraint, this methodology tempts to optimize multiple objectives: the number of UAVs required, the total length of the routing and the balance of the lengths of the different routes within the routing plan. The proposed approach based on a meta-heuristic gives relevant results in a relatively short period of time.

Keywords: Co-operative UAVs; Multiple-depots vehicle routing problem; path planning; vehicle routing problem with moving depots; TABU search meta-heuristic.

1. INTRODUCTION

The use of robots has many applications in the military as well as in the civilian areas. In particular, unmanned vehicles can carry out hazardous and difficult tasks in hostile or dangerous environments. Removing the operator from the vehicle is a very promising challenge to reduce the risk to human life in these types of missions. In case of extreme situations such as natural disasters or critical battlefields, part of the missions could be entirely dedicated to a cooperative team of aerial and ground unmanned vehicles. These unmanned vehicles, tasked as a team at a high level, can efficiently perform an aerial ground mapping of large predefined areas for missions such as exploration, surveillance, target detection, tracking and search and rescue operations. Indeed, the deployment of such a team is well suited to achieve these missions since the cooperation of ground and aerial vehicles exhibit strong and complementary characteristics, leading to efficient synergies and greater reliability. Unmanned Aerial Vehicles (UAVs), such as small rotorcrafts, are easily transportable and can be launched quickly to explore a large area and search for targets. However, these small UAVs cannot carry heavy payloads such as large LIDAR and cameras. Therefore, sensors requirements, combined with other uncertainties, may limit the UAVs capabilities of precise detection and localization of targets on the ground. On the other hand, Unmanned Ground Vehicles (UGVs), such as Armed Forced Vehicles (AFVs), benefit from high resolution sensor capabilities but they may be limited by their short range due to obscured field of view in the presence of ground obstacles. Then, the use of cooperative aerial and ground vehicles is challenging and much effort and research work have been pushed into exploiting the complementarity capabilities of UAVs and UGVs. The deployment of autonomous vehicles is gaining ground thanks to many achievements and current progress in modern technologies such as flight controls, sensors and computing capabilities. UAVs and UGVs can be highly automated and do not require the direct control of the operator during the missions. Path-planning algorithms can efficiently plan in advance optimal trajectories for the unmanned vehicles to follow. Path-planning algorithms also aim to find a specific trajectory which optimizes a specific objective function such as the shortest path in terms of time or energy supply requirements. In this paper, the path-planning problem is investigated for UAVs docked on AFVs and tasked as a cooperative team to cover a large area and visit some designated targets. This problem is finally equivalent to solve the well-known Multiple Depot Vehicle Routing Problem. It also focuses on the deployment of a fleet of UAVs during a surveillance mission in a realistic and dynamic environment. It is assumed that UAVs are first stored on the ground vehicles and docked on recharging stations, before being launched to visit some predefined targets.

This work continues the efforts of Leonard (2015) in solving the dynamic multiple travelling salesmen problem. Leonard's Ant Colony optimization provides near-optimal routing accounting for multiple ground vehicles with only one UAV docked on each UGV and a dynamic set of targets. The present paper takes a slightly different approach and uses different algorithms in order to extend Leonard's results. First, TABU search optimization is used to solve the problem. Then, several constraints are added to the framework: UAVs capabilities are limited by their endurance, more than one UAV can be launched from an UGV. Although the constraint of a dynamic

Published by Elsevier. This is the Author Accepted Manuscript issued with: Creative Commons Attribution Non-Commercial No Derivatives License (CC:BY:NC:ND 4.0).
The final published version (version of record) is available online at DOI:10.1016/j.ifacol.2017.08.1229. Please refer to any applicable publisher terms of use.

set of targets is released, the possibility to add a new target during the optimization process is permitted in the present work. Moreover, the UGVs are considered here as dynamic and can move forward while the UAVs are flying. The minimization of an objective function in a realistic and dynamic environment where the ground vehicles are moving is the main contribution of this work.

This paper is organized as follows: first, a literature review on Multiple Depot Vehicle Routing Problems and related resolution methodologies, pertaining to this research, is conducted in Section 2. Then, Section 3 defines mathematically the problem for which Section 4 offers a resolution methodology. Section 5 highlights the performances of the proposed algorithms based on the analysis of the test results.

2. RELATED WORK

2.1 Vehicle Routing Problems

The Vehicle Routing Problem (VRP) consists of a multiple tasks allocation problem. It aims to assign a cooperative team of vehicles a set of orders, which constitute geographically spread targets to visit, in the most optimal way based on a predefined objective function while conforming to various constraints. In the multiple Travelling Salesmen Problem (m-TSP), the aim is to allocate each moving vehicles a list of targets while minimizing an objective function such as the maximum length of one route. Each vehicle visits the targets as ordered in its own list and then returns to its starting position. Each list of targets consists then in a tour or a route. The VRP extends the m-TSP, by adding some capacity constraints to the vehicles. A wide part of the literature tackles this problem in the context of a supply chain problem. For instance, the first original formulation of the VRP was proposed by Dantzig at al. in 1959, who focused on the computation of a set of optimal routes for a fleet of gasoline delivery trucks. The analogy is relatively straight forward. Indeed, while the UGVs are equivalent to the depots or factories, the UAVs can represent the vehicles, such as trucks, doing the delivery or the collection by some customers which are by analogy similar to the targets.

