Solving the Time-Dependent Shortest Path Problem Using Super-Optimal Wind

Adam Schienle

1 Introduction

With air travel steadily on the rise and the increased fuel burn associated to it, it is ever more important that aircraft fly efficient routes. Planning such routes is a fundamental process of flying: commonly, a route is planned a few hours before the flight, focussing on key factors such as overfly costs and fuel burn. According to the Air Transport Action Group [1], around 1.5 billion barrels of fuel are burnt every year, corresponding to 93.75 billion USD [6]. A decrease of just 0.25% would add up to 234.375 million USD. There is also a visible impact for airlines: Lufthansa's total fuel consumption in 2016 amounted to 9 055 550 tons [7]. Decreasing this by 0.25% leads to 22 639 tons less fuel being burnt, or savings of almost 11.67 million USD per year. In terms of CO_2, this is equivalent to a reduction of more than 70 tons per year [7].

The need for efficient routes gives rise to the Flight Planning Problem (FPP), which is the problem of finding a minimum cost trajectory between two airports on the Airway Network, a directed graph. In general, the objective function consists of several summands, such as fuel costs, overfly costs and crew costs. In this paper, however, we shall concentrate on minimising the fuel costs. We further assume that aircraft fly levelly on a given altitude and neither climb nor descend. In this setting, fuel consumption is equivalent to flight time, which reduces FPP to the *Horizontal Flight Planning Problem* (HFPP). Since winds have a strong impact on flight time and because of the time-dependency of the weather, we can model HFPP as a Time-Dependent Shortest Path Problem (TDSPP).

TDSPP has been extensively studied in the literature, with particular emphasis on road networks. Dijkstra's algorithm yields an optimal solution in polynomial time; however, for large networks, several speedup techniques have been developed,

A. Schienle (✉)
Zuse Institute Berlin, Takustraße 7, 14195 Berlin, Germany
e-mail: schienle@zib.de

© Springer International Publishing AG, part of Springer Nature 2018

N. Kliewer et al. (eds.), *Operations Research Proceedings 2017*, Operations Research Proceedings, https://doi.org/10.1007/978-3-319-89920-6_1

allowing to curb runtimes by several orders of magnitude with respect to Dijkstra's algorithm [2]. Most of them rely on a preprocessing phase, in which either some shortest paths or other auxiliary data is precomputed and stored to speed up the query. For a comprehensive survey, see [2].

Throughout this paper, a *weighted graph* will always refer to a pair (G, T), consisting of the actual (directed) graph G and a possibly time-dependent weight function $T\colon A \times [0,\infty) \to [0,\infty)$, mapping an arc $a \in A$ and a time $\tau \in [0,\infty)$ to the travel time $T(a,\tau)$ on a.

The ground distance $d_G(a)$ of an arc $a \in A$ on the Earth's surface is constant, and we assume that aircraft fly with constant air speed[1] v_A. In contrast, the *ground speed* $v_G(a,\tau)$ of an aircraft is dependent on the prevailing wind conditions on the arc and given by the formula

$$v_G(a,\tau) = \sqrt{v_A^2 - w_C(a,\tau)^2} + w_T(a,\tau) \quad \forall a \in A, \tau \in [t_0, t_r], \tag{1}$$

where $w_C(a,\cdot)$ and $w_T(a,\cdot)$ are the crosswind and trackwind components of the wind vector, i.e., the components perpendicular and parallel to the current flight direction. Ground speed and ground distance are linked via the relation

$$T(a,\tau) = \frac{d_G(a)}{v_G(a,\tau)}. \tag{2}$$

2 Super-Optimal Wind

We are looking to solve the TDSPP model of HFPP to optimality by using an appropriate shortest path algorithm. A natural choice would be Dijkstra's algorithm; in practice, however, the time to plan a flight is limited and for the most part, this process takes place shortly before the aircraft departs. In particular, this means that query times should be as short as possible. In this paper, we restrict ourselves to the discussion of the A* algorithm, introduced in [5]. For an overview of other algorithms and their applicability to HFPP, see [3].

