IET Intelligent Transport Systems

Research Article

IET Journals
The Institution of Engineering and Technology

Improved approach for time-based taxi trajectory planning towards conflict-free, efficient and fluent airport ground movement

ISSN 1751-956X
Received on 16th December 2017
Revised 8th July 2018
Accepted on 7th August 2018
E-First on 2nd October 2018
doi: 10.1049/iet-its.2018.5193
www.ietdl.org

Tianci Zhang[1,2], *Meng Ding*[1,2] ✉, *Hongfu Zuo*[1,2]

[1]*College of Civil Aviation, Nanjing University of Aeronautics and Astronautics, 29 Jiangjun Avenue, Nanjing 211106, People's Republic of China*
[2]*Laboratory of Civil Aircraft Health Monitoring and Intelligent Maintenance, 29 Jiangjun Avenue, Nanjing 211106, People's Republic of China*
✉ *E-mail: nuaa_dm@nuaa.edu.cn*

Abstract: The ever-growing air traffic demand arouses an urgent need for improved airport ground movement efficiency. New operational concepts are emerging which use time-based taxi trajectories to reduce uncertainty and make more efficient use of the airport resource. In this study, an improved approach is proposed for time-based taxi trajectory planning, which is formulated as the shortest path problem with time windows and the maximum traversal time constraint. With the introduction of the taxi time in the cost and the maximum traversal time constraint to limit the waiting time, more efficient and fluent ground movement of aircraft can be realised. An A*-based solution algorithm is developed for the investigated problem, which utilises the arrival time interval and dominance-based comparison to search for the best solution. Experimental results on real-world problem instances demonstrate the effectiveness of the proposed approach as well as its advantages over the existing approach.

1 Introduction

Safe and efficient airport ground movement plays a key role in achieving fluent traffic flow through an airport [1, 2]. Currently, the advanced surface movement guidance and control system is being deployed in many airports around the world, which enhances the surveillance and management of airport surface traffic, and enables the newly developed concept of trajectory-based taxi operations [3, 4]. Trajectory-based taxi operations assign a time-based (or four-dimensional) taxi trajectory to each departing/arriving aircraft, which specifies not only the route on the taxiway that aircraft should follow during taxiing, but also the required times of arrival at specified intermediate positions along the route [5, 6]. By this means, conflict-free and efficient taxi trajectories can be generated in the planning stage, which largely eliminates the uncertainty and unpredictability in the ground movement and facilitates the holistic optimisation of airport operations including the apron, runway and near terminal area [4, 7].

Mixed-integer programming methods are proposed in [8, 9] to generate conflict-free and efficient time-based taxi trajectories for a group of aircraft. However, as the problem is NP-hard, such exact methods are often computationally demanding and not suitable for online decision support in the dynamic environment of an airport [3, 4]. More efficient approaches based on sequential routing are proposed in [10–12], which generate time-based taxi trajectories for a group of aircraft in the predefined order. The sequential routing approach divides the problem by aircraft and uses quickest path algorithms to search for feasible trajectories, which greatly reduces the computational cost compared with the exact methods [8, 9]. A key issue for the sequential routing approach is to ensure conflict-free movement of forthcoming aircraft with respect to the already determined trajectories. In [10], the required times of arrival at each intersection of the taxiway for different aircraft is separated between one another. However, this cannot ensure the avoidance of head-on conflicts within a taxiway segment between two adjacent intersections. A more effective approach for conflict avoidance is therefore used in [11, 12], which reserves the time interval during which an aircraft will go through each zone of the taxiway along the assigned taxi trajectory. Forthcoming aircraft can only go through these zones within time intervals not overlapping with the reserved time intervals. Such feasible time intervals are often referred to as free time windows. To find out a feasible trajectory, a search graph is built upon the layout of the taxiway, and the quickest path algorithm is then applied, which iteratively expands the partial solution paths towards the destination. In each expansion, the accessibility to the neighbouring vertex in the search graph is examined considering the reserved time intervals, and the partial solution path with the earliest arrival time at the neighbouring vertex is recorded. When the destination vertex is selected for expansion, the quickest path to the destination (i.e. the path with the earliest completion time) is determined. However, it is worth noting that the earliest completion time does not necessarily mean the most efficient solution, as aircraft might start taxiing very early and have overlong taxi time. Moreover, the earliest arrival time at a vertex does not guarantee that aircraft can enter the next zone of the taxiway immediately. For example, when the next zone is still reserved for another aircraft, the aircraft needs to wait in the current zone until the next zone is cleared. As there is no restriction on the maximum traversal time for any zone, long time of waiting might be required during taxiing as long as the aircraft has the earliest completion time. However, long time of waiting is not recommended during active runway crossing for safety reasons. When occurring at the traffic merging points, long time of waiting may block other aircraft, leading to adverse impact on the overall efficiency of the ground movement. Furthermore, long time of waiting will also impede fluent movement of the aircraft and lead to stop-and-go during taxiing, which will cause more fuel consumption and emissions [4, 13].

1.1 Contributions

According to the above discussions, desirable time-based trajectories should guarantee conflict-free, efficient and fluent ground movement of aircraft. Among the abovementioned algorithms, algorithms in [11, 12] are specifically devised for conflict-free time-based taxi trajectory planning. On this basis, this paper proposes an improved approach towards the efficiency and fluency goals. The proposed approach (i) explicitly penalises the taxi time in the cost to avoid solutions with small completion time but overlong taxi time, and (ii) uses the maximum traversal time constraint to avoid blocking safety critical regions and eliminate long time of waiting during taxiing. According to our knowledge, none of the existing studies on constrained shortest path problems considers the time windows and the maximum traversal time constraint simultaneously (see e.g. [14–17]). Due to the use of the earliest arrival time during search, potentially better routing options

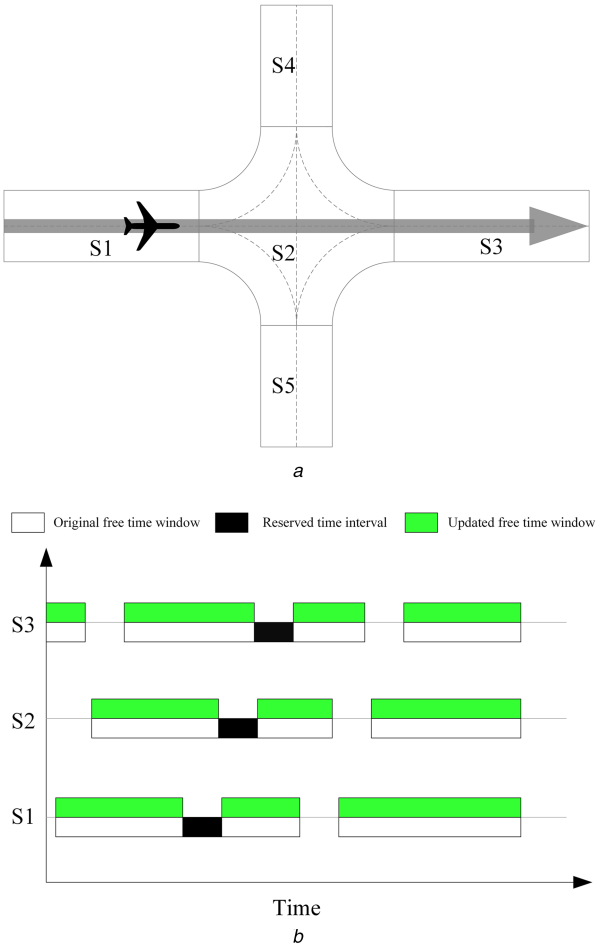

Fig. 1 *Illustration of time interval reservation and free time window update*

(a) Assigned taxi trajectory and the zones of the taxiway, ***(b)***Reserved time intervals and the original and updated free time windows

could be lost if the maximum traversal time constraint is directly introduced into the algorithms of [11, 12].

