




Feeder Routing for Air-to-Air Refueling Operations

Christoph Hansknecht^{1*}, Imke Joormann², Bernd Korn³,
Fabian Morscheck³, and Sebastian Stiller¹

¹Institute for Mathematical Optimization, TU Braunschweig, Germany

²Institute of Automotive Management and Industrial Production, TU Braunschweig, Germany

³Institute of Flight Guidance, DLR, Braunschweig, Germany

*Corresponding author, c.hansknecht@tu-braunschweig.de

September 27, 2019

We consider the problem of routing a fleet of feeders for civil air-to-air refueling operations. In the *air-to-air refueling problem*, a fixed set of cruisers requires refueling by a fleet of feeders at fixed locations and fixed points in time. A typical objective function is to minimize the fuel consumption or the total number of required feeders.

We formulate a discrete optimization problem based on an ODE model for the fuel consumption of the feeders. The fuel consumption of a feeder depends on its weight. The weight changes over time and depends significantly on the fuel mass stored in the feeder. The fuel mass stored depends on the length of the route and the fuel mass for the requests served. We prove \mathcal{NP} -hardness of the problem and develop a column generation approach. We prove several structural properties of the model that allow us to improve the solution method. The resulting method is applicable in practice, which we demonstrate by conducting computational experiments on instances for both random generated demands and demands based on real-world air-traffic. We compare the optimized routes to state-of-the-art solutions.

It turns out that mathematical optimization techniques on average reduce the fuel consumption of the feeder fleet by more than half.

Keywords – Air-to-Air Refueling, Branch-and-Price, Column Generation, NP-hardness, Sustainable Aviation

1. Introduction

According to the International Civil Aviation Organization, it is expected that the world scheduled passenger traffic will quadruple by 2045 [24]. To reduce the environmental impact, especially the carbon footprint, the European Commission’s “Flightpath 2050” defines a target of 75% CO₂ reduction per passenger-kilometre through technological development by 2050 [14]. Moreover, the price for kerosene is forecasted to more than double by 2050 from the current price of approximately US\$0.05 kW⁻¹ h⁻¹ [40]. As a result, considerable effort has already been made in order to increase the fuel efficiency of aircraft operations.

One promising concept to further increase efficiency is the introduction of air-to-air refueling operations: Dividing a flight range into smaller parts and refueling on air between parts allows to design aircrafts with reduced weight, which can traverse the flight range with a decreased fuel burn. The cruiser-feeder approach to air transport has been studied as part of the RECREATE (REsearch on a

CRuiser Enabled Air Transport Environment) project, resulting in a predicted 4.5%–6% reduction in operating costs [34].

Aside from the construction of the cruiser and feeder aircraft, a key aspect regarding the overall efficiency of an air-to-air refueling operation is the scheduling of the operation itself. While the optimized design of the cruisers ensures a decreased fuel consumption of the cruisers themselves, the feeder aircrafts consume fuel while serving the cruisers. To increase the efficiency of the overall operation, it is imperative to keep these additional costs as low as possible. While previous research [34, 16, 32] laid focus on design aspects as well as on the cruiser operation, the feeder side of the problem consists of two different optimization goals, both influencing the cost caused by the feeder fleet.

The first goal is the reduction in fuel consumption of the feeders. In earlier simulations, a simple distribution system was employed in order to assign the feeders in order to satisfy the cruisers' demands. A more thorough investigation of the underlying combinatorial problem is likely to yield solutions with significantly reduced fuel consumption.

The second goal is the reduction of the size of the feeder fleet, which is not only beneficial in itself, but may also yield a reduction with respect to the size of the feeder bases. A reduction of the cost of the required infrastructure in turn benefits the entire cruiser-feeder system. Again, an optimized feeder distribution could allow for the reduction of the fleet size as well as the necessary feeder infrastructure on the ground.

Still, we would like to point out that these two goals can be adversarial. Specifically, a larger feeder fleet size may very well yield savings in terms of fuel consumption, whereas an increased fuel budget may allow for a smaller fleet. We will proceed to consider both goals separately, giving some examples for the trade-off between the objectives later on.

Structure of the Paper

First, we formally define the air-to-air refueling problem. Second, we give a literature overview and briefly discuss the underlying scenario and its use in related work. Section 2 is devoted to the physical model of the feeder fuel consumption across different flight phases.

We go on to introduce two competing integer programming models based on this physical model in order to solve the air-to-air refueling problem in Section 3. Aside from formulating the models, we adapt a well-known labeling algorithm [3] to solve the emerging pricing problem and discuss some details pertaining to the Branch-and-Price framework.

In Section 4, we proceed to computationally evaluate our formulations both on the original scenario and a number of artificially generated instances. We compare the running times of the formulations and examine the quality of the obtained optimal solutions related to the preexisting state-of-the-art heuristic. Lastly, we give our conclusion as well as an outlook regarding potential future work in Section 5.

1.1. Problem Definition

An instance of the air-to-air refueling problem consists of a set \mathcal{R} of refueling requests. Each request $r \in \mathcal{R}$ has an origin $orig(r)$ and a destination $dest(r)$, a time $\theta(r)$, and a requested fuel mass $M^{req}(r)$. The time $\theta(r)$ determines when the refueling operation must start. The coordinates $orig(r)$ and $dest(r)$ are the endpoints of the route along which the refueling operation takes place. The feeders operate from a base b . In order to fulfill a series r_1, \dots, r_k of requests in \mathcal{R} , a feeder departs from the base b , moves to $orig(r_1)$, performs a refueling operation at time $\theta(r_1)$ arriving at $dest(r_1)$, moves to $orig(r_2)$ and so on, until finally returning to the base b from $dest(r_k)$. Note that while feeders are allowed to loiter between requests, each request $r \in \mathcal{R}$ must be served precisely at $\theta(r)$.

Coordinates are given by longitude / latitude pairs, and feeders / cruisers are assumed to fly on shortest paths (on great circle routes) between coordinates. We denote the distance between two coordinates p and p' by $d(p, p')$. We compute this distance using the Haversine formula [29]. This distance function is metric. Furthermore, we assume for technical reasons that the coordinates of the base, the request origins, and the request destinations are pairwise different.

We suppose that the feeders are uniform with respect to their physical properties and flight characteristics. Specifically, feeders maintain a constant speed v during the entire refueling operation, they have a constant *lift over drag ratio* denoted by L/D , a *specific fuel consumption* sfc , and an *efficiency* X satisfying

$$X = \frac{v \cdot L/D}{sfc}.$$

Furthermore, each feeder has an *Operational Empty Weight* M^{ac} and a *Maximum Take-off Weight* $M^{\text{takeoff}} > M^{\text{ac}}$. The difference $M^{\text{max}} := M^{\text{takeoff}} - M^{\text{ac}}$ is available to store the fuel, which is either delivered to cruisers or burned during the flight of the feeder itself.

1.2. Related Work

Air-to-air refueling has received a lot of attention over the last years. We refer to [43] for an overview on the state of research in air-to-air refueling, including engineering aspects such as hose design, position tracking and rendezvous scheduling. Apart from the RECREATE project (see above), several approaches regarding the scheduling have been made. In [15], a multi-objective IP is used to determine the optimal rendezvous points for the refueling process (see also the references therein for further solution approaches for this specific problem). Using a reformulation as a parallel machine scheduling problem with due dates, the authors of [27] allow the refueling to take place somewhere in a certain interval and minimize the tardiness. In contrast to the previous application in military operations, civil air-to-air refueling must take the safety and comfort of passengers into account. Hence, in [44], it is assumed that there should be no flight maneuvers on the side of the cruisers involved. This amounts to a flight guidance problem, where an optimization model is used to shape the trajectory of the feeder such that its velocity vector aligns with the velocity vector of the cruiser near their rendezvous point.

In the same line of thought, we regard the rendezvous points (and times) as fixed, leading to a routing problem for the feeders. Routing problems are among the most famous combinatorial optimization problems. The problem of finding a shortest path for single vehicles has evolved from the usage of Dijkstra's algorithm to highly sophisticated preprocessing schemes (see [4]), including aspects such as time dependence [9] and multi-modality [8].

Whereas these problems are generally polynomially solvable, there are a number of routing problems which are \mathcal{NP} -hard to solve, including the problem of finding shortest paths subject to resource constraints [5] or time-windows [10].

While it is possible to derive combinatorial algorithms for \mathcal{NP} -hard problems, oftentimes it is advisable to use (mixed) integer programming [45] techniques instead (we shall follow this approach as well). The field of integer programming has been studied extensively (for a survey, see [26]) for the last sixty years, yielding theoretical results as well as well-tested, ready-to-use software, which can easily be adapted to specific use cases.

Apart from topics such as cutting planes, branching rules, and symmetry handling, an import technique is that of column generation (see [33]). A column generation approach allows for the exploitation of the problem structure in order to find decompositions into subproblems. While some progress has been made regarding generic column generation algorithms (see [17]), the structure of a specific combinatorial optimization problem cannot generally be automatically recognized, necessitating the adaptation of existing integer programming software to specific problems. We will follow this approach in order to solve the air-to-air refueling problem.

The air-to-air refueling problem is closely related to the vehicle routing problem (VRP), a generalization of the famous traveling salesman problem (TSP) as well as the Dial-a-Ride (DARP) problem. Both problems have received a lot of attention in combinatorial optimization (see [30, 19] for summaries), which has led to the development of heuristic algorithms as well as exact formulations.

The heuristics employed to solve the VRP fall into the categories of sequential construction routines and iterative improvement procedures (see [31] for a summary). The sequential construction of VRP solutions is usually performed using so-called *cluster-first, route-second* algorithms, which employ different clustering techniques in order to find a partition of the requests into subsets to be served by individual vehicles.

Many widely used approaches to solve VRPs / DARPs to optimality are indeed based on mixed integer programming techniques. Depending on various problem characteristics, different formulations are used by different authors. On the one hand, there are formulations based on arc variables [7]. On the other hand, decompositions including path-based subproblems often yield good results [11]. We have opted to use a path-based approach as well, since we are dealing with the fact that the physical model used for the fuel consumption of the feeder aircraft makes arc-based formulations impractical, c.f. Section 2.

A notable extension of the VRP incorporates time windows into the routing problem: Each request must be served within a certain time window, thereby restricting the set of feasible tours (see [10, 28]). We would like to point out that while the air-to-air refueling problem can be seen as a special case of time-window based VRPs (where the time-windows are fixed to single points), we chose to employ a different approach in order to handle time dependence. Specifically, the fixed refueling times allow us to

incorporate the time dependence into the underlying graph without having to perform a complete time expansion.

More recently, the VRP has been reexamined from an environmental standpoint. Instead of minimizing the travel time and/or vehicle utilization costs, attention has been turned to lowering the carbon footprint of vehicle fleets. To decrease fuel consumption, the authors of [13] include fuel consumption/refueling of street vehicles into the VRP. The authors model fuel consumption as being given in terms of a fixed rate of gallons per mile and consider the problem of minimizing the total mileage, comparing the computational performance of several heuristics with that of a MIP solver based on a two-index formulation.

Another area of research focuses on the VRP of electric vehicles (see [41, 23]). Specifically, electric vehicles must be charged at charging stations, which are still sparsely distributed in many road networks. Therefore the authors of [41] include charging stations as separate vertices in the VRP. Part of the difficulty lies in the fact that the limited battery charge requires frequent charging stops. What is more, recharging times of electric vehicles can range to more than one hour, resulting in significant downtimes and necessitating careful planning. The energy consumption for road segments is given as a fixed value regardless of mass, battery charge or alike, making the problem in some sense complementary to ours.

1.3. Underlying Scenario

To generate an air-to-air refueling problem, an underlying cruiser scenario is needed. The cruiser network for this scenario is based on the transatlantic traffic on 7/1/2011, extracted from Eurocontrol data. A more detailed description can be found in [34]. The following will give a short overview on the scenario design: To generate benefits with air-to-air refueling in civil aviation, a long range flight has to be replaced by an aircraft constructed for shorter ranges. Currently no aircraft fits the needed layout. Thus the aircrafts used in the scenario have been designed for this purpose (see [16]). Furthermore, a reference aircraft has been devised to calculate the fuel consumption of a direct flight without air-to-air refueling.

Today's feeder aircrafts are multi-role aircrafts and are therefore constructed for refueling and as transport aircrafts. To benefit from civil air-to-air refueling, the feeder aircraft has to be designed specifically for this task. The feeder used in the following scenario is a joint wing tanker model from TU Delft [16] for the use in civil air-to-air refueling.

The scenario uses eight feeder bases as point of origin for the feeder aircraft. These feeder aircraft could refuel the cruiser aircraft at any point within the range of the feeder. The refueling point has been optimized to minimize the fuel consumption of the cruiser and an idealized feeder [34]. Between the airports and the refueling point, all aircrafts fly direct great circle routes. For comparison, the reference aircraft also use direct routes.

In the original simulation, the feeders' scheduling has been based on a first-come-first-serve method. Airborne feeders have been prioritized over feeders on the ground to reduce the necessary number of feeders. Thus the feeder scheduling has neither been optimized for fuel consumption nor to minimize the number of feeders at any base. Solving these problems will provide a clearer picture of the necessary resources on the feeder side.