The intricacy with A* is to find a suitable potential function $\pi_t\colon V \to [0,\infty)$, which for every $v \in V$ underestimates the cost of a shortest v-t-path in (G, T). We define the reduced cost of an arc $(u, v) \in A$ at time τ as

$$T'((u,v),\tau) = T((u,v),\tau) - \pi_t(u) + \pi_t(v), \tag{3}$$

and call π_t *feasible on* (G, T) if for every arc $(u, v) \in A$ and for every $\tau \geq 0$, we have $T'((u,v),\tau) \geq 0$. If π_t is feasible, running A* is equivalent to running Dijkstra's algorithm on G using the reduced costs.

[1] Speed relative to the surrounding air mass.

To obtain a feasible potential function, we have to find a lower bound for the travel time on the arcs. To this end, we introduce the concept of *Super-Optimal Wind* to underestimate the travel time. While it is possible to minimise the travel time function directly, this takes too long for practical purposes. Furthermore, it requires knowledge of the airspeed in advance, as opposed to constructing the Super-Optimal Wind vector.

We assume that weather is given for a finite set of times $\{t_0, t_1, \ldots, t_r\}$, and between the t_i, the weather data is interpolated to obtain the wind vector $w(a, \tau)$. Let $t_0 = \tau_0 < \tau_1 < \cdots < \tau_n = t_r$ be a discretisation of $[t_0, t_r]$ such that $\tau_i - \tau_{i-1} = \triangle$ for some $\triangle > 0$ and for all $i = 1, \ldots, n$. To ensure that for every $i \in \{0, \ldots, n-1\}$ we always find a $j \in \{0, \ldots, r-1\}$ such that $[\tau_i, \tau_{i+1}] \subset [t_j, t_{j+1}]$, we require that $r \mid n$. We then define for $i = 1, \ldots, n$

$$\underline{w}_C^{(i)}(a) = \min_{\tau \in [\tau_{i-1}, \tau_i]} |w_C(a, \tau)| \quad \text{and} \quad \overline{w}_T^{(i)}(a) = \max_{\tau \in [\tau_{i-1}, \tau_i]} w_T(a, \tau),$$

which are the minimum crosswind and maximum trackwind on each discretisation step. The vector defined through its cross- and trackwind components

$$w_{\text{s-opt}}^{(i)}(a) = (\underline{w}_C^{(i)}(a), \overline{w}_T^{(i)}(a))$$

is called *Super-Optimal Wind* vector, and is used to overestimate the ground speed (note that by (2), this is equivalent to underestimating the travel time). We define

$$\overline{v}_G^{(i)}(a) = \sqrt{v_A^2 - \underline{w}_C^{(i)}(a)^2} + \overline{w}_T^{(i)}(a),$$

and let $\overline{v}_G(a) := \max_{i \in \{1, \ldots, n\}} \overline{v}_G^{(i)}(a)$. It is easy to prove the following lemma:

Lemma 1 *The inequality $v_G(a, \tau) \leq \overline{v}_G(a)$ holds for all $\tau \in [t_0, t_r]$.*

Note that in particular, if $v_G^*(a) = \max_{\tau \in [t_0, t_r]} v_G(a, \tau)$ denotes the maximum ground speed in $[t_0, t_r]$, we also have

$$\overline{v}_G(a) \geq v_G^*(a). \tag{4}$$

Define $r_a^* := \max_{\tau \in [t_0, t_r]} \sqrt{w_C(a, \tau)^2 + w_T(a, \tau)^2}$, the maximum overall wind speed on $a \in A$. Under the condition that $v_A \geq 2r_a^*$, which in practice is always the case, we obtain

Theorem 1 *Suppose $v_A \geq 2r_a^*$. Then there exists a constant $C > 0$ such that*

$$0 \leq \overline{v}_G(a) - v_G^*(a) \leq C\triangle.$$

The first inequality follows directly from (4), and the proof for the second inequality can be found in [3]. Analogous to the ground speed, we define

$$\underline{T}(a) = \min_{i\in\{1,\dots,n\}} \underline{T}^{(i)}(a) := \min_{i\in\{1,\dots,n\}} \frac{d_G(a)}{\overline{v}_G^{(i)}}.$$

Letting $T_a^* = \min_{\tau\in[t_0,t_r]} T(a,\tau)$ and following (2), one readily obtains

Corollary 1 *Suppose $v_A \geq 2r_a^*$. Then there exists a constant $C' > 0$ such that for any arc $a \in A$, we have*

$$0 \leq T_a^* - \underline{T}(a) \leq C'\triangle.$$

In particular, $\underline{T}(a)$ underestimates the travel time needed to traverse an arc, and the error is bounded linearly in the discretisation step.