Therefore, an A*-based solution algorithm is proposed in this paper, which can effectively handle the maximum traversal time constraint via two key schemes: (i) the use of the arrival (and exit) time intervals, and (ii) the dominance rule for comparing different partial solution paths. Extending the earliest arrival time to the arrival time interval helps to capture all possible routing options while still respecting the reserved time intervals and the maximum traversal time constraint. The dominance rule operates over the arrival time interval and considers the cost of the partial solution path, which ensures effective comparison between different partial solution paths to the same intermediate position. It is worth pointing out that the quickest path algorithm in [11, 12] can be regarded as special instances of the algorithm proposed in this paper, in which the arrival time interval has reduced to the earliest arrival time. The proposed algorithm also provides a general solution approach for a class of problems in other fields with similar traversal time constraints (e.g. the routing of automated guided vehicles in flexible manufacturing and container handling systems [18–22]).

1.2 Structure of the paper

The rest of the paper is organised as follows. Section 2 describes the investigated problem and discusses the impact of the maximum traversal time constraint on the routing process. Section 3 presents the solution algorithm for the investigated problem. Section 4 presents experimental results on problem instances for a real-world airport to demonstrate the effectiveness and advantages of the proposed approach. Section 5 concludes the paper and presents future research directions.

2 Problem description

This paper investigates the problem of determining the time-based taxi trajectory (i.e. taxi route with required time of arrival at every zone of the taxiway along the route) for departing/arriving aircraft satisfying the following requirements: (i) to ensure conflict-free movement, the aircraft should avoid using any zone of the taxiway within the time intervals reserved for other aircraft (see Section 2.1); (ii) to make the movement fluent, the aircraft should taxi without overlong waiting time at any intermediate position of the taxiway, which is guaranteed by the maximum traversal time constraint (see Section 2.2); (iii) the resulting trajectory should have the least cost, which is defined as the linear combination of the completion time and the taxi time. Combining the abovementioned requirements leads to a specific type of constrained shortest path problem, which is referred to as the shortest path problem with time windows and the maximum traversal time constraint (SPPTW-MTTC) in this paper.

2.1 Time interval reservation and free time window update

When a taxi trajectory is assigned to an aircraft, the time interval during which the aircraft is expected to go through each zone will be reserved in order to avoid conflict with forthcoming aircraft [11, 12]. An illustration of this concept is presented in Fig. 1. Assume the assigned taxi trajectory of an aircraft passes through zones S1–S3 of the taxiway, as shown in Fig. 1*a*. The corresponding time interval reservations are as shown in Fig. 1*b*. Each of the reserved time intervals is within one of the original free time windows of the corresponding zone. For conflict-free routing of forthcoming aircraft, the set of free time windows for each zone is then updated by removing the reserved time interval from the corresponding free time window, as shown in Fig. 1*b*. It should be noted that the original free time windows of each zone in Fig. 1*b* are already split by the time intervals reserved for other aircraft.

2.2 Maximum traversal time constraint

In existing studies [10–12], when the aircraft arrives at a zone of the taxiway within a free time window, it is feasible for the aircraft to exit from the zone until the end of the free time window. As a result, all the time points after an unimpeded traversal time could be picked as the actual exit time, as shown in Fig. 2. The duration between the arrival time and the actual exit time is referred to as the traversal time. The aircraft needs to wait if the traversal time is larger than the unimpeded traversal time $t_{\min}$. Although such waiting is conflict free, the aircraft may have to stop and hold if long waiting time is required, which goes against the goal of fluent movement. To avoid this issue, the maximum traversal time (i.e. $t_{\max}$ in Fig. 2) should be taken into consideration, which is determined by the minimum taxi speed $v_{\min}$ allowed in the corresponding zone. After imposing the maximum traversal time constraint, the set of feasible exit times usually becomes smaller.

It is worth noting that a direct introduction of the maximum traversal time constraint to the quickest path algorithm in existing studies [11, 12] could make it lose potential routing options during the search. This will lead to either an inferior solution or failure in finding a feasible solution (even though better or feasible solutions do exist). Fig. 3 presents an illustration of this issue, where different cases of routing from zone x to its neighbouring zone y are examined. In Fig. 3*a*, the maximum traversal time constraint is not introduced. Assume t_1 and t_2 are two available times at which the aircraft can arrive at zone x. The feasible exit times due to t_1 and t_2 are determined as described in Fig. 2. The feasible arrival times at zone y are then determined by the overlaps between the feasible exit times from zone x and the free time windows of zone y. As shown in Fig. 3*a*, the feasible exit times due to t_1 includes those due to t_2. As a result, there are more feasible arrival times at zone y due to t_1, which span two free time windows. In contrast, the feasible arrival times at zone y due to t_2 only meet the second free time window. Moreover, if the aircraft arrives at zone x at t_1, even earlier feasible arrival times can be found in the second free

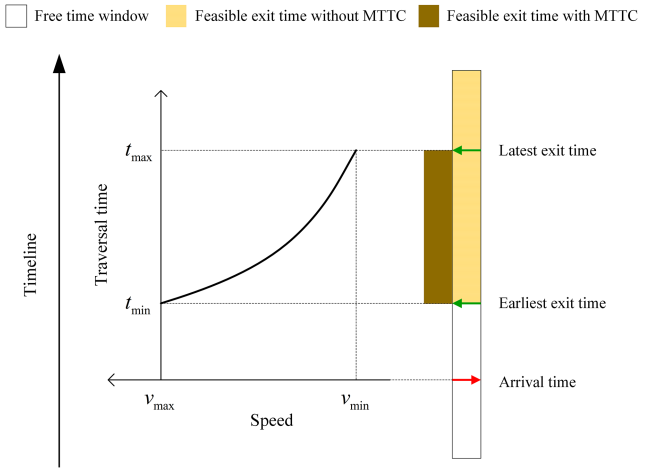

Fig. 2 *Illustration of the maximum traversal time constraint*

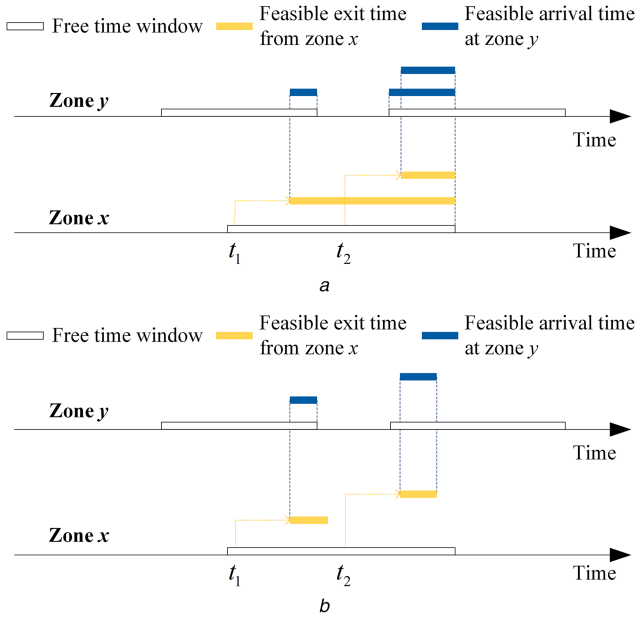

Fig. 3 *Impact of the maximum traversal time constraint on the routing process*

(a) Original case without the maximum traversal time constraint, ***(b)***Case after introducing the maximum traversal time constraint

time window of y than those corresponding to t_2. Therefore, if there is no maximum traversal time constraint, an earlier arrival time at zone x can result in more free time windows available for expansion from x to y with possibly earlier arrival time in each of the available time windows. The abovementioned principle applies to the expansion to zone y and so forth for the subsequent zones. Therefore, in order to have more routing options, it is legitimate to keep only the earliest feasible arrival time in each free time window and ignore other feasible arrival times. This is exactly the strategy used in the quickest path algorithm [11, 12].