2. Physical Model

To keep calculation times within reasonable boundaries, the model used for the fuel calculation employs some simplifications. During the whole flight the lift over drag ratio L/D is constant and differs only between the different aircraft types. The used L/D is the L/D in the cruise phase and the optimal altitude. As the feeder aircraft has been designed for refueling, the refueling altitude is its optimal altitude. While the cruiser aircraft spends most time at cruiser altitude, the feeder aircraft spends significant time in climb and descent. Thus, the calculated fuel consumption will be slightly lower than the real fuel consumption.

The specific fuel consumption sfc is assumed to be constant as well, which leads to the same effect as mentioned with the L/D . Furthermore, the fuel calculation assumes constant flight conditions. Thus, the additional fuel spent during acceleration is not taken into account. Finally, the aircrafts fly on great circle routes and turn instantly. The additional fuel spend during the turn is again not part of the fuel calculation.

Let M denote the fuel mass of the feeder as a function of the time θ throughout the different flight phases. Fuel is consumed at a rate which depends linearly on the total weight of the aircraft, i.e., there

exists a constant c such that

$$\dot{M}(\theta) = c \cdot (M^{\text{ac}} + M(\theta)).$$

In the following we will go into details regarding different operational phases and their respective fuel consumptions. We would like to point out that the resulting linear ODE can be solved explicitly using exponential functions (for an overview on ODEs, see [21]). Note that this model is widely used in the area of aeronautical engineering, yielding the so-called *Breguet range equation*. For more details on the topic, see [2, Chapter 5.12].

Free flight

During free flight the only cause for a change in fuel mass is the fuel burn of the feeder itself. The fuel burn depends on the total mass of the feeder, given by $M(\theta) + M^{\text{ac}}$. Additional parameters are the flight speed v of the feeder, its lift over drag ratio L/D , and its specific fuel consumption sfc . The fuel burn is then given as

$$\dot{M}(\theta) = -\frac{M^{\text{ac}} + M(\theta)}{L/D} \cdot sfc. \quad (1)$$

For a given fuel mass M^0 at a time θ_0 , this yields the solution

$$M(\theta) = (M^0 + M^{\text{ac}}) \cdot \exp\left(\frac{-(\theta - \theta_0)v}{X}\right) - M^{\text{ac}}, \quad (2)$$

where $X := v \cdot (L/D)/sfc$ denotes the *aircraft efficiency* (see [36]).

Climb

The climb of a feeder from its base involves the ascent to the cruising altitude of the cruisers to be refueled. We assume that the ascent is conducted at a fixed rise angle γ . To account for the increased fuel burn during the climb, we adjust the lift over drag ratio to

$$(L/D)' := \frac{1}{1/(L/D) \cos(\gamma) + \sin(\gamma)}.$$

This leads to a decreased efficiency X^{climb} in the otherwise unmodified Equation (2). The constant angle γ translates into a fixed distance d^{climb} and a corresponding duration $\Delta\theta^{\text{climb}} := d^{\text{climb}}/v$ in order to reach the required altitude. As a result, feeders can begin serving requests only after the minimum climb duration of $\Delta\theta^{\text{climb}}$ after takeoff.

With respect to the distances, we assume that if the distance between the base and the initial position of the initial cruiser to be refueled is less than d^{climb} , the climb is flown in a suitable pattern connecting base and request. If on the other side the distance between base and cruiser is more than d^{climb} , then the *advance* of the feeder towards the cruiser consists of an initial climb followed by a free flight phase with a sufficiently long duration.

Descent

The descent phase is in principle no different than the free flight phase itself, except for the fact that the final descent to the base can be executed in gliding flight while the engines are executed in idle speed. We denote this gliding distance by d^{gliding} . During the gliding phase, fuel is burnt at a constant rate ρ^{gliding} independent of the mass of the feeder. If the distance from the feeder's current position to the base does not exceed d^{gliding} , then the entire distance is traversed gliding. Otherwise, an initial free flight phase is executed to get within a distance of d^{gliding} to the base.

Refueling

The refueling operation itself is the most complex of the different phases. It is conducted in three steps. First, the feeder approaches the cruiser and connects its refueling boom to the cruisers fuel receptacle. We let $\Delta\theta^{\text{approach}}$ be the corresponding *approach duration*. As soon as the contact is established, fuel is pumped from feeder to cruiser at a constant rate. The rate is chosen such that the *wet contact duration* $\Delta\theta^{\text{contact}}$ is fixed. After the fuel transfer is completed, the boom is disconnected and the feeder retreats from the cruiser, requiring a *retreat duration* $\Delta\theta^{\text{retreat}}$. Since the entire refueling maneuver is executed at constant speed v , the different durations translate into equivalent distances. The approach

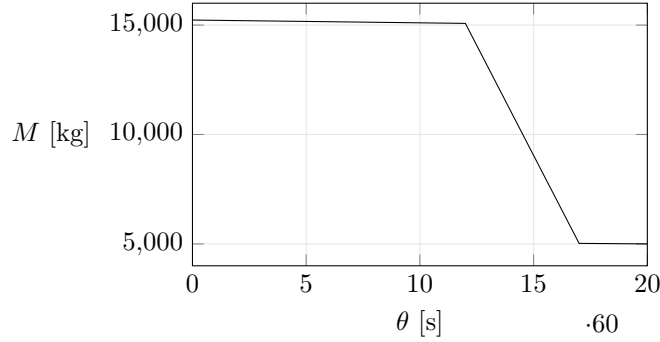


Figure 1: The fuel mass of a feeder during a refueling operation including approach, wet contact, and retreat. Requested fuel mass: $M^{\text{req}} = 10\,000$ kg, final fuel mass: 5 000 kg, parameters according to Table 1.

towards and retreat from the cruiser correspond to free flights (where the difference in fuel mass is given by Equation (2)). The pumping rate is determined by the requested amount of fuel $M^{\text{req}}(r)$, and the contact duration θ^{contact} . The change in fuel mass during contact time is therefore given by

$$\dot{M}(\theta) = - \left(\frac{M^{\text{ac}} + M(\theta)}{L/D} \cdot sfc + \frac{M^{\text{req}}(r)}{\Delta\theta^{\text{contact}}} \right),$$

implying that the fuel mass during contact time is equal to

$$M(\theta) = \left(M^0 + M^{\text{equiv}}(r) \right) \cdot \exp \left(\frac{-v \cdot \Delta\theta^{\text{contact}}}{X} \right) - M^{\text{equiv}}(r), \quad (3)$$

i.e., the solution of the free flight equation of type (1) with an *equivalent* mass of

$$M^{\text{equiv}}(r) := M^{\text{ac}} + X \frac{M^{\text{req}}(r)}{v \cdot \Delta\theta^{\text{contact}}}.$$

For a given value M^0 , we compute the fuel M^{contact} at the beginning of the contact using (2), which we then substitute into (3) as an initial value (thereby preserving continuity), yielding a value M^{retreat} at the beginning of the retreat phase, which we substitute into equation (2) again, to compute the fuel mass at the end of the refueling maneuver. An example of the fuel mass during a refueling operation is shown in Figure 1 (observe the rapid change during the wet contact). The entire refueling operation has a duration of $\Delta\theta^{\text{refuel}}$ defined as

$$\Delta\theta^{\text{refuel}} := \Delta\theta^{\text{approach}} + \Delta\theta^{\text{contact}} + \Delta\theta^{\text{retreat}}.$$

Base refueling

The refueling of the feeder itself is conducted at the base before the feeder takes off to subsequently serve a number of cruisers. We assume that the base refueling operation takes a fixed duration of $\Delta\theta^{\text{base,refuel}}$ independent of the refueling amount.

Initial fuel mass and fuel burn

We can use the established behavior of the fuel mass during the different phases in order to define an *initial fuel mass* function $\mu : \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$, describing the initial fuel mass depending on the final fuel mass M^{final} and the duration $\Delta\theta$ of an operational phase. In a similar fashion, we let $\Delta\mu : \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$ be the *fuel burn function*, i.e., the fuel that is consumed by the feeders themselves. For any free flight, climb, or descent, the fuel burn coincides with the fuel difference, whereas during the wet contact between feeder and cruiser, the burned fuel is equal to the fuel difference minus the requested amount of fuel.

Table 1: Model parameters

| | Description | Symbol | Value | |
|------------------|----------------------------|-------------------------------------|---------|--------------------|
| Feeder | Maximum Take-off Weight | M^{takeoff} | 62 933 | kg |
| | Operational Empty Weight | M^{ac} | 14 881 | kg |
| | Maximum fuel mass | M^{max} | 42 456 | kg |
| | Efficiency | X | 18 393 | NM |
| | Speed | v | 240.3 | m s^{-1} |
| Climb | Required climbing distance | d^{climb} | 87.2 | km |
| | Climbing efficiency | X^{climb} | 6 621.5 | NM |
| Descent | Gliding distance | d^{gliding} | 156.8 | km |
| | Gliding fuel rate | ρ^{gliding} | 160 | kg h^{-1} |
| Durations | Approach duration | $\Delta\theta^{\text{approach}}$ | 12 | min |
| | Contact duration | $\Delta\theta^{\text{contact}}$ | 5 | min |
| | Retreat duration | $\Delta\theta^{\text{retreat}}$ | 3 | min |
| | Base refueling duration | $\Delta\theta^{\text{base,refuel}}$ | 30 | min |

Remark 2.1 (Monotonicity). It is easy to see that both μ and $\Delta\mu$ are non-increasing in both their arguments. The advantages of this observation are twofold: On the one hand it is sufficient to consider feeders arriving back at the base without any fuel to spare. On the other hand, the functions μ and $\Delta\mu$ agree with the notion of metric distances: It is optimal for the feeders to spend as little time in the air as possible.

The inclined reader can find the formal definitions of both functions in Appendix A. All relevant parameter values used by us are listed in Table 1.

3. Formulations

We formulate the refueling problem as a covering problem based on paths through a *refueling graph* $D = (V, A)$. To this end, let $r \in \mathcal{R}$ be some request. We begin by defining the takeoff, landing and *base refueling* time of r as

$$\begin{aligned}\theta^{\text{takeoff}}(r) &:= \theta(r) - \max\left(\Delta\theta^{\text{climb}}, \frac{d(b, \text{orig}(r))}{v}\right), \\ \theta^{\text{landing}}(r) &:= \theta(r) + \Delta\theta^{\text{refuel}} + \frac{d(\text{dest}(r), b)}{v}, \text{ and} \\ \theta^{\text{base,refuel}}(r) &:= \theta^{\text{takeoff}}(r) - \Delta\theta^{\text{base,refuel}}.\end{aligned}$$

The vertices V of the refueling graph are given as $V := V^{\text{tail}} \cup V^{\text{head}} \cup V^{\text{base}}$, where

$$\begin{aligned}V^{\text{tail}} &:= \{r^{\text{tail}} \mid r \in \mathcal{R}\}, \quad V^{\text{head}} := \{r^{\text{head}} \mid r \in \mathcal{R}\}, \text{ and} \\ V^{\text{base}} &:= \{r^{\text{takeoff}} \mid r \in \mathcal{R}\} \cup \{r^{\text{landing}} \mid r \in \mathcal{R}\} \cup \{r^{\text{base,refuel}} \mid r \in \mathcal{R}\}.\end{aligned}$$

Every vertex $u \in V$ has an associated time $\theta(u)$, given as

$$\theta(u) := \begin{cases} \theta^{\text{takeoff}}(r) & \text{if } u = r^{\text{takeoff}} \in V^{\text{base}}, \\ \theta^{\text{landing}}(r) & \text{if } u = r^{\text{landing}} \in V^{\text{base}}, \\ \theta^{\text{base,refuel}}(r) & \text{if } u = r^{\text{base,refuel}} \in V^{\text{base}}, \\ \theta(r) & \text{if } u = r^{\text{tail}} \in V^{\text{tail}}, \text{ and} \\ \theta(r) + \Delta\theta^{\text{refuel}} & \text{if } u = r^{\text{head}} \in V^{\text{head}}. \end{cases}$$

We can use this definition to order the vertices in V^{base} according to their times via $s := v_1^{\text{base}}, \dots, v_N^{\text{base}} :=: t$, where $\theta(v_1) < \dots < \theta(v_N)$. We say that a request $\hat{r} \in \mathcal{R}$ is *reachable* from r iff the speed v is sufficient

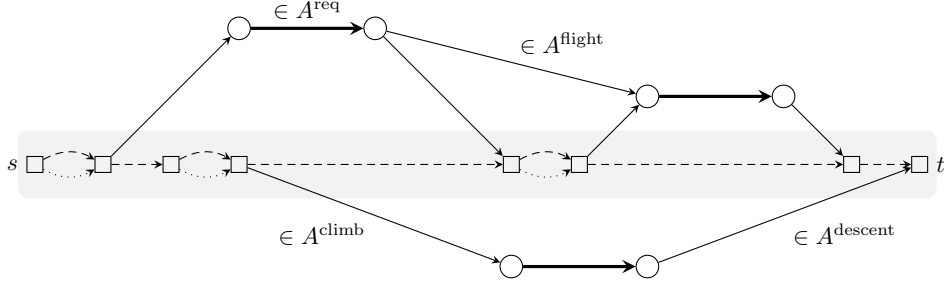


Figure 2: A refueling graph used to model a refueling problem with three requests. Refueling arcs A^{req} are drawn thick, base waiting arcs A^{wait} dashed, base refueling arcs dotted, and transit arcs A^{transit} solid. The base itself is shaded.

