European Journal of Operational Research 304 (2023) 779–796

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Innovative Applications of O.R.

Feeder routing for air-to-air refueling operations☆

Christoph Hansknecht[a,*], Imke Joormann[b,d], Bernd Korn[c,d], Fabian Morscheck[c], Sebastian Stiller[a,d]

[a] Institute for Mathematical Optimization, TU Braunschweig, Universitätsplatz 2, Braunschweig 38106, Germany
[b] Institute of Automotive Management and Industrial Production, TU Braunschweig, Mühlenpfordtstr. 23, Braunschweig 38106, Germany
[c] Institute of Flight Guidance, DLR, Lilienthalplatz 7, Braunschweig 38108, Germany
[d] Cluster of Excellence SE²A–Sustainable and Energy-Efficient Aviation, TU Braunschweig, Germany

ARTICLE INFO

Article history:
Received 2 June 2021
Accepted 13 April 2022
Available online 26 April 2022

Keywords:
Routing
Branch-and-price
Air-to-air refueling
NP-hardness

ABSTRACT

With the ever increasing volume of air traffic comes an enormous environmental impact, specifically due to carbon emissions. A promising approach to reduce this impact is the introduction of en route air-to-air refueling, allowing for the design of aircraft with reduced weight and a decreased fuel burn. Aside from the design of aircraft, the key challenge lies in the planning of the refueling operation in order to minimize the fuel burn of the feeder fleet. The *air-to-air refueling problem* is a variant of a vehicle routing problem in which a fleet of feeders are to perform air-to-air refueling operations for a fixed set of cruisers. In this paper, we devise an IP-based method capable of solving this problem supported by an ODE model of the feeders' fuel consumption. The algorithmic key to solve the air-to-air refueling problem lies in separating the problem of finding routes serving cruisers and assigning sets of routes to individual feeders. We demonstrate the effectiveness of our methods both on real-world and artificial instances, solving all problems to optimality and significantly outperforming state-of-the-art heuristic methods.

1. Introduction

According to the International Civil Aviation Organization, it is expected that the world scheduled passenger traffic will quadruple by 2045 (ICAO, 2018). To reduce the environmental impact, especially the carbon footprint, the European Commission's "Flightpath 2050" defines a target of 75 % CO_2 reduction per passenger-kilometer through technological development by 2050 (European Comission, 2011). Moreover, the price for kerosene is forecasted to more than double by 2050 from the current price of approximately $0.05[US\$]kW^{-1}\ h^{-1}$ (Schmied, Wüthrich, Zah, Friedl et al., 2015). As a result, considerable effort has already been made in order to increase the fuel efficiency of aircraft operations.

Especially long-range aircraft operations require a large amount of fuel. Indeed, the authors of (Bennington & Visser, 2005) observed that the weight of the fuel required for a typical Boeing 747 mission amounts to almost the weight of the aircraft itself. Consequently, the authors concluded that a single air-to-air refueling operation conducted mid-flight by a state-of-the-art military tanker could yield a payload improvement of up to 111 %.

While air-to-air refueling operations are well established in a military context, civilian operations pose several additional challenges, several of which were examined in Mo (2017). For example, many air forces use tankers trailing hoses terminating in a fuel nozzle. The refueling maneuver consists of an approach from behind by the receiving aircraft, which has a probe with a fuel nozzle. Such a maneuver is suited to the agility of fighter jets rather than larger, commercially operated aircraft. The author argues instead for a so-called *boom*, a rigid telescopic system less susceptible to turbulence (see Timmermans & La Rocca, 2014) and advocates for a refueling maneuver where the tanker, also referred to as the *feeder*, approaches the receiving aircraft, called the *cruiser*, from behind and below. This configuration, which is inverse to its military counterpart, offers several advantages. First, the maneuvering is almost exclusively carried out by the feeder, therefore causing minimal discomfort to the passengers. Second, the feeder is kept in the downwash of the cruiser rather than the opposite. This way the cruiser has no need for additional thrust in the maneuver and can be fully optimized for cruise flight.

The cruiser-feeder approach to air transport was further studied as part of the RECREATE (REsearch on a CRuiser Enabled Air Transport Environment) project (Recreate, 2015). In particular, the

☆ 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.

* Corresponding author.
E-mail address: c.hansknecht@tu-braunschweig.de (C. Hansknecht).

https://doi.org/10.1016/j.ejor.2022.04.016
0377-2217/

project included a flight simulation of the refueling maneuver based on state-of-the-art simulators conducted with the help of professional airline pilots in order to establish the airworthiness of the cruiser-feeder operation even during adversarial conditions such as turbulence or engine failures. In addition, optimized designs for both cruiser (La Rocca, Li, van der Linden, & Elmendorp, 2014) and feeder (Li & La Rocca, 2014) aircraft were proposed in order to minimize their respective fuel consumption while taking the mission requirements into account. While the cruiser was designed for transporting 250 passengers over a range of 5000NM (nautical miles), the feeder aircraft was designed to be significantly smaller than existing military tankers in order to reduce the feeder's own fuel consumption during the refueling missions. The feeder is capable of transporting about forty metric tons of fuel mass as payload to be offloaded to about three cruisers during each refueling mission. Refueling missions were designed to be performed within a 250 NM radius around a *refueling base* within a typical duration of one to three hours.

The RECREATE project identified that long-range flights contribute most to the overall fuel consumption and identified locations for refueling bases along the frequently used routes, concluding that commercial air-to-air refueling operations can be implemented in a way that is both economically and ecologically beneficial. A simulation of the overall system predicted a 4.5–6% reduction in operating costs (Morscheck & Li, 2015).

While the project assured the feasibility and airworthiness of the air-to-air refueling concept, the effort in terms of the optimization of the overall system has largely been put into aircraft designs. In contrast, our goal here is to show that the overall efficiency can be improved even more by focusing on the scheduling part of the operation, especially that of the feeder fleet. 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.

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 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 (Aneja, Aggarwal, & Nair, 1983) to solve the emerging pricing problem and discuss some details pertaining to the Branch-and-Price framework.

In Section 4, we evaluate our formulations computationally 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. We finish in Section 5 with our conclusion and an outlook regarding potential future work.

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 $M^{\text{req}}(r)$. 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 $d(p, p')$ and compute d by using the Haversine formula (Korn & Korn, 2013), which is metric. Furthermore, we assume for technical reasons that the coordinates of the base, the request origins, and the request destinations are pairwise different.

1.2. Related work

Air-to-air refueling has received a lot of attention over the last years. We refer to Thomas, Bhandari, Bullock, Richardson, & du Bois (2014) 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 (Ferdowsi, Maleki, & Rivaz, 2018), 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 Kaplan & Rabadi (2012) allow the refueling to take place within 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 Tsukerman, Weiss, Shima, Löbl, & Holzapfel (2018), 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 Bast et al., 2016), including aspects such as time dependence (Delling & Wagner, 2009) and multi-modality (Delling, Pajor, & Wagner, 2009). Whereas these problems are often 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 (Beasley & Christofides, 1989) or time-windows (Desrosiers, Dumas, Solomon, & Soumis, 1995).

While it is possible to derive combinatorial algorithms for $\mathcal{NP}$-hard problems, oftentimes it is advisable to use (mixed) integer programming (Wolsey, 1998) techniques instead. The field of integer programming has been studied extensively (for a survey, see Jünger et al., 2009) for the last sixty years, yielding theoretical results as well as well-tested, ready-to-use software, which can be adapted to specific use cases. Apart from topics such as cutting planes, branching rules, and symmetry handling, an important technique is that of column generation (see Lübbecke & Desrosiers, 2005). 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 Gamrath & Lübbecke, 2010), 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 Golden, Raghavan, & Wasil, 2008; Laporte, 1992 for summaries), which has lead 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 Laporte, Gendreau, Potvin, & Semet, 2000 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 the specific variant, the formulations are either based on arc variables (Cordeau, 2006) or path variables (Dumas, Desrosiers, & Soumis, 1989).

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 Desrosiers et al., 1995; Kohl, Desrosiers, Madsen, Solomon, & Soumis, 1999). 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. We discuss the relation between the air-to-air refueling problem and the VRP more closely in Section 3.2 following the introduction of the underlying physical model.

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 Morscheck & Li (2015). 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 aircraft used in the scenario have been designed for this purpose (see La Rocca et al., 2014). Furthermore, a reference aircraft has been devised to calculate the fuel consumption of a direct flight without air-to-air refueling.

