Brandenburg
University of Technology
Cottbus - Senftenberg

An Iterative Graph Expansion Approach for the Scheduling and Routing of Airplanes

Fabian Gnegel
Armin Fügenschuh

Cottbus Mathematical Preprints
COMP # 3(2019)

An Iterative Graph Expansion Approach for the Scheduling and Routing of Airplanes

Fabian Gnegel and Armin Fügenschuh

March 22, 2019

F. Gnegel, Brandenburg University of Technology Cottbus - Senftenberg, Platz der Deutschen Einheit 1, 03046 Cottbus, Germany
E-mail address: gnegel@b-tu.de

A. Fügenschuh, Brandenburg University of Technology Cottbus - Senftenberg, Platz der Deutschen Einheit 1, 03046 Cottbus, Germany
E-mail address: fuegenschuh@b-tu.de

Abstract

A tourism company that offers fly-in safaris is faced with the challenge to route and schedule its fleet of airplanes in an optimal way. Over the course of a given time horizon several groups of tourists have to be picked up at airports and flown to their destinations within a certain time-window. Furthermore the number of available seats, the consumption of fuel, the maximal takeoff weight, and restrictions on the detour of the individual groups have to be taken into account. The task of optimally scheduling the airplanes and tour groups belongs to the class of vehicle routing problems with pickup and delivery and time-windows. A flow-over-flow formulation on the time expanded graph of the airports was used in the literature in order to model this problem as a mixed integer linear program. Most of the benchmark problems however could not be solved within a time limit of three hours, which was overcome by formulating the problem for a simplified (time-free) graph and the use of an incumbent callback to check for feasibility in the original graph. While this approach led to very good results for instances, where few time-free solutions were infeasible for the original problem, some instances remained unsolved. In order to overcome this problem we derive two new exact formulations that include time as variables. Although these formulations by themselves are not better than the approach from the literature, they allow for an effective construction of graphs which can be interpreted as intermediate graphs between the graph of airports and the expanded graph with vertices for each visit. Using similar relaxation techniques to the time-free approach and constructing these graphs based on solutions of the relaxations guarantees that only critical airports are expanded. A computational study was performed in order to compare the new formulations to the methods from the literature. Within a time limit of 3 hours the new approach was able to find proven optimal solutions for all previously unsolved benchmark instances. Furthermore the average computation time of all benchmark instances was reduced by 90 percent.

Keywords: Mixed Integer Linear Programming, Vehicle Routing Problem, Time-Dependent Airplane Routing, Dynamic Graph Expansion.

1 Introduction and Literature Review

A company in southern Africa is offering safaris at several locations and is flying groups of tourists from one location to another with small airplanes (with up to 12 passenger seats). Everyday several tour groups have to be picked up at their respective current locations and to be flown to their next destination within a given time-window. The operating cost mainly depends on the overall distance flown by the airplanes, so that an optimized routing is crucial to the success of the company. Since the tour groups usually consist of less people than the number of seats on the airplanes they are not necessarily flown directly to their destination. It is possible that their routes include intermediate stops for refueling the airplane or picking up other groups on the way. The number of intermediate stops and the length of the detour however are bounded in order to reduce the inconvenience this practice imposes on each group. Furthermore fuel consumption has to be tracked, not only because not all airports provide fuel for each airplane, but also due to differing weight limits for takeoff and landing at the airports. We call this problem the air-travel scheduling and routing problem (ASRP).

Although the ASRP is very specific in its constraints it does fit into the general framework of vehicle routing problems with pickup and delivery and time-windows (VRP-PD-TW), to which just finding a solution is already NP-hard, as shown by Savelsbergh [15], for the less general traveling salesman problem with time-windows. In the literature, see for example Savelsbergh and Sol [16], general mixed integer linear programs (MILP) for VRP-PD-TW can be found. These VRP problems are usually defined on graphs whose only vertices are origins and destinations of the transportation requests. In the ASRP, however, the number of stops at airports that provide fuel is not known a priori, which necessitates the inclusion of additional vertices. Furthermore in the ASRP several requests share origins and destinations, which in the VRP-PD-TW setting require separate vertices connected by arcs of length zero. This introduces short cycles, and therefore weakens the linear relaxations. For this reason a new, very specific MILP was derived for the ASRP.

Fügenschuh et al. [9] were the first to study the ASRP. They proposed a time-indexed MILP formulation based on a time-expanded network [7] derived from the graph of airports. Already for problem instances with very few transportation requests, airplanes and airports the required number of variables and constraints was too large to be handled by state-of-the-art MILP solvers. Hence, when they implemented the time-indexed formulation for instances derived from real data, even if the fleet was restricted to two airplanes, a 3 hour time limit for a MILP solver did not suffice to find an optimal solution and prove its optimality.

In order to reduce the problem size they then used, what they called time-free relaxations. This relaxation is structurally the same model but defined for the graph of airports with integer (and not binary) variables for the airplane route instead of the time expanded graph with binary variables. Since this formulation avoids all time indexes, it is by no means guaranteed that solutions of the relaxations could be expanded to feasible solutions of the time-indexed model. In fact, they might not even form continuous paths for the airplanes. The benefit of this relaxation, however, is that often within seconds it provided much better lower bounds than the classical LP relaxation of the time-expanded models. With an incumbent callback that cut off all solutions without a feasible expansion and a high quality primal heuristic they were then able to reduce the optimality gap on average to less than 5 % and even prove optimality of the solution in many cases.

The time-free formulation in [9] can be seen as a special case of what is sometimes referred to as a time-bucket formulation [6] and has been applied to many graph based problems with time-windows. The idea of these formulations is to use a time-expanded formulation, in which the vertices are associated with time-intervals instead of single points of time and link them with arc if one interval can be reached from the other. Other authors such as Wang and Regan [20] call this the under-constrained network. Wang and Regan [21] also study the convergence of different approaches of refinement of the intervals. The time-free approach in [9] formulation takes it to the extreme as it uses only one interval (the whole time horizon) for each vertex.

Recently these kind of graphs have been used by Boland et al. [2] and He et al. [12] in their dynamic discretization discovery algorithm, which they successfully applied to routing problems constrained by time-windows. The idea of this algorithm is to define relaxations based on time-bucket formulations and iteratively refine the buckets until the optimal solution of the relaxation is feasible for the time expanded graph and therefore proven to be the optimal solution. Another use of time-bucket formulations was found by Dash et al. [6], where a good time-bucket formulation for the traveling salesman problem with time-windows is found by a heuristic and cutting planes are used to cut off all infeasible solutions occurring due to the coarse refinement of the graph.

The problem we faced when using these methods was that even while the relaxations still permitted cycles, the the time limit was exceeded for most benchmark instances. We attribute this mainly to the fact that airports can be visited more than once and that it is not a priori known how often an airport has to be visited. Cycles are therefore not necessarily infeasible and cannot be cut off with additional valid inequalities for eliminating subtours as Vu et al. [19] did to enhance the performance of the dynamic discretization discovery algorithm and Dash et al. [6]

did to guarantee feasibility.

Another and maybe more common approach for dealing with time-window constraints is to include the aspect of time explicitly with variables. For example, Solomon and Desrosiers [17] use a model in their survey for vehicle routing problems with time-windows that includes variables for the service time of each vertex. Gavish and Graves [10] give the first MILP formulation that uses time variables indexed by arcs instead of vertices. However, both of these formulations cannot directly be applied to the ASRP problem studied here, since vertices as well as arcs might appear multiple times in the airplane paths of the optimal solution.

In this work we derive two formulations for the ASRP that combine ideas from time-bucket formulations with arc based time-variables and extend the results and ideas we presented in [11]. We construct a network in which multiple vertices correspond to a single airport (as in the time-bucket formulation), but we do not associate a certain time or time-bucket with the vertices. Instead we require that each vertex can only be visited at most once and leave the arrival times as variables in the MILP. It can therefore be considered as an expanded formulation with variable time assignments. The first formulation, ASRP-ARC, is closer to the formulation given in [9] and uses arc based variables for the requests. The second, ASRP-PATH, uses an extensive formulation for the paths of the requests, where an x-variable in the MILP corresponds to a feasible path of a tour group. Such a formulation was first used for vehicle routing problems by Balinski and Quandt [1] and has been used as the basis of several column generation approaches by Toth and Vigo [18]. The time-free relaxation in [9] can be obtained from the first MILP by using only one vertex per airport and allowing vertices to be visited more than once. With the time-free relaxation as a basis we derive an algorithm, similar to the dynamic discretization discovery algorithm of Boland et al. [2], to iteratively increase the number of vertices associated with each airport. Only few iterations of this algorithm were necessary to find relaxations for all benchmark instances that have an optimal solution that is feasible and therefore also optimal for the ASRP. Furthermore, at the same time average computation time for the whole solution process, comparing to the time-free approach run on the same hardware, was reduced from more than 4000 seconds to less than 400 seconds.

