Time-Dependent Network Assignment
Strategy for Taxiway Routing at Airports

Hojong Baik, Hanif D. Sherali, and Antonio A. Trani

A time-dependent network assignment strategy is proposed for efficiently handling aircraft taxiway operations at airports. The suggested strategy is based on the incremental assignment technique that is frequently adopted in many urban transportation studies. The method assumes that the current aircraft route is influenced by previous aircraft assignments in the network. This simplified assumption obviates the need for iterative rerouting procedures for attaining some pure equilibrium state, which in any case might not be achievable in practical airport taxiway operations. The main benefit of applying the time-dependent network assignment approach to taxiway operations is the reduction and avoidance of possible conflicts that produce delays. Also proposed is a prototype of a fully time-dependent network assignment scheme that dispatches aircraft based also on future anticipated assignment. The suggested methodology could be adopted in the deployment of automated taxiway guidance systems that are planned for future implementation at congested airports.

Assigning taxiway paths for arriving and departing flights is an important responsibility of air traffic controllers at towered airports. Even for small airports, this might be important if automated conflict resolution is desired. The simplest way to establish aircraft taxiway paths is to use routes based on shortest-distance paths. Such paths are usually static in the sense that they are independent of any temporal changes in traffic conditions on the taxiway network. An approach based on this concept is relatively easy to implement but is unlikely to produce optimal results on a systemwide basis because traffic conditions at an airport network change quickly over time. Another option is to apply a time-dependent network assignment strategy that considers changes in traffic conditions over time.

The main purpose of this paper is to develop a time-dependent network assignment (TDNA) approach that can provide ground controllers with effective routing plans that tend to reduce taxiing delays at airports. The plans developed by the proposed methodology involve taxiway routes from gates to runways (or more precisely to departure queues) for departing aircraft, and from runways to gates for arriving aircraft.

The taxiway operational problem is described first, and previous research on network assignment and shortest-path algorithms is briefly reviewed. Then detailed procedures are presented for the suggested methodology along with computational results obtained for one sample airport. A way to extend the suggested model to accommodate a fully time-dependent network assignment strategy also is discussed. Finally, some possible applications of this work for use in fast-time computer simulation models and real-time air traffic control contexts are presented.

PROBLEM DESCRIPTION

The main issue involved in the taxiway routing problem is to find an efficient set of taxi routes for both arriving and departing aircraft between runways and gates, given a time-dependent flight schedule that includes the starting time, origin and destination points, and the network capacity constraints. The taxiway routing problem can be stated more formally as follows:

Given a network configuration of runways and taxiways, including a set of origin nodes, O; and a set of destination nodes, D; and time-dependent taxiing demands from each origin to each destination for a certain (rolling) horizon, find a set of optimal routes for the departing aircraft to lead them from gates to departure queues, and for the arriving aircraft to lead them from runway exits to gates, in order to minimize total travel time.

To the authors’ knowledge, solutions to the taxiway routing problem have been implemented inside several fast-time airport simulation models such as SIMMOD and TAAM. These models, however, adopt static shortest-path (SSP) algorithms that produce suboptimal solutions when multiple aircraft ground paths are viewed collectively. The static path algorithms are simple to implement but have several inherent drawbacks. First, they neglect any temporal changes in the traffic demand pattern on the taxiway network. Second, most procedures of this type assume a “directed” network. In contrast, an airport taxiway link is neither strictly directed nor undirected in that it can be used in either direction unless there is an aircraft taxiing in the opposite direction. In other words, almost all links on the taxiway network are “bi-directional,” in that the link’s operational direction can change over time. To consider time-dependent characteristics of the usage of such taxiway links, a TDNA strategy that has a look-ahead function for each link’s operational direction needs to be introduced.

The aircraft taxiway routing problem has characteristics similar to the TDNA problem that is typically considered in urban transportation planning (1, 2). The following section presents a review of past research on the time-dependent network assignment problem to highlight some of the similarities and differences within the present context.

REVIEW OF TDNA ALGORITHMS

Network Assignment Strategies

Similar to the static network assignment problem, there are two types of time-dependent network assignment problems: (a) time-dependent system optimal assignment problem (TDSO), which seeks to minimize the total system travel time over the planning
horizon, and (b) time-dependent user equilibrium assignment problem (TDUE), which seeks time-dependent user path assignments so that the time-dependent travel times for all used paths connecting a given origin-destination (O-D) pair are less than or equal to the time-dependent travel times on any unused routes (2).