Today's feeder aircraft are multi-role aircraft and are therefore constructed for refueling and as transport aircraft. 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 (La Rocca et al., 2014) 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 (Morscheck & Li, 2015). Between the airports and the refueling point, all aircraft 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: First of all, we assume that all aircraft maintain a constant speed v at all times. While the cruisers maintain a fixed altitude and direction, feeders begin their routes by climbing to the required altitude and finish with a descent before landing. They fly on great circle routes, turn instantly between flight legs, and are indistinguishable with respect to their physical properties and flight characteristics.

Each feeder has an *Operational Empty Weight* M^{ac} and a *Maximum Take-off Weight* $M^{\text{takeoff}} > M^{\text{ac}}$. The difference $M^{\max} := M^{\text{takeoff}} - M^{\text{ac}}$ is available to store fuel, which is either delivered to cruisers or burned during the flight of the feeder itself. Consequently, the fuel mass $M(\theta)$ and therefore the mass of the entire feeder depends on the time θ. In the following we describe the individual flight phases with respect to their fuel consumption.

Free flight. During the free flight phase, the feeder maintains stable speed and altitude. Consequently, the behavior of the mechanical system representing the feeder can be analyzed using basic principles of mechanics (see for example Alrasheed, 2019, Ch. 3): The *drag* D must be equal to the *thrust* T, while the *lift* L must be equal to the total mass $M^{\text{ac}} + M(\theta)$. T is achieved by a jet engine consuming fuel at a rate of $\dot{M}(\theta) = sfc \cdot T$, where the constant sfc denotes the *specific fuel consumption* of the engine. We further assume that each feeder has a constant *lift over drag ratio*L/D. Both sfc and L/D have been optimized for the speed and altitude of these refueling missions during the feeder design.

By combining these facts we obtain the following ordinary differential equation (ODE) describing the fuel consumption of the feeder:

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

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 (Anderson, 1999, Chapter 5.13, "Range").

Fortunately, this ODE can be solved explicitly (for an overview on ODEs, see Hairer, Nørsett, & Wanner, 1993). Specifically, for a

given fuel mass M^0 at an initial time θ_0, the solution is given by

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

where $X := v \cdot (L/D)/sfc$ denotes the *aircraft efficiency* (see Nangia, 2006).

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

$$(L/D)' := \left[(L/D)^{-1}\cos(\gamma) + \sin(\gamma)\right]^{-1}.$$

This leads to a decreased efficiency X^{climb} in the otherwise unmodified Eq. (2). The constant angle γ translates into a fixed distance d^{climb} and a corresponding duration $\Delta\theta^{climb} := d^{climb}/v$ in order to reach the required altitude. As a result, feeders can begin serving requests only after the minimum climb duration of $\Delta\theta^{climb}$ 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 hand 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 $\rho^{gliding}$ 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 of a cruiser $r \in \mathcal{R}$ itself is the most complex of the different phases. It is conducted in three steps, each of which has a constant duration. First, the feeder approaches the cruiser and connects its refueling boom to the cruisers fuel receptacle. We let $\Delta\theta^{approach}$ be the corresponding *approach duration*. As soon as the contact is established, fuel is pumped from feeder to cruiser at a constant rate of $M^{req}(r)/\theta^{contact}$ over the *wet contact duration* $\Delta\theta^{contact}$. After the fuel transfer is completed, the boom is disconnected and the feeder retreats from the cruiser, requiring a *retreat duration* $\Delta\theta^{retreat}$. Since the entire refueling maneuver is executed at constant speed v, the different durations translate into equivalent distances. The approach towards and retreat from the cruiser correspond to free flights (where the difference in fuel mass is given by Eq. (2)) while the change in fuel mass during the wet contact time is given by

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

The solution of this ODE for a fixed M^0 is given by

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

where $M^{equiv}(r)$ is an *equivalent* aircraft mass of

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

In order to calculate $M(\theta)$ during the entire maneuver based on a given initial value M^0, we begin by computing the fuel $M^{contact}$ at the beginning of the contact using (2), which we then substitute into (2) 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. The entire refueling operation has a duration of $\Delta\theta^{refuel}$ defined as

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

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 $\Delta\theta^{base,refuel}$ independent of the refueling amount.

Table 1
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
	UOOO	Alykel Airport	136
Atlantic	EINN	Shannon Airport	300
	BIRK	Reykjavík Airport	120
	BGSF	Kangerlussuaq Airport	196
	CYQX	Gander International Airport	1428
	LPLA	Lajes Field	158
	CYYR	CFB Goose Bay	580
	CYYQ	Churchill Airport	46
	LUKK	Chisinau International Airport	82

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 $\mu : \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$, 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 $\Delta\mu : \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ 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. The inclined reader can find the formal definitions of both functions in Appendix B. All relevant parameter values used by us are listed in Table A.4 in Appendix A.

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.

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

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

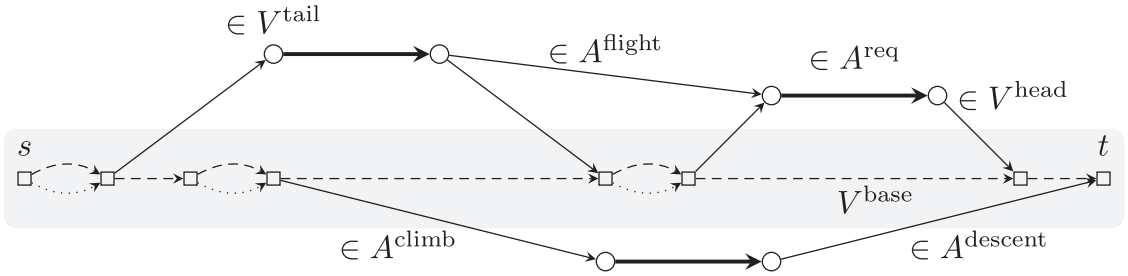


Fig. 1. A refueling graph used to model a refueling problem with three requests. Refueling arcs A^{req} are drawn thick, base waiting arcs A^{wait} dashed, base refueling arcs dotted, and transit arcs A^{transit} solid. Vertices belonging to requests are represented by circles, those corresponding to the base (drawn shaded) are displayed by squares.

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

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

If this inequality is not tight, we assume that the feeder spends the excess time in a holding pattern awaiting the arrival of the cruiser corresponding to request $\hat{r}$ while maintaining the speed of v. 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 Fig. 1.

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. Let $P := (a_1, \dots, a_{k-1})$ be a path with arcs $a_i := (u_i, u_{i+1})$, $i = 1, \dots, k-1$, and vertices $V(P) := \{u_i \mid i = 1, \dots, k\}$. The initial fuel mass $\mu_P : V(P) \to \mathbb{R}_{\geq 0}$ 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. Similarly, the fuel burn $\Delta\mu_P$ of a path P can be defined recursively based on the values $\mu_P(u)$.

Complexity. Even if the number of feasible path in the graph is restricted, the refueling problem is still hard to solve, an observation that will dictate our branching choices later on.

Theorem 3.1. *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. The proof can be found in Appendix D. □

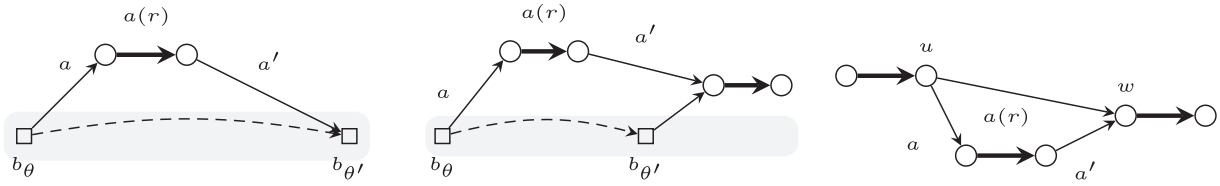
Corollary 3.2. *Minimizing a function $c : \mathcal{P} \to \mathbb{R}_{\geq 0}$ over an air-to-air refueling instance is $\mathcal{NP}$-hard in the strong sense even if all transit arcs are fixed.*

3.1. A set-covering formulation

We formulate the air-to-air refueling problem as a covering of request arcs with feasible paths, leading to the following formulation:

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

Note that this formulation covers the set of request arcs rather than to partition them. Relaxing a partitioning problem to a set covering problem is generally preferable, if possible. 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 Borndörfer, 1998 for an in-depth explanation). However, we need to ensure that the covering formulation yields correct solutions for the air-to-air refueling problem. To this end, we show that a request arc never needs to be in more than one path. We use the notion of shortcut paths (see also Fig. 2):



(a) Shortcut by replacing with waiting arc.

