

An A* Algorithm for Flight Planning Based on Idealized Vertical Profiles

Marco Blanco¹ ✉

Lufthansa Systems GmbH & Co. KG, Raunheim, Germany
Zuse Institute, Berlin, Germany

Ralf Borndörfer ✉

Zuse Institute, Berlin, Germany

Pedro Maristany de las Casas ✉

Zuse Institute, Berlin, Germany

Abstract

The Flight Planning Problem is to find a minimum fuel trajectory between two airports in a 3D airway network under consideration of the wind. We show that this problem is NP-hard, even in its most basic version. We then present a novel A* heuristic, whose potential function is derived from an idealized vertical profile over the remaining flight distance. This potential is, under rather general assumptions, both admissible and consistent and it can be computed efficiently. The method outperforms the state-of-the-art heuristic on real-life instances.

2012 ACM Subject Classification Mathematics of computing → Graph algorithms; Mathematics of computing → Paths and connectivity problems; Mathematics of computing → Combinatorial optimization; Mathematics of computing → Discrete optimization

Keywords and phrases shortest path problem, a-star algorithm, flight trajectory optimization, flight planning, heuristics

Digital Object Identifier 10.4230/OASICS.ATMOS.2022.1

1 Introduction

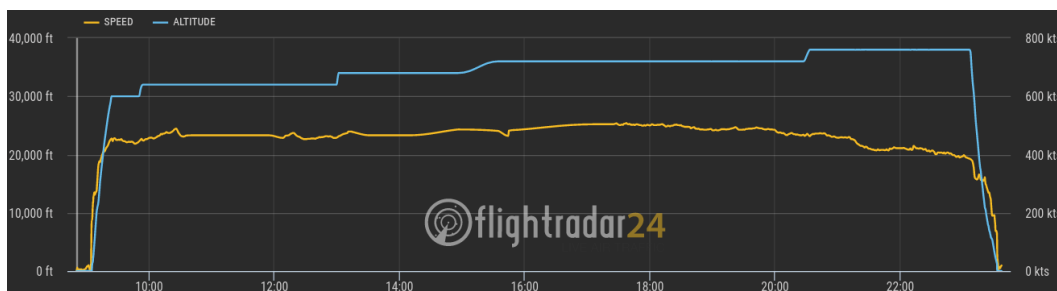
The Flight Planning Problem (FPP) seeks to compute a flight trajectory between two airports that minimizes fuel consumption. In this paper we consider a basic version subject to weather conditions, aircraft performance, and an airway network.

Weather forecasts for flight planning are usually provided on a 4D grid, which specifies a wind vector for each coordinate, altitude, and time. These data can be interpolated on all 4 dimensions to obtain a single wind vector acting on each flight segment, see [4] for more details. For the purposes of this paper, it suffices to think of wind as a function that maps time to an effective *air distance* that is needed to traverse a given segment.

Aircraft performance specifies how the state of the aircraft changes as a function of the flight phase and various parameters. Namely, for the current weight, the current altitude, the target distance, and the local wind condition, the performance function computes the weight after a cruise, climb, or descent phase along a flight segment. For cruise phases, the influence of the wind can be subsumed into the distance to the cruise target. The fuel consumption is then the weight difference, while the cruise time can be easily calculated from the speed (which we assume here to be constant) and the distance. For climbs and descents, distance, consumption, and time are more difficult to compute, since they depend on the vertical angle, which in turn depends on the aircraft weight. In accordance with the literature, see, e.g., [15], we assume that, *ceteris paribus*, a higher weight results in a higher consumption,

¹ Corresponding author





■ **Figure 1** Vertical profile on a flight from Amsterdam to Santiago. The blue graph represents the aircraft's altitude over time. Image obtained from FlightRadar24.com on the 9th of July 2021.

that cruising is in general more efficient as the altitude increases, until an *optimal cruise altitude* is reached, and that a smaller aircraft weight results in a steeper climb/descent angle. Moreover, a climb between two given altitudes might not be possible if the aircraft weight exceeds a certain threshold. These properties produce a vertical profile shape that is known as *step-climb*. Namely, to fly efficiently, an aircraft climbs from the departure airport to the highest altitude reachable in a single climb. Then, it cruises on this altitude until it has burned enough fuel and is light enough to climb further. This is repeated until the optimal cruise level is reached. Finally, the aircraft needs to start the final descent. See Figure 1 for a real-life example.

The *Airway Network* is a directed graph with a three-dimensional embedding covering the airspace around the Earth. It arises from a set of waypoints (2D coordinates) connected by airway segments (straight lines) on a set of discrete flight levels (altitudes); airports are modeled as a particular type of waypoints. The horizontal profile of a legal flight trajectory must consist of a contiguous sequence of airway segments connecting the two airports. Vertically, cruise phases are only allowed on one of the flight levels, while a climb or descent phase must be started at a waypoint (it cannot be started from the interior of a segment, but can and usually does end in the interior).^b

The literature on the FPP varies greatly in the extent and depth at which the technical aspects of the problem are treated. [6] is an extensive work that goes into great detail. To the best of our knowledge, it presented the first dynamic programming algorithm which runs on a 3D graph. [9] uses a dynamic programming approach to minimize fuel consumption during the cruise phase for a fixed horizontal route. [18] computes a trajectory on a search space where the horizontal route is not restricted by waypoints and segments by splitting the problem into a horizontal and a vertical component, which are solved sequentially using dynamic programming approaches. [12] gives a realistic and detailed survey of the most relevant cost components and restrictions, as well as an excellent review of previous work. The authors sketch some possible ways of solving the problem, such as decomposition into horizontal and vertical optimization (2D+2D) or 4D search, all on a high level.

