Mathematical Programming, Series B manuscript No.
(will be inserted by the editor)

Orientation-based Models for {0,1,2}-Survivable Network Design: Theory and Practice

Markus Chimani · Maria Kandyba ·
Ivana Ljubić · Petra Mutzel

the date of receipt and acceptance should be inserted later

Abstract We consider {0,1,2}-Survivable Network Design problems with node-connectivity constraints. In the most prominent variant, we are given an edge-weighted graph and two customer sets $\mathscr{R}_1$ and $\mathscr{R}_2$; we ask for a minimum cost subgraph that connects all customers, and guarantees two-node-connectivity for the $\mathscr{R}_2$ customers. We also consider an alternative of this problem, in which 2-node-connectivity is only required w.r.t. a certain root node, and its prize-collecting variant.

The central result of this paper is a novel graph-theoretical characterization of 2-node-connected graphs via orientation properties. This allows us to derive two classes of ILP formulations based on directed graphs, one using multi-commodity flow and one using cut-inequalities. We prove the theoretical advantages of these directed models compared to the previously known ILP approaches. We show that our two concepts are equivalent from the polyhedral point of view. On the other hand, our experimental study shows that the cut formulation is much more powerful in practice. Moreover, we propose a collection of benchmark instances that can be used for further research on this topic.

Preliminary parts of this paper appeared in the Proceedings of the European Symposium on Algorithms (ESA) 2007 and Proceedings of the International Conference on Combinatorial Optimization and Applications (COCOA) 2008.

Maria Kandyba was supported by the German Research Foundation (DFG) through the Collaborative Research Center "Computational Intelligence" (SFB 531).

Ivana Ljubić was supported by the Hertha-Firnberg Fellowship of the Austrian Science Foundation (FWF).

Petra Mutzel was partially supported by the Austrian Research Promotion Agency (FFG) under grant 811378 (NetQuest project)

Markus Chimani, Maria Kandyba, Petra Mutzel
Chair XI Algorithm Engineering, TU Dortmund; Otto-Hahn-Str. 14, 44227 Dortmund, Germany
E-mail: {markus.chimani,maria.kandyba,petra.mutzel}@tu-dortmund.de

Ivana Ljubić
Dep. of Statistics and Decision Support Systems, University of Vienna; Brünnerstr. 72, 1210 Vienna, Austria
E-mail: ivana.ljubic@univie.ac.at

1 Introduction

Survivable Network Design problems have been an important research topic in the last decades. Thereby, we ask for a topology design of, e.g., telecommunication networks. The task is to connect a given set of customers using a set of potential route-segments, satisfying certain connectivity requirements, and minimizing the investment costs. Resulting networks may be required to be robust against failures: after a certain number of arbitrarily chosen connection or node failures, alternative connections should still exist in order to guarantee the functionality of the whole network.

1.1 Considered Problems

Recall that a graph is *2-node-connected* (*2-edge-connected*) if after the removal of any node (edge) it is still connected. A maximal 2-node-connected subgraph of a graph—i.e., a *2-node-connected component*—containing more than one edge is referred to as a *block*.

Given an undirected graph $G = (V,E)$, a cost function $c : E \to \mathbb{R}^+$, and a vector of connectivity requirements $\rho \in \{0,1,2\}^{|V|}$. A solution of the $\{0,1,2\}$*-Node-Connected Steiner Network* problem (*2NCON*)[1] [26] is a subgraph $N = (V_N, E_N)$ of G which contains all nodes $v \in V$ with $\rho_v > 0$, minimizes $\sum_{e\in E_N} c_e$ and satisfies the following connectivity property: for every pair of nodes $s,t \in V_N$, N contains $\rho_{st} := \min\{\rho_s, \rho_t\}$ *node-disjoint* paths connecting them. We can relax the problem by replacing the node-disjointness with *edge-disjointness*, and obtain the $\{0,1,2\}$*-Edge-Connected Steiner Network Problem* (*2ECON*). For simplicity, we define $\mathscr{R}_i := \{v \in V \mid \rho_v = i\}$ for all $0 \le i \le 2$, and call the set $\mathscr{R} := \mathscr{R}_1 \cup \mathscr{R}_2$ the *customer nodes*. We can assume that $|\mathscr{R}_2| \ge 2$, since otherwise we obtain the traditional Steiner tree problem. When speaking about both 2ECON and 2NCON problems, we summarize them under the term *2CON*.

Some real-world tasks require a survivable connection between a customer and a special root node $r \in V$. This can be formalized by the *{0,1,2}-root-connected Steiner network problem* (2RSN), which is closely related to the 2NCON but requires the nodewise 2-connectedness only with the root r. Each node $v \in \mathscr{R}$ has to have ρ_v node-disjoint paths with r. Such a root node can represent an important connection hub or an already existing infrastructure network which should be extended by connecting new customers as it was the case in [30,31].

For the latter problem we also consider a prize-collecting variant, called the *2-root-connected prize-collecting Steiner network problem* (2RPCSN). In this case, we are given a prize function $p : \mathscr{R} \to \mathbb{R}^+$, representing the potential gain of revenue if node $v \in \mathscr{R}$ is included into the solution network. The overall goal is to maximize the profit defined as the difference between the gains of the nodes contained in the solution and the total network installation costs. This is equivalent to finding a subgraph $N = (V_N, E_N)$ that minimizes $\sum_{e\in E_N} c_e - \sum_{v\in V_N} p_v$. Thereby, connectivity requirements of those customers taken into the network need to be satisfied as this is the case for the 2RSN.

[1] In the literature there are various names for this problem, with sometimes slightly differing definitions. 2NCON is also known as {0,1,2}-(N)SND (Node Survivable Network Design) and Generalized Steiner Network problem.

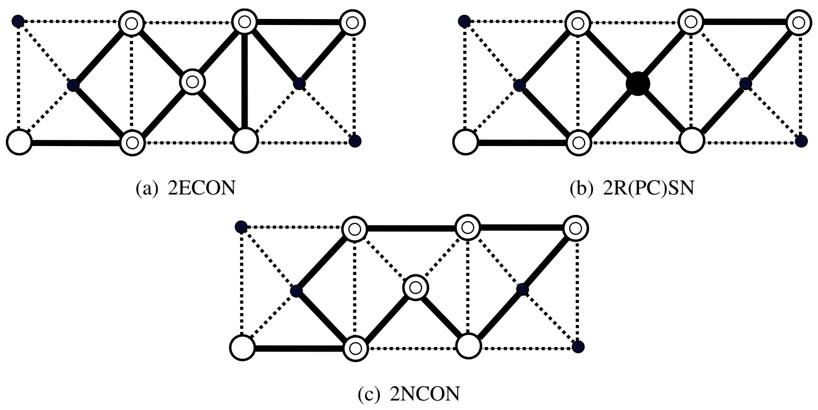


Fig. 1 Feasible networks for different $\{0,1,2\}$-SND problems. Bold edges belong to the solution networks whereas the dashed ones not. Double circles represent $\mathscr{R}_2$ nodes, white circles correspond to $\mathscr{R}_1$ and small circle to $\mathscr{R}_0$. (b) The black node in the middle represents the root r.

In the following, we summarize 2ECON, 2NCON and 2R(PC)SN under term *$\{0,1,2\}$-SND* problems. Figures 1(a), 1(b) and 1(c) illustrate examples of feasible solutions of 2ECON, 2R(PC)SN and 2NCON, respectively.

1.2 Previous Work

$\{0,1,2\}$-SND problems are in general NP-hard, as they, e.g, contain the NP-hard Steiner tree problem as a special case. A lot of research has been conducted on $\{0,1,2\}$-SND problems, both in the fields of effective heuristics and approximation algorithms, see [14] for an overview. Within the scope of this paper, we will concentrate on exact integer linear programming (ILP) formulations. We are motivated by the fact that strong ILP models, when used in conjunction with recent advances in CPU power and ILP solvers, allow to solve large real-world instances for other network problems to provable optimality within reasonable time bounds; see, e.g., [4,23]. In particular, for the case of the k-Cardinality tree problem, it was shown in [4] that for instances with up to 1000 nodes, exact algorithms outperform metaheuristics even in terms of computation time. Furthermore, ILP formulations often form the basis of approximation schemata.

ILPs based on undirected graphs. For 2CON problems Grötschel, Monma and Stoer [10] described cut-based ILPs. The central idea thereby is to express the connectivity requirements by undirected cuts. Wagner et al. [31] formulated an ILP for 2R(PC)SN using basically the same idea[2]. Apart from such formulations, the problem can also be formulated in terms of multi-commodity flow, as done by Raghavan [24] for 2CON and by Wagner et al. in [30] for 2R(PC)SN. The corresponding polytopes for 2CON have been widely investigated and different classes of valid inequalities have been derived, i.p., in papers by Stoer and Raghavan (with coauthors Grötschel, Monma and Magnanti) [10–12,20]. In the following, we will reference their theses [24,26] for simplicity and consistent notations. A survey on polyhedral results can be found in [15].

[2] Although the paper's title uses the term "directed cut", it turns out to be equivalent to the traditional undirected approach discussed in Section 5.

Orientation based ILPs. An *orientation* of an undirected graph G' is a directed graph $\hat{G}$, which is obtained by transforming each edge of G' into a (single) directed arc. For 2ECON, it was shown by Chopra [7] and Raghavan [24] that considering a certain orientability of feasible solutions leads to ILP formulations that are polytope-wise stronger than the undirected formulations mentioned above. These ILPs exploit the following characterization of 2-edge-connected graphs that was shown by Robbins in 1939:

Theorem 1 (Characterization of 2-edge-connected graphs [25]) *An undirected graph $G' = (V', E')$ is 2-edge-connected if and only if there exists an orientation $\hat{G}$ of G' such that for every pair of nodes $u, v \in V'$ there are directed paths $(u \to v)$ and $(v \to u)$ in $\hat{G}$.*

It has been a long-standing open problem whether a similar characterization for 2-node-connected graphs exists. Unfortunately, Theorem 1 cannot be exploited in order to characterize 2-node-connected graphs. This was the main hindrance why there were no orientation-based formulations for $\{0,1,2\}$-SND problems with node-connectivity constraints [24, p. 183],[26, pp. 32,134] .

1.3 Our contribution

In this paper, we mainly consider $\{0,1,2\}$-SND problems with node-connectivity constraints. Whereas problems with edge-connectivity requirements have been widely treated in the literature, providing numerous theoretical and algorithmical results, less results are known regarding the node-connectivity, in general.

The central result of this paper is the characterization of 2-node-connected graphs via orientation properties. This new graph-theoretical result allows us to derive two classes of ILP formulations for 2RSN, 2RPCSN and 2NCON, based on directed graphs: DFLOW is based on multi-commodity flow, DCUT is based on directed cuts. We prove the theoretical advantages of these directed models compared to the previously known ILP approaches. On the other hand, we show that our two concepts are equivalent from the polyhedral point of view. Nonetheless, our experimental study shows that the cut formulation is much more powerful in practice: Based on DCUT, we develop a Branch-and-Cut algorithm for $\{0,1,2\}$-SND problems which allows us to solve test instances with up to 4900 nodes to provable optimality. Moreover, we propose a collection of benchmark instances for $\{0,1,2\}$-SND problems that can be used for further research on this topic.