(b) Shortcut by delaying take-off from θ to θ'.

(c) Shortcut via alternative arc (u, w).

Fig. 2. Different ways to shortcut a sequence $a, a(r), a'$ of arcs covering a request $r \in \mathcal{R}$ in a refueling graph.

Definition 3.3. Let P be a path serving a request $r \in \mathcal{R}$ and let a, $a(r) = (r^{\text{tail}}, r^{\text{head}})$, a' be the sequence of arcs surrounding the request r, where $a = (u, r^{\text{tail}})$ and $a' = (r^{\text{head}}, v)$. The *shortcut path* P' with respect to the request r is constructed as follows: If both u and w are base vertices, say b_θ and $b_{\theta'}$ with $\theta < \theta'$, then there must be a subpath of waiting arcs connecting b_θ and $b_{\theta'}$. The sequence a, $a(r)$, a' in P is replaced by that subpath. If either u or w are base vertices, P' is constructed by delaying the takeoff or advancing the landing and inserting some waiting arcs at appropriate positions. If both u and w are request vertices, the sequence a, $a(r)$, a' is replaced by the arc (u, w), which is guaranteed to exist since w is reachable from u.

Lemma 3.4. *Let P be a path, and 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.*

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. □

Since the number of paths does not increase when we replace P by P', we can also use shortcuts when minimizing the number of feeders. In any case, we can solve the covering formulation (SC) and shortcut paths during a post processing phase if the optimal solution covers any request arc more than once.

3.2. Relationship to vehicle routing problems

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) and $|\mathcal{R}|$ many customers, yielding a graph akin to that of a VRP. The times $\theta(r)$ at which the requests must be served are trivial time-windows, suggesting that the air-to-air refueling problem can be solved using VRP formulations such as the commonly used arc-based two- or three-index formulations (see Dumas et al., 1989; Ropke, Cordeau, & Laporte, 2007).

Unfortunately, this approach is not easily applicable to the air-to-air refueling problem owing to the nature of the fuel consumption. Specifically, while the nonlinearity of the function μ is hidden from the combinatorial model, the dependency of μ on the final fuel mass needs to be handled correctly, preventing us from simply reusing a VRP formulation. In order to obtain a purely arc-based formulation we would have to expand the graph based on fuel levels.

We opt against this approach due to the vast increase in problem size: The refueling graphs for the given instances (see Section 4.1) are already very dense, consisting of up to about one million arcs. The different fuel levels range from 0 to $M^{\max} \approx 40,000$ kilogram. Even considering a moderate resolution of steps of 100 kilogram, 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).

On the other hand, the fact that the time windows for the refueling requests consist of single points in time facilitates the solution based on a time-expansion: Since the serving time is fixed, the time-expansion of the corresponding vertices is trivial, leaving the base vertex as the only vertex which is copied, and thereby significantly reducing the overhead of the time-expansion.

While using a time-expansion can be applied to arbitrary vehicle routing or dial-a-ride problems, we do not believe that this approach would yield any benefits compared to state-of-the-art formulations, since the approach specifically relies on the fact that request vertices do not need to be time-expanded.

3.3. Branch-and-price

A problem in solving formulation (SC) 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 Lübbecke & Desrosiers, 2005) to generate new path variables as needed.

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 is 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. Additionally, the two branches are unbalanced, since branching a path P to one forces all paths P' sharing a transit arc with P to zero, whereas branching a path P to zero has little effect on the overall problem.

Compound flows. To alleviate this problem, we include binary *compound flow* variables y_a for every $a \in A^{\text{transit}}$ together with linking constraints into formulation (SC). We then instruct the IP solver to primarily branch on compound flow variables by assigning appropriate branching priorities. We adopt this approach in order to let the underlying solver choose branching candidates according to the built-in, highly sophisticated branching rules, which generally outperform custom-built ones.

Unfortunately, the size of the refueling graphs in terms of transit arcs significantly slows down the solution process. We therefore opt to generate compound flow variables during a second col-

Algorithm 1: Pricing loop, executed at every Branch-and-Bound node.

loop
- $(x^*, y^*, \lambda^*, \delta^*) \leftarrow$ Solve node LP
- $P_{\text{opt}} \leftarrow$ Compute feasible path minimizing $\bar{c}$ ▷ `Using Algorithm 2`
- **if** $\bar{c}(P_{\text{opt}}) < 0$ **then**
  - Add P_{opt} to node LP and **continue**
- **foreach** $r \in \mathcal{R}$, $a(r) := (u, w)$ **do**
  - $A_{\text{fract}} \leftarrow \{a \in \delta^-(u) \mid y^*_a \notin \{0, 1\}\}$
  - $A_{\text{comp}} \leftarrow \{a \in \delta^-(u) \mid$ node LP contains compound variable $y_a\}$
  - **if** $\exists\, a \in A_{\text{fract}}$ *and* $A_{\text{comp}} \cap A_{\text{fract}} = \emptyset$ **then**
    - Add variable y_a, linking constraint to node LP and **continue**
- **break**

umn generation step. The entire pricing procedure is laid out in Algorithm 1 , details regarding the pricing of paths are discussed in the next paragraph.

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.

Ultimately however, owing to the hardness of the air-to-air refueling problem with fixed arcs (Theorem 3.1), branching on path variables may still be necessary, as we cannot expect to obtain 0/1 solutions even after branching out all compound flow variables. We therefore have to take forbidden and required paths into account in the pricing problem.

The pricing problem. The pricing problem consists of finding 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 (SC) 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

$$\bar{c}_P := c_P - \sum_{a \in P \cap A^{\text{req}}} \lambda_a - \sum_{a \in P \cap A^{\text{transit}}} \delta_a. \tag{3}$$

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 Handler & Zang, 1980), there are a number of algorithms, both IP-based and combinatorial, which solve many real-world problems to optimality within reasonable time (see Beasley & Christofides, 1989 for a summary on the subject). We implement a variant of the labeling scheme introduced in Aneja et al. (1983); see Algorithm 2 . 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 (Line 16), where $\ell = (M, c)$ is defined to dominate $\ell' = (M', c')$ iff $\ell \neq \ell'$, $M \leq M'$, and $c \leq c'$.

Algorithm 2: Labeling scheme to find constrained shortest paths.

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

Output: Feasible (s, t)-path P_{opt} with minimum cost with respect to c

1 $L_u \leftarrow \emptyset$ for all $u \in V \setminus \{t\}$, $L_t \leftarrow \{(0, 0)\}$
2 $(c_{\text{opt}}, P_{\text{opt}}) \leftarrow (\infty, \emptyset)$
3 **for** $j \leftarrow n$ **downto** 1 **do**
4 **foreach** *label* $\ell_j \leftarrow (M^j, c_j) \in L_{u_j}$ **do** ▷ `$M^j$, $c_j$: fuel, cost of label $\ell_j$`
5 $P_j \leftarrow$ the (u_j, t)-path corresponding to ℓ_j
6 **foreach** $a \leftarrow (u_i, u_j) \in \delta^-(u_j)$ **do**
7 $M^i \leftarrow \mu(a, M^j)$
8 $c_i \leftarrow c_j + c(a, M^j)$
9 **if** $c_i + \underline{c}(s, u_i) \geq c_{\text{opt}}$ **then continue**
10 **if** $M^i + \underline{\mu}(s, u_i) > M^{\max}$ **then continue**
11 **else**
12 $P \leftarrow$ composition of the path achieving $\underline{c}(s, u_i)$, arc a, and P_j
13 **if** P *is feasible and* $c(P) < c_{\text{opt}}$ **then**
14 $(c_{\text{opt}}, P_{\text{opt}}) \leftarrow (c(P), P)$
15 **if** (M^i, c_i) *is not dominated in* L_{u_i} **then**
16 Insert (M^i, c_i) into L_{u_i}, remove newly dominated labels from L_{u_i}
17 **foreach** *label* $\ell_s \leftarrow (M, c) \in L_s$ **do**
18 $P \leftarrow (s, t)$-path corresponding to ℓ_s
19 **if** $c(P) < c_{\text{opt}}$ **then** $(c_{\text{opt}}, P_{\text{opt}}) \leftarrow (c(P), P)$
20 **return** P_{opt}

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. We therefore perform the labeling from the target vertex t towards the source s according to a topological ordering similarly to the labeling algorithm proposed in Hernandez, Feillet, Giroudeau, & Naud (2016). We employ the following techniques: Firstly, we use the fact that the refueling graph D is acyclic. Rather than having to maintain a priority queue of labels, it is sufficient to expand all labels for each vertex exactly once (Line 3). Furthermore, due to the monotonicity of both considered objective functions, we utilize $\underline{c}$ 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 use $\underline{\mu} := \mu(\cdot, 0)$ as a lower bound to identify labels which are not contained in any feasible (s, t)-path (Lines 9, 10). Finally, we employ heuristically found paths to update an upper bound on the cost of the optimal path (Line 14).

