Proceedings of the 17th World Congress
The International Federation of Automatic Control
Seoul, Korea, July 6-11, 2008

Large-scale Task/Target Assignment for UAV Fleets Using a Distributed Branch and Price Optimization Scheme ⋆

Sertac Karaman * **Gokhan Inalhan** **

* *Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: sertac@mit.edu).*
** *Faculty of Aeronautics and Astronautics, Istanbul Technical University, Istanbul, 34469 TURKEY (e-mail: inalhan@itu.edu.tr)*

Abstract: In this work we consider the large-scale distributed task/target assignment problem across a fleet of autonomous UAVs. By using delayed column generation approach on the most primitive non-convex supply-demand formulation, a computationally tractable distributed coordination structure (i.e. a market created by the UAV fleet for tasks/targets) is exploited. This particular structure is solved via a fleet-optimal dual simplex ascent in which each UAV updates its respective flight plan costs with a linear update of way-point task values as evaluated by the market. We show synchronized and asynchronous distributed implementations of this approximation algorithm for dynamically changing scenarios with random pop-up targets. The tests performed on an in-house built network mission simulator provides numerical verification of the algorithm on a) bounded polynomial-time computational complexity and b) hard real-time performance for problem sizes on the order of hundred waypoints per UAV.

1. INTRODUCTION

During the last decade, Unmanned Air Vehicles (UAVs) have enjoyed practical success at scenarios involving reconnaissance, surveillance and active tracking. Two of the major drivers that make such unmanned systems favorable over manned vehicles are physical working conditions (such as remote locations with harsh terrain or chemical/radioactive spill) and the scenario constraints (such as uninterrupted day long service) that make the operation of manned systems or vehicles impractical, risky or just cost ineffective.

With the ever growing involvement of UAVs in complex application areas (such as dynamically changing urban rescue operations), the types and the number of tasks easily outgrow one vehicle's (or a set of UAV operators' command and control) limited capabilities and thus require a fleet of UAVs working autonomously in a collaborative fashion to achieve desired operational goals. In this work, we show a distributed optimization method that allows collaborative task/target assignment across such autonomous UAV fleets for cases in which the number of tasks/targets [1] is very large (i.e. on the order of tens to hundreds per UAV) and dynamically change.

For large-scale assignment problems with limited resources, the optimal/near-optimal allocation of such resources becomes a crucial deciding factor for the success of a mission. The problem of task assignment in optimization framework is inherently a binary integer linear program (a special case of mixed integer linear programming (MILP)) and has been used widely for formulating UAV task assignment problems [4,5]. In this work, we exploit a structural decomposition of the most general form

Fig. 1. The in-house developed network mission simulator allows real-time simulation across interoperable manned-unmanned fleets, and the mission control center.[25]

of this task assignment formulation through column generation. The column generation approach essentially provides a method for breaking up column-dense large scale optimization problems into subcomponents that are integrated to a growing master problem which is optimized one step at a time. Together with [8] a good survey of column generation in integer programming is presented in [17]. We refer the reader to the use of column generation technique in integer programming for many other problems as outlined in [15] and [16].

There has been considerable effort in literature to use column generation with shortest path problems [11] and vehicle routing problems with time windows [12], [13], [14]. However these solution formulations are aimed at solving the centralized assignment problem in step-by-step progressive fashion in com-

⋆ The work is funded in part by DPT HAGU program administrated by ITU-ROTAM.

[1] Through out the discussion, we will use "tasks" and "targets" interchangeably, both meaning a waypoint (within the flight plan) which corresponds to an activity.

parison to the distributed implementation structure as illustrated in this work.

Specifically, in Section II, we reconstruct the most primitive non-convex supply-demand form of the assignment problem with extreme point solutions that conveniently correspond to selection of waypoints each of the UAVs. This column-dense extreme point formulation is used for employing a delayed-column generation optimization scheme which involves (a restricted form of) a master and sub-problem formulations. Both of these steps are shown to be implementable at the local computer of any of the UAVs requiring only communication for passing flight path information(i.e. specific waypoint selections) and the costs associated with these flight paths. This two-step solution corresponds to a fleet-optimal dual simplex ascent method in which each UAV updates its respective flight plan costs with a linear update of way-point task values as evaluated in the market created by the UAVs for the targets. We devote Section III for algorithmic implementation details of this method on the standard fleet-optimal shortest path target assignment problem. In addition, we provide insight on the real-time implementations on an in-house built network simulator shown in Fig. 1.

2. MAIN TASK/TARGET ASSIGNMENT PROBLEM

The m vehicle n target assignment problem, shown in Eq. 1, pays close resemblance to the general "transportation problem" [6] with notably two differences :

Problem A:

$$\begin{aligned} \min\ & c_1(y_1)+c_2(y_2)+\cdots+c_m(y_m) \qquad (1)\\ \text{subject to}\quad & Iy_1+Iy_2+\ldots Iy_m = 1_{n\times 1}\\ & \sum_j y_{1j} \geq 1\\ & \sum_j y_{2j} \geq 1\\ & \vdots\\ & \sum_j y_{mj} \geq 1\\ & y_{ij}\in\{0,1\} \end{aligned}$$

First, is the nonlinear cost structure $c_i(y_i)$ for each i_{th} vehicle in which y_i corresponds to a selection of targets (i.e. waypoints). This nonlinear form is a mask for entailing both each of the vehicles' internal constraints and also the nonlinear dependence of choice of an arbitrary way-point assignment such as $y_i = [0\ 1\ 1\ 0\ldots 0]'$ to the corresponding cost such as the shortest path : $c_i(y_i) = \textit{shortest path between the way-points selected} - y_i$. Second, is the nature of the supply constraint which demands that the "supplier should at least supply one unit", i.e. each UAV should at least visit at least one target.