2.1 The Super-Optimal Wind Potential Function

For the A* algorithm, we seek to find a good and feasible potential function. For HFPP, we can exploit the fact that in our application, there is a small number of possible target nodes (corresponding to airports). Since our objective in HFPP is to minimise travel time, we construct the weighted graph $(G, \underline{T})$, where $\underline{T}\colon A \to [0,\infty)$ maps an arc $a \in A$ to the underestimated travel time $\underline{T}(a)$ obtained through the Super-Optimal Wind computation, i.e., $\underline{T}(a) \leq T(a,\tau)$ for all $\tau \in [t_0, t_r]$ and all arcs $a \in A$. Note that $(G, \underline{T})$ is a weighted graph with static arc weights, and we can without effort compute an all-to-one shortest path tree for each target node t. We then define a potential function for HFPP as

$$\pi_t(v) = \min\left\{\textstyle\sum_{a\in P} \underline{T}(a)\colon P \text{ is a } (v,t)-\text{path}\right\}.$$

Note that this is equivalent to running the ALT-Algorithm [4] with the target node as the only landmark.

Theorem 2 *The following two statements hold:*

(i) $\pi_t(\cdot)$ *is feasible in* $(G, \underline{T})$.
(ii) $\pi_t(\cdot)$ *is feasible in* (G, T).

For details on the proof, see [8]. In particular, Theorem 2 yields that running the A* algorithm on (G, T) is equivalent to running Dijkstra's algorithm on the reduced cost graph (G, T') obtained from (3), and A* visits at most as many nodes as Dijkstra's algorithm.

2.2 Validation of Super-Optimal Wind

Theorem 1 and Corollary 1 state that the absolute error of the overestimated ground speed with respect to the optimum ground speed is bounded linearly in the dis-

Table 1 Errors and runtimes of Super-Optimal Wind computation

Altitude (ft)	Segments (#)	Av. error (%)	Max. error (%)	Computation time (s)
37000	344936	0.041	5.263	2.50
34000	344920	0.045	5.882	1.59
31000	338567	0.045	8.333	2.52

cretisation step. To assess the quality of the travel time underestimation with Super-Optimal Wind computationally, we ran it on several real-world instances (cf. [3]), each instance using 28 threads.

As our weather prognoses are given at times t_i all spaced three hours apart, a natural choice for the discretisation step is $t_{i+1} - t_i = \tau_{i+1} - \tau_i = \triangle = 3h$. We found this choice to already yield excellent results, as shown in Table 1, which contains the average and maximum values of the relative error $\rho(a) = \frac{T(a)-T_a^*}{T_a^*} \forall a \in A$. The results show that the Super-Optimal Wind is an excellent underestimator in practice, and can be computed fast.

3 A Case Study

In the following, we investigate the effect of wind on a route. In particular, we consider a flight between Taipei-Taoyuan (TPE) and New York-John F. Kennedy (JFK). We use weather data from the 25th April 2017, starting the route on the same day at 0300 UTC. We assume an aircraft flying at 37 000ft ($\approx$11 277 m).

Often, routes lie close to the geodesic, but if aircraft can take advantage of strong tailwinds, they commonly divert to areas with more favourable winds. In Fig. 1, we observe that the search space for A* is doughnut-shaped, which is due to the fact that on that day, there was an unusually strong jetstream on the Northern Pacific, rendering the Pacific route shown in green more efficient than the polar route (red), which would seem a more natural choice. When one compares the ground distances of the northerly route to the Pacific route, one finds the red route to be almost 1880 km shorter than the green route – but considering wind, the green route is 131 s faster than the red route, or roughly 0.26% of the total travel time. As this translates directly to fuel burn, it makes sense to favour the seemingly longer Pacific route over the polar route.

In Fig. 1, we also observe that A* visits significantly fewer arcs than Dijkstra's algorithm. This also impacts the runtime: between TPE and JFK, A* yields a speedup factor of 11 over Dijkstra's algorithm. For a more detailed discussion on the speedup of A* over many instances, we refer the reader to [8].