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 2, since no path containing a must be added to the current subproblem.

Still, according to Theorem 3.1, 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 is 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 is fixed to zero, in which case we have to find a path minimizing (3) not contained in $\mathcal{P}^0$, corresponding to a routing problem with *forbidden paths*. This problem has been examined previously (Pugliese & Guerriero, 2013), leading to a dynamic programming algorithm, which solves the shortest path problem with forbidden paths (SPPFP) in polynomial time (Smith & Savelsbergh, 2014).

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 2 to accept the 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 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.

Pricing performance. We found that despite 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. We also attempted to improve the computational performance by using Algorithm 2 as a heuristic by returning the first path with negative reduced costs found during the execution of the algorithm. While the first pricing steps are sped up based on this approach, it is ultimately necessary to solve the pricing problem optimally in order to prove the optimality of a given fractional solution of the master problem. The heuristic generation of paths seem to have an adverse effect on the solution process in terms of the overall running time, likely due to the fact that the corresponding variables become redundant in subsequent iterations and do not appear in the optimal solution.

A simple primal heuristic. We employ the greedy heuristics due to (Johnson, 1974) in order to obtain an initial solution when minimizing the required number of feeders. To this end, we repeatedly find paths maximizing the number of yet uncovered requests using Algorithm 2 until a feasible solution is found.

3.4. Fundamental path graph formulation

In the following, we discuss some of the drawbacks of the previously defined formulation (SC), and derive an improved formulation. The techniques employed revolve around decomposing the paths into their constituent pieces in such a way that the problems of finding new paths through the graph and assigning these paths to feeders are more clearly separated.

As a motivating example, consider the case of minimizing the number of feeders serving two requests r and r' that can be served by a single feeder. If the column generation has already generated a path serving only r and a path serving only r', the combined single path must still be generated in a subsequent pricing step. The entire problem therefore largely depends on the column generation to exhaustively explore the set $\mathcal{P}$. This dependency becomes more pronounced if paths consist of a large number of subpaths.

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 use the following definition:

Definition 3.5 (Fundamental Path)**.** A path P is called *fundamental* if it begins with a base refueling arc and ends with a descent arc, while all other arcs of P are in $A^{\text{transit}} \cup A^{\text{req}} \cup A^{\text{wait}}$. Furthermore, it must hold for each $a \in (P \cap A^{\text{wait}})$ that $\Delta\theta(a) < \Delta\theta^{\text{base,refuel}}$.

Remark 3.6. A subject related to fundamental paths is the concept of *partial paths* (see Munari, Dollevoet, & Spliet, 2016; Petersen & Jepsen, 2009), 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 specific 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.

As mentioned above, in Hernandez et al. (2016), the authors generate variables according to *trips* of a VRP rather than *routes*, where a route consists of multiple trips carried out in sequence by a single vehicle. While the approaches share some similarity, there are some subtle differences as well. Notably, the VRP studied in Hernandez et al. (2016) relies on a time horizon based on uniform steps, whereas the relevant time points $\theta(V)$ in our case are generally unevenly distributed, requiring a more careful analysis.

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 resulting subpaths by $P_1, \dots, P_k$. It is easy to see that the paths P_i, $i = 1, \dots, k$, are fundamental paths.

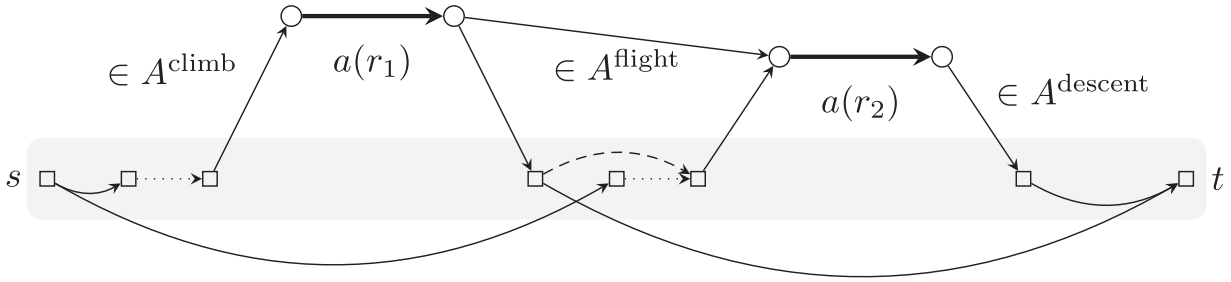


Fig. 3. A fundamental path graph D_{fund}. Refueling arcs are drawn thick, base waiting arcs dashed, base refueling arcs dotted, and transit / auxiliary arcs solid.

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 3.7 (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.

With these definitions, we see that so far we have not handled conflicts and compatibilities in a very sensible way. In fact, formulation (SC) 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.

Decompositions have always played a crucial role in order to solve vehicle routing and dial-a-ride problems. As an example, the authors of Dumas et al. (1989) 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 3.9). 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.

The fundamental path graph. 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 $A^{\text{base,refuel}}_{\text{fund}}$, $A^{\text{refuel}}_{\text{fund}}$ 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 $A^{\text{wait}}_{\text{fund}}$ to be the set of waiting arcs. Finally, 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 $A^{\text{aux}}_{\text{fund}}$ be the corresponding set of arcs.

Note that while the refueling graph and the fundamental path graph are related, there is a number of subtle differences (see Figs. 1 and 3 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.

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 arcs from some path in $\mathcal{P}_{\text{fund}}$ yields a fundamental path as defined above and extend the definition of μ and $\Delta\mu$ to the auxiliary arcs of D_{fund} (where no fuel is consumed), enabling us to compute the fuel consumption and fuel burn of paths in $\mathcal{P}_{\text{fund}}$.

We reformulate (SC) in terms of fundamental path variables, including compound variables and linking constraints as introduced above. We first consider the minimization of the fuel consumption $\Delta\mu$ and later turn towards the minimization of the number of feeders. The key difference between the objectives lies in the fact that the fuel consumption is additive with respect to the decomposition into fundamental paths, i.e., the fuel consumption of a path is equal to the sum of the fuel consumption along its fundamental subpaths. The reformulation in terms of fundamental paths is therefore simply given by

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

We use Algorithm 2 to find fundamental paths with negative reduced costs in D_{fund} (which is also acyclic), and generate branching variables using Algorithm 1.

Note that the restrictions regarding the waiting time between requests make it impossible to formulate (FP) as a set covering problem. Specifically, we can no longer shortcut request arcs by delaying takeoffs or advancing landings, since shortcutting paths may increase base waiting times beyond $\Delta\theta^{\text{base,refuel}}$, essentially splitting one fundamental path into two. On the other hand, we can expect solutions of (FP) to consist of many relatively short paths, which can be freely recombined.