2 Two New MILP Formulations of the Problem

Following the notations introduced by Fügenschuh et al. [9] in the original description of the ASRP we give a detailed description of it to provide the context and necessary notations for understanding the following sections.

The fleet of airplanes of the company is denoted by the set $\mathcal{P}$, whose elements correspond to the airplanes that have to be scheduled and routed. The fleet of airplanes is inhomogeneous and therefore different airplanes might require different parameter values in the model. For each airplane $p \in \mathcal{P}$ the number of passenger seats is given by $\overline{s}_p$ and the required type of fuel by $\rho_p \in \mathcal{F}$, where $\mathcal{F}$ is the set of all fuel types.

When we use the term airport in the problem description we use it rather generously as even improvised runways that provide nothing but a location to takeoff and land might be considered airports in the ASRP. The set of all airports are denoted by $\mathcal{V}$ and the flight connections between them by $\mathcal{A} \subseteq \mathcal{V} \times \mathcal{V}$. For each airport $i \in \mathcal{V}$ we denote by $\mathcal{F}_i$ the set of available fuel types. For an airplane $p \in \mathcal{P}$ and a trip $(i,j) \in \mathcal{A}$ the flight distance is given by $d_{i,j} \in \mathbb{R}^+$, the traveling cost by $c^p_{i,j} \in \mathbb{R}^+$, the flight duration by $\delta^p_{i,j} \in \mathbb{R}^+$, the required fuel by $\gamma^p_{i,j} \in \mathbb{R}^+$, the fuel capacity by $\overline{f}^p_{i,j} \in \mathbb{R}^+$, and the maximum total payload at takeoff by $\overline{w}^p_{i,j} \in \mathbb{R}^+$.

Each tour group is associated with a flight request (request, for short), $r \in \mathcal{R}$, with $\mathcal{R}$ as a notation for the set of all requests. For a request $r \in \mathcal{R}$ the number of required seats $s_r \in \mathbb{N}$ and the total weight of the passengers and their luggage w_r has to be specified. The routes of the airplanes and requests are further restricted as follows. Each airplane $p \in \mathcal{P}$ has a given starting point $D_p \in \mathcal{V}$ and a given final location $A_p \in \mathcal{V}$. For each request $r \in \mathcal{R}$ the departure airport is specified by $D_r \in \mathcal{V}$ and the destination by $A_r \in \mathcal{V}$. Their routes are further restricted by an upper bound on the maximal allowed detours $\overline{d}_r \in \mathbb{R}^+$ and a maximal allowed number of intermediate stops $M_r \in \mathbb{N}_0$.

For the scheduling constraints, the airplanes can only operate between a time $\tau^p_{\text{dep}} \in \mathbb{R}^+$ and a time $\tau^p_{\text{arr}} \in \mathbb{R}^+$, and the requests $r \in \mathcal{R}$ cannot be picked up earlier than $\tau^r_{\text{dep}} \in \mathbb{R}^+$ and have to arrive at their destination before $\tau^r_{\text{arr}} \in \mathbb{R}^+$. Finally, there are lower bounds $\underline{\varphi}_p \in \mathbb{R}^+$ and upper bounds $\overline{\varphi}_p \in \mathbb{R}^+$ for the amount of fuel carried at the initial departure and similar bounds $\underline{\psi}_p \in \mathbb{R}^+$ and $\overline{\psi}_p \in \mathbb{R}^+$ on the amount of fuel at the final airport of each airplane $p \in \mathcal{P}$, in order to avoid solutions with low-fuel airplanes far away from refueling airports.

Now, in order to derive the new MILP formulation for this problem we introduce some additional notations. We denote by n_i a given upper bound on the number of trips starting or ending in airport $i \in \mathcal{V}$ (if not given in the data valid bounds can be calculated by taking flight durations and the service times of the airplanes into account). We use these to define the extended set of airports $\mathcal{V}^{\text{ext}}$ as the set of all tuples $i = (i_1, i_2)$ with $i_1 \in \mathcal{V}$ and $i_2 \in \{1, \dots, n_{i_1}\}$ and the extended set of trips $\mathcal{A}^{\text{ext}} \subset \mathcal{V}^{\text{ext}} \times \mathcal{V}^{\text{ext}}$, with $(i,j) \in \mathcal{V}^{\text{ext}} \times \mathcal{V}^{\text{ext}}$ iff $(i_1, j_1) \in \mathcal{A}$. For ease of notation all parameters defined for airports in $\mathcal{V}$ and trips in $\mathcal{A}$ may be indexed by corresponding airports in $\mathcal{V}^{\text{ext}}$ or trips in $\mathcal{A}^{\text{ext}}$ and take the same value. An overview for of the sets and parameters can be found in Table 1 and Table 2.

Notation	Explanation
$\mathcal{P}$	set of airplanes
$\mathcal{V}$	set of airports
$\mathcal{A}$	set of connections between airports
$\mathcal{R}$	set of requests
$\mathcal{F}$	set of fuel types
$\mathcal{V}^{\text{ext}}$	extended set of airports
$\mathcal{A}^{\text{ext}}$	extended set of connections

Table 1: Sets for describing the ASRP.

Parameter	Explanation
$\overline{s}_p$	Number of seats on airplane p
ρ_p	Required fuel type of airplane p
$\mathcal{F}_i$	available fuel types at airport i
$d_{i,j}$	distance between airports i and j
$c^p_{i,j}$	cost of airplane p flying from i to j
$\delta^p_{i,j}$	duration of airplane p flying from i to j
$\gamma^p_{i,j}$	fuel consumption of airplane p on trip (i,j)
$\overline{f}^p_{i,j}$	maximal fuel capacity of airplane p on trip (i,j)
$\overline{w}^p_{i,j}$	maximal payload of airplane p on trip (i,j)
s_r	number of passengers of request r
w_r	weight associated with request r
D_p, A_p	arrival and departure airports of airplane p
D_r, A_r	arrival and departure airports of request r
$\overline{d}_r$	maximal allowed detour for request r
M_r	maximal number of stops for request r
$[\tau^p_{\text{dep}}, \tau^p_{\text{arr}}]$	timw-window of airplane p
$\left[\tau^r_{\text{dep}}, \tau^r_{\text{arr}}\right]$	time-window of request r
$\underline{\varphi}_p$	minimal amount of fuel on airplane p at departure
$\overline{\varphi}_p$	maximal amount of fuel on airplane p at departure
$\underline{\psi}_p$	minimal amount of fuel on airplane p at arrival
$\overline{\psi}_p$	maximal amount of fuel on airplane p at arrival
n_i	upper bound on the visits to airport i

Table 2: Parameters in the mathematical model.

2.1 A Time-Flow Formulation for the ASRP

The time-expanded formulation introduced by Fügenschuh et al. [9] had the major draw back that the MILPs were too large to be solved by state-of-the-art solvers, even for problems with only two airplanes. They circumvented this problem by eliminating the time index and using an incumbent callback to check if the solution obtained by this relaxation was feasible for the original model. If this was not the case the incumbent was rejected. The derivation of high quality time-bucket formulations is difficult since the time-windows are associated with the requests and not the nodes (airports), furthermore sub-tour elimination cuts, which have been successfully used for many touring problems [5, 18], cannot be used since airports can be visited multiple times and cycles therefore often appear in feasible solutions. Thus we give a new formulation, that uses a commodity flow formulation with variables indexed by the arcs (which can traced back to Gavish and Graves [10] for the traditional VRP) of a network. This network has the extended set of airports $\mathcal{V}^{\text{ext}}$ as vertices and the extended set of trips $\mathcal{A}^{\text{ext}}$ as arcs. As it is a model for the same problem, the variables and constraints are mostly adapted from the ones proposed in [9] to the extended network, with additional variables and constraints for the modeling of the aspect of time.

