Optimization Letters manuscript No.
(will be inserted by the editor)

The Aircraft Routing Problem with Refueling

Tanya E. Kannon · Sarah G. Nurre · Brian J. Lunday · Raymond R. Hill

Received: date / Accepted: date

Abstract In this paper, we present, test, and compare two novel methods to solve the aircraft routing problem with aerial refueling with a multicriteria objective function. We present a mixed-integer linear program (MILP) that utilizes a combination of a network transformation and a formulation that creatively decouples refueling decisions from the nodes within the network. We also present a dynamic program (DP) that, when coupled with an alternative network transformation to account for the multiple criteria within the objective function, applies a node-labeling approach based on a modification of Dijkstra's algorithm. We test and compare these alternative solution methods on a set of 264 synthetically-generated instances representing 66 combinations of network size and the frequency of aerial refueling point availability. Invoking CPLEX using the C++ callable library to solve the MILP and applying the DP in C++, we found that the application of the DP yields a 98.97% reduction in the required computational effort, on average, relative to the MILP; the MILP fails to find an optimal solution within a 3600-second time limit for selected instances of networks having at least 80 nodes and for all instances of networks having at least 350 nodes. In contrast, the DP is more robust than the MILP, as it only requires longer than 3600 seconds to solve selected instances of networks having over 3000 nodes.

Keywords Aircraft routing · network routing · node splitting · mixed-integer linear programming · dynamic programming

Department of Operational Sciences (ENS)
Air Force Institute of Technology (AFIT)
Bldg. 641, 2950 Hobson Way
Wright Patterson AFB, OH 45433, USA
Tel.: +1-937-255-3636 (x4624)
Fax: +1-937-656-4943
E-mail: brian.lunday@afit.edu

1 Introduction

In this study, we consider the problem of determining the route for an aircraft to traverse from a predetermined starting location to a destination while ensuring enough fuel is on board to complete the trip, and where aerial refueling is possible at selected intermediary points, depending on the aircraft routing decision. Within this context, the problem seeks to determine the route that minimizes a weighted combination of the number of refueling operations required and the total distance of the route. (In lieu of distance, one may alternatively consider cost, time, fuel expenditure, or even the difficulty of the route.) We model this routing problem using a directed acyclic network, wherein nodes within the network represent spatial points at which routing decisions and/or aerial refueling operations may occur, and arcs represent the paths connecting the nodes in the network. Thus, we seek to route an aircraft from a designated start node to a designated terminus node through a series of intermediary nodes. For this problem, we assume that the aircraft begins with a full tank of fuel, and some subset of the intermediary nodes allows for aerial refueling via a tanker aircraft. Moreover, when visiting an aerial refueling node, an aircraft has the option whether to refuel; a decision must be made for each such node traversed. For the purpose of this study, if an aircraft elects to utilize a refueling node, we further assume that the fuel tank is filled. No partial refueling operations are considered, although such extensions may be readily examined from the formulations we consider herein.

We note that, if the length of the path is preemptively weighted over the number of refueling operations conducted and if a path exists through the network for which refueling operations are not necessary, our problem reduces to a shortest-path problem which may be solved via Dijkstra's shortest path algorithm (see, e.g., Ahuja et al. [1]). In the absence of such a preemptive weighting and depending upon the topology of the network and the parameters of its arcs, it may be preferable for an aircraft to refuel en route so that it can traverse a shorter path than the shortest path that does not require refueling. The possibility that one path may be shorter than another but require more fuel to traverse exists because the length of a flight path is not absolutely correlated with its fuel usage; factors specific to its flight legs (i.e., arcs within the network) that affect fuel usage include the airspeed of the aircraft, the wind speed and direction, and the altitude (i.e., air density) of the aircraft. However, rather than focus on the aforementioned possible preference, we consider herein only nontrivial problems for which at least one refueling operation is necessary to traverse the network from the start node to the terminus node.

The aircraft routing problem has been examined extensively within the commercial airline industry, but the majority of such research does not consider refueling requirements and, as such, is fundamentally different than the focus of the paper. For example, Barnhart et al. [3] considered a combined aircraft fleet assignment and routing problem which seeks to assign aircraft to flight legs and routes such that all flight legs are completed. Desaulniers et al. [10] examined a similar problem that sought to determine the daily aircraft

schedule and routes by specifically determining the time of the day for each flight leg and aircraft used. Biggs-Bartholomew et al. [4,5] examined an aircraft routing problem in planar two-dimensional Euclidean space using both global optimization and heuristics. Royset et al. [22] solved a routing problem specific to military aircraft, incorporating enemy threats, total risk, and fuel consumption. (See Crino et al. [9], Harder et al. [14], and O'Rourke et al. [20] for other noteworthy examples of research on optimal military aircraft routing.) Barnhart et al. [2] provided an extensive survey of the research relevant to these type of aircraft routing problems. However, this broad category of research does not address the problem of routing fuel-capacitated aircraft and the corresponding need to conduct aerial refueling operations en route between the source and terminus nodes of a network being traversed, and so we look elsewhere for modeling and solution techniques to inform our approach.

Several recent works have examined aircraft routing problems that incorporate refueling operations. Bush [8] considered the problem of simultaneously routing an aircraft, locating a fixed number of aerial refueling points to be serviced by refueling aircraft based at fixed locations, and determining the amount of fuel to transfer at each refueling point in order to minimize the total amount of fuel utilized by both the traversing aircraft and the refueling aircraft. The author developed and demonstrated a heuristic for constructing and improving upon a feasible routing and refueling solution. Erdoğan and Miller-Hooks [12] considered a variation of the vehicle routing problem for which they accounted for the refueling needs of alternative energy vehicles. The authors formulated and solved a mixed-integer program to determine a series of routes, wherein each route must be able to be completed by the vehicle before returning to the starting location to refuel, and from which the routes are conjoined for the vehicle to visit all customers. Sundar and Rathinam [24] examined the problem of determining the route for an Unmanned Aerial Vehicle (UAV) through a series of targets and refueling depots such that all targets are visited while seeking to minimize the total fuel expended. The authors solved the problem directly using a mixed-integer programming formulation, and they also developed and tested an approximation algorithm and an accompanying improvement heuristic.