to reach \hat{r} after serving r , i.e.,

$$\theta(r) + \Delta\theta^{\text{refuel}} + \frac{d(\text{orig}(r), \text{dest}(\hat{r}))}{v} \leq \theta(\hat{r}).$$

The arcs A are defined as $A := A^{\text{req}} \cup A^{\text{climb}} \cup A^{\text{descent}} \cup A^{\text{flight}} \cup A^{\text{base,refuel}} \cup A^{\text{wait}}$, where

$$\begin{aligned} A^{\text{req}} &:= \{(r^{\text{tail}}, r^{\text{head}}) \mid r \in \mathcal{R}\}, \\ A^{\text{climb}} &:= \{(r^{\text{takeoff}}, r^{\text{tail}}) \mid r \in \mathcal{R}\}, \\ A^{\text{descent}} &:= \{(r^{\text{head}}, r^{\text{landing}}) \mid r \in \mathcal{R}\}, \\ A^{\text{flight}} &:= \{(r^{\text{head}}, \hat{r}^{\text{tail}}) \mid r, \hat{r} \in \mathcal{R} \text{ and } \hat{r} \text{ is reachable from } r\}, \\ A^{\text{base,refuel}} &:= \{(r^{\text{base,refuel}}, r^{\text{takeoff}}) \mid r \in \mathcal{R}\}, \text{ and} \\ A^{\text{wait}} &:= \{((v_k^{\text{base}}, v_{k+1}^{\text{base}})) \mid k = 1, \dots, N-1\}. \end{aligned}$$

Since all coordinates are pairwise different, and all travel times are positive, the refueling graph D is acyclic and every arc $a := (u, w)$ has an associated time window $\Delta\theta(a) := [\theta(u), \theta(w))$. As a result, we can (with a slight abuse of notation) extend the initial fuel function μ to map from $A \times \mathbb{R}_{\geq 0}$ to $\mathbb{R}_{\geq 0}$. Specifically, for an arc $a \in A$, we let $\mu(a, \cdot) := \mu(\Delta\theta(a), \cdot)$. The same holds for the fuel burn function $\Delta\mu$. For notational convenience, we also define the set of *transit arcs* as $A^{\text{transit}} := A^{\text{climb}} \cup A^{\text{descent}} \cup A^{\text{flight}}$. An example of a refueling graph is depicted in Figure 2.

Remark 3.1. Our construction shares some similarity with a time-expansion of a regular vehicle routing problem. Indeed, the contraction of the request arcs and all waiting / base refueling arcs leads to a graph with one depot (the contraction of the base arcs, i.e., the base) and $|\mathcal{R}|$ many customers.

However, apart from the base, no other vertices are expanded in the sense of being copied multiple times, largely due to the fact that the requests are fixed in time. As a result, we can avoid discretizing the time horizon and manage to find optimal solutions for all problems within a reasonable amount of time (see Section 4).

Objective functions

As mentioned in the introduction, our goal in the air-to-air refueling problem is to reduce operating costs, determined by both fuel consumption and fleet size. In the former case, the objective function is given by the fuel burn function $\Delta\mu$. In the latter case, every feeder simply contributes a cost of one. Equivalently, we can define the cost function based on the arcs of the refueling graphs, by assigning a value of one to all arcs in $\delta^+(s)$, and zero to all others. In any case, the monotonicity assumption from Remark 2.1 is justified. In the following, we will use c to denote the objective function, understanding that c is either $\Delta\mu$ or the number of feeders.

Feasible paths

In the following, we define the initial fuel of a path $P := (a_1, \dots, a_{k-1})$ with arcs $a_i := (u_i, u_{i+1})$, $i = 1, \dots, k-1$, and vertices $V(P) := \{u_i \mid i = 1, \dots, k\}$. Specifically, the initial fuel mass $\mu_P : V(P) \rightarrow \mathbb{R}_{\geq 0}$

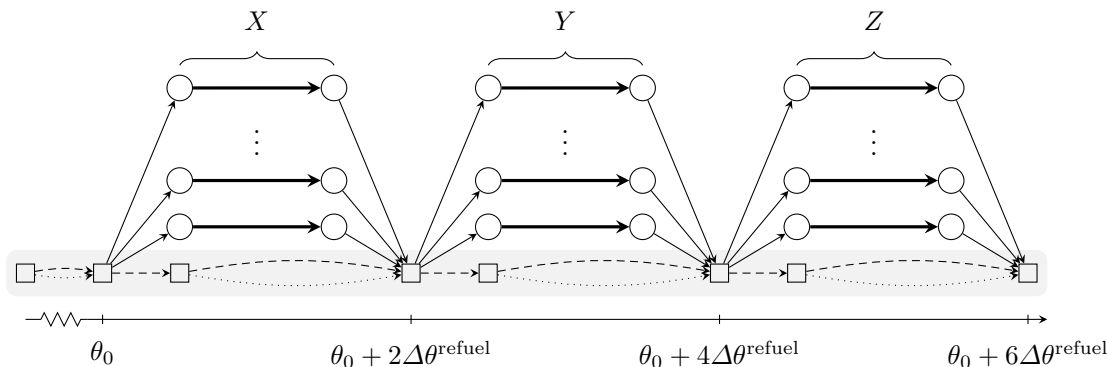


Figure 3: The refueling graph required for the proof of Theorem 3.2.

at each vertex can be defined recursively via

$$\mu_P(u_i) := \begin{cases} 0 & \text{if } i = k, \\ \mu(a_i, \mu_P(u_{i+1})) & \text{if } i < k. \end{cases}$$

We call a path P *feasible* iff $\mu_P(u) \leq M^{\max}$ for all $u \in V(P)$ and denote the set of feasible (s, t) -paths by \mathcal{P} . Note that there is a one-to-one correspondence between the trajectories of feeders and feasible paths through the refueling graph. We define the fuel burn of a path in a similar fashion:

$$\Delta\mu_P(u_i) := \begin{cases} 0 & \text{if } i = k, \\ \Delta\mu(a_i, \mu_P(u_{i+1})) & \text{if } i < k. \end{cases}$$

3.1. Complexity

In the following, we will show the hardness of the air-to-air refueling problem:

Theorem 3.2. *The problem of minimizing the number of feeders of an air-to-air refueling problem is \mathcal{NP} -hard in the strong sense even if all transit arcs are fixed.*

Proof. We reduce from the \mathcal{NP} -complete NUMERICAL 3-DIMENSIONAL MATCHING (N3DM) problem introduced in [18]. An instance of N3DM is given by disjoint sets X , Y , and Z , each of cardinality q , a weight function $s : X \cup Y \cup Z \rightarrow \mathbb{Q}_{>0}$, and a budget $B \in \mathbb{Q}_{>0}$, such that $\sum_{a \in X \cup Y \cup Z} s(a) = qB$. The problem asks for a partition of $X \cup Y \cup Z$ into q sets, each containing exactly one element from each X , Y , and Z , where the sum of the weights of each set is equal to B .

We will construct an instance of the air-to-air refueling problem which has a feasible solution with q feeders iff the given N3DM instance is feasible. To this end, let $\mathcal{R} := X \cup Y \cup Z$ be the set of requests. We fix the position of the base and place the origins / destinations of the requests on opposite endpoints of a diameter of a circle with radius $\varrho := \Delta\theta^{\text{refuel}}v/2$ (≈ 288 km) around the base. This choice of ϱ implies that the refueling operation can be carried out between those endpoints. For $r \in \mathcal{R}$ we let

$$\theta(r) := \begin{cases} \theta_0 + 1/2\Delta\theta^{\text{refuel}} & \text{if } r \in X, \\ \theta_0 + (2 + 1/2)\Delta\theta^{\text{refuel}} & \text{if } r \in Y, \text{ and} \\ \theta_0 + (4 + 1/2)\Delta\theta^{\text{refuel}} & \text{if } r \in Z \end{cases}$$

for some fixed θ_0 . Note that the radius permits the feeders to perform the climb during the advance towards individual requests (since $d^{\text{climb}} = 87.2$ km $< \varrho$). We now fix the transit arcs in order to force the feeders to serve the requests without any flight arcs, yielding the refueling graph shown in Figure 3.

We go on to give values for the requested fuel masses M^{req} . To this end, we note that for fixed values of $\Delta\theta$, as is the case in our construction, the initial fuel function μ is affine in the final fuel mass M^{final} during the flight, advance, refueling, and descent phase. What is more, μ is also an affine function with respect to the requested fuel mass M^{req} in the case of the refueling phases (this is easily obtained from the formal definitions in Appendix A). Since the final fuel mass of one flight phase is equal to the initial

fuel mass of the succeeding phase, the total amount of required fuel in order to serve requests $x \in X$, $y \in Y$, and $z \in Z$ is of the form

$$\alpha_x M^{\text{req}}(x) + \alpha_y M^{\text{req}}(y) + \alpha_z M^{\text{req}}(z) + \beta$$

for positive constants α_x , α_y , α_z , and β . In particular, the value of β corresponds to the burned fuel mass for a series of empty requests. From the parameters in Table 1, it is easily calculated that $\beta \approx 873 \text{ kg} < M^{\text{max}}$. Note that while these constants may not be expressible as rational numbers, for considerations of complexity we can round them to rational numbers of sufficient precision.

Observe that it is possible to scale the N3DM instance (i.e., both B and s) by a positive factor to obtain an equivalent decision problem. Hence, we scale the instance such that $B = M^{\text{max}} - \beta$ and define the requested fuel mass for $r \in \mathcal{R}$ as

$$M^{\text{req}}(r) := \begin{cases} 1/\alpha_x \cdot s(x) & \text{if } r \in X, \\ 1/\alpha_y \cdot s(y) & \text{if } r \in Y, \text{ and} \\ 1/\alpha_z \cdot s(z) & \text{if } r \in Z. \end{cases} \quad (4)$$

It is easy to see that there is a one-to-one correspondence between feasible solutions of the matching instance and solutions of the air-to-air refueling instance consisting of exactly q feeders: On the one hand, given a partition of $X \cup Y \cup Z$ into sets, each consisting of x , y , and z from the respective sets, we assign the three requests to a feeder. By the choice of M^{req} , the tour must be feasible.

On the other hand, consider a feasible solution of the air-to-air refueling problem with at most q feeders. Since the requests in X (as well as in Y and Z) are parallel in time, no feeder can serve more than three requests. Since there are $3q$ requests, every feeder must serve exactly one request from each X , Y , and Z . The fact that the tours of the feeders are feasible and the choice of M^{req} ensure that the solution corresponds to a feasible N3DM solution.

Lastly, recall that N3DM is \mathcal{NP} -hard in the strong sense, i.e., the size of the numbers $s(\cdot)$ is bounded by a polynomial in the input length of the given instance. The values $M^{\text{req}}(\cdot)$ are obtained from s and B via an initial scaling with a factor of $(M^{\text{max}} - \beta)/B$, followed by an application of (4). Since the respective factors are constant, the sizes of $M^{\text{req}}(\cdot)$ remain polynomially bounded. \square \square

Corollary 3.3. *Minimizing a function $c : \mathcal{P} \rightarrow \mathbb{R}_{\geq 0}$ over an air-to-air refueling instance is \mathcal{NP} -hard in the strong sense even if all transit arcs are fixed.*

3.2. Formulation as Exact Cover

The problem of minimizing a cost function while serving all refueling requests is equivalent to finding a minimum cost exact cover of the request arcs by feasible paths:

$$\begin{aligned} \min \quad & \sum_{P \in \mathcal{P}} c_P x_P \\ \text{s.t.} \quad & \sum_{P \in \mathcal{P}: a \in P} x_P = 1 \quad \forall a \in A^{\text{req}} \\ & x_P \in \{0, 1\} \quad \forall P \in \mathcal{P}. \end{aligned} \quad (5)$$

We make the following observation:

Lemma 3.4. *Any solution x^* of (5) consists of paths which are arc-disjoint in A^{transit} .*

Proof. Due to (5), the paths must obviously be arc-disjoint in A^{req} . Any arc in $A^{\text{transit}} \setminus A^{\text{req}}$ has at least one endpoint which is incident to a request arc. Let (u, w) be a request arc. Since $\delta^+(u)$ and $\delta^-(w)$ consist only of the request arc (u, w) , the claim follows immediately. \square \square

One problem in solving formulation (5) is the (potentially exponential) number of paths in \mathcal{P} . To overcome this, we employ a column generation technique (for a summary on the approach, see [33]) to generate new path variables as needed. We will give details on the pricing problem below.