The FPP can be seen as a special route planning problem. In this domain, A* algorithms achieve excellent running times. The main idea of these algorithms is to guide a Dijkstra-like search towards the potential function which is built in a preprocessing stage. Potential functions map the nodes of the graph onto estimates that bound the cost of a shortest path

^b In practice, the final descent to the destination airport is an exception, since it can be started everywhere. In this work, we ignore this exception for the sake of simplicity and without a significant impact on the results.

to the target node from below. There is extensive literature that focuses on the task of computing such heuristic functions and thus designing A* algorithms for routing on (time dependent) road networks [7, 1, 22], routing for electric vehicles [2], or in multiobjective scenarios [16, 17].

A* algorithms have also been considered for the FPP. A series of papers by a group from the University of Southern Denmark studies the problem in a very realistic way, presents new algorithms, and tests them on real-world data: [13] optimizes the vertical profile of a given 2D-route by a simple A* algorithm whose lower bounds are calculated from the minimum consumption on each arc. It also shows that even though the FIFO property^c does not hold due to the unpredictable nature of weather, the Dijkstra algorithm in practice nearly always finds an optimal solution. [14] provides an algorithm to solve flight planning under consideration of traffic restrictions, for the case of a constant flight altitude. It is based on storing multiple labels per node/altitude pair. [11] discusses the *free-route* case, where the flight area is not limited by an Airway Network. The most relevant article for our work is [15]. It considers the same setting as ours plus flight restrictions, which are handled by the algorithm from [15]. The main contribution of the paper are two variants of an A*-type algorithm on a three-dimensional graph, called *All Descents* and *Single Descent*. The first one uses very conservative lower bounds on the arc lengths, which are used for a backwards search that defines the potentials; these are both admissible and consistent under the FIFO assumption. The Single Descent algorithm calculates the potentials partially before the start of the search and partially during the expansion of the labels. It is much faster than the All Descents variant, but the potentials are neither admissible nor consistent. However, the computational results testify a very small error on real-world instances. We will use the Single Descent algorithm as a benchmark in our computations.

This paper builds on our previous work [4], which investigates the FPP restricted to a constant altitude. It presents a method for calculating lower bounds on travel-time on arcs by using a concept called *super-optimal wind*. This in turn is used to construct potentials for an A* algorithm.

While the addition of altitudes requires a much more sophisticated approach, the distance underestimation techniques of [4] are a critical component of our new algorithm. [3] presents a heuristic that handles complex overflight costs by reducing them to classical costs on arcs by solving a Linear Program. This approach can be trivially combined with most others, including the one we present. Finally, [21] also investigates a horizontal variant of the FPP, which considers both weather and overflight costs. It introduces efficient pruning techniques that reduce the graph before the start of the search algorithm. These techniques can also be easily incorporated in a step preceding an A* search.

The FPP is a time-dependent shortest path problem on the Airway Network subject to weather conditions and aircraft performance. We show that it is NP hard, a basic fact that, as far as we know, has hitherto not been noted. As such, the FPP cannot be solved by a Dijkstra-type label setting algorithm. However, as this approach is efficient and produces excellent results, it is commonly used in practice and also as our benchmark in this paper. In this vein, we present an A*-algorithm that improves on Dijkstra's algorithm. Its potential function is the cost of an idealized vertical trajectory over a lower bound of the total remaining flight distance. The construction of this idealized trajectory is based on the above mentioned assumptions about optimal vertical profiles. We show that it can be calculated efficiently on-the-fly, during the label expansion, and further sped-up by a pre-calculation of parts of

^c The FIFO property states that early arrival is always beneficial.

the climb phase that depends only on the aircraft type, i.e., the aircraft performance function. This leads to a fast algorithm, which is essential in order to account for the latest weather forecast and the newest flight restrictions. On a set of real-world instances, our approach is on average 7-10 times faster than Dijkstra's algorithm and 30-40% faster than the Single Descent algorithm of [15].

The paper is structured as follows. In Section 2 we present a mathematical model of the Flight Planning Problem (FPP) that generalizes the Time-Dependent Shortest-Path-Problem (TDSPP). We also present the first NP-hardness proof for the FPP; this proof extends to a large family of TDSPPs. Section 3 presents an A*-algorithm for the FPP. Its potential function computes the cost of an idealized vertical profile over a lower bound of the total remaining flight distance. Under certain assumptions on aircraft performance, this potential is admissible and consistent, and it can be computed efficiently. In Section 4, we compare our implementations of the new A*-algorithm, Dijkstra's algorithm, and the Single Descent algorithm. The results show that the potential calculation pays off by drastically reducing the number of expanded labels and the runtime. They also show that our consistency assumptions are satisfied to a reasonable degree.

2 The Flight Planning Problem

We represent the Airway Network by a directed graph $G = (V, A)$. Each waypoint gives rise to multiple nodes, corresponding to the different flight levels H ; denote by $h(v) \in H$ the flight level of node v . We assume that the departure and the arrival airport are located not on the ground but on the lowest flight level h_0 ^d. Likewise, each segment gives rise to multiple arcs: One cruise arc for each flight level and one climb or descent arc for each combination of two flight levels. We assume that the highest flight level is the optimal cruise level, since it does not make sense to fly higher. Both aircraft performance functions and wind are handled by a *propagation function* $\tau : W \times T \times A \rightarrow (W \cup \{\infty\}) \times T$; here, $W \subset \mathbb{R}$ is a set of weights, ∞ represents an infeasible state, and $T \subset \mathbb{R}$ a set of times. Then, the propagation function maps the state of the aircraft at the tail of an arc to its state after traversing the arc. We assume the following propagation properties.