Depending on the assumptions and constraints, many variants of the VRP exist and for each sub-problem a specific mathematical model can be developed. A generic mathematical description for a standard VRP with capacity constraints is given by Freitas (2012). A large panel of problem variants can be generated by adding or relaxing different constraints to the original formulation. Carlton, in 1995, has proposed a hierarchical classification scheme of the main variants of the Vehicle Routing Problem. His classification highlights the Capacited Vehicle Routing Problem (CVRP), the Periodic Vehicle Routing Problem (PVRP), the Distance-constrained Vehicle Routing Problem (DVRP), the Vehicle Routing Problem with Time Windows (VRPTW), the Vehicle Routing Problem with Back-hauls (VRPB), the Vehicle Routing Problem with Pick-up and Delivery (VRPPD) and the Multiple Depot Vehicle Routing Problem (MDVRP). Hybrids of these variants are also sometimes investigated such as the Capacited and Distance-constrained Vehicle Routing Problem (DCVRP). This classification is well represented in the diagram presented in Fig. 1 (Montoya-torres at al., 2015).

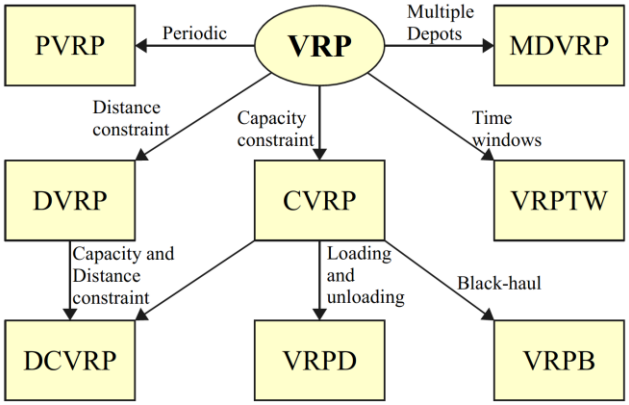


Fig. 1. Classification of the different variants of the Vehicle Routing Problem (VRP).

2.2 Resolution Methodologies

The VRP is a well-known combinatorial optimisation problem. A wide panel of possible resolution methodologies exists for this kind of problems. The potential optimization strategies include exacts methods, heuristics algorithms and meta-heuristics algorithms. The VRP has been classified as an NP-hard problem. Winston at al. (2003) have highlighted that this class of problems cannot be solved efficiently to optimality in terms of computational load. A problem is also qualified as NP-hard if the computational load required to solve the problem increases exponentially with the size of the problem. Therefore, exact methods which aim to find out the optimal solution become quickly inoperative for bigger size problems because of the high computational load requirements. Consequently, the majority of papers found in the open literature regarding VRP have investigated the use of resolution methodologies which give a promising compromise between the near-optimality of the solution and the computational load. Indeed, from a more realistic and operational point of view, operators, which implement such strategies, often prefer having an approximate but good enough solution in a relatively short period of time. This explains the interest of researchers in the development of heuristics algorithms and meta-heuristics strategies.

Meta-heuristics are powerful and efficient techniques used to solve a large panel of NP-hard combinatorial problems. The aim of the meta-heuristics is similar to the one of the heuristics: find out a feasible solution whose quality is acceptable in a short period of time. Contrary to classical heuristics, meta-heuristics use a generic scheme which is completely independent of the specific problem. A meta-heuristic consists in an iterative decision making process that guides and leads the operations of subordinate heuristics by rigorously analysing the situation at each iteration to generate the best exploration of the solution space in a very short period of time. Moreover, meta-heuristics often involve mechanisms and algorithms which enable the solution search to avoid being trapped in a local minimum. Several types of meta-heuristics

have been applied to the VRP. Comparing to classical heuristics, they perform a more efficient neighbourhood search of the solution space and are more likely to generate better solutions since they can escape local minimum.

As described by Cordeau at al. (2007), meta-heuristics used for VRP can be broadly divided into three categories: local search (simulated annealing, TABU search), population search (genetic search) and learning mechanism (neural networks, Ant Colony optimization). Among local search meta-heuristics, TABU Search (TS) was first proposed by Glover (1986) and has become since that time one of the best and most widely used local search methods for combinatorial optimization problems. TS explores the solution space by looking at the possible neighbour solutions. TS moves from a current solution x_k identified at iteration k to the best solution $x_k + 1$ in a subset of the neighbourhood $N(x_k)$ of x_k. Since $x_k + 1$ does not necessarily have a lower cost than x_k, a TABU mechanism is implemented into the algorithm to prevent the search from cycling over the same sequence of solutions.

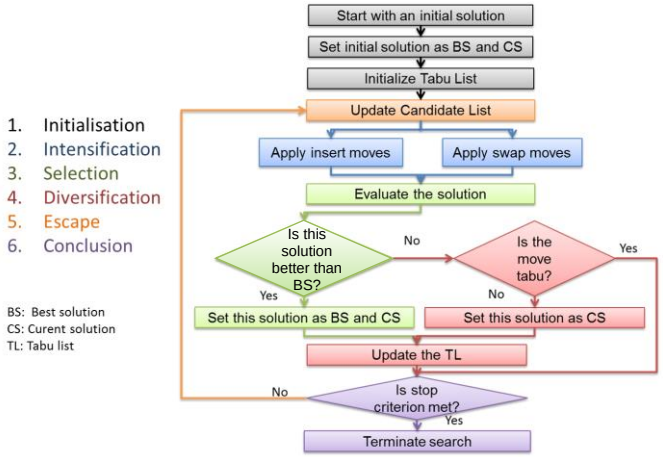


Fig. 2. TABU Search Algorithm.