In terms of the formulation, we would like to point out that in many cases, including this one, incorporating a column generation approach into a Branch-and-Bound scheme is a nontrivial matter. This is due to the fact that the branching decisions made throughout the traversal of the Branch-and-Bound affects the pricing problems to be solved at individual nodes: Whenever a path variable x_P has

been branched to zero or one, the corresponding additional inequality ($x_P \geq 1$ or $x_P \leq 0$) has to be taken into account during the pricing step, which is highly nontrivial. What is more, the two branches are unbalanced in the following sense: Branching a path P to one forces all paths P' sharing a transit arc with P to zero, thereby substantially reducing the number of variables and likely increasing the dual bound of the corresponding subproblem. Conversely, branching a path P to zero has little effect on the overall problem.

To alleviate this problem, we introduce *compound flow* variables x_a for $a \in A^{\text{transit}}$, where the compound flow over the transit arcs is given by additional *linking equations* of the form

$$\sum_{P \in \mathcal{P}: a \in P} x_P = x_a \quad \forall a \in A^{\text{transit}},$$

which we add to the original problem, obtaining the following formulation:

$$\begin{aligned} \min \quad & \sum_{P \in \mathcal{P}} c_P x_P \\ \text{s.t.} \quad & \sum_{P \in \mathcal{P}: a \in P} x_P = 1 \quad \forall a \in A^{\text{req}} \\ & \sum_{P \in \mathcal{P}: a \in P} x_P = x_a \quad \forall a \in A^{\text{transit}} \\ & x_a \in \{0, 1\} \quad \forall a \in A^{\text{transit}} \\ & x_P \in \{0, 1\} \quad \forall P \in \mathcal{P}. \end{aligned} \tag{6}$$

We then instruct the IP solver to primarily branch on compound flow variables by assigning appropriate branching priorities. However, in light of Theorem 3.2 (the problem is \mathcal{NP} -hard even for fixed transit arcs), we cannot expect to obtain 0/1 solutions even if all flow variables are fully branched out. Indeed, several of the computational examples (see Section 4) do exhibit this behavior: there are several Branch-and-Bound nodes in which the optimal solution of the relaxed subproblem has all transit arcs fixed to 0/1 while still containing fractional path variables. Hence, it is necessary to branch on path variables as well as on compound flow variables, a fact which has to be taken into account in the pricing problem (see below).

Unfortunately, the approach laid out so far poses another, more subtle problem: As we have remarked, refueling graphs tend to be rather dense, containing large numbers of arcs. For this reason, the addition of the compound flow variables and the corresponding equations must be carried out carefully so as not to increase the size of the relaxations beyond what is needed.

However, while we should not slow down the LP solver by a vastly increased problem size, we still want to add a sufficient number of compound flow variables as branching candidates in order to take advantage of the sophisticated branching rules built into most IP solvers. To this end, we opted to generate new compound flow variables as needed in a second column generation step. Specifically, in each Branch-and-Bound node, once all path variables have been generated and we fail to find a path with negative reduced costs, we examine the individual request arcs A^{req} . For each request arc $a(r) := (u, w)$, $r \in \mathcal{R}$, we consider the arcs in $\delta^-(u)$. If some of these arcs carry a fractional flow value with respect to the current relaxation and none of the fractional arcs has an associated compound flow variable, we add one new compound flow variable and associated equation. We found that this approach works quite well in practice and seems to provide the underlying IP solver with a sufficiently large number of substantially different branching candidates while keeping the total problem size in check.

Remark 3.5 (Arc-based formulations). In the context of vehicle routing and dial-a-ride problems, it is quite common to use an arc-based two- or three-index formulation, mostly coupled with a column generation framework (e.g., [39, 11]).

We have opted against such a formulation due to the vast increase in problem size: The refueling graphs for the given instances (see Sect. 4.1) are already very dense, consisting of up to about one million arcs. In order to incorporate the fuel dependent functions μ and $\Delta\mu$, we would have to expand the graph along the different fuel values (which range from 0 to $M^{\text{max}} \approx 40\,000$ kg). Even considering a moderate resolution of steps of 100 kg, this would add 400 layers of copies of the refueling graph, significantly slowing down any column generation approach.

Conversely, feasible path variables nicely capture the physical properties of the given model, while keeping the average number of columns per LP reasonably small (usually below 10 000).

The pricing problem

The pricing problem is to find a new column to be added to the relaxation of a given Branch-and-Bound node. In our case we have to find a new feasible path $P \in \mathcal{P}$ with negative reduced costs while respecting previous branching decisions leading up to the current node in the Branch-and-Bound tree.

First, consider the situation at the root node: Given an optimum solution of the LP-relaxation of (6) for some subset $\tilde{\mathcal{P}} \subseteq \mathcal{P}$ of the paths, find a path P with negative reduced costs or decide that no such path exists.

Both the covering constraints and the linking constraints have dual variables $(\lambda_a)_{a \in A^{\text{req}}}$ and $(\delta_a)_{a \in A^{\text{transit}}}$ yielding the reduced costs of P via

$$\bar{c}_P := c_P - \sum_{a \in P \cap A^{\text{req}}} \lambda_a - \sum_{a \in P \cap A^{\text{transit}}} \delta_a \quad (7)$$

Since both of our objective functions can be expressed as fuel-dependent, arc-based functions, the same holds for the reduced costs. However, the definition of the feasible path set \mathcal{P} includes an upper bound on the fuel consumption, turning the pricing problem into a *Constrained Shortest Path Problem* (CSP).

Despite the fact that the CSP is \mathcal{NP} -hard in general (see [22]), there are a number of algorithms, both IP-based and combinatorial, which solve many real-world problems to optimality within reasonable time (see [5] for a summary on the subject). We implemented a variant of the labeling scheme introduced in [3]; see Algorithm 1. For each vertex $v \in V$, the algorithm maintains a set of labels L_v , where each label $\ell \in L_v$ is a tuple $\ell = (M, c)$ of a fuel mass M and a cost value c . To keep the number of labels as small as possible, we only keep non-dominated labels in each L_v , where $\ell = (M, c)$ is defined to dominate $\ell' = (M', c')$ iff $\ell \neq \ell'$, $M \leq M'$, and $c \leq c'$.

For the most part, our adaptations are due to the nature of the refueling problem: Our problem is complicated by the fact that the cost/resource functions depend on the final fuel mass. On the other hand, the monotonicity of μ and $\Delta\mu$ as well as the fact that D is acyclic work to our advantage. We employed the following changes:

- We used the fact that the refueling graph D is acyclic: It is sufficient to expand labels according to a topological ordering of V to ensure that all non-dominated paths are found.
- Due to the monotonicity of both considered objective functions, we were able to use \underline{c} defined as $c(\cdot, 0)$ as a fuel-independent lower bound of the true cost function c in order to discard suboptimal labels.
- Similarly, we used $\underline{\mu}$, obtained from $\mu(\cdot, 0)$, as a lower bound to identify labels which are not contained in any feasible (s, t) -path.
- We used heuristically found paths to update an upper bound on the cost of the optimal path.

We go on to study the impact of branching decisions on the pricing problem. Firstly, consider the subproblem where some compound flow variable x_a has been fixed. If x_a has been fixed to one, the pricing problem does not change at all: While the fixing affects the LP-solution, the only dependency between paths and compound flow variables is due to the linking constraints, which we already take into account. If on the other hand x_a is fixed to zero, we simply exclude the corresponding arc from consideration during the execution of Algorithm 1, since no path containing a must be added to the current subproblem.

Still, according to Theorem 3.2, the problem of optimizing the number of feeders remains \mathcal{NP} -hard even when all transit arcs are fixed to 0/1 values. Thus, we cannot expect to obtain an integral solution to the corresponding relaxation. This means that, at some point, we may have to branch on path variables and modify the pricing problem accordingly.

If a variable x_P has been fixed to one, we know that the arcs in $P \cap (A^{\text{transit}} \cup A^{\text{req}})$ can be excluded, since they are exclusively used by P in the subproblem. The problematic case occurs when a set $\mathcal{P}^0 \subseteq \mathcal{P}$ of paths has been branched to zero, in which case we have to find a path minimizing (7) not contained in \mathcal{P}^0 , corresponding to a routing problem with *forbidden paths*. The problem of routing with forbidden paths has been examined previously [38], leading to a dynamic programming algorithm, which solves the shortest path problem with forbidden paths (SPPFP) in polynomial time [42].

In the more general case of a CSP with forbidden paths, we cannot hope for a polynomial algorithm. However, we can adapt the dynamic programming algorithm to our use case. Specifically, we detach the forbidden path problem from the CSP by means of a preprocessing step, generating label sets L_v corresponding to the paths in $\mathcal{P} \setminus \mathcal{P}^0$ for each vertex $v \in V$. We then adapt Algorithm 1 to accept the

Algorithm 1: A labeling scheme used to find constrained shortest paths

Input : Directed acyclic graph $D = (V, A)$,
 Topological ordering $s = u_1, \dots, u_n = t$ of D
 Costs $c : A \times \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$, Cost bounds $\underline{c} : V \times V \rightarrow \mathbb{R}_{\geq 0}$,
 Initial fuel function $\mu : A \times \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$,
 Initial fuel bound $\underline{\mu} : V \times V \rightarrow \mathbb{R}_{\geq 0}$

Output: Feasible (s, t) -path P_{opt} with minimum cost with respect to c
 $L_u \leftarrow \emptyset$ for all $u \in V \setminus \{t\}$, $L_t \leftarrow \{(0, 0)\}$
 $(c_{\text{opt}}, P_{\text{opt}}) \leftarrow (\infty, \emptyset)$

for $j \leftarrow n$ **downto** 1 **do**

- foreach** label $\ell_j \leftarrow (M^j, c_j) \in L_{u_j}$ **do** $\triangleright M^j, c_j$: fuel, cost of label ℓ_j
 - $P_j \leftarrow$ the (u_j, t) -path corresponding to ℓ_j
 - foreach** $a \leftarrow (u_i, u_j) \in \delta^-(u_j)$ **do**
 - $M^i \leftarrow \mu(a, M^j)$
 - $c_i \leftarrow c_j + c(a, M^j)$
 - if** $c_i + \underline{c}(s, u_i) \geq c_{\text{opt}}$ **then continue**
 - if** $M^i + \underline{\mu}(s, u_i) > M^{\text{max}}$ **then continue**
 - else**
 - $P \leftarrow$ composition of the path achieving $\underline{c}(s, u_i)$, arc a , and P_j
 - if** P is feasible and $c(P) < c_{\text{opt}}$ **then**
 - $(c_{\text{opt}}, P_{\text{opt}}) \leftarrow (c(P), P)$
 - if** (M^i, c_i) is not dominated in L_{u_i} **then**
 - Insert (M^i, c_i) into L_{u_i} , remove newly dominated labels from L_{u_i}

foreach label $\ell_s \leftarrow (M, c) \in L_s$ **do**

- $P \leftarrow (s, t)$ -path corresponding to ℓ_s
- if** $c(P) < c_{\text{opt}}$ **then** $(c_{\text{opt}}, P_{\text{opt}}) \leftarrow (c(P), P)$

return P_{opt}

sets L_v as initial labels during its otherwise unmodified execution. In the case where \mathcal{P}^0 is empty, we simply generate a single label for vertex t instead, skipping the preprocessing step.

For the preprocessing step, consider first the case of a single forbidden path P^0 and a feasible path P (i.e., a path not equal to P^0). If we follow P from t to s , we see a (possibly empty) sequence of arcs shared by P and P^0 up to some vertex $v \neq s$ where the paths split. More formally, there must be some arc $a = (u, v)$ in $\delta^-(v)$ in $P \setminus P^0$ such that P and P^0 have the same (v, t) -subpath. Therefore, it is sufficient to create labels for every such vertex u , where costs and initial fuel values correspond to the composition of the (v, t) -subpath of P^0 and the arc a . If \mathcal{P}^0 consists of a set of internally disjoint subpaths, the situation is quite similar, we simply add labels for each of the paths one after another.

Lastly, consider the general case of several paths in \mathcal{P}^0 not being internally disjoint. In this case, a vertex v in $V \setminus \{s, t\}$ may be contained in a set $\mathcal{P}_v \subseteq \mathcal{P}^0$ of multiple forbidden paths. Before creating a label based on an incoming arc $a = (u, v)$ for a path $P \in \mathcal{P}_v$, we have to ensure two things. Firstly, a must not be in P . Secondly, there must not be a path $P' \in \mathcal{P}_v$ containing a and sharing its (v, t) -subpath with P . Thus, we order the paths in \mathcal{P}_v into buckets according to their (v, t) -subpaths and create labels for each bucket and each arc in $\delta^-(v)$ not contained in any of the paths of the current bucket. The unique (v, t) -subpath corresponding to the bucket is used to assign a cost / initial fuel value to the newly created labels.

Remark 3.6 (Pricing performance). We found that despite of the complexity of the pricing problem, the labeling scheme introduced to solve the CSP is working efficiently in practice. Even for larger instances (see Section 4), less than 20% of the computation time is spent on pricing variables.

One might ask whether a heuristic pricing approach could decrease the time spent to generate new variables. Specifically, it is sufficient to find suboptimal paths, as long as their reduced costs are negative. Algorithm 1 lends itself quite well as a heuristic: If we simply return the first path with negative reduced costs, we can improve the performance of the individual iterations of the pricing procedure.