Using optimal control theory, Friesz et al. (3, 4) and Wie (5) present formulations for time-dependent traffic assignment in continuous time. Ran et al. (6) refine and extend optimal control models to include elastic demands and departure-time choices. Ran et al. (7) also propose time-dependent travel time functions for signalized network links that can be used to solve discrete-time dynamic assignment problems.

Peeta and Mahmassani (2) suggest a formulation for the path-based assignment problem that involves a nonexplicit function of path travel times. Using a mesoscopic simulation model called DYNASMART as a tool to measure experienced path travel times, Peeta and Mahmassani derive the conditions for a TDSS state where the time-dependent marginal travel times for all used paths connecting a given O-D pair are equal to or less than the time-dependent marginal travel times on any unused routes. As a solution algorithm, the method of successive averages (MSA) is used to determine the new path flows for the next iteration. Using the time-dependent experienced link travel times as prescribed by the simulator, the time-dependent shortest paths for all O-D pairs are delineated.

It is widely accepted that the effectiveness of the TDNA problem largely depends on the efficiency of the shortest-path algorithm embedded in the network assignment procedure. This is important when the TDNA problem is implemented in a real-time network control system such as an airport. Adopting this perspective, previous studies on time-dependent, shortest-path algorithms are reviewed next.

**Time-Dependent, Shortest-Path Algorithms**

The time-dependent, shortest-path (TDSP) problem is identical to the static shortest-path (SSP) problem except that, unlike the latter, it also considers temporal changes in link travel times. Labeling algorithms are known to be the most efficient methods to solve SSP problems. (The label in the algorithm represents the tentative shortest-path length from the source node to that node.) There are two types of labeling algorithms: label setting (LS) and label correcting (LC). The LS algorithm sets the label of one node permanently at each iteration, thus increasing the shortest-path vector by one component at each iteration. This scheme is applicable to shortest-path problems having nonnegative link costs. [The LS algorithm is frequently referred to as Dijkstra’s algorithm because Dijkstra was one of the first to discover it independently (8).] On the other hand, the LC algorithm is more generally applicable and follows an iterative update scheme whereby the different components of the shortest-path vector are revised as alternative shorter paths are discovered until the algorithm terminates.

TDSP algorithms also use the same type of labeling methods as devised for the SSP problem. Cook and Halsey (9) extend Bellman’s principle of optimality to solve the underlying TDSP problem. Dreyfus (10) suggests the use of Dijkstra’s algorithm to determine time-dependent shortest-paths where the link costs are any real-valued times. The difference between these two methods is that, whereas Cook and Halsey’s method applies the Principle of Optimality in forward form, Dreyfus’s algorithm is implemented in a backward fashion. Halpern (11) notes a particular limitation of Dreyfus’s approach and reveals that if there exists a y > 0 such that y + c_i(t + y) < c_j(t), for some link (i, j) and time t, where c_i(t) is the link travel time function, then the departure from node i must be delayed, or else the optimal path might include cycles. Kaufman and Smith (12) study the assumptions under which the existing TDSP algorithms would work and make a consistency assumption that ensures a first-in, first-out traversal of all the links in the network at all times. Orda and Rom (13) study various types of waiting-at-nodes scenarios and propose algorithms for these different cases. They show that if waiting is allowed at nodes, then the consistency assumption is not required. They also prescribe an algorithm for identifying optimal waiting times at the source node if waiting is not allowed elsewhere in the network. Sherali et al. (14) prove NP-hardness of various versions of the time-dependent shortest-path and pair of disjoint-path problems, and they develop efficient solution algorithms for different defined degrees of disjointedness.

Gallo and Pallottino (15) suggest that it is more desirable to classify the SSP algorithms based on the data structure that is used to maintain and extract candidate nodes for further considerations. A sorted queue is used for the LS shortest-path algorithm in which a sorting method is imbedded so as to select a node having the least label from the set of candidate nodes. However, it should be noted that if the problem size is large, the sorting algorithm is not inexpensive in terms of computational cost. To address this drawback, a special type of data structure called double-ended queue has been developed, which combines the properties of both the queue and the stack. In the double-ended queue structure, the first time a node is identified as a candidate, it is inserted into the tail of the queue. When, later on, the same node again becomes a candidate node after being removed from the queue, it is inserted at the head of the queue. The node selection process removes candidate nodes from the head of the queue. Maintaining the double-ended queue to handle the candidate nodes, Ziliaskopoulos and Mahmassani (1) devise a time-dependent, shortest-path algorithm.