A very straightforward way to prevent cycles is to forbid the search to return to already evaluated solutions. However, this method would require very large memory capabilities and significant computational loads. Therefore, only a few attributes of the current solutions are recorded and any solution which presents one of these attributes cannot be considered for θ iterations. Generally, the attributes concern the different last movements, which have led to the current solution. The TABU mechanism blocks also a list of temporary forbidden or *tabu* movements which is often called *tabu list*. This process is also similar to a short-term memory. Other mechanisms such as diversification and intensification complete the TS algorithm and make it more robust and efficient. The diversification process ensures possibilities to avoid being trapped in a local minimum by making sure that the search explores a wide section of the solution space. The diversification process registers all key attributes of past solutions and penalizes frequently performed moves. This process is often referred to as long-term memory. On the other hand, the intensification process aims to accentuate the search in the most promising areas of the solution space. The global architecture of the TS meta-heuristic is illustrated in Fig. 2. The performance and the efficiency of the TS rely on the neighbourhood structure, the short-term memory, the long-term memory and the search intensification process.

3. PROBLEM FORMULATION

The formulation is inspired from Crevier et al. (2005) who proposed a resolution methodology to solve the MDVRP with inter-depots routes, based on a three phase strategy composed of adaptive memory, TABU search and set partitioning algorithm. However, in the present work, a different approach to solve the problem is investigated.

Let $G = (V_c \cup V_d(t), A)$ be a directed graph where $V_c = \{v_1, \dots, v_{N_t}\}$ is the target set, $V_d(t) = \{v_{N_t+1}(t), \dots, v_{N_t+1+N_u}(t)\}$ is a set of N_u moving UGVs where t stands for the real time, and $A = (v_i, v_j): v_i, v_j \in V_c \cup V_d(t), i \neq j$ is the arc set. A cost or travel time c_{ij} is associated with the arc (v_i, v_j). In this work the terms cost, travel time and distance are interchangeable since it is assumed that the UAVs fly at a constant speed and at a constant altitude. However, the problem is not fully symmetrical since the UGVs or depots are moving during the operations. Therefore, $c_{ij} \neq c_{ji}$ when $i \in V_d(t)$. The fleet of UAVs is considered as homogeneous meaning that all the UAVs have the same constant airspeed v_{UAV} and maximum endurance D. The maximum endurance constraint also governs the total duration of each route of the final routing plan in order not to exceed this maximum endurance value D. On each ground vehicle $l \in V_d$, a fleet of m_l UAVs are initially available and occupying m_l recharging stations. The complete fleet of UAVs is composed of $K = \sum_{l=1}^{N_u} m_l$ vehicles. When two UAVs need to be launched from the same UGV, a variable penalty time p_{launch} is considered to take into account the time needed to operate successively both UAVs. Each route h is characterized by the list of targets it contains $h = [h(1), \dots, h(N_h)]$ where N_h is the number of targets visited by the route h. Hence define the coefficients e_{ih} and f_{hl} as follows:

$$e_{ih} = \begin{cases} 1 & \text{if target } i \text{ is on route } h \\ 0 & \text{otherwise} \end{cases}$$

$$f_{hl} = \begin{cases} 2 & \text{if route } h \text{ starts and ends at depot } l \\ 1 & \text{if route } h \text{ starts or ends at depot } l\text{, but not both} \\ 0 & \text{otherwise} \end{cases}$$

Let T denotes the set of all routes h in the routing plan. The formulation uses then binary variable x_k^h to allocate a route $h \in T$ to a specific vehicle k.

$$\mathrm{x_h^k} = \begin{cases} 1 & \text{if and only if route } h \in T \text{ is assigned to vehicle } k \\ 0 & \text{otherwise} \end{cases}$$

Another couple of binary variable y_l^k and z_l^k are used to allocate each vehicle respectively to the UGV where it is initially launched and to the UGV where it lands at the end of its mission:

$$\mathrm{y_l^k} = \begin{cases} 1 & \text{if and only if route of vehicle } k \text{ starts from UGV } l \\ 0 & \text{otherwise} \end{cases}$$

$$z_l^k = \begin{cases} 1 & \text{if and only if route of vehicle } k \text{ ends at the UGV } l \\ 0 & \text{otherwise} \end{cases}$$

Let now π_h denotes the travel duration of route h:

$$\pi_h = \sum_{k=1}^{K}\sum_{l=1}^{N_u} x_h^k y_l^k c_{(N_t+l)h(1)}\,(t = t_0) + \sum_{i=1}^{N_h-1} c_{h(i)h(i+1)} + \\ + \sum_{k=1}^{K}\sum_{l=1}^{N_u} x_h^k z_l^k\, c_{h(N_h)(N_t+l)}(t = t_0 + \pi_h)$$

where t_0 is the initial time when the first UAV is launched (it is further assumed that $t_0 = 0$). The travel duration is given by the addition of three terms: the first one is the cost to go from the UGV where the UAV is launched to the first target to visit in route h; the second one is the total cost to go from the first target to the last target of route h, while visiting all targets within the route h; the last one is the cost to go from the last target to visit in route h to the UGV where the UAV lands after completion its mission. Since the position of the UGV where the UAV lands at the end of its route depends itself on the travel duration of the route h, the cost to go from the last target to visit in route h to the UGV where the UAV lands varies also with the travel duration. Consequently, it can be highlighted that the estimation of the travel duration π_h of one route h needs to be calculated iteratively. The mathematical formulation of the problem is then given by (1):

$$\text{Minimize} \quad \sum_{k=1}^{K}\sum_{h=1}^{|T|} \pi_h x_h^k \tag{1}$$

subject to the following constraints:

$$\sum_{k=1}^{K}\sum_{h=1}^{|T|} e_{ik} = 1 \qquad \forall i \in \{1, \dots, N_t\} \tag{2}$$

$$\sum_{l=1}^{N_u} y_l^k \le 1 \qquad \forall k \in \{1, \dots, K\} \tag{3}$$

$$\sum_{l=1}^{r} f_{hl} = 2 \qquad \forall h \in T \tag{4}$$

$$\sum_{k=1}^{K} y_l^k = \sum_{k=1}^{K} z_l^k \qquad \forall l \in \{1, \dots, N_u\} \tag{5}$$

$$\pi_h \le D \qquad \forall h \in T \tag{6}$$

The constraint defined by (2) ensures that each target is visited exactly once, whereas constraint (3) guarantees that at most one route will be assigned to every UAV. Constraint (4) ensures that each UAV which is launched from a UGV returns on an UGV at the end of its mission, no matter whether it is the same UGV or not. Constraint (5) makes sure that for each vehicle, the number of UAVs launched from this UGV and the number of UAVs returning to this UGV are equal. Finally, (6) constraints the routes to be flyable by one UAV with respect to its maximum endurance.