Clique inequalities. We now turn towards the objective of minimizing the number of feeders required to serve all requests. We cannot simply minimize $c_P \equiv 1$, since compatible fundamental paths can be assigned to the same feeder. To this end, we introduce the *fundamental path conflict graph* $\mathcal{I}_{\text{fund}}$. The graph contains the paths in $\mathcal{P}_{\text{fund}}$ as vertices, where $P, Q \in \mathcal{P}_{\text{fund}}$ are connected by an edge iff the paths are conflicting. Finding an assignment of a set of fundamental paths in $\mathcal{P}_{\text{fund}}$ to a fixed number of feeders is equivalent to solving a graph coloring problem in the conflict graph $\mathcal{I}_{\text{fund}}$. 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:

Lemma 3.8. *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., Eckhoff, 1993) now implies that all time windows intersect in a single interval $I := [\theta_{\min}, \theta_{\max})$. Furthermore, the point $\theta_{\min}$ 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 $u \in V$. □

In order to make the most of Lemma 3.8, 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.,

$$A^\theta(u) := \{a \in A_{\text{fund}} \setminus A^{\text{aux}}_{\text{fund}} \mid \theta(u) \in \Delta\theta(a)\}, \tag{4}$$

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

$$\sum_{a \in A^\theta(u)} y_a \leq \beta. \tag{5}$$

In order to restrict the number of required feeders to some value β while satisfying all requests, we add the clique inequalities to (SC). Minimizing the number of feeders corresponds to minimizing the value of β. The following Theorem establishes the equivalence between the formulations:

Theorem 3.9. *The formulations (SC) and (FP) are equivalent in the following sense: Any optimal solution $(\tilde{x}, \tilde{y})$ of (SC) consisting of at most k paths can be transformed into a solution $(x^*, y^*, \beta^* = k)$ of (FP) satisfying all clique inequalities and vice versa.*

Proof. Let $(\tilde{x}, \tilde{y})$ be an optimal solution of (SC). We assume w.l.o.g. that $\tilde{x}$ is supported by exactly k paths $P_1, \dots, P_k$. 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. We add auxiliary arcs (from s and to t) to each Q_i^j to turn it into an (s,t)-path in D_{fund} and let $\mathcal{Q}$ be the resulting set of fundamental paths. We set $x^*_Q := 1$ for each path $Q \in \mathcal{Q}$ and $x^*_P := 0$ for each path $P \in \mathcal{P}_{\text{fund}} \setminus \mathcal{Q}$. It is easy to see that the paths in x^* serve all requests, and that we can set $y^* := \tilde{y}$ in order to satisfy the linking constraints of (FP). We note that setting $\beta^* := k$ satisfies all clique inequalities (5): For each vertex $u \in V$, the arcs in $A^\theta(u)$ are present in D, being a subset of a directed (s,t)-cut. Since D is acyclic, the cut can intersect with at most k of the arcs in $P_1, \dots, P_k$. The same holds true with respect to D_{fund} and $\mathcal{Q}$.

Conversely, consider an optimal solution (x^*, y^*) of (FP) and let $\mathcal{Q} := \{Q_1, \dots, Q_\ell\}$ be the fundamental 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 (SC) with an objective value of ℓ. Since (x^*, β^*) is optimal, it must hold that $\ell = k$.

If on the other hand $\ell > k$, then we have to combine the fundamental paths into $\beta^* := k$ 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 (SC) with an objective value of β^* as required. □

Finally, note that the conflict graph $\mathcal{I}_{\text{fund}}$ is 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. What is more, the transformation laid out in Theorem 3.9 preserves the cost of paths with respect to the fuel burn $\Delta\mu$, enabling us to minimize the total fuel consumption while restricting the number of feeders.

4. Computational experiments

All experiments were conducted using an implementation in the C++ programming language (available at https://github.com/chrhansk/refueling) compiled using the GNU C++ compiler with full optimization. We used version 6.0.0 of the SCIP (Achterberg, 2009) optimization suite and version 8.1 of Gurobi (Gurobi Optimization, 2018) as underlying LP solver. All measurements were taken on an Intel Core i7-965 processor clocked at 3.2 gigahertz. We set a time limit of 3600 seconds.

4.1. Instances

In the following, we will briefly discuss the instances used for the computational experiments (see Table 1 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. Recall that we touched on the real-world data underpinning the instances in Section 1.3. In particular, each cruiser in the scenario corresponds to an aircraft in the real-world traffic scenario mentioned above.

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 aircraft.

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, Reykjavík 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 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 covering 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 poses 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 144km from the base. Additionally, the cruisers follow their flight corridors, resulting in

Table 2
Performance of formulations (SC) and (FP) for the minimization of the number of feeders.

Instance		Set covering		Fundamental path	
		Running time (seconds)	Gap	Running time (seconds)	Gap
Asia	UBBB	3600	0.33	2.89	0
	UTDD	3600	0.11	0.06	0
	UAAH	3600	0.11	0.11	0
	USNN	3600	0.2	0.23	0
	UNNT	3600	0.08	1.70	0
	UWPS	3600	0.06	1.00	0
	UKFF	3600	0.2	25.87	0
	UOOO	0.34	0	0.18	0
Atlantic	EINN	2.34	0	1.53	0
	BIRK	3600	0.12	0.08	0
	BGSF	3600	0.35	1.08	0
	CYQX	3600	–	2882.28	0
	LPLA	0.22	0	0.15	0
	CYYR	3600	12	24.08	0
	CYYQ	0.18	0	0.08	0
	LUKK	0.08	0	0.11	0

Table 3
Optimal vs. state-of-the-art heuristic solution values across all instances. State-of-the-art solutions are due to Morscheck & Li (2015, Fig. 11, "sim Feeder").

Instance		Exact solution		State-of-the-art	
		# Feeders	Fuel burn (kilogram)	# Feeders	Fuel burn (kilogram)
Asia	UBBB	18	149,878	20	314,696
	UTDD	9	47,190	11	92,276
	UAAH	9	60342.7	9	122,061
	USNN	5	38351.6	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	87915.2	11	180,365
Atlantic	EINN	18	156,632	18	364,027
	BIRK	8	64465.9	9	144,334
	BGSF	10	113,210	11	222,811
	CYQX	49	636,261	59	1,537,251
	LPLA	9	78201.5	9	173,388
	CYYR	22	272,846	27	643,749
	CYYQ	5	39163.8	6	160,562
	LUKK	7	36899.7	9	90,335

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 1300 arcs for the base at Churchill Airport to 7138 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.

Artificial instances. In addition to the 16 instances from Table 1, we generate 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, 1000, and 2000. The instances are based on requests of around 10,000 kilogram each during a 48 hours time window. The flight corridors are either *narrow* or *wide*, the flight paths either *unidirectional* or *bidirectional*. For each variant, we use ten different random seeds, yielding a total of 200 instances. More details are given in Appendix Appendix C. See Hansknecht (2019) to access the original data.

4.2. Results

We are 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 (Morscheck & Li, 2015), are depicted in Table 3. A comparison of the computation times between the set covering formulation (SC) and the fundamental path formulation (FP) is shown in Table 2. 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. As seen in Table 2, the running times of both algorithms are mainly dependent on the number of requests of the individual instances. The largest instance, Gander International Airport (CYQX) is also the most challenging to solve.

We see that formulation (SC) 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, formulation (FP) solves all instances to optimality in under fifty minutes with the vast majority of instances solved within one minute. Conversely, minimizing the fuel burn is much easier, with formulation (FP) solving all instances in under one minute. We suspect that minimizing the number of feeders introduces additional symmetries into the problem, impairing the solution process.

A closer look at the behavior of the solver when minimizing the number of feeders for the hardest of the non-artificial instances CYQX, reveals that both formulations give a solution with 54 feeders in the root node, but the generated variables are insuf-

ficient to achieve a better solution. In particular, the variables required for the optimal solution with 49 feeders were not generated in the root LP, necessitating the exploration of other nodes in the Branch-and-Bound tree. However, while formulation (FP) solves the initial root LP in approximately one minute, formulation (SC) does not even finish the root node within the prescribed time limit. Consequently, formulation (FP) has enough time to explore the Branch-and-Bound tree sufficiently to find the optimal solution.

With respect to the artificial instances the results (Table E.5 and Fig. E.5) show that unidirectional instances are easier to solve than the bidirectional ones, likely due to being at least partly decomposable with respect to both directions. Similarly, a wider corridor seems to make the instances more difficult to solve, with a particularly large difference between the instances with 1000 requests in the unidirectional setting. We found that the narrow instances generally require fewer variables and fewer feeders in their respective optimal solutions than their wide counterparts. Presumably, the requests are easier to consolidate and assign to the feeders, simplifying the optimization problem.

The difference between the formulations is remarkable across all artificial instances: While formulation (FP) solves even larger instances consisting of 1000 requests in under 10 minutes, formulation (SC) fails to solve any instance within the prescribed time limit. Formulation (FP) only breaks down when attempting to solve the hardest artificial instances, consisting of 2000 requests.

