TU WIEN

! ac ||||| ALGORITHMS AND
COMPLEXITY GROUP

Exact Approaches for Network Design Problems with Relays

Markus Leitner, Ivana Ljubić, Martin Riedler, and Mario Ruthmair
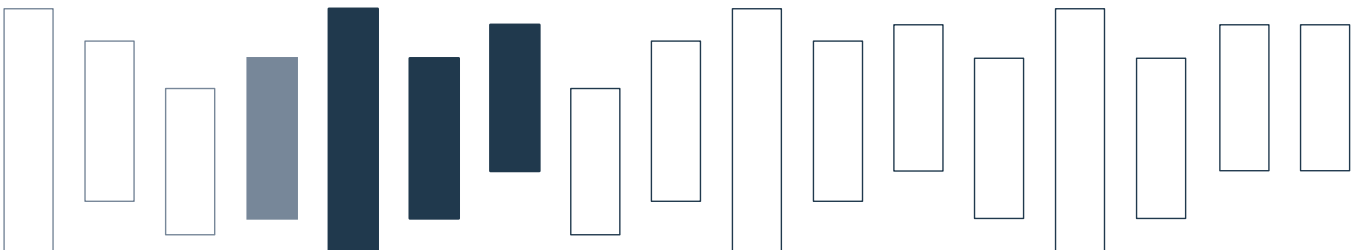
Exact Approaches for Network Design Problems with Relays

Markus Leitner

Department of Statistics and Operations Research, University of Vienna, Vienna, Austria, markus.leitner@univie.ac.at

Ivana Ljubić

ESSEC Business School of Paris, Cergy-Pontoise, France, ljubic@essec.edu

Martin Riedler

Institute of Logic and Computation, TU Wien, Vienna, Austria, riedler@ac.tuwien.ac.at

Mario Ruthmair

Department of Statistics and Operations Research, University of Vienna, Vienna, Austria, mario.ruthmair@univie.ac.at

March 2018

Abstract

In this article we consider the network design problem with relays (NDPR) which gives answers to some important strategic design questions in telecommunication network design. Given a family of origin-destination pairs and a set of existing links these questions are: (1) What are the optimal locations for signal regeneration devices (relays) and how many of them are needed? (2) Could the available infrastructure be enhanced by installing additional links in order to reduce the travel distance and therefore reduce the number of necessary relays?

In contrast to previous work on the NDPR which mainly focused on heuristic approaches, we discuss exact methods based on different mixed integer linear programming formulations for the problem. We develop branch-and-price and branch-price-and-cut algorithms that build upon models with an exponential number of variables (and constraints). In an extensive computational study, we analyze the performance of these approaches for instances that reflect different real-world settings. Finally, we also point out the relevance of the NDPR in the context of electric mobility.

Key words: Network Design with Relays, Telecommunications, Integer Programming, Branch-and-Price

1 Introduction

The network design problem with relays (NDPR) was introduced by Cabral et al. (2007) for modeling the design of telecommunication networks when the maximum distance a commodity (i.e., signal) can travel is bounded from above by some threshold and when distances exceeding this limit can be covered by locating special, commodity regenerating equipment (*relays*) at intermediate locations. Well known applications in communication networks arise from signal deterioration and thus there is the need to regenerate them after some maximum distance by using rather expensive devices (e.g., repeaters), see, e.g., Cabral et al. (2007). Without regeneration, the transmitted information might be falsified or the signal might be lost. As regeneration devices are usually expensive the goal is to use as few devices as possible (see Chen et al. 2010). As an alternative to placing relays the distance along certain connections may also be reduced by installing additional edges.

Problem Definition.

The network design problem with relays (NDPR) is defined on an undirected graph $G = (V, E, c, w, d)$ with relay costs $c \colon V \to \mathbb{N}_{>0}$, edge costs $w \colon E \to \mathbb{N}_{\geq 0}$ and edge lengths $d \colon E \to \mathbb{N}_{\geq 0}$.

(a) Instance	(b) Optimal Solution

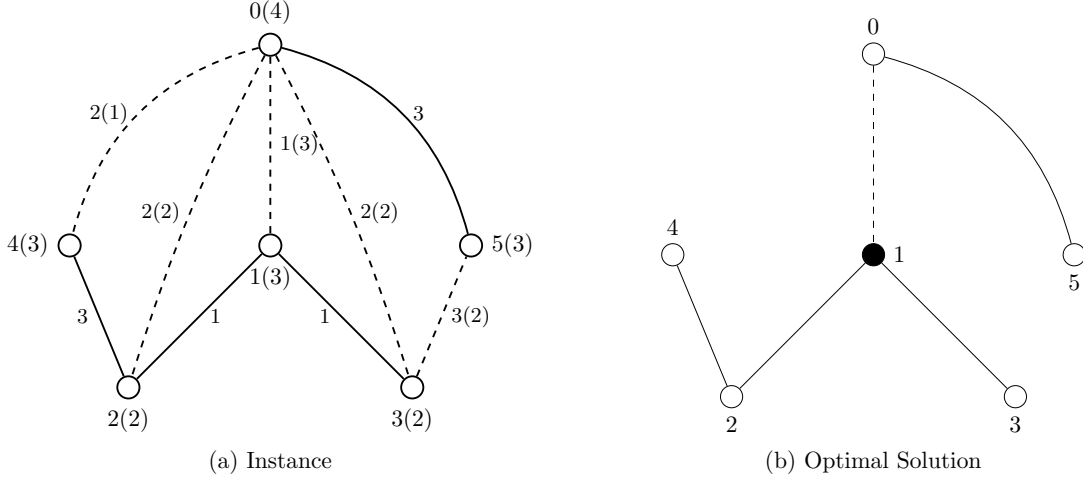Figure 1: An NDPR instance and its optimal solution for $d_{\max} = 4$ and $\mathcal{K} = \{(0,3), (0,4), (2,5), (3,4)\}$ in which a relay is installed at node 1 and the augmenting edge $\{0,1\}$ is selected. Nodes are labeled by node index and relay installation costs in parentheses. Edges are labeled by their lengths and installation costs in parentheses (for augmenting edges). Solid lines indicate free edges and dashed lines augmenting edges. Black nodes mark the selected relays in the solution.

The edge set E is the disjoint union of the set of free (e.g., existing) edges $E^0 = \{e \mid w(e) = 0\}$ and the set of augmenting edges $E^* = \{e \mid w(e) > 0\}$. Furthermore, $\mathcal{K} \subseteq V \times V$ is the set of commodities. Parameter $d_{\max} \in \mathbb{N}_{>0}$ defines the maximum distance a commodity can traverse without regeneration.

The NDPR consists of selecting augmenting edges $\hat{E} \subseteq E^*$ to install and nodes $\hat{V} \subseteq V$ where relays are to be placed minimizing the resulting costs $\sum_{i \in \hat{V}} c_i + \sum_{e \in \hat{E}} w_e$. A solution is feasible iff all commodity pairs from $\mathcal{K}$ can communicate using the edges in $\hat{E} \cup E^0$ and relays at nodes $\hat{V}$. Thereby, two distinct nodes $s, t \in V$ can communicate if there exists a walk $W = (s = v_0, v_1, v_2, \ldots, v_k = t)$, $v_i \in V$, $0 \leq i \leq k$, that does not contain a subwalk $W' = (v_l, v_{l+1}, \ldots, v_{l+m})$, $0 \leq l \leq k-1$, $m \geq 0$, $l+m \leq k$, whose length $d(W') = \sum_{j=l}^{l+m-1} d_{j,j+1}$ is greater than $d_{\max}$ and which does not contain a relay at an intermediate node, i.e., $v_j \notin \hat{V}$ for $l < j < l+m$. A walk satisfying these conditions is called a feasible walk and a feasible path is defined analogously.

An example of an NDPR instance together with an optimal solution is given in Figure 1.

Our Contribution and Structure of the Article.

Available literature for the NDPR mainly deals with heuristic approaches (see, e.g., Cabral et al. (2007), Kulturel-Konak and Konak (2008), Konak (2012)). On the contrary, this article provides a comprehensive computational study of mixed integer linear programming (MILP) models and underlying exact algorithms for the NDPR. A common characteristic of the presented MILP models is that they require an exponential number of variables representing paths in a so-called communication graph. The communication graph contains the same nodes as the original graph G and an edge between every two nodes that can be connected via a feasible path without installing relays.

For deriving computationally viable optimization tools for the proposed MILP models, we implemented two branch-price-and-cut algorithms. Their performance is assessed on a large set of benchmark instances—some of them taken from the available literature, and some newly generated ones.

The article is organized as follows. In the remainder of this section we summarize our notation and provide an overview of the related literature. In Section 2 we prove some structural

2

properties of feasible/optimal solutions that will help us to create tighter formulations. In Section 3 we discuss transformations of the input graph and associated MILP formulations. Section 4 provides the details of our algorithmic framework followed by the presentation of our computational results on a diverse family of benchmark instances in Section 5. Finally, we discuss challenges and open questions for further research in Section 6 where we also summarize our main conclusions.

1.1 Notation and Assumptions

To ease notation in many of the results and formulations introduced in the following, we will consider the node pairs (i.e., commodities) to be directed pairs $(u, v) \in \mathcal{K}$. This assumption is without loss of generality, since there are no costs nor capacities associated with the routing decisions in the NDPR. Furthermore, we define the set of *sources* $\mathcal{K}^{S} = \{u \mid \exists v \in V \text{ such that } (u, v) \in \mathcal{K}\}$ and the set of targets $\mathcal{K}_{u}^{T} = \{v \mid (u, v) \in \mathcal{K}\}$ that have to be reached by source $u \in \mathcal{K}^{S}$. Similarly, the set of all *targets* is defined as $\mathcal{K}^{T} = \{v \mid \exists u \in V \text{ such that } (u, v) \in \mathcal{K}\} = \bigcup_{u \in \mathcal{K}^{S}} \mathcal{K}_{u}^{T}$.

Let $\delta(S) = \{\{i, j\} \in E \mid i \in S, j \in V \setminus S\}$ denote the set of edges incident to $S \subseteq V$ in the undirected graph $G = (V, E)$. For a directed graph $G' = (V', A')$ and node subset $S \subseteq V'$, we define $\delta^{+}(S) = \{(i, j) \in A' \mid i \in S, j \in V \setminus S\}$ and $\delta^{-}(S) = \{(i, j) \in A' \mid i \in V \setminus S, j \in S\}$ as the set of outgoing and incoming arcs, respectively.

Finally, observe that edges $e \in E$ such that $d_e > d_{\max}$ as well as commodities $(u, v) \in \mathcal{K}$ for which G contains a feasible path between the endpoints, using free edges from E^0 only and no relays, can be removed in a preprocessing step. Thus, we assume without loss of generality that neither of them exists in the given input.

1.2 Related Work and Applications in Transportation

The NDPR was introduced by Cabral et al. (2007) who showed that the problem is NP-hard and described heuristic approaches intended to solve large instances. Furthermore, a MILP formulation of the problem based on an exponential number of variables is described. Contrary to our path variables derived in the communication graph (cf. Section 3.1), Cabral et al. (2007) use variables representing entire walks between the commodities in the original graph (including the placement of relays). Their formulation is used to derive lower bounds by means of column generation. At the same time, the subset of generated columns is used to compute heuristic solutions in a subsequent branch-and-bound phase. Computational results are discussed for instances with up to 62 nodes, 103 edges and 10 commodities.

A hybrid metaheuristic combining a genetic algorithm with local search has been proposed by Kulturel-Konak and Konak (2008) whereas an improved genetic algorithm is given in Konak (2012). In the latter article, Konak also introduces a variant of the MILP formulation by Cabral et al. (2007) by using separate variables to represent walks and relay placements. However, no computational studies concerning this model were conducted. Instead, some observations regarding the set covering constraints introduced in this formulation are used in the design of the proposed genetic algorithm. Computational results are given on instances with up to 160 nodes and 3624 edges and 10 commodities. Lin et al. (2014) proposed a tabu search approach for the NDPR. Their solution method computes solutions of almost as good quality as the genetic algorithm from Konak (2012) but requires less computing time. Very recently, Xiao and Konak (2017) presented a variable neighborhood search for the NDPR that is combined with an exact algorithm for the relay placement. Independently from our work an alternative branch-and-price approach was developed in Yıldız et al. (2018). The authors propose MILP formulations on a so-called virtual network with an exponential number of edges. Their computational study mainly focuses on a tree formulation specifically designed for the single source case, i.e., $|\mathcal{K}^{S}| = 1$.

A related version of the NDPR, defined on a directed graph and called the directed network design problem with relays (DNDPR), has been introduced in Li et al. (2012) where a compact MILP model and a branch-and-price algorithm have been proposed. The problem definition is however slightly different from the NDPR: in the DNDPR only simple paths are allowed for connecting commodity pairs, whereas the NDPR also allows using walks. In fact, allowing multiple node visits in general makes NDPR solutions significantly less expensive than DNDPR solutions, see Section 2 for further details. The compact formulation given in Li et al. (2012)

3

exploits the fact that, along a path connecting a commodity pair, each node can be visited at most once. More precisely, it uses a single variable per node to record the distance from the source at which it is reached. Consequently, this formulation cannot be used for solving the NDPR. Their branch-and-price algorithm, which is very similar to the one considered by Cabral et al. (2007), is in general capable of allowing cycles but in the computational experiments the authors only consider the acyclic case. Finally, Kabadurmus and Smith (2015) study an extension of the NDPR which considers survivability, edge capacities, and allows for k-splitting of the routes of a commodity.

Application in Transportation

Yıldız and Karaşan (2015) provide a detailed survey on applications of relay placement problems in transportation. In some of these applications dealing with hub location issues, hubs are interpreted as relays, i.e., the location of hubs naturally corresponds to the placement of relays. The underlying problems are concerned with identifying physical locations of hubs which may serve as places for the exchange of drivers, trucks and trailers, or as stations where drivers can rest (cf. Üster and Kewcharoenwong 2011, Campbell and O'Kelly 2012). Moreover, hubs can be used for switching transportation means or simply for storing the consignment to be picked up by other drivers (see Üster and Maheshwari 2007). Since certain road sections might be more costly to use (e.g., if additional tolls need to be paid) or certain possibilities to extend the road network might exist, the consideration of edge selection is relevant as well. The placement of refueling stations for alternative-fuel vehicles is another important area where the NDPR arises as a subproblem. The distance constraints considered in the NDPR are well motivated by the typical range restrictions of such vehicles, see e.g., Schneider et al. (2014) and the survey by Pelletier et al. (2016).

Additionally, we want to point out the relation to the so called minimum cost path problem for plug-in hybrid electric vehicles (PHEV) considered by Arslan et al. (2015). In this problem a PHEV needs to travel from an origin to a destination node using gasoline refueling and electric charging stations while minimizing refueling, charging and traveling costs. The authors solve the problem with a mixed integer quadratically constrained formulation. The considered problem is very similar to the NDPR. The charging stations can be viewed as some kind of relays and the vehicle corresponds to a single commodity. The primary difference lies in the fact that refueling and charging stations need to be modeled as two different types of relays which introduces additional complexity. Nevertheless, solution techniques introduced in this paper for the NDPR (in particular, the modeling of path segments between two consecutive charging stations and the generation of the communication graph) might also be relevant for solving the PHEV problem. The placement of refueling stations has also been considered in Capar et al. (2013) and Yıldız et al. (2016) where the goal is to select locations for the refueling stations such that the total volume of the refueled demand is maximized for a given set of origin-destination pairs. Yıldız et al. (2016) solve the problem with a branch-and-price approach that uses an exponential number of path variables to model route feasibility.