For the LP relaxation of the problem, one can consider $y_{ij} \in \{0,1\} \rightarrow 0 \leq y_{ij} \leq 1$. [2]

The integer solution type of this particular problem suggest the well-known Dantzig-Wolfe decomposition to explore a distributed structure of the Problem A. Specifically, using the resolution theorem, any feasible way-point selection for each UAV within its respective bounded polyhedron region $P_i = \{\sum_j y_{ij} \geq 1; 0 \leq y_{ij} \leq 1\}$ can be described as:

$$y_i = \sum_{q\in Q_i} x_i^q y_i^q$$

with convexity constraints $\forall i = 1,\ldots,m$

$$\sum_{q\in Q_i} x_i^q = 1; \quad x_i^q \geq 0$$

Here y_i^q correspond to the extreme point solutions or just the selection of way-points that will be visited in a particular flight plan. Q_i is the set of all the extreme point solutions or just the collection of all combinations of possible way-point selections. x_i^q is the convex combination coefficients for a particular q_{th} way-point combination.

Using this explicit solution representation, the original formulation's cost components can be transformed via $c_i(x_i^q y_i^q) = x_i^q c_i(y_i^q) = x_i^q c_i^q$ and $d_i^q = Iy_i^q = y_i^q$ is again just the selection of way-points that will be visited in a particular flight plan. In Dantzig-Wolfe decomposition, this is known as the master problem :

$$\begin{aligned} \min\ & \sum_{q\in Q_1} c_1^q x_1^q + \cdots + \sum_{q\in Q_m} c_m^q x_m^q \qquad (2)\\ \text{subject to}\ & \sum_{q\in Q_1} d_1^q x_1^q + \cdots + \sum_{q\in Q_m} d_m^q x_m^q = b_o = 1_{m\times 1}\\ & \sum_{q\in Q_1} x_1^q = 1\\ & \vdots\\ & \sum_{q\in Q_m} x_m^q = 1\\ & x_i^q \geq 0 \quad \forall q\in Q_i,\ \ i=1,\ldots,m \end{aligned}$$

or in compact form :

Master Problem:

$$\begin{aligned} \min\ & c_1' x_1 + \cdots + c_m' x_m \qquad (3)\\ & D_1x_1 + D_2x_2 + D_mx_m = b_o\\ & f_1' x_1 = 1\\ & \vdots\\ & f_m' x_m = 1\\ & x_i^q \geq 0 \quad \forall q\in Q_i,\ \ i=1,\ldots,m \end{aligned}$$

where $f_i = [1\ldots 1]'$ is of appropriate length. $D_1 = [d_1^1\ |d_1^2|\ldots d_1^q\ \ldots]$, $q \in Q_1$.

Before describing the sub-problems, note that the number of variables in the Master Problem is much larger than the number of constraints. For this we consider solving the Master Problem by keeping only a small number of variables in the problem that is adequate for forming the basis for simplex formulation. We call this problem with less number of variables the Restricted Master Problem (RMP). [3] i.e.,

[2] Note that, in explicit formulation there is no need for upper-bounding y_j because it is naturally bounded by the interconnecting assignment constraint. For guaranteed integer valued solutions in LP relaxed form refer to [1].

[3] The compact form of the RMP can be represented as :

Restricted Master Problem:

$$\begin{aligned}
\min \quad & \sum_{q\in\bar{Q}_1} c_1^q x_1^q + \cdots + \sum_{q\in\bar{Q}_1} c_m^q x_m^q \\
\text{subject to} \quad & \sum_{q\in\bar{Q}_m} d_1^q x_1^q + \cdots + \sum_{q\in\bar{Q}_m} d_m^q x_m^q = b_0 \\
& \sum_{q\in\bar{Q}_1} x_1^q = 1 \\
& \vdots \\
& \sum_{q\in\bar{Q}_m} x_m^q = 1 \\
& x_i^q \geq 0 \quad \forall q \in Q_i, \;\; i = 1,\ldots,m
\end{aligned}$$

where $\bar{Q}_i$ is a subset of Q_i. The RMP in this case includes a smaller number of columns when compared to the MP. For this reason it is called as the restrictive master problem. If we let π_j be the dual variables associated with n coupling constraints and μ_i be the dual values associated with the m convexity constraints and define B as :

$$B = \begin{bmatrix} \bar{D}_1 & \bar{D}_2 & \ldots & \bar{D}_m \\ [1\ldots1] & [0\ldots0] & [0\ldots0] & [0\ldots0] \\ [0\ldots0] & [1\ldots1] & [0\ldots0] & [0\ldots0] \\ \vdots & \vdots & \vdots & \vdots \\ [0\ldots0] & [0\ldots0] & [0\ldots0] & [1\ldots1] \end{bmatrix} \tag{4}$$