However, the abovementioned principle does not hold when the maximum traversal time constraint is introduced. In this case, the aircraft cannot wait for arbitrary time within a zone to meet the free time windows of the neighbouring zones. Fig. 3*b* shows the results after introducing the maximum traversal time constraint for the same routing process as in Fig. 3*a*. The feasible arrival times at zone y corresponding to arrival times t_1 and t_2 at zone x are now within two different free time windows of y, respectively. If the free time windows of a neighbouring zone z of zone y only overlap with the second free time window of y, the only chance of expansion from y to z is that the aircraft arrives at zone x at t_2. In other words, when the maximum traversal time constraint exists, earlier arrival time (i.e. t_1) at x does not necessarily lead to more routing options. Either inferior solutions or failure in finding out a feasible solution could be met at the end of the search if only t_1 is kept.

In light of the above discussions, a more sophisticated strategy is devised in the proposed algorithm in Section 3 to handle the maximum traversal time constraint. Instead of the earliest arrival time, the entire set of feasible arrival times (i.e. the arrival time interval) is utilised during the search for the desirable time-based taxi trajectory. Additionally, the comparison between different partial solution paths is realised by the dominance rule, as will be detailed in Section 3.2. It is worth noting that the proposed algorithm will first generate a relaxed form of time-based taxi trajectory which has an arrival time interval at each zone. For conflict-free routing of forthcoming aircraft, the exact arrival time at each zone will then be determined to make appropriate time interval reservations and update the free time windows as shown in Fig. 1. In this paper, the arrival time at each zone is determined sequentially from the destination backward to the starting position: the arrival time at the destination is the starting time of the corresponding arrival time interval. The arrival time at each of the rest zones along the trajectory is determined by the arrival time at the next zone subtracting the unimpeded traversal time of the present zone. If the resulting time cannot meet the corresponding arrival time interval, the ending time of the arrival time interval is selected as the arrival time at the present zone. The taxi time is then determined as the duration between the arrival time at the starting position and the arrival time at the destination. The abovementioned approach results in the time-based taxi trajectory with the earliest arrival time at the destination (within the corresponding arrival time interval) and satisfying the maximum traversal time constraint for all zones along the trajectory.

3 Solution algorithm

In this section, an A*-based algorithm (see e.g. [23]) is proposed to search for the time-based taxi trajectory meeting the requirements described in Section 2. The fundamental concepts used in the proposed algorithm are summarised in Table 1. The proposed algorithm searches upon a graph with temporal nodes as the vertexes. Similar to [12], temporal node $\eta_{x,m,i}$ is a triple of zone x, the access node m of x and the ith free time window of x. It indicates that zone x will be entered through node m within the ith free time window. For each temporal node, there could be different feasible routing options from the starting position, each associated with a specific arrival time interval and forming a partial solution path. For clarity, a partial solution will be denoted as (η, α_η), where η is the last temporal node of the partial solution path, and α_η is the corresponding arrival time interval. The conflict-free expansion of a partial solution path is ensured by accessibility determination, as will be detailed in Section 3.1. The comparison of different partial solution paths to the same temporal node is based on the dominance rule described in Section 3.2. During the search, the already expanded partial solution paths are stored in the close list, while the explored but not expanded partial solution paths are stored in the open list. The cost of partial solution path (η, α_η) is defined as the sum of the starting time of the arrival time interval α_η, the taxi time from the starting position to the end of the partial solution path, and the estimation of the time to finish the remaining journey (i.e. the heuristic in A*). It is worth noting that the taxi time from the starting position to the end of a partial solution path is determined in a similar way to the taxi time of the complete path described in Section 2. The cost of a complete path to the destination is therefore exactly the sum of the completion time and the taxi time.

The detailed steps of the proposed algorithm are presented in Fig. 4. After initialisation (lines 1–2), the algorithm selects the least-cost partial solution path (η, α_η) from the open list for expansion. If the selected partial solution path already reaches the destination, the algorithm finishes and generates the complete path according to the predecessor record $\hat{P}_\eta$ (lines 5–7). The predecessor record $\hat{P}_\eta$ recursively points to the temporal node in the path immediately before η as well as the corresponding arrival time interval until the starting position. If the selected partial solution

Table 1 Definition of concepts

Concept	Symbol	Meaning
free time window	$w_{x,i}$	ith maximum time interval during which zone x is available
exit time interval	β_x	maximum feasible time interval to leave zone x
arrival time interval	α_x	maximum feasible time interval to enter zone x
temporal node	$\eta_{x,m,i}$	triple of zone x, access node m and free time window $w_{x,i}$

Input: Start temporal node s, arrival time interval α_s, and target temporal node d
Output: Time-based taxi trajectory tr

1: Initialise the open list L_o, close list L_c and the predecessor record $\widehat{P}$ as empty;
2: Add (s, α_s) to L_o;
3: **while** $L_o \neq \varnothing$ **do**
4: $(\eta, \alpha_\eta) \leftarrow \arg\min_{L_o} cost(\eta, \alpha_\eta)$; /* select the least-cost partial solution path from open list */
5: **if** $\eta = d$ **then**
6: Generate tr from the destination backward to the origin according to $\widehat{P}_\eta$;
7: **return** tr;
8: **else**
9: $L_o \leftarrow L_o \backslash \{(\eta, \alpha_\eta)\}$;
10: $L_c \leftarrow L_c \cup \{(\eta, \alpha_\eta)\}$;
11: $(A, \alpha_A, P_A) \leftarrow \Gamma(\eta, \alpha_\eta)$; /* determine accessible temporal nodes */
12: **foreach** $q \in A$ **do**
13: $Flag \leftarrow true$;
14: **foreach** $(p, \alpha_p) \in L_c^q$ **do**
15: **if** $(p, \alpha_p) \prec (q, \alpha_q)$ **then** /* dominance comparison */
16: $Flag \leftarrow false$;
17: **break;**
18: **end**
19: **end**
20: **if** $Flag = true$ **then**
21: **foreach** $(p, \alpha_p) \in L_o^q$ **do**
22: **if** $(q, \alpha_q) \prec (p, \alpha_p)$ **then** /* dominance comparison */
23: $L_o \leftarrow L_o \backslash \{(p, \alpha_p)\}$;
24: **end**
25: **else if** $(p, \alpha_p) \prec (q, \alpha_q)$ **then** /* dominance comparison */
26: $Flag \leftarrow false$;
27: **break;**
28: **end**
29: **end**
30: **end**
31: **if** $Flag = true$ **then**
32: $L_o \leftarrow L_o \cup (q, \alpha_q)$; /* add partial solution path to open list */
33: $\widehat{P}_q \leftarrow \widehat{P}_\eta \cup P_q$;
34: **end**
35: **end**
36: **end**
37: **end**
38: **return** *null*

Fig. 4 *Proposed algorithm for SPPTW-MTTC*

path does not reach the destination, the algorithm removes (η, α_η) from the open list and adds it to the close list (lines 9–10). In line 11, the partial solution path (η, α_η) is expanded to the neighbouring temporal nodes within function $\Gamma(\cdot)$. All the accessible temporal nodes and the corresponding arrival time intervals are returned in A and α_A, respectively. The predecessors for the accessible temporal nodes are recorded in P_A. Each accessible temporal node and the corresponding arrival time interval form a candidate partial solution path. L_c^q (L_o^q) in line 14 (line 21) collects all partial solution paths to temporal node q from the close list L_c (open list L_o). In lines 14–19, for each candidate partial solution path (q, α_q), if (q, α_q) is dominated by any of the partial solution paths in L_c^q, the flag is set to false and no further operation for (q, α_q) is needed. Then, if any of the partial solution paths in L_o^q is dominated by (q, α_q), it will be removed from the open list (lines 22–24). Otherwise if (q, α_q) is dominated instead, the flag will again be set to false, and no further operation for (q, α_q) is needed (lines 25–28). Finally, if (q, α_q) is not dominated by any of the existing partial solution paths, this new partial solution path is added to the open list and the predecessor for the last temporal node of the partial solution path is added to the record (lines 31–34). The above process repeats until the open list is empty or the complete path to the destination is found.

