A branch and cut algorithm for the time-dependent profitable tour problem with resource constraints

Gonzalo Lera-Romero * Juan José Miranda Bront †

gleraromero@dc.uba.ar, jmiranda@utdt.edu

Abstract

In this paper we study the time-dependent profitable tour problem with resource constraints (TDPTPRC), a generalization of the profitable tour problem (PTP) which includes variable travel times to account for road congestion. In this problem, the set of customers to be served is not given and must be determined based on the profit collected when visited, keeping a balance with the total travel time. In this paper, we propose a mixed integer linear programming (MILP) formulation that exploits the travel time function to reduce the size of a standard formulation from the literature. We derive four new families of valid inequalities and study the connections among them, as well as their associated separation problems. We develop a tailored Branch and Cut (BC) algorithm including these new families in addition to some well known valid inequalities from related problems. Computational results on three different problems, with alternative resources and objectives, show that the approach is flexible and effective. The algorithm achieves significant reductions in the computing times on benchmark instances from the related literature, and outperforms a recent method proposed for the time-dependent traveling salesman problem with time windows.

Keywords: Profitable Tour Problem; Time-Dependent Travel Times; Integer Programming; Branch and Cut

1 Introduction and literature review

Congestion in large cities and populated areas is one of the major challenges in urban logistics, becoming one of the critical issues in city planning and last-mile logistics due to its economic, social and environmental impact. Most of the research devoted to the Vehicle Routing Problem (VRP) considers that travel times between any two locations are fixed along the time horizon. However, in the last few years there has been a trend to enrich these models by incorporating more complex travel time functions to capture the effect of congestion, known as time-dependent VRPs (TDVRP, see, e.g., Gendreau et al. (2015)). Although different models have been proposed to capture the effect of congestion on the travel times, the one proposed in Ichoua et al. (2003) has become a standard within the TDVRP literature.

Exact approaches for multi-vehicle variants of the classical VRP generally resort to Integer Linear Programming (MILP) and decomposition techniques, resulting in *Branch and Price* (BP) and *Branch-cut and price* (BCP) algorithms. The time-dependent profitable tour problem with resource constraints (TDPTPRC) arises naturally as the pricing problem within a column generation algorithm for set partitioning-based approaches for multi-vehicle TDVRPs.

*Instituto de Investigación en Ciencias de la Computación (ICC) CONICET, Universidad de Buenos Aires, Buenos Aires, Argentina.

†Universidad Torcuato Di Tella, Buenos Aires, Argentina and CONICET.

In general, the presence of the so-called resources encode intra-route operational constraints and the profits stand for the dual variables associated with constraints associated to vertices in the master problem. In its time-independent version, the TDPTPRC can be formulated as an Elementary Shortest Path Problem with Resource Constraints (ESPPRC), which is known to be strongly $\mathcal{NP}$-hard. From an algorithmic standpoint, if one of the resources is known to be *restrictive* and (potentially) limits the length of the routes the pricing problem is tackled using labeling algorithms (see, e.g., Feillet et al. (2004)). This is the strategy followed by Dabia et al. (2013) for the TDVRP with time windows (TDVRPTW), Sun et al. (2018) for the TDPTP with vehicle capacity, time windows and pickup and deliveries, and Huang et al. (2017) for a variant of the TDVRPTW with path flexibility.

If no such a resource is available, one alternative is to consider a relaxation of the problem by allowing some type of infeasibility, which is later handled during the enumeration, at the expense of eventually deteriorating the quality of the lower bound of the LP relaxation. The current standard and the most effective approach is the *ng-route* relaxation proposed by Baldacci et al. (2011) for the VRP, and used in the context of the TDVRPs by Spliet et al. (2018) for a variant of the time window assignment problem with time-dependent travel times. In the context of the classical VRP, another stream of research considers tackling the ESPPRC using mixed integer programming, as proposed in Drexl & Irnich (2014), Drexl (2013), Jepsen et al. (2014) and Taccari (2016). We highlight the last two papers, where Branch and Cut (BC) algorithms are developed. One of the main motivation to explore this technique relies on the flexibility to incorporate different operational constraints as well as particular structures induced by branching and/or cuts included in the master problem. Based on this previous experience, we investigate a similar approach for the TDPTPRC in a preliminary version of this paper[1] Lera-Romero & Miranda-Bront (2018) where two different MILP formulations are compared experimentally.

Vehicle routing problems with profits, including the PTP and the TDPTPRC, have their own practical relevance in different contexts, as stated in (Toth & Vigo, 2014, Chapter 10). In this kind of problems, the set of customers to serve is not given in advance and becomes part of the decision to be taken in addition to the order in which they must be visited. Most of the research in this aspect has been devoted to the single vehicle case, including the *profitable tour problem* (PTP), the *prize collecting traveling salesman problem* (PCTSP) and the *orieentering problem* (OP), among others (see, e.g., Toth & Vigo (2014) for an updated literature review). Specifically for TDVRPs, the TDPTP with time windows and pickup and deliveries has been tackled with labeling algorithms in Sun et al. (2018), and different heuristics have been proposed for the TDOP with time windows in Verbeeck et al. (2014, 2017).

An additional motivation for studying TDVRPs with profits is that the findings and developments can be extended to more classical single-vehicle variants, such as the time-dependent traveling salesman problem (TDTSP) and the TDTSP with time windows (TDTSPTW), where all customers are required to be visited. These two variants received some attention recently, in particular regarding exact algorithms. The TDTSP is first considered in Cordeau et al. (2014), where the time-dependent travel time function is studied and the authors propose a lower bound obtained by solving an auxiliary time-independent TSP. The bound is shown to be tight under certain traffic conditions, and is later embedded in a BC algorithm. Recently, this approach is improved in Arigliano et al. (2018). An extension of the framework proposed in Cordeau et al. (2014) is considered for the TDTSPTW in Arigliano et al. (2015). From a computational standpoint, the effectiveness of the approach strongly depends on the quality of the initial lower bound, which the BC algorithm finds difficult to improve during the enumeration in the BC tree. A more classical BC algorithm is proposed in Montero et al. (2017), where

[1]Published in the EUROALIO 2018 proceedings.

an MILP formulation proposed in Sun et al. (2018) is studied and strengthened with classical cutting planes, preprocessing and several initial heuristics generalized from the TSP with time windows. The results outperform the ones reported in Arigliano et al. (2015). Recently, a dynamic discretization discovery (DDD) algorithm has been proposed for the TDTSPTW in Vu et al. (2018). To the best of our knowledge, this is so far the best algorithm in the literature for the TDTSPTW, that can be adapted to minimize not only the makespan, as the previous approaches in the literature, but the duration of the tour by eventually delaying the departure from the depot.

1.1 Our contributions

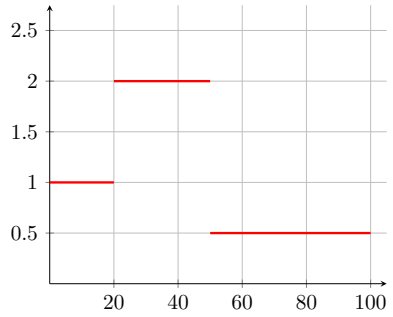
We develop a BC algorithm for the TDPTPRC building upon the preliminary results presented in Lera-Romero & Miranda-Bront (2018). The major contributions of the paper are the following. First, we propose a tailored formulation for the TDPTPRC which, depending on the structure of the time-dependent travel time function, reduces the overall number of variables and constraints. Furthermore, we prove that this adaptation does not impact in terms of the quality of the lower obtained by the LP formulation. Second, we present a polyhedral study for this model, focusing on exploiting the time-dependent structure for the problem. In particular, we propose four families of valid inequalities, including one which generalizes a well known family from a related problems. Third, we develop efficient separation routines for the new families of valid inequalities which, combined with some other valid inequalities from the literature, are incorporated into a tailored BC algorithm. Fourth, we conduct extensive computational experiments to evaluate the BC algorithm on benchmark instances from the literature in three different contexts. In particular, we consider a variant of the TDPTPRC on pricing instances for the TDVRP without time windows, where the capacity is not a limiting resource, obtained from the well-known Solomon instances. Additionally, we incorporate time windows and pickup and deliveries and compare with the results reported for a standard MILP formulation in Sun et al. (2018), which are significantly improved. Finally, we adapt the formulation and the BC algorithm to the TDTSPTW and evaluate the efficiency using the benchmarks proposed in Vu et al. (2018), showing that our approach is able to improve the results presented therein in three of the four scenarios considered.

The rest of the paper is organized as follows. In Section 2 we define the TDPTPRC and present most of the notation used throughout the paper. The MILP formulation, originally proposed in Lera-Romero & Miranda-Bront (2018), is reviewed and described in Section 3, including the theoretical results regarding the quality of the LP relaxation. The polyhedral study is presented in Section 4, and the details regarding the BC algorithm are discussed in Section 5. The numerical experiments are reported in Section 6 and, finally, we present the conclusions and future directions in Section 7.

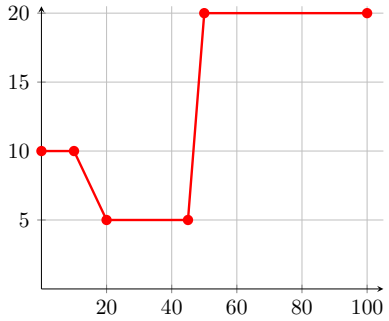
2 Problem definition

In this section we define the TDPTPRC in its general fashion. We refer the reader to Dabia et al. (2013) for details regarding its incorporation within a BP framework based on a set partitioning formulation for the TDVRP.

The time-dependent network is defined as follows. Let $D = (V, A)$ be a digraph, with $V = \{v_s, v_e, 1, 2, \cdots, n\}$ the set of vertices representing the customers, and A the set of arcs representing their connections. Two distinguished vertices, v_s and v_e, denote the starting and ending point of the path, respectively. Typically, these two vertices represent the depot and therefore no incoming arcs to v_s and outgoing arcs from v_e are considered.