4. RESOLUTION METHODOLOGY

To solve the problem described in Section 3, a simulation of the scenario is implemented using MATLAB/Simulink. In the present section, a detailed resolution methodology is proposed to solve this time-constrained multiple depot vehicle routing problem with moving depots and inter-depots routes.

The solution is obtained by using a sequential algorithm which means that the algorithm builds one route at a time. After a new route has been generated, a local search meta-heuristic based on TS runs to optimize the routing with the current number of routes. This optimization phase also looks for the best solution with the current number of routes in terms of cost which satisfy all the constraints and in particular the maximum endurance constraint of the UAVs. If after this optimization process, the best solution found so far does not satisfy all the constraints, a new route is generated and the same loop is applied with an additional route. Finally, at the end of this loop a wider TS is performed on each route independently to check for better neighbour solutions. To use the TS meta-heuristic, it is required to generate first an initial solution. At the first step of the routine, the initial solution consists of one route visiting all the targets generated by a heuristic method without taking care of the maximum endurance constraint. For the next iterations of the routine, the TS algorithm uses as initial solution the best solution found so far for the routing plan in terms of costs. This routine is illustrated in Fig. 3.

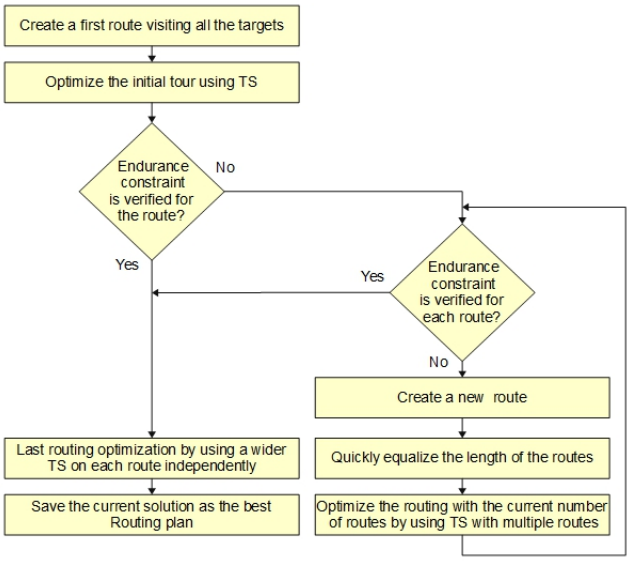


Fig. 3. General sequential methodology to solve the time-constrained multiple depot vehicle routing problem.

The initial solution composed of only one route can be generated by using various heuristics. This first solution is also obtained by using a sequential nearest neighbour heuristic. The criteria to choose the UGV from where the UAV is launched is based on a clustering of the targets. First, all the targets are associated to their nearest UGV. The UGV which is associated to the greatest number of targets is then chosen as the UGV from where the UAV is launched if the capacity of the UGV in terms of UAV docked on it enables it to do so (meaning here that there is at least one UAV docked on the UGV). The capacity in terms of available UAVs of this UGV is also decreased by 1. Then, the heuristic generates a route by applying a greedy algorithm as in a classical TSP. This algorithm allocates tasks to the UAV so that it starts from the determined UGV and then repeatedly visits the nearest target until all have been visited. Since only one UAV is launched, the only free recharging station is the one left by the UAV. Thus, the UAV returns to the same UGV at the end of its

routing plan. This UGV will now be located at a different position than its initial position since it has moved while the UAV was flying. To compute the final position of the UGV, an iterative calculation method is implemented.

4.1 Tabu search algorithm for a single route

After generating a first routing composed of a single route, it is necessary to optimize this initial solution. This optimization phase is performed by a TS algorithm specifically designed to optimize the path of a single route which visits all the targets associated to this route. This problem is also equivalent to a TSP with a moving depot. Moreover, this TS algorithm will be further used in the intensification process of the global TS algorithm which solves the general problem with multiple routes. Since this algorithm is used quite extensively during the optimization search, the structure of the neighbourhood has to be relatively simple in order to find a good compromise between the quality of the neighbourhood and computational efficiency. Thus, only one type of movement is implemented for this TS algorithm. The only explored movements in the search area are swap moves between two targets within the route that is optimized. This belongs also to one of the moves referred to as 1-opt. For each swap move between two targets i and j ($i \neq j$), the cost of the corresponding neighbour solution is gathered in an $N_h \times N_h$ matrix called *neighbourhood cost matrix* $M_{nbr,cost}$. The neighbourhood is then explored by choosing the best movement among the whole neighbourhood corresponding to the move associated to the lowest value in $M_{nbr,cost}$. To avoid cycling, a short-term memory is implemented within this TS algorithm. Every swap move is characterised by two targets which define its attributes. A $N_h \times N_h$ matrix called *tabu tenure matrix*, $M_{tabutenure}$, makes then an inventory of the last performed swap moves in order to avoid reverse moves. When a swap move between two targets i and j ($i \neq j$) occurs, this move becomes *tabu* during the next θ iterations and $M_{tabutenure}(i,j) = M_{tabutenure}(i,j) = \theta$. This means that this permutation is not allowed during the next θ iterations.