In the proposed algorithm, the determination of the accessibility to neighbouring temporal nodes (line 11) and the dominance comparison between partial solution paths (lines 15, 22 and 25) are two key subroutines. The former decides how to perform expansion in a conflict-free manner. The latter ensures effective comparison and preservation of candidate partial solution paths.

3.1 Accessibility determination

When expanding the partial solution path $(\eta_{x,m,i}, \alpha_\eta)$ to the neighbouring temporal node $\eta_{y,n,j}$, the exit time interval β_x of zone x is determined as follows:

$$\beta_x = [ts(\alpha_x) + t_{m,n},\ \min\left(te(\alpha_x) + t_{m,n} + \Delta_{m,n}, te(w_{x,i})\right)],$$

where $t_{m,n} = d_{m,n}/v_{m,n}$ is the unimpeded traversal time determined by the distance $d_{m,n}$ and the unimpeded taxiing speed $v_{m,n}$ between nodes m and n. The functions $ts(\cdot)$ and $te(\cdot)$ retrieve the starting and ending time of a time interval, respectively. $t_{m,n} + \Delta_{m,n}$ is the maximum traversal time from m to n, where $\Delta_{m,n}$ represents the longest waiting time allowed. The arrival time interval α_y at the neighbouring zone y is then determined by calculating the overlap between the exit time interval β_x of x and the free time window $w_{y,j}$ associated with $\eta_{y,n,j}$. If a valid arrival time interval α_y exists, we refer to $\eta_{y,n,j}$ as an accessible temporal node of $(\eta_{x,m,i}, \alpha_\eta)$.

In line 11 of the proposed algorithm (see Fig. 4), all neighbouring temporal nodes of the partial solution path will be checked within function $\Gamma(\cdot)$ in the abovementioned way. A collection of the accessible neighbouring temporal nodes and the corresponding arrival time intervals will be returned. Each pair of the accessible temporal node and the corresponding arrival time interval forms a candidate partial solution path, which will be further compared with the existing partial solution paths in the close/open list based on the dominance rule described in the next section.

3.2 Dominance comparison

The purpose of dominance comparison is to identify inferior partial solution paths finishing at a specific temporal node. When a new (candidate) partial solution path is found during expansion, it will be compared with the existing partial solution paths in the close/open list and will be discarded if dominated by any of them. If the new partial solution path dominates any existing one in the open list, the latter will be removed from the open list. In the proposed algorithm, the following dominance rule is used.

Dominance rule: For partial solution paths (p, α_p) and (q, α_q) finishing at the same temporal node, we say (p, α_p) dominates (q, α_q) if $\alpha_p \supseteq \alpha_q$ and $\text{cost}(p, \alpha_p) < \text{cost}(q, \alpha_q)$.

For clarity, (p, α_p) dominates (q, α_q) is denoted as $(p, \alpha_p) \prec (q, \alpha_q)$. It is easy to verify that the dominance relation is transitive (i.e. if x dominates y and y dominates z, then x dominates z). Hence, a new partial solution path dominated by an existing one in the close list will never dominate any partial solution path in the

open list. Therefore, the flag is set to false in line 16 of the proposed algorithm, avoiding further comparison within the open list. Similarly, no further comparison is needed once a new partial solution path is dominated by an existing one in the open list (lines 25–28).

In light of the above discussions, the use of the arrival time interval ensures that all potential routing options from the selected partial solution path to the neighbouring temporal nodes are kept during expansion. Then, the dominance comparison between the newly formed partial solution paths and the existing ones in the close/open list ensures that all superior partial solution paths are preserved. Therefore, if conflict-free paths respecting the maximum traversal time constraint exist, the proposed algorithm will find the least-cost solution according to the property of A* [23].

3.3 Time complexity of the proposed algorithm

The computational cost of identifying all accessible neighbouring temporal nodes (line 11) is determined by the number of neighbouring zones and the number of free time windows in each neighbouring zone. In real-world applications, the number of neighbouring zones will be limited by a constant which represents the maximum number of outgoing branches of any taxiway intersection. The number of the free time windows for a neighbouring zone, denoted as n_w, is proportional to the number of aircraft with already assigned trajectories going through the neighbouring zone. Therefore, the worst case time complexity of identifying all accessible temporal nodes is $O(n_w)$ according to the analysis in [12]. During expansion, there could be more than one non-dominated partial solution path to the same temporal node. Let n_p be the maximum number of non-dominated partial solution paths to any temporal node. The dominance comparisons (lines 12–35) have a time complexity of $O(n_p \cdot n_w)$. Therefore, the time complexity of an entire iteration (lines 8–36) is $O(n_p \cdot n_w)$.

In the worst case, all the non-dominated partial solution paths to every intermediate temporal node will be expanded once, which requires $n_p \cdot n_w$ iterations in total. The overall time complexity of the proposed algorithm is therefore $O(n_p^2 \cdot n_w^2)$.

4 Experimental results

4.1 Dataset and settings

In this section, the proposed approach is tested with a dataset of 8664 aircraft landing on or taking off from the Shanghai Pudong Airport, which approximately corresponds to one week's operations in 2014. The runway and taxiway layouts for the investigated dataset are shown in Fig. 5, which include three parallel runways and 269 taxiway zones with 603 access/exit nodes. Some hotspots of taxiway congestion are also highlighted with filled ellipses in Fig. 5 according to the resulting taxi trajectories of the proposed approach. The information of runway usage and average traffic volume per day on each runway is presented in Table 2.

The original dataset provides the scheduled departing or arriving time of aircraft. The exact time at which each aircraft is ready for taxiing is manually assigned by randomly changing the scheduled departing or arriving time within a range of 1 h. The dataset with the manually assigned aircraft readiness time is used as input for time-based taxi trajectory planning. The impact of pushback and runway sequence for departing aircraft is not considered. The first-come-first-served strategy is used to determine the trajectory planning order of each aircraft. According to [8, 24], the unimpeded taxiing speeds are set to 8 m/s for straight movements and 5.14 m/s for turns, respectively. The minimum taxiing speed is assumed to be 5.14 m/s. When moving along the shortest route unimpededly, the average taxi time of an aircraft for the investigated dataset is 539.2 s.

All the algorithms in the experiment are implemented in C++ and run on a personal computer with 8 GB RAM and 2.5 GHz Intel Core i7 CPU.

