

A HYBRID GENETIC ALGORITHM FOR THE TARGET VISITATION PROBLEM

ASHWIN ARULSELVAN, CLAYTON W. COMMANDER, AND PANOS M. PARDALOS

ABSTRACT. In this paper we consider the problem of determining an optimal path for an unmanned aerial vehicle which needs to visit multiple targets. The objective is to minimize the travel distance while maximizing the utility of the visitation order. This is known as the TARGET VISITATION PROBLEM and has several applications including combat search and rescue, environmental assessment, and disaster relief. First, we provide a mathematical model based on integer linear programming and prove that the problem is \mathcal{NP} -complete. Then we describe the implementation of a genetic algorithm for finding approximate solutions. The heuristic is then hybridized by the implementation of a local search procedure. Numerical results are presented demonstrating the effectiveness of the proposed procedure.

1. INTRODUCTION

Path planning problems are among the most studied topics in operations research [7, 8, 27, 36]. In fact, the TRAVELING SALESMAN PROBLEM (TSP), arguably the most famous of all optimization problems falls in to this class [33]. In this paper, we consider the problem of determining the optimal route for an unmanned aerial vehicle (UAV) which needs to visit multiple targets and return to its point of origin. The objective is to minimize the total distance traveled and maximize the utility of the visitation sequence. This is known as the TARGET VISITATION PROBLEM (TVP) and has several applications including combat search and rescue, disaster relief, and environmental assessment [26].

Work on the TVP is limited. It was first posed in a paper by Grundel and Jeffcoat dating from 2004 [26]. In this work, the authors describe the problem and the implementation of a greedy randomized adaptive search procedure (GRASP) for computing approximate solutions. The aforementioned paper was intended to provide an introduction to the problem, not an extensive computational analysis. Instead, the authors provide a combinatorial formulation and examine the similarities between the TVP and two other well-known problems, namely the TRAVELING SALESMAN PROBLEM [33] and the LINEAR ORDERING PROBLEM [17]. We provide a similar analysis in a later subsection.

1.1. Motivation. As technologies advance, the use of unmanned aerial vehicles (UAVs) for civilian and military applications is increasing. Civilian applications include environmental assessment and search and rescue. Moreover, UAVs have been used in military applications for decades and help to ensure that coalition forces maintain a competitive advantage in the global war on terrorism. Often times, path planning for the particular mission, be it civilian or military, is a non-trivial process. Two important, often competing factors are the overall distance traveled by the UAV and the sequence in which various “targets” or “points of interest” are visited [26]. Before moving on to the rigorous mathematical formulation, we provide a simple example demonstrating the idea behind the TVP and motivating its use in several practical military and civilian applications. In [26], the authors provide a similar example in the context of a UAV surveilling an environmental mishap, which we follow with a modification of the theme.

Suppose that the set of points in Figure 1(a) represent a collection of villages in which a sought after terrorist is suspected of hiding out. The point labeled “Origin” is the location of the coalition force. Moreover assume the available intelligence data suggests that the

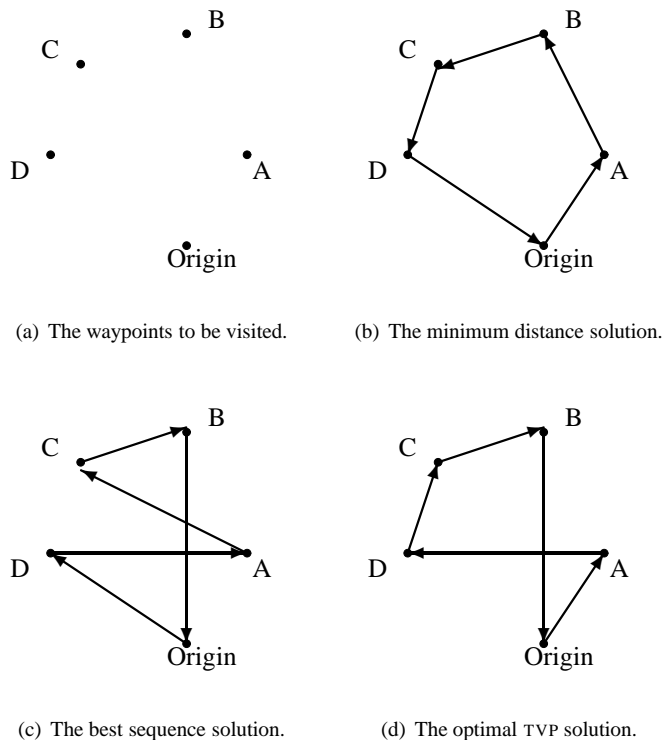


FIGURE 1. This example compares the optimal solution for the TVP instance with the related TSP and LOP solutions.

most likely hiding spot of the terrorist is point D, and the second most likely location is point A. In an application such as the one described, it is well-known that the person of interest moves frequently, and that the older the intelligence date is, the less accurate it becomes. Suppose the coalition force has the ability to launch a miniature UAV from the Origin and have it pass through a pre-established set of waypoints before returning to the starting point. During its flight, the UAV is capable of telemetering data back to the coalition force helping to establish the known location of the terrorist they seek.

The remaining subfigures demonstrate the optimal solutions when various objectives are considered. In Figure 1(b), only the overall distance traveled by the UAV is taken into account. Notice here that point D, the most probable location of the terrorist is visited last. Clearly this is not a desirable visitation sequence. In Figure 1(c), maximizing the preferences in which the villages are visited is the only objective considered. This sequence, while better than the previous is still long and could perhaps be shortened a bit without sacrificing too much time between visiting the highly probable waypoints. The route given in Figure 1(d) does precisely this. Here, a combination of distance and preference is considered in the path planning. We see that this route provides the optimal mixture of visiting “high chance” waypoints quickly, so that the coalition force may act on the intelligence they receive. This simple example demonstrates the importance of considering both distance and visitation sequence when solving a path planning problem for a UAV. As we will see in the next section, when considered individually, these two objectives generalize two well-studied problems in combinatorial optimization. As it turns out, both are \mathcal{NP} -hard which provides some indication as to the complexity of the TVP.

1.2. Idiosyncrasies. Before formally defining the problem statement, we introduce the symbols and notations we will employ throughout this paper. Denote a graph $G = (V, E)$ as a pair consisting of a set of vertices V , and a set of edges E . Let the map $w : E \mapsto \mathbb{R}$ be a weight function defined on the set of edges. We will denote an edge-weighted graph as a pair (G, w) . Thus we can easily generalize an unweighted graph $G = (V, E)$ as an edge-weighted graph (G, w) , by defining the weight function as

$$w_{ij} := \begin{cases} 1, & \text{if } (i, j) \in E, \\ 0, & \text{if } (i, j) \notin E. \end{cases} \quad (1)$$

We use the symbol “ $b := a$ ” to mean “the expression a defines the (new) symbol b ” in the sense of King [31]. Of course, this could be conveniently extended so that a statement like “ $(1 - \epsilon)/2 := 7$ ” means “define the symbol ϵ so that $(1 - \epsilon)/2 = 7$ holds.” We will employ the typical symbol S^c to denote the complement of the set S ; further let $A \setminus B$ denote the set-difference, $A \cap B^c$. Agree to let the expression $x \leftarrow y$ mean that the value of the variable y is assigned to the variable x . Also, let $|N|$ denote the cardinality of the set N . We define $\mathbb{Z}_k := \{0, 1, 2, \dots, k\} \cap \mathbb{Z}$, as the set of integers modulo k . Finally, we will use *italics* for emphasis and SMALL CAPS for problem names. Any other locally used terms and symbols will be defined in the sections in which they appear.

1.3. Organization. In this paper, we describe the implementation of a genetic algorithm for finding approximate solutions for the TVP. The encoding scheme is based on random keys [5]. The heuristic is then hybridized by the implementation of a local search procedure. Numerical results are presented demonstrating the effectiveness of the proposed procedure. The remainder of the paper is organized as follows. Next, we provide a mathematical model for the TVP and prove that finding an optimal solution is \mathcal{NP} -complete. Then, in Section 3 we describe a hybrid genetic algorithm for solving large instances. Computational results are provided in Section 4 comparing the proposed heuristic to a standard genetic algorithm and the optimal solutions as computed by a commercial integer programming package. Finally, we provide some concluding remarks in Section 5.

2. PROBLEM DESCRIPTION

In this section we provide a formal description of the TARGET VISITATION PROBLEM and discuss complexity issues. We also discuss the similarities between the TVP and other combinatorial problems. What follows is a brief survey of this nature.

2.1. Related Problems. Here we briefly examine the similarities between the TARGET VISITATION PROBLEM and two well-known problems in discrete optimization, namely the TRAVELING SALESMAN PROBLEM and the LINEAR ORDERING PROBLEM.