Relation to Regenerator Location/Placement Problems

The regenerator location problem (RLP) introduced by Chen et al. (2010) is closely related to the NDPR but focuses on the placement of regenerators (relays) and no additional edges can be purchased/installed in the network. More precisely, the RLP is a special case of the NDPR for $E^* = \emptyset$ and $\mathcal{K} = \{(u, v) \mid u, v \in V \; u < v\}$, i.e., it is assumed that all edges have zero costs and all node pairs need to communicate. As the RLP is equivalent to the maximum leaf spanning tree problem (MLSTP) and the minimum connected dominating set problem (MCDSP) (see, e.g., Lucena et al. 2010, Gendron et al. 2014) the NDPR generalizes these problems as well. Chen et al. (2010) provide several MILP formulations for the RLP which unfortunately cannot be directly used to solve the NDPR since they are not capable of selecting augmenting edges. We will, however, use the concept of a communication graph provided in Chen et al. (2010) for solving the NDPR, cf. Section 3.1 for a detailed description. Further exact approaches for the RLP have been developed by Rahman et al. (2015) who propose compact formulations and

branch-and-cut algorithms. In a recent contribution by Yıldız and Karaşan (2015), survivability requirements are added to the RLP.

Chen et al. (2015) introduced the so called generalized regenerator location problem (GRLP) which extends the RLP by considering node sets $S \subseteq V$ of potential relay locations and $T \subseteq V$ of terminal nodes which need to be able to communicate with each other, i.e., $\mathcal{K} = \{(i,j) \mid i,j \in T, i < j\}$. Again, the proposed models for the GRLP cannot be applied to the NDPR. On the other hand, the NDPR is able to model the GRLP by assigning infinite costs to the nodes in $V \setminus S$ or by adding constraints that prohibit that the nodes in $V \setminus S$ are chosen as relays. The latter can be easily done in all formulations that will be introduced in the following.

Another recent contribution by Yıldız and Karaşan (2017) considers several practical extensions like routing, bandwidth allocation, and modulation selection. The proposed flow model uses an exponential number of variables and is solved by a branch-and-price approach. Finally, in a broader sense, the NDPR is also related to optimization problems dealing with wavelength division multiplexing, but, in contrast to the NDPR, the routing of an optical signal has to be optimized at two layers, the logical and the physical one (see, e.g., Raghavan and Stanojević 2011, Sung and Song 2003, for further details).

2 Solution Properties

In this section we introduce and prove certain structural properties of optimal NDPR solutions that will be used to derive MILP formulations in the next section. Recall that a solution consists of a subset $\hat{E} \subseteq E^*$ of augmenting edges (together with the free edges E^0), and a subset $\hat{V} \subseteq V$ of nodes where relays have to be installed.

As mentioned above, commodity pairs $(u,v) \in \mathcal{K}$ are assumed to be ordered, so that routing a signal from u to v is determined by a feasible (directed) u,v-walk embedded in the subgraph induced by $\hat{E} \cup E^0$, and using a subset of relays from the set $\hat{V}$. Hence, even though the solution corresponds to an undirected graph (along with the placement of relays), routing decisions are given by directed walks that are incorporated in the solution graph. In our terminology, a feasible walk corresponds to a directed subgraph embedded in the undirected solution graph, so that each edge can be traversed in both directions. The number of visits of a node in a feasible walk is equal to the in-degree of this node in the associated directed graph.

To better illustrate this interplay between the undirected solution graph and the directed subgraph associated with the routing decisions, let us consider the instance given in Figure 2a and assume $d_{\max} = 4$ and $\mathcal{K} = \{(0,3)\}$. The unique optimal solution which is visualized in Figure 2b selects all edges and places a relay at node 2. The routing of a signal from node 0 to node 3 is given by a directed walk $(0, 1, 2, 1, 3)$ embedded in this solution. This walk traverses the edge $\{1, 2\}$ twice and visits node 1 twice forming a cycle, cf. Figure 2c. Notice that the definition of the NDPR does not forbid such cycles. As a matter of fact, enforcing that each commodity is connected by a (simple) path might significantly increase the cost of a solution. In the considered example a relay would need to be placed at node 1 instead of at 2 and edge $\{1, 2\}$ would not be traversed. Consequently, the solution cost would increase from 8 to 12.

In the following, we focus on structural properties dealing with routing decisions in an optimal NDPR solution. We first prove two properties that are concerned with the maximum number of node visits for connecting a single commodity before we show that analogous results also hold when simultaneously considering all commodities with a common source.

Property 1. *In every optimal solution there exists for every pair $(u,v) \in \mathcal{K}$ a feasible walk from u to v visiting each relay at most once.*

Proof. Proof. Let $W = (u, w_0, i, w_1, i, \ldots, w_{n-1}, i, w_n, v)$ be a feasible u,v-walk in an optimal solution that consists of subwalks $\{w_0, w_1, \ldots, w_n\}$ with $i \in V$, $i \neq u, v$, cf. Figure 3. If i is a relay node, then walk $W' = (u, w_0, i, w_n, v)$ visiting node i only once clearly is a feasible u,v-walk as well. By repeating this argument, we will end up with a feasible u,v-walk in which each relay node appears at most once. □

Property 2. *In every optimal solution there exists for every pair $(u,v) \in \mathcal{K}$ a feasible walk from u to v visiting each non-relay node at most twice.*
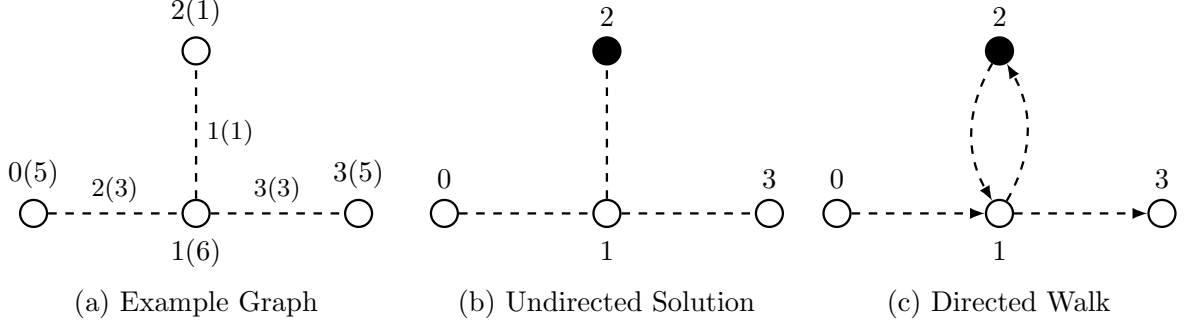
Figure 2: Example instance with a cyclic solution for $d_{\max} = 4$ and $\mathcal{K} = \{(0,3)\}$. Nodes are labeled by node index and relay installation costs in parentheses. Augmenting edges are labeled by their lengths and installations costs in parentheses. Dashed lines indicate augmenting edges. Black nodes mark the selected relays in the solution.
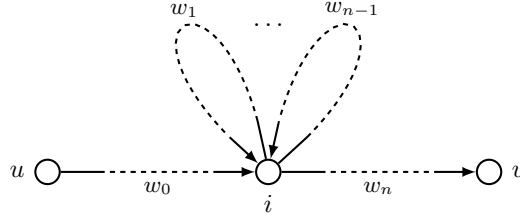

Figure 3: Path from u to v visiting node i n times.

Proof. Proof. Let $i \in V$ be the non-relay node closest to u which is visited $n > 2$ times in a feasible u, v-walk. Let the feasible u, v-walk be represented as $W = (u, w_0, i, w_1, i, \ldots, w_{n-1}, i, w_n, v)$ with subwalks $\{w_0, w_1, \ldots, w_n\}$ that do not visit i, cf. Figure 3. Clearly, each subwalk w_ℓ $(1 \leq \ell \leq n-1)$ that contains no relays, can be deleted from W without violating feasibility. Hence, let us assume that at least one relay is traversed in w_ℓ, for all $1 \leq \ell < n$, and let $r \in V$ be the relay node with minimum distance from/to i in any of the subwalks $w_1, \ldots w_n$ (considered undirected). Let $(i, v_0, v_1, \ldots, v_l = r)$, $v_k \neq i$, $0 \leq k \leq l$, be the corresponding path from i to r with minimum length. Then, $W' = (u, w_0, i, v_0, v_1, \ldots, v_l = r, v_{l-1}, \ldots, v_0, i, w_n, v)$ is a feasible u, v-walk visiting i exactly two times. By repeating this procedure for each non-relay node which is visited more than twice, we can construct a feasible walk with the desired property. $\square$

Definition 1. *A feasible walk w connecting a commodity pair $(u,v) \in \mathcal{K}$ is called* non-redundant *if it does not contain a feasible u, v-subwalk w', $w' \neq w$.*

Remark 1. *Every non-redundant walk satisfies Properties 1 and 2. If a non-relay node is visited twice in a non-redundant walk, then the second visit occurs in terms of a cycle that visits a relay. Each feasible walk can be converted into a non-redundant one using the reduction techniques from the proofs of Properties 1 and 2.*

Theorem 1. *Given a source $u \in \mathcal{K}^{\mathrm{S}}$, in every optimal solution one can embed a digraph rooted at u containing a feasible walk from u to every target $v \in \mathcal{K}_u^{\mathrm{T}}$. In this digraph, each relay has in-degree at most one and each non-relay node has in-degree at most two.*

Proof. Proof. We show the existence of such a digraph $\tilde{G}$ by construction. According to Properties 1 and 2, for each target $v \in \mathcal{K}_u^{\mathrm{T}}$ there exists a feasible walk from u to v with the desired properties. We choose one of the targets $v \in \mathcal{K}_u^{\mathrm{T}}$ and initialize the graph $\tilde{G}$ with arcs determining its feasible walk (satisfying Properties 1 and 2). We then continue to iteratively insert
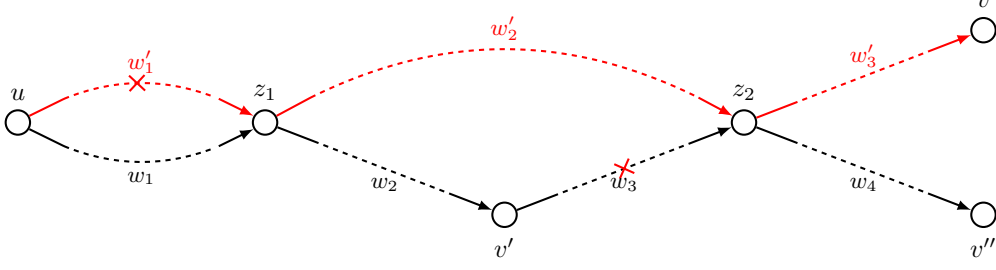
6

Figure 4: A directed graph rooted at u that needs to reach targets $\mathcal{K}_u^{\mathrm{T}} = \{v, v', v''\}$. We illustrate the pruning procedure after adding the red walk to the current digraph $\tilde{G}$ shown in black. The pruning procedure applied to nodes z_2 and z_1 removes subwalks w_3 and w_1', respectively.

non-redundant walks for the remaining targets. Whenever a new walk (associated with a new target) is considered, there are three possibilities: (a) the graph already contains the walk's target, (b) the graph and the walk are node-disjoint (except for the source u), or (c) the graph and the walk share more than one node.

If (a) occurs, we do not modify $\tilde{G}$ and continue with the next target node. Case (b) allows to add the walk to $\tilde{G}$ without violating the desired properties since the in-degrees of nodes already existing in $\tilde{G}$ do not change. The third case (c) is more difficult to handle since simply adding the walk might increase the in-degree for some nodes and thus can destroy the properties we need.

Initially we add the walk to $\tilde{G}$. Let $Z = \{z_1, \ldots, z_n\}$ be the set of nodes on this walk with in-degree greater than one. Each of these nodes might be in conflict with the required properties. To resolve this issue we apply a pruning procedure to each node $z \in Z$.

First, we identify a non-redundant walk in $\tilde{G}$ that reaches z at minimum distance (using as few edges as possible) from the preceding relay or node u. This walk might visit z at most twice using a cycle to a relay to reduce the distance. Then, we delete all incoming arcs of z not contained in this walk. Moreover, we iteratively remove preceding arcs of the already removed ones (not contained in the walk) until we arrive at source node u, a node that is a target, or a node that has an outgoing walk to a target. For an example see Figure 4.

After processing all u, v-walks in this way we can reach all targets of u in $\tilde{G}$. Moreover, walks are added such that all nodes in $\tilde{G}$ are guaranteed to meet the required degree restrictions. $\quad\square$

3 MILP Formulations

The MILP formulations that will be introduced in the following are based on a *communication graph* whose construction from the original graph will be explained next. Afterwards, we will provide two MILP formulations and discuss their properties.

3.1 Communication Graph

Communication graph $G_{\mathrm{C}} = (V, C)$ is defined on the original node set V and its edge set C consists of all node pairs $\{i, j\} \subseteq V$, $i \neq j$, for which at least one path P with length $d(P) \leq d_{\max}$ exists in G, i.e., i and j can communicate using edges from E^* or E^0 without installing relays. Recall that (after preprocessing) every feasible path without relays connecting commodity $(u, v) \in \mathcal{K}$ in G contains at least one edge $e \in E^*$ with positive costs. Since multiple paths can be used for connecting a node pair, it is not clear which one of the potentially exponentially many paths will be used in an optimal solution and thus all of them need to be considered. Our definition of the communication graph extends the one introduced in Chen et al. (2010) and Chen et al. (2015) for solving the (G)RLP. In contrast to their definition, our communication graph considers all edges $e \in E$ and not only those with zero costs.

Figure 5 demonstrates how the existence of common subpaths for several commodities produces a better solution than the one obtained by combining cheapest subpaths of the individual
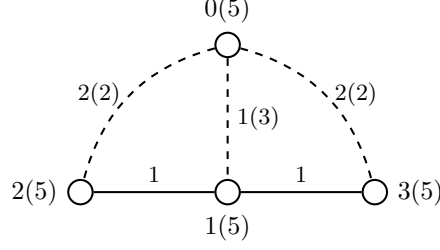
Figure 5: Shortest path connections may be suboptimal due to edge reuse. Paths $(0, 1, 2)$ and $(0, 1, 3)$ are dominated by $(0, 2)$ and $(0, 3)$, respectively. Nevertheless, the optimal solution is $\{\{0, 1\}, \{1, 2\}, \{1, 3\}\}$ ($\mathcal{K} = \{(0, 2), (0, 3)\}$ and $d_{\max} = 3$). Nodes are labeled by node index and relay installation costs in parentheses. Edges are labeled by their lengths and installation costs in parentheses (for augmenting edges). Solid lines indicate free edges and dashed lines augmenting edges.

commodities. Assume $\mathcal{K} = \{(0, 2), (0, 3)\}$ and observe that the cheapest $0, 2$-path is the edge $\{0, 2\}$ and the cheapest $0, 3$-path is edge $\{0, 3\}$, both with a cost of 2 yielding a solution with total cost 4. The union of paths $(0, 1, 2)$ and $(0, 1, 3)$ results, however, in a cheaper solution with total cost 3.