For the Restricted Master Problem the primal and the dual solutions is given as [1]:

$$x_B = B^{-1}b \tag{5}$$

$$[\pi \quad \mu]' = [\bar{c}_1' \ldots \bar{c}_m']' B^{-1} \tag{6}$$

In addition, the dual formulation of the RMP can be uniquely defined as,

Dual of Restricted Master Problem :

$$\begin{aligned}
\max \quad & \sum_{j=1}^{n} \pi_j + \sum_{i=1}^{m} \mu_i \\
\text{subject to} \quad & (d_1^q)'\pi + \mu_1 \leq c_1^q \;\; q \in \bar{Q}_1 \\
& \vdots \\
& (d_m^q)'\pi + \mu_m \leq c_m^q \; q \in \bar{Q}_m
\end{aligned} \tag{7}$$

Note that there is one dual variable π_j associated with each target for $j = 1,...,n$ and one dual variable μ_i associated with each UAV for $i = 1,...,m$.

$$\begin{aligned}
\min \quad & \bar{c}_1'\bar{x}_1 + \cdots + \bar{c}_m'\bar{x}_m \\
& \bar{D}_1\bar{x}_1 + \bar{D}_2\bar{x}_2 + \bar{D}_m\bar{x}_m = b_o \\
& \bar{f}_1'\bar{x}_1 = 1 \\
& \vdots \\
& \bar{f}_m'\bar{x}_m = 1 \\
& \bar{x}_i^q \geq 0 \quad \forall q \in \bar{Q}_i, \;\; i = 1,\ldots,m
\end{aligned}$$

Our aim is to solve the dual of the RMP to find the optimal target assignments. However, in order to solve the RMP the problem is to find new columns with negative reduced costs to enter the basis at any simplex ascent. The reduced cost for any column is,

$$c_i^q - [\pi \quad \mu] \begin{bmatrix} d_1^q \\ 0 \\ \vdots \\ 1_i \\ \vdots \\ 0 \end{bmatrix} = c_i^q - \pi d_1^q - \mu_i \tag{8}$$

Considering a pivoting rule as getting the column with the most negative reduced cost to the basis, the problem finding a column to enter the basis can be posed as an optimization problem as:

$$\min_{q\in Q_i} c_i^q - \pi d_i^q - \mu_i$$

or returning back to our original formulation, we can pose this subproblem simply by :

$$\min_{y_i} c_i(y_i) - \pi y_i - \mu_i$$

These problems for $i = 1,...,m$ will be called the sub-problems (SPs). These SPs actually provide a very elegant way to update of flight the costs for each vehicles flight plan with a linear update of way-point values of the market as created by the UAVs. SPs are binary integer linear programs considering the feasible set However, the special structure of the SP allows for it to be interpreted as a search problem, which searches for the minimum value of updated costs. This basic search problem can be solved in polynomial time, although a binary integer linear program is the class of NP-hard problems in general and no polynomial time algorithm is known for its solution. Generally these binary integer linear programs are solved using the branch and bound techniques.

Notice that in its most general form every sub-problem becomes a binary integer program and master iteration can be kept as a linear program. The solution procedure is then to solve the sub-problems and generate new columns if possible, and solve one master iteration to bind up these columns. The master iteration in this case is one iteration of the simplex algorithm which is detailed in [1]. In the master iteration, revised simplex method can be used so that the master problem includes only the columns that are in the basis. Any column that exists the basis is taken out of the master problem. In the next section, we explore this approach for shortest-fleet path problem.

A general discussion on implementing mixed integer column generation is given in [9] using the two main techniques convexification and discretization. These techniques will not be detailed further in this work, but the Dantzig-Wolfe decomposition technique that is used here with Column Generation is a convexification method [16]. Detailed information on these different methods can be obtained from [9].

3. ALGORITHMIC IMPLEMENTATIONS FOR SHORTEST-FLEET PATH PROBLEM

For algorithmic implementations, we consider one of the basic problems : the problem consists of n targets to be visited by m

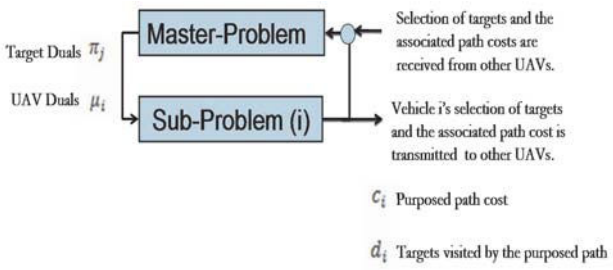

Fig. 2. The synchronized distributed implementation of the main phase II of the algorithm.

UAVs and we would like to find the waypoint selections that results in the minimum total path traveled by the UAV fleet. For easier analysis the problem is examined in two phases.