2.1.1. TRAVELING SALESMAN PROBLEM. The TRAVELING SALESMAN PROBLEM (TSP) is the most studied and widely recognized problem in operations research [11, 13, 17, 33, 40, 41, 45]. It has been the focus of research for over 50 years and remains a challenge even today. Given a graph $G = (V, E)$, a subset of edges $T \subseteq E$ is said to be a *traveling salesman tour* if it is a simple cycle of G having length $|V|$. In this context (and our’s) a tour is a Hamiltonian cycle, but this easily generalizes depending on one’s definition of a tour.

Suppose now that the graph is weighted (G, c) , where c_{ij} represents the cost of traversing arc $(i, j) \in E$ and that $|V| = n$. The objective is now to find a tour of minimum cost. Then if we represent a tour as a permutation π of the nodes, then the goal is to find π that minimizes

$$Z(\pi) = \sum_{i=1}^{n-1} c_{\pi(i), \pi(i+1)} + c_{\pi(n), \pi(1)}. \quad (2)$$

The original integer programming formulation of the TSP is due to Dantzig, Fulkerson, and Johnson (DSJ) [13] and continues to be a popular formulation today. Let $x : E \mapsto \{0, 1\}$ be a decision variable associated with each arc. Then the DSJ formulation of the TSP is given as the following 0-1 integer program [13].

$$\mathcal{DSJ} : \quad \max \sum_{i=1}^n \sum_{j=1}^n c_{ij} x_{ij} \quad (3)$$

subject to

$$\sum_{\substack{i=1 \\ i \neq j}}^n x_{ij} = 1, \forall j \in V, \quad (4)$$

$$\sum_{\substack{j=1 \\ j \neq i}}^n x_{ij} = 1, \forall i \in V, \quad (5)$$

$$\sum_{i \in S} \sum_{j \in S} x_{ij} \leq |S| - 1, \forall S \subset V, 2 \leq |S| \leq n - 1, \quad (6)$$

$$x_{ij} \in \{0, 1\}, \forall i, j \in V. \quad (7)$$

The DSJ formulation contains $n(n - 1)$ integer variables and 2^n constraints. While minimizing the total tour cost, the sets of constraints in (4) and (5) ensure that each node is visited exactly once. Constraint set (6) prevents subtours, by ensuring that feasible solutions are biconnected [45]. The major drawback of the DSJ formulation is the exponential number of subtour elimination constraints.

Another formulation of the TSP that is widely used in practice is due to Miller, Tucker, and Zemlin (MTZ), and was first published in 1960 [35]. The MTZ formulation reduces the number of subtour elimination constraints to be polynomially bounded, at the expense of increasing the number of decision variables. Let $s : V \mapsto \mathbb{R}$ be a bijection where

$$s_i := \text{the relative position of node } i \text{ in the tour.} \quad (8)$$

For example, if node 2 is visited third in the tour, then $s_2 = 3$. The s_i variables are commonly referred to as *sequencing variables* and can have many interpretations [15]. Without the loss of generality suppose that the depot, or point of origin is defined to be node 1. Using the decision variables defined above and the sequencing variables, we can formulate the MTZ TSP as follows:

$$\mathcal{MTZ} : \quad \max \sum_{i=1}^n \sum_{j=1}^n c_{ij} x_{ij} \quad (9)$$

subject to

$$\sum_{\substack{i=1 \\ i \neq j}}^n x_{ij} = 1, \forall j \in V, \quad (10)$$

$$\sum_{\substack{j=1 \\ j \neq i}}^n x_{ij} = 1, \forall i \in V, \quad (11)$$

$$y_i - y_j + n \cdot x_{ij} \leq n - 1, \forall i, j \in V \setminus \{1\}, i \neq j, \quad (12)$$

$$x_{ij} \in \{0, 1\}, \forall i, j \in V, \quad (13)$$

$$y_i \in \mathbb{R}, \forall i \in V \setminus \{1\}. \quad (14)$$

The MTZ formulation contains n^2 decision variables, and $\mathcal{O}(n^2)$ constraints. The sequencing variables used in this model allow considerable flexibility and the ability to model related problems easily. We take advantage of this for our formulation of the TVP in the

following section. It has been shown however that the MTZ formulation is weaker than the DSJ formulation [39]; however, the model can be strengthened by lifting additional edges as shown by Desrochers and Laporte [15].

These represent just a few of the formulations which have been proposed for solving TRAVELING SALESMAN PROBLEMS. For extensive review and analysis of other formulations, the reader is referred to [32] and [39]. As for the computational complexity of the TSP, it is well-known to be \mathcal{NP} -hard. Further, nonmetric instances of the TSP cannot be approximated within a constant factor unless $\mathcal{P} = \mathcal{NP}$ [17]. A thorough review of the problem is available in [28].

2.1.2. LINEAR ORDERING PROBLEM. The LINEAR ORDERING PROBLEM (LOP) is another optimal sequencing problem. Given a set N of n items and a corresponding matrix $\mathbf{D} = \{d_{i,j}\}_{n \times n}$, which represents the preferences for ordering item i before item j , the objective is to find an ordering of the items which maximizes the preferences [9]. Applications abound including ranking athletes or sports teams, ranking preferences to obtain ancestry relationships, and in economics [19]. In terms of the matrix \mathbf{D} , the optimal solution is a permutation π of the rows and columns of \mathbf{D} such that the sum of the values in the upper triangle are maximized. The LOP can be represented as a combinatorial optimization problem as follows:

$$\max Z(\pi) = \sum_{i=1}^{n-1} \sum_{j=i+1}^n d_{\pi(i),\pi(j)},$$

where $\pi(i)$ represents the item in position i of the permutation [26]. An equivalent graph theoretical problem is to find an acyclic tournament in a complete weighted graph in which the sum of the arc weights is maximal [34].

An integer programming formulation for the TVP can be computed as follows. Let $x : N \times N \mapsto \{0, 1\}$ be defined as

$$x_{ij} := \begin{cases} 1, & \text{if } i \text{ is ordered before } j, \\ 0, & \text{otherwise.} \end{cases} \quad (15)$$

Then the LOP admits the following integer programming formulation:

$$\mathcal{LOP} : \quad \max \sum_{(i,j) \in N} d_{ij} x_{ij} \quad (16)$$

subject to

$$x_{ij} + x_{ji} = 1, \quad \forall i, j \in N, \quad (17)$$

$$x_{ij} + x_{jk} + x_{ki} \leq 2, \quad \forall i, j, k \in N, i \neq j, i \neq k, j \neq k, \quad (18)$$

$$x_{i,j} \in \{0, 1\}, \quad \forall i, j \in N. \quad (19)$$

The LOP formulation consists of n^2 decision variables and $\mathcal{O}(n^3)$ constraints. Constraint set (17) ensures that each item is considered only once, thus enforcing the strict precedence relation. The constraints in (18) ensure that the solution is acyclic. Not surprisingly, the decision version of the LOP is known to be \mathcal{NP} -complete, and the corresponding optimization problem is \mathcal{NP} -hard [24, 25].

As we can see, the TARGET VISITATION PROBLEM combines attributes of both the TRAVELING SALESMAN PROBLEM and the LINEAR ORDERING PROBLEM. We are now ready to formally define the TVP and examine several math programming formulations.

2.2. TARGET VISITATION PROBLEM. An instance of the TARGET VISITATION PROBLEM consists of a set $N = \{1, 2, \dots, n\}$ of targets located at distinct points. There is also an associated distance matrix $\mathbf{D} = \{d_{i,j}\}_{m \times m}$, where $m := n + 1$. The $d_{i,j}$ entries represent the distances between nodes $i, j \in N$. Note that the distances may be asymmetric, i.e. $d_{i,j} \neq d_{j,i}$ necessarily. Also, for all targets i , there is a value $d_{0,i}$ which represents the distance from the UAVs point of origin to target i . Furthermore, a matrix $\mathbf{R} = \{\rho_{i,j}\}_{n \times n}$

is provided where $\rho_{i,j}$ represents the preference or utility of visiting target i before target j . This can be interpreted as the assumed “threat level” or relative importance of visiting one target before another. The intuition is that targets with higher priorities should be visited earlier in the sequence.

As mentioned in [26], obtaining the values of $d_{i,j}$ is usually an easy task since literal distance measures or other metrics such as travel time are available or are trivial to calculate. However, deriving the values of $\rho_{i,j}$, the value of visiting target i before target j is not always so simple. There are several methods used by military planners when developing routes for the TVP. The most common method, and the one we adopt in this paper, is known as “target value reconciliation” [26]. In this method a group of experts offer a set of pair-wise rankings for the targets from which the preference matrix is derived. More specifically, for all targets i and j , each expert is to specify a preference of visiting target i before j [10]. The value of $\rho_{i,j}$ is simply the cumulative number of experts who prefer to visit i before j .