In the following for each pair of distinct nodes $b = \{i, j\}$, let $P_b = \{p \mid p$ is an i, j-path in G, such that $d(p) \leq d_{\max}\}$ be the set of all feasible i, j-paths in G. Furthermore, let $P_b^0 = \{p \mid p$ is an i, j-path in G^0, such that $d(p) \leq d_{\max}\}$ be the set of all such paths in $G^0 = (V, E^0)$, i.e., using only free edges. Then, sets $C^0 = \{b = \{i, j\} \mid P_b^0 \neq \emptyset\}$ and $C^* = \{b = \{i, j\} \mid P_b^0 = \emptyset$ and $P_b \neq \emptyset\}$ define the node pairs that can be connected using only free edges and using at least one augmenting edge, respectively. Thus, the edge set C of communication graph $G_{\mathrm{C}} = (V, C)$ corresponding to $G = (V, E)$ is defined as $C = C^* \cup C^0$, i.e., the set of node pairs that can be connected using edges in $E^* \cup E^0$.

Figure 6 illustrates the stepwise generation of a communication graph for the input graph provided in Figure 6a, $d_{\max} = 4$. After initializing C with the set of free edges E^0, all remaining edges $C^0 \setminus E^0$ corresponding to connections that can be established by using only free edges and no relays are added, cf. the edges $\{1, 4\}$ and $\{2, 3\}$ corresponding to paths $(1, 2, 4)$ and $(2, 1, 3)$, respectively, in Figure 6b. Next, all connections possible through the use of augmenting edges (i.e., either augmenting edges or paths containing at least one augmenting edge) are considered, see the dashed edges in Figure 6c. The corresponding node pairs are connected by dotted lines in Figure 6c. Communication graph G_{C} is finally obtained by removing potentially existing multi-edges, cf. Figure 6d where connections in C^0 and C^* are displayed by solid and dotted lines, respectively.

The following property of the communication graphs is crucial for developing the MILP models shown below.

Property 3. *For every commodity pair $(u, v) \in \mathcal{K}$, a non-redundant u, v-walk in the original graph can be mapped to a simple path in the communication graph with relays placed at all intermediate nodes (if any).*

Proof. Proof. According to Remark 1 there exists a non-redundant (feasible) walk in the original graph visiting every relay at most once. Observe that we can partition the walk in the original graph into maximal feasible subwalks such that none of their intermediate nodes are relays. Communication graph G_{C} contains an edge for every feasible path that is not required to visit any relays. Therefore, the mentioned subwalks can be translated to edges of the communication graph leading to the desired simple path. Conversely, each simple path in the communication graph can be translated to a non-redundant path in the original graph. $\square$

Corollary 1. *For a feasible solution we can identify for each source $u \in \mathcal{K}^{\mathrm{S}}$ and all its targets a tree in the communication graph with relays placed at all intermediate nodes where each leaf*

8

(a) Original Graph

(b) Adding all Connections in C^0

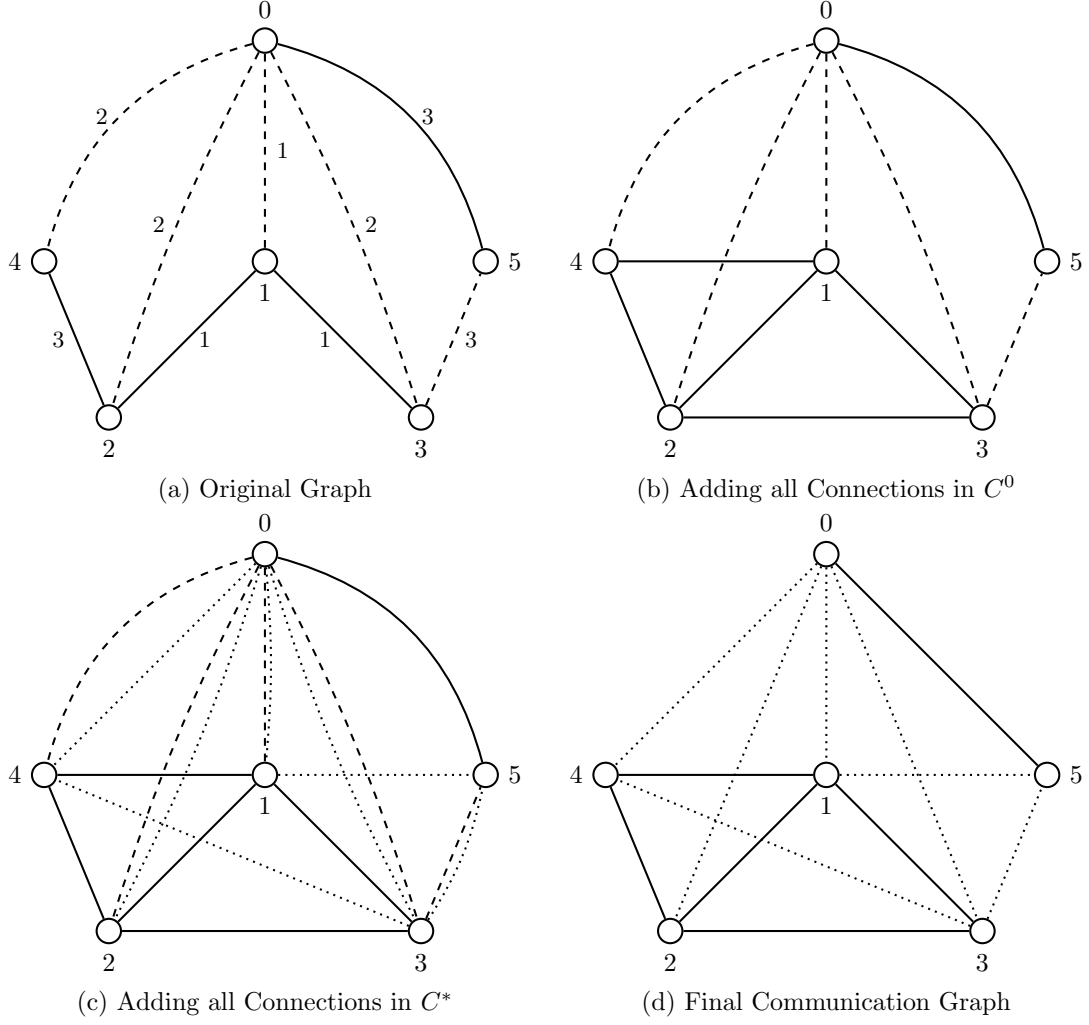(c) Adding all Connections in C^*

(d) Final Communication Graph

Figure 6: Generation of the Communication Graph for $d_{\max} = 4$. Nodes are labeled by node index. Edges are labeled by their lengths. Solid lines indicate free edges, dashed lines augmenting edges, and dotted lines indicate connections from C^*.

9

is a target of u such that each (unique) path to a target corresponds to a feasible walk in the original graph.

Proof. Proof. We use the graph obtained according to the proof of Theorem 1. Then, we partition and translate it in the same way as in the proof of Property 3. ☐

The MILP formulations introduced in the following subsections make use of flows or cut-sets to model feasible paths in the communication graph. In addition, the relation between edges in G_C and (an exponential number of) paths in G is established.

3.2 Multi-Commodity Flow Formulation

We first present a multi-commodity flow formulation on the communication graph (MCF) which uses one set of flow variables for each commodity in $\mathcal{K}$. It utilizes the following design variables defined on G:

$$x_e = \begin{cases} 1, & \text{if } e \text{ is installed in the network} \\ 0, & \text{otherwise} \end{cases} \quad \forall e \in E^*$$

$$y_i = \begin{cases} 1, & \text{if a relay is installed at } i \\ 0, & \text{otherwise} \end{cases} \quad \forall i \in V$$

Using $A(C) = \{(i,j) \mid \{i,j\} \in C\}$ we define flow variables f_{ij}^{uv} for all commodity pairs $(u,v) \in \mathcal{K}$ and each direction of edge $\{i,j\} \in C$, i.e.,

$$f_{ij}^{uv} = \begin{cases} 1, & \text{if the } u,v\text{-path in } G_\mathrm{C} \text{ traverses edge } \{i,j\} \text{ in direction from } i \text{ to } j \\ 0, & \text{otherwise} \end{cases} \quad \forall (i,j) \in A(C)$$

Finally, we use variables λ_b^p that correspond to the paths $p \in P_b$ that have been identified as possible realizations for the connections $b \in C^*$. More precisely:

$$\lambda_b^p = \begin{cases} 1, & \text{if connection } b \text{ is realized by path } p \in P_b \\ 0, & \text{otherwise} \end{cases} \quad \forall b \in C^*$$

Since no augmenting edges need to be purchased when sending flow through edges from C^0, we do not need to consider path variables for $b \in C^0$. The MILP formulation reads as follows:

$$\text{(MCF)} \quad \min \sum_{i \in V} c_i y_i + \sum_{e \in E^*} w_e x_e \tag{1}$$

$$\sum_{(i,j) \in A(C)} f_{ij}^{uv} - \sum_{(j,i) \in A(C)} f_{ji}^{uv} = \begin{cases} 1 & i = u \\ -1 & i = v \\ 0 & i \neq u, i \neq v \end{cases} \quad \forall (u,v) \in \mathcal{K}, \forall i \in V \tag{2}$$

$$-y_i + \sum_{(i,j) \in A(C)} f_{ij}^{uv} \leq 0 \qquad\qquad \forall (u,v) \in \mathcal{K}, \forall i \in V \setminus \{u,v\} \tag{3}$$

$$-(f_{ij}^{uv} + f_{ji}^{uv}) + \sum_{p \in P_b} \lambda_b^p \geq 0 \qquad\qquad \forall (u,v) \in \mathcal{K}, \forall b = \{i,j\} \in C^* \quad (\mu_b^{uv})$$
$$\tag{4}$$

$$x_e - \sum_{p \in P_b : e \in p} \lambda_b^p \geq 0 \qquad\qquad \forall e \in E^*, \forall b \in C^* \quad (\alpha_b^e)$$
$$\tag{5}$$

$$\lambda_b^p \geq 0 \qquad\qquad \forall b \in C^*, p \in P_b \tag{6}$$

$$0 \leq f_{ij}^{uv} \leq 1 \qquad\qquad \forall (u,v) \in \mathcal{K}, \forall (i,j) \in A_\mathrm{C} \tag{7}$$

$$\mathbf{y} \in \{0,1\}^{|V|}, \mathbf{x} \in \{0,1\}^{|E^*|} \tag{8}$$

Constraints (2) ensure that for each commodity $(u, v) \in \mathcal{K}$ one unit of flow is sent from u to v. Every node with outgoing flow that is not the source of the corresponding flow must be a relay node, cf. Property 3. This relation is enforced by inequalities (3). Constraints (4) ensure that flow along a connection $b \in C^*$ is only permitted if at least one of the available path realizations has been selected. Due to Property 3, the solution for each commodity pair $(u, v) \in \mathcal{K}$ will be a path in G_C. Hence, only one arc (i, j) or (j, i) per edge $b = \{i, j\}$ will be selected in each variable set. The last set of inequalities guarantees that for all selected path realizations, the corresponding augmenting edges will be part of the solution. Note that each variable set only considers a single pair $(u, v) \in \mathcal{K}$. Arcs targeting u or leaving v are irrelevant with respect to the flow variables, hence the respective flow variables can be omitted from the formulation.

Pricing Subproblem. Since the number of path variables in formulation (MCF) may be exponentially large we will use column generation for solving its linear programming (LP) relaxation. To formulate the pricing subproblem, let $\mu_b^{uv} \geq 0$ and $\alpha_b^e \geq 0$ be the dual variables associated to constraints (4) and (5), respectively. Then, for each $b \in C^*$, the pricing subproblem (i.e., find a path $p \in P_b$ with negative reduced costs) is defined as

$$\arg\min_{p \in P_b} \left(\sum_{e \in E^* \cap p} \alpha_b^e - \sum_{(u,v) \in \mathcal{K}} \mu_b^{uv} \right)$$

Since the second summation is a constant for a fixed $b \in C^*$, it further reduces to finding

$$\arg\min_{p \in P_b} \sum_{e \in E^* \cap p} \alpha_b^e \qquad \forall b \in C^*$$

which is a weight constrained shortest path problem (WCSPP) defined on the graph $G = (V, E)$ with edge weights $\omega \colon E \to \mathbb{R}_{\geq 0}$, defined as $\omega_e = d_e$, for all $e \in E$ and edge costs $\gamma \colon E \to \mathbb{R}_{\geq 0}$ defined as

$$\gamma_e = \begin{cases} 0, & e \in E^0 \\ \alpha_b^e, & e \in E^* \end{cases} \qquad \forall e \in E$$

Further details regarding pricing will be provided in Section 4.2.

3.3 Cut Formulation

We propose an alternative MILP formulation that is based on a single, *directed communication graph* in which relays are identified by splitting each node i into two copies i_1, i_2. Besides *relay arcs* (i_1, i_2) for each node $i \in V$, we add arcs (j_2, i_1) and (i_2, j_1) for each edge $\{i, j\} \in C$, i.e., all incoming arcs target i_1 and all outgoing arcs emanate from i_2. We obtain graph $G'_C = (V'_C, A'_C)$ with node set $V'_C = \{i_1, i_2 \mid i \in V\}$ and arc set $A'_C = \{(i_2, j_1), (j_2, i_1) \mid \{i, j\} \in C\} \cup A^r_C$ where arcs in $A^r_C = \{(i_1, i_2) \mid i \in V\}$ are used to identify relays. An example of a directed communication graph is shown in Figure 7 where solid arcs represent free connections, dotted arcs connections with augmenting edges and dash-dotted arcs correspond to relays.