2.1.1 Decision Variables

For the decisions concerning the routes of the airplanes we introduce variables

$$\forall\,(i,j)\in\mathcal{A}^{\text{ext}},p\in\mathcal{P}:\; y^{p}_{i,j}\in\{0,1\}, \tag{1a}$$

which are set to 1, iff (i,j) is part of the route of airplane p. Furthermore, due to the formulation on the expanded network it is necessary to decide at which copy of its origin (destination) airport the airplane departs (arrives). This decision is modeled by the variables

$$\forall\,p\in\mathcal{P},i\in\mathcal{V}^{\text{ext}}\text{ s.t. }i_1=D_p: y^{p}_{\text{dep},i}\in\{0,1\}, \tag{1b}$$

$$\forall\,p\in\mathcal{P},i\in\mathcal{V}^{\text{ext}}\text{ s.t. }i_1=A_p: y^{p}_{\text{arr},i}\in\{0,1\}, \tag{1c}$$

where a variable set to 1 corresponds to the decision of starting (finishing) at the respective copy. Analogously the decisions for the routes of the requests are expressed by variables

$$\forall\,r\in\mathcal{R},(i,j)\in\mathcal{A}^{\text{ext}},p\in\mathcal{P}:\; x^{r,p}_{i,j}\in\{0,1\} \tag{1d}$$

and the decisions concerning departure and arrival of the requests by variables

$$\forall\,r\in\mathcal{R},p\in\mathcal{P},i\in\mathcal{V}^{\text{ext}}\text{ s.t. }i_1=D_r: x^{r,p}_{\text{dep},i}\in\{0,1\}, \tag{1e}$$

$$\forall\,r\in\mathcal{R},p\in\mathcal{P},i\in\mathcal{V}^{\text{ext}}\text{ s.t. }i_1=A_r: x^{r,p}_{\text{arr},i}\in\{0,1\}. \tag{1f}$$

The fuel consumption of the airplanes is tracked by continuous variables,

$$\forall\,(i,j)\in\mathcal{A}^{\text{ext}},p\in\mathcal{P}:\; f^{p}_{i,j}\in\mathbb{R}_+, \tag{1g}$$

that take the amount of fuel on board of airplane p at takeoff from i on its trip to j as their values. For the fuel on airplane $p\in\mathcal{P}$ at its first departure and the remaining fuel at its last arrival we further introduce continuous variables

$$\forall\,p\in\mathcal{P},i\in\mathcal{V}^{\text{ext}}\text{ s.t. }i_1=D_p: f^{p}_{\text{dep},i}\in\left[0,\overline{\varphi}_p\right], \tag{1h}$$

$$\forall\,p\in\mathcal{P},i\in\mathcal{V}^{\text{ext}}\text{ s.t. }i_1=A_p: f^{p}_{\text{arr},i}\in\left[0,\overline{\psi}_p\right]. \tag{1i}$$

The payload at takeoff of airplane p on the trip from i to j is given by the continuous variables

$$\forall\,p\in\mathcal{P},(i,j)\in\mathcal{A}^{\text{ext}}:\; w^{p}_{i,j}\in\mathbb{R}_+. \tag{1j}$$

Similarly to the tracking of fuel consumption, the schedule of the airplanes departures and arrivals are tracked by continuous variables

$$\forall\,(i,j)\in\mathcal{A}^{\text{ext}},p\in\mathcal{P}:\; t^{p}_{i,j}\in\mathbb{R}^+, \tag{1k}$$

that take the time airplane p arrives at j after starting at i as its value. For the initial and final time we introduce variables

$$\forall\,p\in\mathcal{P},i\in\mathcal{V}^{\text{ext}}\text{ s.t. }i_1=D_p: t^{p}_{\text{dep},i}\in\left[0,\tau^{p}_{\text{dep}}\right], \tag{1l}$$

$$\forall\,p\in\mathcal{P},i\in\mathcal{V}^{\text{ext}}\text{ s.t. }i_1=A_p: t^{p}_{\text{arr},i}\in\left[0,\tau^{p}_{\text{arr}}\right]. \tag{1m}$$

2.1.2 Constraints

The following constraints are formulated to ensure that each feasible variable assignment corresponds to an admissible solution of the ASRP. In both the movements of the airplanes, requests and the consumption of fuel are modeled as commodity flows. In ASRP, however, we also model the time as another commodity flow. Flow conservation is imposed on the variables for the routes of the airplanes with the constraints

$$\begin{aligned} \forall\, j \in \mathcal{V}^{\text{ext}}, p \in \mathcal{P} : \sum_{i:(i,j)\in\mathcal{A}^{\text{ext}}} y_{i,j}^{p} + \begin{cases} y_{\text{dep},j}^{p}, & \text{if } j_1 = D_p, \\ 0, & \text{else,} \end{cases} \\ = \sum_{k:(j,k)\in\mathcal{A}^{\text{ext}}} y_{j,k}^{p} + \begin{cases} y_{\text{dep},j}^{p}, & \text{if } j_1 = A_p, \\ 0, & \text{else.} \end{cases} \end{aligned} \tag{2a}$$

Analogously flow has to be preserved on the routes of the requests, which is guaranteed by the conditions

$$\begin{aligned} \forall\, j \in \mathcal{V}^{\text{ext}}, r \in \mathcal{R}, p \in \mathcal{P} : \sum_{i:(i,j)\in\mathcal{A}^{\text{ext}}} x_{i,j}^{r,p} + \begin{cases} x_{\text{dep},j}^{r,p}, & \text{if } j_1 = D_r, \\ 0, & \text{else,} \end{cases} \\ = \sum_{k:(j,k)\in\mathcal{A}^{\text{ext}}} x_{j,k}^{r,p} + \begin{cases} x_{\text{dep},j}^{r,p}, & \text{if } j_1 = A_r, \\ 0, & \text{else.} \end{cases} \end{aligned} \tag{2b}$$

Additionally we have to ensure that each request is served by exactly one airplane by requiring

$$\forall\, r \in \mathcal{R} : \sum_{\substack{i\in\mathcal{V}^{\text{ext}} \\ i_1 = D_r}} \sum_{p\in\mathcal{P}} x_{\text{dep},i}^{r,p} = 1. \tag{2c}$$

The bound on the number of intermediate stops of a request is ensured by putting constraints

$$\forall\, r \in \mathcal{R} : \sum_{(i,j)\in\mathcal{A}^{\text{ext}}} \sum_{p\in\mathcal{P}} x_{i,j}^{r,p} \le M_r + 1 \tag{2d}$$

and the restrictions on the detours are ensured by constraints

$$\forall\, r \in \mathcal{R} : \sum_{(i,j)\in\mathcal{A}^{\text{ext}}} \sum_{p\in\mathcal{P}} d_{i,j} \cdot x_{i,j}^{r,p} \le \bar{d}_r. \tag{2e}$$

The airplane routes and request routes are coupled by seat capacity constraints

$$\forall\, (i,j) \in \mathcal{A}^{\text{ext}}, p \in \mathcal{P} : \sum_{\substack{r\in\mathcal{R} \\ (i,j)\in\mathcal{A}^{\text{ext}}}} s_r \cdot x_{i,j}^{r,p} \le \bar{s}_p \cdot y_{i,j}^{p}. \tag{2f}$$

Since there are some airports that provide the required fuel type for an airplane and some that do not, two different flow conditions are put for the fuel variables. For airports $j \in \mathcal{V}$ without fuel provision for an airplane $p \in \mathcal{P}$, that means $\rho_p \notin \mathcal{F}_j$, we impose

$$\begin{aligned} \forall\, j \in \mathcal{V}^{\text{ext}}, p \in \mathcal{P} : \sum_{i:(i,j)\in\mathcal{A}^{\text{ext}}} f_{i,j}^{p} + \begin{cases} f_{\text{dep},j}^{p}, & \text{if } j_1 = D_p, \\ 0, & \text{else,} \end{cases} \\ = \sum_{k:(j,k)\in\mathcal{A}^{\text{ext}}} \left(f_{j,k}^{p} + \gamma_{j,k}^{p} \cdot y_{j,k}^{p} \right) + \begin{cases} f_{\text{arr},j}^{p}, & \text{if } j_1 = A_p, \\ 0, & \text{else.} \end{cases} \end{aligned} \tag{2g}$$

and otherwise, if $\rho_p \in \mathcal{F}_j$ holds

$$\begin{aligned} \forall\, j \in \mathcal{V}^{\text{ext}}, p \in \mathcal{P} : \sum_{i:(i,j)\in\mathcal{A}^{\text{ext}}} f_{i,j}^{p} + \begin{cases} f_{\text{dep},j}^{p}, & \text{if } j_1 = D_p, \\ 0, & \text{else,} \end{cases} \\ \le \sum_{k:(j,k)\in\mathcal{A}^{\text{ext}}} \left(f_{j,k}^{p} + \gamma_{j,k}^{p} \cdot y_{j,k}^{p} \right) + \begin{cases} f_{\text{arr},j}^{p}, & \text{if } j_1 = A_p, \\ 0, & \text{else.} \end{cases} \end{aligned} \tag{2h}$$

For setting the correct bounds on the initial and final values we further require

$$\forall\, p \in \mathcal{P}, i \in \mathcal{V}^{\text{ext}} \text{ s.t. } i_1 = D_p : f^p_{\text{dep},i} \geq y^p_{\text{dep},i} \cdot \underline{\varphi}_p, \tag{2i}$$

$$\forall\, p \in \mathcal{P}, i \in \mathcal{V}^{\text{ext}} \text{ s.t. } i_1 = A_p : f^p_{\text{arr},i} \geq y^p_{\text{arr},i} \cdot \underline{\psi}_p. \tag{2j}$$

Similarly to the request variables the fuel variables are coupled to the airplane variables by capacity constraints

$$\forall\, (i,j) \in \mathcal{A}^{\text{ext}}, p \in \mathcal{P} : f^p_{i,j} \leq \overline{f}^p_{i,j} \cdot y^p_{i,j}. \tag{2k}$$