The combination of the multicriteria objective function and the refueling requirement in our problem sets it apart from the shortest path problem (SPP) and most of its previously studied variants. However, research in these areas does inform our second solution method. There exist several efficient methods to solve SPP, including Dijkstra's Algorithm (e.g., see Ahuja et al. [1] or Bazaraa et al. [6]) and the Bellman-Ford Algorithm [7]. Several endeavors have examined extensions to the SPP with side constraints (e.g., see [11], [13], [17], and [21]), wherein an agent seeks to identify the shortest path through a network, wherein traversing an arc incurs a weight (or utilizes a resource) and the path cannot exceed a limit on the accumulation of a weight (or expenditure of a resource) at the terminus, and possibly at intermediate nodes within the network. In our problem, the aircraft's fuel is a limiting resource that can be replenished at any of a predetermined subset of the nodes. An

equivalence exists between our problem and the SPP with side constraints for two cases. If a feasible path exists between the start node and terminus node that does not require refueling and the path distance is preemptively weighted over the number of refueling operations, then our problem is equivalent. It is also equivalent in the case where every path between the start node and the terminus node contains at most one refueling node; such a structure parallels that examined by Irnich and Desaulniers [17] wherein they examine time as a resource and incorporate deadlines for transiting nodes that relate to our physical deadline of transiting refueling points before exhausting the supply. For any other circumstance, the agent has the ability to select among a set of refueling nodes when refueling is required, so the models and solution methods for the SPP with side constraints are not extensible. Closer to our problem, Smith et al. [23] extended the work of Dumitrescu and Boland [11] by considering instances in which no path exists that does not exceed the weight accumulation threshold, but in which any one of a subset of predetermined arcs allows an agent to reset the accumulated weight to zero by traversing it. The authors developed and tested an effective algorithm to solve this problem via meta-networks, but they sought only to minimize the length of the origin-destination path.

In a predecessor to this research that is closely related to the application examined by Smith et al. [23], Kannon et al. [18] examined the problem of routing a military aircraft from a starting node through a virtual network to a destination node, while considering a fuel limit for the aircraft and a subset of nodes at which the aircraft can replenish its fuel. Seeking strictly to minimize the length of the origin-destination-origin path, the authors showed this problem to be NP-hard, and they tested and compared a greedy heuristic with a node-labeling algorithm that optimally solves the problem. Although the work of Smith et al. [23] and Kannon et al. [18] differ slightly in their assumptions from our problem, they are sufficiently related to motivate our approach. Herein, we examine a multicriteria objective function that accounts for both the length of the path and the number of refueling operations conducted, each of which is of importance for aircraft routing, given the expense of aerial refueling in terms of aircraft, fuel, time, and cost. Whereas the MILP formulation we present in this paper is new, our DP node-labeling algorithm extends the method developed by Kannon et al. [18] for a multicriteria objective function via a network transformation. This network transformation enables a reduced number of node labels as we employ a dominance check.

Main Contributions: The main contributions of this paper are as follows: (i) development of a MILP formulation which decouples refueling decisions from nodes within the network; (ii) expansion of a dynamic programming algorithm to handle a multicriteria objective; (iii) proof of correctness that the DP is an optimal algorithm; (iv) robust computational testing on the two solution methods set forth on networks with various sizes and refuel availability; and (v) validation for the use of of the model and solution methods for the air refueling military application.

The remainder of this paper is organized as follows. In Section 2, we set forth a mixed-integer linear programming (MILP) formulation on an augmented network that utilizes a temporal modeling approach, as well as a dynamic programming (DP) node-labeling algorithm that utilizes an alternative network transformation to account for the multiple criteria within the objective function. We test and compare these models in Section 3 with respect to their required computational effort to solve a set of synthetically-generated instances that represent a breadth of network sizes and frequencies of aerial refueling point availability. In Section 4, we summarize the contributions of this research and propose directions for enhancing its utility in future endeavors.

2 Methodology

In this section, we formulate our MILP and present our DP to optimally route an aircraft from a starting node to a terminus node over a directed network, with the option to refuel at each of a subset of nodes. Consider the following definitions common to both approaches:

Problem Statement: We seek to route an aircraft with a fixed fuel capacity through a network of arc from a designated starting point to a designated ending point, wherein a subset of the intermediary nodes (i.e., locations at which two or more arcs connect) are designated as points at which the aircraft may conduct aerial refueling, and with the objective of minimizing a weighted combination of the total distance traveled and the number of aerial refueling operations conducted.

Sets:

- $i \in N$: set of nodes in the network, with a designated start node, s, and terminus node, t.
  - $i \in N_F$: the set of nodes in the network at which an aircraft *may* refuel.
  - $i \in N_{NF}$: the set of nodes in the network at which an aircraft *may not* refuel.
  - Note: we assign the terminus node $t \in N_{NF}$, where $N = N_F \cup N_{NF} \cup \{s\}$, with the caveat that the aircraft is assumed to depart the start node with a full tank of fuel.
- $(i, j) \in A$: set of directed arcs in the network. For the purpose of this study, we assume the network contains no cycles.
- $G[N, A]$: the underlying network.

Parameters:

- c_{ij}: the cost (in distance) to traverse arc (i, j).
- f_{ij}: the fuel utilized by the aircraft traversing arc (i, j).
- F: the maximum fuel capacity of the aircraft.
- w_1: the nonnegative weight placed upon the objective of minimizing the total distance traveled by the aircraft.

– w_2: the nonnegative weight placed upon the objective of minimizing the number of refueling operations by the aircraft.

