AIAA Guidance, Navigation, and Control Conference and Exhibit
11-14 August 2003, Austin, Texas

AIAA 2003-5586

Optimal vs. Heuristic Assignment of Cooperative Autonomous Unmanned Air Vehicles

Steven Rasmussen[1]
Veridian Inc., Wright-Patterson AFB, Ohio

Phillip Chandler[2], Jason W. Mitchell[3],Corey Schumacher[4], Andrew Sparks[5]
Flight Control Division, Air Force Research Laboratory
Wright-Patterson AFB, Ohio

ABSTRACT

This paper describes investigation of algorithms that generate vehicle and task assignments for autonomous UAVs in cooperative missions. In previous work many algorithms have been examined to solve the problem of assignment of autonomous UAVs to tasks based on their ability to perform those tasks. Because of the complexity of this problem, these algorithms have been based on heuristic solutions to the problem. This work presents details of an algorithm that produces the optimal assignment of UAVs for a given scenario. A set of sample scenarios is used to explain the cooperative control assignment problem and how it changes as complexity increases. Three requirements for an optimal to solution to the assignment of UAVs, in a given scenario, are task coordination, task precedence, and flyable trajectories. The requirements and the motivation behind them are explained. An assignment tree generation algorithm is described. This tree generation algorithm produces the optimal assignment of UAVs given a particular scenario. This algorithm is used to produce optimal solutions to the assignment problem which are use to measure the effectiveness of heuristic based cooperative assignment algorithms.

INTRODUCTION

Advances in technology have made it possible to field autonomous unmanned air vehicles (UAVs) that can be deployed in teams to accomplish important missions such as suppression of enemy air defenses and persistent area dominance. While it is technically possible to field these types of systems, work is needed to develop strategies/algorithms to allow these UAVs to optimize the use of their combined resources to help them accomplish their mission. One promising strategy is to allow the UAVs to cooperate with each other in planning and execution of the mission.

Major portions of proposed missions can be preplanned, but due to limited information about the enemy positions and assets in the battlefield area, the UAVs will have to react to changes in the perceived enemy state during the execution of the mission plan. If the UAVs are able to cooperate with each other, they will be able to optimize the use of their combined resources to accomplish the goals of their mission. While cooperation of this kind is desirable, it can be very complicated to implement.

To facilitate the study of cooperation among UAVs, three representative scenarios, see Figure 1, were developed that incorporate the major challenges of implementing cooperative control algorithms. The scenarios are useful for understanding the basic problem and how it changes as complexity is added. The scenarios highlight the main requirements of the cooperation problem. These requirements are assignment coordination, task precedence, and flyable trajectories. The scenario and cooperation requirements will be described in the next section.

Many different candidate cooperative control algorithms have been developed, implemented, and simulated.[1,2,4,5,6,7,9] But, due to the complexity of this problem, all of these algorithms have been based on heuristic solutions to the problem. In order to judge the effectiveness of these algorithms it is desirable to compare them against an algorithm that produces optimal solutions to the given assignment problems. In section 3 an algorithm is described that produces optimal solutions to the assignment problem based on piecewise optimal trajectories. This algorithm generates a tree of feasible assignments and then chooses the optimal assignment. During generation of the tree all of the requirements are met, but since generation of the tree requires an enumeration of all of the feasible assignments, direct use of this approach is only reasonable for relatively low dimensional scenarios.

The available heuristic cooperative control algorithms were compared to the optimal solution to

[1] AIAA Member, Senior Aerospace Engineer
[2] AIAA Member, Senior Aerospace Engineer
[3] AIAA Member, Visiting Sceintist
[4] AIAA Member, Aerospace Scientist
[5] AIAA Associate Fellow, Aerospace Scientist