The formulation introduced next represents each feasible u, v-walk of a given commodity pair $(u, v) \in \mathcal{K}$ as a directed path from u to v in G'_C with relays placed at all intermediate nodes (cf. Property 3). To this end we utilize binary variables $\mathbf{x}$ for the augmenting edges and path variables $\boldsymbol{\lambda}$, introduced above. In addition, we associate binary variables X_{ij} to the arcs $(i, j) \in A'_C$ of the directed communication graph. Due to the one-to-one correspondence between the arcs in A^r_C and the relays we can use the arc variables directly to identify the relays. The model then reads as follows:

$$\text{(CUT)} \quad \min \sum_{i \in V} c_i X_{(i_1, i_2)} + \sum_{e \in E^*} w_e x_e \tag{9}$$

$$\sum_{a \in \delta^-(W)} X_a \geq 1 \qquad\qquad \begin{aligned} &\forall (u, v) \in \mathcal{K}, \forall W \subset V'_C \\ &v_1 \in W, u_2 \notin W \end{aligned} \tag{10}$$

11

(a) Original Graph
$G = (V, E^0 \cup E^*)$

(b) Communication Graph
$G_C = (V, C)$
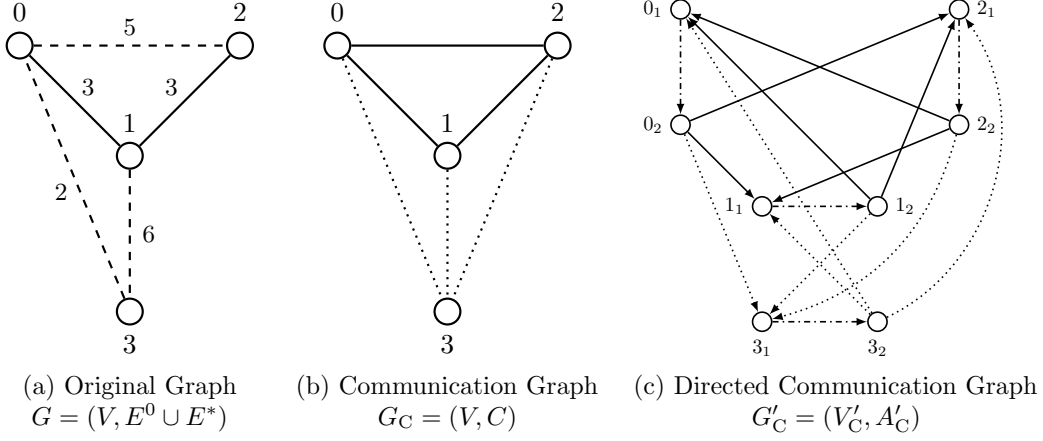
(c) Directed Communication Graph
$G'_C = (V'_C, A'_C)$

Figure 7: Generation of the directed communication graph $G'_C = (V'_C, A'_C)$ for $d_{max} = 7$. Nodes are labeled by node index. Edges are labeled by their lengths. Solid lines indicate free edges, dashed lines augmenting edges, and dotted lines indicate connections from C^*. Dash-dotted arcs correspond to relays.

$$-X_{i_2 j_1} + \sum_{p \in P_b} \lambda_b^p \geq 0 \qquad \forall b = \{i,j\} \in C^* \quad (\mu_b^1) \qquad (11)$$

$$-X_{j_2 i_1} + \sum_{p \in P_b} \lambda_b^p \geq 0 \qquad \forall b = \{i,j\} \in C^* \quad (\mu_b^2) \qquad (12)$$

$$x_e - \sum_{p \in P_b : e \in p} \lambda_b^p \geq 0 \qquad \forall e \in E^*, \forall b \in C^* \quad (\alpha_b^e) \qquad (13)$$

$$\lambda_b^p \geq 0 \qquad \forall b \in C^*, p \in P_b \qquad (14)$$

$$\mathbf{X} \in \{0,1\}^{|A'_C|}, \mathbf{x} \in \{0,1\}^{|E^*|} \qquad (15)$$

We will refer to this model as the *cut formulation on a directed communication graph* (CUT). Cut-set inequalities (10) ensure the existence of a directed path in G'_C from u_2 (the source copy of u) to v_1 (the target copy of v) for each commodity pair $(u,v) \in \mathcal{K}$. By construction, every second arc along this path corresponds to a relay node. Costs for these arcs are included in the objective function. The remaining arcs of type (i_2, j_1) or (j_2, i_1) correspond to paths between i and j, where $b = \{i,j\} \in C^*$. Constraints (11) and (12) ensure that whenever an arc (i_2, j_1) or (j_2, i_1) is used, then at least one corresponding path from $P_{\{i,j\}}$ is selected. Due to the presence of multiple sources, both arcs (i_2, j_1) or (j_2, i_1) can be used in a feasible solution. Finally, constraints (13) ensure that all augmenting edges of the selected paths are included in the solution. Since the number of these constraints is in general exponential, we will separate them dynamically only when violated, see Section 4.3.

Notice that a feasible solution may contain arcs (j_2, i_1) and (i_2, h_1) (for some distinct nodes $i, j, h \in V$), but not necessarily the arc (i_1, i_2). This happens, for example, when node i is both a target and a source, but it is not contained in any connection passing through it (and thus there is no need to install a relay at i). As a consequence, the arcs that correspond to node splitting identify the nodes from V where relays have to be placed, whereas the arcs linking the node copies of i and j map to the paths $p \in P_b$, $b = \{i,j\}$ as in the (MCF) model.

Pricing Subproblem. To formally state the pricing subproblem for variables λ_b^p we associate dual variables $\mu_b^1 \geq 0$ and $\mu_b^2 \geq 0$ to constraints (11) and (12), respectively, and dual variables $\alpha_b^e \geq 0$ to constraints (13). Similar to the previous two cases, for each $b \in C^*$ we need to identify a feasible path with minimum reduced costs

$$\min_{p \in P_b} \left(\sum_{e \in E^* \cap p} \alpha_b^e - \mu_b^1 - \mu_b^2 \right).$$

12

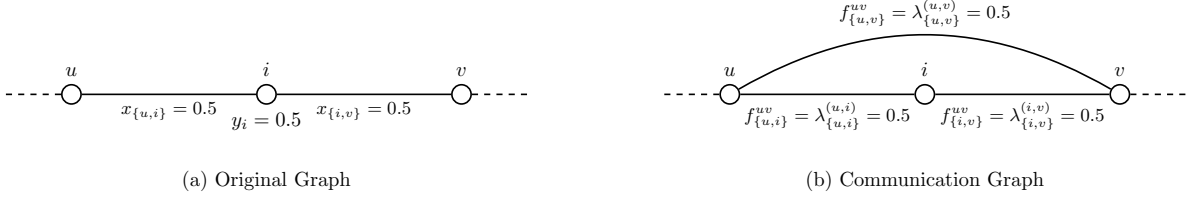(a) Original Graph　　　　　　　　　　　　　　(b) Communication Graph

Figure 8: A partial solution to the LP relaxation of (MCF) that violates connectivity constraints (16) for commodity $(u, v) \in \mathcal{K}$. The total flow in the communication graph from source u to its target v equals 1. However, the minimum u-v cut in the original graph is only 0.5.

Thus, we can solve the pricing subproblem by solving for each $b \in C^*$ a WCSPPs with edge weights set to α_b^e for $e \in E^*$, and to zero otherwise, see Section 4.2 for more details.

Since formulation (CUT) contains an exponential number of variables ($\boldsymbol{\lambda}$) *and* an exponential number of cut-set constraints (10), a column-and-row generation approach is employed to solve its LP relaxation (and a branch-price-and-cut algorithm to find an optimal solution), see, e.g., Barnhart et al. (1998), Desrosiers and Lübbecke (2011). Fortunately, the λ variables are not involved in the connectivity constraints (10). Thus, we can separate these parts so that column generation can be done independently of cut generation, i.e., the added cuts do not influence the structure of the pricing subproblem.

3.4 Valid Inequalities

We now describe several types of valid inequalities that are redundant for the set of feasible solutions but can strengthen the models' LP relaxations.

3.4.1 Connectivity Cuts in the Original Graph

The example given in Figure 8 shows that connectivity constraints (16) can be violated in LP solutions of (MCF) and (CUT), respectively. These constraints ensure that the value of each (undirected) cut separating the source and target of a commodity is at least one. Since cut inequalities including free edges are trivially satisfied, we only consider subsets W inducing cuts without free edges in (16).

$$\sum_{e \in \delta(W)} x_e \geq 1 \qquad \forall W \subset V : \delta(W) \cap E^0 = \emptyset, \exists (u, v) \in \mathcal{K} : u \notin W, v \in W \qquad (16)$$

One can further strengthen the quality of the LP relaxation by replacing undirected cut-set inequalities (16) by their directed counterparts. To this end, we introduce for each root $u \in \mathcal{K}^{\mathrm{S}}$ variables $z_{ij}^u \geq 0$, $\forall \{(i, j) \mid \{i, j\} \in E^*\}$. Variable z_{ij}^u is set to one if one can embed in the original graph a directed path from u to some $v \in \mathcal{K}_u^{\mathrm{T}}$ using arc (i, j). Then, constraints (16) can be enhanced by:

$$\sum_{a \in \delta^-(W)} z_a^u \geq 1 \qquad \forall W \subset V : \delta(W) \cap E^0 = \emptyset, \exists (u, v) \in \mathcal{K} : u \notin W, v \in W \qquad (17)$$

$$z_{ij}^u + z_{ji}^u \leq x_{\{i,j\}} \qquad \forall u \in \mathcal{K}^{\mathrm{S}}, \{i, j\} \in E^* \qquad (18)$$

$$z_{ij}^u, z_{ji}^u \geq 0 \qquad \forall u \in \mathcal{K}^{\mathrm{S}}, \{i, j\} \in E^* \qquad (19)$$

It can easily be seen that the directed connectivity constraints (17) are at least as strong as the undirected ones introduced above. From Figure 9 we conclude that they can be strictly stronger if there exists at least one commodity source with more than one target (in the presence of linking constraints (18)). Note that similar to the undirected variant, we do not add cut-set inequalities for node subsets with incident free arcs.

Since both classes of connectivity constraints are of exponential size we will dynamically separate them, see Section 4.3 for details.
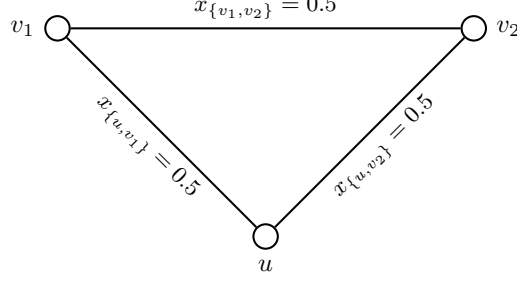
Figure 9: A solution to the LP relaxation for $\mathcal{K} = \{(u, v_1), (u, v_2)\}$ and $w_{\{u,v_1\}} = w_{\{u,v_2\}} = 1$, $w_{\{v_1,v_2\}} = 3$. The solution is optimal with respect to the undirected cut-set inequalities but not with respect to the directed ones which will cut off this LP solution.

Model	Type	B	Graph
(MCF)	BPC	$\sum\limits_{(u,v)\in\mathcal{K}} \mu_b^{uv}$	Communication Graph G_{C}
(CUT)	BPC	$\mu_b^1 + \mu_b^2$	Directed Communication Graph G_{C}'

Table 1: Algorithm Overview. Name (Model), considered decomposition algorithm (Type), model specific pricing subproblem threshold for WCSPP (B), and considered communication graph (Graph).

3.4.2 Relay Constraints.

For the (CUT) model we additionally exploit the fact that a relay has to be placed at some node iff it is an intermediate node along a path within the communication graph. This results in the following constraints that are added to the (CUT) model:

$$\sum_{a\in\delta^-(i_1)} X_a \leq \min(|\mathcal{K}^{\mathrm{S}}|, |\delta(i)|-1) \cdot X_{(i_1,i_2)} \qquad \forall i \notin \mathcal{K}^{\mathrm{S}} \cup \mathcal{K}^{\mathrm{T}} \qquad (20)$$

$$\sum_{a\in\delta^+(i_2)} X_a \leq \min(|\mathcal{K}^{\mathrm{T}}|, |\delta(i)|-1) \cdot X_{(i_1,i_2)} \qquad \forall i \notin \mathcal{K}^{\mathrm{S}} \qquad (21)$$

The first set of constraints makes sure that a relay is installed at each node i which does not belong to any commodity pair, whenever there is an arc entering i_1. Similarly, whenever there is an arc leaving i_2, and node i is not a source, there has to be a relay installed at i.

These constraints benefit from the fact that the (CUT) model uses only one set of variables to ensure connectivity in the communication graph. They are particularly effective if the big-M constants are small, e.g., if only a single source node exists. These constraints turned out to be beneficial not only for strengthening the LP relaxation of the (CUT) model, but also for improving the convergence concerning the dynamically separated inequalities.

4 Algorithmic Framework

We developed branch-price-and-cut (BPC) algorithms for the MILP formulations described in the previous section, see Table 1 for a summary. In the following, after providing some remarks on preprocessing that aims to reduce the size of the problem instances, we present additional details (including separation and pricing procedures) of these algorithms.

4.1 Preprocessing

Recall that at the beginning, we remove all edges $e \in E$ such that $d_e > d_{\max}$. If graph G separates into several connected components, and there exists a pair $(u, v) \in \mathcal{K}$ such that u

14

and v belong to different components, then the instance is clearly infeasible. Note that if the problem admits a feasible solution and it contains more than one connected component, every component describes a separate problem instance that can be solved independently. Next, we identify and remove all pairs in $\mathcal{K}$ that can be connected using solely free edges from E^0 and no relays (this can be easily done by applying shortest path algorithms on (V, E^0)).

For the separation procedures explained in Section 4.3, few commodities with many targets per source are preferable. To this end, we use the fact that commodity pairs can be reordered because in an undirected graph the existence of a feasible u, v-walk implies the existence of a feasible v, u-walk. We heuristically reorder the commodities as follows. First, we compute for each node the number of times it appears in a commodity pair. Then, we iteratively choose the node i with the highest count (breaking ties by node index), reorder the commodity pairs involving i by setting i to be the source, and decrease the counts of all nodes by the number of times i is involved in an associated commodity. The procedure is repeated until the count of every node becomes zero.

4.2 Column Generation

We use column generation to deal with the exponential amount of path variables used in the considered models. The paths we are looking for correspond to the edges of the communication graph. Due to Property 3, these edges correspond to loop-free paths between a pair of nodes i and j , $b = \{i, j\}$. More precisely, for each $b \in C^*$, its reduced costs, denoted by R_b, are calculated as

$$R_b = \min_{p \in P(b)} \sum_{e \in E^* \cap p} \alpha_b^e - B$$

where the value of the constant B depends on the considered formulation and is given in Table 1.

As already mentioned, for each $b \in C^*$, this pricing subproblem is a WCSPP defined on the graph $G = (V, E)$ with non-negative edge weights $\omega_e = d_e$, for all $e \in E$ and non-negative edge costs $\gamma_e = \alpha_b^e$ if $e \in E^*$ and $\gamma_e = 0$, otherwise. The goal is to find a path in G connecting a node pair $b = \{i, j\}$, that minimizes the sum of edge costs and whose weight does not exceed $d_{\max}$.

In our implementation we add one variable corresponding to a least cost path for each $b \in C^*$ in each pricing iteration if it has negative reduced costs. This decision to add at most $|C^*|$ variables in each iteration is based on preliminary experiments indicating that this strategy outperforms other options such as adding only a single variable in each iteration.

Initial set of Columns. We initially add a set of variables ensuring that there exists a feasible solution to the LP relaxation. To this end, we add a variable corresponding to a connection with minimal length for each $b \in C^*$. Such a connection can easily be found with Dijkstra's algorithm (see Dijkstra 1959) using the edge lengths as costs.

Solving the Pricing Subproblems. The WCSPP on a graph with nonnegative edge costs is a weakly NP-hard problem for which fast pseudo-polynomial exact algorithms are available. For the implementation in our models we use the generic resource-constrained shortest path algorithm from the Boost Graph Library (BGL) in version 1.63.0, see BGL. To speed up performance we prevent path expansions leading to costs larger than or equal to B since such paths can never result in negative reduced costs.

4.3 Separation

Depending on the formulation, up to three different families of exponentially-sized constraints can be considered. We separate the three classes in the following order: (i) undirected cut-set inequalities (16) on the original graph, (ii) directed cut-set inequalities (17) on the original graph, and (iii) cut-set inequalities (10) on the communication graph (for the (CUT) model). Only if no violated inequalities of previous classes can be found, we continue with the next class. Violated inequalities of all three classes are identified by maximum flow computations according to the commodity pairs using the algorithm by Cherkassy and Goldberg (1995). Thereby, the

edge (or arc) capacities are set to the current LP solution values plus a small value in order to prefer sparse cuts, i.e., those that contain the fewest edges or arcs, respectively. In case of ties, we always choose a cut that is closest to the target node. To avoid adding too many cuts we only consider inequalities that are violated by a value of at least 0.5.