3.1 Distributed Synchronized

Phase I : Initialization Before the algorithm starts, we assume that all the vehicles communicate their states, i.e. their position in the field and the targets that they have identified and that must be visited. Thus the waypoints to be visited and the position of other UAVs are known by each UAV [4]. There may be some restrictions on some of the waypoints, e.g. a specific waypoint must be visited by a specific UAV which are also all known.

In the initialization phase of the algorithm, each UAV

- Calculates its distance between each and forms $n \times n$ matrix V such that an element V_{ij} is the distance of waypoint i to j. Naturally this matrix is symmetric and all the diagonal elements are zero.
- Computes its distance of every UAV to every waypoint and a $n \times 1$ vector r_i is generated for which r_{ij} is the distance to the j_{th} waypoint.
- Forms its flight plan matrix D_i i.e. the possible combination of waypoints that it can travel
- Find the cost for each waypoint combination (i.e. each vector in D_i) by first calculating the different order of waypoints that can be visited and the calculating the distance for each different ordering using the V_i, r_i. Then, select the minimum cost option (i.e. the shortest path) of such waypoint orderings for all the elements of flight plan matrix form c_i.
- Create the restricted master problem matrix B (in the dual of RMP this refers to B') matrix using the predefined $(n+m) \times (n+m)$ invertible matrix structures in memory. [5]
- Communicate the associated flight cost $\bar{c}_i$ for B to the other UAVs.

Note that this initialization phase is applicable for shortest path problems and can be modified for each application type dependent on the metric to be optimized (such as target priority and target order). In addition one can apply various approximation techniques for shortest path calculation for an arbitrary number of waypoints while providing bounds on optimality.

[4] In real-time applications, we found that this assumption can be relaxed by only identifying the set of overlapping targets

[5] Such initialization matrix selections can be easily created for any combination of n and m and stored in memory. We refer the reader to [1] for selection of such initial matrices.

Phase II : Dual Simplex Ascent Phase two of the algorithm implements the dual simplex ascent over the extreme points. In every iteration of the sub-problem each UAV selects a cost minimizing path for itself and puts it into the master problem to find out if this path is better than the its path in the previous iteration for the whole group.

The sub-problems are solved simultaneously on every UAV thus through this synchronization the master problems are all identical at each UAV . All the UAVs communicate with each other to send the generated better path which is not in the restriced master problem and the cost of this new path. This generated path is basically ones and zeros of the number of waypoints indicating if the selected waypoint is being visited or not. The communication data is thus very small containing the UAV number the waypoints to be visited and the cost of the path that the UAV will follow. After new columns are generated to enter the basis of the master problem the same master problem is solved in every UAV. Since every UAV solves the same problem with exactly same numerical values there is no need to exchange the solutions. This is illustrated in Fig. 2. Since the master problem contains only a few constraints its solution is rather very fast. In the primal restricted master problem form, the master problem consists of inverting a small matrix which has dimensions $(n+m) \times (n+m)$ and its multiplication for a few times. The complete method is given in implementation form as Algorithm 1.

Algorithm 1 Distributed Synchronized Implementation for i_{th} UAV

Input: n waypoint positions and the i_{th} UAVs position.
Output: The fleet shortest path optimal waypoint assignments for i_{th} UAV

PHASE I : Initialization

Compute the distances between the waypoints and form the $n \times n$ matrix V
Compute Compute the distance to the waypoints and form the $n \times 1$ matrix d_i
Compute Compute the distance to the waypoints and form the $n \times 1$ matrix d_i
for every possible path i_{th} UAV can travel **do**
  Add the path to the matrix D_i
  Calculate the cost of this path to vector c_i
end for
Read the $(n+m) \times (n+m)$ initial B matrix from memory
Communicate The costs for corresponding paths in the initial B matrix to start the master iteration

PHASE II : Dual Simplex Ascent

repeat
  Solve the restricted master iteration and find the dual variables vector : $[\pi \quad \mu]' = [\bar{c}'_1 \ldots \bar{c}'_m]' B^{-1}$
  Remove the ineffective flight path (if any) with maximum cost
  Solve the sub-problem corresponding to the i_{th} UAV by updating the flight costs : $c^q_{i\ updated} = c^q_i - \pi d^q_1 - \mu_i$
  Select the minimum negative $c^q_{i\ updated}$ (sub-problem cost) and the associated path :
  Exchange the selected path and its flight cost and the paths and the associated costs generated by other vehicles
  Insert the first feasible flight path with minimum cost solution and its cost to the restricted master problem.
until There are no paths with negative subproblem costs

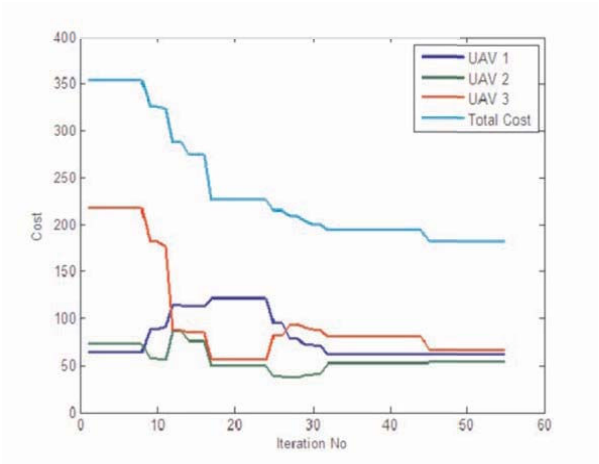

Fig. 3. The cost variation across the iteration steps for a three vehicle ten waypoint scenario.