2.1 Mixed-Integer Linear Programming Formulation

In this subsection, we apply a network transformation to enable an MILP formulation. Given the underlying network $G[N, A]$, we split the nodes $i \in N_F$ such that we have one set of nodes N_{F_1} denoting aerial refueling locations at which the aircraft *does* refuel, and a second corresponding set of nodes N_{F_2} denoting the same refueling locations where the aircraft *does not* refuel. The resulting augmented set of nodes is denoted N^*, where $N^* = N_{F_1} \cup N_{F_2} \cup N_{NF} \cup \{s\}$. Arcs $(i, j) \in A$ that either enter or emanate from nodes $i \in N_F$ are also duplicated accordingly, resulting in an augmented set of arcs, denoted A^*. The upper bound on the increase in network size for the resulting transformation is a two-fold increase in the number of nodes and a four-fold increase in the number of arcs.

Define the set $\tau \in T$ to be the sequence of routing decisions executed by the aircraft to traverse from the start node s to the terminus node t. Specifically, we use this set to signify which arc an aircraft is traversing at each point in time. Because we assume the network has no cycles, we may set $|T| = |N| - 1$ as the upper bound on the number of nodes visited by the aircraft upon departing node s. Furthermore, we define the decision variable $x_{ij}^{\tau} = 1$ if the aircraft traverses arc $(i, j) \in A^*$ for decision $\tau \in T$, and 0 otherwise. This enables us to define the decision variable F^{τ} to be the amount of fuel in the aircraft when making decision $\tau \in T$. Within this framework, we propose the formulation for Model **MILP** as follows:

$$\textbf{MILP: } \min \left(w_1 \sum_{(i,j)\in A^*} \sum_{\tau \in T} c_{ij} x_{ij}^{\tau} + w_2 \sum_{\substack{(i,j)\in A^*: \\ j:j\in N_{F_1}}} \sum_{\tau \in T} x_{ij}^{\tau} \right) \tag{1}$$

$$\text{subject to } \sum_{j:(s,j)\in A^*} x_{sj}^{\tau} = \left\{ \begin{matrix} 1 & \text{if } \tau = 1 \\ 0 & \forall\ \tau \in T\backslash\{1\} \end{matrix} \right\}, \tag{2}$$

$$\sum_{j:(i,j)\in A^*} x_{ij}^{\tau+1} - \sum_{j:(j,i)\in A^*} x_{ji}^{\tau} = 0,\ \forall\ i \in N^*\backslash\{s,t\},\ \tau \in T, \tag{3}$$

$$\sum_{i:(i,t)\in A^*} \sum_{\tau \in T} x_{it}^{\tau} = 1, \tag{4}$$

$$F^1 = F, \tag{5}$$

$$F^{\tau} \leq F^{\tau-1} - \sum_{\substack{(i,j)\in A^*: \\ j:j\in N_{NF}\cup N_{F_2}}} f_{ij} x_{ij}^{\tau-1} + \sum_{\substack{(i,j)\in A^*: \\ j:j\in N_{F_1}}} F x_{ij}^{\tau-1},\ \forall\ \tau = T\backslash\{1\}, \tag{6}$$

$$F^\tau \leq F,\ \forall\ \tau \in T\backslash\{1\}, \tag{7}$$

$$F^\tau - \sum_{(i,j)\in A^*} f_{ij} x_{ij}^\tau \geq 0,\ \forall\ \tau \in T, \tag{8}$$

$$x_{ij}^\tau \in \{0,1\},\ \forall\ (i,j) \in A^*,\ \tau \in T. \tag{9}$$

The objective function (1) calculates a weighted combination of the distance traveled and the number of aerial refueling operations, with the latter component conditioned on decisions to traverse arcs into aerial refueling nodes. With regard to flow balance, Constraint (2) enforces that the aircraft departs node s at $\tau = 1$, Constraint (3) enforces the conservation of flow at all intermediary nodes without allowing loitering and/or delays at any node, and Constraint (4) requires that the aircraft arrive at node t at some point within $\tau \in T$, the set of decisions. The initial fuel level is set via Constraint (5), and Constraint (6) bounds the amount of fuel based on a combination of the routing decision at each stage and whether the aircraft travels to an aerial refueling node at stage τ. Constraint (7) bounds the aircraft fuel level by its capacity, Constraint (8) prevents a decision that would cause the aircraft to run out of fuel between nodes, and Constraint (9) enforces the binary integer restriction on the x_{ij}^τ-variables.

2.2 Node Labeling Algorithm

In this subsection, we present a node-labeling dynamic programming algorithm based on Dijkstra's algorithm [1] with additional label values based on the A* algorithm [15] and those necessary for the tracking of fuel, as well as a network transformation that allows the algorithm to solve our problem. Given the underlying network $G[N, A]$, we again split the nodes $i \in N_F$ as stipulated in Section 2.1. We replace the arc costs $c_{ij} \leftarrow \hat{c}_{ij},\ \forall\ (i,j) \in A^*$, where for arcs entering refueling nodes (i.e., $(i,j) \in A^* : j \in N_{F_1}$), let $\hat{c}_{ij} = w_1 c_{ij} + w_2$ and, for all other arcs, let $\hat{c}_{ij} = w_1 c_{ij}$. This converts our multicriteria objective function into a network representation for a single objective function, wherein traversing an arc incurs a weighted sum of the distance traveled and an indicator variable corresponding to whether traversing that arc entails a refueling operation upon arrival, and which we henceforth refer to as the *weighted cost*. What remains is to identify the least s-t weighted cost path through the network, with refueling as necessary enroute to ensure the aircraft does not run out of fuel.

If applied to solve the SPP for our network $G[N^*, A^*]$, Dijkstra's algorithm initializes the nodes with distance labels of 0 for s and ∞ for $N^*\backslash\{s\}$, and initializes a set of permanently labeled nodes initialized as $\{s\}$. The algorithm iteratively considers the set of permanently labeled nodes, creates a label for each node that can be directly reached from that set if the new label is less than the existing distance label, and assigns the node with the shortest overall distance to the set of permanently labeled nodes. This process continues until

all nodes and their corresponding distance labels are made permanent. The A* algorithm adds an additional label to each node with a heuristic value for the approximate distance to the terminus node, enabling a reduction in the required computational effort.