A feasible solution for the TARGET VISITATION PROBLEM is one in which the UAV leaves its point of origin, visits all targets exactly once, and returns to the origin. The objective is to minimize the distance traveled while maximizing the utility of the visitation sequence [26]. Let π be a permutation of the set of integers $[1, \dots, n+1] \cap \mathbb{Z}$, such that $j =: \pi(i)$ implies that target j is the i^{th} position of the visitation sequence. With this, we can formulate the TVP as the following combinatorial optimization problem first given in [26]:

$$\text{Maximize } Z(\pi) = \left[\sum_{i=1}^{n-1} \sum_{j=i+1}^n \rho_{\pi(i),\pi(j)} \right] - \left[d_{0,\pi(1)} + \sum_{k=1}^{n-1} d_{\pi(k),\pi(k+1)} + d_{\pi(n),0} \right] \quad (20)$$

Permutation based models of combinatorial problems are often useful for gaining an intuitive understanding of the problem. However, integer programming models are usually the most helpful for providing some of the formal properties of the problem [37]. With this in mind we now develop a linear integer programming formulation for the TVP.

The TARGET VISITATION PROBLEM can be conveniently described as a combinatorial problem on a graph. Consider a doubly weighted directed graph (G, d, ρ) , where $V = \{0, 1, 2, \dots, n\}$ is the set of nodes. Suppose that V represents the set of targets, hence $n = |N|$. We include an extra node which represents the origin. Also, assume without the loss of generality that G is a complete graph. For each edge $(i, j) \in E$, there is an associated weight $d_{i,j}$ which represents the distance from target i to target j . Furthermore, for each edge $(i, j) \in E$, there is an associated value $\rho_{i,j}$ which is the preference for the corresponding target pair as described above. Now, let $x : V \times V \mapsto \{0, 1\}$ be a surjection defined by

$$x_{i,j} := \begin{cases} 1, & \text{iff } i = \pi(k) \Rightarrow j = \pi(k+1), \text{ for } k \in \mathbb{Z}_n, \\ 0, & \text{otherwise,} \end{cases} \quad (21)$$

where π is defined as above. Said differently, $x_{i,j} = 1$ implies that $(i, j) \in E$ is a link in the tour. Next, we introduce another surjective function $w : V \times V \mapsto \{0, 1\}$ defined by

$$w_{i,j} := \begin{cases} 1, & \text{iff } i = \pi(k) \Rightarrow j = \pi(l), \text{ for } k, l \in \mathbb{Z}_n \text{ such that } k < l, \\ 0, & \text{otherwise.} \end{cases} \quad (22)$$

Finally, we have a bijection $y : V \mapsto \mathbb{R}$, where

$$y_i := \text{sequence in which target } i \text{ is visited.} \quad (23)$$

With this we can formulate the TARGET VISITATION PROBLEM as the following integer linear program.

$$TV\mathcal{P}1 : \quad \max \sum_{i=1}^n \sum_{\substack{j=1 \\ i \neq j}}^n \rho_{i,j} w_{i,j} - \sum_{i=0}^n \sum_{\substack{j=0 \\ i \neq j}}^n d_{i,j} x_{i,j} \quad (24)$$

subject to

$$\sum_{\substack{i=0 \\ i \neq j}}^n x_{i,j} = 1, \quad \forall j \in V, \quad (25)$$

$$\sum_{\substack{j=0 \\ j \neq i}}^n x_{i,j} = 1, \quad \forall i \in V, \quad (26)$$

$$y_i - y_j + n \cdot x_{i,j} \leq n - 1, \quad \forall i, j \in V \setminus \{0\}, i \neq j, \quad (27)$$

$$w_{i,j} + w_{j,i} \leq 1, \quad \forall i, j \in V, i \neq j, \quad (28)$$

$$y_i - y_j + n \cdot w_{i,j} \geq 0, \quad \forall i, j \in V \setminus \{0\}, i \neq j, \quad (29)$$

$$x_{i,j} \in \{0, 1\}, \quad \forall i, j \in V, \quad (30)$$

$$w_{i,j} \in \{0, 1\}, \quad \forall i, j \in V, \quad (31)$$

$$y_i \in \mathbb{R}, \quad \forall i \in V \setminus \{0\}. \quad (32)$$

This formulation has a total of $3n^2 - 5n + 4$ constraints and $2n^2 - n - 1$ integer variables.

Theorem 1. *The integer programming formulation TVP1 is a correct formulation of the TARGET VISITATION PROBLEM.*

Proof. Notice that in graph theoretical terms, the objective of the TVP is to find a Hamiltonian cycle which is of minimum weight, but which also maximizes the visitation preferences. This is accomplished by the objective function in (33). The $2n$ assignment constraints in (34) and (35) ensure that each target is visited only once in the tour. The $n^2 - 3n + 2$ constraints in (36) are subtour elimination constraints and hence prevent disjoint cycles from occurring in the tour.

The constraints in (28) ensure that only one of $w_{i,j}$ or $w_{j,i}$ is nonzero for all (i, j) pairs. In order to ensure that $w_{i,j} = 1$ only when i is visited before j , we have the $\mathcal{O}(n^2)$ constraints in (29). They insist that $w_{i,j}$ is nonzero only when $y_i < y_j$. Finally, (38), (39), and (40), define the proper domains for the variables used. Thus, a solution to the integer programming formulation $\mathcal{P}1$ characterizes a feasible solution to the TVP. On the other hand, it is clear that a feasible solution to the TVP will define at least one feasible solution to $\mathcal{P}1$. Therefore, $\mathcal{P}1$ is a correct formulation for the TVP. \square

We have shown that formulation TVP1 is correct formulation for the TVP. However, it is possible to formulate a more compact integer programming model which reduces the number of constraints by n^2 . This model is based on the MTZ formulation for the TSP. It is given as follows.

$$TVP2 : \quad \max \sum_{i=1}^n \sum_{\substack{j=1 \\ i \neq j}}^n \rho_{i,j} w_{i,j} - \sum_{i=0}^n \sum_{\substack{j=0 \\ i \neq j}}^n d_{i,j} x_{i,j} \quad (33)$$

subject to

$$\sum_{\substack{i=0 \\ i \neq j}}^n x_{i,j} = 1, \quad \forall j \in V, \quad (34)$$

$$\sum_{\substack{j=0 \\ j \neq i}}^n x_{i,j} = 1, \quad \forall i \in V, \quad (35)$$

$$y_i - y_j + n \cdot w_{i,j} \leq n - 1, \quad \forall i, j \in V \setminus \{0\}, i \neq j, \quad (36)$$

$$x_{i,j} \leq w_{i,j}, \quad \forall i, j \in V, \quad (37)$$

$$x_{i,j} \in \{0, 1\}, \quad \forall i, j \in V, \quad (38)$$

$$w_{i,j} \in \{0, 1\}, \quad \forall i, j \in V, \quad (39)$$

$$y_i \in \mathbb{R}, \quad \forall i \in V \setminus \{0\}. \quad (40)$$

Theorem 2. *Formulation TVP2 is correct.*

Proof. Constraint sets (34) and (35) imply that the indegree and outdegree, in the edge induced subgraph of the solution set, has to be one for all the nodes. Constraint set (36) and (37) together ensures that there cannot be subtours, as this is just the MTZ based formulation for TSP. In order to prove that $w_{ij} \neq w_{ji}$, let us consider constraint set (36). If i is visited before j , then we have $y_i - y_j < 0$ and maximization of the objective function ensures that w_{ij} is one. If j is visited before i , then we have $y_i - y_j > 0$ and hence w_{ij} has to be zero. \square

The TVP represents a set of combinatorial decisions that must be made [37]. Clearly, for any asymmetric instance consisting of n targets there are $n!$ possible routes to consider. Now that we have an integer programming model for the TVP, we can examine the computational complexity. It is not surprising that finding an optimal solution is \mathcal{NP} -hard as we will now show by proving that the recognition version of the problem is \mathcal{NP} -complete. The recognition version of the TVP can be stated as follows: (**K-TVP**) Given an instance of the TARGET VISITATION PROBLEM, does there exist a tour of cost less than or equal to K ?

Theorem 3. *The decision version of the TARGET VISITATION PROBLEM (**K-TVP**) is \mathcal{NP} -complete.*

Proof. To show this, we must prove that (1) $\mathbf{K-TVP} \in \mathcal{NP}$; (2) Some \mathcal{NP} -complete problem reduces to $\mathbf{K-TVP}$ in polynomial time.