Although the main focus of this paper lies on the node-connectivity aspect of $\{0,1,2\}$-SND problems, combining our results with the results on 2ECON in [7,24], we are for the first time able to develop a common Branch-and-Cut framework based on directed graphs which solves the 2ECON, 2NCON and 2R(PC)SN problems.

2 Orientation-based Characterization of 2-Node-Connected Graphs

In general, it is not possible to orient an undirected 2-node-connected graph G' such that for every pair of nodes $v, w \in V'$ there are two node-disjoint directed paths, one from v to w and one from w to v. However, choosing a reference node r—called *root node* in the following—and orienting the graph with respect to r allows us to find a general characterization for all 2-node-connected graphs. The result bears some similarity to *st-numbering* (see e.g. [2]), which is defined relative to some edge $\{s,t\}$. A characterization of biconnected graphs using this concept requires a numbering for each edge of the graph, and hence cannot be directly used in the context of orientation-based formulations. Our characterization on the other hand is constructed such that it is directly applicable in our context:

Theorem 2 (Characterization of 2-node-connected graphs) *An undirected graph $G' = (V', E')$ is 2-node-connected if and only if for an arbitrary chosen root node $r \in V'$ there exists an orientation $\hat{G}$ such that the in-degree of the root node is exactly 1 and for each node $v \in V' \setminus \{r\}$, $\hat{G}$ contains a directed path $(r \to v)$ and a directed path $(v \to r)$, which are node-disjoint except for r and v.*

In order to prove this theorem, we first have to introduce an algorithm for which we will show that it establishes such an orientation for a 2-node-connected graph G'.

Definition 1 (Orienting Procedure) We use $\ell : V' \to [0,1] \cup \{\infty\}$ as a labeling function. Initially we set $\ell(v) := \infty$ for all $v \in V$. We start by identifying a simple cycle Z in G' containing r, and orient its edges consistently in one of the two possible directions. We then label each node on Z with increasing fractional numbers between 0 and 1, according to this orientation, starting with $\ell(r) := 0$. Hence, all edges of Z (except its last edge $\hat{e}$) are oriented from the smaller towards the larger label number. We will now orient the remaining undirected edges in such a way that this invariant is valid for all oriented edges:

We define an *augmenting path $P = (a \to b)$* as a simple path of unoriented edges where only the disjoint start and end nodes are labeled, and $\ell(a) < \ell(b)$. To orient G', we repeatedly find an augmenting path $P = (a \to b)$ and orient it from a to b, labeling all inner nodes with increasing fractional numbers greater than $\ell(a)$ but smaller than $\ell(b)$; these labels are to be unique over all labelings so far.

Observation 1 By the above construction, we guarantee that each labeled node has at least one incoming and one outgoing edge. Furthermore, each oriented edge (except for $\hat{e}$) is oriented from the smaller towards the larger label number. Hence, each oriented path will always contain monotonously increasing label numbers (with the exception of $\hat{e}$). This means that any directed circle starting from r and going through any labeled node v will be simple, and we therefore have node-disjoint paths $(r \to v)$ and $(v \to r)$.

Proof (of Theorem 2) We first show that for every 2-node-connected graph $G' = (V', E')$, there exists a valid orientation. We apply the orienting procedure and due to Observation 1, it remains to show that every edge gets oriented by this process. Assume that at some point there is at least one unoriented edge e left, but we cannot find any augmenting path. Clearly, e has to be part of some path $Q = (c \to d)$ of unoriented edges with labeled nodes c and d. Since neither Q nor its reversal is an augmenting path, we have $\ell(c) = \ell(d)$ and therefore $c = d$, i.e., Q is a cycle of unoriented edges, and none of its nodes except for c are labeled. Since G' is 2-node-connected, there has to be an additional unoriented path from some node $q \in Q$ to some labeled node p $(p, q \neq c)$. But then, the path $(p \to q \to c)$ (or its reversal) would be an augmenting path, which is a contradiction.

The orienting procedure used above guarantees that there is only a single edge $\hat{e}$ to be directed towards r. We therefore can observe that the root has an in-degree 1. Hence, the above algorithm correctly orients any 2-node-connected graph.

On the other hand, we show that, given a valid orientation $\hat{G}$, the graph G' is 2-node-connected. For any pair of nodes $u, w \in V'$ there exists a path $(u \rightarrow w)$ that is, e.g., the concatenation of directed paths $(u \rightarrow r)$ and $(r \rightarrow w)$ and analogously a backward path $(w \rightarrow u)$. Due to Theorem 1 the underlying undirected graph G' is hence 2-edge-connected and does not contain any bridges. We show that all nodes $v \in V' \setminus \{r\}$ share a common block with r. Assume that there are at least two blocks B_1 and B_2 containing the nodes $v_1, v_2 \in V' \setminus \{r\}$, respectively. Then r has to be a cut vertex contained in both blocks. But since a valid orientation requires directed paths $(v_1 \rightarrow r)$ and $(v_2 \rightarrow r)$ there have to be at least two edges being directed towards r which is a contradiction to the validity of the orientation $\hat{G}$. Hence we know that $\hat{G}$ will only contain a single block and therefore it is 2-node-connected. □

3 Solution Structure of the {0,1,2}-SND Problems

The above theorem provides us with an important tool to characterize all feasible solutions of our {0,1,2}-SND problems.

Definition 2 Let (G', U) be a tuple of an undirected connected graph $G' = (V', E')$ and $U \subseteq V'$ with $|V'| \geq 3$ and $|U| \geq 2$. G' with respect to U is:

- $(1,2)$-*edge-connected*, if all nodes $u \in U$ lie in the same non-trivial 2-edge-connected component.
- $(1,2)$-*root-node-connected*, if for a given root node $r \in V'$ each node $u \in U$ belongs to a block that also contains r.
- $(1,2)$-*node-connected*, if all nodes $u \in U$ lie in the same block.

Observation 2 Let (G', U) be a tuple as defined above.
- If G' is $(1,2)$-root-node-connected w.r.t. U, then it is also $(1,2)$-edge-connected w.r.t. U.
- G' is $(1,2)$-node-connected w.r.t. U if and only if G' is $(1,2)$-root-node-connected w.r.t. U, for all possible choices of $r \in U$.

Observation 3 Given an instance of a $\{0,1,2\}$-SND problem with customer sets $\mathscr{R}_1$ and $\mathscr{R}_2$. Let N be any solution of the 2ECON, 2RSN, or 2NCON problem. We can observe that $\mathscr{R} \subseteq V_N$ and $(N, \mathscr{R}_2)$ is $(1,2)$-edge-connected, $(1,2)$-root-node-connected, or $(1,2)$-node-connected, respectively.

Due to the existence of Theorem 1, it was possible to give the characterization of feasible 2ECON networks ([7, 24]). We now briefly rephrase it here:

Theorem 3 *Let (G', U) be a tuple as defined above. G' is $(1,2)$-edge-connected with respect to U if and only if for any node $r \in U$ there exists an orientation $\hat{G}$ such that:*
(P1) For each node $v \in V' \setminus U$, $\hat{G}$ contains a directed path $(r \rightarrow v)$.
(P2) For each node $v \in U \setminus \{r\}$, $\hat{G}$ contains a directed path $(r \rightarrow v)$ and a directed path $(v \rightarrow r)$.

With our Theorem 2 we are now able to give a characterization of all feasible solutions for 2RSN and 2NCON, i.e., graphs that contain $\mathscr{R}_1$ and are $(1,2)$-root-node-connected and $(1,2)$-node-connected with respect to $\mathscr{R}_2$.

Theorem 4 *For a given tuple (G', U) and a root node $r \in V'$, G' is $(1,2)$-root-node-connected with respect to U if and only if there exists an orientation $\hat{G}$ of G' that satisfies the properties (P1), (P2) and:*
(P3) For each node $v \in U \setminus \{r\}$ the directed paths $(v \to r)$ and $(r \to v)$ are node-disjoint except for r and v.

Proof Given a valid orientation, it is trivial to show that (G', U) is $(1,2)$-root-node-connected. Hence, assume (G', U) is $(1,2)$-root-node-connected, and we show that there exists a valid orientation $\hat{G}$ of G'. Let B_i, $i \in I$, be the blocks of G' with $B_i \cap U \neq \emptyset$. For each $i \in I$ we have by definition that $r \in B_i$ and clearly, $U \subseteq \bigcup_{i \in I} B_i$. We orient each B_i according to the orienting procedure defined above. Theorem 2 guarantees that for all nodes $v \in \bigcup_{i \in I} B_i$ the properties (P1), (P2) and (P3) are satisfied. The nodes $v \notin \bigcup_{i \in I} B_i$ form subgraphs attached to $\bigcup_{i \in I} B_i$. By starting a DFS from the corresponding cut nodes and orienting all edges from the lower to the higher DFS index, we obtain directed paths $(r \to v)$ for all $v \in V \setminus \{r\}$ which concludes the construction of $\hat{G}$. Note that if G' represents an optimal solution for the 2RSN problem, these subgraphs will form a tree and are therefore easily orientable by a DFS procedure. $\square$

Theorem 5 *Given a tuple (G', U) as described above. G' is $(1,2)$-node-connected with respect to U if and only if for any root node $r \in U$ there exists an orientation $\hat{G}$ of G' with the properties (P1)–(P3) and additionally:*
(P4) In-degree of r is equal to one.

Proof Given a valid orientation, (G', U) is clearly $(1,2)$-node-connected. Hence, assume (G', U) is $(1,2)$-node-connected, and we show that there exists a valid orientation $\hat{G}$ of G'.

By definition, all nodes $u \in U$ lie in the same block. According to Observation 2, for any chosen root $r \in U$ it is also $(1,2)$-root-node-connected with respect to $U \setminus \{r\}$. Using the orienting procedure above for any arbitrarily chosen root $r \in U$, we obtain an orientation satisfying (P1)–(P3). As there is only one block containing all nodes $v \in U$, the orientation procedure guarantees that r has only one ingoing arc. $\square$

4 Orientation-Based ILP Models

We reformulate the 2RSN and 2NCON problems into directed D2RSN and D2NCON problems, respectively, and provide novel ILP formulations based on two different concepts: one of them uses principles of multi-commodity flow, the other one is based on directed cuts. Finally, we modify the cut-based model for the prize-collecting variant 2RPCSN. Generally, both flow- and cut-based concepts are traditional tools in network design problems. Although their practical performance may differ a lot, both variations often turn out to be theoretically equivalent. See, e.g., [8] for a related overview concerning the traditional Steiner tree problem.

Here and in the following sections, we are given a graph $G = (V, E)$ with edge costs c and the sets $\mathscr{R}_1 \dot{\cup} \mathscr{R}_2 = \mathscr{R} \subseteq V$ as described in Section 1.1. Let $\bar{G} = (V, A)$ be the bidirected graph obtained from G by replacing every undirected edge $\{u, v\} \in E$ by two directed arcs $(u, v), (v, u) \in A$ with costs $c_{uv} = c_{vu} = c_{\{u,v\}}$. To model the directed

2NCON problem, we first choose an arbitrary root node $r \in \mathscr{R}_2$. In the case of the directed 2RSN problem the root node r is already specified. As a shorthand, we define $\mathscr{R}'_i := \mathscr{R}_i \setminus \{r\}$ for $0 \leq i \leq 2$, and $\mathscr{R}' := \mathscr{R} \setminus \{r\}$.