As developed by Kannon et al. [18] and modified for the network topology examined herein, we extend the idea of applying labels to nodes, albeit with a label containing more information than either Dijkstra's algorithm or the A* algorithm. Kannon et al. [18], demonstrates the potential for an exponential number of node labels, which we seek to decrease based on dominance criteria. Denote the label ℓ associated with a node to be comprised of: (i) the weighted cost from the start node $wc(\ell)$; (ii) the estimated weighted cost to the terminus node t, $e(\ell)$; (iii) the fuel level upon arrival $f(\ell)$, where $f(\ell) = F$ if $\ell \in N_{F_1}$; (iv) the predecessor node $p(\ell)$; (v) the label number of the predecessor $p\ell(\ell)$; and (vi) the label number for this node $n(\ell)$, where (v) and (vi) are necessary because each node can have multiple labels. The least weighted cost path from each node to the terminus in the absence of fuel requirements is used for $e(\ell)$ as this is a lower bound (i.e., a monotone admissible heuristic function) on the weighted cost path when fuel is considered.

At the start of the algorithm, the shortest distance path from each node $i \in N$ to the ending location is calculated, which we denote SP_i. Using this information, only the starting node i is given a label with values $\ell = (0, SP_i, F, -, -, 1)$, which assumes the aircraft starts with a full tank of fuel. The algorithm then sets the current node, denoted $c\ell$, to the starting location and makes the only label (ℓ which is associated with the starting node) to be the current permanent label L. The algorithm proceeds by examining all neighbors of the current node (i.e., $i : (c\ell, i) \in A^*$) and, if $f(L) - f_{c\ell i} \geq 0$ (i.e., the aircraft will not run out of fuel when traversing the arc), assigns labels in the following manner. For a neighboring node $i \in N_{NF} \cup N_{F_2} \cup \{t\}$, we add the label

$$(wc(L) + \hat{c}_{c\ell i}, SP_i, f(L) - f_{c\ell i}, c\ell, p\ell(L), k + 1), \tag{10}$$

where the weighted cost is calculated as the weighted cost to travel from $c\ell$ to i, the estimated distance to the ending location is exactly the shortest distance path without regard to fuel; the fuel is calculated based on the current fuel level $f(L)$ minus the fuel it takes to traverse $(c\ell, i)$; the predecessor is set to the current location $c\ell$; the label of the predecessor is calculated based on the label number of L; and the label number for node i is incremented, where k, initialized at 0 is the current number of labels associated with node i. For a neighboring node $i \in N_{F_1}$, the following label is added, with only the third entry differing from Equation (10) to account for the refueling operation upon arrival

$$(wc(L) + \hat{c}_{c\ell i}, SP_i, F, c\ell, p\ell(L), k + 1). \tag{11}$$

We also invoke the following label dominance criteria as a component of our node-labeling algorithm. First, consider a node $i \in N_{F_1}$, at which an aircraft refuels upon arrival. If a node label ℓ is added to node $i \in N_{F_1}$, and it has a weighted cost $wc(\ell)$ strictly less than a label $\bar{\ell}$ already associated with node i,

we can conclude that $\bar{\ell}$ will not lead to an optimal solution; ℓ is dominated by $\bar{\ell}$. The same can be said in the reverse direction; a label ℓ should be not added to a node $i \in N_{F_1}$ if there exists a (dominating) label $\bar{\ell}$ already associated with i having a lesser weighted cost. Alternatively, consider a node $i \notin N_{F_1}$, i.e., a node at which an aircraft does not refuel. The previous domination criterion based on weighted cost alone does not apply because the fuel levels $f(\ell)$ and $f(\bar{\ell})$ may differ; a label that dominates another with respect to the weighted cost alone may not have the required fuel necessary to complete a path to the terminus node t, whereas a label with a greater weighted cost may have sufficient fuel. Therefore, without loss of generality, we identify a label ℓ associated with node $i \notin N_{F_1}$ as being dominated by a label $\bar{\ell}$ associated with node i if both $wc(\ell) < wc(\bar{\ell})$ and $f(\ell) \geq f(\bar{\ell})$. These criteria collectively cause the node-labeling algorithm to disregard labels with a greater weighted cost and the same or a lesser amount of fuel.

Emulating Dijkstra's algorithm with A^* algorithmic enhancements, we make permanent the *label* – instead of the node – having the current smallest sum of the weighted cost plus the estimate cost value. The current node is then set to the node for which this label is associated, and the algorithm continues by examining the neighbors of this node. When a label associated with the terminus is made permanent, the algorithm terminates. The route traversed by the agent can be determined via post-processing by tracking back through the predecessor labels. We now prove the correctness of this optimal algorithm.

Theorem 1 *The DP algorithm for the aircraft routing problem with refueling is an optimal algorithm.*

Proof The algorithm terminates when a label ℓ associated with the terminus node is made permanent, which is claimed to correspond to the optimal minimum cost path while adhering to fuel and replenishment constraints, and for which the s-t path can be determined by tracing back using the predecessor node and predecessor label number. Denote this path p, with the label ℓ made permanent by the algorithm because it has the smallest value for $wc(\ell) + e(\ell)$, where $e(\ell) = 0$ because ℓ is associated with the terminus node t, signifying that the cost of p is exactly $wc(\ell)$.

We make the following observation to aid with the proof: all feasible fuel paths can be attained via an expansion of node labels. That is, there does not exist a feasible fuel path which cannot be found by continuing the node-labeling technique described via the DP where, in the worst case, the DP enumerates all s-t paths through an acyclic network.