- (1) Clearly $\mathbf{K-TVP} \in \mathcal{NP}$ since any solution can be verified in polynomial time to be feasible or not.
- (2) To complete the proof, we show a simple reduction from the HAMILTONIAN CYCLE PROBLEM which is well-known to be \mathcal{NP} -complete [17]. Let $G = (V, E)$ be a graph in which a Hamiltonian cycle has to be determined. Construct a complete graph $\bar{G} = (V, \bar{E})$ with arc distance 1 if $(i, j) \notin E$ and 0 otherwise. Furthermore, construct the preference matrix such that $\rho_{i,j} = k$, where $k \in \mathbb{R}$, for all (i, j) pairs. The objective of the decision problem $\mathbf{K-TVP}$ is to determine if a solution exists with cost $K \leq k \cdot n$. A ‘yes’ instance of the HAMILTONIAN CYCLE PROBLEM on G corresponds to a ‘yes’ instance for the $\mathbf{K-TVP}$ on \bar{G} . Notice that the

cost of the K-TVP tour is simply the sum of the preference costs, i.e. $k \cdot n$, as the distance component of the objective function is zero.

To prove the converse, observe that the cost of any K-TVP tour is at least $k \cdot n$. Thus a ‘yes’ instance of K-TVP would mean that all of the arcs in the tour have zero cost. This implies all of the arcs in the K-TVP tour are present in the graph G and also form a valid tour in G . This results in a ‘yes’ instance for the HAMILTONIAN CYCLE PROBLEM. Thus the proof is complete. \square

As noted in [26], it might be the case that for a particular instance of the TVP, the terms of one of the matrices in the objective function may dominate the other. However, both distance and utility are important factors and should be given equal attention in a solution. Therefore, we use a simple balancing heuristic first given by Grundel and Jeffcoat in [26]. Let π_r be a random permutation of the targets to be visited. Further, define $\gamma \in \mathbb{R}$ such that $\mathbf{R} := \gamma \mathbf{R}$. In order to normalize the \mathbf{D} and \mathbf{R} matrices, we adjust the particular value of γ so that

$$\frac{\sum_{i=1}^{n-1} \sum_{j=i+1}^n \tilde{\rho}_{\pi_r(i), \pi_r(j)}}{d_{0, \pi_r(1)} + \sum_{i=1}^{n-1} d_{\pi_r(i), \pi_r(i+1)} + d_{\pi_r(n), 0}} \approx 1. \quad (41)$$

Then without the loss of generality, the parameter γ can be used to weight either matrix if it is determined that the one of the distance or preference components is more important than the other. Hence increasing the value of γ places more importance on the utility of the visitation sequence relative to the total distance traveled [26].

The complexity of the TVP motivates the need for efficient heuristics since finding optimal solutions for large instances is impractical. Therefore, in the next section we propose the use of genetic algorithm for finding near optimal solutions for the TVP.

3. GENETIC ALGORITHM

Genetic algorithms (GAs) get their name from the biological process which they mimic. Motivated by Darwin’s Theory of Natural Selection [14], these algorithms evolve a *population* of solutions, called *individuals*, over several *generations* until the best solution is eventually reached. Each component of an individual is called a *allele*. Individuals in the population mate through a process called *crossover*, and new solutions having traits, i.e. alleles of both parents, are produced. In successive generations, only those solutions having the best *fitness* are carried to the next generation in a process which mimics the fundamental principle of natural selection, *survival of the fittest* [20]. Figure 2 provides pseudo-code for a standard genetic algorithm. Though the GA does converge in probability to the optimal solution, it is common to stop the procedure after some “terminating condition” (see line 3) is satisfied. This condition could be one of several things including, a maximum running time, a target objective value, or a limit on the number of generations. For our implementation, we use the latter option and the best solution after MaxGen generations is returned.

When designing a genetic algorithm for an optimization problem, one must provide a means to encode the population, define the crossover operator, and define the *mutation* operator which allows for random changes in offspring to help prevent the algorithm from converging prematurely. The encoding scheme we propose for our GA is based on random keys and follows exactly as described by Bean [5]. As mentioned in [5], GAs often have a difficult time maintaining feasibility of solutions in successive generations. This problem is overcome by the use of random keys as an encoding mechanism for the population. Random keys work by encoding the solution vector using random numbers. The feasibility issue is then moved into the objective function, and subsequently all offspring produced are guaranteed to be feasible solutions.

```

procedure GeneticAlgorithm
1  Generate population  $P_k$ 
2  Evaluate population  $P_k$ 
3  while terminating condition not met do
4    Select individuals from  $P_k$  and copy to  $P_{k+1}$ 
5    Crossover individuals from  $P_k$  and put in  $P_{k+1}$ 
6    Mutate individuals from  $P_k$  and put in  $P_{k+1}$ 
7    Evaluate population  $P_{k+1}$ 
8     $P_k \leftarrow P_{k+1}$ 
9     $P_{k+1} \leftarrow \emptyset$ 
10 end while
11 return best individual in  $P_k$ 
end procedure GeneticAlgorithm

```

FIGURE 2. Pseudo-code for generic genetic algorithm.

For the GA implementation for the TVP, we have the following definitions. As mentioned above, solutions are represented by a random vector. To determine the visitation sequence, a random deviate from a distribution which is uniform onto $(0, 1) \in \mathbb{R}$ is generated for each target. The tour is determined by sorting the random numbers and sequencing the targets in descending order of the sort. For example, suppose there are $n = 3$ targets to visit. Then a chromosome such as

$$(.34, .71, .28)$$

would correspond to the visitation sequence

$$2 \rightarrow 1 \rightarrow 3.$$

The objective value of the sequence can be evaluated, thus determining the fitness of the chromosome.

3.1. Evolutionary Mechanisms. In order to evolve the population over successive generations, we use a reproduction method which copies the best individuals in the current generation to the next. We aptly refer to this set the BEST set. This technique ensures that the best solution is monotonically improving in every generation [5]. To breed new solutions, we implement a strategy known as *parameterized uniform crossover* [44]. This method works by selecting two solutions to serve as parents. In our implementation, one parent is chosen at random from the BEST set, and the other is chosen from the entire population (including BEST). Then, for each target to be visited, a biased coin is tossed. If the result is heads, then the allele of the BEST parent is chosen, otherwise the allele is taken from the other parent. The probability that the coin lands on heads is known as `CrossProb`, and is determined empirically. Figure 3 provides an example of a potential crossover when the number of targets is 5 and `CrossProb` = 0.65.

| Coin Toss | T | H | H | T | H |
|-----------|------|------|------|------|------|
| Parent 1 | 0.56 | 0.81 | 0.22 | 0.7 | 0.86 |
| Parent 2 | 0.29 | 0.49 | 0.98 | 0.12 | 0.32 |
| Offspring | 0.29 | 0.81 | 0.22 | 0.12 | 0.86 |

FIGURE 3. An example of the crossover operation. In this case, `CrossProb` = 0.65.

Finally, the mutation operator is defined as follows. Instead of introducing random perturbations to selected offspring, we instead replace a set of individuals having the worst

fitness with new solutions generated at random from the same distribution as the original population. This replacement set is referred to as the WORST set. Using this method, we are able to ensure that the GA does not converge prematurely. This is a common method, sometimes referred to as *immigration* and appears throughout the literature [5, 22]. An overall pictorial view of the generational evolution of the proposed GA is provided in Figure 4.

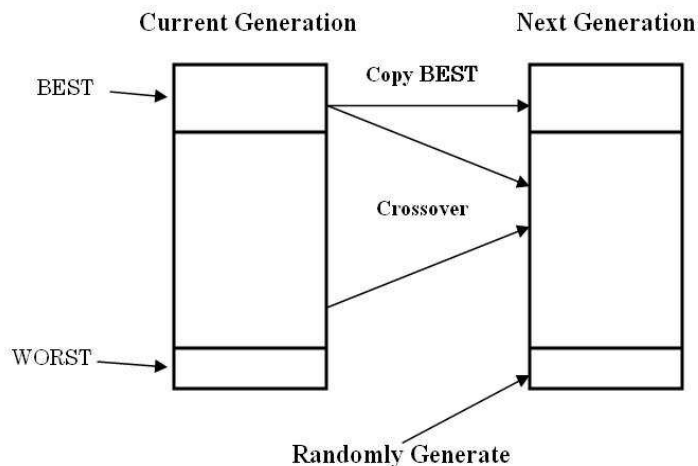


FIGURE 4. Graphical representation of generational evolution.

3.2. Local Search. In addition to the standard GA, we propose a hybridization technique to produce better solutions. In particular we implement a 2-exchange local search on each of the offspring produced by crossover operator. Pseudo-code for this heuristic is provided in Figure 5. A 2-exchange local search is a hill-climbing procedure which examines pairs of alleles and performs a swap. If the resulting swap increases the fitness of the individual, the swap is kept. Otherwise, it is undone and another pair is examined. Such local improvement methods abound in the literature and are used to enhance methods such as greedy randomized adaptive search procedures (GRASP) [43], tabu search [18], and other combinatorial optimization heuristics [1].

4. COMPUTATIONAL RESULTS

The proposed heuristic was implemented in the C++ programming language and compiled using GNU g++ version 3.4.4, using optimization flags -O2. It was tested on a PC equipped with a 1700MHz Intel® Pentium® M processor and 1.0 gigabytes of RAM operating under the Microsoft® Windows® XP Professional environment.