The payload weight is the sum of the weight of the assigned requests and the fuel on board, so we set

$$\forall\, (i,j) \in \mathcal{A}^{\text{ext}}, p \in \mathcal{P} : w^p_{i,j} = \sum_{r \in \mathcal{R}} w_r \cdot x^{r,p}_{i,j} + f^p_{i,j}. \tag{2l}$$

The restrictions on the maximum takeoff and the maximum landing weights depending on the airports are incorporated by upper bounds on the payload weights

$$\forall\, (i,j) \in \mathcal{A}^{\text{ext}}, p \in \mathcal{P} : w^p_{i,j} \leq \overline{w}^p_{i,j} \cdot y^p_{i,j}. \tag{2m}$$

Although time is not a physical commodity the time variables can be seen as an abstract resource that is picked up on the way. Therefore we impose flow conservation constraints

$$\begin{aligned} \forall\, j \in \mathcal{V}^{\text{ext}}, p \in \mathcal{P} : \sum_{i:(i,j) \in \mathcal{A}^{\text{ext}}} \left(t^p_{i,j} + \delta^p_{i,j} \cdot y^p_{i,j}\right) + \begin{cases} \tau^p_{\text{dep}}, & \text{if } j_1 = D_p \text{ and } j_2 = 1, \\ 0, & \text{else,} \end{cases} \\ \leq \sum_{k:(j,k) \in \mathcal{A}^{\text{ext}}} t^p_{j,k} + \begin{cases} \tau^p_{\text{arr}}, & \text{if } j_1 = A_p, \\ 0, & \text{else.} \end{cases} \end{aligned} \tag{2n}$$

The values of the variables for the initial departure (final arrival) time have to be restricted by capacity constraints

$$\forall\, p \in \mathcal{P}, i \in \mathcal{V}^{\text{ext}} \text{ s.t. } i_1 = D_p : \; t^p_{\text{dep},i} \geq y^p_{\text{dep},i} \cdot \tau^p_{\text{dep}}. \tag{2o}$$

In [9] the time-windows of the requests were implicitly imposed by the construction of the time-expanded graph. However, this is not possible when time-variables are used. Therefore we have to use big-M-type constraints, to guarantee feasibility of the schedule of requests. Hence, we impose

$$\forall\, p \in \mathcal{P}, (i,j) \in \mathcal{A}^{\text{ext}}, r \in \mathcal{R} : \; t^p_{i,j} \leq \tau^r_{\text{arr}} + M(1 - x^{r,p}_{i,j}), \tag{2p}$$

in order to guarantee feasibility of the arrival times of the request schedules. These constraints are only restrictive when a request variable is set to 1 and therefore a request is routed on the respective arc. In order to guarantee feasibility of the departure time we analogously impose

$$\forall\, p \in \mathcal{P}, (i,j) \in \mathcal{A}^{\text{ext}}, r \in \mathcal{R} : \; t^p_{i,j} \geq \tau^r_{\text{dep}} - 2M(1 - x^{r,p}_{i,j}) - M(y^p_{i,j} - 1). \tag{2q}$$

The flow conservation constraints only guarantee feasible assignments to the included variables if each vertex can only be visited once. One of the problems with using a commodity flow formulation and allowing nodes of the network to be visited multiple times is illustrated in Figure 1 for the time variables. The flow on the sub-cycle can be shifted by any nonnegative constant K and therefore allowing for non-admissible variable assignments. So, in order to prevent commodities from skipping sub-cycles multiple visits to the same copy of an airports have to be abrogated and we impose

$$\forall\, j \in \mathcal{V}^{\text{ext}}, p \in \mathcal{P} : \sum_{i:(i,j) \in \mathcal{A}^{\text{ext}}} y^p_{i,j} \leq 1, \tag{2r}$$

$$\forall\, j \in \mathcal{V}^{\text{ext}}, p \in \mathcal{P} : \sum_{i:(j,k) \in \mathcal{A}^{\text{ext}}} y^p_{j,k} \leq 1. \tag{2s}$$

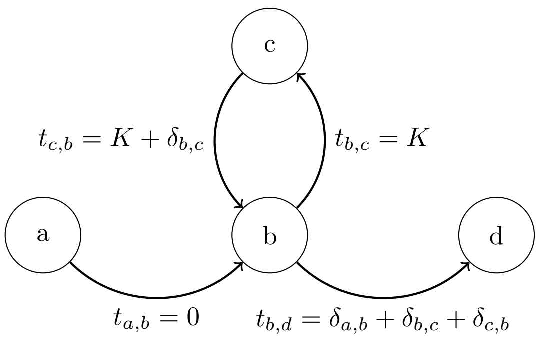
c
a
b
d
$t_{c,b} = K + \delta_{b,c}$
$t_{b,c} = K$
$t_{a,b} = 0$
$t_{b,d} = \delta_{a,b} + \delta_{b,c} + \delta_{c,b}$

Figure 1: Possible variable assignments for a path with sub-cycle.

2.1.3 The MILP

In addition to these constraints we included all cuts that were already identified by Fügenschuh et al. [9] to be effective for this problem in order to define the MILP, that we will refer to as the Time-Flow Formulation (ASRP-ARC) for the Airline Routing and Scheduling Problem:

$$\begin{aligned} \text{minimize} \quad & \sum_{(i,j)\in\mathcal{A}^{\text{ext}}} \sum_{p\in\mathcal{P}} c^p_{i,j} \cdot y^p_{i,j} \\ \text{subject to} \quad & (1a) - (1m), \\ & (2a) - (2s). \end{aligned} \tag{ASRP-ARC}$$

The objective function is simply a linear combination of the respective variables for the airplane routes, because it is assumed that the operating cost only depends on the paths of the airplanes.

2.1.4 Computational Results

In this section we present computational results for the new formulation on the instances from Fügenschuh et al. [9], that they used to show the effectiveness of the time-free approach. These instances were derived from real world data, in which the fleet was restricted to contain two airplanes with 5 or 12 seats. While the number of requests ranges from 10 to 23 and only 8 to 13 airports have to be considered. All results reported in this paper were obtained on a Macbook-Pro 2017 with an 3.1 GHz Intel Core i7 CPU and 16 GB of RAM. For solving the MILPs Cplex 12.6.3.0 was used.
The results for the ASRP-ARC can be found in Table 3 and the results for the time-free approach results are reported in Table 4. In order to allow for a comparison the solver was initialized in both cases with a primal solution found by the heuristic introduced by Fügenschuh et al. [9]. For the results on the ASRP-ARC we further note that a tight estimation of the upper bounds on the number of copies of the airports that provide fuel for an airplane is very difficult and is highly dependent on a lot of the instance parameters. In order to avoid very large upper bounds for the number of visits at those airports, the n_i were estimated heuristically by taking the number of visits in the primal solution plus two. Consequently a reported gap of 0% does not imply that the found solution was proven optimal. However, in all cases the optimal solution coincided with the ones found by an exact method introduced in the later sections.
The computation times reported for both problems show a high variance among the instances, while some could be solved within a second many instances could not be solved to optimality within the three hour time limit. The remaining optimality gap was often not even close to zero but up to 16.5% for ASRP-ARC and up to about 21% for the time-free method. Since the number of indexed copies of airports in the computational study of ASRP-ARC were not guaranteed to allow for all optimal routes to be feasible, the given dual bounds (and therefore the gaps) are not necessary valid for the ASRP. From the average objective value and computation times we however can conclude that the ASRP-ARC allows for solutions of similar quality (even slightly better) to be found in less time on average. It is also noteworthy that while the ASRP-ARC

instance name	b&c dual bd.	obj. val.	gap	b&c time
BUF-AIV	12614	12614	0.00 %	133
BUF-ANT	16466	18519	11.09 %	10800
BUF-BEE	15177	17633	13.93 %	10800
BUF-BOK	12917	12917	0.00 %	482
BUF-EGL	18450	22113	16.56 %	10800
BUF-GNU	14890	17350	14.18 %	10800
BUF-JKL	17756	20774	14.53 %	10800
BUF-LEO	21211	24938	14.94 %	10800
BUF-NAS	15549	15549	0.00 %	266
BUF-OWL	15766	16898	6.70 %	10800
BUF-ZEB	14688	14688	0.00 %	8920
EGL-BEE	16322	16322	0.00 %	2
EGL-GNU	19225	19225	0.00 %	1147
EGL-LEO	19388	19388	0.00 %	2587
GNU-BEE	10906	11165	2.32 %	10800
GNU-JKL	10557	10557	0.00 %	786
GNU-LEO	17863	17863	0.00 %	1178
LEO-AIV	13615	13615	0.00 %	1
LEO-ANT	16678	16678	0.00 %	16
LEO-BEE	17814	18900	5.74 %	10800
LEO-BOK	15372	15372	0.00 %	40
LEO-JKL	17551	17551	0.00 %	320
LEO-NAS	18192	18192	0.00 %	23
LEO-OWL	15827	15827	0.00 %	5
Average	16033	16860	4.17 %	4713

Table 3: Computational results for ASRP-ARC.

needed less computation time on average, the time-free method clearly outperforms it on some of the instances, such as BUF-AIV and BUF-BOK.