The optimal solution of D2RSN is a weight-minimal oriented subgraph $\hat{N} = (V_N, A_N)$ in $\bar{G}$ with $\mathscr{R} \subseteq V_N$ which is $(1,2)$-root-node-connected—i.e., it satisfies the properties (P1)–(P3)—with respect to r and $U = \mathscr{R}_2$. Analogously, if we require $(1,2)$-node-connectedness—i.e., properties (P1)–(P4)—then $\hat{N}$ is an optimal solution of D2NCON. Note that if we ignore both (P3) and (P4), $\hat{N}$ is an optimal solution of the D2ECON problem.

From Theorems 4 and 5 we have:

Corollary 1 *Any feasible solution for 2RSN and 2NCON can be transformed into a corresponding feasible solution for D2RSN and D2NCON, respectively, with the same objective value, and vice versa.*

Remark. One may try to model node-connectivity by only computing edge-connectivity in a modified underlying graph, by replacing each node by a directed arc. This is not valid in our case as the orientability theorems require bidirectedness of the underlying graph.

4.1 A Multi-Commodity Flow Approach

We start with presenting a multi-commodity flow ILP formulation (denoted by DFLOW) for D2RSN, and therefore for 2RSN. We then show how this ILP can be extended to solve the 2NCON problem.

Connecting each customer $v \in \mathscr{R}'$ to a root node r can be expressed by sending exactly one unit of flow from the r to v in $\bar{G}$. To guarantee the redundant connection for each customer $v \in \mathscr{R}'_2$, we send one unit of flow back to the root. Thereby it has to be ensured that the pairs of forward- and backward-flows do not use common nodes and edges except for v and r. The arcs with a positive amount of flow then define our solution network.

We therefore define the set of commodities $\mathscr{C} = \{(r,v) \mid v \in \mathscr{R}'\} \cup \{(v,r) \mid v \in \mathscr{R}'_2\}$. A flow of commodity $\chi \in \mathscr{C}$ on the arc $(i,j) \in A$ is modeled by the continuous variable f^{χ}_{ij}. Finally, we introduce binary variables x_{ij} which are 1, if the solution network contains the arc $(i,j) \in A$ and 0 otherwise.

$$\text{DFLOW}: \qquad \min \sum_{(i,j)\in A} c_{ij} \cdot x_{ij} \tag{1}$$

$$\sum_{i:(i,v)\in A} f^{\chi}_{iv} - \sum_{i:(v,i)\in A} f^{\chi}_{vi} = \begin{cases} -1, \text{ if } v = s \\ 1, \text{ if } v = t \\ 0, \text{ else} \end{cases} \qquad \forall \chi = (s,t) \in \mathscr{C}, \forall v \in V \tag{2}$$

$$\sum_{i:(i,w)\in A} \left(f^{(v,r)}_{iw} + f^{(r,v)}_{iw} \right) \leq 1 \qquad \forall v \in \mathscr{R}'_2, \forall w \in V \setminus \{r,v\} \tag{3}$$

$$0 \leq f^{\chi}_{ij} \leq x_{ij} \qquad \forall (i,j) \in A, \forall \chi \in \mathscr{C} \tag{4}$$

$$x_{vw} + x_{wv} \leq 1 \qquad \forall \{v,w\} \in E \tag{5}$$

$$x_{vw} \in \{0,1\} \qquad \forall (v,w) \in A \tag{6}$$

The *flow-conservation constraints* (2) define the sent flow and ensure the flow balances, while the *coupling constraints* (3) ensure the node-disjointness of the pairs of forward and backward flow, by guaranteeing that at most one unit of flow per commodity pair is sent into any node. The inequalities (4) ensure the flow capacities and bind the flow variables f to the network-defining variables x. For the latter, (5) ensures that we have a unique orientation for the selected edges. Due to Theorem 1, the constraints (2), (4), (5) and (6) ensure that every $\mathscr{R}_2'$ customer belongs to the same edge-biconnected component as r. Theorem 2 and 4, and constraint (3) guarantee that this component is $(1,2)$-root-node-connected with respect to $\mathscr{R}_2'$.

According to Theorem 5, adding the following equation (7) leads to a multi-commodity flow ILP for the 2NCON problem. It ensures that we select only a single arc that is oriented towards the root r:

$$\sum_{(i,r)\in A} x_{ir} = 1. \tag{7}$$

Without (3) and (7), our DFLOW formulation is equivalent to the directed flow formulation for 2ECON originally presented by Raghavan [24, p. 188].

4.2 A Directed Cut Approach

Our second ILP formulation is based on directed cuts; we denote it by DCUT. Its number of variables is independent of $\mathscr{R}$ as it only requires variables x_{ij} for all $(i,j)\in A$. On the other hand, it has an exponential number of constraints. We will see that these are of the traditional cut type and therefore easily and polynomially separable within a Branch-and-Cut approach. A main motivation for DCUT is that cut formulations often outperform flow formulations in practice, as e.g. in [6,17].

Let $S\subset V$, then $\delta^+_G(S) := \{(s,t)\in A \mid s\in S, t\in V\setminus S\}$ and $\delta^-_G(S) := \{(s,t)\in A \mid s\in V\setminus S, t\in S\}$ denote the arcs leaving and entering S, respectively. We may omit the subscript if G is clear from the context. Furthermore, we use the shorthands $G_w := G\setminus\{w\}$, for some $w\in V$, and $x(B) := \sum_{e\in B} x_e$, for some $B\subseteq A$.

$$\text{DCUT}: \qquad \min \sum_{(i,j)\in A} c_{ij}\cdot x_{ij} \tag{8}$$

$$x_{vw}+x_{wv}\le 1 \qquad \forall\{v,w\}\in E \tag{9}$$

$$x(\delta^-(S))\ge 1 \qquad \forall S\subseteq V\setminus\{r\}, S\cap\mathscr{R}'\ne\emptyset \tag{10}$$

$$x(\delta^+(S))\ge 1 \qquad \forall S\subseteq V\setminus\{r\}, S\cap\mathscr{R}_2'\ne\emptyset \tag{11}$$

$$x(\delta^-_{\bar{G}_w}(S_1))+x(\delta^+_{\bar{G}_w}(S_2))\ge 1 \qquad \begin{cases}\forall w\in V\setminus\{r\}, \forall S_1,S_2\subseteq V\setminus\{r,w\},\\ S_1\cap S_2\cap\mathscr{R}_2'\ne\emptyset\end{cases} \tag{12}$$

$$x_{vw}\in\{0,1\} \qquad \forall(v,w)\in A \tag{13}$$

As before, (9) guarantees the unique orientation of chosen edges. The constraints (10) and (11) are called *forward-* and *backward-cuts*, respectively. They ensure the existence of the required paths, and (12) assures the node-disjointness of these paths: After removing any node w, either the forward of the backward path still has to exist.

Thus an optimal solution of the above ILP defines an optimal solution of the 2RSN problem. Analogous to DFLOW, extending DCUT with (7) gives us a cut-based ILP on bidirected graphs for the 2NCON problem. Again, without (12) and (7), our DCUT formulation is equivalent to the directed cut formulation for 2ECON given by Chopra [7].

4.3 Extension to the Prize-Collecting Model

For prize-collecting variants of 2CON problems we cannot easily choose a root node in advance since we do not know any node that will definitively be selected. The situation becomes clear for 2RSN, since even in the prize-collecting setting the prespecified root will always have to be contained in the solution network. We show how the DCUT model can be extended to model this prize-collecting variant 2RPCSN. Observe that we can extend the DFLOW model analogously. As introduced in Section 1.1, we are given a prize p_v for each vertex v.

Additionally to (13), we introduce a second set of binary variables

$$y_v \in \{0,1\}, \forall v \in V.$$

The variable y_v is 1, if v is in the solution network $\hat{N}$, and 0 otherwise. By definition, $y_r = 1$. We obtain the following objective function:

$$\min \sum_{(i,j)\in A} c_{ij} \cdot x_{ij} - \sum_{v \in V} p_v \cdot y_v. \tag{14}$$

We not only require a unique orientation of the edges, but observe that selecting an edge requires both incident nodes to be selected as well:

$$x_{uv} + x_{vu} \le y_v \qquad \forall v \in V, \forall (v,u) \in A. \tag{15}$$

The cut constraints only require a non-zero cut if a vertex in the considered cut set is selected:

$$x(\delta^-(S)) \ge y_v \qquad \forall S \subseteq V \setminus \{r\}, \forall v \in S \cap \mathscr{R} \tag{16}$$

$$x(\delta^+(S)) \ge y_v \qquad \forall S \subseteq V \setminus \{r\}, \forall v \in S \cap \mathscr{R}_2 \tag{17}$$

$$x(\delta^+_{G_w}(S_1)) + x(\delta^-_{G_w}(S_2)) \ge y_v \qquad \begin{cases} \forall w \in V \setminus \{r\}, \forall S_1, S_2 \subseteq V \setminus \{r, w\}, \\ \forall v \in S_1 \cap S_2 \cap \mathscr{R}'_2 \end{cases} \tag{18}$$

5 Polyhedral Comparison

The polyhedral comparison provided in this section is done for 2NCON. The results related to 2R(PC)SN can be derived correspondingly, so we omit their proofs. We first compare the strength of the two concepts proposed above, DCUT and DFLOW. We then compare them with previously known ILP formulations and conclude this section with a hierarchy of the strength of LP relaxations for 2NCON.

5.1 Strength of DCUT and DFLOW

Let

$$\mathscr{P}_{DF} = \{(x,f) \in [0,1]^{|A|} \times [0,1]^{|A| \cdot |\mathscr{R}|} \mid (x,f) \text{ satisfies (2)–(5) and (7)}\}, \text{ and} \tag{19}$$

$$\mathscr{P}_{DC} = \{x \in [0,1]^{|A|} \mid x \text{ satisfies (9)–(12) and (7)}\} \tag{20}$$

be the polyhedra corresponding to LP relaxations of DFLOW and DCUT for 2NCON, respectively. To be able to compare these polyhedra we consider the projection of $\mathscr{P}_{DF}$ into the space of x variables, i.e.,

$$\text{proj}_x(\mathscr{P}_{DF}) = \{x \mid (x,f) \in \mathscr{P}_{DF}\}.$$

Theorem 6 *We have* $\text{proj}_x(\mathscr{P}_{DF}) = \mathscr{P}_{DC}$*, i.e., the* DFLOW *formulation and the* DCUT *formulation are equally strong for 2NCON.*

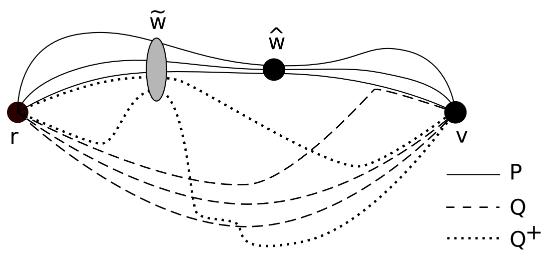
Proof $\underline{\text{proj}_x(\mathscr{P}_{DF}) \subseteq \mathscr{P}_{DC}}$: It is a classical and direct consequence of the max-flow min-cut theorem that if an assignment $(\bar{x},\bar{f})$ for (x,f) is feasible for DFLOW, then $\bar{x}$ is feasible for DCUT, i.e., $\bar{x}$ satisfies the constraints (10)–(12). Assume there is a set $S \subseteq V \setminus \{r\}$ with $v \in S \cap \mathscr{R}'$ and $\bar{x}(\delta^-(S)) < 1$. Then the minimum (r,v)-cut—and therefore the maximum (r,v)-flow—is less than 1. This is a contradiction to the corresponding flow constraint (2) with commodity (r,v). Therefore, the constraints (10) and, analogously also (11), are satisfied.