(A) *Travel speed function for arc ij.*



(B) *Travel time function $\tau_{ij}(t)$.*

FIGURE 1. *Time-dependent travel time model ($L_{ij} = 10$).*

Generally, the resources used to model real life constraints are the vehicle capacity, time windows, service times, and pickup and deliveries among others. In the capacitated version, vehicles have a fixed capacity Q and each customer $i \in V$ has a demand q_i, measured in the same unit. The sum of the customers' demands in a single vehicle route must not exceed its capacity. Another constraints are the so-called time windows, which are defined by a time interval $[a_i, b_i]$ for each $i \in V$ indicating the time interval in which it is feasible to service customer i. In some contexts, the service time s_i for each customer $i \in V$ is considerable, and therefore we say the vehicle is at least s_i units of time at customer i servicing it. Note that under the presence of time windows, the service must begin within their limits. Finally, some problems involve fulfilling requests instead of customers, where each request r_k has a customer i where products must be picked from and a customer, generally noted as $n + i$, where products picked from i must be carried to.

In order to capture the time dependency, we adopt the travel time model proposed in Ichoua et al. (2003). Each arc $ij \in A$ has an associated travel distance L_{ij}. There is a time horizon $[0, T]$ in which operations take place, which is partitioned into H intervals $[T_h, T_{h+1}]$, $h = 0, \ldots, H-1$. The average travel speed for each arc ij during time interval $[T_h, T_{h+1}]$ is assumed to be known, and is denoted by v_{ijh}. This partition is usually referred as *speed profiles*. The model captures the variable travel times by combining the distance to be traveled for an arc with its different travel speeds defined for the arc depending on the starting time of the trip. Formally, we let $\tau_{ij}(t)$ denote the time-dependent travel time for arc $ij \in A$ when departing from customer i at time $t \in [0, T]$, which is computed using the Algorithm 1 from Ichoua et al. (2003). It can be easily shown that the resulting travel time function $\tau_{ij}(t)$ is piecewise linear. Figure 1 illustrates the relation between the travel speed and the travel time function. It is important to remark that the travel time function satisfies the *First-In First-Out* (FIFO) property.

The definition of the travel time function $\tau_{ij}(t)$ induces a new partition of the planning horizon for each arc $ij \in A$. The limits defining this partition can be computed by a simple algorithm and are usually referred as *travel time breakpoints*. For the sake of simplicity, we follow the notation introduced in Montero et al. (2017). Let $T^{ij} = \{T_1^{ij}, \ldots, T_M^{ij}\}$ be the new partition for arc $ij \in A$, where $T_m^{ij} = [w_{ij}^m, w_{ij}^{m+1}]$. We remark that the value of M may be different among arcs. Within each of these intervals, $\tau_{ij}(t)$ remains linear.

Different objective functions are considered for this problem. We consider a variant where each customer $i \in V$ has a profit p_i that is collected if customer i is visited, and therefore given a route $P = v_s, v_1, \ldots, v_k, v_e$ a compromise between its cost c_P and profit $\sum_{v \in P} p_v$ is optimized. The cost of a route can be its makespan, which is the earliest time it can be finished; its duration, being the minimum time required to traverse the route eventually delaying the departure and not necessarily starting at $t = 0$, or any other metric.

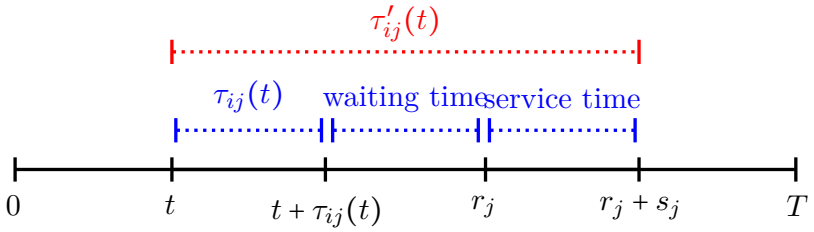


FIGURE 2. *Example of the transformation of travel time function* τ_{ij}.

The TDPTPRC involves finding an elementary path $P = (v_s, v_1, \ldots, v_k, v_e)$ minimizing $c_P - \sum_{v \in P} p_v$, starting and ending within the planning horizon, and fulfilling all resources constraints.

2.1 Waiting and service times

In order to simplify future analysis and models we proceed to show how we can transform instances so that all service times are zero and, even further, so that no waiting times are required in any optimal solution. The main idea behind the transformation is to consider waiting times and service times as part of the travel times. The benefit of having such a transformation is that most of the results from Section 4 are valid for different types of resources. We note that it is possible to achieve this as long as there is no difference between travel times, waiting times, and service times when evaluating the objective function. Figure 2 illustrates how we define the new travel time functions in order to do so.

First, we exhibit the definition of the so-called *ready time function* ρ which given a departing time $t \in [0, T]$ and a path $v_1 \ldots v_k$, it returns the time when the path is finished if departing from v_1 at time t. Next, we formalize this idea with the following definition.

$$\rho_{v_1 \ldots v_k}(t) = \begin{cases} a_{v_1} + s_{v_1} & \text{if } k = 1 \\ \rho_{v_2 \ldots v_k}\left(\max(a_2, t + \tau_{v_1 v_2}(t)) + s_2\right) & \text{otherwise.} \end{cases} \tag{1}$$

Now, given an instance of the TDPTPTW with travel time function τ_{ij} for each $ij \in A$, service time s_i and time window $[a_i, b_i]$ for each vertex $i \in V$, we define the transformed instance I' which is the same as the original instance except for:

(i) $s'_i = 0$ for each $i \in V$

(ii) $a'_i = a_i + s_i$, $b'_i = b_i + s_i$ for each $i \in V$

(iii) $\tau'_{ij}(t) = \max(a_j, t + \tau_{ij}(t)) - t + s_j$

Note that because of these definitions, the ready time function of this new instance can be simplified to

$$\rho_{v_1 \ldots v_k}(t) = \begin{cases} a'_{v_1} & \text{if } k = 1 \\ \rho_{v_2 \ldots v_k}(t + \tau'_{v_1 v_2}(t)) & \text{otherwise.} \end{cases} \tag{2}$$

Furthermore, it is easy to verify that feasibility is preserved considering the new time windows since the service time is added to both the travel times and time windows. Finally, an important observation is that in the transformed instance, optimal solutions do not require waiting times since the new travel time functions ensure that a customer is never reached before its release time. Note, however, that an equivalent optimal solution might exist where the vehicle waits at a customer before departing, but there must be another solution which departs as soon as possible which is optimal as well.

We assume in the remaining sections of this work that instances do not contain service times, and also that they do not require waiting times because if they do, we can transform them beforehand, and deal with the new instances instead. We use these properties in Section 4 to define new families of valid inequalities.

2.2 Notation

Finally, we introduce some further notation used throughout the paper. Let $\delta^-(j) = \{ij \in A\}$, $\delta^+(i) = \{ij \in A\}$, $\delta^+(S,T) = \{ij \in A : i \in S,\ j \in T\}$, $\bar{S} = V - S$. In addition, we denote with $\tau_{ij}(t) = \theta_{ij}^m t + \eta_{ij}^m$, for $t \in T_m^{ij}$, to the linear travel time function in time period $m \in T^{ij}$. Last, we refer to the combination of an arc ij and a time period $m \in T^{ij}$ as the (time-dependent) arc ijm, and we call $\hat{A} = \{ijm \in A \times \mathbb{N} : ij \in A, m \in T^{ij}\}$ to the set of all of them.

3 ILP formulations

3.1 ILP Model

We next present the formulation CTTBF derived from the one originally proposed in Sun et al. (2018), later studied in Montero et al. (2017), and finally considered in our preliminary work Lera-Romero & Miranda-Bront (2018). For the sake of completeness, we include in this Section the main definitions and the resulting formulation.

Let binary variables x_{ij}^m indicate that the path travels directly from i to j departing from i within travel time period $m \in T^{ij}$. Note that we can easily define traditional binary variables x_{ij} in terms of x_{ij}^m. Binary variable y_i takes value 1 iff vertex $i \in V$ is visited. In addition, for each vertex a non-negative variable t_i indicates the departure time from $i \in V$, if visited by the path and zero otherwise.

As observed in Lera-Romero & Miranda-Bront (2018), the travel time function for some of the travel time intervals $m \in T^{ij}$ becomes constant, and therefore $\theta_{ij}^m = 0$ holds. As a result, for each $ij \in A$, we define $\hat{T}^{ij} = \{m \in T^{ij} : \theta_{ij}^m = 0\}$ as the subset of intervals where the travel time function is constant. The value of t_i is decomposed to account for the departure time for each specific arc and travel time period, if any is selected. We define variables $t_{ij}^m = t_i$ if $x_{ij}^m = 1$, and $t_{ij}^m = 0$ otherwise for each $m \in T^{ij} \setminus \hat{T}^{ij}$ and $\hat{t}_{ij} = t_i$ if arc ij is traversed departing from i within one of the time intervals $m \in \hat{T}^{ij}$, and $\hat{t}_{ij} = 0$ otherwise. The resulting MILP formulation (CTTBF) is shown below

$$\min \quad t_{v_e} - t_{v_s} - \sum_{i \in V} p_i y_i \tag{3}$$

$$\text{s.t.:} \quad \sum_{m \in T^{ij}} x_{ij}^m = x_{ij} \qquad ij \in A \tag{4}$$

$$\sum_{ij \in \delta^-(j)} x_{ij} = y_j \qquad j \in V \setminus \{v_s\} \tag{5}$$

$$\sum_{v_s j \in \delta^+(v_s)} x_{v_s j} = y_{v_s} = y_{v_e} = 1 \tag{6}$$

$$\sum_{ik \in \delta^-(k)} x_{ik} - \sum_{kj \in \delta^+(k)} x_{kj} = 0 \qquad k \in V \setminus \{v_s, v_e\} \tag{7}$$

$$\sum_{ij \in \delta^-(j)} \left(\hat{t}_{ij} + \sum_{m \in T^{ij} \setminus \hat{T}^{ij}} (1 + \theta_{ij}^m) t_{ij}^m + \sum_{m \in T^{ij}} \eta_{ij}^m x_{ij}^m \right) \leq t_j \qquad j \in V \setminus \{v_s\} \tag{8}$$

$$\sum_{ij\in\delta^+(i)}\left(\hat{t}_{ij}+\sum_{m\in T^{ij}\setminus\hat{T}^{ij}}t_{ij}^m\right)=t_i \qquad i\in V\setminus\{v_e\} \tag{9}$$

$$w_{ij}^m x_{ij}^m \le t_{ij}^m \le w_{ij}^{m+1}x_{ij}^m \qquad ij\in A, m\in T^{ij}\setminus\hat{T}^{ij} \tag{10}$$

$$\sum_{m\in\hat{T}^{ij}} w_{ij}^m x_{ij}^m \le \hat{t}_{ij} \le \sum_{m\in\hat{T}^{ij}} w_{ij}^{m+1}x_{ij}^m, \qquad ij\in A \tag{11}$$

$$\hat{t}_{ij}, t_{ij}^m, t_i \ge 0 \qquad ij\in A, m\in T^{ij} \tag{12}$$

$$y_i, x_{ij}, x_{ij}^m \in \{0,1\} \qquad ij\in A, m\in T^{ij} \tag{13}$$

The objective function (3) minimizes the duration while maximizing the profits collected. Constraints (4) - (7) define the x_{ij} and y_i variables in terms of x_{ij}^m and establish that the vehicle must depart from v_s, finish at v_e, and impose the classical flow conservation constraints. Constraints (8) compute the ready-time for vertex j, if visited. We remark that these constraints are a strengthened version of the ones considered in Sun et al. (2018); Montero et al. (2017). Constraints (9) express variables t_i in terms of t_{ij}^m and $\hat{t}_{ij}$ as explained in Lera-Romero & Miranda-Bront (2018), and constraints (10) and (11) impose the relation between variables t_{ij}^m, $\hat{t}_{ij}$ and x_{ij}^m. Finally, constraints (12) and (13) establish the domain for all variables. We refer the reader to Lera-Romero & Miranda-Bront (2018) for a detailed explanation regarding constraints (10) and (11).