However, using Algorithm 1 heuristically increases the total computation time significantly. Apparently, the decrease in computation time of the pricing procedure is nullified by the worse quality of the computed paths. What is more, during the majority of pricing iterations, especially when they occur at nodes lower in the Branch-and-Bound tree, no paths are found. Instead, the optimality of the given LP solution is proven, meaning that heuristic paths simply do not exist.

3.3. Formulation as Set Cover

Note that formulation (5) is a textbook case of a set partitioning problem. It is generally preferred, whenever possible, to relax the partitioning problem to a set covering problem instead. This is due to the fact that set covering problems are better understood from a theoretical point of view as well as more tractable in practice (see [6] for an in-depth explanation). The set covering relaxation of (6) is given as

$$\begin{aligned}
\min \quad & \sum_{P \in \mathcal{P}} c_P x_P \\
\text{s.t.} \quad & \sum_{P \in \mathcal{P}: a \in P} x_P \geq 1 \quad \forall a \in A^{\text{req}} \\
& \sum_{P \in \mathcal{P}: a \in P} x_P = x_a \quad \forall a \in A^{\text{transit}} \\
& x_a \in \{0, 1\} \quad \forall a \in A^{\text{transit}} \\
& x_P \in \{0, 1\} \quad \forall P \in \mathcal{P}.
\end{aligned} \tag{8}$$

We proceed to show that the covering relaxation (8) does indeed yield solutions with the same objective function value for both objective functions. To this end, we consider *shortcut paths*. Let P be a path serving a request $r \in \mathcal{R}$ and let $a, a(r), a'$ be the corresponding sequence of arcs connecting vertices u and w (see Figure 4). The shortcut path P' is defined as follows: If both u and w are base vertices, say b_θ and $b_{\theta'}$ with $\theta < \theta'$, we know from the construction of the refueling graph that there is a path of waiting arcs connecting b_θ and $b_{\theta'}$. We replace the sequence $a, a(r), a'$ by that path. If either u or w are base vertices, we form P' by delaying the takeoff or advancing the landing and inserting some waiting arcs at appropriate positions. If both u and w are request vertices, we replace the sequence $a, a(r), a'$ by the arc (u, w) , which is guaranteed to exist since w is reachable from u .

Lemma 3.7. *Let P be a path, P' its shortcut along the sequence $a, a(r), a'$ for $r \in \mathcal{R}$. Then P' is feasible as well. The cost of P' with respect to $\Delta\mu$ does not increase.*

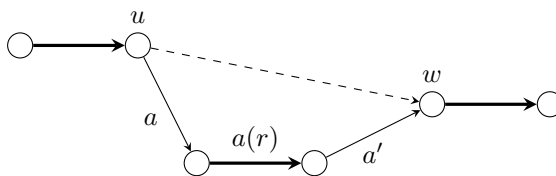


Figure 4: The dashed arc is a shortcut in a sequence of refueling arcs.

Proof. If the sequence $a, a(r), a'$ is replaced by a single flight arc, feasibility follows from the fact that the distance function d is metric, and that the initial fuel function μ is non-increasing in the flight duration (see Remark 2.1).

Regarding the objective function, we observe the following: The monotonicity of $\Delta\mu$ in the flight duration ensures that the cost of the flight arc is no more than that of the sequence $a, a(r), a'$. The monotonicity in the final fuel mass ensures that the costs of the arcs in P' preceding a does not increase.

Similar arguments can be made in case of delayed takeoffs / advanced landings, and of course in the case where a takeoff / refueling / landing sequence is replaced by a sequence of waiting arcs. \square \square

Note that trivially, the number of paths does not increase when we replace P by P' , so we can also use shortcuts when minimizing the number of feeders. We can therefore choose to solve formulation (8) instead of (6) and obtain equivalent solutions.

A simple primal heuristic

Consider the problem of minimizing the number of feeders required to serve all cruisers (i.e., $c_P \equiv 1$). In this case, we can utilize the well-known greedy heuristic for the set covering problem [25] in order to obtain an initial feasible integer solution. Specifically, we perform the following steps:

1. Let $k \leftarrow 1, A' \leftarrow \emptyset$.
2. Let P_k be the feasible path in $(V, A \setminus A')$ containing the maximum number of refueling arcs.
3. Set $A' \leftarrow A' \cup A(P_k), k \leftarrow k + 1$.
4. If there remain uncovered refueling arcs, go to 1.

We find the paths P_k by using a slight adaptation of Algorithm 1: Firstly, we use a cost function c which is -1 on A^{req} and 0 otherwise. Secondly, we restrict the search to refueling arcs not yet covered by the paths P_1, \dots, P_{k-1} , simply by ignoring all arcs in A' during the search. Note that the path computations become increasingly easier as more refueling arcs are covered.

3.4. Fundamental Path Graph Formulation

In this subsection, we will discuss some of the drawbacks of the previously defined formulation (6), and derive an improved formulation. The main shortcoming of (6) is rooted in the fact that the formulation does not include any information about whether or not different paths can be combined to be served by a single feeder. Instead of combining a set of paths, it is necessary to find the combination in a subsequent pricing step. This makes it very hard for any off-the-shelf heuristic to find feasible solutions. Similarly, the larger number of variables increases the computation time required to solve the occurring LP-relaxations and to conduct pricing steps. The problem becomes more pronounced if paths consist of a large number of subpaths.

In the following, we will make the notion of compatibility between different paths more formal. To this end, consider a feeder traversing the graph D along a (nontrivial) path $P \in \mathcal{P}$: After an initial waiting and refueling step, the feeder takes off to serve requests (following a subpath of arcs in $A^{\text{transit}} \cup A^{\text{req}}$). It then returns to the base for an intermediate waiting / refueling period, serves some more request, and so on. An intermediate period corresponds to a sequence of base vertices $b_\theta, \dots, b_{\theta'}$, where $\Delta\theta := \theta' - \theta > 0$. If $\Delta\theta < \Delta\theta^{\text{base,refuel}}$, then the feeder is simply waiting at the base before advancing towards another request. Otherwise, we can assume w.l.o.g. that the feeder is refueling right before taking off at θ' and waiting during $[\theta, \theta' - \Delta\theta^{\text{base,refuel}})$. Formally, we make the following definition:

Definition 3.8 (Fundamental Path). A path P is called *fundamental* iff it has the following properties:

- P begins with a base refueling arc and ends with a descent arc.

- All other arcs of P are in $A^{\text{transit}} \cup A^{\text{req}} \cup A^{\text{wait}}$.
- For all $a \in (P \cap A^{\text{wait}})$ it holds that $\Delta\theta(a) < \Delta\theta^{\text{base,refuel}}$.

Remark 3.9. A subject related to fundamental paths is the concept of *partial paths* (see [37, 35]), in which a vehicle routing problem is decomposed into paths with restricted length in terms of the number of their arcs. Variables are introduced for these partial paths, usually combined with a column generation approach, and the composition of partial paths into proper paths is included in the model. This approach has the upside of working with any type of vehicle routing problem, whereas the concept of fundamental paths is applicable to time-dependent VRPs. Fundamental paths on the other hand make it possible to handle the composition of fundamental paths into tours in a largely implicit fashion while requiring only a few additional constraints.

We can decompose any arbitrary path P into subpaths at vertices which are sources of base refueling arcs. We then remove trailing waiting arcs from the subpaths and denote the subpaths by P_1, \dots, P_k . It is easy to see that the paths P_i , $i = 1, \dots, k$, are fundamental paths.

Each fundamental path P_i with source vertex s_i and target vertex t_i has an associated time window $[\theta(s_i), \theta(t_i)]$. The time window of a path P is then given as the union of the time windows of its fundamental paths.

Definition 3.10 (Conflicting Paths). Two paths P and Q are *conflicting* iff their time windows intersect. If P and Q are not conflicting, then the two paths are *compatible* and can be served by a single feeder.

After having formally defined the concept of conflicting paths, we see that so far we have not handled conflicts and compatibilities in a very sensible way. In fact, formulation (6) simply overestimates the time windows of the paths to be equal to the time horizon of the entire instance. On the upside, this makes the formulation a lot simpler. However, if we take better care of the combinations of different paths, we can separate the problem of finding paths through the refueling graph from the problem of assigning sets of conflict-free paths to individual feeders.

Remark 3.11. Decompositions have always played a crucial role in order to solve vehicle routing and dial-a-ride problems. As an example, the authors of [11] use a *cluster first – route second* approach to facilitate the solution of large-scale dial-a-ride problems. Additionally, the authors divide the time horizon of the problem into different time slices to obtain smaller and easier subproblems.

While we follow a similar approach here, we would like to point out that the decomposition into fundamental paths is exact (as will be shown in Theorem 3.13). Furthermore, the decomposition is carried out in an implicit fashion, requiring only a modified pricing procedure and some additional inequalities added to the master problem.

We can define the problem of assigning fundamental paths to feeders more formally by introducing the *path conflict graph* \mathcal{I} . This graph contains the paths in \mathcal{P} as vertices, where $P, Q \in \mathcal{P}$ are connected by an edge in \mathcal{I} iff the paths are conflicting. Finding an assignment of a set of paths in \mathcal{P} to a fixed number of feeders is equivalent to solving a graph coloring problem in the conflict graph \mathcal{I} .

Coloring problems are in general quite challenging to solve computationally. The large number of paths constituting the conflict graph further complicates matters. Fortunately, the structure of the conflict graph \mathcal{I} enables us to solve the assignment problem in an implicit fashion, which adds little overhead to the formulation. Specifically, we will give a formulation based on fundamental paths, which incorporates the assignment problem while only adding one additional variable and linearly many additional inequalities.

We let $\mathcal{I}_{\text{fund}}$ be the conflict graph induced by the set of fundamental paths and make the following observation:

Lemma 3.12. *The conflict graph $\mathcal{I}_{\text{fund}}$ of fundamental paths is an interval graph. Every clique in $\mathcal{I}_{\text{fund}}$ intersects in $\theta(u)$ for some $u \in V$.*

Proof. We already established that the time windows of fundamental paths are intervals. Since edges in $\mathcal{I}_{\text{fund}}$ correspond to intersecting intervals, the first part follows.

For the second part, consider a collection P_1, \dots, P_k of fundamental paths forming a clique in \mathcal{I} and note that every pair of paths intersect. Helly’s Theorem (see, e.g., [12]) now implies that all time windows intersect in a single interval $I := [\theta_{\min}, \theta_{\max}]$. Furthermore, the point θ_{\min} is given as the left endpoint of the time window of some P_i , $i = 1, \dots, k$, which in turn corresponds to a vertex $u \in V$. \square \square

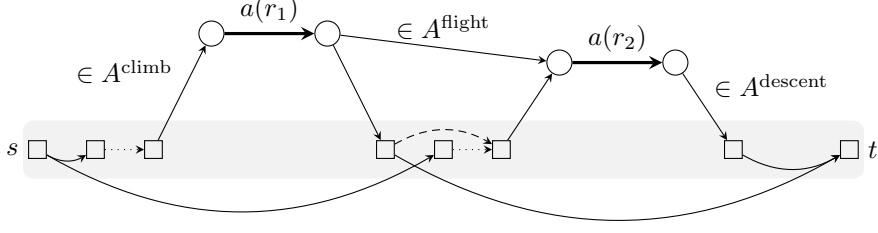


Figure 5: A fundamental path graph D_{fund} . Refueling arcs are drawn thick, base waiting arcs dashed, base refueling arcs dotted, and transit / auxiliary arcs solid.

We go on to modify the refueling graph D in order to obtain a *fundamental path graph* $D_{\text{fund}} = (V_{\text{fund}}, A_{\text{fund}})$ based on the following rules:

- We add (base) refueling, climb, descent, and flight arcs in the same way as in the refueling graph. We denote the corresponding arc sets by $A_{\text{fund}}^{\text{base,refuel}}$, $A_{\text{fund}}^{\text{refuel}}$ and so on.
- We connect each landing vertex b_θ with each takeoff vertex $b_{\theta'}$ if $0 \leq \theta' - \theta < \Delta\theta^{\text{base,refuel}}$, using a waiting arc. We define $A_{\text{fund}}^{\text{wait}}$ to be the set of waiting arcs.
- We add an origin s , a destination t , and auxiliary arcs connecting s to all takeoff vertices, and t to all landing vertices. We let $A_{\text{fund}}^{\text{aux}}$ be the corresponding set of arcs.

In order to make the most of Lemma 3.12, we define for each $u \in V$ the set $A^\theta(u)$ as the set of non-auxiliary arcs whose time-windows contain $\theta(u)$, i.e.,

$$A^\theta(u) := \{a \in A_{\text{fund}} \setminus A_{\text{fund}}^{\text{aux}} \mid \theta(u) \in \Delta\theta(a)\},$$

and use these arc sets in order to define *clique inequalities* of the form

$$\sum_{a \in A^\theta(u)} x_a \leq \beta.$$

Furthermore, we let $\mathcal{P}_{\text{fund}}$ be the set of (s, t) -paths in D_{fund} , where we understand that the removal of the auxiliary variables from some path in $\mathcal{P}_{\text{fund}}$ yields a fundamental path as defined above.