Let $v \in \mathscr{R}'_2$. Since f satisfies DFLOW, there is exactly one unit of commodity (r,v) going from r to v, and one unit of (v,r) going from v to r: the total amount of flow between r and v is 2. The constraints (3) ensure that deleting any node $w \neq r$ in G can decrease this amount by at most one flow unit. Hence there is an (undirected) max-flow—and therefore a minimum undirected cut—of at least 1 between r and v in any G_w, which induces (12).

$\underline{\mathscr{P}_{DC} \subseteq \text{proj}_x(\mathscr{P}_{DF})}$: We show that if an assignment $\bar{x}$ for x is feasible for DCUT, then there exists a flow $\bar{f}$ such that $(\bar{x},\bar{f})$ is feasible for DFLOW. Clearly, all DFLOW constraints only dealing with x-variables are satisfied as they are identical to the DCUT formulation. It remains to show that we can fit flow into the network using the x-values as capacities. Note that the flows of each commodity are mostly independent of each other, as only the coupling constraints (3) define a dependency between the forward and the backward flow for each $v \in \mathscr{R}'_2$. It is clear that we can find a flow from r to any $v \in \mathscr{R}'_1$, since (10) guarantees a minimum cut between r and v of at least 1. Analogously, because of (10) and (11), we can also find a forward and a backward flow $\left(\bar{f}^{(r,v)}, \bar{f}^{(v,r)}\right)$ for each $v \in \mathscr{R}'_2$, and it remains to show that there always exists such a pair of flows which satisfies (3) for all nodes $w \in V \setminus \{r,v\}$.

Let us assume there exists no such pair of flows. Let $\left(\hat{f}^{(r,v)}, \hat{f}^{(v,r)}\right)$ be the flows satisfying (2) and (4) such that the maximal violation of (3) is minimal. Let $\hat{w}$ be a node where such a maximal violation occurs. The paths used by the flow can then be divided into multiple paths which go through $\hat{w}$ and multiple paths which do not go through $\hat{w}$. We denote these path sets by P and Q, respectively. Let $\alpha > 1$ be the amount of flow over P, and we have $\beta = 2 - \alpha < 1$ as the flow over Q.

Due to (12) we know that there exists a set Q^+ of additional paths not going through $\hat{w}$ over which we can send $\beta^+ > 0$ amount of flow, such that $\beta + \beta^+ = 1$. Consider the flow pair $\left(\tilde{f}^{(r,v)}, \tilde{f}^{(v,r)}\right)$, where the flow over $\hat{w}$



(a)

Fig. 2 Sketch for the proof of Theorem 6, part of $\mathscr{P}_{DC} \subseteq \text{proj}_x(\mathscr{P}_{DF})$.

is only 1, and the second flow unit is sent over that paths of Q and Q^+.[3] Since the original flow was minimal in terms of constraint violation, this new pair of flows has a different node $\tilde{w}$ over which at least α flow units are sent, say $\gamma \geq \alpha$. Clearly, $\tilde{w}$ is part of Q^+. Even if Q^+ contributes all of its flow units to γ, we have $\gamma = \gamma' + \beta^+$ and thus: $\alpha \leq \beta^+ + \gamma' = 1 - \beta + \gamma' = 1 - 2 + \alpha + \gamma' \quad \Longrightarrow \quad 1 \leq \gamma'$

The paths of Q cannot contribute to γ, since then we could modify the original flow $\left(\hat{f}^{(r,v)}, \hat{f}^{(v,r)}\right)$ such that (3) is less violated for $\hat{w}$ (without introducing an additional violation of at least α). Thus γ' has contributions from paths of P. Since the new flow sends exactly 1 unit over P, all paths in P have to go through $\hat{w}$ and $\tilde{w}$: otherwise we could choose a path going through $\hat{w}$ and not through $\tilde{w}$ and further reduce the flow through the latter. Hence, in $\left(\hat{f}^{(r,v)}, \hat{f}^{(v,r)}\right)$ both $\hat{w}$ and $\tilde{w}$ have a through-flow of α, and the paths in P can be subdivided into subpaths between r and $\tilde{w}$, $\tilde{w}$ and $\hat{w}$, and $\hat{w}$ and v, assuming w.l.o.g. that $\tilde{w}$ is closer to r than $\hat{w}$. But then we can iterate the above argument, send only 1 unit of flow through $\hat{w}$ and $\tilde{w}$, and find an additional node $\check{w}$ with too much through-flow. This argument can be iterated ad infinitum, requiring an infinitely large graph. □

By analogous proofs we obtain:

Corollary 2 *The* DFLOW *and* DCUT *formulations are equally strong for 2RSN and for 2RPCSN.*

5.2 Comparison to the Undirected Cut Formulation

We compare our DCUT formulation with the currently best known and widely used cut-formulation presented in [26, p. 14], denoted by UCUT: It is based on undirected cuts in the original (undirected) graph, and uses binary variables z_e, for all $e \in E$, that are set to 1 if the corresponding edge is selected into N, and to 0 otherwise. For each pair of $\mathscr{R}_i$ nodes, it requires all their cuts to be at least i ($i \in \{1,2\}$). For all pairs of $\mathscr{R}_2$ nodes, it further requires all cuts to be at least 1, considering all graphs resulting from removing a single node, in order to ensure 2-node-connectedness. With $\delta(S) = \{\{i,j\} \in E \mid i \in S, j \in V \setminus S\}$, we denote the undirected cut induced by $S \subset V$, and we use $z(F) = \sum_{e \in F} z_e$ for sets $F \subseteq E$.

$$\text{UCUT}: \qquad \min \sum_{e \in E} c_e \cdot z_e \tag{21}$$

[3] We can choose this new pair of flows such that (2) and (4) still holds, since, even if the newly routed flow over Q^+ sends the total amount β^+ in a single direction (say from r to v), we know that $\hat{f}$ satisfies (2) and therefore sends at least $\alpha - 1 = \beta^+$ units into each direction. In the modified flow we can hence remove enough directed flow per direction from P to allow valid flow using Q^+ instead.

$$z(\delta(S)) \geq 1 \qquad \forall S \subseteq V, \emptyset \neq S \cap \mathscr{R}_1 \neq \mathscr{R} \tag{22}$$

$$z(\delta(S)) \geq 2 \qquad \forall S \subseteq V, \emptyset \neq S \cap \mathscr{R}_2 \neq \mathscr{R}_2 \tag{23}$$

$$z(\delta_{G_w}(S)) \geq 1 \qquad \forall w \in V, \forall S \subseteq V, \emptyset \neq S \cap (\mathscr{R}_2 \setminus \{w\}) \neq \mathscr{R}_2 \setminus \{w\} \tag{24}$$

$$z_e \in \{0,1\} \qquad \forall e \in E \tag{25}$$

For 2ECON it is known (cf. [7,24]) that directed formulations are stronger than undirected ones, as long as $\mathscr{R}_1 \neq \emptyset$. For 2NCON, we can also show that our (rooted, directed) DCUT formulation is stronger than the (unrooted, undirected) UCUT formulation. Furthermore, this even holds if $\mathscr{R}_1 = \emptyset$.

Let $\mathscr{P}_{UC}$ be the polyhedron corresponding to the LP-relaxation of UCUT. For $\mathscr{P}_{DC}$, we can use the natural projection

$$\text{proj}_z(\mathscr{P}_{DC}) = \{z \in [0,1]^{|E|} \mid x \in \mathscr{P}_{DC}, z_{ij} = x_{ij} + x_{ji} \; \forall e = \{i,j\} \in E\}.$$

Theorem 7 *We have* $\text{proj}_z(\mathscr{P}_{DC}) \subset \mathscr{P}_{UC}$*, i.e., the* DCUT *formulation is strictly stronger than* UCUT *for 2NCON. This also holds if* $\mathscr{R}_1 = \emptyset$.

Proof We first show that $\text{proj}_z(\mathscr{P}_{DC}) \subseteq \mathscr{P}_{UC}$, i.e., that we can generate the undirected constraints from their directed counterparts. Recall that for any set $S \subset V$ we have $z(\delta(S)) = z(\delta(V \setminus S)) = x(\delta^-(V \setminus S)) + x(\delta^-(V))$. Consider any constraint (22) with its corresponding set S. If $r \in S$, we can consider (10) and have $x(\delta^-(V \setminus S)) \geq 1$. Analogously, if $r \notin S$, we have $x(\delta^-(S)) = x(\delta^+(V \setminus S)) \geq 1$. Therefore, in both cases we have $z(\delta(S)) \geq 1$ and the undirected constraint is hence satisfied.

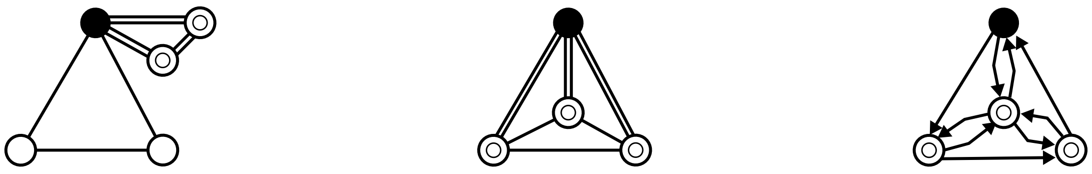
Consider any constraint (23) with its corresponding set S; we show: $z(\delta(S)) = x(\delta^-(S)) + x(\delta^+(S)) \geq 2$. If $r \in V \setminus S$, the inequalities (10) and (11) directly give the above formula. If $r \in S$, we can consider the cut set $V \setminus S$ instead of S, as $z(\delta(S)) = z(\delta(V \setminus S))$. Using the same argument for the graph G_w we can generate the inequalities (24) from the inequalities (12) with $S_1 = S_2$.

It remains to show that $\text{proj}_z(\mathscr{P}_{DC}) \neq \mathscr{P}_{UC}$. We therefore use a triangular graph with $\rho = 1$ for each node. The solution $\bar{z}_e = 0.5$ for each edge e is feasible for UCUT, but there is no corresponding flow in DCUT which would be feasible. We can obtain an example with $|\mathscr{R}_2| \geq 2$ by attaching a feasible network of $\mathscr{R}_2$ nodes to one of the triangle's nodes, cf. Figure 3(a).

Figure 3(b) shows that $\text{proj}_z(\mathscr{P}_{DC}) \neq \mathscr{P}_{UC}$ even if $\mathscr{R}_1 = \emptyset$. The drawing represents a feasible fractional solution $\bar{z} \in \mathscr{P}_{UC}$, whereby all nodes are $\mathscr{R}_2$ customers. The black node denotes the root. Due to the constraint (7) there exists no corresponding feasible solution $x \in \mathscr{P}_{DC}$ with $\bar{x}_{vw} + \bar{x}_{wv} = \bar{z}_{vw}$ for all $\{v,w\} \in E$. $\square$

Note, however, that in general there are instances and root node selections such that both polytopes are equivalent.