3.2 Equivalence between TTBF and CTTBF

Based on the experimental results obtained in Lera-Romero & Miranda-Bront (2018), the strengthened version of the TTBF and CTTBF relaxations seem to be equivalent. In this section, we show that model TTBF proposed in Sun et al. (2018) and strengthened in Lera-Romero & Miranda-Bront (2018) is equivalent to CTTBF. Moreover, we define a procedure to transform solutions from TTBF to solutions of CTTBF and viceversa. Let z^*_{TTBF} and z^*_{CTTBF} be the optimal objective value of the LP relaxation for TTBF and CTTBF, respectively.

Proposition 1. $z^*_{TTBF} = z^*_{CTTBF}$.

Proof. Recall that CTTBF is constructed by aggregating some variables t_{ij}^m into a smaller subset of variables $\hat{t}_{ij}$ formulations CTTBF and TTBF are similar. In particular variables x_{ij}^m and y_i remain the same. Therefore, we consider only those constraints that involve variables t_{ij}^m. Constraints (8) from CTTBF replace constraints

$$\sum_{ij\in\delta^-(j)}\left(\sum_{m\in T^{ij}}(1+\theta_{ij}^m)t_{ij}^m+\sum_{m\in T^{ij}}\eta_{ij}^m x_{ij}^m\right)\le t_j \quad j\in V\setminus\{v_s\} \tag{14}$$

from the formulation TTBF. Furthermore, constraints (9) are the adaptation of the TTBF constraints

$$\sum_{ij\in\delta^+(i)}\sum_{m\in T^{ij}} t_{ij}^m = t_i \quad i\in V\setminus\{v_e\} \tag{15}$$

Then, constraints (10) remain the same, however constraints (11) are an aggregation of the original constraints over $m\in\hat{T}^{ij}$

$$w_{ij}^m x_{ij}^m \le t_{ij}^m \le w_{ij}^{m+1}x_{ij}^m \quad ij\in A, m\in T^{ij}. \tag{16}$$

Now, we show how to transform feasible solutions between TTBF and CTTBF while keeping the same objective value. Transforming solutions from TTBF to CTTBF is done by using the definition of $\hat{t}$. Given a (fractional) solution of the linear relaxation of TTBF $z=\langle x,y,t\rangle$, we construct a solution $z'=\langle x',y',t',\hat{t}'\rangle$ of the linear relaxation of CTTBF as follows:

- $x' = x, y' = y$.
- $t'_i = t_i$ for each $i \in V$.
- $t'^m_{ij} = t^m_{ij}$ for each $ij \in A, m \in T \setminus \hat{T}^{ij}$.
- $\hat{t}'_{ij} = \sum_{m \notin \hat{T}^{ij}} t^m_{ij}$ for each $ij \in A$.

Replacing $\hat{t}'_{ij}$ by its definition in equations (8) and (9) we obtain the original constraints (14) and (15). It is easy to see that constraints (11) also hold for z'. Therefore, since the objective function only depends on variables t_i and y_i and they have the same value in z, respectively, and z' we have a valid solution z' for CTTBF with the same objective function than z for TTBF.

Next we concentrate in the remaining case, let $z' = \langle x', y', t', \hat{t}' \rangle$ be a feasible solution of CTTBF, we construct a solution $z = \langle x, y, t \rangle$, so that it is feasible and has the same objective value, as follows:

- $x = x', y = y'$.
- $t_i = t'_i$ for each $i \in V$.
- $t^m_{ij} = t'^m_{ij}$ for each $ij \in A, m \notin \hat{T}^{ij}$.
- $t^m_{ij} = w^m_{ij} x^m_{ij} + \beta_{ij} \alpha^m_{ij}$ for each $ij \in A, m \in \hat{T}^{ij}$ where
  - (i) $\beta_{ij} = \hat{t}'_{ij} - \sum_{m' \in \hat{T}^{ij}} w^{m'}_{ij} x^{m'}_{ij}$,
  - (ii) $\alpha^m_{ij} = \frac{w^{m+1}_{ij} x^{m+1}_{ij} - w^m_{ij} x^m_{ij}}{\sum_{m' \in \hat{T}^{ij}} w^{m'+1}_{ij} x^{m'+1}_{ij} - w^{m'}_{ij} x^{m'}_{ij}}$.

First, observe that $\beta_{ij}\alpha^m_{ij}$ is non-negative for each arc $ij \in A$ and $m \in \hat{T}^{ij}$ as constraints (11) impose that $\sum_{m' \in \hat{T}^{ij}} w^{m'}_{ij} x^{m'}_{ij} \leq \hat{t}'_{ij}$. Therefore, the first inequality of constraints (16) holds for each arc $ij \in A$ and $m \in \hat{T}^{ij}$. Additionally, note that constraints (11) also impose that $\hat{t}'_{ij} \leq \sum_{m' \in \hat{T}^{ij}} w^{m'+1}_{ij} x^{m'}_{ij}$, and therefore it is easy to obtain, with some algebraic manipulation, that the second inequality of constraints (16) also holds for each arc $ij \in A$ and $m \in \hat{T}^{ij}$. Furthermore, it can be easily shown that

$$\hat{t}'_{ij} = \sum_{m \in \hat{T}^{ij}} t^m_{ij}$$

Therefore, since $\sum_{m \in \hat{T}^{ij}} t^m_{ij} = \hat{t}'_{ij}$ then we know that constraints (14) and (15) are valid because (8) and (9) are valid for z'. Finally, note that variables t_i, y_i are the same than t'_i, y'_i implying that the objective value of both solutions is the same. □

3.3 Resource and objective adaptations

The model can be easily adapted to handle different objective functions and resources. Minimizing makespan instead of duration can be easily done by setting $t_{v_s} = 0$. Alternatively, TSP variants where each vertex must be visited exactly once can be recovered by setting $y_i = 1$ for $i \in V$. Additionally, time windows can be modeled with a constraint that imposes a restriction on variables t_i.

$$a_i y_i \leq t_i \leq b_i y_i \quad i \in V \tag{17}$$

To model demands and vehicle capacity, on the other hand, we have two different approaches depending on whether or not demands can take negative values. If all demands are positive, then it suffices to require that the sum of the routes' demands do not exceed the vehicle capacity.

$$\sum_{i \in V} y_i d_i \leq Q \tag{18}$$

For pick and delivery variants, recall that the demand for the delivery is modelled as a negative quantity. We handle the case of negative demands using the same approach as Sun et al. (2018). Let variables $Q_i \geq 0$ represent the demand carried by the vehicle after servicing customer i. We must verify that any Q_i is smaller or equal than the vehicle capacity.

$$Q_i + q_j \leq Q_j + M(1 - x_{ij}) \quad ij \in A \tag{19}$$

$$0 \leq Q_i \leq \min\{Q, Q + q_i\} \quad i \in V \tag{20}$$

4 Polyhedral study

In this section we define several families of valid inequalities for CTTBF to improve the quality of the lower bound provided by the LP relaxation by exploiting the time-dependent structure explicitly. Since they only consider variables x_{ij}^m these inequalities are valid also for TTBF. In Sections 4.1 and 4.5 we present families of valid inequalities from the literature which are also valid for CTTBF. In Section 4.2 we introduce two families of valid inequalities which generalize traditional inequalities for the time-dependent case. Finally, in Section 4.4 we discuss the relationship among some of families of inequalities.

4.1 Generalized Cut Set inequalities

The first family of inequalities we present is known as *Generalized Cut Set* (GCS), explained in Taccari (2016), which is a generalization of the classical *Subtour Elimination Constraints* (SEC) for the case where not all customers are required to be visited. Given a set $S \subseteq V \setminus \{v_s, v_e\}$ with $|S| \geq 2$, and a vertex $k \in S$, the GCS inequality is defined by

$$\sum_{ij \in \delta^+(S)} x_{ij} \geq y_k \tag{21}$$

Generally speaking, these inequalities impose that for any subset S of vertices not including the end depot, where at least one of them is visited, there must be at least one arc leaving S, otherwise there would be a subtour. It is straightforward to see that this family of inequalities is valid for CTTBF.

4.2 Time-dependent inequalities

The following families of valid inequalities exploit the particular structure of the time-dependent travel time function. For the sake of simplicity, we first introduce some notation used throughout the section. Recall that $\hat{A}$ is the set of all time-dependent arcs, and T_m^{ij} denotes the interval when departure from customer i occurs for time period m. Similarly, we define the interval $\vec{T}_m^{ij} = [w_{ij}^m + \tau_{ij}(w_{ij}^m), w_{ij}^{m+1} + \tau_{ij}(w_{ij}^{m+1})]$ when it is possible to arrive to customer j if departing from customer i in T_m^{ij}.