**DEVELOPMENT OF TDNA ALGORITHM FOR AIRPORT TAXIWAY OPERATIONS**

**Network Assignment Strategy**

The network assignment strategy proposed in this paper for the taxiway routing problem is a quasi time-dependent network assignment (QTDNA) strategy. The term “quasi” is used to qualify that the proposed strategy is designed to achieve neither a complete TDUE nor a pure TDSS state. Unlike as in urban transportation networks, there are relatively few vehicles traversing an airport network. The taxiing demand in the present context is typically too small to warrant making the type of minute controls to individual traffic that are often conducted while attaining an equilibrium (or optimal) state.

In addition to the computational aspect, there is another practical rationale for not applying TDUE (or TDSS) concepts at an airport. In the TDUE (or TDSS) framework, it is assumed that vehicles select their best paths in such a way that their own travel times (or marginal travel times) are minimized. It is also assumed that all vehicles have equal priorities so that the rule of first-come, first-served is maintained during travel. At an airport, however, ground controllers often provide higher priorities to certain taxing aircraft than to others to facilitate the overall traffic flow or to balance the use of runways. Such a nonsystematic situation might keep the taxiway network problem from achieving a pure TDUE (or TDSS) state.
In the QTDNA strategy, it is assumed that the current aircraft route is influenced only by the set of aircraft previously assigned to the network. (This assumption will be referred to as a precedence assumption.) This simplified assumption rules out the necessity of iterative rerouting procedures, thereby reducing the complexity of computations. Figure 1 depicts a flowchart to implement the QTDNA method. Fundamentally, the method is based on the incremental assignment technique that constructs aircraft paths in a sequential fashion, one at a time. The sequence of assignments is based on a prescribed schedule that is determined a priori by a separate module that solves the underlying “aircraft sequencing problem.” After an aircraft is assigned a route, the information on the affected links that lie on the taxing path are updated for use over the successive time slices. In particular, once an aircraft is assigned to a certain path, the opposite direction of each link included in the path is blocked during the time slice occupied by the aircraft. This can be implemented by increasing the travel time of the conflicting link to a sufficiently large number that would preclude its selection in a time-dependent shortest-path that is subsequently identified for any other aircraft. In practice, all the information about the dynamically loaded aircraft on each link is maintained in a fixed sized queue data structures array, called a time-dependent aircraft flow table.

Figure 2 presents a simple example for an aircraft $V_i$ that traverses from O to D along the path O→1→2→D. Here, the route of aircraft $V_i$ is traced over time slices and links, and is recorded in a corresponding cell in the time-dependent aircraft flow table. Then, a time-dependent link travel time table is updated by adding certain induced travel-time delays on the path links [i.e., on (O, 1), (1, 2), and (2, D)] and by adding large travel times on the opposite links [i.e., on (1, O), (2, 1), and (D, 2)] over corresponding time slices. The travel times for the links can be computed by applying any known link performance functions that represent the relationships between the link travel times and the link flows. (To the authors’ knowledge, no research has been conducted to study airport taxiway link performance functions. As such, we assume that link travel times increase linearly as the number of aircraft on the link increases.) Once the time-dependent link travel time table is updated, the time-dependent shortest-path for the next aircraft to be assigned to the network can be computed.

It is instructive to note the common link usage issue at this point. Consider the following example to illustrate this feature. Suppose that aircraft $A$ is assigned first and that it uses a link $(i, j)$ during some time interval $[t^A_i, t^A_j]$, so that during this interval, the reverse direction is blocked, and the forward direction has its travel-time function updated. Now suppose that another aircraft $B$ is assigned and needs to use the link $(i, j)$ in the same direction as that used by aircraft $A$. If the duration of this use is disjoint with the interval $[t^A_i, t^A_j]$, then there is no interaction with aircraft $A$. However, if the occupancy interval, $[t^B_i, t^B_j]$, of aircraft $B$ for link $(i, j)$ overlaps with that for $A$, we need to ensure that even if $t^j_i < t^B_i$, aircraft $B$ is queued after aircraft $A$. To reflect this fact, the revised travel-time function on link $(i, j)$ is defined after $A$ has been loaded on it so that for each time slot of entrance on link $(i, j)$ for which a conflict with aircraft $A$ would occur over $[t^j_i, t^B_j]$, the corresponding travel time considers waiting to queue up behind aircraft $A$ in addition to any congestion-related delays.