Based on investigations in the literature, θ is defined as a function of the number of targets within the route, $= \sqrt{N_h}$, where N_h is the number of targets within the route h. To promote the diversification, a long-term memory is used. It consists of a frequency-based memory. A $N_h \times N_h$ matrix called *frequency matrix*, M_{freq}, gathers the occurrences of each move. Consequently, the algorithm can find the most frequent moves and penalize them in the long-term to force the diversification. After a number ζ of iterations without any improvement (ζ = 25 by default), the algorithm forces the current solution to apply one swap move which has one of the lowest frequency, in order to explore differently the solution space. The algorithm stops when a stopping criterion is met which is a predefined number of iterations $N_{iter,TS}$ and gives as output the optimized route. At the first iteration of the global routine, if the optimised single route of the routing plan is infeasible and does not respect the maximum endurance constraint of the UAV, a new route is generated. Then, an algorithm is implemented to balance the utilisation of the UAVs, so that all the UAVs are associated to routes of approximately same length. This process also aims to quickly allocate missions to the different UAVs that are about the same flight duration. This algorithm generates as output a solution which will be then used by the TS algorithm with multiple routes.

4.2 Tabu search algorithm for multiple routes

Once a new route has been generated, the solution is optimized by using a TS algorithm with inter-routes moves. This more evolved version of the TS algorithm has nevertheless many similarities in its structure with the TS algorithm designed for single route problems described earlier. Indeed, the classical steps of a TS algorithm are also implemented. First, the algorithm starts from the initial routing plan. Then, it explores the neighbourhood to search for better solutions by using moves between the targets. The main specificity of this algorithm relies on its different neighbourhood structure since it involves multiple routes. Consequently, the neighbour solutions can be either obtained by applying *inter-routes moves* (involving two different routes) or *intra-route moves* (involving only one single route). To obtain a better quality of the neighbourhood, two different kind of moves are explored in this algorithm: swap moves (permutation of two targets) and relocation moves (insertion of one target after a designated one). Both moves can be either inter-routes or intra-route. To limit the computational cost a strategy similar to the one developed by Gendreau et al. (1994) is implemented to apply sequentially the different moves between one target from the first selected route and the p-closest targets from this target within the second selected route. The algorithm also computes the p-neighbourhood of the first selected target. By default, $p = p_{intra} = 3$ for intra-routes moves and $p = p_{inter} = 2$ for inter-routes moves.

For each UAV involved in the moves applied to the current solution, the locations of the UGVs when the UAV need to land are re-computed. To explore a wider region of the solution space, the algorithm checks the interest in terms of cost to change the UGV from where one route starts. One route can be re-allocated to another UAV based on another UGV, if the first target to visit in the route is exchanged by another one while applying either a swap move or a relocation move. When such moves are performed, the algorithm reallocates the route to the closest UGV, with at least one available UAV, to this new first target. As suggested by Gendreau at al. (1994), intermediate infeasible solutions are allowed during the search. The same technique which consists in adding the constraints in the objective function is implemented in the present neighbourhood search. A penalized objective function given by (7) evaluates the cost of the global routing with regards to the respect of the different constraints and secondary objectives.

$$c'(x) = \kappa \sum_{h \in T} \pi_h + \lambda \max_{h \in T} \pi_h + \alpha \sum_{h \in T} (\pi_h - D) + \\ + \beta \sum_{h \in T} (\pi_h - \widetilde{m}) \qquad \text{where } \widetilde{m} = \frac{\sum_{h \in T} \pi_h}{|T|} \qquad (7)$$

A flexibility on the objectives and the constraints is also implemented thanks to the tunable parameters α, β, κ, λ. By default, $\alpha = 1$, $\beta = 1$, $\kappa = 1$ and $\lambda = 0$. This parametrisation

means that the main objective is the minimization of the sum of the length of the routes within the routing plan. The term linked to α penalizes infeasible neighbours which do not respect the maximum endurance constraint while the term associated to β penalizes routing plans with route lengths which are not well balanced. These penalization terms are generally much lower than the main term. This statement makes them relevant and efficient to guide the optimization process through better solutions while allowing infeasible solutions which significantly minimizes the main objective function.

A diversification process is also implemented by varying the value of α and β. This process also regulates the possibilities of intermediate infeasible solutions. Initially, α and β are set equal to 1 and, at each iteration, if the last μ solutions were all feasible (or all infeasible) regarding to the associated constraint they are either reduced (or increased) by dividing (or multiplying) them by a factor $(1 + \delta)$. The parameters μ and δ are respectively set by default equal to 5 and 0.01. For α, a solution is considered as feasible if the maximum endurance constraint is strictly respected for all routes. For β, a solution is said feasible if the routes are well balanced. To avoid cycling, a short-term memory slightly different from the previous one is implemented within this TS algorithm. Every swap moves is characterised by two targets which define its attributes, while relocation moves only have a single attribute given by the relocated target. A $N_u \times N_t \times m$ tensor called *tabu tenure tensor* $T_{tabutenure}$ makes then an inventory of the last performed moves in order to avoid reverse moves. This short-term memory avoids the targets implied by the selected move to be re-inserted in their original route for q iterations. It also aims to explore the solution space by investigating further some specific directions of the solution space.