Suppose P is a feasible solution, and $t \in [0, T]$ the best departing time for route P, which as noted in Section 2.1 does not contain waiting times. Suppose arcs $ij, jk \in A$ are used in P

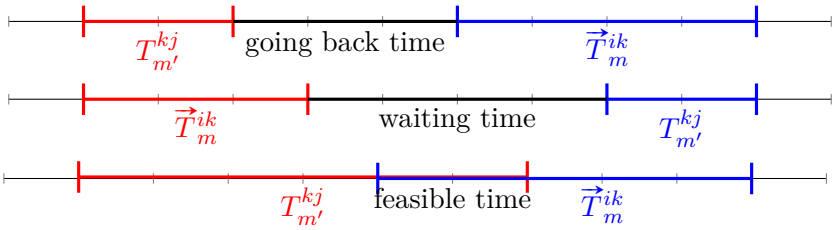


FIGURE 3. *Example on the concept of time-dependent arc successor.*

departing from each in time periods $m \in T^{ij}$, $m' \in T^{jk}$ respectively. Intuitively, since no waiting times are required, then m and m' must be somehow related. In particular, the time when it is feasible to arrive at j if departing from i in time period m should overlap with the feasible time to depart from j to k in time period m'. We use this idea to define the notion of *arc successor*.

Consider a pair of time-dependent arcs $ijm, jkm' \in \hat{A}$, we say jkm' is a *successor* of ijm if $\vec{T}^{ij}_{m} \cap T^{jk}_{mk'} \neq \varnothing$. Furthermore, we define the set of successors $\hat{\delta}^{+}(ijm)$ of arc $ijm \in \hat{A}$ as

$$\hat{\delta}^{+}(ijm) = \{jkm' \in \hat{A} : \vec{T}^{ij}_{m} \cap T^{jk}_{m'} \neq \varnothing\} \tag{22}$$

More generally, we extend the definition of successors of a set of arcs $F \subseteq \hat{A}$ as

$$\hat{\delta}^{+}(F) = \bigcup_{ijm \in F} \hat{\delta}^{+}(ijm) \setminus F, \tag{23}$$

which accounts for the arcs that are successors to at least one arc in F but do not belong to F.

Figure 3 shows three different situations. In the first two, both arcs can not appear in the same solution as they either imply waiting times or require traveling back in time. The last scenario depicts an example of two arcs that can be simultaneously active in an optimal solution when arriving to k somewhere in the feasible time interval.

4.3 Time-dependent flow inequalities

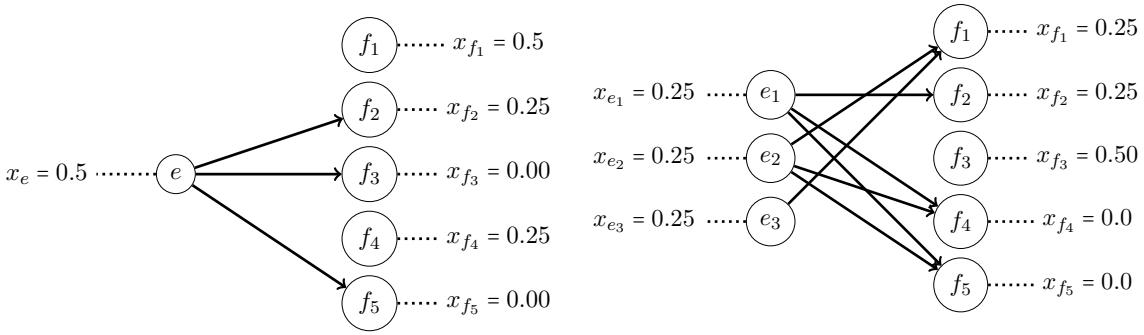
In this section we present a new family of valid inequalities that is based on the well known flow conservation constraints. The idea is that the flow conservation constraints must hold at each visited vertex, but also the flow must be consistent in terms of time-dependency.

The intuition behind the time-dependent flow inequalities is the following. Consider a directed graph $\hat{L}(D) = \langle V^L, A^L \rangle$ where there is a vertex in V^L for each time-dependent arc in $\hat{A}$ and there is an arc from vertex $e \in V^L$ to vertex $f \in V^L$ if $f \in \hat{\delta}^{+}(e)$. We call $\hat{L}(D)$ the time-dependent line digraph of G and, to avoid confusion, we will note its arcs as $e \to f$. Clearly, all feasible integer solutions without waiting times must be a path in $\hat{L}(D)$, otherwise they would include a sequence of time-dependent arcs that force waiting times. As a result, flow must be conserved in $\hat{L}(D)$. We do not distinguish vertices of $\hat{L}(D)$ from time-dependent arcs $\hat{A}$ since there is a bijection among them. Figure 4 shows an example of a valid fractional solution in terms of the classical flow constraints (7) which turns into an invalid solution when considering time-dependency. We show next in Proposition 2 that flow must be conserved between a time-dependent arc and its successors.

Proposition 2. *The Time-dependent Single Flow inequalities defined by*

$$x^{m}_{ik} \leq \sum_{kjm' \in \hat{\delta}^{+}(ikm)} x^{m'}_{kj} \quad ikm \in \hat{A}, k \neq v_e \tag{24}$$

are valid for CTTBF.



(A) *Violated single time-dependent flow inequality for* $e \in \hat{A}$, $\hat{\delta}^+(e) = \{f_1, f_2, f_3, f_4, f_5\}$.

(B) *Violated time-dependent flow inequality for arcs* F *entering vertex* $k \in V$ *defined as* $F = \{e_1, e_2, e_3\} \subseteq \hat{A}$, $\hat{\delta}^+(F) = \{f_1, f_2, f_3, f_4, f_5\}$.

FIGURE 4. *Example of the time-dependent flow inequalities.*

Proof. Let $\bar{x}$ be an integer solution representing path P. The case of $\bar{x}^m_{ij} = 0$ is trivial. Conversely, $\bar{x}^m_{ij} = 1$ means that arc ijm belongs to P. Since P is finite and elementary then there exists an arc $jlm' \in P$ which follows ijm. It follow that arc jlm' must be a successor of ijm, otherwise it would not be in an optimal solution. □

We extend this notion by taking a group of arcs entering some vertex $k \in V \setminus \{v_e\}$ instead of one. Let $F \subseteq \hat{A}$ be a set of arcs entering k, i.e. $ijm \in F$ iif $j = k$. We know that flow entering and leaving arc k must be equal, therefore the flow entering k through arcs in F must leave in their successors $\hat{\delta}^+(F)$. Note we do not consider $k \in \{v_s, v_e\}$ since flow is not conserved at those vertices.

Proposition 3 (TDFI)**.** *The* Time-Dependent Flow inequalities (TDFI)

$$\sum_{ikm \in F} x^m_{ik} \leq \sum_{kjm' \in \hat{\delta}^+(F)} x^{m'}_{kj} \quad k \in V \setminus \{v_s, v_e\}, F \subseteq \{ikm \in \hat{A}\} \tag{25}$$

are valid for CTTBF.

Proof. The proof is analogous to the proof of Proposition 2 if we notice that at most one variable can be selected on each side of the inequality in an integer solution because of constraints (4). □

Next, we illustrate some situations that are captured by these inequalities. For instance, suppose we have a customer $k \in V \setminus \{v_s, v_e\}$ and we have a subset $\hat{A}_{in} \in \hat{A}$ of the arcs arriving at k before time $t \in [0, T]$, and a subset $\hat{A}_{out} \subseteq \hat{A}$ of the arcs departing from vertex k strictly after time t. It is easy to see that at most one arc in $\hat{A}_{in} \cup \hat{A}_{out}$ can be selected in an optimal integer solution, because otherwise the solution would incur in waiting times. This particular case is captured by the TDFI by choosing $F = \{ikm \in \hat{A} : \max(\vec{T}^{ik}_m) \leq t\}$. Similarly, a set of arcs arriving to k after time t and another with arcs departing from k before time t can not contain two arcs selected at the same time. This situation represents solutions that travel back in time. Again, we can handle such scenarios by choosing proper sets of arcs F to define TDFI.

4.3.1 Time-dependent Cut Set inequalities

The last family of inequalities is a generalization of the GCS to handle time dependency. The idea behind this family of inequalities is to cut fractional solutions inducing a subtour in the

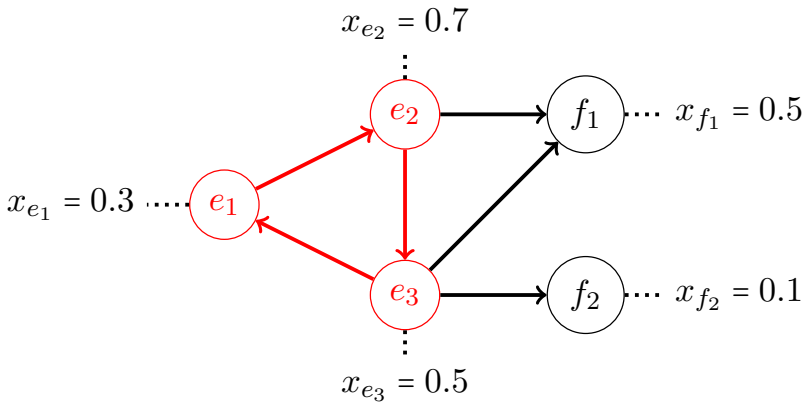


FIGURE 5. *Violated TDCSI inequality for* $S = \{e_1, e_2, e_3\}$ *and* $\hat{\delta}^+(S) = \{f_1, f_2\}$.

digraph formed by the time-dependent arcs. In Proposition 4 we introduce a first family of inequalities, which is later strengthened in Proposition 5.

Following the idea of the GCS we say that given a set of arcs S not entering v_e, the flow of the successors of S must be at least the flow passing through each arc in S. Figure 5 depicts the situation where a subtour of time-dependent arcs e_1, e_2, e_3 is present and flow from arc e_2 is greater than the flow leaving S. The next family of valid inequalities aims to forbid this situation in feasible solutions of the LP relaxation.

Proposition 4. *The inequalities defined as*

$$x^m_{ik} \leq \sum_{jlm' \in \hat{\delta}^+(S)} x^{m'}_{jl} \quad S \subseteq \{ijm \in \hat{A} : j \neq v_e\}, ikm \in S \tag{26}$$

are valid for the CTTBF.