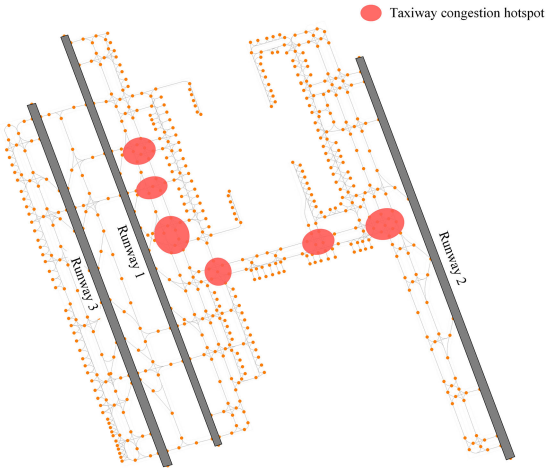

Fig. 5 *Runway and taxiway layout for the investigated dataset*

Table 2 Runway usage and traffic volume information

	Usage	Traffic volume per day
runway 1	takeoff	410.4
runway 2	takeoff and landing	518.9
runway 3	landing	308.4

4.2 Results

The improvement of ground movement efficiency through taxi time penalisation is first demonstrated by comparing between the proposed approach and a simplified variant of the proposed approach with the cost function being the completion time only. Then, a comprehensive comparison with the existing approach is presented to demonstrate the capability of the proposed approach to eliminate long waiting time using the maximum traversal time constraint and to handle the maximum traversal time constraint using the arrival time interval. The existing approach uses the quickest path algorithm to search for time-based taxi trajectories. The quickest path algorithm is parameter free. For fair comparison, the different approaches are tested under the same condition of free time windows for each zone, which are continuously updated after each assignment of the time-based taxi trajectory.

As mentioned in Section 1, the existing approach takes the (earliest) arrival time at the terminating node of a partial solution path as the cost. Although this scheme ensures that the solution with the earliest completion time can be found, overlong taxi time may appear in some cases. Therefore, in addition to the arrival time, the proposed approach also penalises the taxi time in the cost. To demonstrate the effect of such penalisation, a comparison of the completion time and taxi time obtained with and without taxi time penalisation in the cost function of the proposed algorithm is presented in Figs. 6*a* and *b*, respectively. Each data point represents the pair of the resulting completion times for an aircraft in Fig. 6*a* and taxi times in Fig. 6*b*. The horizontal axis corresponds to the value without taxi time penalisation. The vertical axis corresponds to the value with taxi time penalisation. According to Fig. 6*b*, 1294 aircraft finds time-based taxi trajectories with decreased taxi time through taxi time penalisation. The average reduction of taxi time for each aircraft is 7.5 s. However, the completion time increases in some cases, as shown in Fig. 6*a*. The average increase of completion time for each aircraft is 1.5 s.

Fig. 7*a* further shows the reduction of the taxi time and the increase of the completion time for individual aircraft which has completion time or taxi time changed through taxi time penalisation. The diagonal line in Fig. 7*a* means the reduction of the taxi time and the increase of the completion time are equal. According to the result in Fig. 7*a*, the extent of taxi time reduction is clearly larger than that of completion time increase for most aircraft; only for a few aircraft the reduction of the taxi time is similar to the increase of the completion time. This result demonstrates that the proposed solution algorithm finds the least-cost solution for all the aircraft. Moreover, as shown in Fig. 7*b*, for

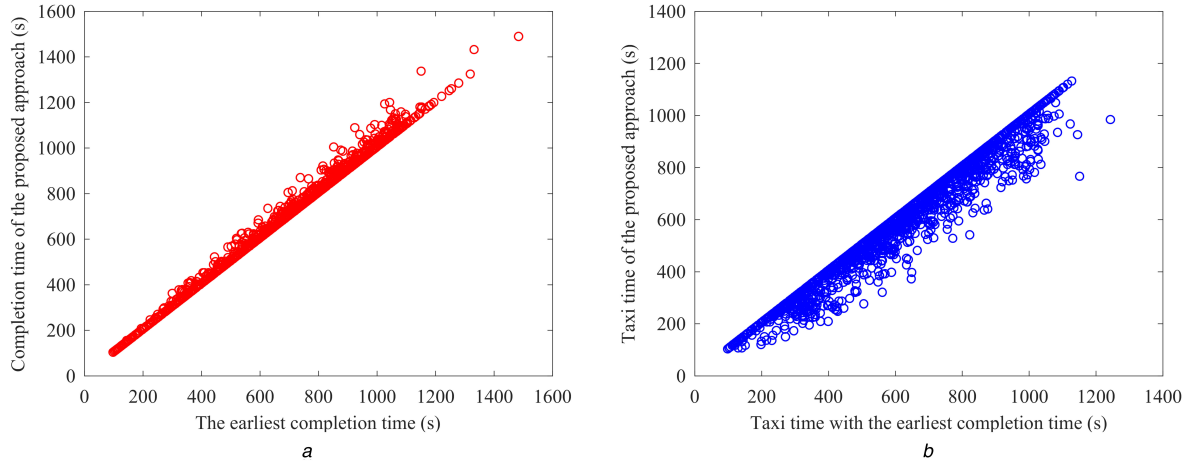

Fig. 6 *Comparison of the completion time and taxi time obtained with and without taxi time penalisation in the cost function of the proposed algorithm*
(a) Completion time for each aircraft, ***(b)*** Taxi time for each aircraft

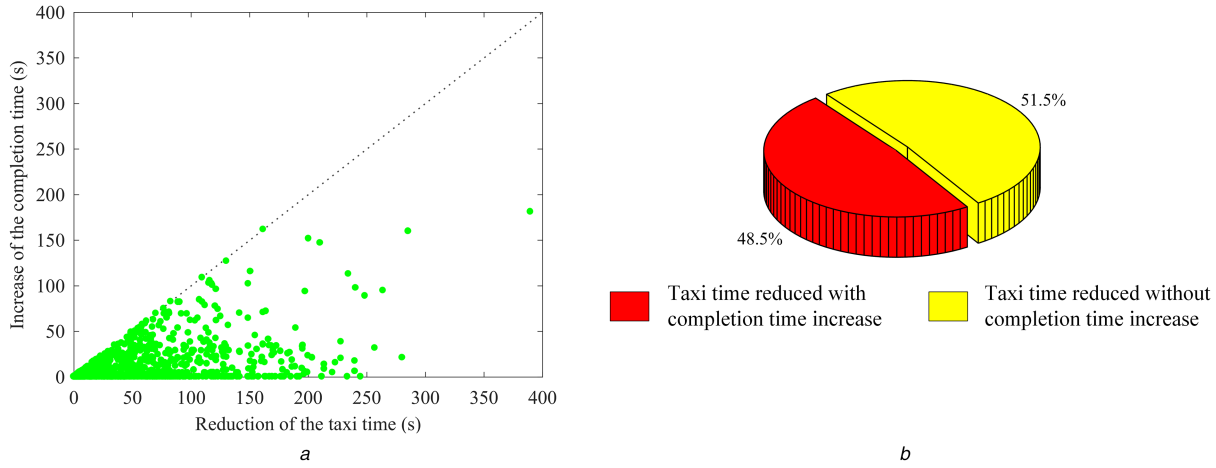

Fig. 7 *Comparison of the extent of changes for the taxi time and completion time through taxi time penalisation*
(a) Results for the aircraft with changed completion time or taxi time, ***(b)*** Percentage of aircraft for which the taxi time is reduced with or without completion time increase

51.5% of the aircraft (i.e. the data points on the horizontal axis of Fig. 7*a*), more efficient solutions can be found through taxi time penalisation without increasing the completion time. Therefore, the penalisation of the taxi time proves to be an effective scheme to reduce taxi time and improve the movement efficiency.