► **Assumption 1.** *Let $\tau : W \times T \times A \rightarrow (W \cup \{\infty\}) \times T$ be a propagation function. For $w_1, w_2 \in W$, $t \in T$, $a_1, a_2 \in A$, $\tau(w_1, t, a) = (w^1, t^1)$, and $\tau(w_2, t, a) = (w^2, t^2)$, it holds:*

- i) $w_1 > w^1$ and $t < t^1$,
- ii) $w_1 < w_2, a_1 = a_2 \implies (w_1 - w^1) < (w_2 - w^2)$,
- iii) $w_1 = w_2, a_1, a_2$ cruise arcs with a_2 on a higher level $\implies (w_1 - w^1) > (w_2 - w^2)$,
- iv) *ceteris paribus*, a descent burns less fuel than a cruise, which burns less than a climb, and a direct descent, if possible, is the most economic way to reach the destination.
- v) For fixed $a \in A$, $t \in T$, the air distance along a at time t (i.e. the effectively traversed distance, after consideration of wind) is proportional to $w_1 - w^1$.

i) states that traversing an arc decreases the weight (by burning fuel) and increases time. ii) means that fuel consumption increases with weight. iii) says that fuel consumption on a cruise phase decreases with altitude^e, iv) is clear. v) states that consumption increases with air distance, which is very intuitive. With these definitions, the FPP can be stated as follows.

^d In our data, this corresponds to an altitude of 300m; the final descent ends on FL h_0 .

^e Recall that we assume that the highest flight level is the optimal one.

► **Definition 1.** Let $G = (V, A)$ be an Airway Network and $v^{DEP}, v^{DEST} \in V$ be the nodes corresponding to the departure and destination airports, respectively. Let $t^0 \in T$ and $w^0 \in W$ be the weight and time at departure, and $\tau : W \times T \times A \rightarrow W \times T$ a propagation function. The Flight Planning Problem (FPP) seeks to find a path $((v_0, v_1), (v_1, v_2), \dots, (v_{n-1}, v_n)) \subset A, n \in \mathbb{N}$, and corresponding sequences of weights $(w_0, w_1, \dots, w_n) \subset W$ and times $(t_0, t_1, \dots, t_n) \subset T$. It must hold that $v_0 = v^{DEP}$, $v_n = v^{DEST}$, $w_0 = w^0$, $t_0 = t^0$, and $\tau(w_i, t_i, (v_i, v_{i+1})) = (w_{i+1}, t_{i+1}) \in W \times T$ for each arc (v_i, v_{i+1}) in the path. The objective is to minimize $w^0 - w_n$.^f

While some variants of the FPP investigated in the literature are solvable in polynomial time [4] under certain assumptions, others are clearly NP-hard ([14], [5]). [13] notes that the FIFO property does not hold under the presence of wind, but that by itself does not have any implications on the computational complexity of the problem.

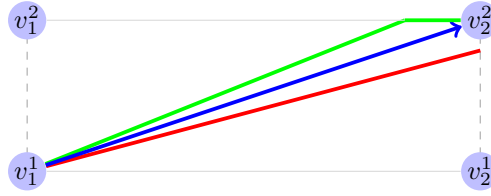
In this section, we show that the version of the FPP considered in this paper is NP-hard, even without consideration of wind. We first note that the weight parameter in the FPP is equivalent to the time parameter in the classical Time-Dependent Shortest Path Problem (TDSPP), such that we can think of fuel propagation functions as traversal-time functions. It is well known that the FIFO property is a sufficient but not a necessary condition for the TDSPP to be solvable in polynomial time, while [20] gave the most widely cited proof that the TDSPP can be NP-hard in non-FIFO networks. They construct travel time functions on a finite domain that have a constant value except for one point. As our fuel propagation functions do not have this structure, [20]’s argument cannot be applied. The same holds for the proof in [23]. To the best of our knowledge, no other published proofs would apply to the FPP. We therefore give a new simple NP-hardness proof based on a more general argument.

Consider the situation in Figure 2. Essentially, an arc representing a climb can sometimes only be flown if the aircraft’s weight is small enough, as otherwise the higher level cannot be reached before the end of the segment represented by the arc. In other words, the consumption given by τ on this arc is finite for weights up to a certain value and jumps to infinity for weights above that value. The phenomenon in Figure 2 is not as contrived as it may seem. To give the reader an idea of the variability of the climb angle: An Airbus A340 with a typical weight of 200t needs roughly 150km of horizontal flight to climb from 5000m altitude to 10000m altitude. Around this weight, an increase of 1kg roughly leads to an increase of 1m in horizontal distance. The FPP is thus a generalization of the TDSPP that allows at most one jump discontinuity in each travel time function, while the proof in [20] assumes two.

► **Theorem 2.** *If traversal time functions are allowed to have at most one jump discontinuity, the TDSPP is NP-Hard.*

Proof. Inspired by [10], we do a reduction from the Exact Path Length (EPL) problem, which is NP-hard according to [19]. Consider a directed graph $G = (V, A)$ with non-negative lengths on the arcs $c : A \rightarrow [0, \infty)$, two nodes s, t , and $L > 0$. The EPL consists of determining whether an (s, t) -path of length L exists. For the reduction, we define travel time functions on G as follows. Let $M \gg 0$ be a very large number. Without loss of generality we assume that the departure time is 0. For each $a \in A$ and $\tau \in [0, \infty)$, and using $h(a)$ to denote the

^f Of course, since w^0 is constant, this is equivalent to maximizing w_n .



■ **Figure 2** The green profile represents a climb along a segment and between two levels. The climb is steep enough that the higher level is reached before the end of the segment, and the aircraft can cruise until reaching node v_2^2 . This climb is represented in the graph by the blue arc (v_1^1, v_2^2) . The red profile shows a climb started with a higher weight. This leads to a flatter climb, making it impossible to reach v_2^2 . Thus, when starting with this weight, the blue arc has an infinite consumption.

head of a , we define [§]

$$T(a, \tau) = \begin{cases} M & \text{if } a = (v, t), \tau < L - c(a) \\ c(a) & \text{else.} \end{cases}$$

If the last arc $a = (v, t)$ of any (s, t) -path is entered at time $\tau < L - c(a)$, the objective value will be larger than M . Thus, L is the smallest possible arrival time, and if a path with this arrival time exists, the TDSPP will find it. Clearly, any path with travel time L also has length L , and vice-versa. Consequently, feasible solutions of the EPL problem correspond to optimal solutions for the TDSPP constructed above. This completes the reduction. ◀