Assume that p is not an optimal path. Therefore, there exists a different path $\bar{p}$ having a lower weighted cost that adheres to the fuel constraints and refueling possibilities. Consider the existing labels in the stage of the algorithm at which the label ℓ is made permanent. The optimal path $\bar{p}$ to the terminus node must emanate from a different label than ℓ, either a non-permanent or a permanent label. However, all non-permanent labels in the network have a value for $wc(\ell) + e(\ell)$ (i.e., the weighted cost plus estimated cost, the latter of which is a lower bound on the actual cost) at least as large as $wc(\ell)$. Hence, no

non-permanent label can be associated with $\bar{p}$. Moreover, all permanent labels were examined at this stage, resulting in additional labels to neighboring nodes if applicable. Therefore, if there exists a permanent label $\bar{\ell}$ associated with $\bar{p}$, then there exists at least one non-permanent label also associated with $\bar{p}$, which we have shown does not exist. Therefore, $\bar{p}$ does not exist. □

3 Computational Testing and Evaluation

In this section, we develop a battery of test instances of varying network size and the frequency of aerial refueling point availability, and we examine the relative performance of our solution methods with regard to their required computational effort and optimality gap.

To test the performance of the two solution methods, we randomly generated single-source, single-sink fully connected acyclic networks as follows. Given a user-defined network size of $|N|$ nodes, denoted $\{s, 1, 2, ..., |N| - 3, |N| - 2, t\}$, a directed arc (i, j) initially exists if $i < j$, if $i = s$, or if $j = t$. This network topology results in every intermediary node lying on an s-t path and prevents the generation of leaf nodes (or even branches) that would serve no purpose for an aircraft traversing a network. For each instance, we utilized $F = 26000$ based on the fuel capacity (measured in pounds) of an F22 Raptor with two external fuel tanks (see Lockheed Martin Inc. [19] for selected aircraft specifications), and we randomly generated integer-valued f_{ij}-parameters for each arc (i, j) using the discrete uniform distributions shown in Table 1. For the purpose of f_{ij}-parameter generation, we number node s as 0, and we number node t as $|N|-1$. As a baseline, we selected $U[5400, 6000]$ when $(j-i) = 1$ for the fuel usage over an arc (i.e., a single flight leg), and the successive distributions in Table 1 ensure that the triangle inequality holds, e.g., such that it requires less fuel to fly directly from node i to node $i + 2$ than by flying through node $i+1$ enroute. In general, the distribution for $(j-i) = k$ is generated on $U[l_k, u_k]$, where $u_k = l_{k-1} + l_1$ and where l_k is some percentage of u_k. For our instance parameter generation, as represented in Table 1, we utilized $l_k = 0.9u_k$. We completed the network topology for our instances by removing arcs when $f_{ij} > F$ (which necessarily occurs for $(j-i) > 7$ in accordance with Table 1), as an aircraft would run out of fuel if it attempted to traverse such an arc.

Table 1 Distributions utilized for f_{ij}-values

$(j-i)$	**Distribution**
1	$U[5400, 6000]$
2	$U[9720, 10800]$
3	$U[13608, 15120]$
4	$U[17107, 19008]$
5	$U[20256, 22507]$
6	$U[23091, 25656]$
7	$U[25642, 28491]$

Once the network instance topology was finalized, we generated c_{ij}-parameters to be $c_{ij} = U[0.95, 1.05]\psi f_{ij},\ \forall\ (i,j) \in A$, with $\psi = \frac{1600}{18000}$. Herein, the value for ψ converts the fuel consumption required to traverse an arc into a distance measurement (in nautical miles) based on the ratio of the unclassified estimate of the F22's range to its internally-stored fuel capacity [19], whereas the discrete uniform distribution $U[0.95, 1.05]$ imposes a random element to reflect a variance in flight profiles (i.e., speed and altitude) between arcs. Finally, we select $w_1 = 1$ and $w_2 = 100$, where w_2 approximates the opportunity cost in the distance not traveled (in nautical miles) due to the conduct of an aerial refueling operation, thereby converting all components of our objective function to common units. To conclude our instance generation, given a user-defined parameter m that indicates the lexicographic frequency of aerial refueling point availability, we designated each intermediary node $i \in N$ as optional for aerial refueling (i.e., $i \in N_F$) if $i \pmod m = 0$, and $i \in N_{NF}$ otherwise. Figure 1 illustrates an instance of the resulting network topology for $(|N|, m) = (10, 2)$, with the (c_{ij}, f_{ij})-parameters shown for each arc and with each of the nodes in N_F encircled twice.