Connectivity in the Original Graph.

As noted in Section 3.4.1, directed cuts on the original graph are stronger than their undirected counterpart for commodity sources that need to be considered to more than one target. Therefore, if $|\mathcal{K}_u^{\mathrm{T}}| = 1$ for some $(u, v) \in \mathcal{K}$, we only add undirected cut-set inequalities (16) for this commodity pair. Otherwise, we add variables z^u and consider the directed constraints (17). That way, we always ensure the strongest variant of the connectivity inequalities while avoiding unnecessary overhead whenever possible. We note that such a separation strategy also benefits from the aforementioned reordering of the commodity pairs.

For connectivity cuts based on the original graph, we also consider so-called nested cuts (see, e.g., Ljubić et al. (2006)): We set the arc capacities of just added cuts to one and repeat the flow computation to possibly find other violated inequalities. The procedure is continued until no further violations can be detected. Observe that the capacity updates influence the subsequent separation steps. To avoid an unwanted bias we consider the commodity pairs in a random order based on a fixed seed.

Connectivity in the Communication Graph.

Since connectivity constraints (10) on the communication graph are not redundant, their separation is not optional, i.e., they need to be applied at least to all integer solutions encountered during the branch-and-bound procedure. In our implementation, we additionally use these cuts to cut off fractional solutions, applying the maximum-flow procedures described above.

4.4 Initial Pool of Inequalities

We now shortly summarize the set of valid inequalities that are used to initialize our models.

Cuts in the Original Graph.

As mentioned above, both types of the original graph connectivity cuts are dynamically separated. To speed up convergence we add a subset of these inequalities a priori to the model:

$$\sum_{a \in \delta^-(v)} z_a^u = 1 \qquad\qquad \forall (u, v) \in \mathcal{K} : |\mathcal{K}_u^{\mathrm{T}}| > 1, \delta(v) \cap E^0 = \emptyset \qquad (22)$$

$$\sum_{a \in \delta^-(i)} z_a^u \leq \sum_{a \in \delta^+(i)} z_a^u \qquad \forall u \in \mathcal{K}^{\mathrm{S}} : |\mathcal{K}_u^{\mathrm{T}}| > 1, \forall i \in V \setminus (\mathcal{K}^{\mathrm{S}} \cup \mathcal{K}^{\mathrm{T}}), \delta(i) \cap E^0 = \emptyset \qquad (23)$$

If undirected cuts are separated for at least one commodity pair (i.e., $\exists u \in \mathcal{K}^{\mathrm{S}} : |\mathcal{K}_u^{\mathrm{T}}| = 1$), we also add the following inequalities since each commodity source and target node has at least one incident edge.

$$\sum_{e \in \delta(i)} x_e \geq 1 \qquad\qquad \forall i \in V : i \in \mathcal{K}^{\mathrm{S}} \cup \mathcal{K}^{\mathrm{T}}, \delta(i) \cap E^0 = \emptyset \qquad (24)$$

Similarly, we know that relays are never isolated. Thus, we add the following type of inequalities to (MCF):

$$\sum_{e \in \delta(i)} x_e \geq y_i \qquad\qquad \forall i \in V : i \notin \mathcal{K}^{\mathrm{S}} \cup \mathcal{K}^{\mathrm{T}}, \delta(i) \cap E^0 = \emptyset \qquad (25)$$

Equivalent constraints are considered for the (CUT) model by replacing y_i by $X_{(i_1, i_2)}$.

Cuts in the Communication Graph.

We add all constraints from Section 3.4.2 a priori to model (CUT), extended by the following inequalities that ensure that each target has at least one incoming arc and each source has at least one outgoing arc.

$$\sum_{a \in \delta^-(v_1)} X_a \geq 1 \qquad\qquad \forall v \in \mathcal{K}^{\mathrm{T}}$$

$$\sum_{a \in \delta^+(u_2)} X_a \geq 1 \qquad\qquad \forall u \in \mathcal{K}^{\mathrm{S}}$$

4.5 Heuristic

Feasible NDPR solutions and initial upper bounds for our algorithms are obtained by using heuristic (CH1) originally introduced in Cabral et al. (2007). Its basic idea is to iteratively compute a solution by solving the problem for the individual commodities. In each iteration all previously added augmenting edges and relays are assigned zero costs. In our implementation we perform ten runs of (CH1) in which we vary the order in which the commodities are considered (fixed seed random order) and finally adopt the best solution found. Columns required to represent the respective solution are added to the initial formulation.

For each commodity (i.e., in each iteration) we need to solve the minimum cost path problem with relays (MCPPR). Our implementation uses a variant of the pseudo-polynomial dynamic programming algorithm introduced by Laporte and Pascoal (2011). Their algorithm for the MCPPR solves the problem on a directed graph. In the undirected variant we need to make sure that, once an edge has been traversed in one direction, using it in the other direction incurs no additional costs. The simplest way of handling this is to augment the dynamic programming states by a set of already used edges, see Algorithm 4.1 for the adjusted pseudocode. Each state is a tuple of the form $x = (\pi_x^{\mathrm{c}}, \pi_x^{\mathrm{d}}, \xi_x, v_x, E_x^*)$ where π_x^{c} denotes the cost of the current walk, π_x^{d} the distance from the last relay or the starting node along the walk, ξ_x a reference to the preceding state, v_x the final node of the walk, and E_x^* the set of already traversed edges. As suggested in Laporte and Pascoal (2011), the list of states L is ordered according to non-decreasing cost to allow for early termination once a state containing the target node as final node is reached.

After the ten runs of (CH1) we perform a final run for which we set the costs of all relays and edges selected by the best solution to zero. The idea behind this run is to remove possible redundancies with respect to the selected relays and edges. Thereby, it is important to break ties regarding the ordering of L by prioritizing states with smaller π_x^{d}. We denote the modified algorithm by (CH1+).

4.6 Solver Configuration

Our algorithms are implemented in C++ using SCIP 3.2.1 (see Gamrath et al. 2016) as branch-price-and-cut framework and CPLEX 12.6.3 as LP solver. The dual simplex method has been used for solving the LP relaxations as it outperformed other options (primal simplex, barrier) in preliminary experiments. All experiments have been performed in single thread mode with presolving, probing, and the solvers general purpose heuristics turned on. General purpose cutting planes have been deactivated.

5 Computational Study

In this section we first give details on benchmark instances which are then used to compare the performance of the developed branch-price-and-cut algorithms and to demonstrate their advantages and drawbacks.

Input: Graph $G = (V, E, c, w, d)$, $E = E^0 \cup E^*$, pair $(s, t) \in \mathcal{K}$
Data: M_{di} ... cheapest path to i at distance d, L ... set of unexpanded states

1 $L \leftarrow \{(0, 0, NULL, s, \emptyset)\}$
2 forall $d \in \{0, \ldots, d_{\max}\}, i \in V$ **do** $M_{di} \leftarrow NULL$
3 while $L \neq \emptyset$ **do**
4 $\quad$ select first $x \in L$ and remove it from L
5 $\quad$ **forall** $\{v_x, j\} \in \delta(v_x)$ **do**
6 $\quad\quad$ $\hat{d} \leftarrow \pi_x^{\mathrm{d}} + d_{\{v_x, j\}}$ $\qquad\qquad\qquad\qquad\qquad$ // arrival distance at j
7 $\quad\quad$ **if** $\hat{d} \leq d_{\max}$ **then**
8 $\quad\quad\quad$ $\hat{c} \leftarrow \pi_x^{\mathrm{c}}$ $\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ // cost at j
9 $\quad\quad\quad$ **if** $\{v_x, j\} \notin E^0 \cup E_x^*$ **then** $\hat{c} \leftarrow \hat{c} + w_{\{v_x, j\}}$
10 $\quad\quad\quad$ $\hat{E}^* \leftarrow E_x^* \cup (\{v_x, j\} \cap E^*)$ $\qquad$ // traversed augmenting edges at j
11 $\quad\quad\quad$ **if** $j \neq t \wedge (M_{0j} = NULL \vee \hat{c} + c_j < \pi_{M_{0j}}^{\mathrm{c}})$ **then** $\qquad$ // expansion with relay at j
12 $\quad\quad\quad\quad$ $M_{0j} \leftarrow (\hat{c} + c_j, 0, x, j, \hat{E}^*)$
13 $\quad\quad\quad\quad$ $L \leftarrow L \cup \{M_{0j}\}$
14 $\quad\quad\quad$ **end**
15 $\quad\quad\quad$ **if** $M_{\hat{d}j} = NULL \vee \hat{c} < \pi_{M_{\hat{d}j}}^{\mathrm{c}}$ **then** $\qquad$ // expansion without relay at j
16 $\quad\quad\quad\quad$ $M_{\hat{d}j} \leftarrow (\hat{c}, \hat{d}, x, j, \hat{E}^*)$
17 $\quad\quad\quad\quad$ $L \leftarrow L \cup \{M_{\hat{d}j}\}$
18 $\quad\quad\quad$ **end**
19 $\quad\quad$ **end**
20 $\quad$ **end**
21 end
22 return $\arg \min_{x \in M_{dt}: 0 \leq d \leq d_{\max}} \pi_x^{\mathrm{c}}$

Algorithm 4.1: Dynamic programming algorithm for the minimum connected path problem with relays in an undirected graph

5.1 Benchmark Instances

We consider three groups of benchmark instances: 1) instances from Cabral et al. (2007), 2) instances introduced by Konak (2012), and 3) an entirely new set of instances (ARLP), generated to reflect some of the real-world properties not covered by the previous two families.

Cabral instances.

These instances have been introduced by Cabral et al. (2007), see Table 2 for an overview. They are extremely sparse 4-grid graphs in which each node is connected only to its direct vertical and horizontal neighbors. All edges have costs greater than zero (i.e., $E^0 = \emptyset$) and the maximum distance is equal to 70 for all instances. There are 180 instances in this family: for a fixed input graph and the given number of commodities, 10 instances are generated by sampling the set of commodities. All commodities share one node, i.e., we can reorder them such that $|\mathcal{K}^{\mathrm{S}}| = 1$. The number of nodes varies between 20 and 60. Although $|\mathcal{K}| \in \{5, 10\}$, we point out that for some instances the "effective" number of commodities is smaller than specified by the instance. This is due to two reasons. First, some instances contain the same commodity more than once. Second, some instances contain commodities for which the source and target are identical. Since commodities of this type are trivially connected by the empty path, we simply ignore them. Column $|\mathcal{K}|$ in Table 2 reports the average number of effective commodity pairs.

Konak instances.

These instances which were generated by randomly placing and connecting nodes on a grid were originally introduced by Konak (2012). The number of nodes varies between 40 and 160. The length of each edge $\{i, j\}$ is set to the Euclidean distance between i and j while its cost is either set equal to the edge length (type I) or to $d_{\max} - d_{\{i,j\}}$ (type II). The basic instance properties ($|V|$, $|E^0|$, $|E^*|$, $|\mathcal{K}|$, and $d_{\max}$) are shown in Tables 3 and 4. Instances with an identical number

of nodes and the same $d_{\max}$ are based on the same graph and only the number of commodities differs. There are 40 instances in total, 20 of each type. Notice that also for this family of instances, the number of commodities is extremely low ($|\mathcal{K}| \in \{5, 10\}$). In some instances free edges are present but their number is always rather small.

ARLP instances.

This newly generated set of benchmark instances is intended to complement the previous two sets available from the literature. Both, Cabral and Konak instances assume $|\mathcal{K}| \in \{5, 10\}$. On the contrary, ARLP instances aim to simulate applications where many node pairs needs to communicate. We refer to this set as augmented RLP (ARLP) instances since we require all nodes to communicate with each other, as it is the case for the RLP (cf. Section 1.2). In contrast to the RLP, the set of augmenting edges E^* is not empty, and in contrast to the Konak and Cabral instances, a significant number of zero cost edges exists.

The instances have been generated as follows. Nodes are placed randomly on a 100×100 grid and edges with length equal to the Euclidean distance (rounded up) between two nodes are added whenever this distance does not exceed 30. Each edge is chosen to be a free edge with probability 20, 50, or 80 % in instance subsets `20F`, `50F`, and `80F`, respectively. The costs w_{ij} of augmenting edges $\{i, j\}$ are chosen randomly according to a normal distribution with parameters $\mu = d_{ij}, \sigma = 5$ (rounded up). Relay costs are chosen randomly according to the normal distribution $\mu = 10 \cdot \bar{w}, \sigma = 20$ (rounded up) where $\bar{w}$ denotes the average costs of augmenting edges. Finally, $d_{\max} = 50$ for all instances and $\mathcal{K}$ contains all pairs that cannot be connected using solely free edges, i.e., $\mathcal{K} = \{(u, v) \mid (u, v) \in V \times V, u < v\} \setminus C^0$ (see Section 4.1).

In addition, a second set of instances (denoted as ARLP-p25) with a smaller number of commodities has been created. Each such instance is generated from an ARLP instance by adopting each commodity with a probability of 25 %. The main characteristics of sets ARLP and ARLP-p25 are summarized in Tables 5 and 6, respectively. The instances are already preprocessed, in the sense that our preprocessing procedures do not apply. Especially, we only consider instances that are connected, i.e., they consist of a single connected component. Furthermore, we define the set $\mathcal{K}$ so that commodity pairs that can be connected only using free edges and without relays are not included.

The ARLP and the ARLP-p25 instance sets are available at `https://www.ac.tuwien.ac.at/research/problem-instances/#Network_Design_Problem_with_Relays`.

5.2 Computational Results

Test results reported in this section have been obtained on an Intel Xeon E5540 machine with 2.53 GHz. The computing time limit has been set to 7200 seconds and the memory limit to 8 GB RAM. As discussed in Section 4.3 we use a violation threshold of 0.5 when separating strengthening inequalities. This threshold is only considered when solving the problem to integer optimality. When reporting LP bounds in this section, we add all violated inequalities. This means that independent experiments are conducted for the two cases. This leads to situations in which the integer run finds an optimal solution but the LP run terminates due to the time or the memory limit as a result of excessive separation of cutting planes. Conversely, it is also possible that the LP gap is tighter than the final optimality gap of the integer run if the latter cannot progress fast enough.

Tables 2–6 summarize the results. Both models (MCF) and (CUT) are compared with respect to the LP relaxation gap (LP Gap [%]), the final optimality gap (Opt. Gap [%]), the used computing time (t [s]), and the number of priced columns (Columns). LP and optimality gaps are computed as $100 \cdot \frac{UB^* - LB}{UB^*}$ where UB^* is the best known upper bound and LB is the lower bound obtained by the respective algorithm. The upper bounds shown in the tables in the column UB^* are printed bold iff the given value is shown to be the optimal objective value by any of the considered algorithms. Entries marked with "ML" indicate that an experiment has been terminated due to the memory limit. Furthermore, for some of the most challenging instances it was impossible to obtain a lower bound within the imposed time limit. In these cases the LP or optimality gap cannot be computed and respective fields are marked with "TL".