3 An A* Algorithm based on Idealized Vertical Profiles

3.1 The state of the art

Recall that an A* algorithm is based on a potential function $\pi : V \rightarrow \mathbb{R} \cup \{\infty\}$. π is said to be *admissible* if $\pi(v)$ is a lower bound on the costs from v to the target for $v \in V$. It is *consistent* if $\pi(u) - \pi(v) \leq c(u, v)$ for $(u, v) \in A$, where c is the cost function. Given that $\pi(t) = 0$ for the target t , which we can and henceforth will assume w.l.o.g., consistency implies admissibility, and a consistent and admissible potential guarantees that an A* algorithm finds the same solution as Dijkstra's algorithm. Tight, consistent potentials lead to a faster A* algorithm. The Single Descent (SD) algorithm of [15] pursues such an idea for the FPP: For each segment, a lower bound on the fuel consumption over it is computed as a cruise on the optimal flight level, with the optimal wind conditions, and the minimum possible aircraft weight. On a 2D projection, a backwards Dijkstra search from the destination airport is done w.r.t. these arc costs. The resulting distances are used as initial potentials. In the forward search during the expansion of each label, a descent from that label to the ground is calculated and the corresponding consumption is added to the initial potential. Since the distance traversed by that descent is now covered both by a cruise (initial potential) and a descent (first correction), a second and final correction step is made: The distance of that descent is traversed from the label in cruise mode, and this consumption is subtracted from the preliminary potential, thus defining the final potential.^h Despite the resulting potentials being neither admissible nor consistent, the ensuing, label-setting, SD algorithm

[§] Note that this step function does not satisfy the FIFO property, despite similar functions in the literature preserving it, such as [8].

^h This is how we interpret the algorithm. Unfortunately, the paper is not very detailed, in particular, w.r.t. the construction of the lower bounds.

is very effective in practice and marks the current state-of-the-art. Our motivation for improvement is that the initial potentials can be very loose for labels that are far away from the destination, since SD assumes a very low weight and no climbs. This can lead to a significant underestimation of the consumption. The on-the fly correction in the forward search mitigates this problem, but does not resolve it completely.

3.2 Basic framework

We propose an A* algorithm based on a simple ideaⁱ. In practice, flight routes are constrained by the airway network and by wind conditions. If neither of these existed, but cruise phases were still constrained to flight levels, and the routes would always follow the step-climb pattern, see again Figure 1. Namely, the horizontal route component would be the great-circle line connecting both airports, while the vertical route component would consist of a series of climb-cruise-climb sequences up to the optimal flight level. There, the aircraft would cruise until the final descent is started, which would take it straight to the destination airport. The consumption arising from this *idealized vertical profile* (IVP) on a lower bound on the flight distance is an admissible (and, as it will turn out, also consistent) potential for an A* algorithm. Calculating the IVP during the search is too costly, as the decrease in the number of labels would be offset by the effort to compute the potentials. However, it will turn out that this problem can be overcome by a combination of preprocessing and on-the-fly calculations. A formal description is as follows.

A crucial element is the distance underestimation. The results in this section are of a general nature and the specific type of underestimation is not important. In our implementation, we will obtain using the technique introduced in [4] (“super-optimal” wind calculations combined with a backwards search).

► **Definition 3.** Let $c^{IVP}(w, h, d, h_T)$ be the minimum amount of fuel that is needed to fly, assuming no wind influence, the distance d by doing some combination of climb/cruise/descent phases, starting at altitude h with weight w , and finishing at altitude h_T ; if the distance is too short to reach the target altitude, it is the amount of fuel that is needed to make an immediate descent to the target altitude; if the target cannot be reached, it is infinity. In the first two cases, the vertical profile $p^{IVP}(w, h, d, h_T)$ of the associated trajectory is called idealized vertical profile (IVP).

► **Assumption 2.** Every IVP consists of a finite and alternating series of climb and cruise phases followed by a single descent phase.

► **Assumption 3.** $c^{IVP}(w, h, d_1, h_T) \leq c^{IVP}(w, h, d_2, h_T)$ for distances $d_1 \leq d_2$.

In other words, Assumption 2 means that the IVP looks like the one in Figure 1. It also implies that the highest level reached by the IVP is at most the aircraft’s optimal cruise altitude.^j Assumption 3 means that, ceteris paribus, longer trajectories are more expensive.

► **Proposition 4.** For an FPP and $v \in V$, consider a (v^{DEP}, v) -path in G that reaches v with weight w at time t . Let $h(v)$ be the flight level at v . Let \underline{d} be a lower bound on the distance from v to v^{DEST} obtained by a backwards search from v^{DEST} using lower bounds

ⁱ The algorithm is label-setting and necessarily heuristic, as the FPP is NP hard.

^j See Section 2 for the definition of the optimal cruise altitude.

on the (time-dependent) arc distances as costs; recall that h_0 is the lowest flight level. Then the function

$$c : V \times W \rightarrow [0, \infty], (v, w) \mapsto c^{IVP}(w, h(v), \underline{d}, h_0)$$

is admissible and consistent.^k

Proof. Admissibility follows from Assumptions 2 and 3, as well as part v) of Assumption 1. Indeed, the air distance traversed by the best possible trajectory following any (v, v^{DEST}) -path in G is at least \underline{d} , and its vertical profile is constrained by starting climbs only over waypoints. Hence, it burns more fuel than the IVP $p^{IVP}(v, w, \underline{d}, h_0)$.

For consistency, consider an arc $(u, v) \in A$, a weight w_u , and a time t_u . Traversing the arc with the initial state (w_u, t_u) leads to the state $\tau(w_u, t_u, (u, v)) = (w_v, t_v)$, so the consumption on the arc is $w_u - w_v$. We must thus prove $c(u, w_u) - c(v, w_v) < w_u - w_v$.