When a swap move between two targets i and j $(i \neq j)$ occurs, all moves which would imply one of the target to be re-inserted in their original route become *tabu* during the next q iterations. The same process is applied for targets involved in relocation moves. Then, at each iteration, the tabu tenures of all the forbidden moves are decreased by 1 until they reach the value 0 to be permitted again. As in the previous TS algorithm, θ is defined as a function of the number of targets within the route. The long-term memory consists of exactly the same frequency-based memory used to diversify the search in the TS algorithm for single route. Comparing to the first TS algorithm, an intensification process is also implemented in the present algorithm. This intensification process aims to intensify the search in promising areas of the solution space. This intensification occurs when the current solution has a lower cost than the best solution found so far and thus replaces it. In that case, a post-optimization process enables to quickly check if better solutions can be found from this new solution by applying a TS algorithm for single route on each route of the new solution. This intensification also checks if better solutions can be obtained by applying only intra-route swap moves on each route independently. The possibility to add a target has been implemented within this algorithm. Since the search optimization can take a long time for larger size problems with many targets, it is important to have the possibility to add a target during the search optimization process.

5. VALIDATION AND RESULTS

To validate the resolution methodology presented in the previous section, several tests were conducted to check the performances of the implemented algorithms and to select the different tunable parameters to optimize the search. The performances of the different algorithms used to solve the problem have been tested. In particular, they showed that, for both TS meta-heuristic algorithm for a single route and TS meta-heuristic algorithm for multiple routes, using a smaller neighbourhood structure gives a good compromise between computational efficiency and the quality of the solution. In most cases, the TS algorithm for a single route managed to find a solution close to the one found with the complete neighbourhood structure with less than 7% of cost difference in only half of the time. Moreover, for the TS algorithm for multiple routes, using smaller neighbourhood structures does not significantly affect the search to find a good solution. In the majority of cases, when the algorithm uses smaller neighbourhood structures, it manages to find equivalent solutions in terms of cost to the solution found with the greatest neighbourhood structure (with less than 5% of difference). In some particular scenarios, using smaller neighbourhood structures can force the search to guide the solution towards a better solution than it has found with the complete neighbourhood structure. The global minimum of the objective function is indeed not necessarily obtained by passing by all the local minima. Consequently, it is interesting in terms of efficiency to use smaller neighbourhood structures because it divides by more than two the computational load but still manages to find relatively good solutions in terms of costs. In this section, the general analysis of the final solutions will be presented.

5.1 Algorithm performances

Tests were designed to analyse the solutions provided by the global resolution methodology. The following test gives as output the average total cost of the routing plan, the number of UAVs required and the average computational load. It was conducted by evaluating the solution for 5 different scenarios for each number of targets tested. For each number of targets, the tests gives the average of the outputs for the five different scenarios and error bars represent minimum and maximum values of these outputs. During this test, the computational load is monitored by managing the number of iterations performed by the two different TS algorithms. For the TS algorithm for single route, the parameters chosen were: $p = 5, n = N_t$ and $t_{max,TS} = 10s$. For the TS algorithm for multiple routes, the parameters chosen were: $p_{intra} = 3$, $p_{inter} = 2$, $n = N_t$, $\delta = 0.01$ and $t_{max,TS} = 180s$ (routing composed of two routes) or $t_{max,TS} = 300s$ (otherwise). Fig. 4 shows the obtained results. It provides a good estimation of the average number of UAVs (with 25 mins of maximum endurance) required to cover a square search area of 12 km side with a predefined number of targets and three UGVs. Depending, on the scenario, the number of UAVs can vary. With greater number of targets, the number of UAVs required

is more likely to have greater variations, since the targets can be more or less spread out throughout the search area. Similar results could be obtained for different parameters in order to size the fleet of UAVs needed depending on the search area and the number of UGVs. Moreover, Fig. 4 shows that this resolution methodology manages to find a solution of acceptable quality to this NP-hard problem in a relatively linear computational time. For greater numbers of targets, the variations of computational loads for different scenarios increase since intensification process occur more often because the algorithm is more likely to find some ameliorations through the search optimization with TS for multiple routes. For some specific numbers of targets such as $N_t = 20$, Fig. 4 shows that depending on the scenarios, a different number of UAVs is required for the same number of targets (either two or three UAVs for $N_t = 20$).

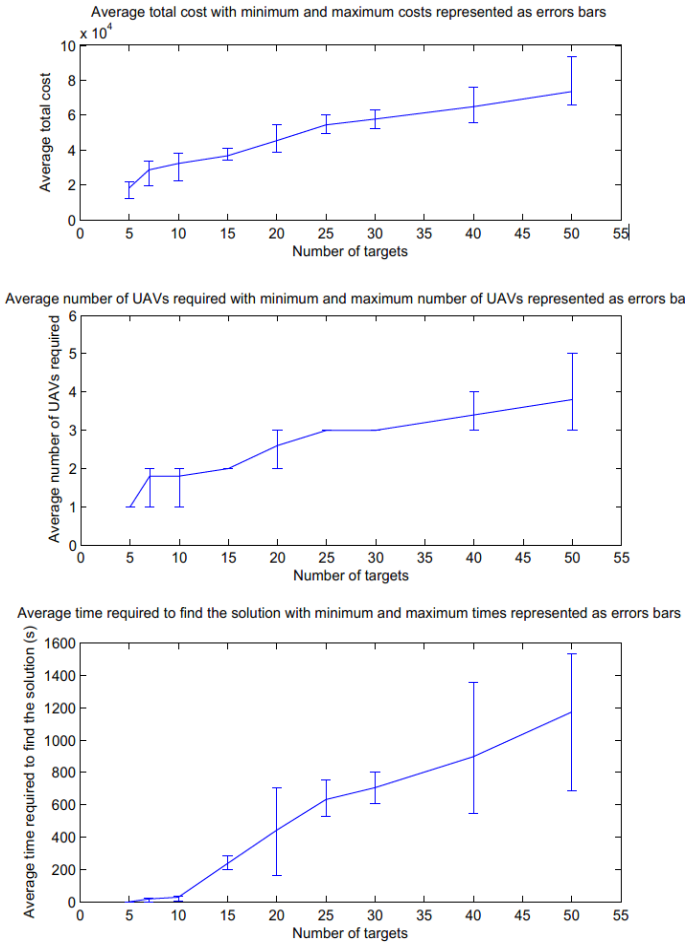


Fig. 4. Comparison of the solutions provided by the global resolution methodology for different numbers of targets.