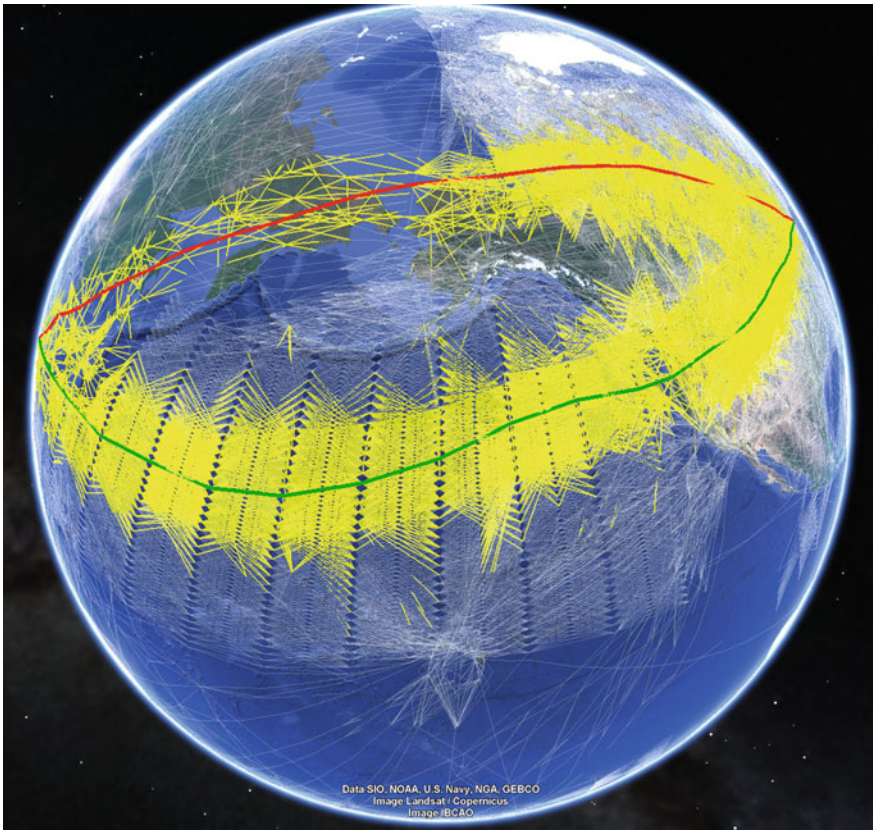

Fig. 1 Search spaces for Dijkstra's algorithm (white) and A* (yellow) between TPE and JFK. The route closest to the geodesic is marked red, the shortest route shown in green (Map data: Google, Landsat/Copernicus/IBCAO)

References

1. Air Transport Action Group (ATAG). (2017). Facts and figures. http://www.atag.org/facts-and-figures.html. Retrived 07 Aug 2017.
2. Bast, H., Delling, D., Goldberg, A., Müller-Hannemann, M., Pajor, T., Sanders, P., Wagner, D., & Werneck, R. F. (2015). Route planning in transportation networks. Technical report, Microsoft Research. Updated version of the technical report MSR-TR-2014-4.
3. Blanco, M., Borndörfer, R., Hoang, N. D., Kaier, A., Schienle, A., Schlechte, T., & Sclobach, S. (2016). Solving time dependent shortest path problems on airway networks using super-optimal wind. *16th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2016)*, *54*. (epub ahead of print).
4. Goldberg, A. V., & Harrelson, C. (2004). Computing the shortest path: A* search meets graph theory. Technical Report MSR-TR-2004-24, Microsoft Research, Vancouver, Canada, July 2004.
5. Hart, P. E., Nilsson, N. J., & Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, *4*, 100–107.

6. International Air Transport Association. IATA Price Analysis. (2017). https://www.iata.org/publications/economics/fuel-monitor/Pages/price-analysis.aspx. Retrived 07 Aug 2017.
7. Lufthansa Group. (2017). Balance – Nachhaltigkeitsbericht der Lufthansa Group 2016. https://www.lufthansagroup.com/fileadmin/downloads/de/verantwortung/LH-Nachhaltigkeitsbericht-2017.pdf. Retrived 07 Aug 2017.
8. Schienle, A. (2016). Shortest paths on airway networks. Master's thesis, Freie Universitt Berlin.