Let \underline{d}_u and \underline{d}_v be the lower bounds used to calculate the potentials $c(u, w_u)$ and $c(v, w_v)$ from IVPs $p^{IVP}(w_u, h(u), \underline{d}_u, h_0)$ and $p^{IVP}(w_v, h(v), \underline{d}_v, h_0)$ and let $d(u, v, w_u, t_u)$ be the actual air distance traversed on (u, v) . We now distinguish three cases; Case 1 is the standard “en route” case, Cases 2 and 3 come up close to the destination, when the lower bound on the remaining distance becomes small.

Case 1: Neither $p^{IVP}(w_u, h(u), \underline{d}_u, h_0)$ nor $p^{IVP}(w_v, h(v), \underline{d}_v, h_0)$ are immediate descents. Then

$$\begin{aligned} c(u, w_u) &= c^{IVP}(w_u, h_u, \underline{d}_u, h_0) \leq c^{IVP}(w_u, h_u, d(u, v, w_u, t_u) + \underline{d}_v, h_0) \\ &\leq w_u - w_v + c^{IVP}(w_v, h_v, \underline{d}_v, h_0) \\ &= w_u - w_v + c(v, w_v), \end{aligned}$$

where the first inequality follows from the triangle inequality $\underline{d}_u \leq d(u, v, w_u, t_u) + \underline{d}_v$ for distance lower bounds and Assumption 3, and the second from the optimality of the IVP $p^{IVP}(w_u, h_u, d(u, v, w_u, t_u) + \underline{d}_v, h_0)$, which burns at most the same amount of fuel as the concatenation of (u, v) and the IVP $p^{IVP}(w_v, h(v), \underline{d}_v, h_0)$.

Case 2: Only $p^{IVP}(w_v, h(v), \underline{d}_v, h_0)$ is an immediate descent. The same argument as in Case 1 applies, since the total distance traversed on (u, v) and then on the descent from v will be longer than the distance traversed by the IVP starting at u .

Case 3: $p^{IVP}(w_u, h(u), \underline{d}_u, h_0)$ is an immediate descent. Now the relative lengths of lower bounds on the traversed air distances are unclear, because a descent is steeper with a lower weight, possibly causing the profile via v to end up with a smaller distance bound. However, we can use Assumption 1 iv) on aircraft performance which implies that a direct descent burns less fuel than any other combination of flight phases leading to the same altitude. ◀

3.3 Calculation of the Idealized Vertical Profile

In the previous section, we proved that the A* potentials calculated using the IVP method are admissible and consistent under consideration of two assumptions. Now, we sketch how these potentials can be computed in practice.

Assumption 2 states that an IVP follows a step-climb procedure until reaching the highest level that allows a direct descent to the ground. Each cruise in this profile is just long enough to burn enough fuel to reach a weight that allows a further climb. Once the highest level is reached, the aircraft cruises until the point where it starts the final descent.

^k We define admissibility and consistency for a function with domain $V \times W$ in the canonically extended way. It is easy to see that all known properties still hold.

To speed-up the first part of the calculation we observe that the weight at the end of a cruise and at the start of a subsequent climb is constant for a fixed flight level. This is because, by definition, this is the largest weight that allows starting a climb from that level. Similarly, the distances of each such phase are constant. This allows us to pre-compute, for each pair of levels, the total consumption and total distance corresponding to a step climb between these levels, as well as the weight at the start of this step-climb.

The second phase of the calculation, consisting of a single cruise followed by a descent, is trickier. The reason is that the cruise distance plus the descent distance must equal the remaining distance, but the descent distance depends on the weight at its start, which in turn depends on the length of the cruise. In practice, the top of descent is computed by an iterative procedure that progressively adjusts the cruise distance until a total cruise+descent distance is reached that is close enough to the target distance. This procedure can be very time-consuming, which makes it another good candidate for pre-computation. The difficulty is that more parameters are involved than in the step-climb case: Both the weight before the cruise-descent and the remaining distance are unclear.

We solve this problem in the following pre-processing step: For each flight level, we calculate the maximum descent distance from that level to the ground. We then consider a discretization of the complete weight range (that is, from the aircraft's dry operating weight to its maximum take-off weight). For each weight in this discretization, we compute the (time-consuming) IVP on the remaining distance.

The complete calculation of the potentials is described in Algorithm 1 for a weight w , a flight level h , and a remaining distance d . In a nutshell, we compute the IVP as described above. Step-climbs are not calculated on-the-fly, instead we use the pre-calculated data. Near the destination airport, we use the second batch of pre-calculated data and interpolate the weight; of course the potential is only admissible and consistent if this discretization is fine enough. The step in line 3 is the most expensive part of the algorithm, but it needs to be calculated only for nodes that are very close to the destination airport.

4 Computational Results

In this section, we benchmark the performance of our A* algorithm using potentials from Idealized Vertical Profiles (IVP) against Dijkstra's algorithm (D) and the Single Descent algorithm (SD). In case of Single Descent, our implementation tries to follow the description in [15] as far as we could, filling in some gaps using our best judgement. To make the comparison fair, all algorithms use the same data structures, in particular, the same priority queue, such that the only difference is in the calculation of the potentials; for Dijkstra's algorithm, there are of course none. The programming language is C++, compiled with GCC 7.5.0. All computations were performed on a machine with 95 GB of RAM and an Intel(R) Xeon(R) Gold 5122 processor with 3.60GHz and 16.5 MB cache.

4.1 Instances

The airway network, the weather, and the aircraft performance data were provided by our industrial partner Lufthansa Systems. The airway network consists of 410387 waypoints, 878884 airway segments, and 232 flight levels. A naive construction would result in a graph with over 95 million (410387×232) nodes and over 47 billion ($878884 \times 232 \times 231$) arcs. However, a large majority of those nodes and arcs are not flyable, for example due to the waypoints and segments not available on the corresponding altitudes, or because

■ **Algorithm 1** Potential calculation.