An undirected-cut-based formulation that is quite similar to UCUT also exists for 2R(PC)SN. The main difference is that the cuts are defined with respect to a given root. However, it is easy to show that, in general, when $\mathscr{R}_1 \neq \emptyset$ this formulation gives weaker lower bounds compared to our DCUT formulation for 2R(PC)SN [6]. For $\mathscr{R}_1 = \emptyset$, DCUT and UCUT are equivalent for 2R(PC)SN. We can summarize our findings as:


(a) $\mathscr{R}_1 \neq \emptyset$: The network is infeasible for sc DCut, but feasible for UCUT.

(b) $\mathscr{R}_1 = \emptyset$: The network is infeasible for sc DCut, but feasible for UCUT.

(c) The node-partition inequalities are not satisfied.

Fig. 3 In the above figures, the root node is denoted by the black circle. A empty circle denotes a $\mathscr{R}_1$ node, a double circle denotes a $\mathscr{R}_2$ node. Single edges correspond to a fractional solution of 0.5, double edges correspond to a fractional solution of 1.

Corollary 3 *For our considered problem classes, the relationship between* DCUT *and* UCUT *is established as below.* $\surd$ *means that the* DCUT *formulation is stronger,* $\Leftrightarrow$ *means that both are equivalent. The relationships hold true independent on whether* $\mathscr{R}_0 = \emptyset$ *or not; clearly,* $|\mathscr{R}_2| \geq 2$.

	2ECON	2NCON	2RSN, 2RPCSN
$\mathscr{R}_1 = \emptyset$	$\Leftrightarrow$	$\surd$	$\Leftrightarrow$
$\mathscr{R}_1 \neq \emptyset$	$\surd$	$\surd$	$\surd$

5.3 Comparison to the Undirected Flow Formulation

There also exists a flow-based multi-commodity ILP on undirected graphs (UFLOW) which is equivalent to UCUT from the polyhedral point of view [24, p. 26]. It has been strengthened by Raghavan [24, pp. 180–181] to obtain the previously strongest formulation for 2NCON. It uses two multi-commodity flows g and h simultaneously: g represents directed flow for the induced 2ECON problem, h represents an non-oriented flow with node-disjointness constraints.[4] The two flows are bound to each other only by their common use of the z_e variables. We denote Raghavan's formulation as MFLOW (mixed flow) and show that our new formulation is beneficial.

$$\text{MFLOW}: \qquad \min \sum_{e \in E} c_e \cdot z_e \tag{26}$$

[4] Note that this formulation has been developed for general kNCON problems, where it is called *improved undirected flow formulation with node-disjointness constraints*.

$$\sum_{i:(i,v)\in A} g^{\chi}_{iv} - \sum_{i:(v,i)\in A} g^{\chi}_{vi} = \begin{cases} -1, \text{ if } v = s \\ 1, \text{ if } v = t \\ 0, \text{ else} \end{cases} \qquad \forall \chi = (s,t) \in \mathscr{C}, \forall v \in V \tag{27}$$

$$g^{\chi}_{vw} + g^{\chi'}_{wv} \leq z_{vw} \qquad \forall \{v,w\} \in E, \forall \chi, \chi' \in \mathscr{C} \tag{28}$$

$$g^{\chi}_{vw} \geq 0 \qquad \forall (v,w) \in A, \forall \chi \in \mathscr{C} \tag{29}$$

$$\sum_{i:(i,v)\in A} h^{\chi}_{iv} - \sum_{i:(v,i)\in A} h^{\chi}_{vi} = \begin{cases} -2, \text{ if } v = s \\ 2, \text{ if } v = t \\ 0, \text{ else} \end{cases} \qquad \forall \chi = (s,t) \in \mathscr{D}, \forall v \in V \tag{30}$$

$$0 \leq h^{\chi}_{vw} \leq z_{vw} \qquad \forall (v,w) \in A, \forall \chi \in \mathscr{D} \tag{31}$$

$$\textstyle\sum_{i:(v,i)\in A} h^{\chi}_{vi} \leq 1 \qquad \forall \chi = (s,t) \in \mathscr{D}, v \in V \setminus \{\chi\} \tag{32}$$

$$z_e \in \{0,1\} \qquad \forall e \in E \tag{33}$$

Thereby, we consider an arbitrary ordering $\langle v_1, v_2, v_3, \dots \rangle$ of the nodes of $\mathscr{R}_2$ and define $v_0 := v_{|\mathscr{R}_2|}$. We then obtain the "cyclic" commodity set $\mathscr{D} := \{(v_i, v_{i+1}) \mid 0 \leq i < |\mathscr{R}_2|\}$ [24, pp. 89–92]. The set of commodities $\mathscr{C}$ is defined as in DFLOW.

Let $\mathscr{P}_{MF}$ be the polyhedron of the feasible solutions of the LP-relaxation MFLOW. We consider the projections of $\mathscr{P}_{DF}$ and $\mathscr{P}_{MF}$ into the space of z variables, i.e.:

$$\text{proj}_z(\mathscr{P}_{DF}) = \{z \in [0,1]^{|E|} \mid (x,f) \in \mathscr{P}_{DF},\ z_{ij} = x_{ij} + x_{ji}\ \forall \{i,j\} \in E\}, \text{ and} \tag{34}$$

$$\text{proj}_z(\mathscr{P}_{MF}) = \{z \in [0,1]^{|E|} \mid (z,g,h) \in \mathscr{P}_{MF}\}. \tag{35}$$

We also consider extended projections including the flow variables $f \in [0,1]^{|A|\cdot|\mathscr{C}|}$, i.e., variables not in the objective function. Let

$$\text{proj}_{z,f}(\mathscr{P}_{DF}) = \{(z,f) \mid (x,f) \in \mathscr{P}_{DF}, z_e = x_{ij} + x_{ji}\ \forall e = \{i,j\} \in E\}$$

be the projection of $\mathscr{P}_{DF}$ into the variable space of z and retaining the flow f. Let

$$\text{proj}_{z,f}(\mathscr{P}_{MF}) = \{(z,f) \mid (z,g,h) \in \mathscr{P}_{MF}, f = g\}$$

be the projection of $\mathscr{P}_{MF}$ ignoring the h flow. In other words, we identify the flows f and g.

We show that the lower bounds obtained by the LP-relaxations of our DFLOW formulation are at least as tight as those of the mixed flow formulation. Therefore note that the flow f is a kind of natural fusion of the flow g and the node-disjointness properties of h.

Theorem 8 DFLOW *is at least as strong as* MFLOW, *i.e.,* $\text{proj}_z(\mathscr{P}_{DF}) \subseteq \text{proj}_z(\mathscr{P}_{MF})$. *Furthermore we even have* $\text{proj}_{z,f}(\mathscr{P}_{DF}) \subset \text{proj}_{z,f}(\mathscr{P}_{MF})$.

Proof We show that for any feasible solution $(\bar{x}, \bar{f})$ of DFLOW we can obtain a feasible solution $(\bar{z}, \bar{g}, \bar{h})$ of MFLOW, using $\text{proj}_{z,f}$ as described above. Based on these projections, it is easy to see that (27), (28), (29), and (33) are satisfied. It remains to show that we can always find a feasible flow solution h within the network G with the projected edge capacities z. Let $\chi = (s,t) \in \mathscr{D}$. If $\chi \in \mathscr{C}$, we can choose $\bar{h}^{\chi}_e := \bar{f}^{\chi}_e + \bar{f}^{(t,s)}_e$, which satisfies (30), (31), (32) due to (4) and proj_z.

Assume $\chi \notin \mathscr{C}$. We look at the maximum (s,t)-flow in G with capacities z and consider any corresponding minimum (s,t)-cut; let S be the cut set containing s, and $V \setminus S$ contains t. W.l.o.g. assume that $r \in S$. Since (2) is satisfied for the commodities (r,t) and (t,r), the maximum undirected (r,t)-flow, and therefore also the maximum undirected (s,t)-flow h^{χ}, is at least 2. Assume we cannot send h^{χ} without violating (32). Then there exists a single node w such that the total capacity κ of the cut edges which do not send their flow through w is less than one. If $w \neq r$, (3) guarantees that we can send two (undirected) flow units between r and t whereby at most one unit is sent through w. This is a contradiction to κ. If $w = r$, (7) guarantees an in-flow into r of exactly 1 for both the (s,r) and the (t,r) flow. Hence, using the two 1-flows $\bar{f}^{(s,r)}$ and $\bar{f}^{(t,r)}$ we can send an undirected flow of at least 1 between s and t without using r, which is again a contradiction to κ.

To establish the second claim it is enough to construct a feasible flow g in MFLOW which sends more than one unit into the root r. This is infeasible for DFLOW as (7) is violated. □

DFLOW requires less variables and constraints than MFLOW, hence:

Observation 4 DFLOW is more compact than MFLOW.

Our formulation answers the question by Raghavan [24, p. 183] whether his flow variables g, h can be bounded together more tightly. Note that Theorem 2 is crucial for the validity of our approach, which explains why this compact formulation could not be used legitimately before.

5.4 Additional Cut-Constraints

Recall that UCUT without (24) is the traditional undirected cut formulation for 2ECON. It has been shown in [26] that the latter formulation can be strengthened by adding certain classes of valid inequalities that are NP-hard to separate. Chopra [7] showed that his directed 2ECON cut formulation inherently includes one of these classes, namely the *partition inequalities*; in [26, pp. 130–134] it was shown that the latter formulation also includes the class of the (polynomially separable) *Prodon inequalities*. Moreover, Raghavan showed that his improved undirected multi-commodity flow formulation for kECON, which for $k = 2$ is equivalent to both directed flow and directed cut formulations for 2ECON, also includes the *odd-hole inequalities* and the *combinatorial-design inequalities* [24, pp. 165–180]. By dropping the constraints (12) and (7) from DCUT, we obtain the directed cut formulation for 2ECON. Hence we can conclude:

Proposition 1 DCUT *and* DFLOW *inherently ensure the validity of the partition, Prodon, odd-hole, and combinatorial-design inequalities.*

For a node $v \in V$ let $\{W_1, \dots, W_p\}$ be a proper partition of $V \setminus \{v\}$ into p non-empty sets such that $W_i \cap \mathscr{R}_2 \neq \emptyset$ for each $1 \leq i \leq p$. The *node-partition inequalities*—i.e., undirected partition inequalities where one node is removed from the graph—strengthen the UCUT formulation and can be written as:

$$z(\delta_{G_v}(W_1, \dots, W_p)) \geq p - 1 \tag{36}$$

Thereby, $\delta_{G_v}(W_1,\dots,W_p)$ defines the set of edges separating the sets $W_1,\dots,W_p$ in the graph $G-v$.

These inequalities were first proposed by [9] and then generalized in [26, pp. 91–94]. It was an open question [24, p. 183] whether MFLOW would induce this constraint class. Our following result constitutes a negative answer for this question:

Proposition 2 *In general, none of the above formulations induces the node-partition inequalities* (36).