As a basis for comparison, we examine the results for the hybrid genetic algorithm (HGA) with the standard genetic algorithm (GA). In addition, we have implemented the integer programming model for the TARGET VISITATION PROBLEM using the CPLEX™ version 10 optimization suite from ILOG [12]. CPLEX contains an implementation of the simplex method [29], and uses a branch and bound algorithm [45] together with advanced cutting-plane techniques [30, 38]. The instances were tested using the CPLEX default settings. The algorithms were tested on a set of randomly generated instances varying in size from 8-16 targets¹. Due to the complexity of the problem, CPLEX was unable to obtain optimal solutions for instances with $|N| > 16$. For each instance, the number of

¹The test problems may be downloaded from <http://plaza.ufl.edu/clayton8/tvp.tar.gz>.

```

procedure LocalSearch( $X$ )
1   $X^* \leftarrow X$ 
2   $f(X^*) \leftarrow f(X)$ 
3  temp  $\leftarrow 0$ 
4  while  $X$  is not locally optimal do
5      for  $i = 1$  to  $|X|$  do
6          for  $j = 1$  to  $|X|$  do
7              temp  $\leftarrow X(i)$ 
8               $X(i) \leftarrow X(j)$ 
9               $X(j) \leftarrow$  temp
10             if  $f(X) > f(X^*)$  then
11                  $X^* \leftarrow X$ 
12             else
13                 temp  $\leftarrow X(i)$  /* undo swap */
14                  $X(i) \leftarrow X(j)$ 
15                  $X(j) \leftarrow$  temp
16             end if
17         end for
18     end for
19 end while
20 return ( $X^*$ )
end procedure LocalSearch

```

FIGURE 5. 2-exchange local search.

“experts” used to derive the utility matrix is 10. Also, the matrices were balanced using the heuristic described in Equation (41) above. For each instance, the maximum distance between the targets varied from 20 to 150 units. The distance matrix for each instance was generated uniformly at random with some user defined upper and lower bounds. The instances were created using a simple problem generator written in C++. The instances used in this paper are non-symmetric and non-metric; however, this option is built in to the generator. It is a reasonable assumption to have non-symmetric instances, since in a real-world scenario the matrix \mathbf{D} might represent other factors than simply the distance between two targets such as an extra cost or risk associated with visiting a particular target. In this case $d_{ij} \neq d_{ji}$ necessarily. Furthermore, it is assumed that the UAV is capable of traveling all cycles in the graph. Depending on the individual application and factors such as the number of targets and the size of the battlespace, it may be reasonable to impose a hard constraint on the total distance traveled in the TVP tour. This would model problem scenarios when battery consumption and fuel capacity are critical. In the instances tested we do not impose such constraints; however doing so would most likely reduce the overall CPLEX running time.

We mention here that in the instances considered, the priority functions have been normalized by the distance function in order to avoid the domination of one cost function by the other. This justifies the cost of objective function. For a better realization of this fact, we made test runs on the instances solving both the TSP and LOP problems separately. These results are included in Table 1. To further illustrate this, Table 2 provides the objective function value of the LOP, TVP, and TSP when one objective is considered and solved to optimality. The LOP solution (non-optimal component) corresponding to the TSP optimal route (optimal component) and the TSP solution (non-optimal component) corresponding to the optimal LOP route (optimal component) have been presented. The non-optimal components are weaker when compared to each component in the optimal TVP solution. This numerical evidence supports the claim that in order to find high quality TVP routes, it is

not advisable to decouple the problem into simply a TRAVELING SALESMAN PROBLEM or LINEAR ORDERING PROBLEM, but rather to consider both objectives in concert.

The alternative way of solving a biobjective optimization problem is by determining a set of efficient solutions. An solution is efficient if both the cost functions are dominated by any other solution [42]. This is achieved by treating the problem with a single objective function and recursively solving the problem by recomputing the new costs based on the cost obtained from the previous iteration. This however assumes that the both the cost functions involves with the same problem. In our case, we have one cost function to be employed for a TSP problem and another cost function for a LOP problem and an efficient algorithm for solving both the problems has to be designed for this purpose, which is beyond the scope of this paper.

TABLE 1. Comparative results of the optimal solutions to the corresponding TSP, LOP, and TVP for each instance. The absolute value of the TSP solutions are reported.

| Instance | | LOP | TVP | TSP |
|----------|---------|------------------|------------------|------------------|
| Name | Targets | Optimal Solution | Optimal Solution | Optimal Solution |
| rand8-1 | 8 | 110.085 | 60.2766 | 31 |
| rand8-2 | 8 | 197.139 | 115.944 | 55 |
| rand8-3 | 8 | 335.74 | 195.333 | 76 |
| rand8-4 | 8 | 59.2222 | 29.0074 | 20 |
| rand8-5 | 8 | 646.16 | 314 | 221 |
| rand10-1 | 10 | 259.926 | 157.404 | 74 |
| rand10-2 | 10 | 247 | 208 | 21 |
| rand10-3 | 10 | 734.569 | 520.679 | 149 |
| rand10-4 | 10 | 720.167 | 532.5 | 97 |
| rand10-5 | 10 | 565.781 | 365.125 | 105 |
| rand12-1 | 12 | 208.29 | 124.179 | 56 |
| rand12-2 | 12 | 491.677 | 318.38 | 104 |
| rand12-3 | 12 | 646.772 | 420.959 | 142 |
| rand12-4 | 12 | 944.093 | 594.546 | 169 |
| rand12-5 | 12 | 640.055 | 472.354 | 89 |
| rand14-1 | 14 | 204.414 | 137.609 | 39 |
| rand14-2 | 14 | 549.708 | 405.774 | 72 |
| rand14-3 | 14 | 897.804 | 631.711 | 153 |
| rand14-4 | 14 | 292.389 | 176.631 | 65 |
| rand14-5 | 14 | 976.921 | 679.625 | 131 |
| rand16-1 | 16 | 518.62 | 381.934 | 63 |
| rand16-2 | 16 | 706.98 | 431.531 | 164 |
| rand16-3 | 16 | 571.735 | 415.339 | 99 |
| rand16-4 | 16 | 707.22 | 421.658 | 162 |
| rand16-5 | 16 | 364.152 | 249.939 | 68 |

4.1. Numerical Results. We begin by presenting some comparative results of the heuristics. As Gonçalves et al. correctly indicate, despite the massive amounts of literature on genetic algorithms, there is little knowledge of how best to tune the parameters for a given application [23]. For all of the instances tested, the parameters used for the genetic algorithm (GA) and the hybrid genetic algorithm (HGA) are given in Table 3. It has been shown in the literature that parameters similar to those we implemented have been effective when implementing a hybrid GA [6, 21, 22, 23]. Table 4 presents the comparative results

TABLE 2. The corresponding objective function values of each of the LOP, TSP, and TVP are given for each instance. For each column, one of the objectives is considered and the problem solved to optimality. The solution of the remaining two problems is given when evaluated with the optimal function value.

| Instance | | LOP opt | | TVP opt | | TSP opt | |
|----------|---------|---------|---------|---------|---------|---------|---------|
| Name | Targets | TSP | LOP | TSP | LOP | TSP | LOP |
| rand8-1 | 8 | -100 | 110.085 | -31 | 91.2766 | -31 | 91.2766 |
| rand8-2 | 8 | -174 | 197.139 | -77 | 192.944 | -55 | 161.486 |
| rand8-3 | 8 | -242 | 335.74 | -115 | 310.333 | -76 | 219.592 |
| rand8-4 | 8 | -39 | 59.222 | -29 | 58.0075 | -20 | 44.6444 |
| rand8-5 | 8 | -502 | 646.16 | -221 | 574 | -221 | 501.84 |
| rand10-1 | 10 | -139 | 259.926 | -94 | 251.404 | -74 | 203.68 |
| rand10-2 | 10 | -55 | 247 | -37 | 245 | -21 | 195 |
| rand10-3 | 10 | -648 | 734.569 | -199 | 719.679 | -149 | 575.743 |
| rand10-4 | 10 | -401 | 720.167 | -120 | 652.5 | -97 | 580 |
| rand10-5 | 10 | -344 | 565.781 | -123 | 488.125 | -105 | 391.979 |
| rand12-1 | 12 | -184 | 208.29 | -64 | 188.179 | -56 | 170.941 |
| rand12-2 | 12 | -328 | 491.677 | -110 | 428.38 | -104 | 392.211 |
| rand12-3 | 12 | -379 | 646.772 | -149 | 569.959 | -142 | 514.653 |
| rand12-4 | 12 | -721 | 944.093 | -279 | 873.546 | -169 | 674.352 |
| rand12-5 | 12 | -513 | 640.055 | -138 | 610.354 | -89 | 512.341 |
| rand14-1 | 14 | -150 | 204.414 | -48 | 185.609 | -39 | 170.287 |
| rand14-2 | 14 | -336 | 549.708 | -84 | 489.774 | -72 | 427.967 |
| rand14-3 | 14 | -707 | 897.804 | -157 | 788.711 | -153 | 670.773 |
| rand14-4 | 14 | -280 | 292.389 | -93 | 269.631 | -65 | 195.421 |
| rand14-5 | 14 | -774 | 976.921 | -176 | 855.625 | -131 | 708.104 |
| rand16-1 | 16 | -399 | 518.62 | -92 | 473.934 | -63 | 421.125 |
| rand16-2 | 16 | -671 | 706.98 | -185 | 616.934 | -164 | 564.846 |
| rand16-3 | 16 | -392 | 571.735 | -110 | 525.338 | -99 | 457.987 |
| rand16-4 | 16 | -693 | 707.22 | -178 | 599.658 | -162 | 565.597 |
| rand16-5 | 16 | -290 | 364.152 | -83 | 332.939 | -68 | 288.484 |