Require: w, h, d , max. descent distance function $d^{max}(\cdot)$, preprocessed step-climb- and final descent data.

```

1:  $w_0 = w$ 
2: if  $d < d^{max}(h)$  then
3:   Calculate the IVP from this point by evaluating all possible step climbs followed by
   on-the-fly final descent iterations. If the distance is too short, do a simple descent.
4:    $w \leftarrow w - \text{IVP consumption}$ 
5: else
6:   Climb to the highest level  $h_1$  that is reachable and satisfies
    $d - \text{climb distance} \leq d^{max}(h_1)$ 
7:    $w \leftarrow w - \text{climb consumption}$ 
8:    $d \leftarrow d - \text{climb distance}$ 
9:    $h \leftarrow h_1$ 
10:  if it's not possible to climb further then
11:    Read from the precalculated results what is the maximal weight on this level that
    allows a climb. Cruise until that weight is reached.
12:     $w \leftarrow w - \text{cruise consumption}$ 
13:     $d \leftarrow d - \text{cruise distance}$ 
14:    There is a set of pre-calculated step-climbs starting at the  $h$  with weight  $w$ .
15:    Choose the maximal  $h_2$  such that the step-climb to  $h_2$  satisfies
    $d - \text{step-climb distance} \leq d^{max}(h_2)$ 
16:     $w \leftarrow w - \text{step climb consumption}$ 
17:     $d \leftarrow d - \text{step climb distance}$ 
18:     $h \leftarrow h_2$ 
19:  end if
20:  Cruise until  $d - \text{cruise distance} = d^{max}(h)$ 
21:   $w \leftarrow w - \text{cruise weight}$ 
22:   $d \leftarrow d - \text{cruise distance}$ 
23:  In the weight discretization, find the closest weights  $w_1, w_2$  s.t.  $w_1 \leq w \leq w_2$ 
24:  Let  $c_1$  be the pre-computed consumption for  $w_1, h$  and  $c_2$  the pre-computed
   consumption for  $w_2, h$ .
25:   $w \leftarrow w - \frac{w-w_1}{w_2-w_1}c_2 + \frac{w_2-w}{w_2-w_1}c_1$ 
26: end if
27: return  $w_0 - w$ 

```

the segment is too short for a given climb. Furthermore, the availability of certain arcs depends on the current weight, further complicating things. In our implementation, we generate the graph dynamically, therefore it is difficult to give an absolute graph size. We use propagation functions for two aircraft models, an Airbus A320 (suitable for short-haul flights) and an Airbus A340 (used for middle- to long-haul flights), derived by interpolation from corresponding tables. Unfortunately, this data, which consists of tables with millions of entries, is only an approximation of the real performance functions. It turns out that Assumption 2 is prevalent, but not always satisfied. This breaks consistency of the IVP

algorithm such that it does not necessarily find the same solution as Dijkstra’s. However, our computational results show that the resulting gap between Dijkstra’s and the IVP A*-algorithm is mostly extremely small or non-existent, i.e., this data problem is marginal.

The OD-pairs were defined in the same way as in [15]. For the long-haul test set, we chose a set of 20 major airports evenly distributed around the globe. All pairs with great-circle-distances between 4000km and 11000km were considered, resulting in 202 ordered pairs. For the short-haul test set we did the same thing on the basis of a set of 19 major airports in Europe, using 500km and 4000km as distance bounds. This results in 294 ordered pairs. We calculate the short-haul flights with the A320 and the long-haul ones with the A340.

To define the take-off weight, we run Dijkstra’s algorithm once on each instance, starting with the maximum possible amount of fuel. We multiply the resulting consumption by 1.2 and fix this number as the amount of fuel at take-off.

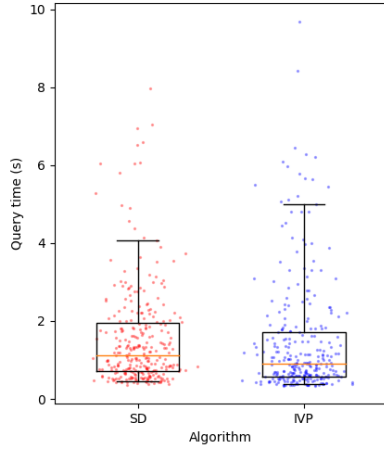
4.2 Methodology

As is customary in the shortest-path literature, we separate runtimes into two categories: Those in a preprocessing phase, which is instance-independent, and those in a query phase, which includes the shortest path calculation and all instance-dependent preprocessing stages. We ignore the runtime of procedures that are identical across all variants. This includes the construction of the graph, the initialization of the search algorithm, etc. More precisely: Dijkstra’s algorithm (D) does not need any preprocessing. For Single Descent (SD), we consider the calculation of the minimum cruise consumption on each arc as a preprocessing operation, as dependent only on the aircraft and the weather forecast, but not on the OD-pair. The backwards search to determine the potentials is included in the query time. Idealized Vertical Profiles (IVP) require a substantial preprocessing phase for the pre-calculation of step-climbs and final-descent stages, for which we choose a weight discretization with steps of 1000kg. This preprocessing effort depends only on the aircraft, but not on the weather forecast, and not on the OD-pair. It therefore can be done once for each aircraft, which makes the associated preprocessing time irrelevant. For the sake of completeness, we nevertheless report it. As with SD, the backwards search to determine the minimum distance from each node to the destination is included in the query time. Both SD and IVP require the calculation of lower bounds on the air distance for which we use the super-optimal wind technique from [4]. Since these computations are identical for both algorithms, we omit them. We run each calculation thrice and report the smallest time.

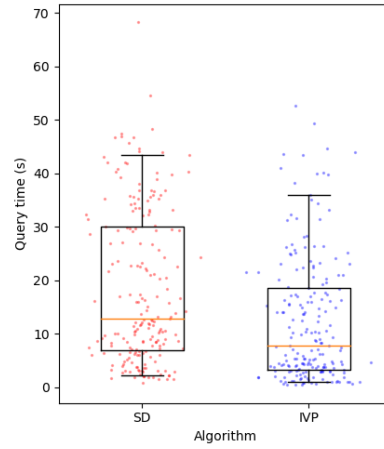
4.3 Results

Figures 3 and 4 show the query times of all three algorithms. The results are summarized in Tables 1 and 2 (short-haul and long-haul instance sets, respectively). For each statistic, we list both the arithmetic mean (*ar mean*) and the geometric mean (*geo mean*). The names used for the statistics are self-explanatory with the possible exception of *nr. labels*. This is the total count of labels that were expanded during the search.