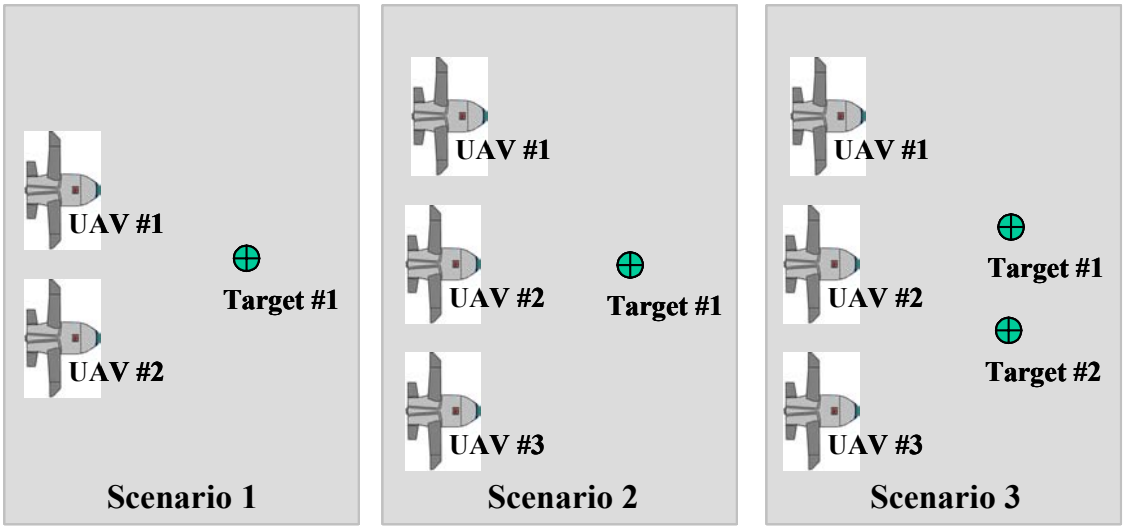

Figure 1 Representative scenarios.

judge their performance. This was accomplished by selecting a statistically significant set of scenarios. Variations between the scenarios consisted of number, position, and heading of vehicles and targets. The heuristic and optimal algorithms were evaluated with each scenario in the set and the results compared. These results are described in section 4.

THE ASSIGNMENT PROBLEM

In order to understand the assignment problem, the three scenarios, shown in Figure 1, are used. In the scenarios the UAVs are required to perform three tasks to prosecute each target, i.e. *classify*, *attack*, and *verify* the kill of the targets. Each of these tasks has requirements governing its execution. Target classification is required to ensure that the subject object is the intended target and not a decoy or some other non-target. To complete a classification task a vehicle must follow a trajectory that places the vehicle's sensor footprint on the target from one of a given set of headings. After a target has been successfully classified UAVs can attack it from any heading with no requirement for sensor footprint offset. UAVs are killed during the attack. After the target has been attacked, the UAVs must verify that the target was killed. In order to verify kill, a UAV can approach the target from any heading; however, the UAV's trajectory must allow the vehicle's sensor footprint to cross over the target. Thus there is a requirement for sensor footprint offset.

The tasks for each target must be accomplished in order, i.e. the target must be classified before it can be attacked and attacked before it can be verified. This means that any algorithm that produces cooperative task assignments must enforce *precedence* of the tasks. In order to utilize the UAVs in an efficient manner, each task must accomplished once, i.e. UAVs are not allowed to attack a target twice, unless the target is verified alive after an attack or there is a predefined need for multiple attacks. This means that *task coordination* must be enforced in any optimal solution to this assignment problem. In order to make efficient use of the UAVs the assigned trajectories must be *flyable*, i.e. an average fixed-wing UAV cannot fly a trajectory with a 90-degree turn. If the trajectories assigned to UAV are not flyable, the timing and geographical coordination of the cooperative mission may be invalidated. The scale of the scenario is important in determining the impact of *flyable trajectories*, i.e. if the turn radius of the vehicle is very small when compared to the distance between the targets, then a flyable trajectory may not have a negative impact on the mission.

Based on the analysis above, any optimal cooperative control algorithm for prosecuting targets must enforce the following requirements:

1. *Task Coordination*: Vehicle task assignments must be coordinated to ensure that every task is completed and each task is only assigned to one UAV.
2. *Task Precedence*: Tasks performed on targets must be in the following order: classify, attack and verify, i.e. a target cannot be attacked until it has been classified.
3. *Flyable Trajectories*: The UAVs must be assigned trajectories that they can follow.

OPTIMAL SOLUTION

To understand the need for an optimal solution to the cooperative assignment problem, a tree generation

program, described later, was used to generate the probability distribution of assignment costs for a 3-UAV, 2-target scenario, see Figure 2. A different method for constructing costs was noted by Fowler[3]. The cost used in this study is the total distance traveled by all of the vehicles while prosecuting the targets. In the figure, the height of the bars represents the percentage of feasible assignments contained in the range of the bar. The range of the bars has been normalized to the optimal assignment. Therefore, in Figure 2, the optimal cost is one and other trajectory costs are indicated as a factor relative to the optimal. A dashed line represents the mean cost and dashed-dot lines represent one standard deviation away from the mean. For the case shown in Figure 2, the optimal cost is 66,622 feet, the maximum cost is 166,695 feet, and the mean cost is 105,672 feet. The cost of the worst assignment possible is about 2.5 times more than the cost of the optimal assignment. The mean cost is about 1.5 times more the optimal and the optimal cost is outside of the standard deviation range. This indicates that using an algorithm that finds an assignment that is close to the optimal can lead to a significant performance increase over an algorithm that is based on a random choice from this distribution.