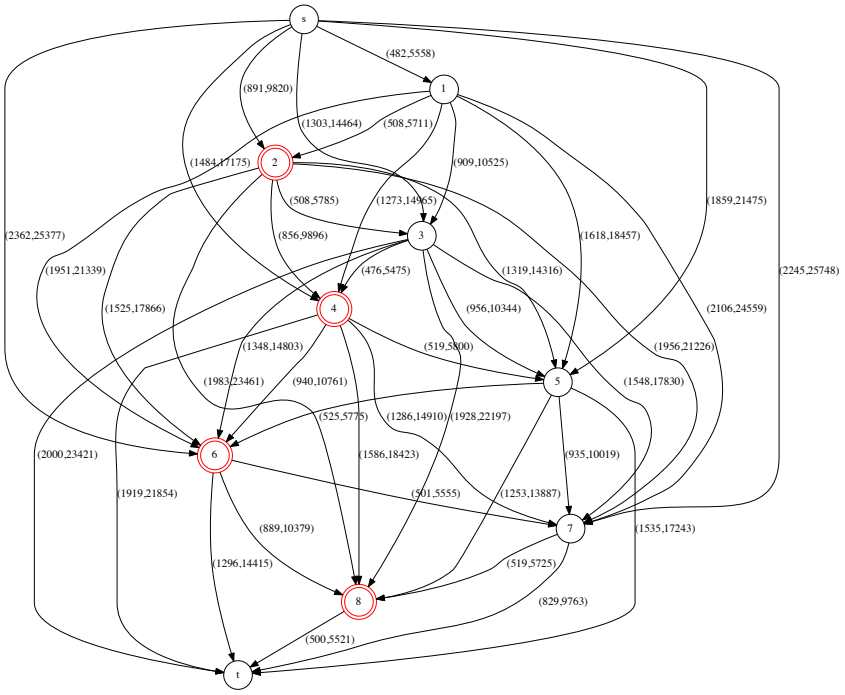

Fig. 1 Example instance with $(|N|, m) = (10, 2)$

As a block design for testing, we generate a total of 264 instances, four instances each of 66 combinations of variations in the two user-defined parameters: network size, $|N| = 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, 120, 130, 140, 150, 200, 250, 300, 350, 400, 450, 500$; and the lexicographic frequency of aerial refueling point availability, $m = 2, 3, 4$. Although it is possible to examine instances with $m \geq 5$, such a case may yield an infeasible problem wherein the minimum fuel required between two optional refueling nodes exceeds F, and so we restrict our attention in this study to problems we know to be feasible, thereby isolating our testing to examine the relative utility of the solution methods. For each of our 264 test instances, we solved Model MILP using CPLEX (Version 12.5) invoked through the C++ callable library and via our DP algorithm using C++. All runs were implemented on a computer having an Intel Core i5 Processor (3.1 GHz speed) and 12 GB of RAM. We invoked CPLEX with a relative optimality tolerance of $\epsilon = 0.01$, and we terminated either solution method if an optimal solution was not reported within 3600 seconds (i.e., one hour).

To compare the respective efficacy of the solution methods, we capture for each $(|N|, m)$-combination the average required computational effort and, for the MILP, the average relative optimality gap (%) upon termination. We calculate the average optimality gap attained by the MILP using two methods. The first method is denoted as the *attained* optimality gap, computed as the absolute gap between the lowest feasible upper bound and the greatest lower bound returned by CPLEX, divided by the greatest lower bound returned by CPLEX.[1] The second method is the *actual* optimality gap, computed as the absolute gap between the lowest feasible upper bound returned by CPLEX and the optimal objective function value attained by the DP, divided by the optimal objective function value attained by the DP. Whenever the attained and actual optimality gaps are close, the lower bound identified by CPLEX is tight.

Table 2 reports these assessment measures for both the MILP and DP. The first two columns tabulate the number of nodes $|N|$ in the network and the frequency-of-aerial-refueling frequency parameter m. The following three columns represent the Attained Optimality Gap (%), the Actual Optimality Gap (%), and the required computational effort for the MILP. The last column represents the computational effort required for the DP solution method. An individual cell represents the average of the measures for the four instances for a given $(|N|, m)$-combination. For an instance for which the MILP terminates without attaining an optimal solution within the specified time limit, we use 3600 seconds in the calculation of the respective average required computational efforts. For many of the larger networks, CPLEX was unable to identify a feasible solution within the one hour time limit; we identify these results in Table 2 by $^{\dagger i}$ for the number i out of 4 instances for which this occurred.

[1] We use the attained optimality gap rather than the relative optimality gap reported directly by CPLEX, as the commercial solver computes its reported gap by dividing the the absolute optimality gap by the magnitude of the best integer-feasible objective function value [16] (i.e., the lowest feasible upper bound).

Table 2 Average Required Computational Effort and Optimality Gap for Model and $(|N|, m)$-combinations. $^{\dagger i}$ indicates that i out of 4 instances did not find a feasible solution within the 3600 second time limit.

		MILP			DP
		Attained	Actual		
$\|N\|$	m	Opt. Gap (%)	Opt. Gap (%)	Time (s)	Time (s)
10	2	0.00%	0.00%	0.023	0.001
	3	0.00%	0.00%	0.017	0.001
	4	0.00%	0.00%	0.020	0.001
20	2	0.00%	0.00%	0.138	0.005
	3	0.00%	0.00%	0.122	0.004
	4	0.00%	0.00%	0.153	0.004
30	2	0.00%	0.00%	0.532	0.024
	3	0.00%	0.00%	0.365	0.014
	4	0.00%	0.00%	0.440	0.012
40	2	0.00%	0.00%	1.335	0.040
	3	0.00%	0.00%	1.044	0.028
	4	0.00%	0.00%	3.473	0.022
50	2	0.00%	0.00%	2.883	0.083
	3	0.00%	0.00%	2.276	0.049
	4	0.00%	0.00%	25.548	0.035
60	2	0.00%	0.00%	4.743	0.118
	3	0.00%	0.00%	4.904	0.077
	4	0.00%	0.00%	217.365	0.052
70	2	0.00%	0.00%	8.589	0.189
	3	0.00%	0.00%	6.140	0.102
	4	0.00%	0.00%	758.113	0.073
80	2	0.00%	0.00%	13.590	0.258
	3	0.00%	0.00%	10.519	0.134
	4	2.43%	0.00%	3600.000	0.102
90	2	0.00%	0.00%	24.238	0.323
	3	0.00%	0.00%	16.583	0.186
	4	2.40%	0.01%	3600.000	0.131
100	2	0.00%	0.00%	36.845	0.432
	3	0.00%	0.00%	31.796	0.226
	4	3.88%	0.14%	3600.000	0.180
110	2	0.00%	0.00%	58.333	0.531
	3	0.00%	0.00%	37.522	0.284
	4	3.36%	0.04%	3600.000	0.206
120	2	0.00%	0.00%	81.739	0.619
	3	0.00%	0.00%	69.584	0.354
	4	5.36%	0.25%	3600.000	0.249
130	2	0.00%	0.00%	119.983	0.767
	3	0.00%	0.00%	77.917	0.402
	4	5.29%	0.28%	3600.000	0.301
140	2	0.00%	0.00%	144.112	0.891
	3	0.00%	0.00%	115.356	0.501
	4	6.15%	0.44%	3600.000	0.352
150	2	0.00%	0.00%	378.677	1.078
	3	0.00%	0.00%	236.538	0.569
	4	6.68%	0.82%	3600.000	0.416
200	2	0.00%	0.00%	1054.702	2.047
	3	0.00%	0.00%	1413.537	1.151
	4	10.49%$^{\dagger 1}$	2.72%$^{\dagger 1}$	3600.000	0.870
250	2	0.57%	0.11%	3125.260	3.448
	3	0.07%	0.00%	2660.795	1.836
	4	32.75%	23.27%	3600.000	1.594
300	2	38.18%	36.15%	3600.000	5.591
	3	1.39%$^{\dagger 2}$	0.00%$^{\dagger 2}$	3600.000	2.935
	4	-$^{\dagger 4}$	-$^{\dagger 4}$	3600.000	2.300
350	2	52.51%	48.19%	3600.000	7.695
	3	45.17%	41.86%	3600.000	4.247
	4	40.44%	24.24%	3600.000	3.157
400	2	55.02%	48.32%	3600.000	12.449
	3	-$^{\dagger 4}$	-$^{\dagger 4}$	3600.000	6.091
	4	-$^{\dagger 4}$	-$^{\dagger 4}$	3600.000	4.391
450	2	56.17%	49.06%	3600.000	14.312
	3	-$^{\dagger 4}$	-$^{\dagger 4}$	3600.000	10.298
	4	46.64%	24.47%	3600.000	8.009
500	2	56.27%	48.84%	3600.000	18.952
	3	51.13%	43.89%	3600.000	10.466
	4	-$^{\dagger 4}$	-$^{\dagger 4}$	3600.000	8.760