The query times of both SD and IVP are far superior to Dijkstra’s algorithm. Furthermore, IVP outperforms SD by roughly 5-12% on the short-haul instances and by 33-40% on the long-haul instances. As one would expect, the number of labels expanded by IVP is much smaller than that of the other two algorithms. This reduction is so significant that the expensive potential calculations are compensated. The cost of these calculations can best be seen by observing the number of labels expanded in the long-haul instances. IVP expands around 243k labels on average (geometric mean), which is less than half of those expanded by SD, while the speedup is 1.76 (geometric mean).



■ **Figure 3** Short-haul runtimes.



■ **Figure 4** Long-haul runtimes.

■ **Table 1** Computational results on the short-haul instances.

preprocessing (s)	D		SD		IVP	
	-		0.19		11.16	
	ar mean	geo mean	ar mean	geo mean	ar mean	geo mean
query (s)	11.11	8.71	1.56	1.20	1.47	1.05
cost (kg)	4096.26	3698.07	4107.08	3709.44	4096.27	3698.07
nr. labels	472528.44	379276.36	50298.22	33548.52	41580.47	19310.30
query speedup w.r.t. D (s)	-	-	9.55	7.33	9.64	7.44
query speedup w.r.t. D (×)	-	-	8.06	7.24	9.72	8.33
cost gap (kg)	-	-	10.81	0.00	0.01	0.00
cost gap (%)	-	-	0.31	0.00	0.00	0.00
labels (% of D)	-	-	10.09	8.85	7.11	5.09

As can be seen in Figures 3 and 4, but also in the tables, the speedup of IVP w.r.t SD is much more pronounced in the long-haul instances. This is to be expected for various reasons: One is that in SD, both the cruise consumption estimation and the descent consumption are much nearer to the actual consumptions when flying close to the destination airport, which is the case for a big part of the search on short-haul flights. Another reason is that IVP does more expensive calculations in the area close to the destination airport – the proportion of this area to the whole search space is much larger in the short-haul case.

The preprocessing time of IVP (72s in the long-haul case) is definitely longer than that of SD but still very manageable, especially considering that it needs to be done only once per aircraft model. In practice, airlines acquire new aircraft so seldom that even a preprocessing time of several days would be acceptable.

Concerning the quality of the solutions: As expected (see Section 4.1), the gap¹ between the values returned by IVP and Dijkstra is not always zero, meaning that, for the data available to us, the IVP potentials are not consistent. Nevertheless, both IVP and SD yield

¹ We do not say *optimality gap* since Dijkstra is not guaranteed to be optimal due to the NP-hardness of the problem.

■ **Table 2** Computational results on the long-haul instances.

preprocessing (s)	D		SD		IVP	
	-		0.20		71.86	
	ar mean	geo mean	ar mean	geo mean	ar mean	geo mean
query (s)	65.78	57.03	18.52	12.87	12.33	7.30
cost (kg)	57086.24	55678.96	57100.34	55693.62	57099.54	55690.96
nr. labels	2637588.37	2340711.69	686281.66	504259.75	418759.61	243834.10
query speedup w.r.t. D (s)	-	-	47.26	41.08	53.45	46.77
query speedup w.r.t. D (×)	-	-	5.36	4.43	11.36	7.81
cost gap (kg)	-	-	14.09	0.00	13.30	0.00
cost gap (%)	-	-	0.03	0.00	0.02	0.00
labels (% of D)	-	-	24.84	21.54	14.56	10.42

results of a very good quality. For both variants and both test cases, the geometric mean of the gap w.r.t. Dijkstra is 0.00%, meaning that the gap is extremely small except for a few outliers. Finally, the arithmetic mean shows a small improvement of IVP over SD, especially on short-haul instances.

Another possible reason is the weight discretization used for calculating the consumption in the last section of the IVPs. However, a discretization of 1kg instead of the 1000kg used in our calculations did not yield a noticeable improvement in the solutions' quality, while slightly increasing the runtime of both queries and preprocessing. Thus, it is not included in the presented results.

5 Conclusion

In this paper, we investigated the Flight Planning Problem (FPP), which is a generalization of the Time-Dependent Shortest-Path Problem (TDSPP). We presented the first proof of its NP-hardness, which extends to a more general family of TDSPP variants.

We also introduced an A* algorithm based on potentials derived from Idealized Vertical Profiles (IVPs). We showed that, under reasonable theoretical assumptions on the aircraft performance functions, IVP potentials are both admissible and consistent, such that a corresponding A* algorithm finds the same solution as Dijkstra's algorithm. We show that IVP potentials can be calculated efficiently by a combination of preprocessing and on-the-fly computations.

Our computational results on real-world instances show that the effort to calculate IVP potentials pays off and results in a significant improvement of the overall query time as compared to the state-of-the-art Single Descent algorithm introduced in [15]. Indeed, we obtain a speed-up of up to 40% and a smaller consistency gap.