TABLE 3. Parameters used for the GA and HGA heuristics.

| | |
|-----------------|---------------------------------------|
| CrossProb = 0.7 | Population Size (PopSize) = $2 * N $ |
| MaxGen = 10000 | BEST = $.1 * \text{PopSize}$ |
| | WORST = $.2 * \text{PopSize}$ |

of the HGA and standard GA applied to 25 randomly generated instances. The number of targets ranges from 8 to 16. Five instances were tested for each value of $|N|$. The table provides the instance name and the corresponding number of targets. Next the optimal solutions are provided along with the corresponding computation time required by CPLEX. We tested each of the heuristics 250 times on each instance, and we provide the maximum, minimum, and average solutions computed for each. Finally for both heuristics, we provide the average computation time for the 250 runs as well as the average deviation from the optimum.

Notice that for all 6250 experiments, the hybrid GA computed optimal solutions 99.93% of the time requiring 2.687 seconds of computation time on average. The compares favorably with the average time required by CPLEX to compute the optimal solutions which was 601.428 seconds. We note here that a heuristic solution was used as a starting point for the

computation of the optimal solutions for the instances containing 16 targets. Without this, the running time for CPLEX for these problems was on the order of 75000 seconds. We see also that the standard genetic algorithm performed reasonably well, with an average optimality gap of 4.429%. In addition, the standard GA scaled well averaging less than one half second of computation time for all instances. However, we see that ultimately the hybridized algorithm was the most robust of the two methods. For the HGA, the increase in the average solution time for the large instances is arguably offset by its stellar performance. To the contrary, one might argue that given more time the performance of the standard GA would match that of the hybrid method. However, in the next subsection we perform a probabilistic analysis on the time required for each heuristic to compute a target value, and we shall see that this argument is ultimately untrue.

4.2. Time-to-Target Plots. In this subsection, we investigate the empirical distributions of the heuristic running times. It has been observed [2, 4, 16] that solution times for stochastic heuristics such as genetic algorithms, tabu search, and GRASP fit a two-parameter shifted exponential distribution [3]. More specifically, let $P : \mathbb{R} \mapsto [0, 1]$ be a probability measure on a Borel set. Then the probability of *not* finding a target solution in t time units is given by $P(t) := e^{-(t-\mu)/\lambda}$, where $\lambda \in \mathbb{R}^+$ and $\mu \in \mathbb{R}$.

For each instance, we make 100 runs of both the hybrid GA and the standard GA. The runs are assumed to be independent since each was done using a different seed for the random number generator. For each instance/heuristic pair, the algorithms ran until they calculated a target solution and the required computation time was recorded. Then for each instance, the running times for each heuristic was sorted in descending order. The i^{th} sorted running time, t_i is associated with the probability $p_i := (i - \frac{1}{2})/100$ and the point $z_i = (p_i, t_i)$, for all $i = 1, \dots, 100$ [16]. The z_i points were then plotted in what is known as a *Time-to-Target Plot* (TTTplot) using the publicly available perl software `tttplots`² by Aiex, Resende, and Ribeiro [3].

Since it is unreasonable to provide a TTTplot for each instance tested, instead we provide a representative subset all the instances in Figures 6 - 8. Notice that for all cases the hybrid GA converges faster than the standard GA which is visualized by the fact that the HGA curves are completely to the left of the standard GA curves in all the TTTplots. The TTTplots imply that for a fixed amount of time, the hybrid method has a higher probability of reaching the target solution. For example, consider the plot for instance `rand12-1` shown in Figure 6. We see that given one second of computing time, the probability that the hybrid GA will compute the optimal solution is 0.926, compared with a probability of 0.443 for the standard GA. Likewise, for a fixed probability, the plot indicates that the hybrid method will find the target solution quicker than the simple genetic algorithm. Consider the TTTplot for instance `rand14-2` in Figure 7. In order for the GA to compute the target solution for a fixed probability of 0.6, approximately 10.6 seconds of computation time are required. The hybrid GA requires 2.32 seconds to find the target solution with 60% success. The TTTplots particularly highlight the scalability and robustness of the hybrid genetic algorithm when tested on larger instances as indicated by the near vertical plots of the HGA probabilities in Figure 8. The main point to be made here is not to argue that the hybrid genetic algorithm outperforms the standard metaheuristic in terms of objection function value. Of course, it has been shown that the standard genetic algorithm will converge to the optimal solution with probability 1. What we have demonstrated is that by enhancing the metaheuristic with the application of a local search, we are able to dramatically decrease the computation time required to converge to the optimal solution. We can conclude that for very large-scale instances of combinatorial problems such as the TVP, the advantage of using the hybrid GA enables us to find high quality solutions much faster than the basic genetic algorithm. For problems involving military applications such

²Available at <http://www.research.att.com/~mgcr/tttplots/>.

TABLE 4. This table provides the numerical results for a set of randomly generated instances. The first columns provide information about the instance. Next, the optimal solution and required computation time is listed. Both the HGA and the standard GA were ran 250 times on each instance, and we provide the maximum, minimum, and average solutions computed by each for all 250 tests. The average computation time required by each heuristic to compute the best solution is also listed.

| Instance | | IP Model | | Hybrid GA | | | | | Standard GA | | | | |
|----------|---------|------------------|--------------------|-----------|----------|-----------|---------------|--------------|-------------|----------|-----------|---------------|--------------|
| Name | Targets | Optimal Solution | Execution Time (s) | Max Soln | Min Soln | Avg. Soln | Avg. Time (s) | Avg. Dev (%) | Max Soln | Min Soln | Avg. Soln | Avg. Time (s) | Avg. Dev (%) |
| rand8-1 | 8 | 60.2766 | 0.01 | 60.2766 | 60.2766 | 60.2766 | 0.005 | - | 60.2766 | 56.3404 | 59.8895 | 0.054 | 0.642 |
| rand8-2 | 8 | 115.944 | 0.02 | 115.944 | 115.944 | 115.944 | 0.047 | - | 115.944 | 112.653 | 115.681 | 0.044 | 0.27 |
| rand8-3 | 8 | 195.333 | 0.01 | 195.333 | 195.333 | 195.333 | 0.011 | - | 195.333 | 194.96 | 188.555 | 0.032 | 5.006 |
| rand8-4 | 8 | 29.0074 | 0.02 | 29.0074 | 29.0074 | 29.0074 | 0.027 | - | 29.0074 | 25.8592 | 28.888 | 0.052 | 0.412 |
| rand8-5 | 8 | 314 | 0.03 | 314 | 314 | 314 | 0.001 | - | 314 | 314 | 314 | 0.008 | - |
| rand10-1 | 10 | 157.404 | 3.01 | 157.404 | 157.404 | 157.404 | 0.111 | - | 157.404 | 140.133 | 140.133 | 0.123 | 10.972 |
| rand10-2 | 10 | 208 | 2.87 | 208 | 204 | 207.984 | 0.307 | 0.008 | 208 | 200 | 207.36 | 0.044 | 0.308 |
| rand10-3 | 10 | 520.679 | 0.01 | 520.679 | 520.679 | 520.679 | 0.092 | - | 520.679 | 437.12 | 518.698 | 0.049 | 0.380 |
| rand10-4 | 10 | 532.5 | 2.45 | 532.5 | 532.5 | 532.5 | 0.059 | - | 532.5 | 489.667 | 529.891 | 0.107 | 0.49 |
| rand10-5 | 10 | 365.125 | 4.87 | 365.125 | 365.125 | 365.125 | 0.034 | - | 365.125 | 303.615 | 349.457 | 0.068 | 4.291 |
| rand12-1 | 12 | 124.179 | 45.37 | 124.179 | 124.179 | 124.179 | 0.129 | - | 124.179 | 106.645 | 121.022 | 0.148 | 2.54 |
| rand12-2 | 12 | 318.38 | 61.17 | 318.38 | 318.38 | 318.38 | 0.039 | - | 318.38 | 266.641 | 308.668 | 0.128 | 3.050 |
| rand12-3 | 12 | 420.959 | 51.89 | 420.959 | 420.959 | 420.959 | 0.191 | - | 420.959 | 341.866 | 403.31 | 0.0951 | 4.193 |
| rand12-4 | 12 | 594.546 | 16.44 | 594.546 | 594.546 | 594.546 | 0.427 | - | 594.546 | 487.099 | 580.956 | 0.137 | 2.286 |
| rand12-5 | 12 | 472.354 | 14.68 | 472.354 | 472.354 | 472.354 | 0.305 | - | 472.354 | 409.102 | 456.735 | 0.131 | 3.307 |
| rand14-1 | 14 | 137.609 | 303.55 | 137.609 | 137.609 | 137.609 | 0.792 | - | 137.609 | 110.948 | 128.208 | 0.225 | 6.832 |
| rand14-2 | 14 | 405.774 | 370.01 | 405.774 | 397.503 | 405.741 | 2.128 | 0.008 | 405.774 | 334.807 | 383.21 | 0.29 | 5.561 |
| rand14-3 | 14 | 631.711 | 184.82 | 631.711 | 614.917 | 631.644 | 2.795 | 0.011 | 631.711 | 508.412 | 594.765 | 0.267 | 5.849 |
| rand14-4 | 14 | 176.631 | 301.71 | 176.631 | 172.789 | 176.603 | 3.148 | 0.016 | 176.631 | 146.979 | 164.377 | 0.250 | 6.938 |
| rand14-5 | 14 | 679.625 | 2700.78 | 679.625 | 679.625 | 679.625 | 1.546 | - | 679.625 | 530.617 | 638.161 | 0.225 | 6.101 |
| rand16-1 | 16 | 381.934 | 1353.77 | 381.934 | 376.039 | 381.62 | 8.954 | 0.082 | 381.934 | 298.207 | 351.275 | 0.351 | 8.027 |
| rand16-2 | 16 | 431.531 | 2556.77 | 431.531 | 414.606 | 428.75 | 10.082 | 0.645 | 431.531 | 333 | 387.659 | 0.322 | 10.167 |
| rand16-3 | 16 | 415.338 | 462.5 | 415.338 | 408.324 | 415.074 | 13.358 | 0.064 | 415.338 | 332.868 | 380.319 | 0.329 | 8.431 |
| rand16-4 | 16 | 421.658 | 2810.45 | 421.658 | 409.171 | 419.305 | 12.903 | 0.558 | 417.109 | 314.109 | 386.355 | 0.397 | 8.372 |
| rand16-5 | 16 | 249.939 | 3788.49 | 249.939 | 243.534 | 249.099 | 9.676 | 0.336 | 249.939 | 187.592 | 234.192 | 0.326 | 6.3 |