Across all instances, real and artificial, the running time of formulation (FP) is closely related to the number of Branch-and-Bound nodes examined during the solution process, which in turn mostly depends on the number of its requests. The instances in the Asia Scenario are mostly solved in the root node, except for the largest instance UKFF, which is solved after 19 nodes. The situation is similar for the Transatlantic Scenario, where the largest instance CYQX requires the inspection of 561 nodes to solve. Accordingly, the solution process for the largest artificial instances exhausts the time limit after the exploration of well above 500 Branch-and-Bound nodes. Consequently, many of the largest artificial instances can be solved to optimality within an increased time limit of 12 hours, requiring the exploration of up to 2200 nodes of the Branch-and-Bound tree.

The root LP is generally solved in less than 10 minutes with the remaining time spent exploring other nodes of the tree. We also compared the gaps for the largest artificial instances, which were solved within a time limit of 12 hours but not within 3600 seconds. The gaps are mostly due to suboptimal primal solutions, implying that improved primal heuristics could help in closing the gap further.

Solution quality. The values of both objectives for both the optimal solution and the original distribution system across the different scenarios is shown in Table 3.

It is worth noting that, while the number of feeders does not vary too much between the optimal and the state-of-the-art solution, the fuel consumption of the feeder fleet is drastically reduced in the optimal 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 kilogram of fuel, whereas in the Transatlantic Scenario that value is 56,655,982 kilogram. 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 optimal 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 can be reduced from 59 to 49 and the numbers at CYYR can be reduced from 27 to 22. This failure of the mechanic to scale up to more complex scenarios is unsurprising due to the $\mathcal{NP}$-hardness of the underlying problem.

Regarding fuel consumption, the larger improvements are in part due to the online approach of the original mechanic, which scheduled feeders to take off with full tanks and forced them to return once they are unable to serve further requests. Especially during low traffic the feeders loaded substantially more fuel than necessary, resulting in an increased fuel burn. Furthermore, the mechanic restricted each feeder to at most three consecutive refueling maneuvers before a return to base. Lifting this restriction both saves fuel and reduces the number of feeders in high traffic scenarios.

Lastly, the two objectives can be traded off against each other: Based on the optimal solution with respect to the burnt fuel we have an upper bound on the maximum number of required feeders in order to achieve the lowest possible fuel burn. By optimizing fuel burn while varying the maximum number of feeders we can compute the Pareto front of the two objectives, which is shown in Fig. E.6 for instance EINN, showing that the fuel burn of 201,322 kilogram of the solution with the minimum number of 18 feeders can be reduced by almost a quarter to 157,502 kilogram by employing just one additional feeder.

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 air traffic perspective, the optimal 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.

Regarding any future work, according to Recreate (2015, pp. 3) the implementation of a cruiser-feeder operation is realizable "in the medium-term", i.e., within about 20 years. Consequently, many practical questions are still open. In particular, the commercials aspects beyond fuel savings have so far received little attention, necessitating "the conception of realistic business scenarios".

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

Appendix A. Model parameters

The following numerical values were used for the parameters of the physical model in Section 2:

Table A.4
Model parameters.

Description		Symbol	Value	
Feeder	Maximum Take-off Weight	M^{takeoff}	62,933	kilogram
	Operational Empty Weight	M^{ac}	14,881	kilogram
	Maximum fuel mass	$M^{\max}$	42,456	kilogram
	Efficiency	X	18,393	nautical mile
	Speed	v	240.3	meter per second
Climb	Required climbing distance	d^{climb}	87.2	kilometer
	Climbing efficiency	X^{climb}	6621.5	nautical mile
Descent	Gliding distance	d^{gliding}	156.8	kilometer
	Gliding fuel rate	ρ^{gliding}	160	kilogram per hour
Durations	Approach duration	$\Delta\theta^{\text{approach}}$	12	minute
	Contact duration	$\Delta\theta^{\text{contact}}$	5	minute
	Retreat duration	$\Delta\theta^{\text{retreat}}$	3	minute
	Base refueling duration	$\Delta\theta^{\text{base,refuel}}$	30	minute

Appendix B. Formal definitions

B.1. Definition of the initial fuel functions

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

Free flight. During the free flight phase the fuel mass is according to (2). Thus, the initial fuel is given by

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

Advance. The advance phase requires an initial climb with a duration of $\Delta\theta^{\text{climb}}$, which is conducted with a reduced efficiency X^{climb}. The initial fuel during the climb is obtained according to (2), yielding

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

The entire advance with a duration of $\Delta\theta \geq \Delta\theta^{\text{climb}}$ begins with the initial climb. If the initial request is at a distance of more than d^{climb} from the base, it continues with a free flight phase with a duration of $\Delta\theta - \Delta\theta^{\text{climb}}$. The initial fuel of the advance is therefore given as

$$\mu^{\text{advance}}(M^{\text{final}}, \Delta\theta) := \mu^{\text{flight}}(\mu^{\text{climb}}(M^{\text{final}}), \Delta\theta - \Delta\theta^{\text{climb}})$$

Descent 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

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

Refueling. During contact, the fuel mass is given by (2). Therefore, the initial fuel is equal to

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

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:

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

B.2. Definition of the fuel burn functions

As mentioned above, the fuel burn function $\Delta\mu : \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ 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

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

Appendix C. Artificial instances

To obtain instances similar to the given 16, we generated the request set around some origin according to the following rules:

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

  $$\mathcal{N}(10000, 3000).$$

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

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

- 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.

Appendix D. Proof of Theorem 3.1

Proof. We reduce from the $\mathcal{NP}$-complete Numerical 3-dimensional matching (N3DM) problem introduced in Garey & Johnson (1979). An instance of N3DM is given by disjoint sets X, Y, and Z, each of cardinality q, a weight function $s : X \cup Y \cup Z \to \mathbb{Q}_{>0}$, 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 kilometer) 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 $d^{\text{climb}} = 87.2\,\text{km} < \varrho$). 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 Fig. D.4.

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 B). 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 A.4, it is easily calculated that $\beta \approx 873$ kilogram $< M^{\max}$. Note that while these constants may no 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 $B = M^{\max} - \beta$ and define the requested fuel mass for $r \in \mathcal{R}$ as

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

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 $(M^{\max} - \beta)/B$, followed by an application of (D.1). Since the respective factors are constant, the sizes of $M^{\text{req}}(\cdot)$ remain polynomially bounded. □

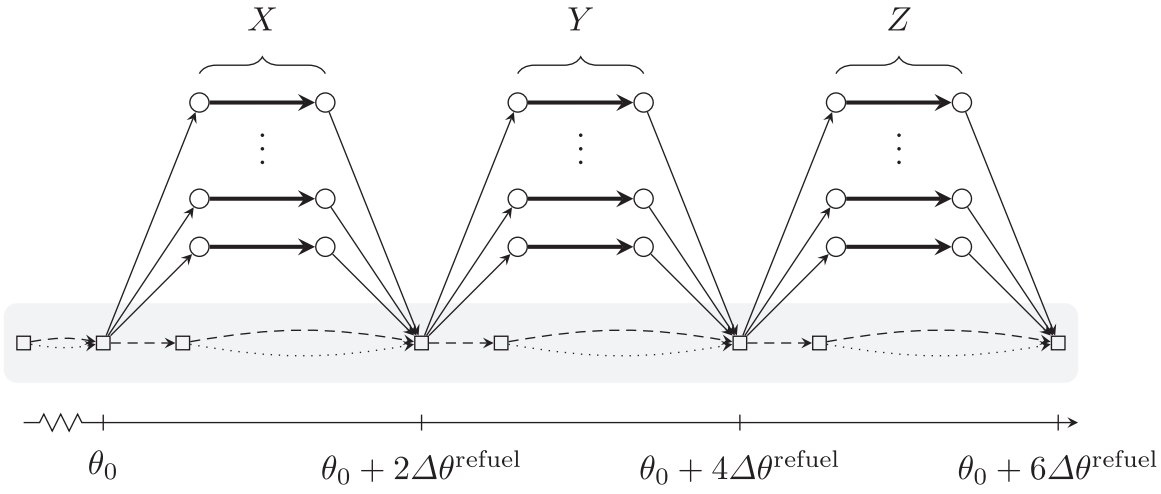
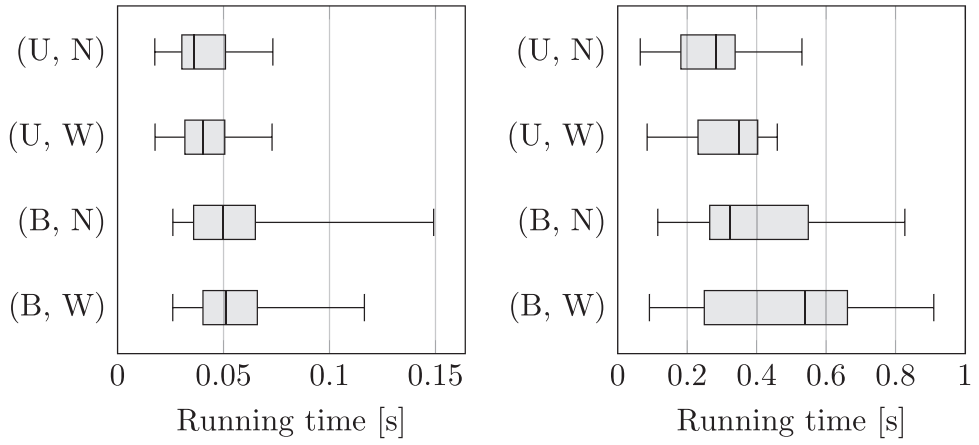


Fig. D.4. The refueling graph required for the proof of Theorem 3.1.