Proof See Figure 3(c) for an example, which denotes the x variables of the arcs. Once more, we use the natural projection $x_{vw}+x_{wv}=z_{vw}$ for all $\{v,w\}\in E$ to obtain an undirected network: When removing the central node, the partition inequality with three partition sets is violated. Yet the solution is feasible for the LP-relaxation of DCUT. $\square$

Some of the undirected node-partition inequalities are, however, inherently included in DCUT for 2NCON:

Proposition 3 *Let $\bar{x}\in\mathscr{P}_{DC}$. Then $\bar{x}$ satisfies the node-partition inequalities for valid partitions of $V\setminus\{r\}$.*

Proof Consider any partition $W_1,\dots,W_p$, $p\geq 2$, of the node set $V\setminus\{r\}$ such that $W_i\cap\mathscr{R}_2\neq\emptyset$. From (10) und (7) it follows:

$$z(\delta_{G_r}(W_1,\dots,W_p)) = \sum_{1\leq i\leq p} x(\delta^+(W_i)) - x(\delta^+(r)) \geq p-1. \quad \square$$

Observation 5 For nodes $v\in\mathscr{R}_2$ with $\deg(v)=2$ in G, the node-partition inequalities are always induced. This holds even for UCUT. Hence, such nodes in general do not represent good choices as the root nodes in DCUT, as we gain more by choosing another $\mathscr{R}_2$ node with larger degree.

We obtain the following hierarchy of formulations for 2NCON:

Corollary 4 *The following hierarchical scheme summarizes the relationships between the LP-relaxations of the ILP models considered throughout this paper for 2NCON.*

$$\begin{tikzcd}
& \boxed{\text{DCut+}} & \\
\boxed{\text{DCut}} \arrow[ur] \arrow[rr, -{Triangle[open]}, {Triangle[open]}-] & & \boxed{\text{DFlow}} \arrow[ul] \\
& \boxed{\text{MFlow}} \arrow[ul, -{Triangle[open]}] \arrow[ur, -{Triangle[open]}] & \\
& \boxed{\text{UCut+}} \arrow[u] & \\
\boxed{\text{UCut}} \arrow[ur] \arrow[rr, -{Triangle[open]}, {Triangle[open]}-] & & \boxed{\text{UFlow}} \arrow[ul]
\end{tikzcd}$$

A filled arrow specifies that the target formulation is strictly stronger than the source formulation. An empty arrow specifies that the target formulation is at least as strong as the source formulation. Thereby, UCut+ *denotes* UCUT *with partition inequalities, and* DCut+ *denotes* DCUT *with node partition inequalities.*

6 The Branch-and-Cut Algorithm

Based on the DCUT approach we developed a Branch-and-Cut code. For a general description of the Branch-and-Cut scheme see, e.g., [32]: In such an algorithm, we start with an initial *partial LP*, i.e., the ILP without the integrality properties and only considering a certain subset of all constraints. We solve the partial LP in order to obtain a *current fractional solution*. A *separation routine* then tries to identify constraints of the full constraint set of the ILP which the current fractional solution violates. We add these constraints to our partial LP and reiterate these steps. If at some point we cannot find any violated constraints, we have to resort to *branching*, i.e., we generate two disjoint subproblems, e.g., by fixing a variable to 0 or 1. By using the fractional solutions as lower bounds, and some heuristic solution as an upper bound (cf. last paragraph of this section), we can prune irrelevant subproblems. In every node of the resulting Branch-and-Bound tree, we apply the separation strategy again.

From the point of formulation strength, using DFLOW instead of DCUT might seem like a reasonable choice in general, as both the number of variables and constraints are bound by a polynomial. But in practice, the latter has certain advantages: it requires much less variables, especially when $\mathscr{R}$ is large. Furthermore, its drawback of an exponential number of constraints can turn out to be beneficial, as the actual computation of an optimal solution will in general not require all of these constraints. The required constraints in DCUT can be easily separated using simple polynomial max-flow algorithms, see below. Hence, we can obtain the optimal fractional solution of the LP-relaxation at the root of the Branch-and-Bound tree in polynomial time, based on the theorem regarding the equivalence of optimization and separation; cf., e.g., [32].

We use the same Branch-and-Cut strategy for all problems, 2R(PC)SN and 2NCON.

Initialization. We start with the constraints (9) and the subset of the constraints (10), (11) for $|S| = 1$. Even for 2NCON and 2RSN, we use an extended DCUT formulation, involving y_v binary variables for $v \in \mathscr{R}_0$. Although they do not strengthen the DCUT model, the following *flow-preservation* constraints significantly speed up the computation time:

$$\sum_{k:(k,i)\in A} x_{ki} \geq y_i \text{ and } \sum_{k:(i,k)\in A} x_{ik} \geq y_i, \quad \forall i \in \mathscr{R}_0.$$

These inequalities specify that no node from $\mathscr{R}_0$ will ever have only incoming or only outgoing arcs. They are especially useful for instances with few customers and comparably long paths connecting them in the optimal solution.

Separation. The cut constraints (10) can be separated in polynomial time via the traditional max-flow separation scheme: after obtaining some LP-relaxation for our partial ILP, we compute the maximum flow from r to each $v \in \mathscr{R}$ in $\bar{G}$ using the arc values of the current solution as capacities. If the resulting value is less then one (or y_v, in case of 2RPCSN), we extract one or more of the induced minimum r-v-cuts—considering *nested-* and *reversal-cuts*, as, e.g., described in [18,16]—and add the corresponding constraints to our ILP model. The cut constraints (11) can be separated analogously.

If there are no violated constraints of type (10) or (11), we solve the separation problem for the constraints of type (12) in an analogous way: for each node $v \in \mathscr{R}_2$ and for each node $w \in V$, $w \neq v$ we compute both the v-r and

r-v maximal flows in $\bar{G}_w$. If the sum of these flows is less than one (or y_v, for 2RPCSN), we add the corresponding inequalities.

Furthermore, to speed up the separation of node-disjointness constraints, we use the following idea: let us consider an integer solution where the constraints (10) and (11) are valid, i.e., we have edge-disjoint paths $(r \to v)$ and $(v \to r)$ for any $v \in \mathscr{R}_2$. If we assume that these paths have a common node w, then there are at least two incoming and two outgoing edges at w. Therefore, when separating constraints (12), we first try nodes $w \in V$ with $\bar{x}(\delta^-(w)) > 1$ and $\bar{x}(\delta^+(w)) > 1$ in our fractional solution. Finally, it turns out to be beneficial to select the cut sets containing the smallest number of arcs among all violated cuts of type (10), (11) or (12). In fact, this simple property is crucial for solving large graphs efficiently in practice.

Primal Heuristic. A fractional solution of an LP-relaxation is used to construct a feasible solution, thus obtaining upper bounds for the optimal solution. We proceed in three steps: after all customers $\mathscr{R}$ are connected by a Steiner tree, we ensure 2-node-connectivity by extending the current solution with shortest paths from r to all $\mathscr{R}_2$ nodes. In a local improvement step, we remove redundant edges, or chains of edges, that decrease the cost without violating the connectivity of the solution. For more details on our primal heuristic see [5,6].

7 Computational Results

We implemented both formulations DCUT and DFLOW—able to solve 2ECON, 2NCON, 2RSN, and 2RPCSN—using CPLEX 9.0's Branch-and-Bound framework. The additionally necessary separation routines for DCUT are implemented in C++ using LEDA 5.1.1 and the efficient max-flow algorithm of [3]. All tests were performed on an Intel Xeon 2.33Ghz CPU with 2GB of RAM per process, and a time limit of 2 hours per problem instance.

Our computational study concentrates on the instances of the $\{0,1,2\}$-SND problems with node-connectivity requirements for which the directed formulations are stronger from polyhedral point of view, cf. Corollary 3. In general, until now only few computational results for $\{0,1,2\}$-SND problems with node-connectivity requirements were published in literature and there is no common benchmark set of instances for them. Therefore, one of our additional aims was to create such a benchmark library.[5] We therefore collected the available test instances used by various authors for different problem settings and included them in our TSNDLib (*Topological Survivable Network Design Library*) [27]. On the cited webpage, one can also find information on the computational results conducted on each of the benchmark sets. In the following section, we briefly describe these instances and discuss their usefulness for our computational study.

Rationale. In this section, we give a brief overview on the experiments performed for all the combinations of instance sets, formulations and problem classes. The focus of this paper is to formally establish the soundness and strength of orientation-based formulations for node-connectivity, and show its general applicability in practice.

[5] A known library of test instances SNDLib [22] provides instances for survivable *capacitated* network design problems, i.e., problems where the aim is not only to topologically design a network, but demand routing and capacity issues have to be considered.

We therefore only describe the main findings of these experiments. More thorough analysis specific to certain problem classes can be found in [5,6].

7.1 Benchmark Instances

Complete euclidean graphs. In [15], Kerivin et al. published their results on the 2-node-connected spanning network problem, the special case of 2NCON where $\mathscr{R}_2 = V$. Thereby, they used graphs from the TSPLib [28] which are complete graphs with euclidean distances as edge costs. For these special problems instances, the optimal 2-edge-connected solution is also 2-node-connected [21]. Therefore, the 2ECON model is sufficient for solving them and it follows from Corollary 3 that in this case the (more compact) undirected model is preferable from theoretical point of view. Hence we will not consider these problems in the following.

However, most real-world applications of {0,1,2}-SNDP seem to be based on rather sparse graphs [1,26]. For the 2R(PC)SN problems, Bachhiesl [1] proposed the following three different benchmark sets that also have been used in [30,31]:

ClgS and ClgM. These instances use the real-world access net data of the city district Cologne-Ossendorf. For our experiments we consider the small (ClgS) and medium (ClgM) sized instance sets. Thereby, each set contains 25 instances. The underlying graphs have 190 nodes and 377 edges, and 1757 nodes and 3877 edges, respectively. The instances differ in the customer nodes, and have 3–6 $\mathscr{R}_1$, and 2–3 $\mathscr{R}_2$ customers.

Grid. This set contains artificial instances based on grid graphs with 100, 400, 900,..., 4900 nodes. For each graph size there are 2×15 instances, using two different cost functions, respectively. They have 5–13 $\mathscr{R}_1$ and 3–8 $\mathscr{R}_2$ customers.

PCSTLib$^+$. The PCSTLib benchmark [13], was used in several studies, e.g., [18,19], and contains graphs divided into two groups K and P, where 15%–27% and 34%–50% of the nodes are customers, respectively. The former are similar to street map layouts. In each group, the underlying graphs have 100 and 400 nodes. For the {0,1,2}-SND problems, PCSTLib has been augmented to PCSTLib$^+$ such that roughly $1/3$ of the customer nodes are selected to be in $\mathscr{R}_2$.

The above three benchmark sets can be used not only for the 2R(PC)SN but also for the 2NCON problem, interpreting the given root node as an $\mathscr{R}_2$ customer. Due to the diversity of these instances and the partial real-world aspect, we take them as a basis for our computational study. As the original Cologne and Grid instances have rather few customers—which seems unusual in practice—we also generated modified instances:

ClgS$^+$ These are the ClgS instances, selecting 20% (10 % $\mathscr{R}_1$ and 10% $\mathscr{R}_2$) of the nodes as customers.

Grid$^+$ These are the Grid instances, selecting 20% (10 % $\mathscr{R}_1$ and 10% $\mathscr{R}_2$) of the nodes as customers.