19

Instance	$|V|$	$|E^*|$	$|\mathcal{K}|$	(CH1+)	LP Gap [%] (MCF)	(CUT)	IP Opt. Gap [%] (MCF)	(CUT)	t [s] (MCF)	(CUT)	Columns (MCF)	(CUT)
4A5B70L5K	20	31	4.5	100.2	**0.7**	**0.7**	**0.0**	**0.0**	< 1	< 1	**21**	24
4A5B70L10K	20	31	7.9	103.8	1.4	**1.2**	**0.0**	**0.0**	1	1	**34**	51
5A5B70L5K	25	40	4.2	105.0	1.6	**1.4**	**0.0**	**0.0**	1	< 1	**34**	39
5A5B70L10K	25	40	8.5	105.7	1.6	**1.5**	**0.0**	**0.0**	1	1	**49**	70
6A5B70L5K	30	49	4.7	101.3	0.6	**0.2**	**0.0**	**0.0**	1	1	**48**	50
6A5B70L10K	30	49	8.8	102.1	2.0	**1.5**	**0.0**	**0.0**	4	3	**78**	102
7A5B70L5K	35	58	4.5	101.6	1.4	**1.2**	**0.0**	**0.0**	1	2	57	**50**
7A5B70L10K	35	58	8.4	102.3	2.1	**1.7**	**0.0**	**0.0**	6	5	**98**	115
8A5B70L5K	40	67	4.7	103.3	1.6	**1.2**	**0.0**	**0.0**	4	3	90	**73**
8A5B70L10K	40	67	8.9	106.0	2.9	**2.6**	**0.0**	**0.0**	11	7	**127**	142
9A5B70L5K	45	76	4.8	105.3	**1.3**	**1.3**	**0.0**	**0.0**	4	2	88	**66**
9A5B70L10K	45	76	9.0	105.3	2.4	**1.9**	**0.0**	**0.0**	19	8	176	**158**
10A5B70L5K	50	85	5.0	103.5	2.2	**1.6**	**0.0**	**0.0**	8	7	127	**111**
10A5B70L10K	50	85	9.4	104.3	2.4	**2.3**	**0.0**	**0.0**	32	14	210	**184**
11A5B70L5K	55	94	4.6	100.8	1.9	**1.8**	**0.0**	**0.0**	5	4	103	**86**
11A5B70L10K	55	94	9.1	105.7	1.3	**0.4**	**0.0**	**0.0**	45	11	242	**170**
12A5B70L5K	60	103	4.7	104.7	1.2	**0.8**	**0.0**	**0.0**	9	8	142	**114**
12A5B70L10K	60	103	9.0	103.0	3.9	**2.6**	**0.0**	**0.0**	179	26	357	**252**

Table 2: Results on the Cabral instances. Column $|\mathcal{K}|$ reports the average number of effective commodity pairs. (CH1+) gives the ratio between the objective value obtained by the heuristic and the best known upper bound. We report the LP gap, the optimality gap, the total computing time in seconds (t [s]), and the number of priced columns. Each row reports a mean value over a set of ten instances. Best values are marked bold.

Finally, for the heuristic (CH1+) we report the percentage increase with respect to the best known upper bound given by $100 \cdot \frac{UB^H}{UB^*}$ where UB^H is the objective value obtained by (CH1+).

5.2.1 Cabral instances

Mean values of the computational results obtained for instances from set Cabral are provided in Table 2. Each row corresponds to ten instances for the given instance graph and the number of commodities.

We first notice that both algorithms provide very small LP gaps, with the gaps from (CUT) being consistently smaller than those from (MCF). This can be explained by the fact that the big-M coefficients in inequalities (20) are equal to one since all commodities have the same source in this instance set. This advantage concerning the quality of LP bounds carries over to the integral runs leading to significantly smaller computing times for the (CUT) model. Both models require a comparable number of columns to solve the instances to optimality. In general, the number of priced columns is rather low, which can be explained by the sparsity of the considered input graphs.

Our results constitute a clear improvement compared to the results reported by Cabral et al. (2007) where these instances have been introduced. Whereas Cabral et al. (2007) provide only heuristic solutions with relatively large optimality gaps (with up to 20 % with respect to their best-performing arc-path based formulation), we are able to solve all instances to provable optimality—in most cases within a few seconds only. Moreover, our (CUT) formulation features very small LP gaps that range between 0.2 % and 2.6 % on these instances.

5.2.2 Konak instances

In contrast to the Cabral instances, for this data-set, the number of targets per given source does not exceed two (and is usually only one). Hence, the structure of optimal solutions on the communication graph is less arborescence-like, it is rather an intersection of multiple source-target walks. In Tables 3 and 4 we compare the performance of the proposed exact approaches for instances of type I and type II, respectively. We report the LP gaps, the final optimality gaps, the overall computing times and the numbers of priced columns. Among the instances of type I and II, 11 out of 20 and 18 out of 20 are solved to optimality, respectively. Interestingly, it turns out that instances with edge lengths equivalent to the costs (type I) are significantly harder to solve for our algorithms than those where edge costs and edge lengths are inversely correlated (type II).

Konak Type I. We first compare the LP gaps reported in Table 3. Considering only those cases when both algorithms are able to finish the computation of the LP bound, we observe that the gaps of both formulations are the same. This is not surprising since the benefits of inequalities (20) diminish due to multiple source and target nodes in each instance. In general, these instances are much harder to solve for our algorithms than those from the Cabral set. The main reason is that the graphs are much denser which leads to higher separation and pricing efforts.

The performance for the complete runs (until finding an optimal integer solution or reaching the time limit) is consistent with the LP bound results. While the majority of instances with at most 60 nodes could be solved by both algorithms, the final optimality gaps are quite large for instances with 80 and more nodes. Both algorithms feature similar computing times with a slight advantage for the (MCF) model. The main advantage of the (CUT) formulation is that it can provide bounds for all instances. The (MCF), on the other hand, cannot provide bounds for the two most difficult instances within the time limit of two hours. We observe that our algorithms improve the initial upper bounds received from the heuristic (reported in the column (CH1+)) for all but two instances with 80 nodes and the instances with 160 nodes.

Konak Type II. The results shown in Table 4, compared with those obtained for instances of type I, clearly indicate that for our algorithms type II instances are easier to solve than the type I instances. The above discussed relation between the two approaches remains roughly the same, both regarding the quality of the LP gaps and the overall performance. In total 18 out of 20 instances of this group could be solved to optimality by the (MCF), three more than by the (CUT) model. The LP gaps are much smaller than for the type I instances and for two instances the LP gap is even zero. The upper bounds obtained from (CH1+) have been shown to be optimal by our algorithms in two cases and have been improved by them for all remaining ones except the largest one (160N_10K_35L).

5.2.3 ARLP instances

Recall that the Cabral and Konak instances contain very few commodities and almost no free edges. The influence of the ratio of free edges to augmenting edges on the proposed approaches as well as the influence of a larger number of commodities is therefore investigated on the set of ARLP instances. Results obtained for the ARLP instances and all considered percentages of free edges (20 %, 50 %, or 80 % of all available edges) are provided in Table 5. It is not surprising that the most difficult instances are those with only 20 % free edges and that the instances become significantly easier to solve with an increasing number of free edges. Notice that, due to the huge number of commodities, the model (MCF), which performed quite well on the other two data-sets, can now only deal with the smallest instances with 40 nodes and some instances with 50 nodes. For larger instances, (MCF) always hits the memory limit, due to its excessive size. Clearly, using the (CUT) model greatly helps to overcome this issue.

As before, (MCF) provides the same LP gaps as (CUT) whenever both models manage to terminate within the time limit. The ARLP instances are more challenging for the (CUT) model as well. This can be explained by the excessive number of cuts that need to be generated, especially for the calculation of LP bounds, where no violation threshold is used. Moreover,

21

Instance	$|V|$	$|E^0|$	$|E^*|$	$|\mathcal{K}|$	d_{max}	UB^*	(CH1+)	LP Gap [%] (MCF)	(CUT)	Opt. Gap [%] (MCF)	(CUT)	t [s] (MCF)	(CUT)	Columns (MCF)	(CUT)
40N_5K_30L	40	0	198	5	30	**473.80**	102.7	**11.9**	**11.9**	**0.0**	**0.0**	**16**	27	299	**227**
40N_5K_35L	40	0	272	5	35	**352.08**	102.7	**21.0**	**21.0**	**0.0**	**0.0**	1173	**533**	1012	**767**
40N_10K_30L	40	0	198	10	30	**518.98**	108.2	**15.9**	**15.9**	**0.0**	**0.0**	180	**152**	**401**	421
40N_10K_35L	40	0	272	10	35	**399.36**	112.3	**23.1**	**23.1**	6.6	8.0	7200	7200	1043	**1027**
50N_5K_30L	50	0	279	5	30	**283.79**	120.3	**0.0**	**0.0**	**0.0**	**0.0**	**1**	5	232	**186**
50N_5K_35L	50	0	372	5	35	**260.24**	100.0	**1.2**	**1.2**	**0.0**	**0.0**	**6**	27	565	**499**
50N_10K_30L	50	0	279	10	30	**540.39**	103.8	**13.0**	**13.0**	**0.0**	**0.0**	**158**	411	**540**	557
50N_10K_35L	50	0	372	10	35	**404.32**	111.9	**11.8**	**11.8**	**0.0**	**0.0**	**204**	261	**1177**	1307
60N_5K_30L	60	0	305	5	30	**509.12**	103.2	**21.5**	**21.5**	**0.0**	**0.0**	**39**	70	**509**	541
60N_5K_35L	60	0	412	5	35	**377.02**	105.8	**10.9**	**10.9**	**0.0**	**0.0**	**48**	94	**939**	1146
60N_10K_30L	60	0	305	10	30	**678.84**	107.0	**24.4**	**24.4**	**0.0**	**0.0**	**886**	1063	**703**	730
60N_10K_35L	60	0	412	10	35	**499.64**	112.9	**19.0**	**19.0**	**0.0**	**0.0**	**891**	1400	**1498**	1520
80N_5K_30L	80	0	641	5	30	353.86	105.2	**15.7**	**15.7**	9.1	10.0	7200	7200	3887	**3784**
80N_5K_35L	80	0	853	5	35	334.21	103.2	**19.9**	TL	**16.1**	16.7	7200	7200	9110	**8394**
80N_10K_30L	80	0	641	10	30	513.02	100.0	**36.7**	TL	**32.0**	35.2	7200	7200	**3613**	3815
80N_10K_35L	80	0	853	10	35	516.91	100.0	**43.0**	TL	44.4	**42.2**	7200	7200	**6952**	8592
160N_5K_30L	160	3	2770	5	30	298.31	100.0	**29.7**	TL	29.6	**28.2**	7200	7200	**15941**	23489
160N_5K_35L	160	3	3621	5	35	314.52	100.0	TL	37.0	58.1	**37.2**	7200	7200	**20303**	30877
160N_10K_30L	160	3	2770	10	30	470.54	100.0	TL	TL	TL	**44.1**	7200	7200	**11229**	23027
160N_10K_35L	160	3	3621	10	35	484.97	100.0	TL	TL	TL	**50.2**	7200	7200	**11872**	36042

Table 3: Results on the Konak instances (type I). Column UB^* provides the best known upper bounds, optimal bounds are marked bold. (CH1+) gives the ratio between the objective value obtained by the heuristic and the best known upper bound. We report the LP gap, the optimality gap, the total computing time in seconds (t [s]), and the number of priced columns. Best values are marked bold. If an algorithm failed to compute a lower bound due to the time limit, the respective gap entry is marked with "TL".

Instance	$|V|$	$|E^0|$	$|E^*|$	$|\mathcal{K}|$	d_{max}	UB^*	(CH1+)	LP Gap [%] (MCF)	(CUT)	Opt. Gap [%] (MCF)	(CUT)	t [s] (MCF)	(CUT)	Columns (MCF)	(CUT)
40N_5K_30L	40	1	197	5	30	**247.27**	100.1	**0.0**	**0.0**	**0.0**	**0.0**	< 1	1	111	**80**
40N_5K_35L	40	0	272	5	35	**111.30**	101.7	**7.1**	**7.1**	**0.0**	**0.0**	9	17	459	**314**
40N_10K_30L	40	1	197	10	30	**292.62**	100.1	**4.4**	**4.4**	**0.0**	**0.0**	13	18	293	**267**
40N_10K_35L	40	0	272	10	35	**140.51**	101.0	**7.7**	**7.7**	**0.0**	**0.0**	61	**45**	690	**625**
50N_5K_30L	50	1	278	5	30	**119.80**	100.0	**0.0**	**0.0**	**0.0**	**0.0**	< 1	1	**98**	104
50N_5K_35L	50	0	372	5	35	**155.57**	105.2	**1.0**	**1.0**	**0.0**	**0.0**	**2**	33	**422**	547
50N_10K_30L	50	1	278	10	30	**279.70**	100.6	**2.0**	TL	**0.0**	**0.0**	**3**	28	**247**	432
50N_10K_35L	50	0	372	10	35	**206.22**	102.1	**1.6**	TL	**0.0**	**0.0**	**32**	127	**828**	1303
60N_5K_30L	60	3	302	5	30	**317.32**	104.0	**14.2**	**14.2**	**0.0**	**0.0**	**7**	26	**337**	444
60N_5K_35L	60	0	412	5	35	**166.35**	100.0	**0.0**	**0.0**	**0.0**	**0.0**	< 1	3	**315**	446
60N_10K_30L	60	3	302	10	30	**414.32**	120.4	**12.2**	**12.2**	**0.0**	**0.0**	**52**	88	**444**	583
60N_10K_35L	60	0	412	10	35	**242.32**	100.1	**5.2**	**5.2**	**0.0**	**0.0**	**21**	53	**748**	1016
80N_5K_30L	80	2	639	5	30	**134.73**	108.6	**4.9**	**4.9**	**0.0**	**0.0**	**32**	167	**1455**	1705
80N_5K_35L	80	1	852	5	35	**104.04**	100.4	**2.0**	**2.0**	**0.0**	**0.0**	**46**	306	**2307**	2899
80N_10K_30L	80	2	639	10	30	**187.17**	105.0	**8.1**	**8.1**	**0.0**	**0.0**	695	**625**	2756	**2501**
80N_10K_35L	80	1	852	10	35	**168.62**	102.2	**11.2**	TL	**0.0**	12.9	4382	7200	**7947**	9114
160N_5K_30L	160	9	2764	5	30	**78.61**	106.2	**6.6**	**6.6**	**0.0**	4.2	1491	7200	**12812**	23985
160N_5K_35L	160	9	3615	5	35	**68.15**	101.4	**9.0**	**9.0**	**0.0**	6.7	5273	7200	**29676**	30003
160N_10K_30L	160	9	2764	10	30	112.06	106.1	**6.5**	TL	5.7	11.6	7200	7200	**16850**	26344
160N_10K_35L	160	9	3615	10	35	117.19	100.0	TL	TL	**10.3**	33.4	7200	7200	**22324**	34909

Table 4: Results on the Konak Instances (type II). Column UB^* provides the best known upper bounds, optimal bounds are marked bold. (CH1+) gives the ratio between the objective value obtained by the heuristic and the best known upper bound. We report the LP gap, the optimality gap, the total computing time in seconds (t [s]), and the number of priced columns. Best values are marked bold. If an algorithm failed to compute a lower bound due to the time limit, the respective gap entry is marked with "TL".

the effects of the connectivity cuts in the original graph diminish on this data-set, due to the presence of a substantial number of free edges. Overall, (CUT) is clearly the best-performing model for this data-set. It is able to find optimal solutions for 22 out of 24 instances. The (MCF) model, on the other hand, manages to solve only six instances to optimality. As for the other instance sets, the upper bounds obtained from (CH1+) are improved for almost all instances by our algorithms which only fail to improve them for one of the most challenging problems (80N50L20F_A) and prove optimality of one solution obtained by (CH1+).