Proof. As is the case of the GCS, the key to this proof is to notice that no arc entering the end depot v_e is selected in S, and therefore if arc $ikm \in S$ is selected then there must be a path of successors from ikm to the end depot that are selected. Let $S \subseteq \{ijm \in \hat{A} : j \neq v_e\}$ be a set of arcs, $ikm \in S$ an arc included in S, and $\bar{x}$ an integer solution of CTTBF. If $\bar{x}^m_{ik} = 0$, the inequality is trivially satisfied. Otherwise, let $\bar{x}^m_{ik} = 1$, then as $\bar{x}$ represents a feasible integer solution it is associated with a path $e_1 \to \cdots \to e_r \to ikm \to e_{r+1} \to \cdots \to e_s$ of vertices in $\hat{L}(D)$ where e_s arrives to the end depot. Since $ikm \in S$ and $e_s \notin S$ there must be at least one vertex e_l between e_{r+1} and e_s which is not in S. Let l be the index of such first arc. We have that $e_{l-1} \in S$, and $e_l \notin S$, and also that $e_l \in \hat{\delta}^+(e_{l-1})$, therefore $e_l \in \hat{\delta}^+(S)$. As a result, the right hand side of the inequality is at least one. □

As mentioned before, inequalities (26) can be strengthened considering that only one arc entering each vertex $k \in V$ can be selected in any integer solution. Proposition 5 shows this new family of inequalities and proves its validity.

Proposition 5 (TDCSI)**.** *The Time-Dependent Cut Set inequalities (TDCSI) defined as*

$$\sum_{ikm \in S \cup \hat{\delta}^+(S)} x^m_{ik} \leq \sum_{ijm \in \hat{\delta}^+(S)} x^m_{ij} \quad k \in V, S \subseteq \{ijm \in \hat{A} : j \neq v_e\} \tag{27}$$

are valid for the CTTBF.

Proof. Note that only one arc $ikm \in \hat{A}$ entering vertex $k \in V$ can be selected in any integer solution. Therefore, we know that the left hand side value is at most one. The case when this expression is exactly 0 is trivially satisfied, so we consider only the case when the left hand side value is one. So, we have two possible scenarios

- if $\bar{x}_{ik}^m = 1$ for some $ikm \in S$, then by Proposition 4 the right hand side is at least one.
- if $\bar{x}_{ik}^m = 0$ for all $ikm \in S$, then the left hand side variables are a subset of the right hand side, so the inequality is valid.

The strengthened version of the TDCSI, therefore, is valid for CTTBF. □

4.4 Relationship among families of valid inequalities

In this section we discuss the relationship among different families of inequalities. First, we show that the TDCSI generalize both the TDFI and the GCS. Then, we establish a connection between the TDCSI and some valid inequalities proposed for another variant of the TDTSP.

4.4.1 Valid inequalities for the CTTBF

We begin by showing that the TDFI can be expressed as a particular case of the TDCSI.

Remark 1. *The TDFI inequalities are a particular case of the TDCSI.*

Proof. Let $k \in V \setminus \{v_s, v_e\}$ be a customer and $F \subseteq \{ikm \in \hat{A}\}$ be a set of time-dependent arcs entering k that define a TDFI. It can be easily verified that the former inequality is equivalent to the TDCSI defined by $S = F$ and vertex k. □

Observe that, although the TDCSI imply the TDFI, in Section 4.6.2 we show that TDFI has a fast separation procedure, while the separation of the TDCSI is done by using min-cut algorithms which usually have slower running times. Next, we also show how the classical GCS can be seen as a particular case of the TDCSI. The idea behind this analysis is the fact that both inequalities attempt to separate solutions with subtours.

Proposition 6. *The TDCSI generalize the GCS.*

Proof. Let $S \subseteq V \setminus \{v_s, v_e\}$ a set with at least two customers and $k \in S$ a customer that define a GCS inequality. Let $S' = \{ijm \in \hat{A} : j \in S\}$ be a set of time-dependent arcs that define a TDCSI. Firstly, note that constraints (4) and (5) imply that $y_k = \sum_{ikm \in S' \cup \hat{\delta}^+(S')} x_{ik}^m$ since, by definition, all of the arcs entering $k \in S$ belong to S'. Additionally, we have that $\hat{\delta}^+(S') = \{ijm \in \hat{A} : ij \in \delta^+(S)\}$. To see this, note that for each arc $ij \in \delta^+(S)$, by definition $i \in S$, and therefore, all time-dependent arcs $kim \in \hat{A}$ entering vertex i belong to S'. Also we have that $j \notin S$, and since every arc $ijm \in \hat{A}$ must be a successor of at least one other arc entering i, otherwise it can be removed, we have that $ijm \in \hat{\delta}^+(S')$. Finally, it is easy to see that each arc $ijm \in \hat{\delta}^+(S')$ verifies that $ij \in \delta^+(S)$, by noting that i must be the head of any arc in S' and j can not be in S. □

4.4.2 Connections to other problems

We briefly discuss about the relation between the TDCSI with other time-dependent problems, in particular with the TDTSP where the cost of an arc depends on its position in the tour (see e.g. Picard & Queyranne (1978)). The problem can be defined on a particular multipartite network, where vertices in this network encode a combination between the vertices in the original

problem with its position in the tour. Therefore, there is an implicit definition of *successors* that shares some of the ideas proposed in this paper.

Some recent exact approaches for this variant of the TDTSP focus on polyhedral results and BC algorithms, and some connections can be established with these problems. By considering an appropriate definition of the set of successors of a time dependent arc, the so-called *time-dependent cycle inequalities* proposed in Miranda-Bront et al. (2014) are generalized by the TDCSI. Another example are the *admissible flow constraints* proposed in Abeledo et al. (2012) which present a similar structure as well, although their definition somehow implicitly encode the particular structure of the multipartite graph. Furthermore, these two families of inequalities also share characteristics with some valid inequalities for the Generalized TSP.

The relation among these families of valid inequalities exceeds the scope of this paper, and further investigations could follow to precisely establish the connections among the different problems. We highlight, however, that our notion of successors captures the nature of the time dependency and depends directly on the travel times between customers.

4.5 Precedence-constrained TSP inequalities

One of the variants considered to evaluate the performance of the BC algorithm is the TDTSPTW. Under the presence of time windows and with the constraint that all customers must be visited, precendences can be inferred among customers using this information. Several families of valid inequalities have been proposed that exploit these precedences, which are quite effective in improving the LP relaxation even when considering time-dependent travel times as shown in Montero et al. (2017). Thus, following their approach, we also include the well known π, σ and (π, σ) inequalities when tackling the TDTSPTW.

4.6 Separation routines

In this Section we present the separation routines for the inequalities introduced in the previous sections. First, we briefly mention the separation routines considered for the valid inequalities adapted from related problems. Then, we present the separation algorithms for the new families, which are explained in detail in Appendix A.

4.6.1 Valid inequalities from related problems

The GCS are separated following the standard approach described in Taccari (2016). As regards the π, σ and (π,σ) we follow the original approach proposed in Balas et al. (1995). We remark that, in both cases, the separation algorithms are based on solving a series of min-cut problems in a subgraph.

4.6.2 Time-dependent Flow inequalities

The worst case complexity of the separation problem for the TDFI is polynomial. The algorithm uses a traditional dynamic programming approach, similar to the standard one for the Subset Sum problem. Recall that our aim is to maximize the violation of the TDFI defined in (25), which formally translates to $\max_{F,k} \sum_{ikm \in F} x^m_{ik} - \sum_{kjm' \in \hat{\delta}^+(F)} x^{m'}_{kj}$ with F, k holding the inequality conditions.

Let $k \in V \setminus \{v_e\}$ be a vertex, $I = \{e_1, \cdots, e_r\}, O = \{f_1, \cdots, f_s\}$ the sets of time-dependent arcs entering and leaving k respectively, and $\bar{x}$ a fractional solution of CTTBF. Suppose I is sorted by the start of the arrival interval of its arcs $\min(\vec{T}_{e_1}) \leq \cdots \leq \min(\vec{T}_{e_r})$ and O by the start of the departure interval $\min(T_{f_1}) \leq \cdots \leq \min(T_{f_s})$. Given an index $i \in [1 \dots r]$, we define $next(i)$

as the index of the first arc in O whose departure interval is after the end of the arrival interval of i, and $weight(i,j)$ as the sum of the values of $\bar{x}$ for all the successors of e_i that have index at least j.

$$next(i) = \min\left\{j : \min(T_{f_j}) > \max(\vec{T}_{e_i}) \vee j = s+1\right\} \tag{28}$$

$$weight(i,j) = \sum_{j' \geq j, f_{j'} \in \hat{\delta}^+(e_i)} \bar{x}_{f_{j'}} \tag{29}$$

Now, we can define the dynamic programming function f as the maximum violation for k considering only entering arcs $i \dots r$ and leaving arcs $j \dots s$ as

$$f(i,j) = \max\left\{f(i+1,j), f(i+1, \max\{next(i), j\}) + \bar{x}_{e_i} - weight(i,j)\right\}, \tag{30}$$

with boundary conditions $f(i,j) = 0$ if $i = r+1$.

Proposition 7. *Let $\hat{\Delta}$ be the maximum number of time-dependent arcs incident to a vertex. The exact separation procedure for the TDFI is polynomial and can be solved in $\mathcal{O}(|V|\hat{\Delta}^2)$.*

A detailed proof of Proposition 7 can be found in A.

4.6.3 Time-dependent Cut Set inequalities

In the case of the TDCSI the exact separation is also polynomial but computationally more expensive, which is expected as it is a generalization of the TDFI. The general idea is similar to the separation of the GCS inequalities, in this case we consider the time-dependent line digraph $\hat{L}(D)$. Then, we solve one min-cut problem for each customer $k \in V \setminus \{v_s, v_e\}$ where the capacity is now on the vertices of $\hat{L}(D)$ so we need to perform some adaptations to this digraph resulting in an arc-capacitated network we call D^k_{flow}.

Proposition 8. *The exact separation procedure for the TDCSI is polynomial and can be solved in $\mathcal{O}(|V||\hat{A}|^3)$.*

A detailed proof of Proposition 8 and the exact separation routine can be found in A.

5 Branch-and-cut algorithm

In this section we describe the main ingredients of the BC algorithm.

5.1 Preprocessing techniques

We apply some classical preprocessing techniques widely used for VRPs with time windows, including tightening, arc removal, inference of precedences among vertices, etc. Due to space limitations we skip the details, but refer the reader to Montero et al. (2017) for the key ideas applied by our algorithm.

5.2 Initial heuristic

An important consideration for every branch and bound algorithm is having available good quality upper bounds, hopefully, to reduce the number of nodes explored on the search tree and achieve a lower overall running time. Our aim is to find a fast initial heuristic which gives tight initial solutions to feed the branch and bound algorithm with.