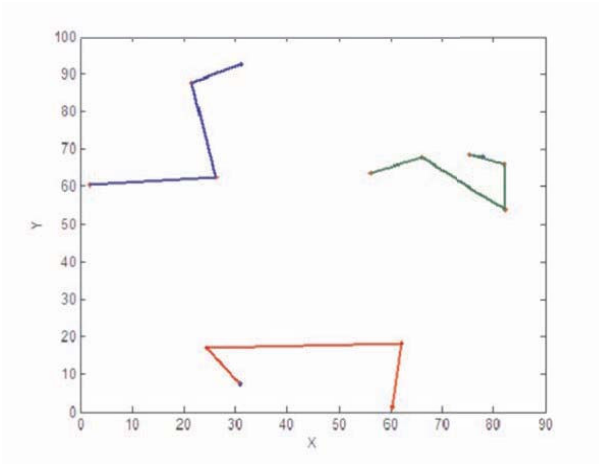

Fig. 4. The waypoint selection for a three vehicle ten waypoint scenario.

It is clear from the above algorithm that the communication phase between the iterations is limited and both the initialization and the optimization problems are well distributed to the UAVs. There are several other advantages of the approach followed here. The first advantage of the approach followed here is that the sub-problems running in each vehicle requires much less memory than a standard centralized mixed-integer program. This advantage allows relatively cheap hardware in the UAVs. Another advantage of the column generation process is that in each iteration cost of the master problem reduces. Although it is sometimes possible to come across with a non-integer solution in the master problem, using a round-up algorithm leads to a suboptimal solution quickly. Thus the algorithm can be quit returning a suboptimal solution which is very advantageous in hard-real-time applications. This solution is guaranteed to approach the optimal solution in each iteration. It is also possible to pose upper and lower bounds for the optimal cost of the problem such that if the algorithm is quit in the middle, the distance of the solution to the optimal solution can be known. Figs. 3 and 4 show a typical cost convergence and the solution that can be found using the algorithm. Fig. 5 provides a detailed look (simulation covering 600 scenarios) at the computational behavior of algorithm. Notice that a total of 250 targets can be solved across 5 UAVs for around 50 seconds.

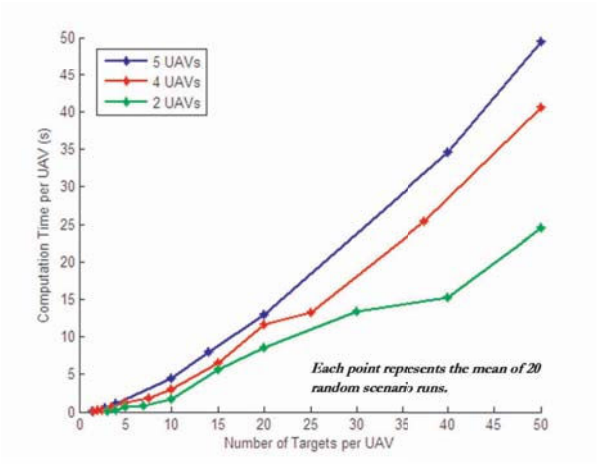

Fig. 5. The computational behavior of the algorithms shows strong correlation with the number of way-points per UAV given a particular snap-shot of scenario

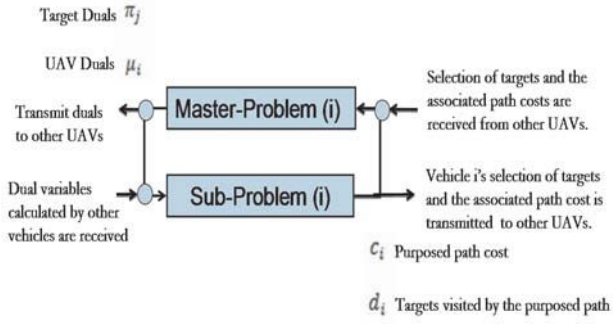

Fig. 6. The asynchronous distributed implementation of the algorithm.

Another feature of the method is that the algorithm can run in a totally asynchronous mode with arbitrary local initial B (initial set of waypoint selection batch) selection. The structure of this implementation is shown in Fig. 6.

3.2 Network Mission Simulator : Dynamic Task Assignment for Real-Time Implementations

Since the task assignment algorithm has to run in real-time, real-time performance and the implementation in real-time becomes of great importance. There are several examples of real-time implementation of task assignment and path planning algorithms (see [21] for example).

For real-time implementation of the algorithm, a sub-optimal algorithm can be considered that takes a number of targets from all set of targets to be assigned to the vehicles in the fleet. This subset of the targets can be chosen as the closest subset in this case. The assignment process can be re-executed taking one more target into the optimization problem when one of the targets is visited by one of the UAVs. Using this scheme the number of targets in the optimization problem is always fixed and the optimization is executed for every new target together with the targets that were used in the optimization problem before but were not visited.