For the existing approach based on the quickest path algorithm mentioned in Section 1, the waiting time is not explicitly constrained during routing. As a result, long waiting time may exist in the solution. Fig. 8*a* shows the empirical cumulative distribution of the resulting waiting times in different zones using the proposed approach (i.e. SPPTW-MTTC) and the existing approach (i.e. quickest path problem with time windows (QPPTW)). For clarity, the results with no waiting are not included in Fig. 8*a*. It can be noticed that there are much more waiting situations when the existing approach is used. For about 20% of the illustrated situations, the waiting time for the existing approach is larger than 50 s; the largest waiting time is 567.6 s. As a result, there is an average waiting time of 18.2 s per aircraft for the existing approach, as shown in Table 3. When the proposed approach is used, the situations of waiting can be largely reduced. The average waiting time per aircraft is only 0.3 s, which is reduced by 98.5% compared with the result of the existing approach, while the largest waiting time is 25.3 s, reduced by 95.5% compared with the result of the existing approach. Moreover, the ground movement efficiency has also improved when the proposed approach is used. As shown in Fig. 8*b*, more aircrafts tend to have smaller taxi time. The reduction of the average taxi time is 3.9% (see Table 3). However, it is worth noting that the benefits on the waiting time and taxi time are achieved at the expense of a small increase of the completion time, as shown in the last column of Table 3. It should also be noted that due to the use of the arrival time interval, multiple (non-dominated) partial solution paths to the same temporal node could exist during the search of the proposed approach, which often leads to larger computational time than that of the quickest path algorithm. The average computational time for an aircraft using the proposed approach is 2.4 s, which is still possible for real-time application with increased computing power.

Fig. 9 illustrates the spatial trajectories and the corresponding time interval reservations of a typical example for which long waiting time during taxiing is required in the solution of the quickest path algorithm. In this instance, the aircraft moves from the gate to the runway. Different solutions are generated by the proposed approach and the quickest path algorithm. The corresponding spatial trajectories for the two solutions deviate at zone S159 and then fuse at zone S089, as shown in Fig. 9*a*. The time interval reservations for the two solutions are shown in Fig. 9*b*. The aircraft has the same arrival time at the destination runway in both solutions. However, for the solution generated by the quickest path algorithm, the aircraft starts taxiing immediately and arrives early at zone S078. Then, it has to wait within S078 for the clearance of the next zone for about 290 s. This might result in stop-and-go and lead to extra fuel consumption and emissions. In contrast, the solution generated by the proposed approach does not require long waiting time during taxiing. The aircraft can start taxiing later and move more fluently to the runway. In this case, the aircraft can be pushed back later from the gate, as it will reduce fuel consumption and emissions to wait at the gate with engines off.

As mentioned in Section 2.2, when the maximum traversal time constraint is directly considered in the quickest path algorithm, the latter often fails to find the best or even a feasible solution due to the loss of feasible routing options during expansion. A

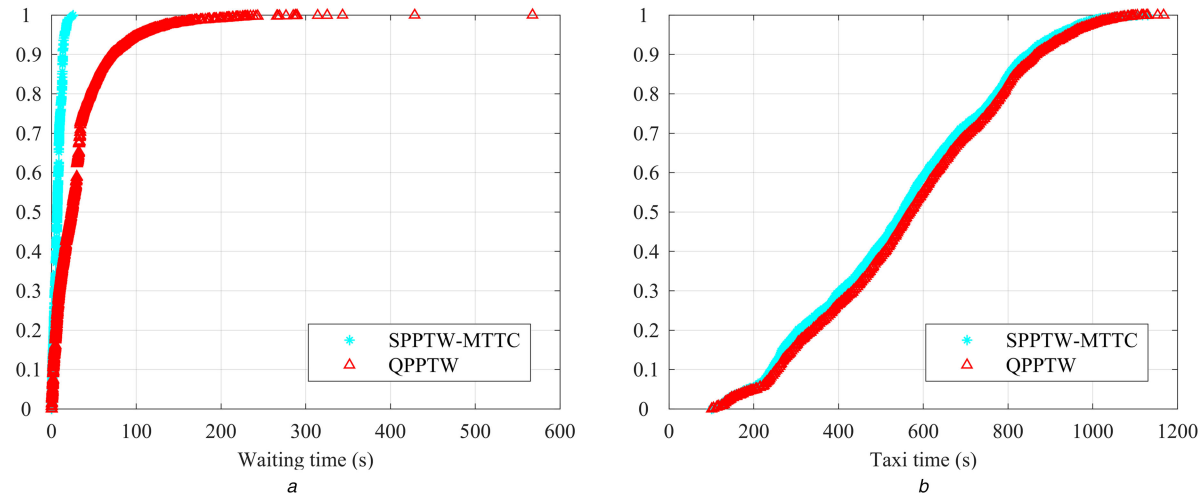

Fig. 8 *Empirical cumulative distributions of waiting time and taxi time for different approaches*
(a) Waiting time, ***(b)***Taxi time

Table 3 Comparison with the existing approach

	Average waiting time, s	Longest waiting time, s	Average taxi time, s	Average completion time, s
QPPTW	18.2	567.6	567.3	588.8
SPPTW-MTTC	0.3	25.3	545.3	590.6
reduction by SPPTW-MTTC	98.5%	95.5%	3.9%	−0.3%

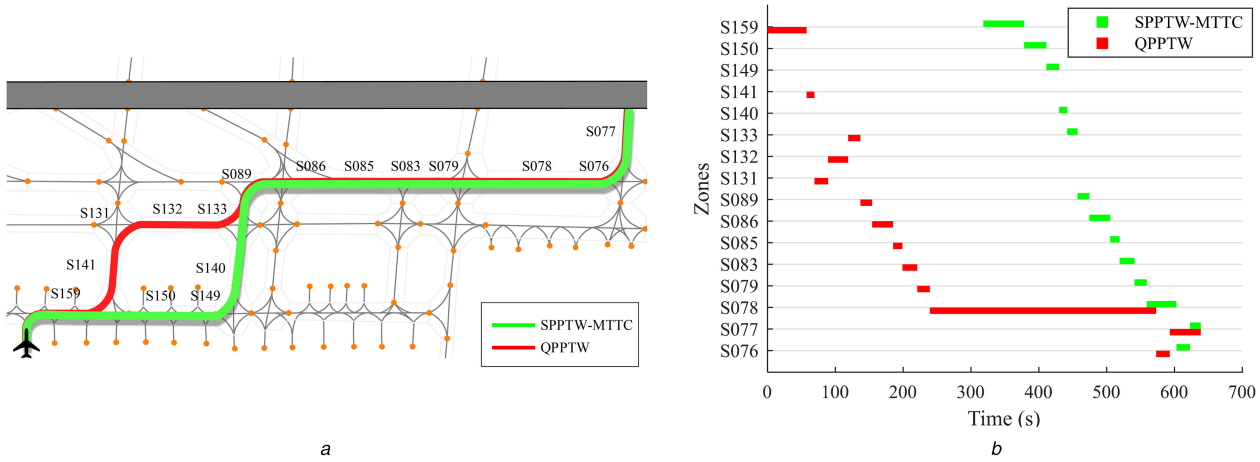

Fig. 9 *Illustration of the solutions for a typical example generated by different approaches*
(a) Spatial trajectories, ***(b)*** Reserved time intervals in different zones of the taxiway