In order to restrict the number of required feeders to some value β while satisfying all requests, we reformulate model (6) in terms of paths through the fundamental path graph D_{fund} :

$$\begin{aligned} \sum_{a \in A^\theta(u)} x_a &\leq \beta & \forall u \in V_{\text{fund}} \\ \sum_{P \in \mathcal{P}_{\text{fund}}: a \in P} x_P &= x_a & \forall a \in A_{\text{fund}}^{\text{transit}} \\ \sum_{P \in \mathcal{P}_{\text{fund}}: a \in P} x_P &= 1 & \forall a \in A_{\text{fund}}^{\text{req}} \\ x_a &\in \{0, 1\} & \forall a \in A_{\text{fund}}^{\text{transit}} \\ x_P &\in \{0, 1\} & \forall P \in \mathcal{P}_{\text{fund}}. \end{aligned} \tag{9}$$

Note that while the refueling graph and the fundamental path graph are related, there is a number of subtle differences (see Figures 2 and 5 for examples of the respective graphs). The key difference lies in the way the waiting periods between requests are represented: In the case of a refueling graph, the waiting periods are modeled using a series of waiting arcs. In contrast to this, the fundamental path graph allows any path to finish directly after any landing vertex by skipping to t . Symmetrically, by skipping from s to any takeoff vertex, a path can be confined to a later time period. While the fundamental path graph contains waiting arcs as well, these arcs are short with respect to $\Delta\theta$ and we can expect a smaller number of them for most instances.

This difference translates to the respective formulations (6) and (9): We can expect a solution of the set covering formulation to consist of few paths, each covering a large number of requests, whereas a solution of the fundamental path formulation comprises a larger number of paths containing fewer arcs.

While limiting the number of paths is achieved explicitly in the set covering formulation, the same effect is achieved using clique inequalities in the fundamental path graph.

To incorporate the different objective functions, we proceed as follows: In case we wish to minimize the number of feeders, we add $\min \beta$ as an objective function (letting β be an optimization variable). In case we want to minimize fuel consumption without any constraint regarding the number of feeders, we do not need to include the variable β and the clique inequalities at all. If on the other hand we want to minimize fuel consumption while restricting the number of feeders, we can fix β to some value while minimizing.

We would like to remark that D_{fund} is a directed acyclic graph as well. We can extend the definition of μ and $\Delta\mu$ to the auxiliary arcs (where no fuel is consumed) and use Algorithm 1 to find paths with negative reduced costs in $\mathcal{P}_{\text{fund}}$. We can also generate branching variables as we did in case of the previous formulation, once again yielding a Branch-and-Price framework. On the other hand, the restrictions regarding the waiting time between requests make it impossible to reformulate (9) as a set covering problem. Specifically, we are no longer able to shortcut request arcs by delaying takeoffs or advancing landings, which might increase base waiting times beyond $\Delta\theta^{\text{base,refuel}}$. These long base waiting arcs are specifically excluded from the constructed fundamental path graph.

Theorem 3.13. *The formulations (5) and (9) are equivalent in the following sense: Any optimal solution \tilde{x} of (5) can be transformed into a solution (x^*, β^*) of (9) with the same objective value and vice versa.*

Proof. Let \tilde{x} be an optimal solution of (5). Let P_1, \dots, P_k be the paths contained in \tilde{x} . We decompose all P_j , $j \in [k]$, into fundamental paths Q_i^j , $i \in [k_j]$ for some k_j , as described above and let \mathcal{Q} be the resulting set of all paths. We add auxiliary arcs (from s and to t) to each $Q_i \in \mathcal{Q}$ to turn it into an (s, t) -path in D_{fund} and set the corresponding entry of x^* to one. It is easy to see that the paths of the resulting solution serve all requests. It is straightforward to incorporate the values of the compound flow variables x_a into x^* as well. Finally, we let $\beta^* := k$ be the number of paths in the solution \tilde{x} . Note that this choice of β^* ensures that all clique inequalities are satisfied: We know from Lemma 3.4 that the paths P_1, \dots, P_k are arc-disjoint on A^{transit} . Since D is acyclic, these k paths must satisfy the clique inequalities and so must the paths in \mathcal{Q} .

Conversely, consider an optimal solution (x^*, β^*) of (9) and let $\mathcal{Q} := \{Q_1, \dots, Q_\ell\}$ be the paths comprising the solution x^* . If $\ell \leq \beta^*$, i.e., if the solution consists of at most β^* paths, then we can set $\tilde{x}(Q_j) = 1$ for $j = 1, \dots, \ell$ to obtain a solution of (5) with an objective value of ℓ . Since (x^*, β^*) is optimal, it must hold that $\ell = \beta^*$.

If on the other hand $\ell > \beta^*$, then we have to combine the fundamental paths into β^* many paths in D . Recall that the conflict graph $\mathcal{I}_{\text{fund}}$ is an interval graph and therefore a perfect graph. The same holds for the subgraph of $\mathcal{I}_{\text{fund}}$ induced by the paths in \mathcal{Q} . The clique inequalities ensure that there is no clique of size larger than β^* in this subgraph. We can therefore find a coloring of the paths in \mathcal{Q} consisting of at most β^* colors. Each color $i = 1, \dots, \beta^*$ corresponds to a set of pairwise compatible paths, which can be combined into a single path P_i . The paths P_i form a solution \tilde{x} of (5) with an objective value of β^* , as required. \square

Remark 3.14. The conflict graph $\mathcal{I}_{\text{fund}}$ is in fact chordal. Therefore, a coloring of $\mathcal{I}_{\text{fund}}$ (and any induced subgraph) can be easily obtained by ordering the intervals according to their endpoints and coloring them greedily.

4. Computational experiments

All experiments were conducted using an implementation in the C++ programming language compiled using the GNU C++ compiler with the optimizing option `-O2`. We used version 6.0.0 of the SCIP [1] optimization suite and version 8.1 of GUROBI [20] as underlying LP solver. All measurements were taken on an Intel Core i7-965 processor clocked at 3.2 GHz. We set a time limit of 3600s.

4.1. Instances

In the following, we will briefly discuss the instances used for the computational experiments (see Table 2 for the complete list). The instances fall into two scenarios, namely the Asia and the Transatlantic Scenario, corresponding to the respective regions of the world.

Table 2: Instances considered during the computation

| | Code | Name | # Requests |
|----------|------|-------------------------------------|------------|
| Asia | UBBB | Heydar Aliyev International Airport | 312 |
| | UTDD | Dushanbe International Airport | 78 |
| | UAAH | Balkhash Airport | 112 |
| | USNN | Nizhnevartovsk Airport | 84 |
| | UNNT | Tolmachevo Airport | 300 |
| | UWPS | Saransk Airport | 304 |
| | UKFF | Simferopol International Airport | 540 |
| | U000 | Alykel Airport | 136 |
| Atlantic | EINN | Shannon Airport | 300 |
| | BIRK | Reykjavik Airport | 120 |
| | BGSF | Kangerlussuaq Airport | 196 |
| | CYQX | Gander International Airport | 1 428 |
| | LPLA | Lajes Field | 158 |
| | CYYR | CFB Goose Bay | 580 |
| | CYYQ | Churchill Airport | 46 |
| | LUKK | Chisinau International Airport | 82 |

For each scenario, a number of eight tanker bases was chosen, subjecting to the constraint that the bases should be located at an existing airport with a runway of a length sufficient for the feeder aircrafts.

In the Transatlantic Scenario, likely refueling positions for most flights are located over the northern Atlantic, south of Greenland. Thus, with Gander International Airport, CFB Goose Bay, Kangerlussuaq Airport, Reykjavik Airport and Shannon Airport, five of the eight feeder bases were chosen located in this region. To serve the southern traffic between South America, the Caribbean and south Europe, Lajes Field on the Azores was used as a further feeder base. Churchill Airport was selected to serve flights to the West Coast from central Europe and Chisinau International Airport to serve flights going to the Middle East.

The feeder base selection in the Asia Scenario proved more difficult, since the refueling positions for the flights in the scenario cover a wide area over west and middle Asia, with some flights having refueling positions even in the eastern parts. To serve a wide variety of flights, the eight bases have been divided into two groups. The first one, consisting of Heydar Aliyev International Airport (Baku), Saransk Airport, and Simferopol International Airport is dedicated to flights between Europe, the Middle East, and India. The airports Dushanbe International Airport, Balkhash Airport, Nizhnevartovsk Airport, Tolmachevo Airport, and Alykel Airport form a line from the north to the south in central Asia in order to cover the wide arc of flights connecting to east and southeast Asia.

The workload of the feeder bases varies in both scenarios. In the Asia Scenario, the number of requests per base ranges from 78 at Dushanbe International Airport (Instance 1) to 540 at Simferopol International Airport. In the Transatlantic Scenario the difference between the requests at the different bases is even higher, ranging from 46 request at Churchill Airport to 1428 request at Gander International Airport. Furthermore, the requests are not equally distributed over the whole day. Most feeder bases have peak traffic times and times with small or no workload at all. Thus, each instance posed different challenges for the optimization.

Regarding the cruiser positions, the origins / destinations were chosen such that the base is (approximately) at the center of the circle defined by the origin / destination of the individual requests. Based on the refueling time $\Delta\theta^{\text{refuel}}$ and the speed v of the feeder, this fixes the requests at distances of about 144 km from the base. Additionally, the cruisers follow their flight corridors, resulting in angles around the base being distributed around those corresponding to an overall flight path.

The differences in the number of requests and their distribution also affect the sizes of the refueling graphs used during computations, ranging from just 230 vertices and 1 300 arcs for the base at Churchill Airport to 7 138 vertices and 994 247 arcs at Gander International Airport. It is worth noting that the graphs are rather dense, which is due to the fact that, in theory, feeders can spend hours of time in holding patterns while waiting for cruisers to arrive.

Table 3: Performance of the set covering formulation (8) and the fundamental path formulation (9) (for the minimization of the number of feeders).

| Instance | Set covering | | Fundamental path | | |
|----------|------------------|-------|------------------|----------|---|
| | Running time (s) | Gap | Running time (s) | Gap | |
| Asia | UBBB | 3 600 | 0.33 | 2.89 | 0 |
| | UTDD | 3 600 | 0.11 | 0.06 | 0 |
| | UAAH | 3 600 | 0.11 | 0.11 | 0 |
| | USNN | 3 600 | 0.2 | 0.23 | 0 |
| | UNNT | 3 600 | 0.08 | 1.70 | 0 |
| | UWPS | 3 600 | 0.06 | 1.00 | 0 |
| | UKFF | 3 600 | 0.2 | 25.87 | 0 |
| | U000 | 0.34 | 0 | 0.18 | 0 |
| Atlantic | EINN | 2.34 | 0 | 1.53 | 0 |
| | BIRK | 3 600 | 0.12 | 0.08 | 0 |
| | BGSF | 3 600 | 0.35 | 1.08 | 0 |
| | CYQX | 3 600 | – | 2 882.28 | 0 |
| | LPLA | 0.22 | 0 | 0.15 | 0 |
| | CYYR | 3 600 | 12 | 24.08 | 0 |
| | CYYQ | 0.18 | 0 | 0.08 | 0 |
| | LUKK | 0.08 | 0 | 0.11 | 0 |

Artificial instances

In addition to the 14 instances from Table 2, we generated several artificial instances to study both the computational performance of our formulations and the effect of instance sizes and properties. We chose sizes (in terms of $|\mathcal{R}|$) of 100, 200, 500, and 1 000. The instances are based on requests of around 10 000 kg each during a 48 hour time window. The flight corridors are either *narrow* or *wide*, the flight paths either *unidirectional* or *bidirectional*. For each variant, we used ten different random seeds, yielding a total of 160 instances. More details are given in Appendix B. The set of instances (both original and artificial) may be accessed based on the DOI [10.6084/m9.figshare.8305988.v1](https://doi.org/10.6084/m9.figshare.8305988.v1).

4.2. Results

We were able to find at least some solution for all instances with respect to both objective functions within the prescribed time limit of one hour. The resulting costs, as well as a comparison with the state-of-the-art solution [34], are depicted in Table 5. A comparison of the computation times between the set covering formulation (8) and the fundamental path formulation (9) is shown in Table 3. In case an instance was not solved to optimality, the table shows the remaining gap, defined as $(p - d)/d$, where p is the objective function value of the best known feasible solution and d is the best known lower bound.

Running times

In terms of running times (see Table 3), the size of the instance plays a significant role. Indeed, the most challenging instance is also the largest: Gander International Airport (CYQX). Instances with fewer than 100 requests are solved rather quickly. The relationship between size and computation time is more clearly visible for the fundamental path formulation, while there are some outliers with respect to the set covering formulation. Specifically, the instance at Shannon Airport (EINN) is solved remarkably quickly given its size.

Regarding the different formulations, the fundamental path formulation is clearly superior to the set covering formulation: The latter fails to solve the larger instances within the time limit, and, in case of the feeder base at CYQX, even fails to solve the root LP in order to obtain a nontrivial lower bound. In contrast, the fundamental path formulation solves all instances to optimality in under 50 min.