The flow of the algorithm is as follows. First a specific number of targets are taken to the problem for optimization. These targets are determined from the ones that are available with a small communication between the vehicles. This communication is the initialization of the Algorithm 1. Then the Algorithm 1 is

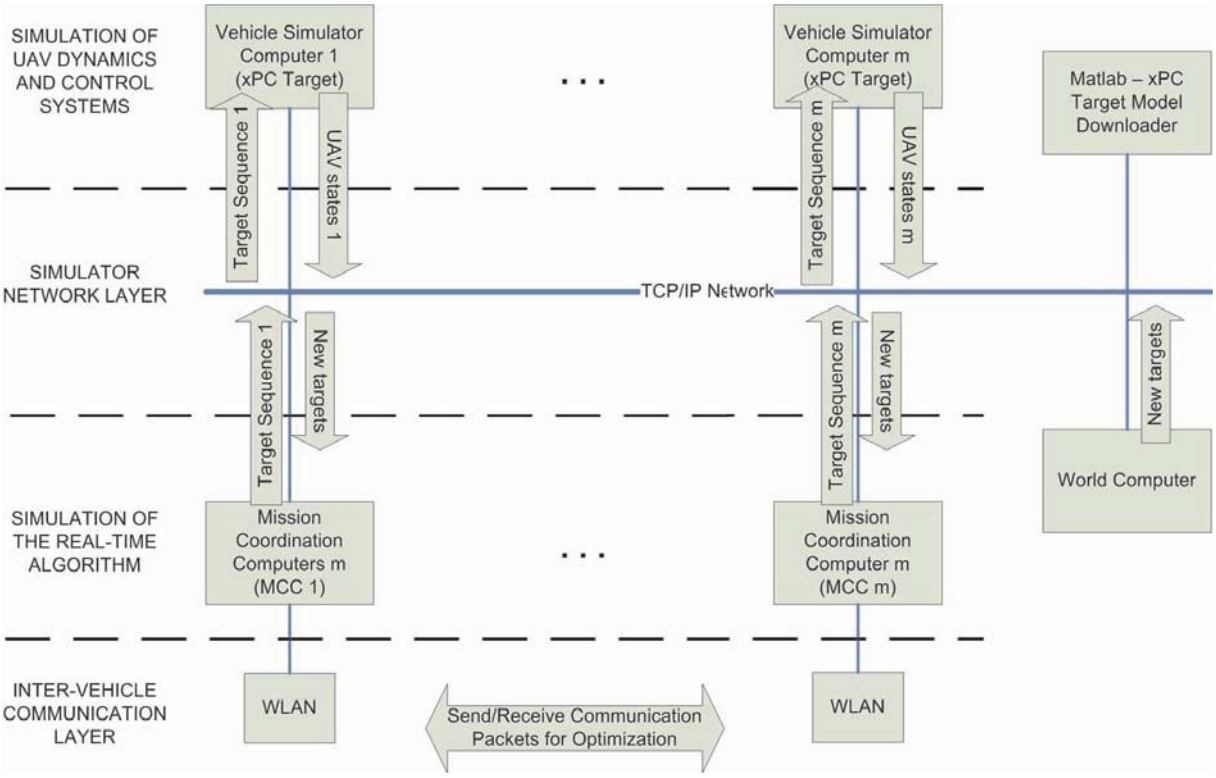

Fig. 7. The in-house developed Network Simulator (NSIM) allows rapid prototyping and HIL testing of various coordination algorithms among manned and unmanned fleets, and the mission control center.

executed for distributed solution. Afterwards the UAVs start to get to the targets that are assigned to them. Whenever one UAV reached a target a new target is taken to the optimization problem to be considered with other unvisited targets. Algorithm 1 is re-executed to find a new solution. The algorithm is executed in this manner as the mission continues.

After the re-execution of the algorithm one of the UAVs can change its route to another target leaving the one it is already traveling to one of the other UAVs. This is possible since the new target taken into the optimization problem changes the assignments and now one target can be more easily visited by another UAV considering the newly entering target in hand. Note that after a target is fixed to a j_{th} vehicle, we update its flight costs with $c_i = c_i + \pi_{assigned}$ to create balance as new targets appear in the scenario.Using such a scheme the solution achieved is a sub-optimal solution but it can be easily implemented in real-time.

The numeric implementation of such scenarios for four vehicles 200 hundred waypoints and then again for 500 waypoints are given in Figs. 8 and 9 respectively. Fig. 10 provides an extensive verification of the restricted horizon polynomial-time computational growth.

The in-house developed network mission simulator allows real-time simulation across interoperable manned-unmanned fleets, and the mission control center[25]. The hardware structure within the network simulator is tailored to mimic the distributed nature of each of the vehicle's processors and communication modules. Open-source flight simulation software, FlightGear, is modified for networked operations and it is used as the 3D visualization element for the pilot and the mission controls. The UAV dynamics and low-level control algorithms are embedded within the xPC target rack. Equipped with 3D flight simulation displays and touch-screen C2 interface at the desktop pilot level, the platform also allows us to rapidly prototype and test pilot-unmanned fleet supervisory control and pursuit-evasion game designs.