comparison between the result of the quickest path algorithm considering the maximum traversal time constraint (i.e. QPPTW-MTTC) and the result of the proposed approach is presented in Table 4. For 342 of the 8664 aircraft in the dataset, no feasible solution can be found when the quickest path algorithm is directly used to solve the investigated problem. For the rest 8322 aircraft in the dataset, both approaches can find feasible solutions. The corresponding results are shown in the last three columns of Table 4. The longest waiting time, the average taxi time and the average completion time of the quickest path algorithm are all larger than those of the proposed approach. To illustrate the cause of the difference, a typical example for which the quickest path algorithm fails to find a feasible solution is presented in Fig. 10. In this example, the quickest path algorithm stops search after 171 expansions due to encountering an empty open list. All the partial solution paths identified during the search are illustrated in Fig. 10*a*. The rectangles represent the temporal nodes of the partial solution paths, each of which is depicted with a triple of the zone (S), access node (N) and free time window (W). The partial solution path in line with the solution found by the proposed approach is highlighted with solid lines in Fig. 10*a*. The expansion processes of both methods along the highlighted partial solution path are shown in Fig. 10*b*. Due to the use of the earliest arrival time and the existence of the maximum traversal time constraint, the resulting exit time intervals of the quickest path algorithm are usually much shorter than those of the proposed approach, which eventually makes the temporal nodes at zone S168 not accessible. It is worth noting that for some other aircraft of the dataset, suboptimal paths to the destination will be found using the quickest path algorithm, which leads to increased average taxi time and completion time as shown in Table 4.

5 Conclusions

To realise conflict-free, efficient and fluent airport ground movement, this paper proposes an improved approach for time-based taxi trajectory planning. The investigated problem is formulated as the SPPTW-MTTC. By penalising the taxi time in the cost, inefficient solutions with overlong taxi time can be avoided (often without increase of the completion time). By introducing the maximum traversal time constraint, waiting time during taxiing can be largely reduced, ensuring fluent ground movement of aircraft. An A*-based algorithm is proposed to solve the investigated problem. To handle the maximum traversal time constraint, the proposed algorithm utilises the arrival time interval for expansion, which keeps all the feasible arrival times instead of

Table 4 Comparison with the quickest path algorithm considering the maximum traversal time constraint

	Number of failed instances	Longest waiting time, s	Average taxi time, s	Average completion time, s
QPPTW-MTTC	342	73.0	588.9	613.3
SPPTW-MTTC	0	24.2	547.5	589.5
reduction by SPPTW-MTTC	100%	66.8%	7.0%	3.9%

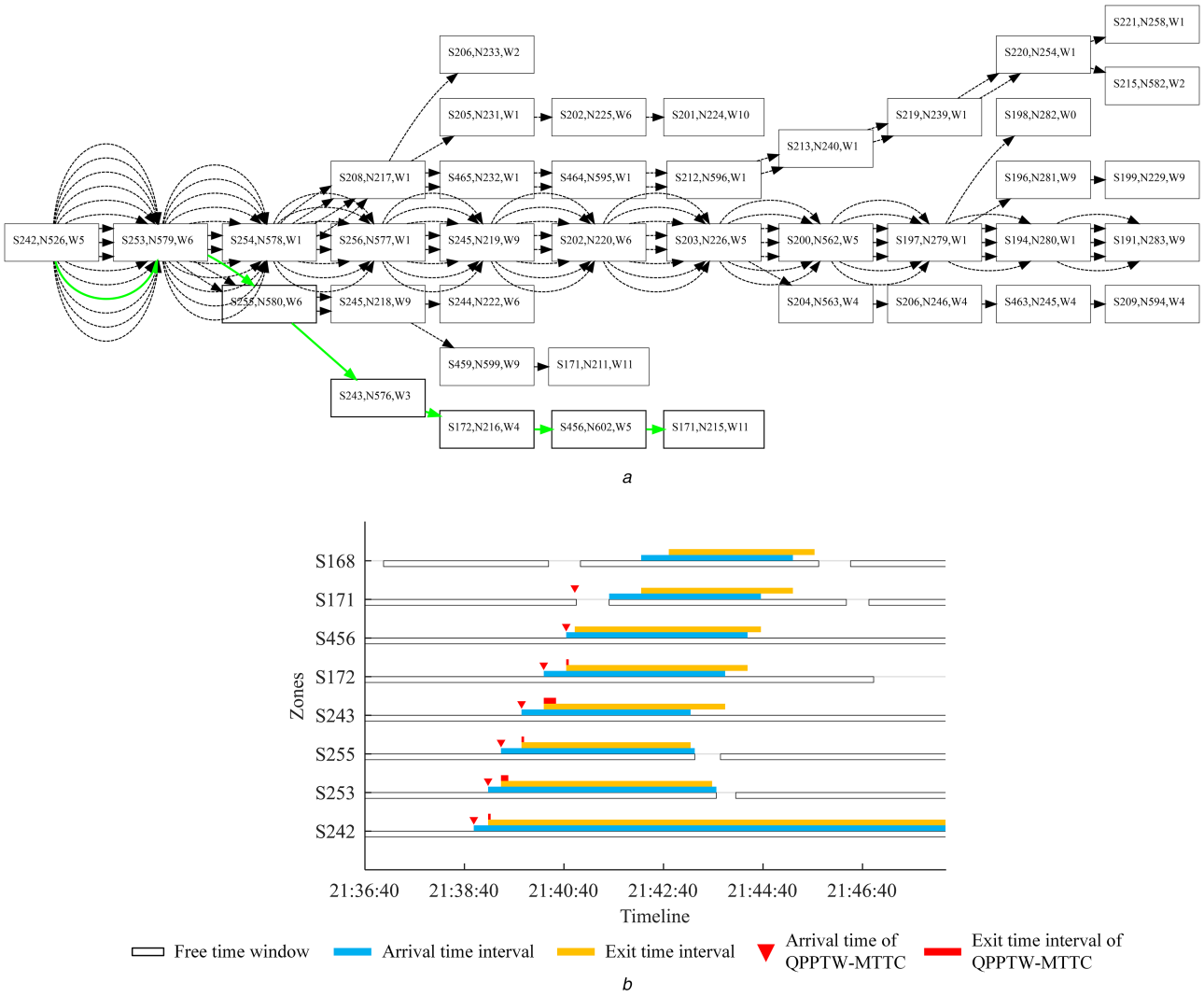

Fig. 10 *Illustration of a typical example for which the quickest path algorithm fails to find a feasible solution when the maximum traversal time constraint is considered*

(a) Snapshot of all the partial solution paths found by the quickest path algorithm, ***(b)*** Comparison between the expansion processes of the quickest path algorithm and the proposed approach

only the earliest one as in the existing approach. Experimental results demonstrate the advantages of the proposed approach in reducing the waiting time and taxi time, and validate the capability of the solution algorithm in handling the maximum traversal time constraint.

The results also indicate that the fluent time-based taxi trajectories found by the proposed approach enable later pushback of departing aircraft. This will be useful for target start-up approval time (TSAT) allocation, as the main goal of TSAT is to delay the start-up and pushback of aircraft to avoid long time of waiting in the runway holding area with engines running [25, 26]. However, to achieve the optimal TSAT allocation, other factors need to be considered at the same time, such as runway sequencing, pushback management, the interaction with arriving aircraft and the impact of uncertainties in relevant operations [27–30]. In addition to the connection with TSAT allocation, the improved time-based taxi trajectory planning capability of the proposed approach will also facilitate the more ambitious far-term trajectory-based surface operations, where full four-dimensional trajectories including detailed speed profiles will be generated for more precise guidance and control of airport ground movement [31, 32]. With detailed speed profiles, the required arrival time at each zone can be more realistically assigned, making it possible to quantify the potential reduction of fuel consumption and emissions with the improved time-based taxi trajectory planning capability. Moreover, as it is often necessary to dynamically adjust taxi trajectories in case of deviations or disruptions invalidating the planned trajectories, it is worthwhile to further enhance the computational performance of the proposed approach. Additionally, the application of the proposed solution algorithm to other problems such as conflict-free routing of automated guided vehicles is also worth investigation.