Upon examination of the results reported in Table 2, immediately visible is the scalability difference between the solution methods as the number of nodes increases. The DP scales extremely well; it has an average elapsed time below 15 seconds for all but one of the $(|N|, m)$-combinations. The slowest performance observed for an instance by the DP was 19.449 seconds. In contrast, the MILP does not scale well; CPLEX terminates due to the limit on computational effort for all instances having $|N| \geq 350$, with the time limit being reached for test instances starting at the moderately sized networks having $|N| = 80$ nodes. Further, CPLEX is unable to find a feasible solution for 23 of the test instances within the one hour time limit, with the smallest such instance having $|N| = 200$ nodes. From this we can conclude that using the MILP – even as a heuristic – is not a viable option for the application at hand for larger networks. Moreover, the larger-sized networks for which the MILP does attain solutions yields large optimality gaps at termination. Given the attained and actual optimality gaps are close for such instances, future efforts should focus on the determination of high quality *feasible* solutions rather than improved relaxations if the MILP is to be a viable solution method.

Also identifiable is the trend that the DP requires more computational effort, on average, when aerial refueling locations are more plentiful (i.e., with greater frequency corresponding to a lower m value) for a given number of nodes in the network, $|N|$. This trend is directly impacted by the network transformation we utilize, wherein the augmented graph has an almost 50% increase in the number of nodes when $m = 2$, compared to a nearly 25% increase when $m = 4$. For example, an original 200 node network is augmented to 299-, 266-, and 249-node networks for $m = 2, 3, 4$, respectively. An interesting trend is also present when examining the computational times for the MILP. For more than half of the $|N|$ tests the average required computational effort is lowest when $m = 3$, highest for $m = 4$, with $m = 2$ in the middle. While the authors do not have a definitive explanation for this trend, we speculate that CPLEX performs best when there are a moderate number of refueling options ($m = 3$) and struggles when there are too many ($m = 2$) or too few ($m = 4$), corresponding to a larger size of the formulation and a greater challenge in finding a feasible solution, respectively.

Because it has been observed that the DP scales well as the number of nodes $|N|$ increases, we performed a secondary set of experiments to test the DP on much larger networks. The purpose of these experiments is to roughly determine the maximum network size (in terms of nodes) such that the DP can be used as a real-time decision making tool. Maintaining the same experimental design, we generate four test instances each for networks with $|N| = 550, 600, 650, 700, 750, 800, 850, 900, 950, 1000, 1250, 1500, 1750, 2000, 2250, 2500, 2750, 3000, 3250, 3500, 3750, 4000$ and $m = 2, 3, 4$. For these tests, no time limit is set, and the DP is run until the optimal solution is attained and the elapsed computational time is recorded. Results for this set of experiments are presented in Figure 2. Observable in Figure 2 is that the computational effort required for the DP to attain an optimal solution first exceeds 3600 seconds when $|N| = 3000$ for $m = 2$, and when $|N| = 3750$ for $m = 3, 4$. It is rare to

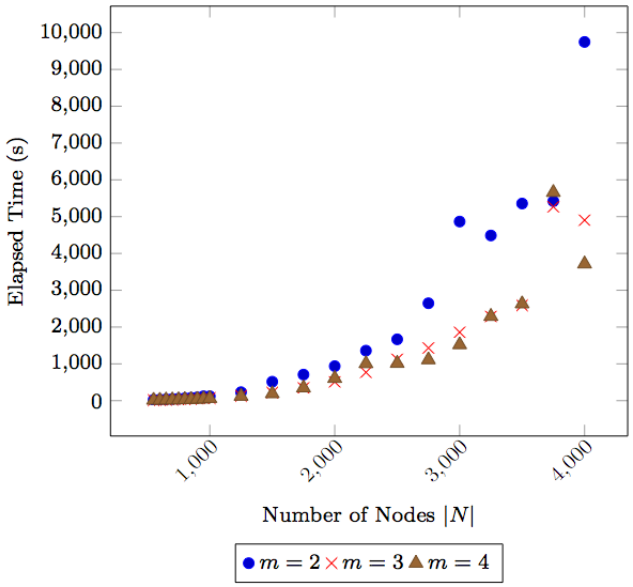

Fig. 2 Average Required Elapsed Time for the DP when tested on networks with an increasing number of nodes.

encounter networks of this large size for the application at hand (e.g., see [18]). Hence, the tractability of up to approximately 3000 node-sized problems for solution via the DP is a quite promising result that demonstrates its viability as a solution method.