Instance	$|V|$	$|E^0|$	$|E^*|$	$|\mathcal{K}|$	UB^*	(CH1+)	LP Gap [%]		Opt. Gap [%]		t [s]		Columns	
							(MCF)	(CUT)	(MCF)	(CUT)	(MCF)	(CUT)	(MCF)	(CUT)
40N50L20F_A	40	26	124	724	**874**	122.5	TL	**10.7**	TL	**0.0**	7200	**142**	**132**	1340
40N50L20F_B	40	35	123	688	**874**	113.3	TL	**13.4**	TL	**0.0**	7200	**139**	147	1196
40N50L50F_A	40	89	78	513	**837**	103.9	**15.2**	**15.2**	**0.0**	**0.0**	2504	**14**	**59**	83
40N50L50F_B	40	71	72	586	**876**	108.3	**6.4**	**6.4**	**0.0**	**0.0**	2683	**7**	76	**74**
40N50L80F_A	40	146	32	443	**516**	100.0	**0.0**	**0.0**	**0.0**	**0.0**	10	**2**	1	**0**
40N50L80F_B	40	154	35	423	**777**	100.9	**6.6**	**6.6**	**0.0**	**0.0**	155	**5**	18	23
50N50L20F_A	50	44	212	1111	**815**	106.5	TL	**5.9**	ML	**0.0**	ML	**416**	ML	**3973**
50N50L20F_B	50	59	235	1022	**656**	131.2	TL	**2.8**	ML	**0.0**	ML	**72**	ML	**3032**
50N50L50F_A	50	132	157	719	**543**	104.4	**9.6**	TL	ML	**0.0**	ML	**11**	ML	**114**
50N50L50F_B	50	117	132	873	**775**	100.8	TL	**5.8**	ML	**0.0**	ML	**30**	ML	**211**
50N50L80F_A	50	175	51	788	**630**	143.8	**11.1**	TL	**0.0**	**0.0**	211	**10**	25	**7**
50N50L80F_B	50	212	58	682	**572**	119.2	**0.0**	**0.0**	**0.0**	**0.0**	29	**5**	1	**0**
60N50L20F_A	60	72	269	1549	**775**	111.1	ML	**5.5**	ML	**0.0**	ML	**257**	ML	**3685**
60N50L20F_B	60	63	268	1588	**976**	129.7	ML	**17.8**	ML	**0.0**	ML	**1397**	ML	**3725**
60N50L50F_A	60	216	204	1036	**628**	115.8	TL	ML	ML	**0.0**	ML	**44**	ML	**201**
60N50L50F_B	60	197	200	1103	**743**	103.0	TL	ML	ML	**0.0**	ML	**58**	ML	**255**
60N50L80F_A	60	311	85	854	**503**	119.7	**2.1**	TL	ML	**0.0**	ML	**16**	ML	**16**
60N50L80F_B	60	283	74	1041	**624**	110.1	**9.9**	ML	ML	**0.0**	ML	**18**	ML	**4**
80N50L20F_A	80	123	525	2729	1084	100.0	ML	**35.1**	ML	**34.0**	ML	**7200**	ML	**19111**
80N50L20F_B	80	124	545	2659	790	125.8	ML	TL	ML	**9.6**	ML	**7200**	ML	**21190**
80N50L50F_A	80	335	342	1916	**498**	120.9	ML	ML	ML	**0.0**	ML	**65**	ML	**71**
80N50L50F_B	80	366	375	1902	**541**	135.5	ML	TL	ML	**0.0**	ML	**159**	ML	**424**
80N50L80F_A	80	548	148	1834	**577**	101.4	ML	ML	ML	**0.0**	ML	**67**	ML	**5**
80N50L80F_B	80	597	158	1532	**549**	159.2	ML	ML	ML	**0.0**	ML	**85**	ML	**4**

Table 5: Results on the ARLP instances. Column UB^* provides the best known upper bounds, optimal bounds are marked bold. (CH1+) gives the ratio between the objective value obtained by the heuristic and the best known upper bound. We report the LP gap, the optimality gap, the total computing time in seconds (t [s]), and the number of priced columns. Best values are marked bold. Entries marked with "ML" indicate that an experiment has been terminated due to the memory limit. If an algorithm failed to compute a lower bound due to the time limit, the respective gap entry is marked with "TL".

ARLP-p25 instances. The purpose of evaluating our algorithms on this family of instances was to study the influence of the number of commodities to the algorithmic performance. Recall that for the ARLP instances, each node pair is a commodity, i.e., $|\mathcal{K}|$ is in $O(|V|^2)$. For the ARLP-p25 instances, the number of commodities is reduced to a quarter. In general, the results indicate that as long as the number of commodities remains $O(|V|^2)$, the NDPR is much more difficult to solve than when the number of commodities is fixed to a small constant value (as this was the case for the Cabral and the Konak instances). Moreover, the remaining commodities still enforce solutions that guarantee full connectivity since the optimal objective values do not change for the corresponding instances. The detailed results are provided in Table 6.

Again, the LP gaps of both models are the same whenever both of them terminated within the time limit. However, this time more LP bounds have been obtained due to the smaller number of commodities. The quality of the bounds is roughly comparable to those of the ARLP

Instance	$|V|$	$|E^0|$	$|E^*|$	$|\mathcal{K}|$	UB^*	(CH1+)	LP Gap [%]		Opt. Gap [%]		t [s]		Columns	
							(MCF)	(CUT)	(MCF)	(CUT)	(MCF)	(CUT)	(MCF)	(CUT)
40N50L20F_A_p25	40	26	124	181	**874**	119.2	**11.4**	**11.4**	11.7	**0.0**	7200	**106**	**499**	1429
40N50L20F_B_p25	40	35	123	172	**874**	102.7	**14.6**	**14.6**	11.2	**0.0**	7200	**109**	**529**	1238
40N50L50F_A_p25	40	89	78	129	**837**	102.2	**15.2**	**15.2**	**0.0**	**0.0**	152	**10**	**58**	81
40N50L50F_B_p25	40	71	72	147	**876**	104.8	**6.4**	**6.4**	**0.0**	**0.0**	112	**3**	**80**	96
40N50L80F_A_p25	40	146	32	111	**516**	100.0	**0.0**	**0.0**	**0.0**	**0.0**	2	**1**	2	**0**
40N50L80F_B_p25	40	154	35	106	**777**	100.9	**6.6**	**6.6**	**0.0**	**0.0**	20	**2**	**16**	22
50N50L20F_A_p25	50	44	212	278	**815**	106.0	TL	**6.1**	TL	**0.0**	7200	**611**	**572**	4396
50N50L20F_B_p25	50	59	235	256	**656**	122.0	TL	**2.8**	TL	**0.0**	7200	**209**	**547**	4256
50N50L50F_A_p25	50	132	157	180	**543**	112.0	**9.6**	**9.6**	**0.0**	**0.0**	179	**5**	**142**	153
50N50L50F_B_p25	50	117	132	219	**775**	100.0	**5.8**	**5.8**	**0.0**	**0.0**	4431	**11**	273	**184**
50N50L80F_A_p25	50	175	51	197	**630**	124.0	**11.1**	**11.1**	**0.0**	**0.0**	26	**2**	17	**3**
50N50L80F_B_p25	50	212	58	171	**572**	119.2	**0.0**	**0.0**	**0.0**	**0.0**	6	**1**	**1**	3
60N50L20F_A_p25	60	72	269	388	**775**	116.3	TL	**5.5**	TL	**0.0**	7200	**2174**	**366**	4904
60N50L20F_B_p25	60	63	268	397	**976**	126.0	TL	**18.4**	TL	**0.0**	7200	**4412**	**343**	4037
60N50L50F_A_p25	60	216	204	259	**628**	115.8	**8.0**	TL	**0.0**	**0.0**	6777	**46**	**220**	237
60N50L50F_B_p25	60	197	200	276	**743**	122.3	**8.5**	**8.5**	4.7	**0.0**	7200	**45**	**242**	360
60N50L80F_A_p25	60	311	85	214	**503**	148.7	**2.1**	**2.1**	**0.0**	**0.0**	125	**7**	25	**24**
60N50L80F_B_p25	60	283	74	261	**624**	124.5	**9.9**	**9.9**	**0.0**	**0.0**	81	**16**	15	**14**
80N50L20F_A_p25	80	123	525	683	1095	100.0	TL	**35.8**	ML	**33.9**	ML	7200	ML	**19296**
80N50L20F_B_p25	80	124	545	665	1024	100.0	TL	TL	ML	**35.4**	ML	7200	ML	**21959**
80N50L50F_A_p25	80	335	342	479	**498**	120.9	**1.1**	TL	**0.0**	**0.0**	1890	**28**	155	193
80N50L50F_B_p25	80	366	375	476	**541**	141.6	TL	TL	TL	**0.0**	7200	**181**	224	538
80N50L80F_A_p25	80	548	148	459	**577**	100.2	**2.6**	TL	**0.0**	**0.0**	826	**40**	17	**14**
80N50L80F_B_p25	80	597	158	383	**549**	132.1	**9.7**	TL	**0.0**	**0.0**	1580	**55**	13	**12**

Table 6: Results on the ARLP-p25 instances. Column UB^* provides the best known upper bounds, optimal bounds are marked bold. (CH1+) gives the ratio between the objective value obtained by the heuristic and the best known upper bound. We report the LP gap, the optimality gap, the total computing time in seconds (t [s]), and the number of priced columns. Best values are marked bold. Entries marked with "ML" indicate that an experiment has been terminated due to the memory limit. If an algorithm failed to compute a lower bound due to the time limit, the respective gap entry is marked with "TL".

instances. The number of optimal solutions found does not change when reducing the number of commodities, i.e., optimal solutions have been found for 22 out of 24 instances. For the (MCF) the number of instances solved to optimality greatly increases from 6 to 14, mainly due to the smaller number of commodities reducing the size of the model. Surprisingly, the impact on the algorithmic performance of (CUT) is much smaller. In fact, in several cases the reduced instances are even harder to solve. As mentioned above the optimal solutions remain the same as for the original instances. Therefore, we "loose" constraints that might help to prove optimality earlier. This is particularly relevant to the (CUT) model where the number of variables is independent of the number of the commodity pairs. As before, one solution obtained from (CH1+) is shown to be optimal and all but one of the remaining upper bounds from (CH1+) are improved by our approaches.

Sensitivity Analysis.

The number of feasible walks realizing a connection of some commodity pair depends on the distance limit in relation to the edge lengths. More specifically, it depends on the average number of edges that can be part of a feasible subwalk that uses no relays. In the following, we want to investigate the effect of this characteristic on our algorithms. To this end, we consider the Cabral instances and vary the distance limit $d_{\max}$.

The unmodified Cabral instances feature a distance limit of 70 and the edge lengths are chosen uniformly at random from the interval $[10, 30]$. Therefore, a feasible walk of maximal
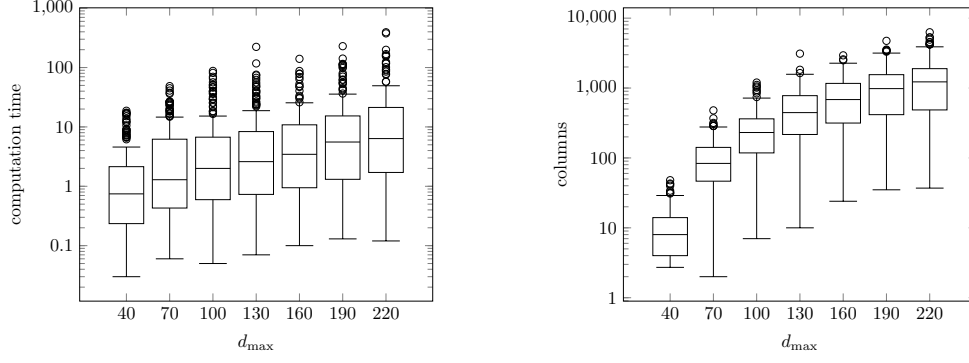
Figure 10: Sensitivity analysis regarding the distance maximum $d_{\max}$ on the Cabral instances for the (CUT) model. Each box considers 180 instances. Both box plots use a logarithmic scale.

length consists on average of three to four edges. Increasing $d_{\max}$ by a value of 30 means that a feasible walk without relays may contain at least one additional edge. We consider distance limits between 40 and 220 in steps of 30 for our experiment. The results are visualized in terms of box plots in Figure 10 for the (CUT) model.

Due to the increasing number of possible options for connecting the commodity pairs it can be expected that the instances become harder to solve as the distance limit increases. This can be verified in terms of the box plots. However, we can also see that the computing times increase only moderately. Per step the median computing time rises by roughly one second. Similarly, the number of columns required to solve an instance to optimality increases. The respective results obtained with the (MCF) model are quite similar except that the baseline lies a bit higher. In total, we can conclude that the performance of our algorithms remains quite robust against changes to the distance limit.

Solution Shape and Characteristics.

In this section, we analyze the structure of optimal solutions for two small examples corresponding to instances 40N30C50L20F_A and 40N30C50L80F_A from the ARLP set with 20 % and 80 % of free edges, respectively. In Figures 11a and 12a we visualize the two corresponding optimal solutions. We notice that for 40N30C50L20F_A, despite the fact that there are 724 commodities, only three relays need to be installed (with a reasonable complement of augmentation edges) to enable these communications. In Figure 11, we compare the optimal solution of 40N30C50L20F_A with the one obtained for the same input graph, but with a much smaller number of commodities: the set of commodities $\mathcal{K}'$ is obtained by removing all commodities except those containing the node that is involved in the fewest commodities. The structure of the obtained optimal solution is similar to the original one, with three installed relays and a similar number of augmentation edges. Along with the results reported for the ARLP instances, this figure indicates that the estimated investment costs do not increase linearly with the number of commodities. In general there are higher investment costs for setting up the infrastructure, and once it is established, marginal increase in the number of commodities will not be reflected in the increase of the set-up costs (cf. the solution values in Tables 5 and 6).

Figure 12 visualizes the optimal solutions for 40N30C50L80F_A and its "sparser" variant (with the set of commodities $\mathcal{K}'$ constructed as above), respectively. We observe that when 80 % of available edges are free, the need to install augmentation edges almost vanishes but a certain number of relays still needs to be installed to enable communications. Comparing the optimal solution for the sparser problem with the original one, we notice that the number of relays can be reduced, if the source node is centrally located, as this is the case in the shown example.