In order to arrive at an optimal solution to assigning UAVs in a cooperative mission, a tree approach was taken. Since the total number of assignment permutations is given by

$$N_P = m^o o! \qquad (1)$$

where m is the number of vehicles, and o is the number of tasks, enumeration of all of the possible assignments is only possible for very small assignment problems. As shown in the plot in Figure 3, the number of perturbations for 4 UAVs and 9 tasks, (3 targets), is $O(10^{10})$. Building a tree from all of the assignment perturbations possible enforces the task coordination requirement, but does not enforce precedence or flyable trajectories. In order to reduce the number of assignment choices that must be calculated, the precedence requirement is enforced.

One method for generating UAV to task assignments is to build a tree of all the possible permutations of UAV to target assignments, as illustrated for scenario 1 in Figure 4. The tree structure enforces *Task Coordination* between the UAVs, but it does not enforce the *Flyable Trajectories* or *Task Precedence* requirements. When the *Task Precedence* requirement is applied to the tree the number of possible assignments is reduced from 48 to 8 as indicated in Figure 4 by the faded entries. Applying the *Flyable Trajectories* requirement to the tree can also reduce the possible assignments if the UAVs cannot find trajectories that maintain the task precedence. It is important to note that although scenario 1 contains only two UAVs and one target, the number of assignment permutations is relatively large, i.e. 48. The addition of one extra UAV in scenario 2 raises the total number of permutations to 162 and, as shown in Figure 5, the number of feasible assignments to 18. Scenario 3 adds an extra target, which increases the possible

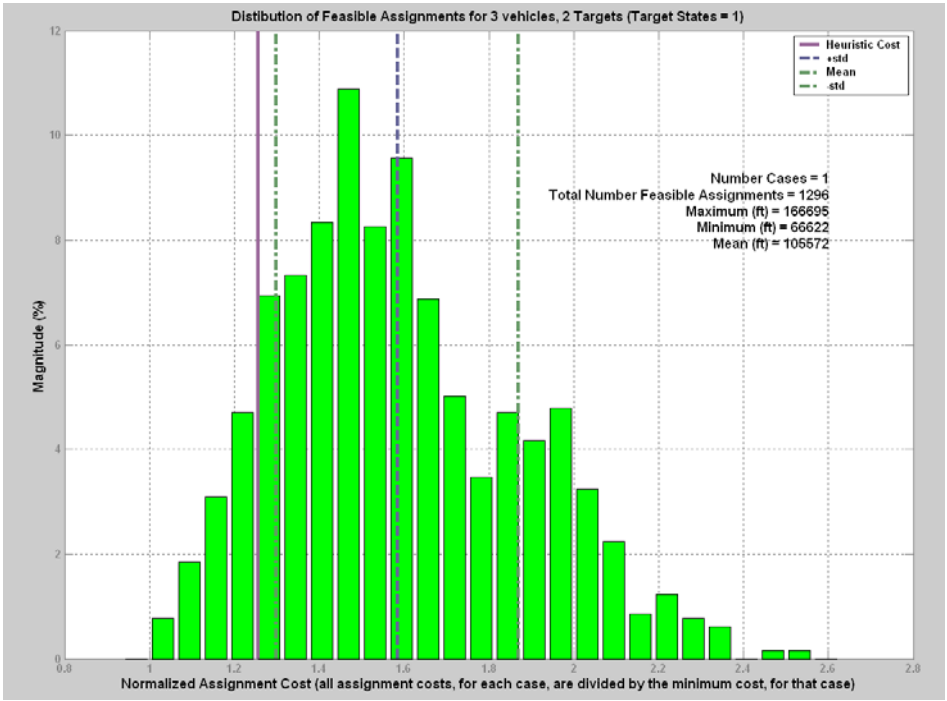

Figure 2 Probability distribution of assignment costs.

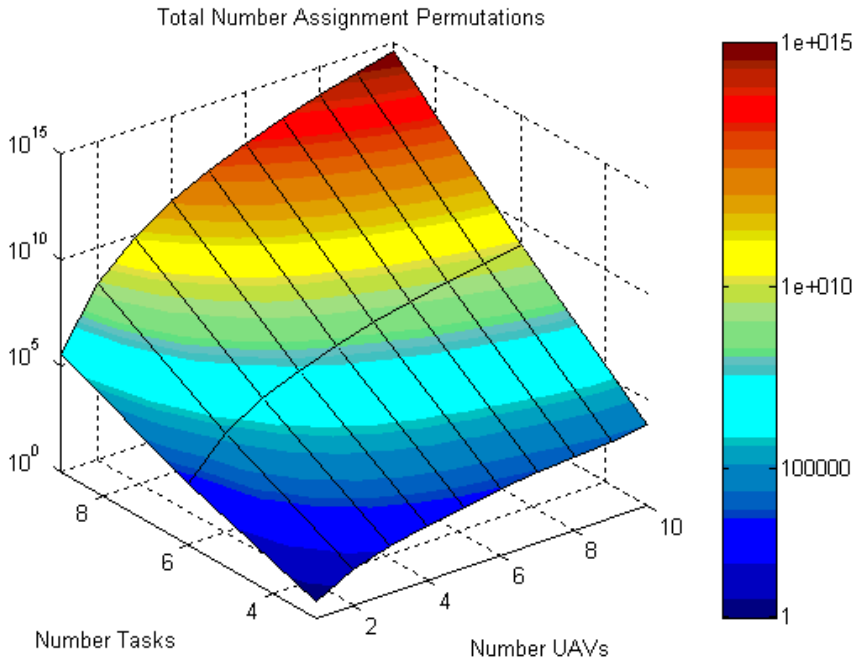

Figure 3 Permutations of assignments.