A test was implemented to compare the efficiency of the algorithm for variable neighbourhood structures. In this test, it was considered that $p = p_{inter} = p_{intra}$. For 15 scenarios with $N_t = 26$, this test compares the costs of the routing plan and the time needed to find the best solution for variable neighbourhood structures. The results are depicted in Fig. 5.

To investigate the performance of the TS algorithm with multiple routes, another test was conducted which evaluates the success of the short-term memory and the diversification process along the search. This test has been conducted for only one scenario for different values of δ which is the coefficient which adjusts the penalizing coefficients α and β in the penalized objective function. The penalizing coefficients α and β are bounded between 0 and 10. The results can be seen in Fig. 6. Generally speaking, Fig. 6 shows that the short-term memory and the diversification process manage successfully to explore a wide area of the solution space since no strict periodicity is observed on the cost of the current solution. This means that the search explores different areas.

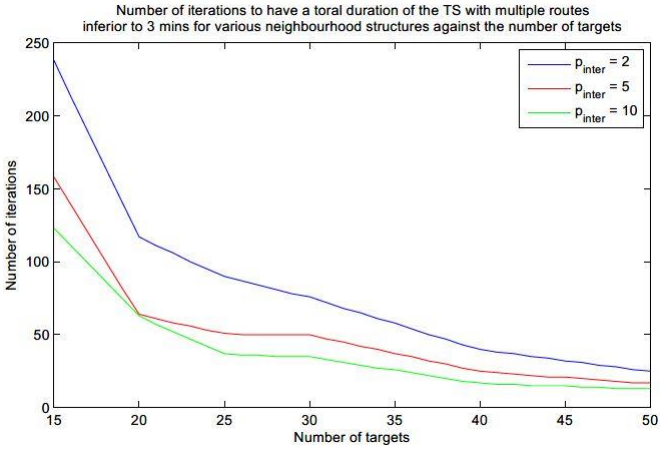


Fig. 5. Maximum number of iterations to perform the TS algorithm for multiple routes in less than $t_{max,TS} = 180s$ for various neighbourhood structures against the number of targets.

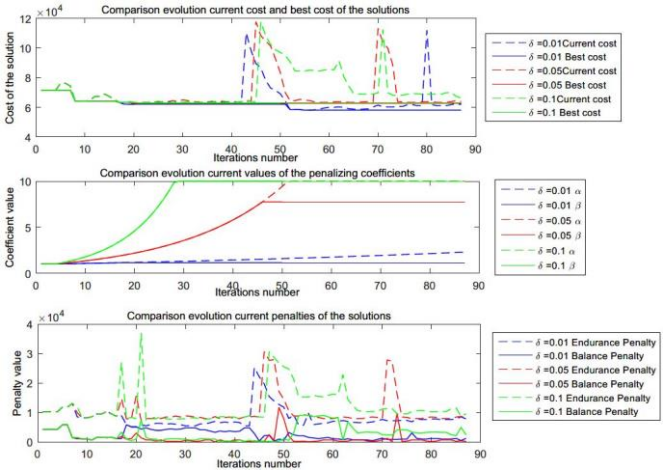


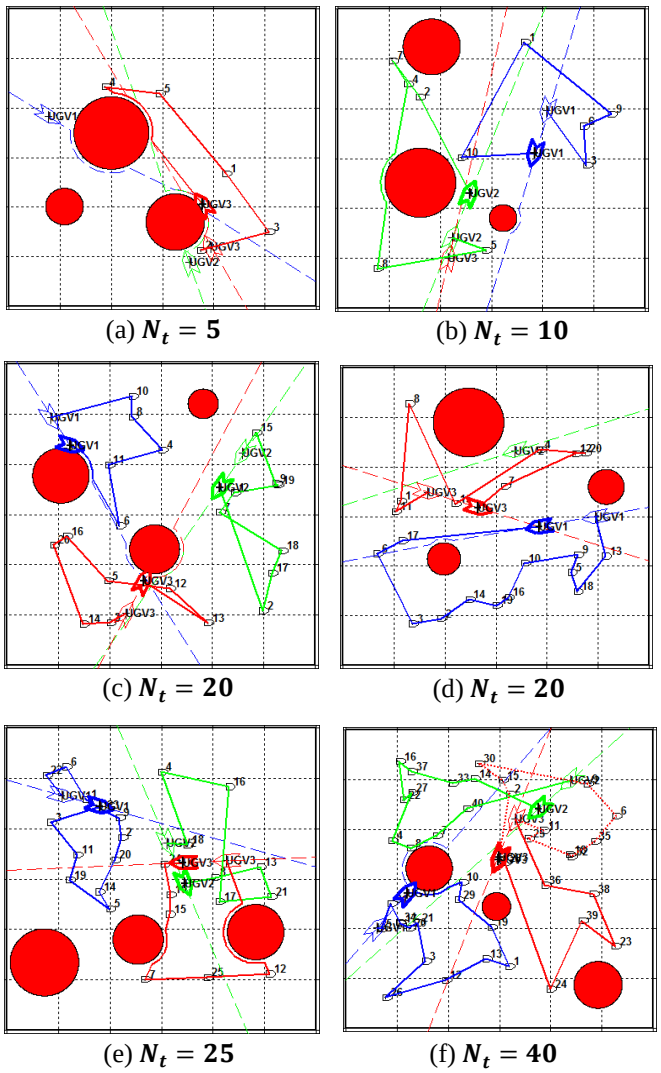
Fig. 6. Comparison of the performance of the TS algorithm for multiple routes – influence of δ.