Appendix E. Additional tables and figures

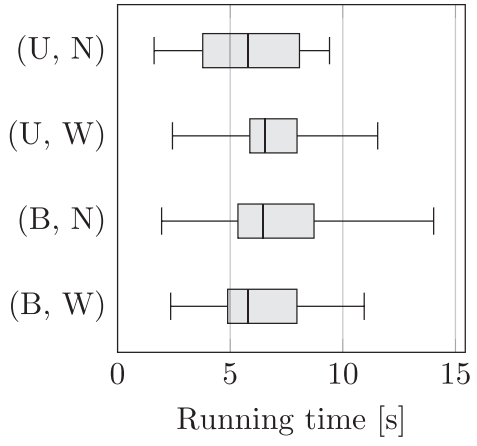
Table E.5
Performance of the set covering formulation (SC) and the fundamental path formulation (FP) over the artificial instances. An instance is defined by its size, whether it is uni- (U) or bidirectional (B), and whether its flight corridor is narrow (N) or wide (W), the objective is to minimize the number of required feeders. Running times are averaged arithmetically, including unsolved instances (counting with their time limit). The time limit was set to 3600 seconds except for the last one, marked with an asterisk, where it was increased to 12 hours.

Instance			Set covering		Fundamental path	
			# Solved	Running time (seconds)	# Solved	Running time (seconds)
100	U	N	10	0.13	10	0.04
		W	10	0.15	10	0.04
	B	N	10	0.19	10	0.06
		W	9	360.15	10	0.06
200	U	N	9	720.73	10	0.27
		W	7	1080.49	10	0.31
	B	N	5	1800.61	10	0.41
		W	4	2160.48	10	0.49
500	U	N	2	2882.08	10	5.77
		W	2	2882.66	10	6.89
	B	N	2	2881.55	10	7.16
		W	2	2882.82	10	6.33
1000	U	N	0	3600	10	98.88
		W	0	3600	10	193.19
	B	N	0	3600	10	223.23
		W	0	3600	10	271.27
2000	U	N	–	–	7	2606.16
		W	–	–	2	3183.06
	B	N	–	–	0	3600
		W	–	–	0	3600
2000*	U	N	–	–	10	3046.04
		W	–	–	10	6135.35
	B	N	–	–	6	25712.35
		W	–	–	6	25829.47

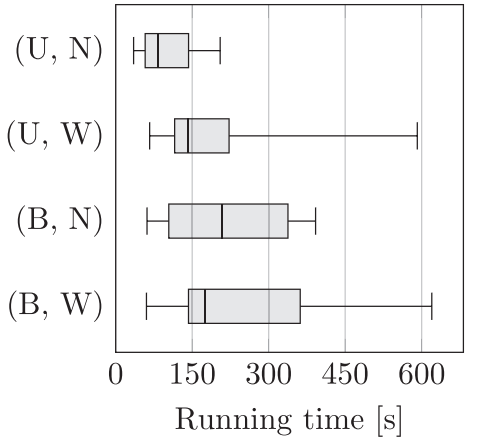


(a) Size: 100

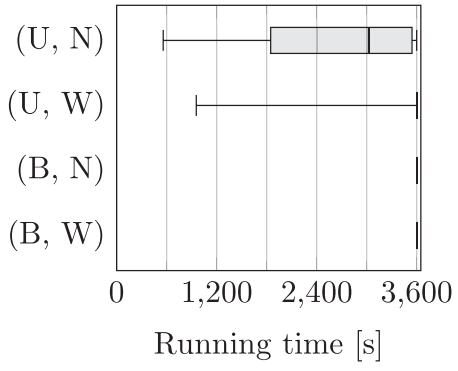
(b) Size: 200



(c) Size: 500



(d) Size: 1 000



(e) Size: 2 000

Fig. E.5. Distribution of the running times of the fundamental path formulation (FP) from Table E.5.

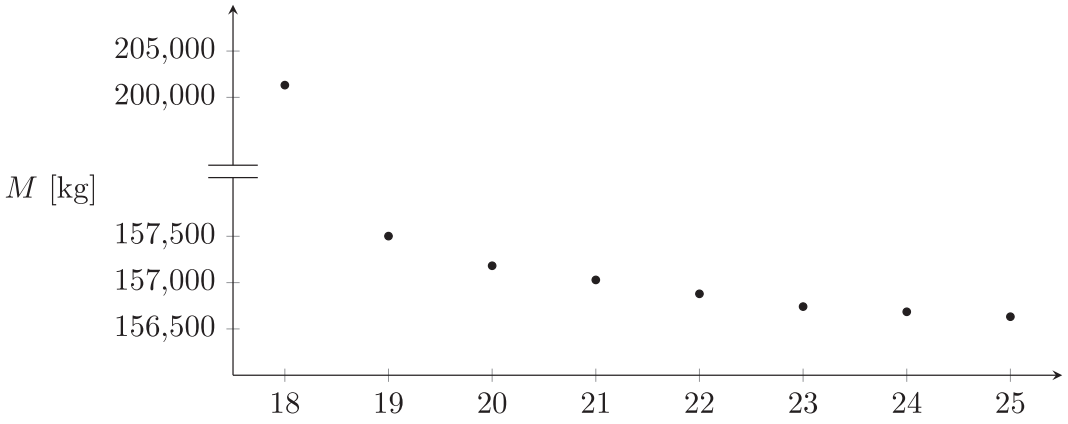


Fig. E.6. The trade-off between fuel burn and number of feeders for instance EINN.

References

Achterberg, T. (2009). SCIP: Solving constraint integer programs. *Mathematical Programming Computation, 1*(1), 1–41. https://doi.org/10.1007/s12532-008-0001-1.

Alrasheed, S. (2019). *Principles of mechanics: Fundamental university physics*. Springer Nature.

Anderson, J. D. (1999). *Aircraft performance and design*. McGraw-Hill.

Aneja, Y. P., Aggarwal, V., & Nair, K. P. (1983). Shortest chain subject to side constraints. *Networks, 13*(2), 295–302. https://doi.org/10.1002/net.3230130212.

Bast, H., Delling, D., Goldberg, A., Müller-Hannemann, M., Pajor, T., Sanders, P., et al., (2016). Route planning in transportation networks. In *Algorithm engineering* (pp. 19–80). Springer. https://doi.org/10.1007/978-3-319-49487-6_2.

Beasley, J. E., & Christofides, N. (1989). An algorithm for the resource constrained shortest path problem. *Networks, 19*(4), 379–394. https://doi.org/10.1002/net.3230190402.

Bennington, M., & Visser, K. (2005). Aerial refueling implications for commercial aviation. *Journal of Aircraft, 42*(2), 366–375.

Borndörfer, R. (1998). *Aspects of set packing, partitioning, and covering*. TU Berlin Ph.D. thesis..

Cordeau, J.-F. (2006). A branch-and-cut algorithm for the dial-a-ride problem. *Operations Research, 54*(3), 573–586. https://doi.org/10.1287/opre.1060.0283.

Delling, D., Pajor, T., & Wagner, D. (2009). Accelerating multi-modal route planning by access-nodes. In *Algorithms - ESA 2009* (pp. 587–598). Berlin, Heidelberg: Springer Berlin Heidelberg.