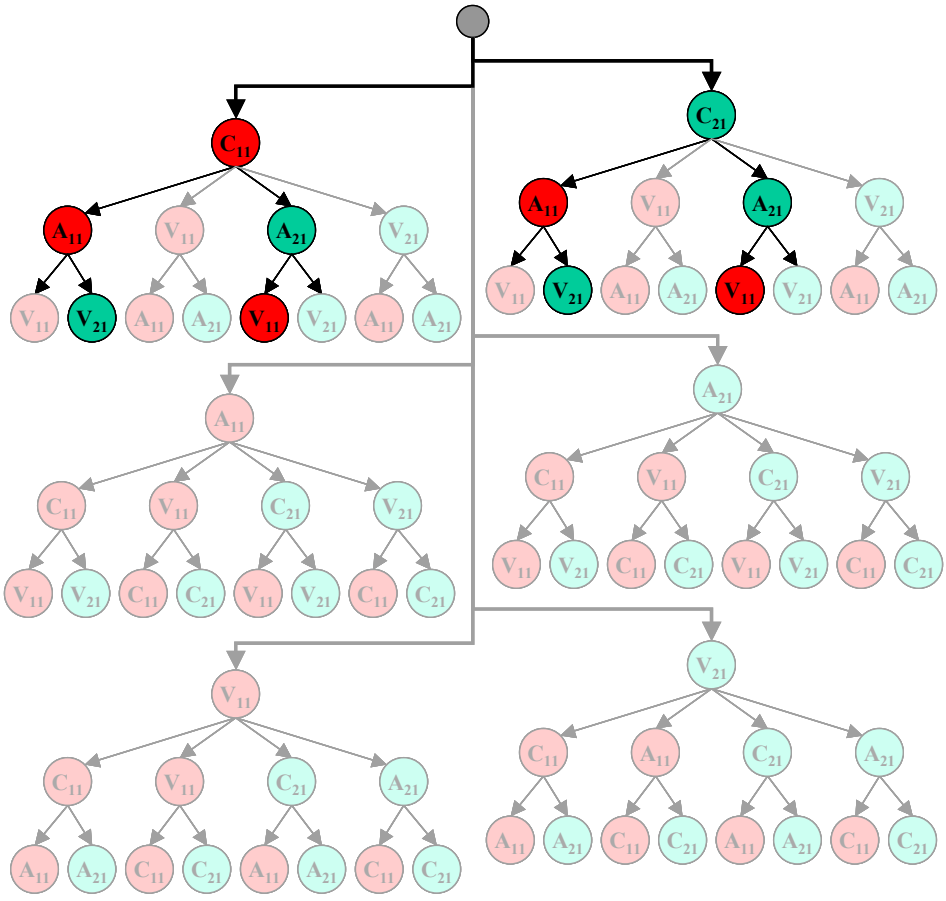

Figure 4 All possible assignment permutations for scenario #1.

assignments to 524880 and the feasible assignments to 13178. Figure 6 shows a partial plot of the feasible tree for scenario 3.

The nodes in the tree represent tasks that need to be accomplished and the arcs link nodes that can be aggregated into an assignment. Each node and arc has an associated vehicle ID, vehicle's previous task ID, a task type, and a target ID. The nodes also have a time of task completion and the arcs have an associated cost. In Figure 5, the cost of each node is labeled b_{ijkl}, where *i* is the vehicle ID, *j* is the vehicle state ID, *k* is the task ID, and *l* is the target ID. Likewise, the time of task accomplishment at each node is labeled, t_{ijkl}. The tree is constructed using the following procedure:

1. Nodes, and their associated costs and times, are added to represent each vehicle accomplishing the next task required by each target.
2. For each of the nodes,
   - The task that was accomplished at the previous node is removed from the target's task list.
   - The target's task prerequisite time is updated.
   - A subset of nodes are added that represent each living vehicle accomplishing the next task required by each target. The nodes are only added if the time of task accomplishment is after the target's task prerequisite time.
3. Step 2 is repeated until all tasks have been assigned or until there are no vehicles alive.