Regarding the artificial data, the results are even more pronounced (see Table 4 for a summary): While for the fundamental path formulation, even the largest instances can be solved in under 5 min,

Table 4: Performance of the set covering formulation (8) and the fundamental path formulation (9) over the artificial instances. Running times are averaged arithmetically, including unsolved instances (counting with the time limit of 3 600 s). An instance is defined by its size, whether it is uni- (U) or bidirectional (B), and whether its flight corridor is narrow (N) or wide (W).

| Instance | | | Set covering | | Fundamental path |
|----------|---|---|------------------|------------------|------------------|
| | | | Solved instances | Running time (s) | Running time (s) |
| 100 | U | N | 10 | 0.13 | 0.04 |
| | | W | 10 | 0.15 | 0.04 |
| | B | N | 10 | 0.19 | 0.06 |
| | | W | 9 | 360.15 | 0.06 |
| 200 | U | N | 9 | 720.73 | 0.27 |
| | | W | 7 | 1 080.49 | 0.31 |
| | B | N | 5 | 1 800.61 | 0.41 |
| | | W | 4 | 2 160.48 | 0.49 |
| 500 | U | N | 2 | 2 882.08 | 5.77 |
| | | W | 2 | 2 882.66 | 6.89 |
| | B | N | 2 | 2 881.55 | 7.16 |
| | | W | 2 | 2 882.82 | 6.33 |
| 1000 | U | N | 0 | 3 600 | 98.88 |
| | | W | 0 | 3 600 | 193.19 |
| | B | N | 0 | 3 600 | 223.23 |
| | | W | 0 | 3 600 | 271.27 |

the set covering formulation performs increasingly poorly as the instance size increases, and none of the largest instances can be solved within the prescribed time limit.

Solution quality

As mentioned before, we are interested in minimizing the feeder fleet size as well as the combined fuel consumption. We collected the values of both objectives for both the exact solution and the original distribution system across the different scenarios in Table 5. Firstly, it is worth noting that, while the number of feeders does not vary too much between the exact and the state of the art solution, the fuel consumption of the feeder fleet is drastically reduced in the exact solution across all instances.

Specifically, the feeder fuel burn could be reduced by 55 % in the Asia Scenario and 58 % in the Transatlantic Scenario. When considering the two scenarios as a whole, we should of course take the cruisers' fuel consumption into account as well. In the Asia Scenario, the cruisers burn 50 543 902 kg of fuel, whereas in the Transatlantic Scenario that value is 56 655 982 kg. The improvement in feeder fuel consumption translates into combined savings of 2.12 % and 3.23 % of the total fuel consumption for the respective scenarios.

The state-of-the-art heuristic assigned feeders in an online fashion while favoring the assignment of airborne feeders over the feeders located at the feeder base. The exact solutions for the fleet size were able to save one or two feeders at smaller feeder bases, but due to the relatively simple scenarios the original distribution worked quite well. Conversely, larger instances could profit from the new solutions: The number of necessary feeders at **CYQX** could be reduced from 59 to 49 and the numbers at **CYYR** could be reduced from 27 to 22. Apparently the simple distribution mechanic is unable to scale up to more complex scenarios.

Regarding fuel consumption, we see several reasons for the larger improvements: Due to the fact that the original distribution mechanic assigned feeders using an online approach, the feeders were scheduled to take off with full tanks and return to the base once unable to satisfy further demands. In contrast, planning entire tours ahead of time makes it possible to determine the amount of fuel required in order to satisfy all requests exactly. Specifically, during low traffic feeders only need an amount of fuel sufficient

Table 5: Exact vs. state-of-the-art heuristic solution values across all instances. State-of-the-art solutions are due to [34, Fig. 11, “sim Feeder”]

| Instance | Exact solution | | State-of-the-art | | |
|----------|----------------|----------------|------------------|----------------|-----------|
| | # Feeders | Fuel burn (kg) | # Feeders | Fuel burn (kg) | |
| Asia | UBBB | 18 | 149 878 | 20 | 314 696 |
| | UTDD | 9 | 47 190 | 11 | 92 276 |
| | UAAH | 9 | 60 343 | 9 | 122 061 |
| | USNN | 5 | 38 352 | 6 | 85 155 |
| | UNNT | 12 | 164 496 | 15 | 353 037 |
| | UWPS | 17 | 132 237 | 18 | 314 244 |
| | UKFF | 20 | 248 145 | 22 | 583 643 |
| | UOOO | 11 | 87 915 | 11 | 180 365 |
| Atlantic | EINN | 18 | 156 632 | 18 | 364 027 |
| | BIRK | 8 | 64 466 | 9 | 144 334 |
| | BGSF | 10 | 113 210 | 11 | 222 811 |
| | CYQX | 49 | 636 261 | 59 | 1 537 251 |
| | LPLA | 9 | 78 202 | 9 | 173 388 |
| | CYR | 22 | 272 846 | 27 | 643 749 |
| | CYYQ | 5 | 39 164 | 6 | 160 562 |
| | LUKK | 7 | 36 900 | 9 | 90 335 |

for the refueling of one or two cruisers, resulting in lower fuel consumption of the feeder itself. During high traffic times, the feeders take off with almost full tanks, reducing the benefits of planning entire tours ahead of time. However, the original heuristic restricted feeders to three consecutive refueling maneuvers before returning back to the base. Conversely, the exact solutions do not depict this behavior of the feeders. The removal of this restriction both saves fuel and reduces the number of feeders in the high traffic instances.

Of course, apart from these shortcomings of the simple distribution mechanic, it is (unsurprisingly) the case that the mechanic, as a heuristic algorithm, fails to deliver exact solutions to the underlying \mathcal{NP} -hard problem.

Lastly, regarding the trade-off between number of tankers and fuel consumption, Figure 6 shows the Pareto front with respect to both objectives in the case of the feeder base at Shannon Airport (EINN). We see that the addition of a single feeder substantially decreases fuel consumption, whereas further increases in fleet size yield marginal improvements in fuel consumption.

5. Conclusion and Future Work

In this paper, we introduced the air-to-air refueling problem as a variant of a vehicle routing problem. We established a simplified, sufficiently accurate physical model of the fuel consumption during refueling operations, which we then used to derive a Branch-and-Price framework incorporating an adapted labeling algorithm to solve the pricing subproblem. We used the concept of fundamental path and conflict intervals in order to derive a different formulation.

We compared both formulations with respect to their running times across several instances and found that the reformulation performs significantly better than the original one, bringing the computation time down to less than half an hour even for the largest instances.

From the airtraffic perspective, the exact solutions more than halved the fuel consumption of the feeders across the different instances, resulting in significant savings in terms of operating costs. Similarly, we were able to obtain solutions with a significantly reduced number of feeders.

The practical tractability of the air-to-air refueling problem suggests that it is possible to incorporate the refueling problem into the scheduling of routes for the cruisers along the different bases. Furthermore, it may be worth investigating how the solutions can be made robust against uncertainties regarding the refueling times and locations. Finally, any improvement of the design of both the cruiser and the feeder

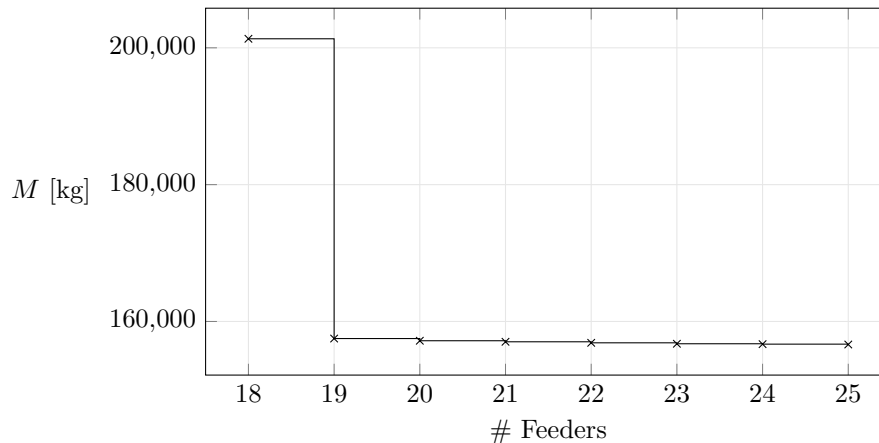


Figure 6: The Pareto front of the solutions with respect to the number of feeders and the fuel consumption in the case of the base at EINN.

aircrafts with respect to fuel consumption may yield additional benefits to the overall operating costs.

Acknowledgments

The authors would like to thank Ralf Borndörfer for several helpful comments.

The second author would like to acknowledge the funding by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany’s Excellence Strategy – EXC 2163/1- Sustainable and Energy Efficient Aviation – Project-ID 390881007.

References

- [1] Tobias Achterberg. “SCIP: solving constraint integer programs”. In: *Mathematical Programming Computation* 1.1 (2009), pp. 1–41. DOI: 10.1007/s12532-008-0001-1.
- [2] John David Anderson. *Aircraft performance and design*. McGraw-Hill, 1999. ISBN: 978-0-07116-010-0.
- [3] Yash P Aneja, Vijay Aggarwal, and Kunhiraman PK Nair. “Shortest chain subject to side constraints”. In: *Networks* 13.2 (1983), pp. 295–302. DOI: 10.1002/net.3230130212.
- [4] Hannah Bast et al. “Route planning in transportation networks”. In: *Algorithm engineering*. Springer, 2016, pp. 19–80. DOI: 10.1007/978-3-319-49487-6_2.
- [5] John E Beasley and Nicos Christofides. “An algorithm for the resource constrained shortest path problem”. In: *Networks* 19.4 (1989), pp. 379–394. DOI: 10.1002/net.3230190402.
- [6] Ralf Borndörfer. “Aspects of set packing, partitioning, and covering”. PhD thesis. 1998. ISBN: 978-3-82654-351-7.
- [7] Jean-François Cordeau. “A branch-and-cut algorithm for the dial-a-ride problem”. In: *Operations Research* 54.3 (2006), pp. 573–586. DOI: 10.1287/opre.1060.0283.
- [8] Daniel Delling, Thomas Pajor, and Dorothea Wagner. “Accelerating Multi-modal Route Planning by Access-Nodes”. In: *Algorithms - ESA 2009*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 587–598. ISBN: 978-3-642-04128-0.
- [9] Daniel Delling and Dorothea Wagner. “Time-Dependent Route Planning”. In: *Robust and Online Large-Scale Optimization: Models and Techniques for Transportation Systems*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 207–230. ISBN: 978-3-642-05465-5. DOI: 10.1007/978-3-642-05465-5_8.

- [10] Jacques Desrosiers et al. “Time constrained routing and scheduling”. In: *Handbooks in operations research and management science* 8 (1995), pp. 35–139. DOI: 10.1016/S0927-0507(05)80106-9.
- [11] Yvan Dumas, Jacques Desrosiers, and François Soumis. “Large scale multi-vehicle dial-a-ride problems”. In: *Groupe d’études et de recherche en analyse des décisions* (1989). ISSN: 0711–2440.
- [12] Jürgen Eckhoff. “Helly, Radon, and Carathéodory type theorems”. In: *Handbook of convex geometry*. Elsevier, 1993, pp. 389–448. ISBN: 978-0-44489-596-7.
- [13] Sevgi Erdoğan and Elise Miller-Hooks. “A green vehicle routing problem”. In: *Transportation Research Part E: Logistics and Transportation Review* 48.1 (2012), pp. 100–114. DOI: 10.1137/1.9781611973594.fm.
- [14] European Commission. *Flightpath 2050. Europe’s Vision for Aviation*. Accessed April 16, 2019. 2011. URL: <https://ec.europa.eu/transport/sites/transport/files/modes/air/doc/flightpath2050.pdf>.
- [15] Farzaneh Ferdowsi, Hamid Reza Maleki, and Sanaz Rivaz. “Air refueling tanker allocation based on a multi-objective zero-one integer programming model”. In: *Operational Research* (19, 2018). DOI: 10.1007/s12351-018-0402-5.
- [16] P.P.M van der Linden G. La Rocca M. Li and R.J.M. Elmendorp. “Conceptual design of a passenger aircraft for aerial refueling operations”. In: *29th Congress of the International Council of the Aeronautical Sciences (ICAS 2014)*. International Council of Aeronautical Sciences - ICAS. 2014, pp. 162–172. ISBN: 978-1-63439-411-6.
- [17] Gerald Gamrath and Marco Lübbecke. “Experiments with a generic Dantzig-Wolfe decomposition for integer programs”. In: *International Symposium on Experimental Algorithms*. Springer. 2010, pp. 239–252. DOI: 10.1007/978-3-642-13193-6_21.
- [18] Michael R Garey and David S Johnson. *Computers and Intractability*. W.H. Freeman and Company, San Francisco, 1979.
- [19] Bruce L Golden, Subramanian Raghavan, and Edward A Wasil. *The vehicle routing problem: latest advances and new challenges*. Vol. 43. Springer Science & Business Media, 2008. DOI: 10.1007/978-0-387-77778-8.
- [20] LLC Gurobi Optimization. *Gurobi Optimizer Reference Manual*. 2018. URL: <http://www.gurobi.com>.
- [21] Ernst Hairer, Syvert Nørsett, and Gerhard Wanner. *Solving Ordinary Differential Equations I: Nonstiff Problems*. 1993. DOI: 10.1007/978-3-540-78862-1.
- [22] Gabriel Y Handler and Israel Zang. “A dual algorithm for the constrained shortest path problem”. In: *Networks* 10.4 (1980), pp. 293–309. DOI: 10.1002/net.3230100403.
- [23] Gerhard Hiermann et al. “The Electric Fleet Size and Mix Vehicle Routing Problem with Time Windows and Recharging Stations”. In: *European Journal of Operational Research* 252.3 (2016), pp. 995–1018. DOI: 10.1016/j.ejor.2016.01.038.
- [24] ICAO. *Long-Term Traffic Forecasts. Passenger and Cargo*. Accessed April 16, 2019. Apr. 2018. URL: <https://www.icao.int/sustainability/Pages/eap-fp-forecast-scheduled-passenger-traffic.aspx>.
- [25] David S. Johnson. “Approximation algorithms for combinatorial problems”. In: *Journal of Computer and System Sciences* 9.3 (1974), pp. 256–278. ISSN: 0022-0000. DOI: [https://doi.org/10.1016/S0022-0000\(74\)80044-9](https://doi.org/10.1016/S0022-0000(74)80044-9). URL: <http://www.sciencedirect.com/science/article/pii/S0022000074800449>.
- [26] Michael Jünger et al. *50 Years of integer programming 1958-2008: From the early years to the state-of-the-art*. Springer Science & Business Media, 2009. ISBN: 978-3-540-68274-5. DOI: 10.1007/978-3-540-68279-0.