as the TVP time is usually critical. Time spent searching for a good solution can lead to the loss of equipment or the death of personnel in the battlefield. In these cases, the advantage of the hybrid method is clear. The quicker a solution can be computed, the faster the system can be deployed and the competitive advantage is retained on the battlefield.

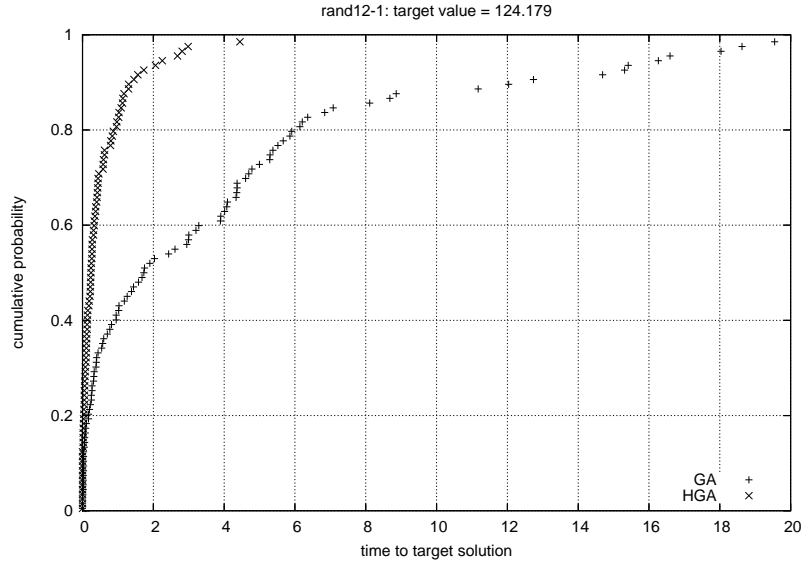


FIGURE 6. Time-to-Target plot comparing the Hybrid GA and standard GA for instance rand12-1. The target value is the optimal solution for the problem.

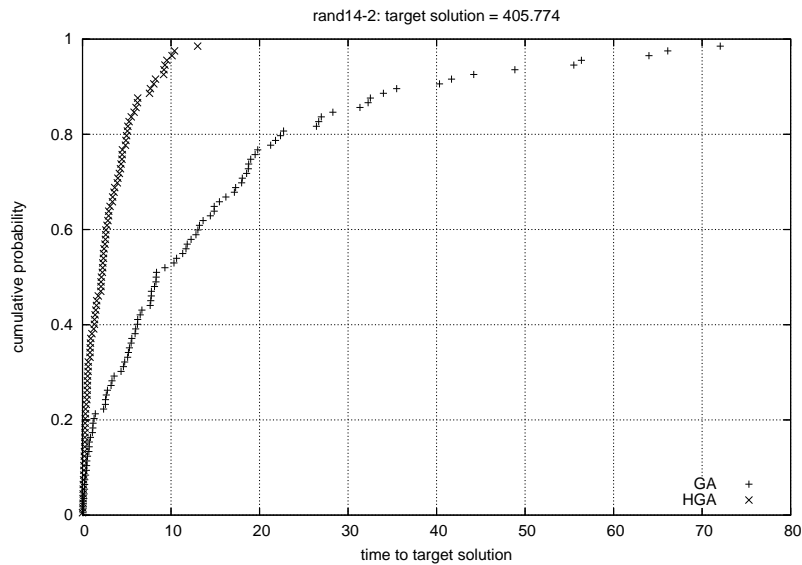


FIGURE 7. Time-to-Target plot for instance rand14-2. As above, the target value is the optimal solution.

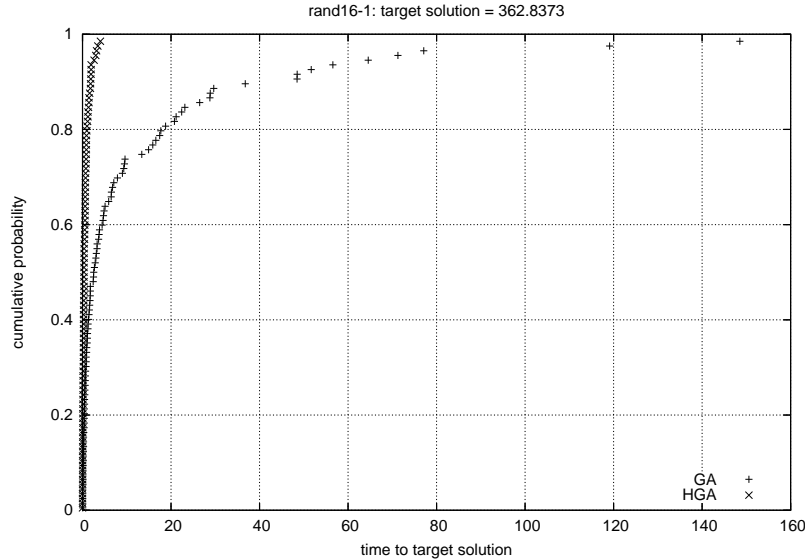


FIGURE 8. Time-to-Target plot for instances rand16-1. The target value is .95 times the optimal solution.

5. CONCLUSION

In this paper, we studied the so-called TARGET VISITATION PROBLEM whose objective is to plan the sequence for an unmanned aerial vehicle to visit a set of targets which minimizes the total distance traveled and maximizes the utility of the sequence. Until now, the literature on this problem has been slight [26]. This paper presents the first extensive computational analysis for the problem. First we provided a mathematical model for the TVP based on integer linear programming and proved that finding an optimal solution is \mathcal{NP} -complete. To overcome the computational complexity, we described the implementation of a random keys based genetic algorithm for finding near optimal solutions [5]. The heuristic was then hybridized by the implementation of a local search procedure. The numerical results presented demonstrated the effectiveness of the proposed procedure. Out of 6250 experiments, the hybrid heuristic calculated optimal solutions for over 99.9% of the trials in a fraction of the time required by the commercial integer programming solver CPLEX.