Once the tree is constructed, the number of nodes at the end of the branches of the tree represents the number of feasible assignments. Beginning at one the end nodes and traversing the tree back to a starting node can obtain the actual vehicle/task assignments. The total cost of each assignment is obtained by summing the cost of the arcs. The costs can be calculated by assuming straight-line segments between the targets and the vehicles or based on flyable trajectories. In either case, if the vehicle is calculated to arrives at a target before the prerequisite task time, a path lengthen algorithm can be used to adjust the vehicles arrival time. With path lengthening it is possible to lower the total cost of assignments and reduce the total time needed to prosecute the targets.

This algorithm was implemented in a program written in C++. The program is capable of using either straight-line segments as trajectories or it can calculate optimal trajectories between points. When using optimal trajectories, the program can return the waypoints necessary for the UAVs to implement the optimal assignment. The program is capable of path lengthening. When straight-line trajectories are used, the path-lengthening algorithm simply returns the distance required by the target's task prerequisite time.

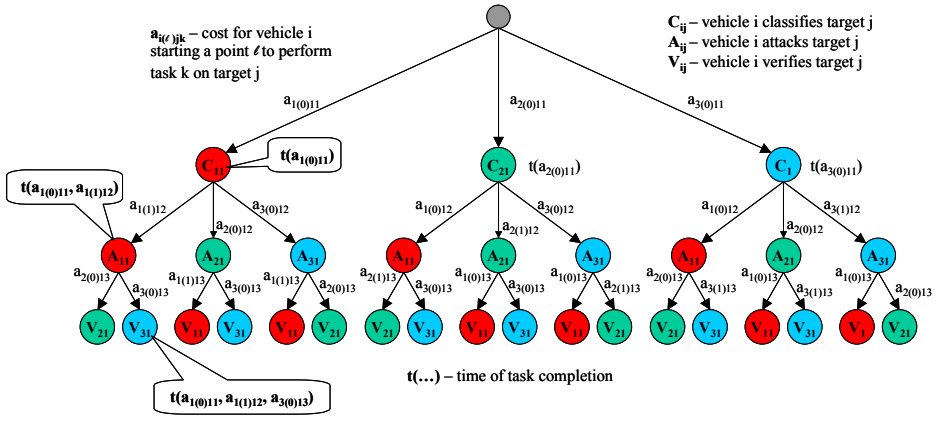

Figure 5 Feasible assignments only for scenario #2.

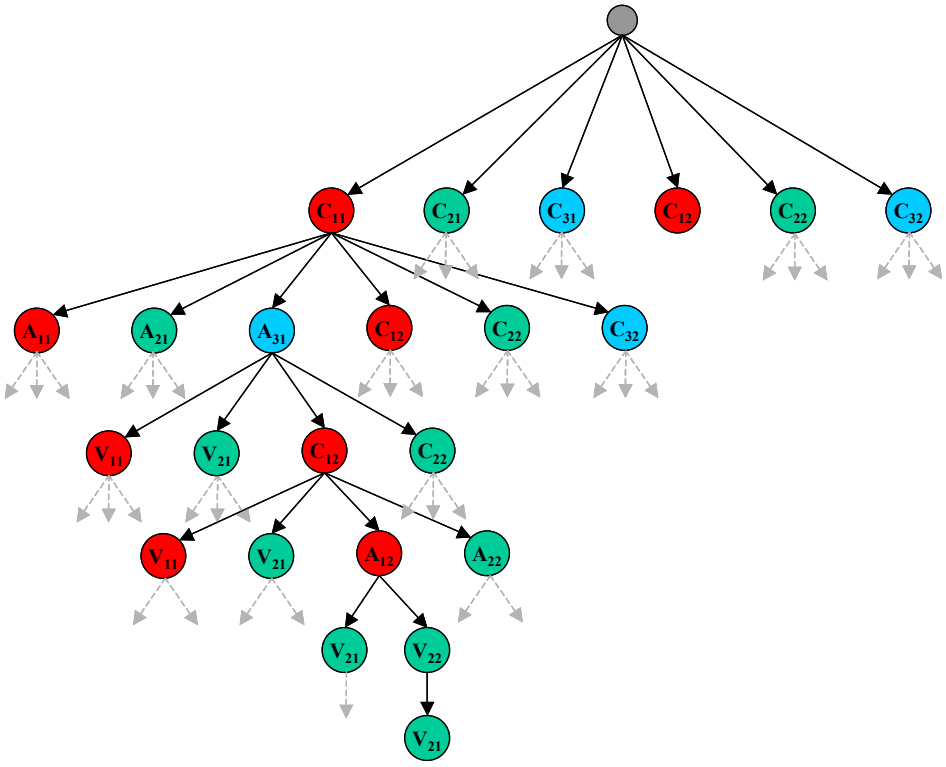

Figure 6 Sample of feasible assignments for scenario #3.