In this particular set-up, we use the Mission Coordination Computers (MCCs) to embed the real-time implementation of the task/target assignment algorithm and run simulations if they are in a real mission. Vehicle dynamics and the low level control algorithms for each UAV is embedded within a unique xPC target modules. They simulate an autonomous UAV which receives a sequence of targets from a UDP based communication channel and execute these commands to reach to the targets. MCCs run a C program that is used as a bridge between the communication layer between the UAVs and the Matlab based computation layer of each of the UAVs. Each UAVs MCC send its respective targets to the UAV control systems during the execution of the target assignment algorithm. The multi-vehicle simulator system also includes a world computer which is responsible for simulation of the mission. This computer is the computer that informs the MCCs if new target appears in the scenario. The implementation is illustrated in Fig. 7.

4. CONCLUSION

A distributed task allocation algorithm is designed and implemented for real-time operation in this work. The performance of the distributed algorithm is seen to be well better than the central one. This is attributed to two main reasons, namely the distributed nature of the algorithm and the polynomial time sub-problem structure. The former reason was expected since the algorithm contains very small communication messages to be

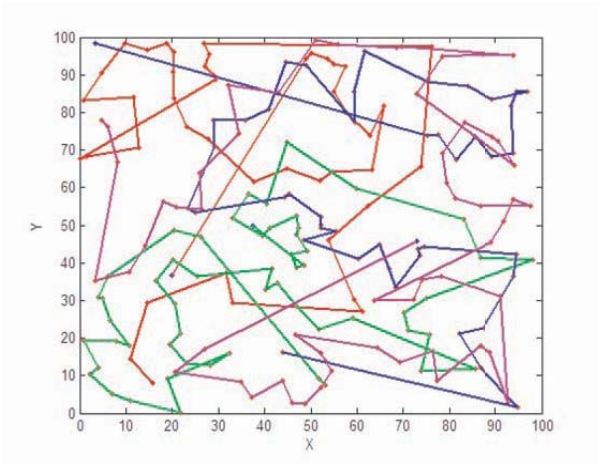

Fig. 8. The waypoint routes for a random pop-up task-target assignment for four vehicles. The algorithm is implemented in the receeding horizon mode for two hundred waypoints.

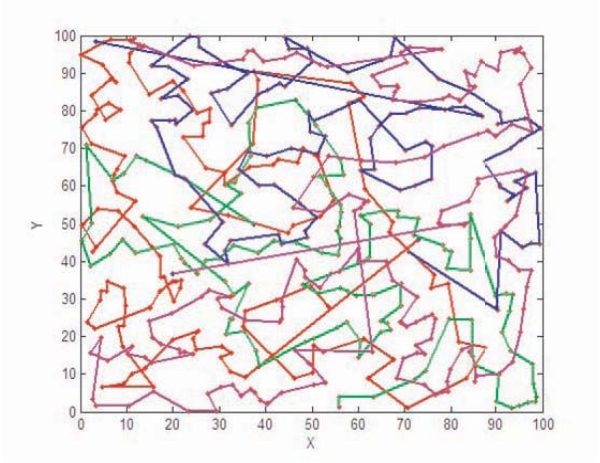

Fig. 9. The waypoint routes for a random pop-up task-target assignment for four vehicles. The algorithm is implemented further in the receeding horizon mode for five hundred waypoints.

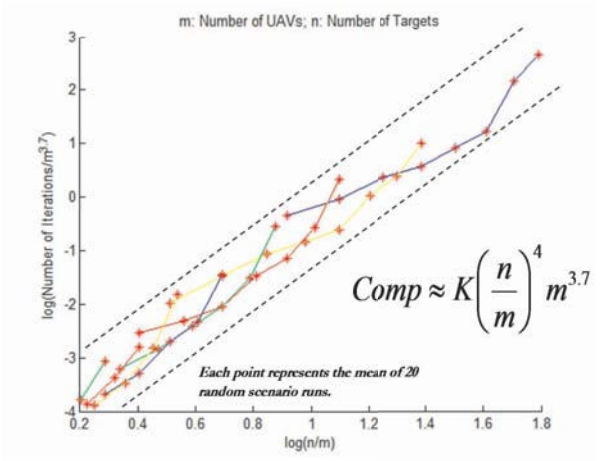

Fig. 10. The numeric computational complexity analysis suggests polynomial-time growth for restricted time horizons.

exchanged between the decision making agents and the distribution uses more than one processing units. The latter reason is a good advantage of the decomposition algorithm used; an NP-hard sub-problem structure is seen to be polynomial-time solvable for restricted horizons after decomposition.

A testbed for real-time distributed task allocation algorithms was also presented. This simulator is a general purpose simulator for testing UAV dependent algorithms. The tests performed on an in-house built network mission simulator provides numerical verification of the algorithm on a) bounded polynomial-time computational complexity and b) hard real-time performance for problem sizes on the order of hundred waypoints per UAV.

ACKNOWLEDGEMENTS

The authors would like to acknowledge Ahmet Cetinkaya, Oktay Arslan and Mirac Aksugur for their work in the development of the network mission simulator.