Since the TVP is a relatively new problem in the literature, there are several directions for future research. Clearly other metaheuristics can be implemented and compared with GA approach. However, due to the computational complexity, advanced cutting plane techniques such as branch and cut or branch and cut and price should most like be a focus in order to determine optimal solutions for large-scale instances of the problem. Extensions to the model proposed are also possible such as imposing a constraint on the total distance traveled and generalizing the model to include multiple vehicles. Visiting mobile targets would present other interesting challenges beyond the model presented in this paper.

6. ACKNOWLEDGEMENTS

We gratefully acknowledge the Air Force Office of Scientific Research for providing partial funding for this project. Finally, we are thankful to Dr. Mauricio G.C. Resende of AT&T Labs Research for making the time-to-target plotting software `tttplots` publicly available [3]. Finally, we are grateful to the associate editor and two anonymous referees whose insightful comments and suggestions greatly improved the quality of this paper.

REFERENCES

- [1] E. Aarts and J.K. Lenstra, editors. *Local Search in Combinatorial Optimization*. Wiley, 1997.
- [2] R.M. Aiex, M.G.C. Resende, and C.C. Ribeiro. Probability distribution of solution time in GRASP: An experimental investigation. *Journal of Heuristics*, 8:343–373, 2002.
- [3] R.M. Aiex, M.G.C. Resende, and C.C. Ribeiro. TTTPLOTS: A perl program to create time-to-target plots. *Optimization Letters*, published online, DOI: 10.1007/s11590-006-0031-4, 2006.
- [4] R. Battiti and G. Tecchiolli. Parallel biased search for combinatorial optimization: Genetic algorithms and TABU. *Microprocessors and Microsystems*, 16:351–367, 1992.
- [5] J.C. Bean. Genetic algorithms and random keys for sequencing and optimization. *ORSA Journal on Computing*, 6(2):154–160, 1994.
- [6] L.S. Buriol, M.G.C. Resende, C.C. Ribeiro, and M. Thorup. A hybrid genetic algorithm for the weight setting problem in OSPF/IS-IS routing. *Networks*, 46:36–56, 2005.
- [7] S.I. Butenko, R.A. Murphey, and P.M. Pardalos, editors. *Cooperative Control: Models, Applications, and Algorithms*. Springer, 2003.
- [8] S.I. Butenko, R.A. Murphey, and P.M. Pardalos, editors. *Recent Developments in Cooperative Control and Optimization*. Springer, 2004.
- [9] S. Chanas and P. Kobylanski. A new heuristic algorithm solving the linear ordering problem. *Computational Optimization and Applications*, 6:191–206, 1996.
- [10] B.H. Chiarini, W. Chaovalitwongse, and P.M. Pardalos. A new algorithm for the triangulation of input-output tables in eEconomics. In P. Pardalos, A. Migdalas, and G. Baourakis, editors, *Supply Chain and Finance*. World Scientific, 2004.
- [11] N. Christofides. Worst-case analysis of a new heuristic for the traveling salesman problem. Technical report, Carnegie Mellon University, 1976.
- [12] ILOG CPLEX. <http://www.ilog.com/products/cplex>, Accessed October 2006.
- [13] G. Dantzig, R. Fulkerson, and S. Johnson. Solution of a large-scale traveling-salesman problem. *Operations Research*, 2, 1954.
- [14] C. Darwin. *The Origin of Species*. Murray, sixth edition, 1872.
- [15] M. Desrochers and G. Laporte. Improvements and extensions to the miller-tucker-zemlin subtour elimination constraints. *Operations Research Letters*, 10:27–36, 1991.
- [16] P. Festa, P.M. Pardalos, L. Pitsoulis, and M.G.C. Resende. GRASP with path-relinking for the weighted MAXSAT problem. *ACM Journal of Experimental Algorithmics*, accepted, 2006.
- [17] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, 1979.
- [18] F. Glover. Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research*, 5:533–549, 1986.
- [19] F. Glover, T. Klasterin, and D. Klingman. Optimal weighted ancestry relationships. *Management Science*, 20:B1190–B1193, 1974.
- [20] D.E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Kluwer Academic Publishers, 1989.
- [21] J.F. Gonçalves and J.R. Almeida. A hybrid genetic algorithm for assembly line balancing. *Journal of Heuristics*, 8:629–642, 2002.
- [22] J.F. Gonçalves, J.J.M. Mendes, and M.G.C. Resende. A hybrid genetic algorithm for the job shop scheduling problem. *European Journal of Operations Research*, 167:77–95, 2005.
- [23] J.F. Gonçalves, J.J.M. Mendes, and M.G.C. Resende. A random keys based genetic algorithm for the resource constrained project scheduling problem. *Computers and Operations Research*, accepted, 2006.
- [24] M. Grötschel, M. Jünger, and G. Reinelt. A cutting plane algorithm for the linear ordering problem. *Operations Research*, 32:1195–1220, 1984.
- [25] M. Grötschel, M. Jünger, and G. Reinelt. On the acyclic subgraph polytope. *Mathematical Programming*, 33:28–42, 1985.
- [26] D.A. Grundel and D.E. Jeffcoat. Formulation and solution of the target visitation problem. In *Proceedings of the AIAA 1st Intelligent Systems Technical Conference*, 2004.
- [27] D.A. Grundel, R.A. Murphey, and P.M. Pardalos, editors. *Theory and Algorithms for Cooperative Systems*. World Scientific, 2004.
- [28] G. Gutin and A. Punnen. *The Traveling Salesman Problem and Its Variations*. Kluwer Academic Publishers, Dordrecht, 2002.
- [29] F.S. Hillier and G.J. Lieberman. *Introduction to Operations Research*. McGraw Hill, 2001.
- [30] R. Horst, P.M. Pardalos, and N.V. Thoai. *Introduction to Global Optimization*, volume 3 of *Nonconvex Optimization and its Applications*. Kluwer Academic Publishers, 1995.
- [31] J. King. Three problems in search of a measure. *American Mathematical Monthly*, 101:609–628, 1994.
- [32] A. Langevin, F. Soumis, and J. Desrosiers. Classification of traveling salesman problem formulations. *Operations Research Letters*, 9:127–132, 1990.

- [33] E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, and D.B. Shmoys, editors. *The Traveling Salesman Problem*. Wiley, 1985.
- [34] H.W. Lenstra. The acyclic subgraph problem. Technical Report Report BW26, Mathematisch Centrum, Amsterdam, 1973.
- [35] C. Miller, R. Tucker, and R. Zemlin. Integer programming formulations and traveling salesman problems. *Journal of the ACM*, 7:326–329, 1960.
- [36] R.A. Murphey and P.M. Pardalos, editors. *Cooperative Control and Optimization*. Springer, 2002.
- [37] C.A.S. Oliveira, P.M. Pardalos, and T.M. Querido. Integer formulations for the message scheduling problem on controller area networks. In D. Grundel, R. Murphey, and P. Pardalos, editors, *Theory and Algorithms for Cooperative Systems*, pages 353–365. World Scientific, 2004.
- [38] C.A.S. Oliveira, P.M. Pardalos, and T.M. Querido. A combinatorial algorithm for message scheduling on controller area networks. *Int. Journal of Operations Res.*, 1(1/2):160–171, 2005.
- [39] M. Padberg and T. Sung. An analytical comparison of different formulations of the traveling salesman problem. *Mathematical Programming*, 52:315–357, 1991.
- [40] C.H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.
- [41] C.H. Papadimitriou and S. Vempala. On the approximability of the traveling salesman problem. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computation*, 2000.
- [42] V. Pareto. *Cours d'économie politique*. Geneva: Librairie Droz, 1964.
- [43] M.G.C. Resende and C.C. Ribeiro. Greedy randomized adaptive search procedures. In F. Glover and G. Kochenberger, editors, *Handbook of Metaheuristics*, pages 219–249. Kluwer Academic Publishers, 2003.
- [44] W.M. Spears and K.A. DeJong. On the virtues of parameterized uniform crossover. In *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 230–236, 1991.
- [45] L. Wolsey. *Integer Programming*. Wiley, 1998.

(A. ARULSELVAN) CENTER FOR APPLIED OPTIMIZATION, DEPT. OF INDUSTRIAL AND SYSTEMS ENGINEERING, UNIVERSITY OF FLORIDA, GAINESVILLE, FL, USA.

E-mail address: ashwin@ufl.edu

(C.W. COMMANDER) AIR FORCE RESEARCH LABORATORY, MUNITIONS DIRECTORATE, AND, DEPT. OF INDUSTRIAL AND SYSTEMS ENGINEERING, UNIVERSITY OF FLORIDA, GAINESVILLE, FL USA.

E-mail address: clayton.commander@eglin.af.mil

(P.M. PARDALOS) CENTER FOR APPLIED OPTIMIZATION, DEPT. OF INDUSTRIAL AND SYSTEMS ENGINEERING, UNIVERSITY OF FLORIDA, GAINESVILLE, FL, USA.

E-mail address: pardalos@ufl.edu