COMPARISON OF HEURISTIC ALGORITHMS TO OPTIMAL

In order to compare the heuristic algorithms to the optimal algorithm an experiment was constructed using the MultiUAV simulation[8]. The simulation was set up to have 3 vehicles and 2 targets, i.e. scenario 3. For each simulation run the vehicle started at the same location and searched a given area with a given search pattern. At the beginning of each simulation run the position and heading of each target was selected using random draws from a uniform distribution. The simulation was run 100 times for each heuristic assignment algorithm. The heuristic assignment algorithms used in the study were the Iterative Capacitated Transhipment[9] and Relative Benefits[7] algorithms. During the simulation runs the state of the vehicles and targets were saved each time the assignment algorithms were called. Based on this saved data, an optimal assignment was generated for each heuristic assignment. Figure 7 shows a sample of the trajectories generated by the Iterative Capacitated Transhipment algorithm, on the left, and the optimal assignment, on the right. This is one of the worst cases, but it clearly shows how the choice of single assignments in an iterative algorithm can lead to suboptimal results.

In order to compare the algorithms the costs were compared. The cost was defined as the total distance travel by all of the vehicles while prosecuting the targets. For this 3-vehicle, 2-target scenario, the two heuristic algorithms produced the same assignments. These assignments were compared to the optimal assignment by subtracting the optimal from them and then dividing by the optimal cost, i.e.

(heuristic cost – optimal cost)/optimal cost

A plot of the distribution of the normalized difference in costs is shown in Figure 8. From the distribution it can been seen that about 40% of the heuristic assignments were within 10% of the optimal and about 24% of the heuristic assignments were more than 50% longer that the optimal. The mean heuristic assignment cost was about 23% more than the optimal. Figure 9 shows a cumulative average of the normalized difference versus the number of case considered. This figure shows that after about 30 cases the cumulative average settles to about 23%.

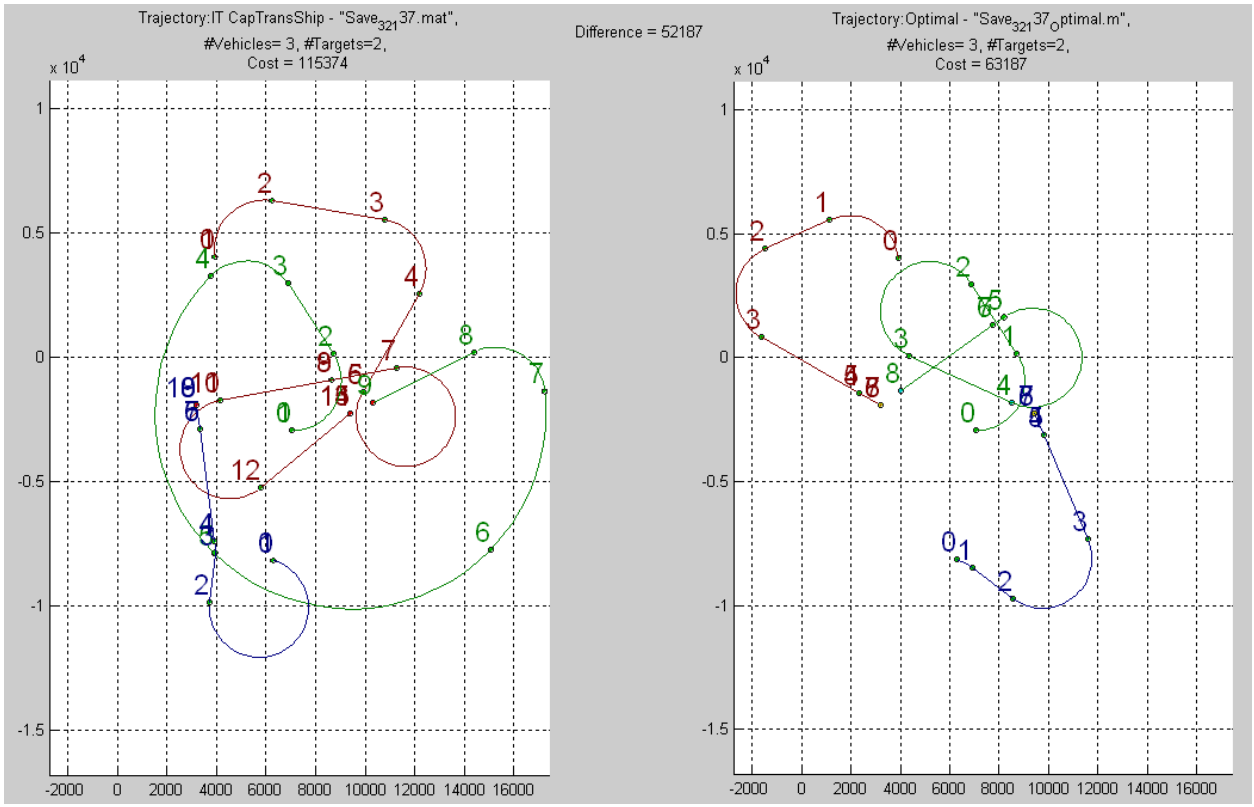

Figure 7 Sample of trajectories generated by the heuristic algorithm and the optimal algorithm.