REFERENCES

[1] D. Bertsimas, J.N. Tsitsiklis, Introduction to Linear Optimization, Athena Scientific, 1997.
[2] L. Brickman, Mathematical Introduction to Linear Programming and Game Theory, Springer-Verlag, 1989.
[3] R.S. Garfinkel, G.L. Nemhauser, Integer Programming, John Wiley, 1972.
[4] T. Schouwenaars, E. Feron, B. de Moor, and J. P. How, "Mixed Integer Programming for Multi-vehicle Path Planning," in Proceedings of the European Control Conference, European Union Control Association, Porto, Portugal, September, 2001, pp. 2603-2608.
[5] J. Bellingham, M. Tillerson, A. Richards, J. P. How, "Multi-Task Allocation and Trajectory Design for Cooperating UAVs," in Cooperative Control: Models, Applications and Algorithms at the Conference on Coordination, Control and Optimization, November 2001, pp. 1-19.
[6] L. Lasdon, Optimization Theory for Large-Scale Systems, Macmillan, 1970.
[7] J. Desrosiers, M.E. Lbbecke, "A Primer in Column Generation", in Eds. G. Desaulniers, J. Desrosiers, M.M. Solomon, Column Generation, 2005, pp. 1-32.
[8] G. Desaulniers, J. Desrosiers, M.M. Solomon, Column Generation, Springer-Verlag, 2005.
[9] F. Vanderback, "Implementing Mixed Integer Column Generation", in Eds. G. Desaulniers, J. Desrosiers, M.M. Solomon, Column Generation, 2005, pp. 331-358.
[10] R.K. Ahuja, T.L. Magnati, J.B. Orlin, Network Flows: Theory, Algorithms, and Application, Prentice Hall, 1993.
[11] S. Irnich, G. Desaulniers, "Shortest Path Problems with Time Windows", ", in Eds. G. Desaulniers, J. Desrosiers, M.M. Solomon, Column Generation, 2005, pp. 33-66.
[12] B. Kallehauge, J. Larsden, O.B.G. Madsen, M.M. Solomon, "Vehicle Routing Problem with Time Windows", ", in Eds. G. Desaulniers, J. Desrosiers, M.M. Solomon, Column Generation, 2005, pp. 67-98.
[13] E. Donna, C.L. Pape, "Branch-and-Price Heuristics: A Case Study on the Vehicle Routing Problem with Time Windows", ", in Eds. G. Desaulniers, J. Desrosiers, M.M. Solomon, Column Generation, 2005, pp. 99-130.
[14] A. Chabrier, "Vehicle Routing Problem with Elementary Shortest Path Based Column Generation", Computers and Operations Research, vol. 33, 2006, pp. 2972-2990.

[15] M. Ehrgott, J. Tind, "Column Generation in Integer Programming with Applications to Multicriteria Optimizaiton" Unpublised working paper, 2006, draft available online *http* : *//www.math.ku.dk/ tind/workingpapers.html*.

[16] M. E. Lubbecke, J. Desrosiers, "Selected Topics in Column Generation", to appear in Operations Research, available online at Optimization Online *http* : *//www.optimization − online.org/DB_HTML/*2002/12/580.*html*.

[17] C. Barnhart, E.L. Johnson, G.L. Nemhauser, M.W.P. Savelsbergh, P.H. Vance, "Branch-and-Price: Column Generation for Solving Huge Integer Programs", Operations Research vol. 46, no. 3, 1998, pp. 316-329.

[18] F. Vanderback, M.W.P. Savelsbergh, "A Generic View of Dantzig-Wolfe Decomposition in Mixed Integer Programming", Operations Research Letters, vol. 34, 2006, pp. 296-306.

[19] T.K. Ralphs, M.V. Galati, "Decomposition and Dynamic Cut Generation in Integer Linear Programming", Mathematical Programming A, vol. 106, 2006, pp. 261-285.

[20] P. Toth, D. Vigo, The Vehicle Routing Problem, SIAM Publications, 2002.

[21] A. Richards, Y. Kutawa, J. How, "Experimental Demonstrations of real-time MILP control", AIAA Control, Navigation and Guidance Conference and Exhibit, 2003.

[22] K. Melhorn, M. Ziegelmann, "Resource Constrained Shortest Paths", Proceedings of the 8th Annual European Symposium on Algorithms, no. 1879 in Lecture Notes in Computer Science, Springer, 2000, pp. 326-337.

[23] M. Minoux, "A Class of Combinatorial Problems with Polynomially Solvable Large Scale Set Covering/Partitioning Relaxations", RAIRO Rech. Opr., vol. 21, 1987, pp. 105-136.

[24] E.L. Johnson, A. Mehrotra, G.L. Nemhauser, "Min-cut Clustering", Mathematical Programming, vol. 62, 1993, pp. 133-151.

[25] A. Cetinkaya and S. Karaman, O. Arslan, M. Aksugur, G. Inalhan, " Design of a Distributed C2 Architecture for Interoperable Manned-Unmanned Fleets," Conference on Cooperative Control and Optimization, Gainsville, FL, 2007