**Time-Dependent, Shortest-Path Algorithm**

The TDSP algorithm suggested by Ziliaskopoulos and Mahmassi (1) is designed to provide time-dependent, shortest-paths for all O-D pairs for each (discretized) starting time. It should be noted that the size and structure of the taxiway network problem are much smaller and simpler than those of an urban transportation network problem. For example, the taxiway system for a large airport might contain several hundred links and nodes. Also, the number of active aircraft at a busy airport might be around 200 or so compared with thousands of vehicles typically considered at any time of day in urban transportation network contexts. Motivated by this, a TDSP algorithm based on Dijkstra’s algorithm is used, having the following characteristics:

- The algorithm provides time-dependent shortest-paths from a single root node to all other nodes starting at any given time $t$.
- The algorithm uses the sorted queue as a data structure for candidate nodes, which renders it as an LS procedure rather than an LC.
A TDSP algorithm for finding time-dependent shortest-paths from a root node, \( r \), to all the other nodes in the network starting at a time \( t \) corresponding to any time slice is described by the following pseudo code.

Procedure \texttt{Initialize}

\[
\text{while(SE list is not empty)} \{
\text{\hspace{0.5cm} } u = \text{Call deQueue};
\text{\hspace{0.5cm} } d_u = \text{Label}(r, u); \} // \text{arrival travel at node } u \text{ starting from } r \text{ at time } t.
\text{\hspace{0.5cm} } \text{Compute timeSlice}_{d_u}; // \text{compute the time slice corresponding to } d_u.
\text{\hspace{0.5cm} } \text{for(} v = \text{all forward star of } d_u \text{)} [\]
\text{\hspace{1.0cm} } d_v = \text{Label}(r, v); // \text{travel time from } r \text{ to } v.
\text{\hspace{1.0cm} } l_{uv} = \text{Find travelTime}(u, v, \text{timeSlice}_{d_u}); // \text{find the travel time for link } (u, v) \text{ at timeSlice}_{d_u}
\text{\hspace{1.0cm} } \text{if}(d_v > d_u + l_{uv}) [\]
\text{\hspace{1.5cm} } \text{Label}(r, v) = d_u + l_{uv}; // \text{update travel time from } r \text{ to } v.
\text{\hspace{1.5cm} } \text{Predecessor}(v) = u; // \text{update predecessor node for node } v
\text{\hspace{1.5cm} } \text{Call enQueue}(v);
\text{\hspace{0.5cm} } \}\] // \text{end for}
\text{\hspace{0.5cm} } \}\] // \text{end while}

Procedure \texttt{deQueue}: // \text{find the closest node from the candidate nodes set (i.e., SE list) using the quick-sort algorithm.}

Procedure \texttt{enQueue}(x): // \text{insert node } x \text{ into the candidate nodes set (i.e., SE list).}

**COMPUTATIONAL RESULTS**

To compare static and time-dependent assignment strategies for the taxiway routing problem, a hypothetical flight schedule at Ronald Reagan (previously called Washington) National Airport will be considered. It is assumed that only two flights are scheduled over the next 2 min. Some relevant information for these flights is given in Table 1.

Figure 3 describes the paths resulting from the two different assignment strategies. (The values shown are the node numbers in the graph along with the estimated arrival times, in seconds, given in parentheses.) Because no flight is on the taxiway when flight DEP_1 is assigned, it can taxi to its destination at an unimpeded speed. Therefore, there is no difference in the static and the time-dependent shortest-paths for flight DEP_1.

In the static assignment strategy, while DEP_2 just taxies along its shortest-path without considering the other flight’s taxiing path, it experiences a conflict with DEP_1 on the link (1024, 1020). On the other hand, in the time-dependent assignment case when DEP_2 is assigned to the network, the TDSP algorithm detects that link (1024, 1020) will be blocked by DEP_1 during the latter’s corresponding occupancy duration. As a result, DEP_2 is assigned to a path that makes a detour so as to avoid this conflict with DEP_1. The difference between the two paths for DEP_2 is highlighted in bold in Figure 3 and illustrated in Figure 4.

Another relevant consideration in the present context might be the total travel time for paths prescribed by the static and time-dependent assignment strategies.
assignment strategies. For this comparison, however, realistic link performance functions should be delineated so that the time-dependent link travel times can be measured with accuracy. In order to obtain such link performance functions, extensive data need to be collected at airports and analyzed using statistical techniques. A surrogate way to accomplish this would be to implement stochastic models into a fast-time simulation model and thereby estimate individual link travel times. To the authors' knowledge, little or no research to find airport taxiway link performance functions has been conducted.