We consider a matheuristic based on the ILP formulation CTTBF (3)-(13), but eliminating the time-dependency and solving an auxiliary time-independent problem with only one time period. The motivation behind this strategy is twofold. First, as a result we obtain a general heuristic that can be applied to any problem that can be solved by the time-dependent model. Second, from an implementation standpoint we only need to adjust the definition of the time periods, since the rest is captured directly by the model. Then, the applicability of this technique as an heuristic depends on two factors: its running time, and its quality.

Regarding the running time, it is easy to see that the time-independent model has considerably less variables and constraints than its time-dependent counterpart. Therefore, it is possible to solve it to optimality in a couple of seconds for many of the time-dependent difficult instances. If optimality is not reached in this short time limit, from preliminary experiments that in general it produces good quality feasible solutions. We acknowledge that this approach may have to be reconsidered when aiming to solve larger instances than the ones considered in our case. The time limit of the initial heuristic set for the experimentation is two seconds.

Additionally, it is important to notice that the transformation to a time-independent instance can be made in several ways. In this work, our choice is to preserve feasibility. Therefore, the transformation made is essentially to set for each arc $e \in A$ its constant time-independent travel time c_e as the worst travel time over the planning horizon by defining a unique time period with the slowest travel speed for e. Any feasible solution using this new constant travel times is also feasible in the time-dependent version even if the problem considers additional operational constraints, such as time windows.

5.3 Cutting plane algorithm

Another effective strategy to improve the quality of the lower bound provided by the LP relaxations is to incorporate valid inequalities in a cutting plane algorithm and strengthen the formulation. As a counterpart, this may lead to slower running times for solving each relaxation. Therefore, the improvement obtained in the lower bound, the time needed to identify the violated inequalities and the size of the resulting (improved) formulation require a good compromise in terms of computing time. Based on limited preliminary experiments and taking into account the behavior reported in Montero et al. (2017), we also develop a Cut and Branch (C&B) algorithm where the cutting phase is executed only at the root node. Then, the resulting information is freezed and solved using a standard MILP solver. This strategy achieves faster running times compared to executing a cutting phase at each node of the enumeration tree. For this reason, we also follow this approach as regards the cutting plane algorithm. We consider an aggressive cutting planes strategy, where all inequalities found are added to the formulation at every iteration. The motivation behind this decision is to compensate the fact that we are only including cuts at the root node.

The families of inequalities GCS, TDFI and TDCSI are valid for all the variants considered in this paper. Therefore, they can be separated for any variant of the CTTBF. However, preliminary experiments show that the TDCSI increase the overall running time and thus are not included in the cutting plane algorithm. This can be due to the fact that TDFI capture most of the time-dependent information regarding the flow in the graph and time-dependent subtours might be too specific. Another alternative could be to improve the separation routine, aiming to diversify the structure of the inqualities added to the formulation. We observe that adding these inequalities as cuts may translate into slower computing times for solving the linear relaxation while bringing a small improvement in the objective function. In this sense, we focus primarily on the impact of the new TDFI combined with the compact model. Finally, all the GCS inequalities of size 2 are included as part of the formulation, since it is known that they

have a positive impact on the LP relaxation.

Additionally, when the problem is a TDTSP with time windows, π, σ, and (π, σ) inequalities are considered in the cutting plane algorithm as well. We remark that (π,σ) inequalities are separated only if no violated π and σ inequalities are found.

As regards the separation routines described in Section 4.6, we set a threshold of $\Delta = 0.1$, meaning that an inequality must have a violation of at least Δ to be added as a cut. Additionally, we only add cuts while the relaxation improved at least 0.1 in the last cutting planes iteration. Intuitively, this prevents the inclusion of inequalities that do not have a big impact on the LP relaxation.

6 Computational results

The experiments are conducted on an Intel(R) Core(TM) i5-4570 3.20GHz CPU workstation with 16GB of RAM, running Ubuntu 16.04.4 LTS, and using C++ as the programming language. The algorithms are evaluated on benchmark instances for three different variants of the TDESPPRC. We use CPLEX 12.8.0 as an MILP solver.

The next sections show the results for the three different problems considered, including the benchmark methods considered in each case. In this sense, a direct comparison with state-of-the-art algorithms for each variant allows to explore the potential to the BC algorithm in different contexts. A subset of the following metrics is reported for each experiment, which may vary depending on the information available from the benchmark methods: the number of instances solved to optimality (OPT), the execution time in seconds (time), the BC root node gap (%rGap), the lower bound at the root node (LB), the upper bound after reaching the time limit (UB), and the number of nodes (nodes) enumerated in the BC tree.

6.1 TDCESPP instances and the impact of TDFI

In this section we focus on analyzing the impact of the new TDFI constraints when used as cuts. We perform our experiments on the benchmark instances proposed in our previous work Lera-Romero & Miranda-Bront (2018). Recall that this dataset is composed of pricing instances that aroused from different points in the execution of a column generation algorihtm on Dabia et al. (2013) instances without the time windows. There are six types of instances and for each one of them five different profits selected as explained before. The methods evaluated, over a two hours time limit, are:

- CTTBF-GCS: The BC algorithm for the TDPTPRC described in Lera-Romero & Miranda-Bront (2018), which is the BC algorithm defined in Section 5 but only considering GCS inequalities and CPLEX cuts.[2]
- CTTBF-All: The BC algorithm defined in Section 5 adapted for the capacitated variant as specified in Section 3.3, with all the cuts valid for this problem enabled (i.e. CPLEX, GCS, TDFI).

Table 1 presents the average results over each of the original instances. The most interesting results from Table 1 are the average time reduction by about %41.08 and also the decrease in the average nodes enumerated by %52.18 when adding the TDFI as cuts. Furthermore, when considering the new inequalities all instances can be solved to optimality within the time limit whereas one instance could not be proved to be optimal with the other configuration. We

[2]We remark that the results reported in Lera-Romero & Miranda-Bront (2018) consider the optimal solution as an initial feasible solution, which may cause differences in terms of the overall execution of the BC algorithm.

		CTTBF-GCS				CTTBF-All		
Type	Inst	OPT	time	%rGap	nodes	OPT	time	%rGap	nodes
C1	5	5	137.34	6.01	20062.20	5	142.72	4.97	11708.60
R1	5	5	437.96	19.07	58587.80	5	457.68	18.44	58378.60
RC1	5	5	577.11	12.34	44314.60	5	475.29	11.71	29808.40
C2	5	4	3489.92	4.27	621992.00	5	1180.85	3.99	186862.80
R2	5	5	857.79	4.62	292900.80	5	723.81	4.50	219136.60
RC2	5	5	190.61	1.76	56369.80	5	62.72	1.71	17262.00
TOT	30	29	860.82	8.01	182371.20	30	507.18	7.55	87192.83

TABLE 1. *Average aggregated results for instances with 25 customers.*

remark that these improvements become more significant when compared to the non-compact formulation TTBF, as reported in Lera-Romero & Miranda-Bront (2018).

6.2 TDPTP with time windows and pickup and delivery

We next evaluate the performance of our model in the TDPTP variant which considers pickup and deliveries, time windows, and capacities. The experiments are carried on over a subset of the instances proposed in Sun et al. (2018). First, we remark that the MILP formulation proposed in Sun et al. (2018) inspired some preliminary work on BC algorithms for other variants of single vehicle time dependent problems, including this paper. Second, we note that this MILP formulation is not the best approach reported in Sun et al. (2018). In fact, they develop a labeling based algorithm which outperforms the former method which exploits the narrow time windows of the instances. In this sense, our objective is to show the performance improvements when using the CTTBF together with the new valid inequalities. Regarding the experiments, we set a time limit of one hour for the execution time.

- ORIG-TTBF: The BC algorithm introduced in Section 3 of Sun et al. (2018) which is a weaker version of CTTBF since some of their constraints has been strengthened. They do not report any family of inequalities being added as cuts to the formulation.
- CTTBF-CTWPD: The BC algorithm defined in Section 5 adapted for the capacitated variant with pickup and deliveries and time windows as specified in Section 3.3, with all the cuts valid for this problem enabled (i.e. CPLEX, GCS, TDFI).

A first look at the results indicate that CTTBF-CTWPD is able to solve only one more instance than ORIG-TTBF, although it obtains a dramatic reduction in the computation times required. We point out, however, that the computer used in Sun et al. (2018) is considerably faster than ours, and that they set a time limit of 24 hours while our method is stopped after only 1 hour. Therefore, even by neglecting the difference about the environment and considering instances solved within 1 hour in both cases, CTTBF-CTWPD is able to solve 6 more instances in far less computation total time. This shows that both the improvements in the formulation combined with the tailored BC algorithm are effective to improve the overall behavior.

6.2.1 TDTSPTW

Finally, we analyze the performance of our algorithm on the TDTSPTW variant. We include two sets of instances, Set1 and w100, originally proposed in Arigliano et al. (2015) and later in Vu et al. (2018). The difference between both datasets is the size of the partition of the planning horizon into time periods where the former has considerably less time periods than the latter. The time limit imposed for these experiments is one hour. The methods considered are

	ORIG-TTBF				CTTBF-CTWPD			
Inst	LB	UB	%fGap	time	LB	UB	%fGap	time
AA10	27.14	27.14	0.00	1.95	27.14	27.14	0.00	0.52
AA15	37.77	37.77	0.00	4664	37.77	37.77	0.00	0.67
AA20	73.7	73.7	0.00	24.93	73.7	73.7	0.00	5.54
AA25	203.01	203.01	0.00	46344.063	203.01	203.01	0.00	302.75
AA30	216.34	543.475	151.21	-	266.72	266.72	0.00	1543.96
BB10	39.26	39.26	0.00	1.451	39.26	39.26	0.00	1.17
BB15	99.58	99.58	0.00	121.029	99.58	99.58	0.00	2.23
BB20	138.05	138.05	0.00	1669.737	138.05	138.05	0.00	28.45
BB25	147.88	147.88	0.00	48616.51	147.88	147.88	0.00	256.38
BB30	245.7	515.404	109.77	-	245.7	366.59	32.98	-
CC10	57.63	57.63	0.00	3.292	57.63	57.63	0.00	1.49
CC15	98.29	98.29	0.00	44.337	98.29	98.29	0.00	2.23
CC20	97.27	97.27	0.00	7935.288	97.27	97.27	0.00	10.03
CC25	209.088	1316.688	529.73	-	194.31	481.83	59.67	-
CC30	237.02	1674.739	606.58	-	254.84	656.81	61.20	-
DD10	85.88	85.88	0.00	1.435	85.88	85.88	0.00	0.74
DD15	99.05	99.05	0.00	54.649	99.05	99.05	0.00	0.72
DD20	182.14	182.14	0.00	4959.477	182.14	182.14	0.00	28.82
DD25	221.391	1419.806	541.31	-	98.6	499.51	80.26	-
DD30	279.51	1733.624	520.24	-	151.48	677.05	77.63	-
AVG			409.81	8174.44			62.35	138.38

TABLE 2. *Comparison of results over the instances for the TDCESPP with time windows and pickup and delivery.*

	Set 1 (952 instances)			w100 (960 instances)		
Method	OPT	time	%rGap	OPT	time	%rGap
DDD-TD-TSPTW	952	10.86	-	960	1.22	-
TTBF-CB	940	26.22	43.56	214	96.62	40.42
CTTBF-TSPTW	951	5.23	3.66	960	12.12	0.94

TABLE 3. *Average results for TDTSPTW with makespan-minimization objective.*

- DDD-TD-TSPTW: The algorithm introduced in Vu et al. (2018) which considers a model with a discretization of the time horizon which is iteratively refined until an optimum is found. Their original algorithm works for the makespan-minimizing problem but an adaption was also provided to solve the duration-minimizing version.