2.2 A Time Flow Formulation with Path-Indexed Variables for the Requests

In model ASRP-ARC the requests are treated as commodities with additional constraints (2d) and (2e). If the routes of all airplanes were fixed the decisions for the tours of the request are reduced to the choice of an airplane and a pickup location. So, in principle, as an alternative to the commodity flow it is possible to have one binary decision variable for each feasible path of the requests. This kind of approach is rarely used for a simple reason: the number of feasible paths in a graph is usually much larger than the number of arcs and therefore much more variables would be required. In our case, however, this alternative approach seems to be promising since the number of intermediate stops and the length of the tours are bounded. Therefore, if M_r or $\overline{d}_r$ are small for a request $r \in \mathcal{R}$, path indexed decision variables might be worth considering. In the benchmark instances which are derived from real-world data no path may include more than 2 intermediate stops which implies that the number of feasible paths for a request is of order $|\mathcal{V}^{\text{ext}}|^2$, which is of the same order as the number of arc based variables.

We denote the set of all paths on $(\mathcal{V}^{\text{ext}}, \mathcal{A}^{\text{ext}})$ with first vertex D_r and final vertex A_r by $\mathbb{P}_r$ for all $r \in \mathcal{R}$ and with a slight abuse of notation (arcs as elements of a path) define

$$\forall\, r \in \mathcal{R}: \ \mathbb{Q}_r := \left\{ p \in \mathbb{P}_r \;\middle|\; \sum_{(i,j)\in p} 1 \leq M_r + 1, \sum_{(i,j)\in p} d_{i,j} \leq \overline{d}_r \right\}, \tag{3}$$

the sets of the feasible paths. In order to derive a MILP based on decision variables indexed by these paths we replace the variables (1d), (1e), and (1f) with variables

$$\forall\, r \in \mathcal{R}, p \in \mathcal{P}, q \in \mathbb{Q}_r: \ x^q_{r,p} \in \{0,1\}. \tag{1d'}$$

instance name	b&c dual bd.	obj. val.	gap	b&c time
BUF-AIV	12614	12614	0.00 %	1
BUF-ANT	16353	20723	21.09 %	10800
BUF-BEE	16197	17633	8.14 %	8937*
BUF-BOK	12917	12917	0.00 %	2
BUF-EGL	19362	22113	12.44 %	10800
BUF-GNU	16762	17350	3.39 %	7525*
BUF-JKL	19155	20774	7.79 %	10800
BUF-LEO	22488	24938	9.82 %	10800
BUF-NAS	15549	15549	0.00 %	38
BUF-OWL	16797	16797	0.00 %	829
BUF-ZEB	14688	14688	0.00 %	162
EGL-BEE	16428	16653	1.34 %	10800
EGL-GNU	16862	20238	16.68 %	10800
EGL-LEO	19287	19388	0.52 %	10800
GNU-BEE	10257	11311	9.31 %	10800
GNU-JKL	10275	11098	7.42 %	10800
GNU-LEO	17863	17863	0.00 %	815
LEO-AIV	13615	13615	0.00 %	8
LEO-ANT	16678	16678	0.00 %	48
LEO-BEE	17306	18870	8.43 %	10800
LEO-BOK	15372	15372	0.00 %	4468
LEO-JKL	17551	17551	0.00 %	591
LEO-NAS	18191	18192	0.00 %	136
LEO-OWL	15827	15827	0.00 %	5
Average	16183	17032	4.43 %	5482

*16GB Memorylimit reached

Table 4: Computational results for the time-free approach.

The constraints then have to be adjusted to these variables. First of all, the constraints (2b),(2d), and (2e) are already used in the definition of the request variables and are therefore omitted without a replacement.

Instead of constraint (2c) we impose

$$\forall\, r \in \mathcal{R}: \sum_{p \in \mathcal{P}} \sum_{q \in \mathbb{Q}_r} x^q_{r,p} = 1 \tag{2c'}$$

which ensures that for each request exactly one feasible path is chosen. The constraints for the weight and number of used seats on the arcs have to be adjusted. Hence, the seat capacity constraint (2f) is replaced by

$$\forall\, (i,j) \in \mathcal{A}^{\text{ext}}, p \in \mathcal{P}: \sum_{r \in \mathcal{R}} \sum_{\substack{q \in \mathbb{Q}_r \\ (i,j) \in q}} s_r \cdot x^q_{r,p} \leq \overline{s}_p \cdot y^p_{i,j} \tag{2f'}$$

and (2l) is replaced by

$$\forall\, (i,j) \in \mathcal{A}^{\text{ext}}, p \in \mathcal{P}: w^p_{i,j} = \sum_{r \in \mathcal{R}} \sum_{\substack{q \in \mathbb{Q}_r \\ (i,j) \in q}} w_r \cdot x^q_{r,p} + f^p_{i,j}. \tag{2l'}$$

Similarly (2p) and (2q), the constraints for the time-windows, are adjusted to the new variables and replaced by

$$\forall\, p \in \mathcal{P}, (i,j) \in \mathcal{A}^{\text{ext}}, r \in \mathcal{R}:\ t^p_{i,j} \leq \tau^r_{\text{arr}} + M - 2M \sum_{\substack{q \in \mathbb{Q}_r \\ (i,j) \in q}} x^q_{r,p} + M y^p_{i,j}, \tag{2p'}$$

$$\forall\, p \in \mathcal{P}, (i,j) \in \mathcal{A}^{\text{ext}}, r \in \mathcal{R}:\ t^p_{i,j} \geq \tau^r_{\text{dep}} - M + M \sum_{\substack{q \in \mathbb{Q}_r \\ (i,j) \in q}} x^q_{r,p}. \tag{2q'}$$

2.2.1 The MILP

Using the adapted constraints and variables we can define the second new MILP formulation for the ASRP, which we call the Time-Flow Formulation with Extensive Request Path Variables ASRP-PATH:

$$\begin{aligned} \text{minimize} \quad & \sum_{(i,j)\in\mathcal{A}^{\text{ext}}} \sum_{p\in\mathcal{P}} c^p_{i,j} \cdot y^p_{i,j} \\ \text{subject to} \quad & (1a) - (1c), (1d'), (1g) - (1m), \\ & (2a), (2g) - (2k), (2m) - (2o), (2r), (2s), \\ & (2c'), (2f'), (2l'), (2p'), (2q'). \end{aligned} \tag{ASRP-PATH}$$

Since the variables for the airplane paths did not change, the same objective function as in ASRP-ARC is used.

2.2.2 Computational Results

Analogously to the computational study on ASRP-ARC we performed a test of ASRP-PATH for the benchmark instances. The results can be found in Table 5. The results show that on average

instance name	b&c dual bd.	obj. val.	gap	b&c time
BUF-AIV	12614	12614	0.00 %	247
BUF-ANT	16111	19194	16.06 %	10800
BUF-BEE	14492	17520	17.29 %	10800
BUF-BOK	12917	12917	0.00 %	1395
BUF-EGL	18473	22113	16.46 %	10800
BUF-GNU	15202	17350	12.38 %	10800
BUF-JKL	18099	20774	12.87 %	10800
BUF-LEO	22122	24938	11.29 %	10800
BUF-NAS	15549	15549	0.00 %	284
BUF-OWL	15770	16898	6.68 %	10800
BUF-ZEB	14010	15088	7.14 %	10800
EGL-BEE	16322	16322	0.00 %	1
EGL-GNU	19225	19225	0.00 %	2119
EGL-LEO	19388	19388	0.00 %	4675
GNU-BEE	11165	11165	0.00 %	2081
GNU-JKL	10557	10557	0.00 %	585
GNU-LEO	17863	17863	0.00 %	2891
LEO-AIV	13615	13615	0.00 %	1
LEO-ANT	16678	16678	0.00 %	13
LEO-BEE	18900	18900	0.00 %	7327
LEO-BOK	15372	15372	0.00 %	39
LEO-JKL	17551	17551	0.00 %	371
LEO-NAS	18192	18192	0.00 %	6
LEO-OWL	15827	15827	0.00 %	5
Average	16084	16900	4.17 %	4518

Table 5: Computational results for ASRP-PATH.

the MILPs derived from ASRP-PATH were solved faster, although not by a large margin. When comparing instance by instance, each formulation seems to be better suited for some.

3 Cutting Planes

Although several benchmark cases could be solved by directly solving the two MILP formulations with a MILP solver, other instances still remain unsolved and it is not clear if they indeed outperform the time-free approach from [9]. A method that is known to have the potential to reduce computation time is to find valid inequalities (cutting planes), that have the potential to tighten the linear relaxations and speed up the solution process (Fügenschuh et al. [8], Marchand et al. [13], Cornuejols [4]). Therefore, in this section we describe two types of additional constraints that can be added to the MILP formulations, such that maybe some but not all optimal solutions are cut off.