**EXTENSION TO PROPOSED NETWORK ASSIGNMENT STRATEGY**

In the proposed QTDNA, it is assumed that the current aircraft route is influenced only by previous aircraft assignments in the network (i.e., the precedence assumption). The principal motivation for this assumption is to facilitate real-time routing decisions with a reasonable computational effort, by avoiding any iterative feedback computations and reassignments due to the effects on the subsequently assigned aircraft. As discussed in the section on network assignment strategy, this assumption results in adding extra waiting delays for any aircraft that enters a common link that is occupied by some previously assigned aircraft, and this could result in increasing the total travel time. However, it should also be recognized that the design of a network assignment strategy should reflect its application perspective. For example, if the strategy is to be implemented in a real-time air traffic control system, the computational effort is of paramount importance. In contrast, if the strategy is applied to an airport planning study, and dispatchment plans are composed off-line in advance, then minimizing the total taxiing time would be of more interest than limiting the computational effort. In this case, a fully time-dependent network assignment algorithm might be more desirable. A way to extend the proposed QTDNA to such a TDNA strategy is suggested below.

The main idea behind the suggested procedure is to first apply the proposed QTDNA procedure in order to compose an assignment scheme for all the aircraft. Because no explicit link performance function is available, a simulation model described by Baik (16) is used next to measure experienced travel times, while considering aircraft kinematic behavior and communication activities between the controllers and the pilots. This provides a more realistic estimate for the total (sum of individual) travel-time criterion. Using the current dispatchment sequence, pairs of aircraft (in turn) are now identified that violate the precedence assumption in the sense

![Diagram](https://via.placeholder.com/150) **FIGURE 4** Depiction of the static versus time-dependent paths: (a) static assignment path, (b) time-dependent assignment path.
that the later aircraft in such a pair arrives at some common link used by both earlier than the previous aircraft. (Such pairs also could be identified based on the later aircraft being precluded to use a link in the opposite direction relative to its occupancy by a previous aircraft, in case this type of event is readily identifiable.) If the switching of any such pair improves the quality of the overall solution as ascertained by updating the routes and their schedules via the QTDNA process and (partially) resimulating the resulting sequence, then this switch is implemented, and the next such pair is considered. (Note that from the viewpoint of computational efficiency, it is important to design the interface between QTDNA and the simulation scheme to update the affected decisions efficiently, rather than repeat the entire process from scratch.) If the switching of each pair of this type yields no improvement in a complete cyclic pass, then the procedure is terminated. Note that in a similar spirit to the foregoing pairwise switches, some limited set of more complex multiway exchange could be explored. The details of the design and computational implementation of such enhanced procedures are recommended for future research.

CONCLUSIONS AND RECOMMENDATIONS

A quasi–time-dependent assignment strategy for the taxiway routing problem that ground controllers face at airports has been introduced. The suggested algorithm is based on an important assumption that the current aircraft route is in the current aircraft route is in the current aircraft route is in the current aircraft route influenced only by the set of aircraft that have previously been assigned to the network. Obviously, the assumption facilitates the computational process by avoiding reiterative computations and also permits the consideration of bi-directional links having different travel time characteristics in either direction that are dependent on previously assigned aircraft. Furthermore, the procedure can readily consider additional aircraft to be inserted subsequently in a rolling horizon framework.

One possible enhancement is to use an advance type of TDSP algorithm that allows waiting at nodes. A comparison of the proposed QTDNA procedure and a fully TDNA strategy as suggested in this paper is also recommended for future study. As discussed, all such procedures also require the development of reliable link performance functions that can provide link travel times with good accuracy.

The proposed algorithmic procedure is capable of detecting aircraft conflicts and congestion effects. It could be implemented as part of an advisory ground control system and used by a real-time airport traffic management system to direct aircraft along taxiways with minimal delays. The algorithm may also be imbedded in any type of microscopic airport simulation package as a computational module to find a shortest-path for a given aircraft. The described procedure has been implemented in the Virginia Tech Airport Simulation Model (VTASIM)—a microscopic airport simulation model developed to study various ground traffic management architectures (16).

ACKNOWLEDGMENTS

The authors would like to acknowledge Randy Stevens of the FAA for his support of NEXTOR—the National Center of Excellence for Aviation Operations Research—under the Advanced Air Traffic Management Agenda. This research also has been supported by the National Science Foundation. The authors also thank the referees for their constructive comments that helped improve the discussion in this paper.

REFERENCES