25

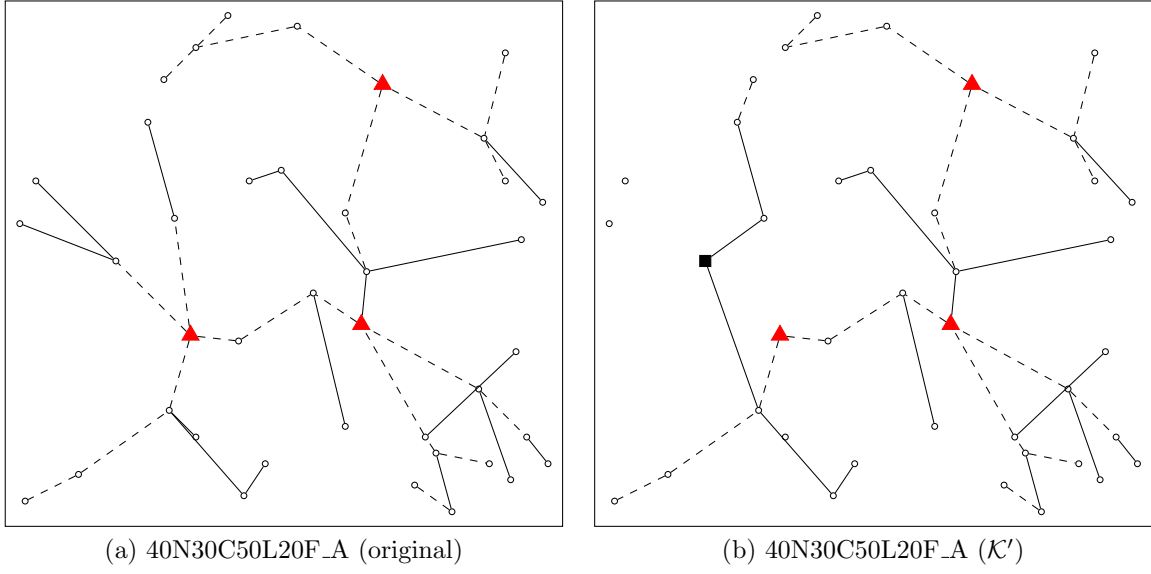(a) 40N30C50L20F_A (original)　　　　(b) 40N30C50L20F_A ($\mathcal{K}'$)

Figure 11: Solutions to ARLP instance 40N30C50L20F_A with different numbers of commodities. Solid lines indicate free edges and dashed lines augmenting edges. Selected relays are marked with triangles. On the right the single source is marked with a square.
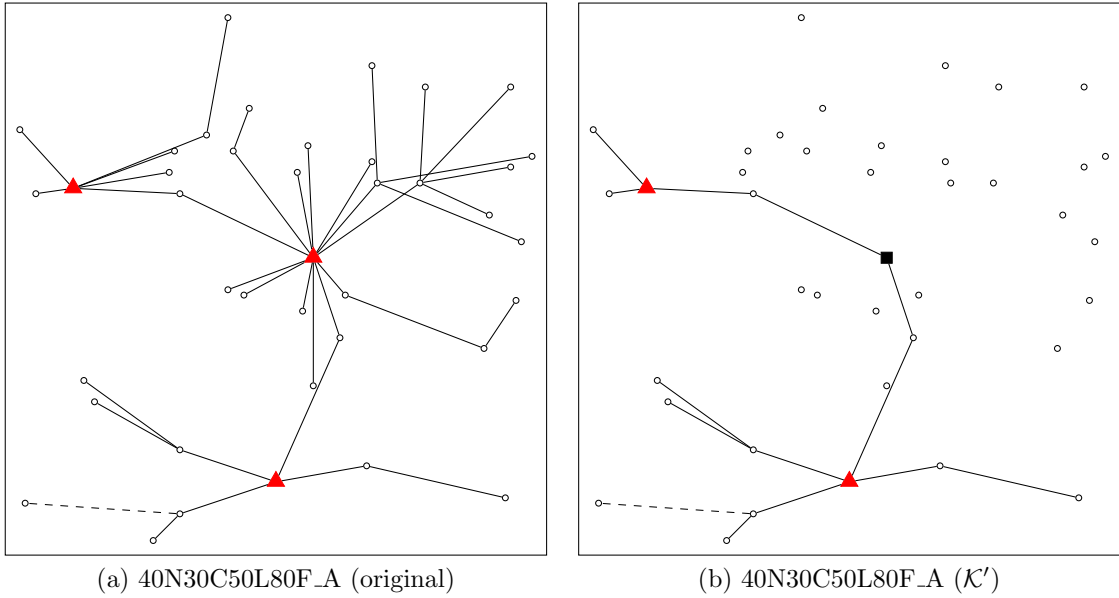

(a) 40N30C50L80F_A (original)　　　　(b) 40N30C50L80F_A ($\mathcal{K}'$)

Figure 12: Solutions to ARLP instance 40N30C50L80F_A with different numbers of commodities. Solid lines indicate free edges and dashed lines augmenting edges. Selected relays are marked with triangles. On the right the single source is marked with a square.

26

6 Conclusion

In this work we introduced new MILP formulations for the solution of the NDPR utilizing an exponential number of variables (and constraints). We proved several conditions and properties of optimal solutions and revised the concept of communication graphs for the NDPR which are exploited in our MILP formulations. Two branch-price-and-cut algorithms have been developed. The first one is based on a multi-commodity flow formulation on an undirected communication graph whereas the second is based on a cut-set formulation on a directed communication graph. The computational study on instances from the literature and a newly created set of instances shows that the cut-set formulation on the directed communication graph has the overall best performance. The multi-commodity flow formulation on the undirected communication graph performs reasonably well, but only up to a limited number of nodes and/or commodities. Due to its excessive number of variables, the latter formulation exhibits serious memory issues that makes it less appealing for practical applications involving larger graphs or a higher number of commodities.

We conducted a sensitivity analysis on the Cabral instances with different $d_{\max}$ restrictions. The results showed that our algorithms are quite robust against changes to this parameter featuring only a comparatively small increase in computing times and priced columns.

Compared to the previous attempt from the literature to develop an exact model for the NDPR by Cabral et al. (2007), the main advantage of our modeling approach is that we do not use variables corresponding to entire walks between commodities (including the decisions for placing relays). Instead, we consider only simple paths between two consecutive relays, whereas the decision where to place the relays is modeled through the communication graph. That way, our pricing draws a computational advantage from the fact that augmenting edges can be simultaneously shared by multiple commodities. The positive effect of our modeling approach is striking for instances with many commodities, where the need for the simultaneous reuse of augmenting edges is even more amplified.

6.1 Future Work

Besides telecommunication network design, the NDPR can be used to answer strategic questions in the context of electric mobility. A company running its operations based on a fleet of electric vehicles (EVs), be it a logistics company or an e-car provider, faces a difficult decision problem of planning the underlying charging infrastructure. Due to expensive EV batteries and their limited range, a stable and robust charging infrastructure is crucial for running the business suitably. Since building and/or renting charging stations is expensive, logistic companies or e-car providers are interested in minimizing the number of charging stations whilst enabling travel between specific locations (cf. commodities). Furthermore, in some metropolises (including Stockholm, Gothenburg, and Singapore) *congestion taxing* or *congestion charging* mechanisms are implemented. This means that shorter distances can be traversed (e.g., via shortcuts through the inner city), but in that case a certain road toll is to be paid (see, e.g., EPASS24, Land Transport Authority). Similarly, urban freeways passing through a downtown area can be subject to compulsory electronic toll (like the case in Santiago de Chile, or some Norwegian cities). Consequently, if a company is interested in building charging stations for its fleet, it also has to gauge whether toll roads are to be used. When making strategic decisions, costs for toll roads are typically estimated over a longer planning period and considered as fixed link costs.

Two main strategic design questions arising in this complex decision process are addressed by the NDPR: Given a family of origin-destination pairs EVs need to travel, and given the existing links that can be traversed:

1. What are the optimal locations for placing the charging stations and how many of them are needed?

2. Could the available infrastructure be enhanced by including additional links (shortcuts), to reduce the travel distances?

The relevance of the NDPR for planning EVs' charging infrastructure has not been sufficiently acknowledged in the existing literature. This is maybe due to the additional aspects that need to be taken into account when dealing with e-mobility. These include restricting

the distance arising from detours necessary for vehicle recharging and the maximum number of (time-consuming) recharging stops an EV requires before reaching its final destination. Although the NDPR does not consider these additional aspects, there is no doubt that the problem plays an important role for e-mobility applications for two reasons: (1) NDPR may appear as a subproblem (i.e., in some decomposition schemes), and (2) the proposed algorithms can be used to derive heuristic solutions in multiple-phase approaches, where the complex decision process is approached step-by-step. Hence, the NDPR provides important insights for the companies running their business with a fleet of EVs. It helps in estimating the initial set-up costs (induced by the installation of recharging stations and potential purchases of road-toll passes). Moreover, by using a correlation between the edge lengths and lengths of the trips, the routing decisions obtained through an NDPR solution implicitly help in estimating a lower bound on the number of required EVs. Similarly, assuming that all trips will be covered, an upper bound on the expected profit can be calculated.

Interesting and more difficult NDPR variants that are important directions for future work include the following aspects: (1) limiting the maximum number of recharging stops (relays) used by a single commodity, (2) limiting the maximum waiting times imposed by recharging, or (3) limiting the overall trip length per commodity.

Acknowledgements

References

O. Arslan, B. Yıldız, and O. E. Karaşan. Minimum cost path problem for plug-in hybrid electric vehicles. *Transportation Research Part E: Logistics and Transportation Review*, 80:123–141, 2015.

C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, and P. H. Vance. Branch-and-price: Column generation for solving huge integer programs. *Operations Research*, 46: 316–329, 1998.

BGL. The boost graph library (BGL). `http://www.boost.org/doc/libs/1_63_0/libs/graph/doc/index.html`. accessed 2017-09-17.

E. A. Cabral, E. Erkut, G. Laporte, and R. A. Patterson. The network design problem with relays. *European Journal of Operational Research*, 180(2):834–844, 2007.

J. F. Campbell and M. E. O'Kelly. Twenty-five years of hub location research. *Transportation Science*, 46(2):153–169, 2012.

I. Capar, M. Kuby, V. J. Leon, and Y.-J. Tsai. An arc coverâĂŞpath-cover formulation and strategic analysis of alternative-fuel station locations. *European Journal of Operational Research*, 227(1):142–151, 2013.

S. Chen, I. Ljubić, and S. Raghavan. The regenerator location problem. *Networks*, 55(3): 205–220, 2010.

S. Chen, I. Ljubić, and S. Raghavan. The generalized regenerator location problem. *INFORMS Journal on Computing*, 27(2):204–220, 2015.

B. V. Cherkassy and A. V. Goldberg. On implementing push-relabel method for the maximum flow problem. In *Proceedings of the 4th International IPCO Conference on Integer Programming and Combinatorial Optimization*, pages 157–171. Springer, 1995.

28

J. Desrosiers and M. E. Lübbecke. Branch-price-and-cut algorithms. In *Wiley Encyclopedia of Operations Research and Management Science*. John Wiley & Sons, Inc., 2011.

E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1 (1):269–271, 1959.

EPASS24. EPASS24: The swedish road toll system. `https://www.epass24.com/`. accessed 2015-08-02.

G. Gamrath, T. Fischer, T. Gally, A. M. Gleixner, G. Hendel, T. Koch, S. J. Maher, M. Miltenberger, B. Müller, M. E. Pfetsch, C. Puchert, D. Rehfeldt, S. Schenker, R. Schwarz, F. Serrano, Y. Shinano, S. Vigerske, D. Weninger, M. Winkler, J. T. Witt, and J. Witzig. The SCIP optimization suite 3.2. Technical Report 15-60, ZIB, Takustr.7, 14195 Berlin, 2016.

B. Gendron, A. Lucena, A. S. da Cunha, and L. Simonetti. Benders decomposition, branch-and-cut, and hybrid algorithms for the minimum connected dominating set problem. *INFORMS Journal on Computing*, 26(4):645–657, 2014.

O. Kabadurmus and A. E. Smith. Multi-commodity k-splittable survivable network design problems with relays. *Telecommunication Systems*, pages 1–11, 2015.

A. Konak. Network design problem with relays: A genetic algorithm with a path-based crossover and a set covering formulation. *European Journal of Operational Research*, 218(3):829–837, 2012.

S. Kulturel-Konak and A. Konak. A local search hybrid genetic algorithm approach to the network design problem with relay stations. In S. Raghavan, B. Golden, and E. Wasil, editors, *Telecommunications Modeling, Policy, and Technology*, volume 44 of *Operations Research/Computer Science Interfaces*, pages 311–324. Springer US, 2008.

Singapore Land Transport Authority. Electronic road pricing (erp). `http://www.lta.gov.sg/content/ltaweb/en/roads-and-motoring/managing-traffic-and-congestion/electronic-road-pricing-erp.html`. accessed 2015-08-02.

G. Laporte and M. M. B. Pascoal. Minimum cost path problems with relays. *Computers & Operations Research*, 38(1):165–173, 2011.

X. Li, Y. P. Aneja, and J. Huo. Using branch-and-price approach to solve the directed network design problem with relays. *Omega*, 40(5):672–679, 2012.

S. Lin, X. Li, K. Wei, and C. Yue. A tabu search based metaheuristic for the network design problem with relays. In *Service Systems and Service Management (ICSSSM), 2014 11th International Conference on*, pages 1–6, 2014.

Ivana Ljubić, René Weiskircher, Ulrich Pferschy, Gunnar W. Klau, Petra Mutzel, and Matteo Fischetti. An algorithmic framework for the exact solution of the prize-collecting steiner tree problem. *Math. Program.*, 105(2-3):427–449, 2006.

A. Lucena, N. Maculan, and L. Simonetti. Reformulations and solution algorithms for the maximum leaf spanning tree problem. *Computational Management Science*, 7(3):289–311, 2010.

S. Pelletier, O. Jabali, and G. Laporte. 50th anniversary invited article—goods distribution with electric vehicles: Review and research perspectives. *Transportation Science*, 50(1): 3–22, 2016.

S. Raghavan and Daliborka Stanojević. Branch and price for wdm optical networks with no bifurcation of flow. *INFORMS Journal on Computing*, 23(1):56–74, 2011.

Q. Rahman, S. Bandyopadhyay, and Y. Aneja. Optimal regenerator placement in translucent optical networks. *Optical Switching and Networking*, 15:134–147, 2015.

M. Schneider, A. Stenger, and D. Goeke. The electric vehicle-routing problem with time windows and recharging stations. *Transportation Science*, 48(4):500–520, 2014.

C. S. Sung and S. H. Song. Branch-and-price algorithm for a combined problem of virtual path establishment and traffic packet routing in a layered communication network. *Journal of the Operational Research Society*, 54(1):72–82, Jan 2003.

H. Üster and P. Kewcharoenwong. Strategic design and analysis of a relay network in truckload transportation. *Transportation Science*, 45:505–523, 2011.

H. Üster and N. Maheshwari. Strategic network design for multi-zone truckload shipments. *IIE Transactions*, 39(2):177–189, 2007.

Y. Xiao and A. Konak. A variable neighborhood search for the network design problem with relays. *Journal of Heuristics*, pages 1–28, 2017.

B. Yıldız and O. E. Karaşan. Regenerator location problem and survivable extensions: A hub covering location perspective. *Transportation Research Part B: Methodological*, 71:32–55, 2015.

B. Yıldız and O. E. Karaşan. Regenerator location problem in flexible optical networks. *Operations Research*, 65(3):595–620, 2017.

B. Yıldız, O. Arslan, and O. E. Karaşan. A branch and price approach for routing and refueling station location model. *European Journal of Operational Research*, 248(3):815–826, 2016.

B. Yıldız, O. E. Karaşan, and H. Yaman. Branch-and-price approaches for the network design problem with relays. *Computers & Operations Research*, 92:155–169, 2018.