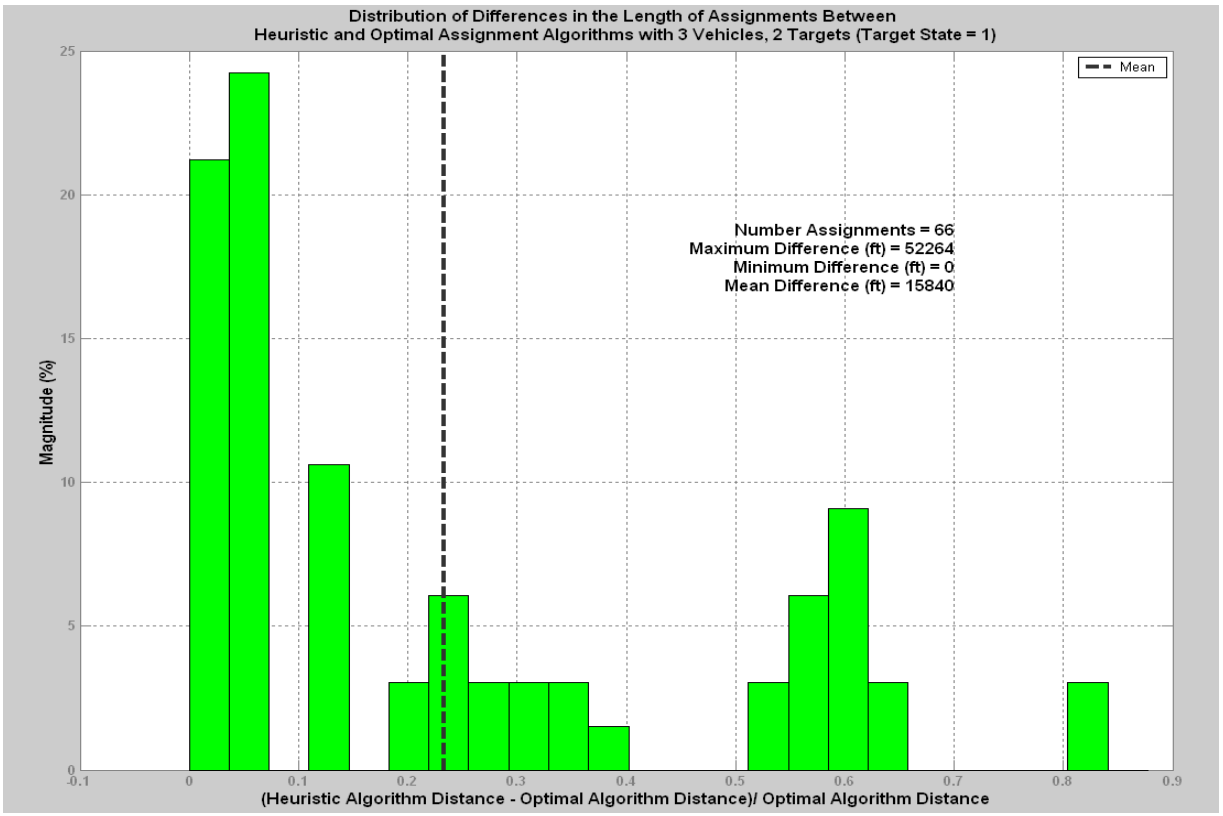

Figure 8 Comparison of Heuristic Algorithm to Optimal Algorithm.