6 Acknowledgments

This work is supported by the Fundamental Research Funds for the Central Universities (NS2016061). The authors thank Dr. Michal Weiszer and Dr. Jun Chen especially for the comments and the help with the writing of this paper. The authors also thank the anonymous reviewers for the constructive feedback.

7 References

[1] Eurocontrol: '*Challenges of growth 2013: the effect of air traffic network congestion in 2035*' (Eurocontrol, Brussels, Belgium, 2013)

[2] Weiszer, M., Chen, J., Stewart, P.: 'A real-time active routing approach via a database for airport surface movement', *Transp. Res. C, Emerg. Technol.*, 2015, **58**, pp. 127–145

[3] ICAO: '*Advanced surface movement guidance and control systems (A-SMGCS) manual*' (International Civil Aviation Organization, Montreal, Canada, 2004)

[4] Okuniek, J.N., Gerdes, I., Jakobi, J.*, et al.*: 'A concept of operations for trajectory-based taxi operations'. 16th AIAA Aviation Technology, Integration, and Operations Conf., Washington, DC, USA, 2016

[5] Hooey, B.L., Cheng, V.H., Foyle, D.C.: 'A concept of operations for far-term surface trajectory-based operations (STBO)', 2014

[6] Foyle, D.C., Hooey, B.L., Bakowski, D.L.*, et al.*: 'Flight deck surface trajectory-based operations (STBO): simulation results and ConOps implications'. Proc. of the Ninth USA/Europe Air Traffic Management Research and Development Seminar (ATM2011), Berlin, Germany, 2011

[7] Weiszer, M., Chen, J., Locatelli, G.: 'An integrated optimisation approach to airport ground operations to foster sustainability in the aviation sector', *Appl. Energy*, 2015, **157**, pp. 567–582

[8] Clare, G.L., Richards, A.G.: 'Optimization of taxiway routing and runway scheduling', *IEEE Trans. Intell. Transp. Syst.*, 2011, **12**, (4), pp. 1000–1013

[9] Marín, Á.G.: 'Airport management: taxi planning', *Ann. Oper. Res.*, 2006, **143**, (1), pp. 191–202

[10] Lesire, C.: 'Iterative planning of airport ground movements'. Proc. of the 4th Int. Conf. on Research in Air Transportation (ICRAT 2010), Budapest, Hungary, 2010

[11] Ravizza, S., Atkin, J.A., Burke, E.K.: 'A more realistic approach for airport ground movement optimisation with stand holding', *J. Sched.*, 2014, **17**, (5), pp. 507–520

[12] Zhang, T., Ding, M., Wang, B.*, et al.*: 'Conflict-free time-based trajectory planning for aircraft taxi automation with refined taxiway modeling', *J. Adv. Transp.*, 2016, **50**, (3), pp. 326–347

[13] Nikoleris, T., Gupta, G., Kistler, M.: 'Detailed estimation of fuel consumption and emissions during aircraft taxi operations at Dallas/Fort Worth International Airport', *Transp. Res. D, Transp. Environ.*, 2011, **16**, (4), pp. 302–308

[14] Pugliese, L.D.P., Guerriero, F.: 'A survey of resource constrained shortest path problems: exact solution approaches', *Networks*, 2013, **62**, (3), pp. 183–200

[15] Desaulniers, G., Villeneuve, D.: 'The shortest path problem with time windows and linear waiting costs', *Transp. Sci.*, 2000, **34**, (3), pp. 312–319

[16] Solomon, M.M., Desrosiers, J.: 'Survey paper–time window constrained routing and scheduling problems', *Transp. Sci.*, 1988, **22**, (1), pp. 1–13

[17] Ioachim, I., Gelinas, S., Soumis, F.*, et al.*: 'A dynamic programming algorithm for the shortest path problem with time windows and linear node costs', *Networks*, 1998, **31**, (3), pp. 193–204

[18] Olmi, R.: '*Traffic management of automated guided vehicles in flexible manufacturing systems*' (Università degli Studi di Ferrara, Ferrara, Italy, 2011)

[19] Fanti, M.P.: 'Event-based controller to avoid deadlock and collisions in zone-control AGVS', *Int. J. Prod. Res.*, 2002, **40**, (6), pp. 1453–1478

[20] Stahlbock, R., Voβ, S.: 'Vehicle routing problems and container terminal operations–an update of research', in Golden, B.L., Raghavan, S., Wasil, E. A. (Eds): '*The vehicle routing problem: latest advances and new challenges*' (Springer, Boston, MA, USA, 2008), pp. 551–589

[21] Smolic-Rocak, N., Bogdan, S., Kovacic, Z.*, et al.*: 'Time windows based dynamic routing in multi-AGV systems', *IEEE Trans. Autom. Sci. Eng.*, 2010, **7**, (1), pp. 151–155

[22] ter Mors, A., Witteveen, C., Zutt, J.*, et al.*: 'Context-aware route planning', in Dix, J., Witteveen, C. (Eds): '*Multiagent system technologies*' (Springer, Berlin Heidelberg, 2010), pp. 138–149

[23] Hart, P.E., Nilsson, N.J., Raphael, B.: 'A formal basis for the heuristic determination of minimum cost paths', *IEEE Trans. Syst. Sci. Cybern.*, 1968, **4**, (2), pp. 100–107

[24] Chen, J., Weiszer, M., Locatelli, G.*, et al.*: 'Toward a more realistic, cost effective and greener ground movement through active routing: a multi-objective shortest path approach', *IEEE Trans. Intell. Transp. Syst.*, 2016, **17**, (12), pp. 3524–3540

[25] Atkin, J.D., Burke, E., Greenwood, J.: 'TSAT allocation at London heathrow: the relationship between slot compliance, throughput and equity', *Public Transp.*, 2010, **2**, (3), pp. 173–198

[26] Okuniek, N., Sparenberg, L.: 'Opportunities and challenges when implementing trajectory-based taxi operations at European and U.S. CDM airports'. 2017 IEEE/AIAA 36th Digital Avionics Systems Conf. (DASC), St. Petersburg, FL, USA, 2017

[27] Benlic, U., Brownlee, A.E., Burke, E.K.: 'Heuristic search for the coupled runway sequencing and taxiway routing problem', *Transp. Res. C, Emerg. Technol.*, 2016, **71**, pp. 333–355

[28] Gerdes, I., Schaper, M.: 'Management of time based taxi trajectories coupling departure and surface management systems'. 11th ATM Seminar, Lisbon, Portugal, 2015

[29] Mori, R.: 'Optimal pushback time with existing uncertainties at busy airport'. Proc. of 29th Congress of the ICAS, St. Petersburg, 2014

[30] Mori, R.: 'Development of a pushback time assignment algorithm considering uncertainty', *J. Air Transp.*, 2017, **25**, (2), pp. 51–60

[31] Cheng, V., Sweriduk, G.D.: 'Trajectory design for aircraft taxi automation to benefit trajectory-based operations'. Proc. of the 7th Asian Control Conf., Hong Kong, China, 2009

[32] Chen, J., Weiszer, M., Stewart, P.*, et al.*: 'Toward a more realistic, cost effective and greener ground movement through active routing: part 1-optimal speed profile generation', *IEEE Trans. Intell. Transp. Syst.*, 2016, **17**, (5), pp. 1196–1209