The reactivity of this RTS algorithm is also highlighted since the solution is perturbed every 25 iterations without any improvement. The influence of δ is shown in Fig. 6. A good compromise has to be done for this parameter. To modify significantly the importance of the penalties towards the search do not seem to lead necessarily towards better results but is just another way to diversify the search.

5.2 Results

The most relevant and interesting maps representing the best solution for different number of targets without inter-UGVs routes are reported in Fig. 7. The coloured large arrows, the small black blocks and the red circles represent the UGVs, the

targets and the obstacles, respectively. The UGVs, the targets and the obstacles are randomly positioned on the map. Each route is represented by a broken line of the same colour as its starting UGV. The final position of the UGV when a UAV lands on it is displayed by a coloured arrow with larger linewidth.



(a) $N_t = 5$ (b) $N_t = 10$

(c) $N_t = 20$ (d) $N_t = 20$

(e) $N_t = 25$ (f) $N_t = 40$

Fig. 7. Most relevant and interesting maps representing the best solution for different number of targets.

It is possible to notice that, depending on the scenarios, a different number of UAVs is required for the same number of targets (Fig. 7(c), 7(d)).

6. CONCLUSIONS

Although a large panel of research exists on the VRP and its variants in the context of UAVs, only a few papers have tackled the dynamic aspects of this specific environment. Among them, the majority has focused on either VRPTW, moving targets or threat uncertainties. Only a few papers have incorporated multiple depots into the solution algorithms. Up to now, it seems that in particular, any of these papers has addressed the problem of moving depots. In the scenario proposed in this work, this side constraint is very meaningful because the depots represent the UGVs where UAVs are initially docked on recharging stations. Once a UAV is launched from the ground vehicle, the ground vehicle keeps doing its own mission and is also moving while the UAV is flying. Therefore, the ground vehicles or depots will not be located at the same place when the UAV will need to recharge. Consequently, in this paper, a new approach to solve a specific variant of the VRP in the context of UAVs has been described. The specificity of the generic scenario relies mainly on the maximal endurance constraint of the UAV and the hypothesis that the UGVs are moving while the UAVs are flying. Results obtained on a series of tests indicate promising performances of the proposed resolution methodology based on TS. Since TS is a meta-heuristic that must be well sized and designed for each particular problem, analysis have been conducted to show how to select the different tunable parameters. The success of this algorithm is mostly based on two main features. First, it allows intermediate infeasible solutions through penalties terms in the objective function, which consequently can provide a better exploration of the solution space. Secondly, the diversification process also enables to regularly perturb the search to explore other regions of the solution space. Moreover, the flexibility on the starting UGVs of the routes during the neighbourhood exploration enables to explore a wider area of the solution space and often leads to better solutions.

REFERENCES

Carlton W. B., A Tabu Search Approach to the General Vehicle Routing Problem. PhD thesis, 1995.

Cordeau J. F., Laporte G., Savelsbergh M. W. P., and Vigo D., "Chapter 6 Vehicle Routing," *Transportation*, vol. 14, no. 06, pp. 367–428, 2007.

Crevier J. C. B. and Laporte G., "The Multi-Depot Vehicle Routing Problem with Inter-Depot Routes," 2005.

Dantzig A. G. B., Ramser J. H., Science S. M., and Oct N., "The Truck Dispatching Problem Stable," *Management*, vol. 6, no. 1, pp. 80–91, 2010.

Freitas J. A. D., "A Tabu Search Metaheuristic Algorithm for the Multiple Depot Vehicle Routing Problem with Time Windows," no. October, 2012.

Gendreau M., Hertz A., and Laporte G., "A Tabu Search Heuristic for the Vehicle Routing Problem," *Management Science*, vol. 40, no. 10, pp. 1276–1290, 1994.

Glover F., "Future paths for integer programming and links to artificial intelligence," *Computers and Operations Research*, vol. 13, no. 5, pp. 533 – 549, 1986. Applications of Integer Programming.

Leonard J., "Ant colony optimization for dynamic multiple travelling salesmen problem," *Journal of intelligent and robotic systems*, 2015.

Montoya-torres J. R., Lopez J., Nieto S., Felizzola H., and Herazo-padilla N., "A literature review on the vehicle routing problem with multiple depots," *Computers and Industrial Engineering*, vol. 79, pp. 115–129, 2015.

Winston W. L., Venkataramanan M. A., *Operations research, Introduction to mathematical programming*, vol. 1. Pacific Grove, CA: Brooks/Col, 2003.

https://dspace.org/

Use of co-operative UAVs to support/augment UGV situational awareness and/or inter-vehicle communications

Quintin, F.; Iovino, Serena; Savvaris, Al; Tsourdos, Antonios

2017-10-18
Elsevier
https://doi.org/10.1016/j.ifacol.2017.08.1229
F. Quintin, S. Iovino, A. Savvaris, A. Tsourdos, Use of Co-operative UAVs to Support /Augment UGV Situational Awareness and/or Inter-Vehicle Communications. IFAC-PapersOnLine, Volume 50, Issue 1, 2017, Pages 8037-8044

Downloaded from a DSpace repository. https://dspace.org/