Delling, D., & Wagner, D. (2009). *Time-dependent route planning* (pp. 207–230)). Berlin, Heidelberg: Springer Berlin Heidelberg.

Desrosiers, J., Dumas, Y., Solomon, M. M., & Soumis, F. (1995). Time constrained routing and scheduling. *Handbooks in Operations Research and Management Science, 8*, 35–139. https://doi.org/10.1016/S0927-0507(05)80106-9.

Dumas, Y., Desrosiers, J., & Soumis, F. (1989). Large scale multi-vehicle dial-a-ride problems. In *Groupe d'études et de recherche en analyse des décisions*.

Eckhoff, J. (1993). Helly, radon, and carathéodory type theorems. In *Handbook of convex geometry* (pp. 389–448). Elsevier.

European Comission (2011). Flightpath 2050. Europe's Vision for Aviation. https://ec.europa.eu/transport/sites/transport/files/modes/air/doc/flightpath2050.pdf. Accessed April 16, 2019.

Ferdowsi, F., Maleki, H. R., & Rivaz, S. (2018). Air refueling tanker allocation based on a multi-objective zero-one integer programming model. *Operational Research*. https://doi.org/10.1007/s12351-018-0402-5.

Gamrath, G., & Lübbecke, M. (2010). Experiments with a generic Dantzig–Wolfe decomposition for integer programs. In *International symposium on experimental algorithms* (pp. 239–252). Springer. https://doi.org/10.1007/978-3-642-13193-6_21.

Garey, M. R., & Johnson, D. S. (1979). *Computers and intractability*. W.H. Freeman and Company, San Francisco.

Golden, B. L., Raghavan, S., & Wasil, E. A. (2008). *The vehicle routing problem: Latest advances and new challenges*: vol. 43. Springer Science & Business Media. https://doi.org/10.1007/978-0-387-77778-8.

Gurobi Optimization, L. (2018). Gurobi Optimizer Reference Manual. http://www.gurobi.com.

Hairer, E., Nørsett, S., & Wanner, G. (1993). *Solving ordinary differential equations i: Nonstiff problems*. Springer-Verlag Berlin Heidelberg. https://doi.org/10.1007/978-3-540-78862-1.

Handler, G. Y., & Zang, I. (1980). A dual algorithm for the constrained shortest path problem. *Networks, 10*(4), 293–309. https://doi.org/10.1002/net.3230100403.

Hansknecht, C. (2019). Air-to-air refueling instances. 10.6084/m9.figshare.8305988.v1

Hernandez, F., Feillet, D., Giroudeau, R., & Naud, O. (2016). Branch-and-price algorithms for the solution of the multi-trip vehicle routing problem with time windows. *European Journal of Operational Research, 249*(2), 551–559. https://doi.org/10.1016/j.ejor.2015.08.040.

ICAO (2018). Long-Term Traffic Forecasts. Passenger and Cargo. https://www.icao.int/sustainability/Pages/eap-fp-forecast-scheduled-passenger-traffic.aspx. Accessed April 16, 2019

Johnson, D. S. (1974). Approximation algorithms for combinatorial problems. *Journal of Computer and System Sciences, 9*(3), 256–278. https://doi.org/10.1016/S0022-0000(74)80044-9.

Jünger, M., Liebling, T. M., Naddef, D., Nemhauser, G. L., Pulleyblank, W. R., Reinelt, G., et al., (2009). *50 years of integer programming 1958–2008: From the early years to the state-of-the-art*. Springer Science & Business Media. https://doi.org/10.1007/978-3-540-68279-0.

Kaplan, S., & Rabadi, G. (2012). Exact and heuristic algorithms for the aerial refueling parallel machine scheduling problem with due date-to-deadline window and ready times. *Computers and Industrial Engineering, 62*(1), 276–285. https://doi.org/10.1016/j.cie.2011.09.015.

Kohl, N., Desrosiers, J., Madsen, O. B., Solomon, M. M., & Soumis, F. (1999). 2-path cuts for the vehicle routing problem with time windows. *Transportation Science, 33*(1), 101–116. https://doi.org/10.1287/trsc.33.1.101.

Korn, G. A., & Korn, T. M. (2013). *Mathematical handbook for scientists and engineers: Definitions, theorems, and formulas for reference and review*. Courier Corporation.

La Rocca, G., Li, M., van der Linden, P., & Elmendorp, R. (2014). Conceptual design of a passenger aircraft for aerial refueling operations. In *29th congress of the international council of the aeronautical sciences (ICAS 2014)* (pp. 162–172). International Council of Aeronautical Sciences - ICAS.

Laporte, G. (1992). The vehicle routing problem: An overview of exact and approximate algorithms. *European Journal of Operational Research, 59*(3), 345–358. https://doi.org/10.1016/0377-2217(92)90192-C.

Laporte, G., Gendreau, M., Potvin, J.-Y., & Semet, F. (2000). Classical and modern heuristics for the vehicle routing problem. *International Transactions in Operational Research, 7*(4–5), 285–300. https://doi.org/10.1111/j.1475-3995.2000.tb00200.x.

Li, M., & La Rocca, G. (2014). Conceptual design of joint-wing tanker for civil operations. In *Raes applied aerodynamics conference, Bristol, UK* (pp. 22–24).

Lübbecke, M., & Desrosiers, J. (2005). Selected topics in column generation. *Operations Research, 53*(6), 1007–1023. https://doi.org/10.1287/opre.1050.0234.

Mo, L. (2017). *Conceptual design study for in-flight refueling of passenger aircraft* Ph.D. thesis..

Morscheck, F., & Li, M. (2015). 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* (pp. 1B5–1). IEEE. https://doi.org/10.1109/DASC.2015.7311337.

Munari, P., Dollevoet, T., & Spliet, R. (2016). A generalized formulation for vehicle routing problems. arXiv preprint arXiv:1606.01935

Nangia, R. (2006). Efficiency parameters for modern commercial aircraft. *The Aeronautical Journal, 110*(1110), 495–510. https://doi.org/10.1017/S0001924000001391.

Petersen, B., & Jepsen, M. K. (2009). Partial path column generation for the vehicle routing problem with time windows. In *Proceedings of the 4th international network optimization conference (INOC 2009)*.

Pugliese, L. D. P., & Guerriero, F. (2013). Dynamic programming approaches to solve the shortest path problem with forbidden paths. *Optimization Methods and Software, 28*(2), 221–255. https://doi.org/10.1080/10556788.2011.630077.

RECREATE (2015). REsearch on a CRuiser Enabled Air Transport Environment. https://trimis.ec.europa.eu/project/research-cruiser-enabled-air-transport-environment. Accessed January 18, 2021.

Ropke, S., Cordeau, J.-F., & Laporte, G. (2007). Models and branch-and-cut algorithms for pickup and delivery problems with time windows. *Networks: An International Journal, 49*(4), 258–272. https://doi.org/10.1002/net.20177.

Schmied, M., Wüthrich, P., Zah, R., Friedl, C., et al., (2015). Postfossile Energieversorgungsoptionen für einen treibhausgasneutralen Verkehr im Jahr 2050: Eine verkehrsträgerübergreifende Bewertung. *Umweltbundesamt Texte, 2015*(30), 1–115.

Smith, O., & Savelsbergh, M. (2014). A note on shortest path problems with forbidden paths. *Networks, 63*(3), 239–242. https://doi.org/10.1002/net.21541.

Thomas, P. R., Bhandari, U., Bullock, S., Richardson, T. S., & du Bois, J. L. (2014). Advances in air to air refuelling. *Progress in Aerospace Sciences, 71*, 14–35. https://doi.org/10.1016/j.paerosci.2014.07.001.

Timmermans, I., & La Rocca, G. (2014). Conceptual design of a flying boom for air-to-air refueling of passenger aircraft. In *Aip conference proceedings: vol. 1618* (pp. 398–401). American Institute of Physics. https://doi.org/10.1063/1.4897758.

Tsukerman, A., Weiss, M., Shima, T., Löbl, D., & Holzapfel, F. (2018). Optimal rendezvous guidance laws with application to civil autonomous aerial refueling. *Journal of Guidance, Control, and Dynamics, 41*(5), 1167–1174. https://doi.org/10.2514/1.G003154.

Wolsey, L. (1998). *Integer programming*. Wiley.