- TTBF-CB: The algorithm for the TDTSPTW proposed in Montero et al. (2017). They consider the original model which has one variable t_{ij}^m for each $ij \in A, m \in T^{ij}$, and also include the GCS inequalities of size two explicitly in the formulation. Moreover, several families of inequalities such as SECs, π, σ and (π, σ), are added dynamically as cuts in the root node.

- CTTBF-TSPTW: The BC algorithm defined in Section 5 adapted for the TSP variant with time windows considering all valid cuts for this problem (i.e. GCS, π, σ, (π, σ), TDFI).

Table 3 presents the average results over each of the datasets for the makespan-minimization objective. Similarly, in Table 4 we present the average results over each of the datasets for the duration-minimization objective. In this case we only compare methods DDD-TD-TSPTW and CTTBF-TSPTW. We aggregate the results to match the format presented in Vu et al. (2018).

The main message in Table 3 is that the strengthened relaxation of CTTBF-TSPTW, in combination with the compact model and the TDFI cuts show a significant improvement over the results achieved in Montero et al. (2017). Particularly, we see that the average %rGap went from about %40 in TTBF-CB to less than %4 in the new formulation. Furthermore, this is translated into shorter computation times as only one instance is not solved by CTTBF-TSPTW

Method	Set 1 (952 instances) OPT	time	w100 (960 instances) OPT	time
DDD-TD-TSPTW	930	143.45	955	106.20
CTTBF-TSPTW	949	13.69	960	26.67

TABLE 4. *Average results for TDTSPTW with duration-minimization objective.*

against the 12 unsolved instances by TTBF-CB, and requiring considerably less total time.

However, when the objective involves finding the minimum makespan tour, our algorithm solves one instance less than DDD-TD-TSPTW, although the average running times are similar and even smaller for Set 1. In contrast, when the objective function minimizes the duration, our method outperforms DDD-TD-TSPTW. Observe that we can consistently solve the instances with this objective, whereas the other method fails to solve to optimality 33 instances.[3] Furthermore, our algorithm is about an 80% faster than DDD-TD-TSPTW. We believe these results show the versatility of the model and how well it adapts to different objectives.

7 Conclusions

In this paper we propose a new MILP formulation TDPT PRC and develop a tailored BC algorithm which, with some adaptations, can be applied to different problems. From a theoretical standpoint, we provide a guarantee that the procedure applied to obtained the CTTBF formulation starting form a strengthened version of the TTBF proposed in Sun et al. (2018) by reducing variables does not impact in the quality of the LP relaxation. Furthermore, we propose several families of valid inequalities exploiting the time-dependency of the travel time function. These families of are studied in detail, and we develop efficient separation algorithms to include them in a BC algorithm. The proposed algorithm is evaluated in three different problems, including two recent benchmarks from the related literature. We remark that our algorithms showed to be quite effective in all cases, specially for two different variants of the TDTSPTW. As future research, we believe it is worth investigating further families exploiting the time dependency, in particular building upon the TDCSI. In addition, significant improvements can be obtained by finding cutting planes connecting the time-dependency with resource-specific constraints, such as time windows. From a practical perspective, the proposed approach could be suitable for scenarios with no limiting resources are available or, for instance, under the presence o soft time windows.

Acknowledgments

This research was partially funded by FONCyT grant PICT-2016-2677 from the Ministry of Science, Argentina.

References

Abeledo, H., Fukasawa, R., Pessoa, A., & Uchoa, E. (2012). The time dependent traveling salesman problem: polyhedra and algorithm. *Mathematical Programming Computation*, *5*, 27–55.

Arigliano, A., Calogiuri, T., Ghiani, G., & Guerriero, E. (2018). A branch-and-bound algorithm for the time-dependent travelling salesman problem. *Networks*, *72*, 382–392.

[3]We remark, also, that our model can handle continuous time while the DDD-TD-TSPTW requires a discretization when minimizing the duration.

Arigliano, A., Ghiani, G., Grieco, A., & Guerriero, E. (2015). *Time Dependent Traveling Salesman Problem with Time Windows: Properties and an Exact Algorithm.* Technical Report.

Balas, E., Fischetti, M., & Pulleyblank, W. R. (1995). The precedence-constrained asymmetric traveling salesman polytope. *Math. Program.*, *68*, 241–265.

Baldacci, R., Mingozzi, A., & Roberti, R. (2011). New route relaxation and pricing strategies for the vehicle routing problem. *Oper. Res.*, *59*, 1269–1283.

Cordeau, J., Ghiani, G., & Guerriero, E. (2014). Analysis and branch-and-cut algorithm for the time-dependent travelling salesman problem. *Transportation Science*, *48*, 46–58.

Dabia, S., Ropke, S., Van Woensel, T., & de Kok, T. G. (2013). Branch and price for the time-dependent vehicle routing problem with time windows. *Transp. Sci.*, *47*, 380–396.

Drexl, M. (2013). A note on the separation of subtour elimination constraints in elementary shortest path problems. *European Journal of Operational Research*, *229*, 595–598.

Drexl, M., & Irnich, S. (2014). Solving elementary shortest-path problems as mixed-integer programs. *OR spectrum*, *36*, 281–296.

Feillet, D., Dejax, P., Gendreau, M., & Gueguen, C. (2004). An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. *Networks*, *44*, 216–229.

Gendreau, M., Ghiani, G., & Guerriero, E. (2015). Time-dependent routing problems: a review. *Comput. Oper. Res.*, *64*, 189–197.

Huang, Y., Zhao, L., Van Woensel, T., & Gross, J.-P. (2017). Time-dependent vehicle routing problem with path flexibility. *Transp. Res. B*, *95*, 169–195.

Ichoua, S., Gendreau, M., & Potvin, J. (2003). Vehicle dispatching with time-dependent travel times. *Eur. J. Oper. Res.*, *144*, 379–396.

Jepsen, M. K., Petersen, B., Spoorendonk, S., & Pisinger, D. (2014). A branch-and-cut algorithm for the capacitated profitable tour problem. *Discrete Optimization*, *14*, 78–96.

Lera-Romero, G., & Miranda-Bront, J. J. (2018). Integer programming formulations for the time-dependent elementary shortest path problem with resource constraints. *Electronic Notes in Discrete Mathematics*, *69*, 53–60.

Miranda-Bront, J. J., Méndez-Díaz, I., & Zabala, P. (2014). Facets and valid inequalities for the time-dependent travelling salesman problem. *European Journal of Operational Research*, *236*, 891–902.

Montero, A., Méndez-Díaz, I., & Miranda-Bront, J. J. (2017). An integer programming approach for the time-dependent traveling salesman problem with time windows. *Computers & OR*, *88*, 280–289.

Picard, J.-C., & Queyranne, M. (1978). The time-dependent traveling salesman problem and its application to the tardiness problem in one-machine scheduling. *Operations Research*, *26*, 86–110.

Spliet, R., Dabia, S., & Van Woensel, T. (2018). The time window assignment vehicle routing problem with time-dependent travel times. *Transp. Sci.*, *52*, 261–276.

Sun, P., Veelenturf, L. P., Dabia, S., & Van Woensel, T. (2018). The time-dependent capacitated profitable tour problem with time windows and precedence constraints. *Eur. J. Oper. Res.*, *264*, 1058–1073.

Taccari, L. (2016). Integer programming formulations for the elementary shortest path problem. *European Journal of Operational Research*, *252*, 122–130.

Toth, P., & Vigo, D. (Eds.) (2014). *Vehicle Routing: Problems, Methods, and Applications*. 2nd ed. Society for Industrial and Applied Mathematics, Philadelphia, PA.

Verbeeck, C., Sörensen, K., Aghezzaf, E., & Vansteenwegen, P. (2014). A fast solution method for the time-dependent orienteering problem. *European Journal of Operational Research*, *236*, 419–432.

Verbeeck, C., Vansteenwegen, P., & Aghezzaf, E. (2017). The time-dependent orienteering problem with time windows: a fast ant colony system. *Annals OR*, *254*, 481–505.

Vu, D. M., Hewitt, M., Boland, N., & Savelsbergh, M. (2018). *Solving Time Dependent Traveling Salesman Problems with Time Windows*. Technical Report.

A Separation routines

A.1 Proof of Proposition 7

Next we prove that the maximum violation of an inequality (25) is exactly $f(1,1)$. The case when $i = r+1$ is trivially satisfied since there are no entering arcs to k. Let $i < r+1$, F^* be the set of arcs entering k that produce the maximum violated TDFI and $S^* = \hat{\delta}^+(F^*)$. We have two cases either $e_i \in F^*$ or $e_i \notin F^*$. If $e_i \notin F^*$ then the maximum violated inequality is the same as the one produced by the subproblem $(i+1, j)$, that is, only considering arcs $e_{i+1} \cdots e_r$. The interesting case is when $e_i \in F^*$, here we know the violation of the inequality (25) defined by F^* is $\bar{x}_{e_i} - weight(i,j)$ plus the maximum violation produced by $F' = F^* \setminus \{e_i\}$ and successors $S' = S^* \setminus \hat{\delta}^+(e_i)$. We next proceed to show that S' contains only arcs in O from $\max\{next(i), j\} \cdots s$, which completes the proof.