References

- 1 Hannah Bast, Daniel Delling, Andrew Goldberg, Matthias Müller-Hannemann, Thomas Pajor, Peter Sanders, Dorothea Wagner, and Renato F. Werneck. Route planning in transportation networks, 2015. doi:10.48550/ARXIV.1504.05140.
- 2 Moritz Baum, Julian Dibbelt, Dorothea Wagner, and Tobias Zündorf. Modeling and engineering constrained shortest path algorithms for battery electric vehicles. *Transportation Science*, 54:1571–1600, November 2020. doi:10.1287/trsc.2020.0981.
- 3 Marco Blanco, Ralf Borndörfer, Nam Dung Hoang, Anton Kaier, Pedro Maristany de las Casas, Thomas Schlechte, and Swen Schlobach. Cost projection methods for the shortest path problem with crossing costs. In Gianlorenzo D'Angelo and Twan Dollevoet, editors,

- 17th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2017), volume 59, 2017.
- 4 Marco Blanco, Ralf Borndörfer, Nam Dũng Hoàng, Anton Kaier, Adam Schienle, Thomas Schlechte, and Swen Schlobach. Solving time dependent shortest path problems on airway networks using super-optimal wind. In *16th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2016)*, volume 54, 2016. in press. doi:10.4230/OASICS.ATMOS.2016.12.
 - 5 Marco Blanco, Ralf Borndörfer, Nam Dũng Hoàng, Anton Kaier, Thomas Schlechte, and Swen Schlobach. The shortest path problem with crossing costs. techreport 16-70, ZIB, 2016.
 - 6 H.M. de Jong. Optimal track selection and 3-dimensional flight planning. techreport, Royal Netherlands Meteorological Institute, 1974.
 - 7 Daniel Delling and Giacomo Nannicini. Core routing on dynamic time-dependent road networks. *INFORMS Journal on Computing*, 24(2):187–201, May 2012. doi:10.1287/ijoc.1110.0448.
 - 8 Jochen Eisner, Stefan Funke, and Sabine Storandt. Optimal route planning for electric vehicles in large networks. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*, AAAI'11, pages 1108–1113. AAAI Press, 2011.
 - 9 Patrick Hagelauer and Felix Antonio Claudio Mora-Camino. A soft dynamic programming approach for on-line aircraft 4D-trajectory optimization. *European Journal of Operational Research*, 107(1):87–95, May 1998. doi:10.1016/S0377-2217(97)00221-X.
 - 10 Edward He, Natasha Boland, George Nemhauser, and Martin Savelsbergh. Time-dependent shortest path problems with penalties and limits on waiting. *INFORMS Journal on Computing*, 2020.
 - 11 Casper Kehlet Jensen, Marco Chiarandini, and Kim S. Larsen. Flight Planning in Free Route Airspaces. In Gianlorenzo D'Angelo and Twan Dollevoet, editors, *17th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2017)*, volume 59 of *OpenAccess Series in Informatics (OASICS)*, pages 1–14, Dagstuhl, Germany, 2017. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/OASICS.ATMOS.2017.14.
 - 12 Stefan E. Karisch, Stephen S. Altus, Goran Stojković, and Mirela Stojković. Operations. In Cynthia Barnhart and Barry Smith, editors, *Quantitative Problem Solving Methods in the Airline Industry*, volume 169 of *International Series in Operations Research & Management Science*, pages 283–383. Springer US, 2012.
 - 13 Anders Nicolai Knudsen, Marco Chiarandini, and Kim S. Larsen. Vertical optimization of resource dependent flight paths. In *ECAI 2016 - 22nd European Conference on Artificial Intelligence, 29 August-2 September 2016, The Hague, The Netherlands - Including Prestigious Applications of Artificial Intelligence (PAIS 2016)*, pages 639–645, 2016. doi:10.3233/978-1-61499-672-9-639.
 - 14 Anders Nicolai Knudsen, Marco Chiarandini, and Kim S. Larsen. Constraint Handling in Flight Planning. In *Principles and Practice of Constraint Programming - 23rd International Conference, CP 2017, Melbourne, VIC, Australia, August 28 - September 1, 2017, Proceedings*, pages 354–369, 2017. doi:10.1007/978-3-319-66158-2_23.
 - 15 Anders Nicolai Knudsen, Marco Chiarandini, and Kim S. Larsen. Heuristic Variants of A* Search for 3D Flight Planning. In *Integration of Constraint Programming, Artificial Intelligence, and Operations Research - 15th International Conference, CPAIOR 2018, Delft, The Netherlands, June 26-29, 2018, Proceedings*, pages 361–376, 2018. doi:10.1007/978-3-319-93031-2_26.
 - 16 P. Maristany de las Casas, A. Sedeño-Noda, and R. Borndörfer. An improved multiobjective shortest path algorithm. *Computers & Operations Research*, page 105424, June 2021. doi:10.1016/j.cor.2021.105424.
 - 17 Pedro Maristany de las Casas, Luitgard Kraus, Antonio Sedeño-Noda, and Ralf Borndörfer. Targeted multiobjective dijkstra algorithm, 2021. doi:10.48550/ARXIV.2110.10978.

- 18 Hok K. Ng, Banavar Sridhar, and Shon Grabbe. Optimizing aircraft trajectories with multiple cruise altitudes in the presence of winds. *Journal of Aerospace Information Systems*, 11(1):35–47, 2014.
- 19 Matti Nykänen and Esko Ukkonen. The exact path length problem. *Journal of Algorithms*, 42(1):41–53, 2002. doi:10.1006/jagm.2001.1201.
- 20 Ariel Orda and Raphael Rom. Traveling without waiting in time-dependent networks is np-hard. Technical report, Department Electrical Engineering, Technion-Israel Institute of Technology, 1989.
- 21 Adam Schienle, Pedro Maristany, and Marco Blanco. A Priori Search Space Pruning in the Flight Planning Problem. In Valentina Cacchiani and Alberto Marchetti-Spaccamela, editors, *19th Symposium on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2019)*, volume 75 of *OpenAccess Series in Informatics (OASICS)*, pages 8:1–8:14, Dagstuhl, Germany, 2019. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/OASICS.ATMOS.2019.8.
- 22 Ben Strasser and Tim Zeitz. A fast and tight heuristic for a* in road networks, 2019. doi:10.48550/ARXIV.1910.12526.
- 23 Tim Zeitz. Np-hardness of shortest path problems in networks with non-fifo time-dependent travel times. *Information Processing Letters*, 179:106287, 2023. doi:10.1016/j.ipl.2022.106287.