Copy Priority Cuts. While the indexing of the airports suggests that there is an ordering, no such ordering has been imposed on the variables explicitly. Hence for any feasible solution there are many solutions only differing in the index of the visited airport copies. Introducing additional constraints that make some of these (often called symmetric) solutions infeasible, has often been a very helpful tool for improving the performance of branch-and-bound solvers (see for example Margot [14], Coelho and Laporte [3]). In order to break the symmetries inherent to the presented formulations the constraints

$$\forall p \in \mathcal{P}, j \in \mathcal{V}^{\text{ext}}, \text{ s.t. } j_2 \neq n_j : \sum_{i:(i,j)\in\mathcal{A}^{\text{ext}}} y^p_{i,j} \geq \sum_{i:(i,j)\in\mathcal{A}^{\text{ext}}} y^p_{i,(j_1,j_2+1)} \tag{4}$$

$$\forall p \in \mathcal{P}, j \in \mathcal{V}^{\text{ext}}, \text{ s.t. } j_2 \neq n_j : \sum_{k:(j,k)\in\mathcal{A}^{\text{ext}}} y^p_{j,k} \geq \sum_{k:(j,k)\in\mathcal{A}^{\text{ext}}} y^p_{(j_1,j_2+1),k} \tag{5}$$

can be added, which we call Copy Priority Cuts. They explicitly impose that an arc to a copy of an airport can only be used if there is already an arc going to the copy with index one less. Inductively this implies that all copies with a lower index have to be visited in order to include a copy of an airport that does not have the index 1, so these have priority for visits.

Useless Detour Cuts. So far we link request paths to airplane paths only by the seat capacity constraints. It is easy to see that each arc included in an optimal path has to either serve a request or contain a refueling airport. Otherwise the path could be shortened by the triangle inequality. Consequently, for airports that do not provide fuel for airplane p we can impose cuts

$$\forall (i,j) \in \mathcal{A}^{\text{ext}}, p \in \mathcal{P} : y^p_{i,j} \leq \sum_{r\in\mathcal{R}} x^{r,p}_{i,j}, \tag{6}$$

$$\forall (i,j) \in \mathcal{A}^{\text{ext}}, p \in \mathcal{P} : y^p_{i,j} \leq \sum_{r\in\mathcal{R}} \sum_{\substack{q\in\mathbb{Q}_r \\ (i,j)\in q}} x^q_{r,p}, \tag{6'}$$

without cutting off any optimal solutions. Since they literally prevent detours without a purpose we call them Useless Detour Cuts.

Time Order Cuts. The Copy Priority Cuts (4), (5) do not require that copies with lower index have to be visited first in the solution, as none of the time variables are restricted. With $\omega_{,j,p}$ as the minimal time for airplane $p \in \mathcal{P}$ to leave from and return to airport $j \in \mathcal{V}$ the constraints

$$\forall p \in \mathcal{P}, j \in \mathcal{V}^{\text{ext}}, \text{ s.t. } j_2 \neq 1 : \ \omega_{j,p,+} \sum_{i:(i,j)\in\mathcal{A}^{\text{ext}}} t^p_{i,(j_1,j_2-1)} \leq (1 - y^p_{i,j})\tau^p_{\text{arr}} + t^p_{i,j} \tag{7}$$

impose an explicit timed ordering of the visited copies.

Time-Window Conflict Cuts. These cuts can only be formulated for the path based formulation and are motivated by the following observation: Two feasible paths for two requests not sharing an arc can only be assigned to the same airplane, if they can be flown in succession without violating the time-windows of the requests. For two requests $r_1, r_2 \in \mathcal{R}$ and two paths $q_1 \in \mathbb{Q}_{r_1}, q_2 \in \mathbb{Q}_{r_2}$ that do not share an arc this can be expressed as conditions

$$\begin{gathered} \tau^{r_1}_{\text{dep}} + \delta^p_{r_1,q_1} + \delta^p_{A_{r_1},D_{r_2}} + \delta^p_{r_2,q_2} > \tau^{r_2}_{\text{arr}} \\ \text{and} \\ \tau^{r_2}_{\text{dep}} + \delta^p_{r_2,q_2} + \delta^p_{A_{r_2},D_{r_1}} + \delta^p_{r_1,q_2} > \tau^{r_1}_{\text{arr}}. \end{gathered} \tag{8}$$

Therefore

$$\forall p \in \mathcal{P}, \forall r_1, r_2, \in \mathcal{R}, q_1 \in \mathbb{Q}_{r_1}, q_2 \in \mathbb{Q}_{r_2}, \text{ s.t. (8) holds} : \ x^{q_1}_{r_1,p} + x^{q_2}_{r_2,p} \leq 1 \tag{9}$$

is a valid inequality for the ASRP-PATH, which we call Time-Window Conflict Cuts.

3.1 Computational Results

The computational study of the previous section was carried out on the same instances, with the cutting planes added to the models. The results for ASRP-ARC with the cuts (4), (5), (6), (7) are presented in Table 6 and the ones for ASRP-PATH with the same cuts and additionally (9) are presented in Table 7.

instance name	b&c dual bd.	obj. val.	gap	b&c time
BUF-AIV	12614	12614	0.00 %	76
BUF-ANT	17607	20049	12.18 %	10800
BUF-BEE	14221	17633	19.35 %	10800
BUF-BOK	12917	12917	0.00 %	320
BUF-EGL	18569	22113	16.03 %	10800
BUF-GNU	14877	17350	14.25 %	10800
BUF-JKL	18350	20774	11.67 %	10800
BUF-LEO	21477	24938	13.88 %	10800
BUF-NAS	15549	15549	0.00 %	236
BUF-OWL	16797	16797	0.00 %	4168
BUF-ZEB	14688	14688	0.00 %	7264
EGL-BEE	16322	16322	0.00 %	1
EGL-GNU	19225	19225	0.00 %	646
EGL-LEO	19388	19388	0.00 %	269
GNU-BEE	11165	11165	0.00 %	806
GNU-JKL	11098	11098	0.00 %	213
GNU-LEO	17863	17863	0.00 %	1633
LEO-AIV	13615	13615	0.00 %	1
LEO-ANT	16678	16678	0.00 %	7
LEO-BEE	17219	18900	8.89 %	10800
LEO-BOK	15372	15372	0.00 %	8
LEO-JKL	17551	17551	0.00 %	48
LEO-NAS	18192	18192	0.00 %	5
LEO-OWL	15827	15827	0.00 %	12
Average	16134	16944	3.52 %	3805

Table 6: Computational results for ASRP-ARC with cuts (4), (5), (6), (7).

In both cases only 7 instances remain with a non-zero gap. Furthermore the average computation time and the average gap were reduced as well. The performance of the two formulations differs from instance to instance, so that we cannot conclude that one clearly outperforms the other. These results further indicate, that these approaches only work well for some instances, as several instances remain unsolved for any of the two formulations and also the time-free approach of [9].

When we looked at the best solution found and the number of copies in the model for each airport (which was not even an exact estimate), there was a significant over-estimation of the number of required copies. The derivation of good estimations, that are correct upper bounds however is difficult, because there might be feasible solutions with very similar objective value, that have a highly differing number of visits to some airports (for example due to refueling). If an a priori estimate is difficult, similar to the time-free approach, information and routes obtained from relaxations could be useful in order to find proven optimal solutions. In the time-free approach the time index was deleted in an aggregated formulation, in the new formulation the variables for the copies of the same airports are candidates for an aggregation approach. In the following we show, how the ideas of the time-free approach can be used effectively for the two new formulations in order to find more compact models, which have solutions that are proven optimal for the ASRP.

instance name	b&c dual bd.	obj. val.	gap	b&c time
BUF-AIV	12614	12614	0.00 %	73
BUF-ANT	17624	18519	4.83 %	10800
BUF-BEE	16259	17633	7.79 %	10800
BUF-BOK	12917	12917	0.00 %	597
BUF-EGL	19429	21900	11.28 %	10800
BUF-GNU	14481	17350	16.53 %	10800
BUF-JKL	18643	20774	10.25 %	10800
BUF-LEO	22056	24938	11.55 %	10800
BUF-NAS	15549	15549	0.00 %	209
BUF-OWL	16797	16797	0.00 %	2409
BUF-ZEB	14688	14688	0.00 %	9530
EGL-BEE	16322	16322	0.00 %	1
EGL-GNU	19225	19225	0.00 %	634
EGL-LEO	19388	19388	0.00 %	703
GNU-BEE	11165	11165	0.00 %	248
GNU-JKL	11098	11098	0.00 %	80
GNU-LEO	17863	17863	0.00 %	4344
LEO-AIV	13615	13615	0.00 %	1
LEO-ANT	16678	16678	0.00 %	5
LEO-BEE	18386	18900	2.71 %	10800
LEO-BOK	15372	15372	0.00 %	6
LEO-JKL	17551	17551	0.00 %	188
LEO-NAS	18192	18192	0.00 %	1
LEO-OWL	15827	15827	0.00 %	2
Average	16324	16872	2.71 %	3943

Table 7: Computational results for ASRP-ARC with cuts (4), (5), (6), (7), (9).