Besides, Wagner [29] proposed an additional artificial benchmark set:

TSPLIB$^+$. Considering the aforementioned TSPLib, these instances are corresponding euclidean Delauney triangulations on varying graph sizes where 25% (10%) of all nodes are $\mathscr{R}_1$ ($\mathscr{R}_2$) customers.

Problem	ILP	ClgS	ClgS$^+$	G 100	G 400	G 900	G 1600	G$^+$ 100	K 100	P 100
2NCON	**DCut**	**0.1**	**0.6**	**0.2**	**2.0**	**22.9**	**74.2**	**0.3**	**0.7**	**0.5**
	DFlow	0.3	446.3	7.0	226.0	1505	(6209)*	22.4	19.9	1500
2RSN	**DCut**	**0.07**	**0.4**	**0.08**	**1.3**	**17.36**	**94.0**	**0.09**	**1.0**	**0.7**
	DFlow	0.4	154.5	6.4	240.2	1554	—	30	13.08	2194
2RPCSN	**DCut**	**0.06**	**0.02**	**0.2**	**2.5**	**34.5**	**114.0**	**0.06**	**0.4**	**0.5**
	DFlow	0.3	7.63	5.94	261.3	1778	—	8.6	19.1	1136

Table 1 Average CPU time in seconds. **G**($^+$) denotes the Grid($^+$) instances; **K** and **P** denote the groups of PCSTLib$^+$. (*) DFLOW solves 67% of the instances. None of the instances not in this table could be solved by DFLOW within 2 hours.

For the 2NCON problem, Stoer [26] used a set of sparse real-world instances with up to 116 nodes. Unfortunately, this data is not available anymore[6]. On the other hand, our results indicate that such small and sparse networks are usually solved within less than a second.

7.2 Comparison of DFLOW and DCUT

We first compare the performance of a default Branch-and-Bound algorithm based on our compact DFLOW formulation to our Branch-and-Cut algorithm based on DCUT. To obtain an unskewed comparison, we turned off all automatic cut-generation etc., usually performed by CPLEX. Our experiments show that DCUT outperforms DFLOW in terms of running time and success ratio on all instance sets and for all considered $\{0,1,2\}$-SND problems.

Running times. See Table 1 for the overview of the corresponding average running times. We only report on instances which could—at least in part—be solved to optimality by both DCUT and DFLOW. Table 2 shows the results for the other (larger) instances that could be solved only by DCUT.

We observe that the runtime performance of DCUT is quite similar on the different problem classes 2NCON, 2RSN and 2RPCSN. The same holds for DFLOW. All instances with less than 200 nodes are solved to optimality by DCUT in less than a second on average. The only instance set where both algorithms perform comparably well is ClgS, which is due to the fact that the underlying LPs of DFLOW are rather small due to the small number of customers, and the overhead of DCUT's cut separation routines is comparably expensive. Note that neither approach requires any branching for ClgS and small Grid instances. Already a slight increase in the number of customers is sufficient for DCUT to outperform DFLOW, see, e.g., the results for Grid instances of size 100. This effect is further amplified by larger underlying graphs, as this results in an even larger increase of variables for DFLOW. While the cut approach is able to solve all Grid instances with up to 3600 nodes to optimality within 1 hour, the largest instances which can be completely solved by DFLOW in 2 hours contain 900 nodes. We see that, due to their high number of customers, DCUT is highly advantageous even for small graphs of PCSTLib$^+$: for the *P* group with 100 nodes, it is 2000–3000 times faster than DFLOW.

For the TSPLib$^+$ instances, the findings are analogous to the ones reported before, as only DCUT was able to solve instances. Interestingly, DCUT solves all instances with up to 1000 nodes within two hours in the context of

[6] Personal communication.

Problem		ClgM		G 2500	G 3600	G 4900		G^+ 400	K 400		P 400
2NCON	average	(969)	24/25	334	930	(3216)	29/30	170	(38)	3/6	260
	median	(56)		219	6871	(2763)		39	(41)		29
2RSN	average	(828)	24/25	143	628	(2547)	24/30	29	(130)	3/6	142
	median	(53)		109	480	(2156)		8.0	(52)		
2RPCSN	average	(643)	24/25	153	658	(3014)	25/30	7.0	382	11/11	243
	median	(33)		83	494	(2356)		3.6	55		202

Table 2 Average and median running time for instances only solved by DCUT. Times in brackets denote that not all instances could be solved within 2 hours: the corresponding second column gives the ratio of solved instances. None of these instances left of the triple vertical line require branching. For PCSTLib$^+$, we report only on feasible instances.

2RPCSN, while some of these instances turned to be more difficult for the 2RSN and 2NCON settings. Figure 4 shows the respective running times of DCUT for the different problem settings.

LP-relaxations. A common measure to assess and compare ILP formulations is to look at the lower bounds resulting from their LP-relaxations, i.e., the solution at the root node of the branch-and-bound tree (LP_r). In our case, these values are identical as the corresponding polytopes are equivalent, cf. Theorem 6.

DCUT also outperforms DFLOW in terms of running times needed to solve the (equivalent) LP-relaxation. When DFLOW is not able to solve the given instance to optimality within 2 hours, it is due to a large size of the LP and the most part of the computation time is needed to solve the root relaxation. By contrast, when branching is required, DCUT uses only a comparably small percentage of the total running time to solve the root relaxation. For the Grid$^+$ instances with 400 nodes, DFLOW cannot even solve the first LP-relaxation within the given time bound. DCUT, on the other hand, requires only 170 seconds on average to solve the ILP, whereby the LP-relaxation is solved within 10–30 seconds. In Figure 5 we visualize these observations w.r.t. 2NCON and the PCSTLib$^+$ instances. The results for 2R(PC)SN are analogous.

7.3 Analysis of DCut Performance

As the performance of our DCUT algorithm is similar for different problem settings, we only consider its performance for the most prominent 2NCON problem in the following. We observe that the LP-relaxation of our ILPs usually gives a strong lower bound. For many instances—i.p. all Grid and ClgS instances—the relaxation already gives an integer, and thus optimal, solution. In Table 3, we report on the quality and time (t_{LP}) of the solutions at the root node, i.e., the LP-relaxation of the full model. For each set we compute the average relative **gap** $:= \frac{(\text{OPT}-\text{LP}_r)}{\text{OPT}}$ in percent, whereby OPT denotes the optimal objective value of the ILP. Additionally, we give the average total runtime t_{ILP}, the average percentage of instances that require branching (req.br.), and the average number of Branch-and-Bound nodes (#BB).

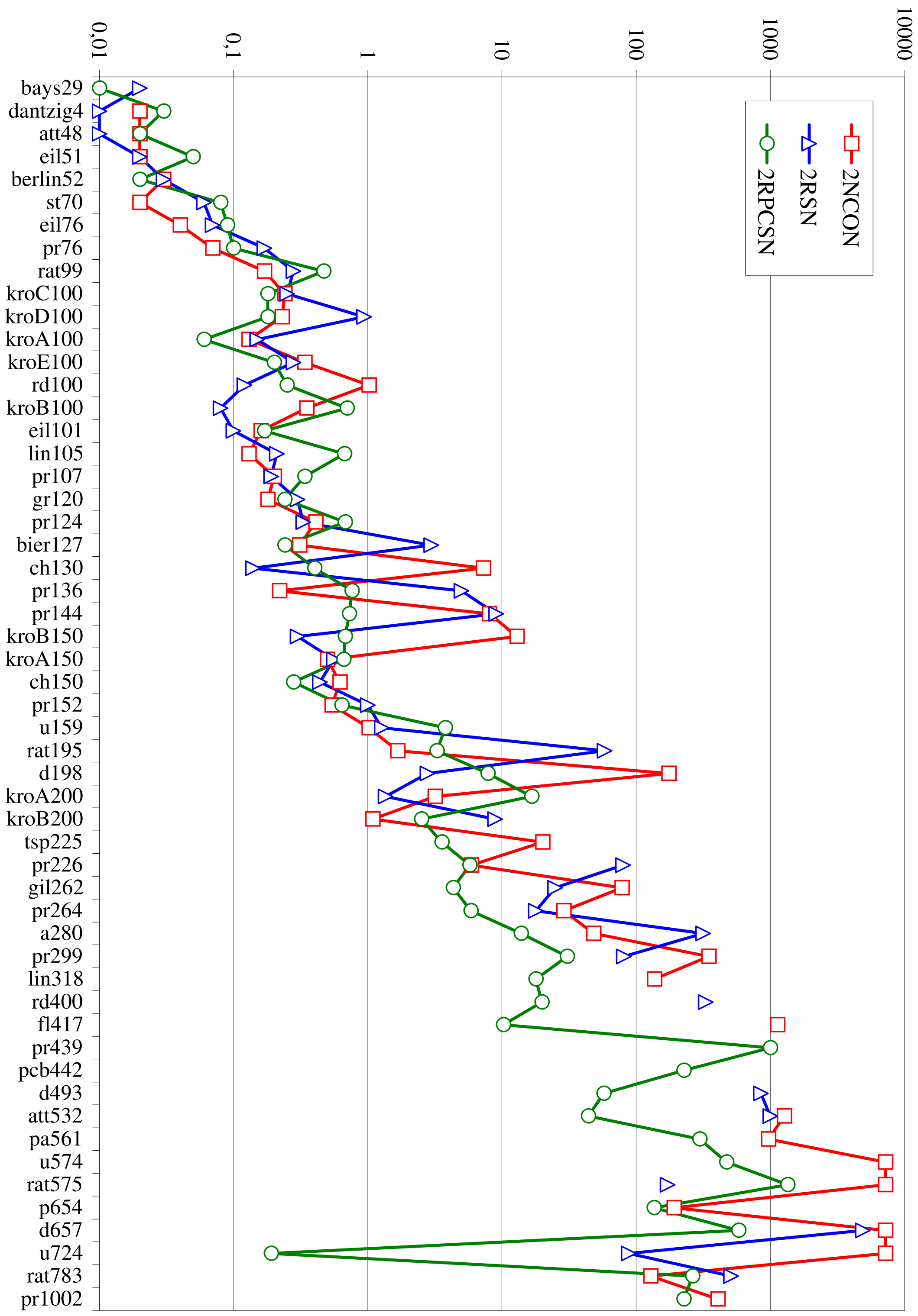


Fig. 4 Running time (in seconds) of DCut for TSPLib$^+$, comparing different problem settings. Missing data points denote that the corresponding problem was not solved to provable optimality within 2 hours.

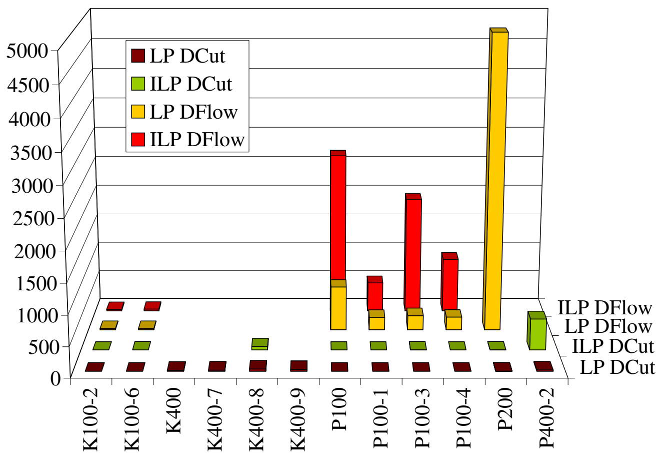


Fig. 5 Consider 2NCON for PCSTLib$^+$. Time required (in seconds) to solve the DFLOW and DCUT ILPs and their corresponding LP-relaxations. Data points are missing, when the computation exceeded the 2 hours timeout limit.