Suppose there exists $i' > i, j' < \max\{next(i), j\}$ such that $f_{j'}$ is a successor of $e_{i'}$ but it is not a successor of e_i. By definition $\vec{T}_{e_{i'}} \cap T_{f_{j'}} \neq \emptyset$, then $\max(T_{f_{j'}}) \geq \min(\vec{T}_{e_{i'}})$. As $j' < next(i)$ and O is sorted we also know that $\min(T_{f_{j'}}) \leq \max(\vec{T}_{e_i})$. Finally, since I is sorted we know $\min(\vec{T}_{e_i}) \leq \min(\vec{T}_{e_{i'}})$. Therefore it holds that $\max(T_{f_{j'}}) \geq \min(\vec{T}_{e_i}) \geq \min(T_{f_{j'}})$ so $f_{j'}$ was a successor of e_i which is a contradiction.

We have proved all the cases, therefore the algorithm must be correct. Note that we can easily reconstruct a set $F \subseteq \hat{A}$ which produces an inequality (25) with violation $f(0,0)$ by using the decision taken at each recursive step.

Finally, we discuss the algorithm complexity. Let $\hat{\Delta}$ be the maximum number of time-dependent arcs incident to a vertex k. Precomputing functions $weight(i,j)$, $next(i)$ for each i, j can be done in $\mathcal{O}(\hat{\Delta}^2)$ time. Then, each step of function f can be computed in constant time, so if memoization is implemented using a matrix then all the possible states of f can be computed in $\mathcal{O}(\hat{\Delta}^2)$. Reconstructing the cut after computing all the states for f can be done in linear time on $\hat{\Delta}$, so the overall complexity of the separation routine is $\mathcal{O}(\hat{\Delta}^2)$ for each vertex $k \in V$. □

A.2 Proof of Proposition 8

For the separation routine we adapt the digraph $\hat{L}(D)$ using common max-flow techniques to construct a network with arc capacities D^k_{flow} for each customer $k \in V \setminus \{v_s, v_e\}$ where $|V| - 2$ min-cut problems are solved. To avoid confusion we call *node* and *connection* to the vertices and arcs of D^k_{flow} and leave vertex and arc for those the original digraph D.

Let $k \in V \setminus \{v_s, v_e\}$ be a customer. We assume k is fixed in the remaining part of the proof because the algorithm must be executed once for each value of k. Let $(\bar{x}, \bar{y})$ be a fractional solution of CTTBF, F^* be the most violated inequality (27). Finally, let $\Delta_k(F) = \sum_{ikm \in F \cup \hat{\delta}^+(F)} x^m_{ik} - \sum_{ijm \in \hat{\delta}^+(F)} x^m_{ij}$ be the violation of the inequality (27) defined by (F, k) where $F \subseteq \{ijm \in \hat{A} : j \neq v_e\}$.

We define the capacitated directed network $D^k_{flow} = \langle V^k_{flow}, A^k_{flow}, s, t, c\rangle$ as the network with two nodes $w^-_e \in V^k_{flow}$ and $w^+_e \in V^k_{flow}$, the in-node of e and out-node of e respectively, for each arc $e \in \hat{A}$, one source $s \in V^k_{flow}$ and one sink $t \in V^k_{flow}$. Additionally, for each arc $e \in \hat{A}$ there is a connection $w^-_e \to w^+_e$. Also, there is a connection $s \to w^-_{ikm}$ from the source s to every arc $ikm \in \hat{A}$ arriving to k, and another connection $w^+_{iv_em} \to t$ from every arc $iv_em \in \hat{A}$ arriving to v_e to the sink t.

$$V^k_{flow} = \{w^+_e : e \in \hat{A}\} \cup \{w^-_e : e \in \hat{A}\} \cup \{s, t\}, \tag{31}$$

$$\begin{aligned} A^k_{flow} = {} & \{s \to w^-_{ikm} : ikm \in \hat{A}\} \cup \{w^+_{iv_em} \to t : iv_em \in \hat{A}\} \\ & \cup \{w^-_e \to w^+_e : e \in \hat{A}\} \cup \{w^+_e \to w^-_f : f \in \hat{\delta}^+(e), e \in \hat{A}\}. \end{aligned} \tag{32}$$

We set the capacity of connections $w^-_e \to w^+_e$ as the flow in $\bar{x}$ for arc e, and infinity for all other connections.

$$c(v_1 \to v_2) = \begin{cases} \bar{x}_e & \text{if } v_1 = w^-_e, v_2 = w^+_e \text{ for some } e \in \hat{A} \\ \infty & \text{otherwise} \end{cases} \tag{33}$$

Next, we prove that the minimum capacity cut S^* of D^k_{flow} is somehow related to the most violated inequality F^*. In particular, it holds that $c(S^*) = \bar{y}_k - \Delta_k(F^*)$.

The proof is divided in two parts. First, we exhibit a cut with capacity at most $\bar{y}_k - \Delta_k(F^*)$, which implies that $c(S^*) \leq \bar{y}_k - \Delta_k(F^*)$. Then we conclude our proof by showing that given the minimum cut S^* it is possible to construct a valid set of arcs $F(S^*)$ such that $\bar{y}_k - \Delta_k(F(S^*)) = c(S^*)$, which in turn implies that $c(S^*) \geq \bar{y}_k - \Delta_k(F^*)$.

First, let $S' = \{w^-_e \to w^+_e : e \in \hat{\delta}^+(F^*)\} \cup \{w^-_{ikm} \to w^+_{ikm} : ikm \notin F^* \cup \hat{\delta}^+(F^*)\}$ be a set of connections of D^k_{flow}. We will show that the capacity of S' is at most $\bar{y}_k - \Delta_k(F^*)$ and that it contains a cut of D^k_{flow}, which has at most that capacity. By definition, the capacity of S' is the sum of the capacities of its connections, therefore its capacity is given by

$$\begin{aligned} c(S') &= \sum_{e \in \hat{\delta}^+(F^*)} c(w^-_e \to w^+_e) + \sum_{ikm \notin F^* \cup \hat{\delta}^+(F^*)} c(w^-_{ikm} \to w^+_{ikm}) \\ &= \sum_{e \in \hat{\delta}^+(F^*)} \bar{x}_e + \sum_{ikm \notin F^* \cup \hat{\delta}^+(F^*)} \bar{x}_{ikm} \\ &= \sum_{e \in \hat{\delta}^+(F^*)} \bar{x}_e + \bar{y}_k - \sum_{ikm \in F^* \cup \hat{\delta}^+(F^*)} \bar{x}_{ikm} \\ &= \bar{y}_k - \Delta_k(F^*). \end{aligned} \tag{34}$$

We proved that $c(S') = \bar{y}_k - \Delta_k(F^*)$, now we proceed to show that $D^k_{flow} \setminus S'$ disconnects s from t. Let $s \to w^-_{e_1} \to w^+_{e_1} \to w^-_{e_2} \to w^+_{e_2} \to \cdots \to w^-_{e_r} \to w^+_{e_r} \to t$ be an arbitrary path from s

to t. We know e_1 must be an entering arc ikm to k as it is connected to s, if $ikm \notin F^*$ then $w^-_{e_1} \to w^+_{e_1} \in S'$ by definition. Suppose $ikm \in F^*$, then we also know that e_r is an arc $i'v_em'$ arriving to v_e as it is connected to t. Also, we know $i'v_em' \notin F^*$ by definition of F^*. Then we have a sequence of arcs $e_1, \cdots, e_r$ such that $e_1 \in F^*$ and $e_r \notin F^*$ and each one is a successor of the previous one, then some arc e_l in the middle must be in $\hat{\delta}^+(F)$ and therefore $w^-_{e_l} \to w^+_{e_l} \in S'$. It is easy to see that S' must contain a an $s-t$ cut of D^k_{flow} with capacity at most $\bar{y}_k - \Delta_k(F^*)$ then we conclude that $c(S^*) \leq \bar{y}_k - \Delta_k(F^*)$.

Next we show that $c(S^*) = \bar{y}_k - \Delta_k(F(S^*))$ where $F(S^*) = \{ijm \in \hat{A} : w^+_{ijm} \in R\}$. In order to do so, we use the result of equation (34) and prove that a connection $w^-_e \to w^+_e \in S^*$ iif $e \in \hat{\delta}^+(F(S^*)) \cup \{ikm \notin F(S^*) \cup \hat{\delta}^+(F(S^*))\}$.

First, note that $w^-_{ikm} \in R$ for all $ikm \in \hat{A}$ and $w^+_{iv_em} \in T$ for all $iv_em \in \hat{A}$, otherwise at least one infinite arc $s \to w^-_{ikm}$ or $w^+_{iv_em} \to t$ would belong to S^*. This implies that $F(S^*)$ does not contain any arc arriving to the end depot and defines a valid inequality (27).

- **Case 1:** $e \in \{ikm \notin F(S^*)\} \Rightarrow w^-_e \to w^+_e \in S^*$. Since $e \notin F(S^*)$ then $w^+_e \in T$. Also, as e is an arc entering vertex k we have that $w^-_e \in R$.
- **Case 2:** $e \in \hat{\delta}^+(F(S^*)) \Rightarrow w^-_e \to w^+_e \in S^*$. Since $e \in \hat{\delta}^+(F(S^*))$ we have that $e \notin F(S^*)$ and as a result $w^+_e \in T$. Also, there must be a predecessor $f \in \hat{A}$ of e which belongs to $F(S^*))$, implying that $w^+_f \in R$. Since connection $w^+_f \to w^-_e$ has infinite capacity $w^-_e \in R$.
- **Case 3:** $w^-_e \to w^+_e \in S^* \Rightarrow e \in \hat{\delta}^+(F(S^*)) \cup \{ikm \notin F(S^*)\}$. Since $w^+_e \in T$, we know that $e \notin F(S^*)$, therefore the case when e enters vertex k is trivially satisfied. When e does not enter vertex k, we know that since R is connected there must exist a predecessor $w^+_f \in V^k_{flow}$ of w^-_e which belongs to R. Therefore, $f \in F(S^*)$ and since $e \notin F(S^*)$ we have that $e \in \hat{\delta}^+(F(S^*))$.

Since $c(S^*) \leq \bar{y}_k - \Delta_k(F^*)$ and $c(S^*) = \bar{y}_k - \Delta_k(F(S^*))$, we conclude that $F(S^*) = F^*$. Finally, the complexity of the algorithm is the complexity of solving $|V|-2$ min-cut problems in networks of $\mathcal{O}(|\hat{A}|)$ vertices and therefore the worst case has a complexity of $\mathcal{O}(|V||\hat{A}|^3)$. □