4 Relaxations of the MILPs Underestimating the Number of Necessary Airport Copies

Although our model does not fit the traditional time-bucket formulation, it still allows for similar relaxation techniques to be used. Instead of introducing nodes representing intervals for including multiple of the discrete points of time, it is possible to reduce the number of copies of airports in $\mathcal{V}^{\text{ext}}$. In order to guarantee that it is in fact a relaxation some slight changes have to be made to the models ASRP-ARC and ASRP-PATH. First of all, since an arc may be traversed multiple times in the optimal solution some y-variables have to be integer instead of binary. Furthermore constraints (2r) and (2s), which enforce that each copy can only be visited once, are in conflict with the idea of using less copies than necessary. It however suffices to exempt one copy from this constraint, resulting in the new conditions

$$\forall\, j \in \mathcal{V}^{\text{ext}}, \text{ s.t. } j_2 \neq n_j, p \in \mathcal{P}: \quad \sum_{i:(i,j)\in\mathcal{A}^{\text{ext}}} y^p_{i,j} \leq 1, \tag{10}$$

$$\forall\, j \in \mathcal{V}^{\text{ext}}, \text{ s.t. } j_2 \neq n_j, p \in \mathcal{P}: \quad \sum_{i:(j,k)\in\mathcal{A}^{\text{ext}}} y^p_{j,k} \leq 1. \tag{11}$$

Now if an arc is traversed multiple times by the same airplane, the corresponding time and fuel variables have to store the summed values of each trip. As a consequence not only their upper bounds have to be omitted but at the same time (2q) has to be changed. We replace it with

$$\forall\, p \in \mathcal{P}, (i,j) \in \mathcal{A}^{\text{ext}}, r \in \mathcal{R}: \; t^p_{i,j} \geq \tau^r_{\text{dep}} - 2M(1 - x^{r,p}_{i,j}) - M(y^p_{i,j} - 1), \tag{12}$$

$$\forall\, p \in \mathcal{P}, (i,j) \in \mathcal{A}^{\text{ext}}, r \in \mathcal{R}:$$

$$t^p_{i,j} \geq \tau^r_{\text{dep}} - 2M\Big(1 - \sum_{\substack{q\in\mathbb{Q}_r \\ (i,j)\in q}} x^q_{r,p}\Big) - M(y^p_{i,j} - 1). \tag{12'}$$

Where the second term softens the constraint if the $y_{i,j}^p$ is greater than 1. With these changes we can define MILPs that are relaxations of ASRP-ARC and ASRP-PATH for any combination of values for n_j with $j \in \mathcal{V}$. From now on we refer to these relaxations as intermediate time-flow formulations for the ASRP and to formulations with correct bounds n_j as the fully expanded formulation. If the optimal solution of any such relaxation is feasible to ASRP-ARC (resp. ASRP-PATH) then it is also optimal for ASRP-ARC (resp. ASRP-PATH). Note that for a sufficient number of copies feasibility is guaranteed, if the instance has a solution. The simplest intermediate time-flow formulation is obtained by setting $n_j = 1$ for all $j \in \mathcal{V}$. It is, however, also the furthest from guaranteeing feasibility. The next step is to find intermediate time-flow formulations that are relatively easy to solve and presumably are feasible. In order to do so we developed Algorithm 1 which has the goal to construct intermediate graphs efficiently based on previously checked relaxations with too few copies of some airports. Since the time-free relaxation of Fügenschuh et al. [9] already had a

Algorithm 1: Iterative Construction of Relaxations

```
   Data: a full parameter description of the airplane routing problem
1  for $i \in \mathcal{V}$ do
2      $\tilde{n}_i \leftarrow 1$;
3  CurrentReduction ← Relaxation of the Time-Flow MILP with $n_i = \tilde{n}_i$ for all $i \in \mathcal{V}$;
4  Solve CurrentReduction;
5  CurrentSolution ← Optimal solution of CurrentReduction;
6  while CurrentSolution is infeasible for Time-Flow MILP do
7      for $i \in \mathcal{V}$ do
8          stops ← the maximum of the total number of arrivals and departures from $i$ in
             CurrentSolution;
9          if $\tilde{n}_i <$ stops then
10             $\tilde{n}_i \leftarrow$ the total number of visits to $i$;
11     CurrentReduction ← Relaxation of the Time-Flow MILP with $n_i = \tilde{n}_i$ for all $i \in \mathcal{V}$;
12     Solve CurrentReduction;
13     CurrentSolution ← Optimal solution of CurrentReduction;
14 OptimalSolution ← CurrentSolution
```

relatively small gap, Algorithm 1 is initialized with a similar relaxation in which each airport has only one indexed copy (line 2). If the obtained routing and schedule is already feasible for a fully expanded formulation, the optimal solution has been found. Otherwise the loop starting in line 6 is entered. Here, only the number of copies of those airports is increased which are visited more often than its current number of copies (lines 9, 10). If in the optimal solution of an intermediate formulation each copy is visited no more than once, then no cycles are included on the routes and feasibility to fully expanded formulations is ensured. Consequently the number of copies of at least one airport is increased in each iteration and the loop terminates in finitely many iterations. We note that for the computational studies we improved Algorithm 1 in such a way that all other infeasible solutions found during the branch-and-bound process with better objective than the currently known optimum are abrogated by introducing additional copies. This led to less loop iterations and therefore a reduced number of MILPs that had to be solved.

5 Computational Results

In order to test the effectiveness of Algorithm 1 for the ASRP-ARC and ASRP-PATH we performed a computational study on the same benchmark instances as before. Its results are then be compared to the ones previously obtained for the models where the number of copies was fixed. The results for the arc based model can be found in Table 8 and the results for the path based model in Table 9. We tested 4 (5 for the ASRP-PATH) different configurations of cutting planes, which all included (4) and (5) as they necessary for applying Algorithm 1, namely: No additional cutting planes, each cutting plane individually, and all cutting planes at the same time. Instead of giving

instance	cutting planes	b&c dual bd.	obj. val.	gap	b&c time	iterations
BUF-AIV	(6),(7)	12614	12614	0.00 %	1	1
BUF-ANT	(6)	18225	18225	0.00 %	80	3
BUF-BEE	(6),(7)	17120	17120	0.00 %	171	2
BUF-BOK	(6),(7)	12917	12917	0.00 %	2	1
BUF-EGL	(6),(7)	21575	21575	0.00 %	865	2
BUF-GNU	(6),(7)	17350	17350	0.00 %	1132	3
BUF-JKL	(6),(7)	20374	20374	0.00 %	6761	4
BUF-LEO	(6)	22888	22888	0.00 %	1112	2
BUF-NAS	(6),(7)	15549	15549	0.00 %	3	2
BUF-OWL	(6),(7)	16797	16797	0.00 %	28	2
BUF-ZEB	(6)	14688	14688	0.00 %	28	2
EGL-BEE	(6),(7)	16322	16322	0.00 %	2	2
EGL-GNU	(6),(7)	19225	19225	0.00 %	6	4
EGL-LEO	(6),(7)	19388	19388	0.00 %	4	2
GNU-BEE	(6),(7)	11165	11165	0.00 %	33	3
GNU-JKL	(6)	11098	11098	0.00 %	14	2
GNU-LEO	(6)	17863	17863	0.00 %	12	2
LEO-AIV	(6),(7)	13615	13615	0.00 %	0	1
LEO-ANT	(6),(7)	16678	16678	0.00 %	1	1
LEO-BEE	(6)	18900	18900	0.00 %	599	3
LEO-BOK	(6),(7)	15372	15372	0.00 %	4	3
LEO-JKL	(6),(7)	17551	17551	0.00 %	3	2
LEO-NAS	(6),(7)	18192	18192	0.00 %	2	3
LEO-OWL	(6),(7)	15827	15827	0.00 %	1	1
Average	(6),(7)	16722	16722	0.00 %	609	2.3
Average	(6)	16710	16729	0.09 %	825	2.4
Average	(7)	16666	16899	1.07 %	1987	2.5
Average	None	16628	16737	0.54 %	2558	2.7

Table 8: Computational results for ASRP-ARC with Alg. 1.

a long list of all results for all of the tested configurations, only the best result per instance and the overall averages are reported. In both cases the results vary between the instances, but overall the inclusion of all cutting planes is best for most instances. Furthermore it is the best on average in both cases followed by only using the Useless Detour Cuts (6). Although they still perform much better than the models with a fixed number of copies the configurations with only (6), (9), (7), or no cutting planes are worse by a rather large margin. This indicates that the cutting plane (6) is the most important for reducing computation times. We observed that without this cut in some of the instances it is possible for solutions to exist that include detours only to reach a copy of higher index. This then allows for the unwanted variable assignments, which were previously depicted in Figure 1. With the addition of (6) these detours are infeasible if no request is served while visiting a copy of an airport, which therefore potentially reduces the number of copies that have to be introduced. When comparing the results for the two different new formulations it is also visible that ASRP-PATH is indeed an improvement over ASRP-ARC. We mainly attribute this to the short maximal length of the requests routes and the small number of copies necessary to find a proven optimal solution. For instances in which this is not the case we expect ASRP-ARC to be superior as the number of variables in ASRP-PATH is very sensitive with respect to those two characteristics.