- [27] Sezgin Kaplan and Ghaith Rabadi. “Exact and heuristic algorithms for the aerial refueling parallel machine scheduling problem with due date-to-deadline window and ready times”. In: *Computers & Industrial Engineering* 62.1 (2012), pp. 276–285. DOI: 10.1016/j.cie.2011.09.015.
- [28] Niklas Kohl et al. “2-path cuts for the vehicle routing problem with time windows”. In: *Transportation Science* 33.1 (1999), pp. 101–116. DOI: 10.1287/trsc.33.1.101.
- [29] Granino Arthur Korn and Theresa M Korn. *Mathematical handbook for scientists and engineers: definitions, theorems, and formulas for reference and review*. 2003. ISBN: 978-0-48641-147-7.
- [30] Gilbert Laporte. “The vehicle routing problem: An overview of exact and approximate algorithms”. In: *European journal of operational research* 59.3 (1992), pp. 345–358. DOI: 10.1016/0377-2217(92)90192-C.
- [31] Gilbert Laporte et al. “Classical and modern heuristics for the vehicle routing problem”. In: *International Transactions in Operational Research* 7.4–5 (2000), pp. 285–300. DOI: 10.1111/j.1475-3995.2000.tb00200.x.
- [32] Mo Li and G La Rocca. “Conceptual design of joint-wing tanker for civil operations”. In: *RAeS Applied Aerodynamics Conference, Bristol, UK*. 2014, pp. 22–24.
- [33] Marco Lübbecke and Jacques Desrosiers. “Selected topics in column generation”. In: *Operations research* 53.6 (2005), pp. 1007–1023. DOI: 10.1287/opre.1050.0234.
- [34] Fabian Morscheck and Mo Li. “Benefits and challenges of a civil air to air refuelling network analysed in a traffic simulation”. In: *Digital Avionics Systems Conference (DASC), 2015 IEEE/AIAA 34th*. IEEE. 2015, 1B5–1. DOI: 10.1109/DASC.2015.7311337.
- [35] Pedro Munari, Twan Dollevoet, and Remy Spliet. “A generalized formulation for vehicle routing problems”. In: *arXiv preprint arXiv:1606.01935* (2016).
- [36] Rajendar Nangia. “Efficiency parameters for modern commercial aircraft”. In: *The Aeronautical Journal* 110.1110 (2006), pp. 495–510. DOI: 10.1017/S0001924000001391.
- [37] Bjørn Petersen and Mads Kehlet Jepsen. “Partial path column generation for the vehicle routing problem with time windows”. In: *Proceedings of the 4th International Network Optimization Conference (INOC 2009)* (2009).
- [38] Luigi Di Puglia Pugliese and Francesca Guerriero. “Dynamic programming approaches to solve the shortest path problem with forbidden paths”. In: *Optimization Methods and Software* 28.2 (2013), pp. 221–255. DOI: 10.1080/10556788.2011.630077.
- [39] Stefan Ropke, Jean-François Cordeau, and Gilbert Laporte. “Models and branch-and-cut algorithms for pickup and delivery problems with time windows”. In: *Networks: An International Journal* 49.4 (2007), pp. 258–272. DOI: 10.1002/net.20177.
- [40] Martin Schmied et al. “Postfossile Energieversorgungsoptionen für einen treibhausgasneutralen Verkehr im Jahr 2050: eine verkehrsträgerübergreifende Bewertung”. In: *UMWELT-BUNDESAMT. TEXTE* 30 (2015).
- [41] Michael Schneider, Andreas Stenger, and Dominik Goeke. “The electric vehicle-routing problem with time windows and recharging stations”. In: *Transportation Science* 48.4 (2014), pp. 500–520. DOI: 10.1287/trsc.2013.0490.
- [42] Olivia Smith and Martin Savelsbergh. “A note on shortest path problems with forbidden paths”. In: *Networks* 63.3 (2014), pp. 239–242. DOI: 10.1002/net.21541.
- [43] Peter R. Thomas et al. “Advances in air to air refuelling”. In: *Progress in Aerospace Sciences* 71 (2014), pp. 14–35. DOI: 10.1016/j.paerosci.2014.07.001.
- [44] Alexander Tsukerman et al. “Optimal Rendezvous Guidance Laws with Application to Civil Autonomous Aerial Refueling”. In: *Journal of Guidance, Control, and Dynamics* 41.5 (2018), pp. 1167–1174. DOI: 10.2514/1.G003154.
- [45] Laurence Wolsey. *Integer programming*. Wiley, 1998. ISBN: 978-0-471-28366-9.

A. Formal Definitions

A.1. Definition of the Initial Fuel Functions

In the following, we give the formal definitions of the function $\mu : \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$ describing the initial fuel mass, depending on the final fuel mass M^{final} and the duration $\Delta\theta$ of the corresponding phase.

Free flight

Recall that in the free flight phase the fuel mass is according to (2). Thus, the initial fuel is given by

$$\mu^{\text{flight}}(M^{\text{final}}, \Delta\theta) := (M^{\text{final}} + M^{\text{ac}}) \cdot \exp\left(\frac{v \cdot \Delta\theta}{X}\right) - M^{\text{ac}} + M^{\text{final}}.$$

Advance

Recall that the advance phase requires an initial climb with a minimum duration of $\Delta\theta^{\text{climb}}$, which is conducted with a reduced efficiency X^{climb} . We first consider the case where $\Delta\theta = \Delta\theta^{\text{climb}}$, in which the initial fuel is given as

$$\mu^{\text{climb}}(M^{\text{final}}, \Delta\theta) := (M^{\text{final}} + M^{\text{ac}}) \cdot \exp\left(\frac{v \cdot \Delta\theta}{X^{\text{climb}}}\right) - M^{\text{ac}} + M^{\text{final}}.$$

Recall that the climb of a feeder is always conducted in order to advance to the origin of some request. The case where $\Delta\theta = \Delta\theta^{\text{climb}}$ corresponds to a situation where the initial request is within a range of d^{climb} of the base. However, if the origin of the request is further from the origin, the duration of the advance is $\Delta\theta > \Delta\theta^{\text{climb}}$. In this case, the advance includes another free flight phase of a duration of $\Delta\theta - \Delta\theta^{\text{climb}}$. The initial fuel of the advance is therefore given as

$$\mu^{\text{advance}}(M^{\text{final}}, \Delta\theta) := \begin{cases} \mu^{\text{climb}}(M^{\text{final}}, \Delta\theta^{\text{climb}}) & \text{if } \Delta\theta = \Delta\theta^{\text{climb}}, \\ \mu^{\text{flight}}(\mu^{\text{climb}}(M^{\text{final}}, \Delta\theta^{\text{climb}}), \Delta\theta - \Delta\theta^{\text{climb}}) & \text{otherwise,} \end{cases}$$

where we understand that $\Delta\theta \geq \Delta\theta^{\text{climb}}$ must hold.

Descent

Recall that the final part of the descent is conducted while gliding, in which case fuel is consumed at a fixed rate of ρ^{gliding} . The gliding distance d^{gliding} determines the maximum gliding duration $\Delta\theta^{\text{gliding}} := d^{\text{gliding}}/v$, and the corresponding fuel consumption is given by

$$\mu^{\text{descent}}(M^{\text{final}}, \Delta\theta) := \begin{cases} \mu^{\text{flight}}(M^{\text{final}} + \rho^{\text{gliding}} \Delta\theta^{\text{gliding}}, \Delta\theta - \Delta\theta^{\text{gliding}}) & \text{if } \Delta\theta > \Delta\theta^{\text{gliding}}, \\ M^{\text{final}} + \rho^{\text{gliding}} \Delta\theta & \text{otherwise.} \end{cases}$$

Refueling

Recall that during contact, the fuel mass is given by (3). Therefore, the initial fuel is equal to

$$\mu^{\text{contact}}(M^{\text{final}}) := (M^{\text{final}} + M^{\text{equiv}}(r)) \cdot \exp\left(\frac{v \cdot \Delta\theta^{\text{contact}}}{X}\right) - M^{\text{equiv}}(r) + M^{\text{final}}.$$

Since the approach / retreat is conducted in free flight, the initial fuel of a refueling operation is easily calculated as

$$\mu^{\text{refuel}}(M^{\text{final}}) := \mu^{\text{flight}}(\mu^{\text{contact}}(\mu^{\text{flight}}(M^{\text{final}}, \Delta\theta^{\text{retreat}})), \Delta\theta^{\text{approach}}).$$

Base waiting / refueling

If a feeder is waiting at the base, then there is no change in fuel, i.e.,

$$\mu^{\text{wait}}(M^{\text{final}}, \Delta\theta) := M^{\text{final}}.$$

If a feeder is being refueled at the base, we assume w.l.o.g. that it must have arrived without any fuel to spare:

$$\mu^{\text{base,refuel}}(M^{\text{final}}, \Delta\theta) := 0.$$

A.2. Definition of the Fuel Burn Functions

As mentioned above, the fuel burn function $\Delta\mu : \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$ is defined in a fashion similar to the initial fuel function μ . Specifically, we have that

$$\begin{aligned}\Delta\mu^{\text{flight}}(M^{\text{final}}, \Delta\theta) &:= \mu^{\text{flight}}(M^{\text{final}}, \Delta\theta) - M^{\text{final}}, \\ \Delta\mu^{\text{climb}}(M^{\text{final}}, \Delta\theta) &:= \mu^{\text{climb}}(M^{\text{final}}, \Delta\theta) - M^{\text{final}}, \\ \Delta\mu^{\text{advance}}(M^{\text{final}}, \Delta\theta) &:= \mu^{\text{advance}}(M^{\text{final}}, \Delta\theta) - M^{\text{final}}, \text{ and} \\ \Delta\mu^{\text{descent}}(M^{\text{final}}, \Delta\theta) &:= \mu^{\text{descent}}(M^{\text{final}}, \Delta\theta) - M^{\text{final}}.\end{aligned}$$

Furthermore, during base refueling / waiting, no fuel is burned, i.e.,

$$\Delta\mu^{\text{wait}} \equiv \Delta\mu^{\text{base,refuel}} \equiv 0.$$

Finally, when the feeder refuels a cruiser (serving a request $r \in \mathcal{R}$), we have

$$\Delta\mu^{\text{refuel}}(M^{\text{final}}) := \mu^{\text{refuel}}(M^{\text{final}}) - M^{\text{req}}(r) - M^{\text{final}}.$$

B. Artificial Instances

To obtain instances similar to the given 14, we generated the request set around some origin according to the following rules:

- The requested fuel masses were sampled according to the (normal) distribution

$$\mathcal{N}(10\,000 \text{ kg}, 3\,000 \text{ kg}).$$

- Regarding the time θ of the requests, we used a time horizon of 48 h peaking every 12 h, i.e., given by the following distribution:

$$\frac{1}{5} \sum_{i=0}^5 \mathcal{N}(12i \text{ h}, 3 \text{ h}).$$

- We distributed the origins of the requests around the base using a radius chosen according to the distribution

$$\mathcal{N}(\Delta\theta^{\text{refuel}}v/2, \Delta\theta^{\text{refuel}}v/20).$$

For some angle $\alpha \in [0, 2\pi)$ and a given distance d , we chose the endpoints to be at the opposite ends of the circle defined by d , the origin being at an angle of α .

- Regarding the choice of α , we sampled angles according to a *narrow* distribution $\mathcal{N}(0, \pi/8)$ and a *wide* distribution of $\mathcal{N}(0, \pi/4)$ in the unidirectional case. In the bidirectional case, we used the distributions $1/2 (\mathcal{N}(0, \pi/8) + \mathcal{N}(\pi, \pi/8))$ and $1/2 (\mathcal{N}(0, \pi/4) + \mathcal{N}(\pi, \pi/4))$, respectively.