4 Conclusions and Recommendations

In this paper, we have studied and compared a mixed-integer linear mathematical programming formulation and a dynamic programming approach to model and solve the problem of routing an aircraft over a directed network without cycles, with a fixed aircraft fuel capacity and a subset of nodes at which the aircraft may conduct an aerial refueling. Both of these solution methods sought to minimize the same objective, a weighted combination of the distance traveled and the number of required aerial refueling operations. Considering their performance over a battery of 264 representative test instances that reflected 66 combinations of variations in network size and the frequency of aerial refueling point availability, we found that the DP is robust in terms of network size and the required computational effort necessary to attain an optimal solution, whereas the MILP fails to be a viable solution method for larger-sized networks based on its inability to find optimal solutions, or even feasible solutions for some instances, within a one hour time limit.

For future study, we propose the further comparison of the two solution methods examined herein, but within the context of a stochastic optimization problem. A shortcoming to this study is the assumption that the fuel used by an aircraft between any two nodes is deterministic. On the contrary, there exists variance in fuel consumption rates based on the aircraft's engine type, airspeed, altitude, engine maintenance, and fuel quality, as well as the manner in which the aircraft is flown, which vary by pilot, weather, and enemy threat. Prior to conducting the stochastic study, we recommend implementing our dynamic programming node-labeling algorithm, but with a modification to existing constraints such that the fuel in the aircraft upon arriving at any node must be greater than some minimum level other-than-zero, thereby safeguarding against a variance in fuel consumption ratios.

Disclaimer

The views expressed in this article are those of the authors and do not reflect the official policy or position of the United States Air Force, the Department of Defense, or the United States Government.

Acknowledgements The authors thank the Editor and two anonymous referees for their constructive comments and suggestions that have greatly helped improve the substance and presentation of this paper.

References

1. R.K. Ahuja, T.L. Magnanti, and J.B. Orlin. *Network flows: Theory, Algorithms, and Applications*. Prentice hall, 1993.
2. C. Barnhart, P. Belobaba, and A.R. Odoni. Applications of operations research in the air transport industry. *Transportation science*, 37(4):368–391, 2003.
3. C. Barnhart, N.L. Boland, L.W. Clarke, E.L. Johnson, G.L. Nemhauser, and R.G. Shenoi. Flight string models for aircraft fleeting and routing. *Transportation Science*, 32(3):208–220, 1998.
4. M.C. Bartholomew-Biggs, S.C. Parkhurst, and S.P. Wilson. Using direct to solve an aircraft routing problem. *Computational Optimization and Applications*, 21(3):311–323, 2002.
5. M.C. Bartholomew-Biggs, S.C. Parkhurst, and S.P. Wilson. Global optimization approaches to an aircraft routing problem. *European Journal of Operational Research*, 146(2):417–431, 2003.
6. M.S. Bazaraa, J.J. Jarvis, and H.D. Sherali. *Linear Programming and Network Flows*. John Wiley & Sons, 2011.
7. R. Bellman. On a routing problem. Technical report, DTIC Document, 1956.
8. B.A. Bush. *Analysis of Fuel Consumption for an Aircraft Deployment with Multiple Aerial Refuelings*. PhD thesis, North Carolina State University, 2006.
9. J.R. Crino, J.T. Moore, J.W. Barnes, and W.P. Nanry. Solving the theater distribution vehicle routing and scheduling problem using group theoretic tabu search. *Mathematical and Computer Modelling*, 39(6):599–616, 2004.
10. G. Desaulniers, J. Desrosiers, Y. Dumas, M.M. Solomon, and F. Soumis. Daily aircraft routing and scheduling. *Management Science*, 43(6):841–855, 1997.
11. I. Dumitrescu and N. Boland. Algorithms for the weight constrained shortest path problem. *International Transactions in Operational Research*, 8(1):15–29, 2001.

12. S. Erdoğan and E. Miller-Hooks. A green vehicle routing problem. *Transportation Research Part E: Logistics and Transportation Review*, 48(1):100–114, 2012.
13. D. Feillet, P. Dejax, M. Gendreau, and C. Gueguen. An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. *Networks*, 44(3):216–229, 2004.
14. R.W. Harder, R.R. Hill, and J.T. Moore. A java universal vehicle router for routing unmanned aerial vehicles. *International Transactions in Operational Research*, 11(3):259–275, 2004.
15. P.E. Hart, N.J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.
16. IBM. CPLEX user's guide, 2014. Accessed June 30, 2014 at: `http://pic.dhe.ibm.com/infocenter/cosinfoc/v12r5/index.jsp`.
17. S. Irnich and G. Desaulniers. *Shortest path problems with resource constraints*. Springer, 2005.
18. T.E. Kannon, S.G. Nurre, R.R. Hill, and B.J. Lunday. The aircraft routing with air refueling problem: Exact and greedy approaches. In *Proceedings of the Industrial and Systems Engineering Research Conference*, Montreal, Canada, 2014.
19. Lockheed Martin Inc. Specifications: F22 raptor, December 2013. Accessed January 31, 2014 at: `http://www.lockheedmartin.com/us/products/f22/f-22-specifications.html`.
20. K.P. O'Rourke, W.B. Carlton, T.G. Bailey, and R.R. Hill. Dynamic routing of unmanned aerial vehicles using reactive tabu search. *Military Operations Research*, 6(1):5–30, 2001.
21. G. Righini and M. Salani. New dynamic programming algorithms for the resource constrained elementary shortest path problem. *Networks*, 51(3):155–170, 2008.
22. J.O. Royset, W.M. Carlyle, and R.K. Wood. Routing military aircraft with a constrained shortest-path algorithm. *Military Operations Research*, 14(3):31–52, 2009.
23. O.J. Smith, N. Boland, and H. Waterer. Solving shortest path problems with a weight constraint and replenishment arcs. *Computers & Operations Research*, 39(5):964–984, 2012.
24. K. Sundar and S. Rathinam. Algorithms for routing an unmanned aerial vehicle in the presence of refueling depots. *IEEE Transactions on Automation Science and Engineering*, 11(1):287–294, 2014.