6 Conclusions

In this paper we presented two new MILP formulations for the ASRP proposed by Fügenschuh et al. [9]. Instead of using arrival times as additional indices these formulations incorporate the aspect of time in form of variables. In computational experiments we found that we had to heuristically bound the number of visits to the airports for them to be applicable. With a state-of-

instance name	cutting planes	b&c dual bd.	obj. val.	gap	b&c time	iter.
BUF-AIV	(6),(7),(9)	12614	12614	0.00 %	1	1
BUF-ANT	(6)	18225	18225	0.00 %	134	3
BUF-BEE	(6),(7),(9)	17120	17120	0.00 %	298	2
BUF-BOK	(6),(7),(9)	12917	12917	0.00 %	2	1
BUF-EGL	(6),(7),(9)	21575	21575	0.00 %	808	2
BUF-GNU	(6),(7),(9)	17350	17350	0.00 %	455	4
BUF-JKL	(6),(7),(9)	20374	20374	0.00 %	3153	4
BUF-LEO	None	22888	22888	0.00 %	857	4
BUF-NAS	(6),(7),(9)	15549	15549	0.00 %	2	2
BUF-OWL	(6),(7),(9)	16797	16797	0.00 %	14	2
BUF-ZEB	(6),(7),(9)	14688	14688	0.00 %	20	2
EGL-BEE	(6),(7),(9)	16322	16322	0.00 %	1	2
EGL-GNU	(6),(7),(9)	19225	19225	0.00 %	3	3
EGL-LEO	(6)	19388	19388	0.00 %	3	2
GNU-BEE	(6)	11165	11165	0.00 %	33	3
GNU-JKL	(6),(7),(9)	11098	11098	0.00 %	7	2
GNU-LEO	(9)	17863	17863	0.00 %	8	2
LEO-AIV	(6),(7),(9)	13615	13615	0.00 %	0	1
LEO-ANT	(6),(7),(9)	16678	16678	0.00 %	0	1
LEO-BEE	(9)	18900	18900	0.00 %	238	4
LEO-BOK	(9)	15372	15372	0.00 %	3	3
LEO-JKL	(9)	17551	17551	0.00 %	4	2
LEO-NAS	(6),(7),(9)	18192	18192	0.00 %	1	3
LEO-OWL	(6),(7),(9)	15827	15827	0.00 %	0	1
Average	(6),(7),(9)	16722	16722	0.00 %	344	2.2
Average	(6)	16710	16718	0.04 %	809	2.5
Average	(7)	16673	16742	0.34 %	1572	2.5
Average	(9)	16651	16724	0.36 %	2036	2.5
Average	None	16672	16722	0.25 %	2012	2.8

Table 9: Computational results for ASRP-PATH with Alg. 1.

the-art MILP solver we were then able to find feasible solutions for the benchmark instances with objective values comparable to the best known upper bounds from the literature. However, due to the use bounded number of visits, no conclusions for the lower bounds can be made. In order to overcome this downside we showed that relaxations which can be interpreted as partially expanded formulations can be used. With the use of Algorithm 1 we were able to effectively construct intermediate graph expansions and solve all benchmark instances to proven optimality. In addition to that we managed to do this while reducing the average computation time by over 90 %. But, further improvements might be possible, since so far only information about incumbent solutions of previous iterations is used in Algorithm 1. All additional information obtained during the branch-and-bound step (such as infeasible branches) is neglected. We observed that a large percentage of the overall computation time is spent until the lower bound reaches the objective value of the previous iteration. A branch-and-bound algorithm that not only cuts off solutions based on the LP-relaxation, but also by using information from branch-and-bound-trees of previous iterations might reduce this time significantly. Finally we note that the proposed methods are not restricted to the ASRP and it seems to be worth investigating whether such a reduction in computation time can be achieved for problems of similar structure. Especially routing problems in which locations can be visited multiple times are candidates for our method, since in this case many conventional methods cannot be applied.

Acknowledgement

This project was funded by the German Research Association (DFG), grant number FU 860/1-1.

References

[1] M. L. Balinski and R. E. Quandt. On an integer program for a delivery problem. *Operations Research*, 12(2):300–304, 1964.

[2] N. Boland, M. Hewitt, L. Marshall, and M. Savelsbergh. The continuous-time service network design problem. *Operations Research*, 65(5):1303–1321, 2017.

[3] L. C. Coelho and G. Laporte. The exact solution of several classes of inventory-routing problems. *Computers & Operations Research*, 40(2):558–565, 2013.

[4] G. Cornuéjols. Valid inequalities for mixed integer linear programs. *Mathematical Programming*, 112(1):3–44, 2008.

[5] G. Dantzig, D. R. Fulkerson, and S. Johnson. Solution of a large-scale traveling-salesman problem. *Journal of the Operations Research Society of America*, 2(4):393–410, 1954.

[6] S. Dash, O. Günlük, A. Lodi, and A. Tramontani. A time bucket formulation for the traveling salesman problem with time windows. *INFORMS Journal on Computing*, 24(1):132–147, 2012.

[7] L. R. Ford and D. R. Fulkerson. *Flows in networks.* Princeton University Press, 1962.

[8] A. Fügenschuh and A. Martin. Computational integer programming and cutting planes. In K. Aardal, G. Nemhauser, and R. Weismantel, editors, *Handbooks in Operations Research and Management Science*, volume 12: Handbook on Discrete Optimization, pages 69–121. Elsevier, Amsterdam, 2005.

[9] A. Fügenschuh, G. Nemhauser, and Y. Zeng. Scheduling and routing of fly-in safari planes using a flow-over-flow model. In G. Reinelt and M. Jünger, editors, *Facets of Combinatorial Optimization: Festschrift for Martin Grötschel*, pages 419–447. Springer Berlin Heidelberg, 2013.

[10] B. Gavish and S. C. Graves. The traveling salesman problem and related problems. Technical Report OR 078-78, Operations Research Center, Massachusetts Institute of Technology, 1978.

[11] F. Gnegel and A. Fügenschuh. A time-flow approach for scheduling and routing of fly-in safari airplanes. In *Operations Research Proceedings 2018*, 2019. 6 pages. To appear.

[12] E. He, N. Boland, G. Nemhauser, and M. Savelsbergh. A dynamic discretization discovery algorithm for the minimum duration time-dependent shortest path problem. In *International Conference on the Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, pages 289–297. Springer, 2018.

[13] H. Marchand, A. Martin, R. Weismantel, and L. Wolsey. Cutting planes in integer and mixed integer programming. *Discrete Applied Mathematics*, 123(1-3):397–446, 2002.

[14] F. Margot. Symmetry in integer linear programming. In *50 Years of Integer Programming 1958-2008*, pages 647–686. Springer, 2010.

[15] M. Savelsbergh. Local search in routing problems with time windows. *Annals of Operations Research*, 4(1):285–305, 1985.

[16] M. Savelsbergh and M. Sol. The general pickup and delivery problem. *Transportation Science*, 29(1):17–29, 1995.

[17] M. M. Solomon and J. Desrosiers. Survey papertime window constrained routing and scheduling problems. *Transportation Science*, 22(1):1–13, 1988.

[18] P. Toth and D. Vigo. *Vehicle routing: problems, methods, and applications.* SIAM, 2014.

[19] D. M. Vu, M. Hewitt, N. Boland, and M. Savelsbergh. Solving time dependent traveling salesman problems with time windows. Technical Report 6640, Optimization Online, 2018.

[20] X. Wang and A. C. Regan. Local truckload pickup and delivery with hard time window constraints. *Transportation Research Part B: Methodological*, 36(2):97–112, 2002.

[21] X. Wang and A. C. Regan. On the convergence of a new time window discretization method for the traveling salesman problem with time window constraints. *Computers & Industrial Engineering*, 56(1):161–164, 2009.

IMPRESSUM

Brandenburgische Technische Universität Cottbus-Senftenberg
Fakultät 1 | MINT - Mathematik, Informatik, Physik, Elektro- und Informationstechnik
Institut für Mathematik
Platz der Deutschen Einheit 1
D-03046 Cottbus

Professur für Ingenieurmathematik und Numerik der Optimierung
Professor Dr. rer. nat. Armin Fügenschuh

E fuegenschuh@b-tu.de
T +49 (0)355 69 3127
F +49 (0)355 69 2307

Cottbus Mathematical Preprints (COMP), ISSN (Print) 2627-4019
Cottbus Mathematical Preprints (COMP), ISSN (Online) 2627-6100

www.b-tu.de/cottbus-mathematical-preprints
cottbus-mathematical-preprints@b-tu.de
doi.org/10.26127/btuopen-4806