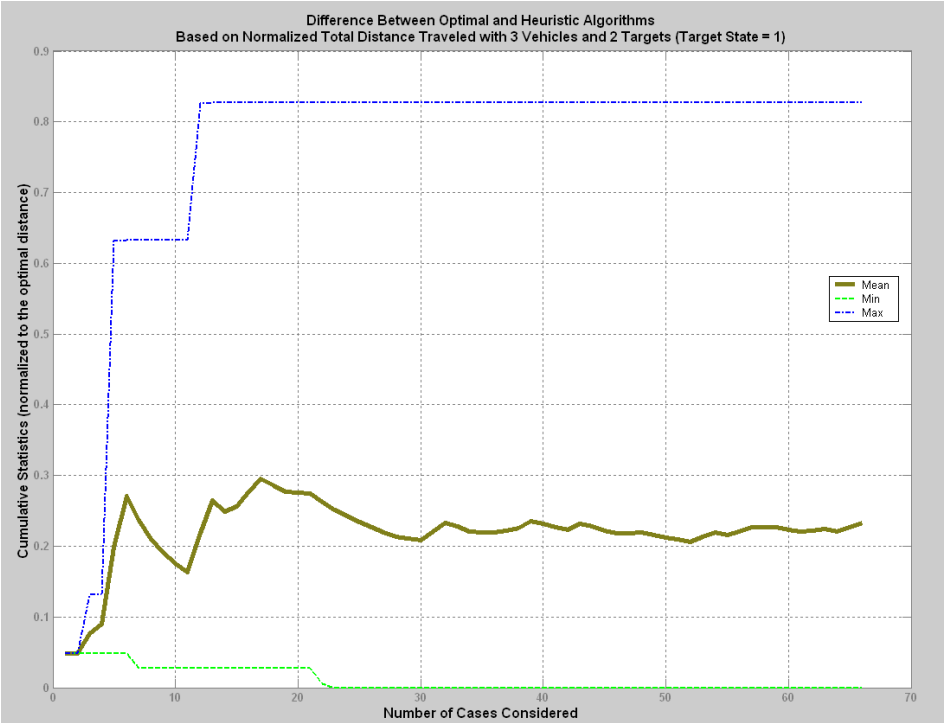

Figure 9 Cumulative Comparison of Heuristic Algorithm to Optimal Algorithm.

CONCLUSION

This work is intended to fill a gap in the development of cooperative control algorithms. At this point in time, many heuristic algorithms have been developed for cooperative control, but the only method to rate their performance is to compare them against themselves. As shown the in the plot of the assignment probability density, Figure 2, there is a significant difference in the costs of assignments near the optimal and those near the mean of the distribution. This indicates that it is important to understand where the assignment solutions of heuristic algorithms fall in the distribution.

From the comparison of heuristic algorithms to the optimal algorithm, we have shown that for the given scenarios the heuristic algorithms performed well. For example, the solid vertical line in Figure 2 represents the cost produced by the heuristic algorithms for the given scenario. While the cost of heuristic solution was about 25% more than the optimal, it was closer to the optimal than the nearest standard deviation from the mean. Of course to know how good or bad the heuristic algorithm is one must have more information about the mission, i.e. moving targets may require quick prosecution, vehicles may not have enough fuel to cover long distances, etc. These comparisons make it possible to make trades such as processing time of the algorithm to the needed optimality.

As we have shown, the tree algorithm can be used to compute optimal solutions, but it can only do this for relatively low numbers of vehicles and targets. Even for these low numbers the processing time involved in computing the optimal assignments precludes using the tree algorithm on a real-time system. Logically there are two directions that can be pursued with this algorithm: (1) use the algorithm to compute optimal assignments to act as baseline cases (2) modify the algorithm to integrate it with a real-time system. The tree generation algorithm is very amenable to parallelization. This makes it possible to implement it on fast multiprocessor computers. This will make it possible to compute optimal assignment for sets of benchmark cases. These cases can then be used to compare candidate assignment algorithms. Changes to the tree algorithm can be made to find assignment solutions for larger dimensional problems in a short enough time to make it possible to implement these algorithms on real systems. One method to do this is to use a dynamic programming solution. Since there are many fast ways to obtain a candidate assignment solution, the dynamic programming approach could be seeded with an assignment solution. The algorithm can then be used to produce assignments that monotonically improve as processing time proceeds. In this manner either an optimal solution can be found or if there is not enough time for the optimal solution, then the seed assignment solution can be improved.

REFERENCES

1. Chandler, P., M. Pachter, K. Nygard, and D. Swaroop, ``Cooperative Control for Target Classification", **Cooperative Control and Optimization**, Kluwer, 2001.
2. Chandler, P. R., et al," Complexity in UAV Cooperative Control", American Control Conference 2002.
3. Fowler, Jeffery M., "Coupled Task Planning for Multiple Unmanned Air Vehicles", tech. rep., AFRL/VACA WPAFB, Dayton, Ohio, 2001.
4. Guo, W. and K. Nygard, ``Combinatorial Trading Mechanism for Task Allocation", 13th International Conference on Computer Applications in Industry and Engineering, June, 2001.
5. Murphy, R.A., "An Approximate Algorithm For a Weapon Target Assignment Stochastic Program," Approximation and Complexity in Numerical Optimization: Continuous and Discrete Problems, editor: P.M. Pardalso
6. Nygard, Kendall E., et al, "Dynamic Network Flow Optimization Models for Air Vehicle Resource Allocation", American Control Conference 2001.
7. Rasmussen, S., Schumacher, C, Chandler, P.R, "Investigation Of Single Vs. Multiple Task Tour Assignments For UAV Cooperative Control", Proceedings of the 2002 AIAA Guidance, Navigation, and Control Conference.
8. Rasmussen, Steven J., et al, "A Multiple Uav Simulation For Researchers", Proceedings of the 2003 AIAA Modeling and Simulation Technologies Conference.
9. Schumacher, C, Chandler, P.R, Rasmussen, S., "Task Allocation for Wide Area Search Munitions Via Network Flow Optimization", Proceedings of the 2001 AIAA Guidance, Navigation, and Control Conference.