	ClgS$^+$	**G$^+$ 100**	**K 100**	**P 100**	**G$^+$ 400**	**K 400***		**P 400**
t_{ILP}	**0.6**	**0.3**	**0.7**	**0.5**	**170**	**38.1**	**—**	**233**
t_{LP}	0.4	0.2	0.6	0.3	21.0	32.3	22.5	27.1
gap	0.12%	0.16%	0.06%	0.18%	0.35%	0.12%	(0.53%)	0.34%
req.br.	42.1%	16.7%	22.2%	100%	100%	33.3%	100%	100%
#BB	1.0	2.0	1.9	17.8	101.3	5.33	(641.3)	253

Table 3 Results for 2NCON computed via DCUT, when branching was required. (*) 50% solved (left column); the right column gives the statistics of the unsolved instances after 2h, using a heuristic upper bound to estimate the gap.

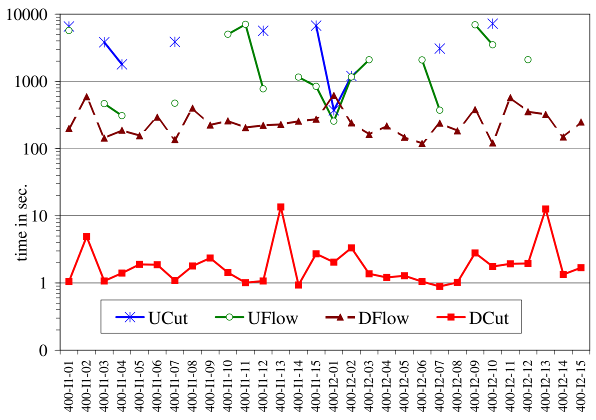
7.4 Directed vs. Undirected Models for 2RPCSN

For the 2RPCSN problem we compared our results for DFLOW and DCUT with the running times of the undirected formulations published in [30,31][7]. For this setting, DCUT clearly outperforms undirected formulations: It solves all Grid instances with up to 3600 nodes and most of the instances with 4900 nodes to provable optimality. For the previous approaches, the largest solvable instance has 400 nodes and the required running times are much longer, cf. Figure 6. See [6] for more details on this comparison.

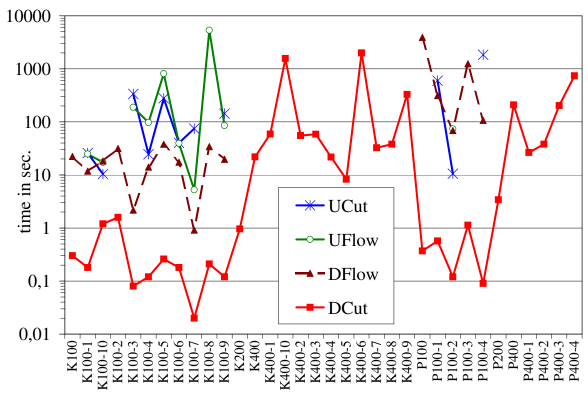
8 Conclusion

In this paper, we showed a new graph-theoretical orientation property for 2-node-connected graphs, and we demonstrated how this can be exploited to obtain new, provably stronger ILP formulations for various classes of {0,1,2}-SND problems. Furthermore, we showed that using the orientation-based formulations is beneficial in practice. To this ends, we introduced a collection *TSNDLib* of known benchmark sets to allow standardized comparisons of various and future approaches. Although the focus of this paper was on the node-connectivity aspect of {0,1,2}-SND problems, our Branch-and-Cut implementation is the first one that solves arbitrary 2ECON, 2NCON and 2R(PC)SN problems using strong directed models.

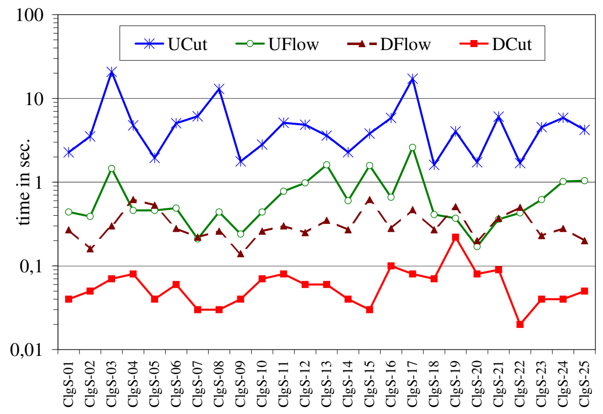
[7] The algorithms by Wagner et al. were run on a stronger Intel Xeon 3.6 GHz machine with CPLEX 10.0.1 and LEDA 5.1



(a) Grid instances with 400 nodes



(b) PCSTLib$^+$ instances



(c) *ClgS* instances

Fig. 6 Diagrams comparing UCUT and UFLOW approaches to our DCUT and DFLOW formulations.

Acknowledgment. We want to thank Daniel Wagner for running his UCUT and UFLOW implementations on our benchmark instances, and Mechthild Opperud, née Stoer, for insightful discussions.

References

1. P. Bachhiesl. The OPT- and the SST-problems for real world access network design – basic definitions and test instances. Working Report NetQuest 01/2005, Carinthia Tech Institute, Klagenfurt, Austria, 2005.
2. U. Brandes. Eager *st*-ordering. In *Proceedings of the 10th European Symposium on Algorithms (ESA 02)*, volume 2461 of *LNCS*, pages 247–256. Springer, 2002.
3. B. V. Cherkassky and A. V. Goldberg. On implementing push-relabel method for the maximum flow problem. *Algorithmica*, 19:390–410, 1997.
4. M. Chimani, M. Kandyba, I. Ljubić, and P. Mutzel. Obtaining optimal *k*-cardinality trees fast. *Journal of Experimental Algorithmics*. Accepted. A preliminary version appeared in: Proceedings of the Ninth Workshop on Algorithm Engineering and Algorithms (ALENEX 2008), pages 27–36, SIAM, 2008.
5. M. Chimani, M. Kandyba, I. Ljubić, and P. Mutzel. Strong formulations for 2-node-connected Steiner network problems. In *Proceedings of the 2nd Annual International Conference on Combinatorial Optimization and Applications (COCOA 2008)*, volume 5165 of *LNCS*, pages 190–200. Springer, 2008.

6. M. Chimani, M. Kandyba, and P. Mutzel. A new ILP formulation for 2-root-connected prize-collecting Steiner networks. In *Proceedings of the 15th European Symposium on Algorithm (ESA 2007)*, volume 4698 of *LNCS*, pages 681–692. Springer, 2007.
7. S. Chopra. Polyhedra of the equivalent subgraph problem and some edge connectivity problems. *SIAM J. Discrete Math.*, 5(3):321–337, 1992.
8. M. X. Goemans and Y. Myung. A catalog of Steiner tree formulations. *Networks*, 23:19–28, 1993.
9. M. Grötschel and C. L. Monma. Integer polyhedra arising from certain network design problems with connectivity constraints. *SIAM J. Discret. Math.*, 3(4):502–523, 1990.
10. M. Grötschel, C. L. Monma, and M. Stoer. Polyhedral Approaches to Network Survivability. In *Reliability of Computer and Communication Networks, Proc. Workshop 1989*, volume 5 of *Discrete Mathematics and Theoretical Computer Science*, pages 121–141. American Mathematical Society, 1991.
11. M. Grötschel, C. L. Monma, and M. Stoer. Computational results with a cutting plane algorithm for designing communication networks with low-connectivity constraints. *Operatios Research*, 40(2):309–330, 1992.
12. M. Grötschel, C. L. Monma, and M. Stoer. Facets for polyhedra arising in the design of communication networks with low-connectivity constraints. *SIAM Journal on Optimization*, 2(3):474–504, 1992.
13. D. S. Johnson, M. Minkoff, and S. Phillips. The prize-collecting steiner tree problem: Theory and practice. In *Proceedings of 11th ACM-SIAM Symposium on Distcrete Algorithms*, pages 760–769, 2000.
14. H. Kerivin and A. R. Mahjoub. Design of survivable networks: A survey. *Networks*, 46(1):1–21, 2005.
15. H. Kerivin, A. R. Mahjoub, and C. Nocq. (1,2)-Survivable networks: facets and branch-and-cut. In M. Grötschel, editor, *The Sharpest Cut*, MPS-SIAM Series in Optimization, pages 121–152. SIAM, 2004.
16. T. Koch and A. Martin. Solving steiner tree problems in graphs to optimality. *Networks*, 32(3):207–232, 1998.
17. I. Ljubić. *Exact and Memetic Algorithms for Two Network Design Problems*. PhD thesis, TU Vienna, 2004.
18. I. Ljubić, R. Weiskircher, U. Pferschy, G. Klau, P. Mutzel, and M. Fischetti. An algorithmic framework for the exact solution of the prize-collecting steiner tree problem. *Math. Prog., Ser. B*, 105(2–3):427–449, 2006.
19. A. Lucena and M. G. C. Resende. Strong lower bounds for the prize-collecting steiner problem in graphs. *Discrete Applied Mathematics*, 141(1-3):277–294, 2003.
20. T. L. Magnanti and S. Raghavan. Strong formulations for network design problems with connectivity requirements. *Networks*, 45(2):61–79, 2005.
21. C. L. Monma, B. S. Munson, and W. R. Pulleyblank. Minimum-weight two-connected spanning networks. *Math. Program.*, 46(2):153–171, 1990.
22. S. Orlowski, M. Pióro, A. Tomaszewski, and R. Wessäly. SNDlib 1.0–Survivable Network Design Library. In *Proceedings of the 3rd International Network Optimization Conference (INOC 2007)*, 2007. http://sndlib.zib.de.
23. T. Polzin and S. V. Daneshmand. Improved algorithms for the Steiner problem in networks. *Discrete Applied Mathematics*, 112(1-3):263–300, 2001.
24. S. Raghavan. *Formulations and Algorithms for the Network Design Problems with Connectivity Requirements*. PhD thesis, MIT, Cambridge, MA, 1995.
25. H.E. Robbins. A theorem on graphs with an application to a problem of traffic control. *American Mathematical Monthly*, 46:281–283, 1939.
26. M. Stoer. *Design of Survivable Networks*, volume 1531 of *LNCS*. Springer, 1992.
27. TSNDLib: Collection of benchmark instances for Topological $\{0,1,2\}$-Survivable Network Design problems, 2008. `http://ls11-www.cs.tu-dortmund.de/TSNDLib/`.
28. TSPLIB. `http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/`.
29. D. Wagner. Generierung und Adaptierung von Testinstanzen für das OPT und SST Problem. Technical Report 03/2007, Carinthia Tech Institute, Klagenfurt, Austria, 2007. In german.
30. D. Wagner, G. R. Raidl, U. Pferschy, P. Mutzel, and P. Bachhiesl. A multi-commodity flow approach for the design of the last mile in real-world fiber optic networks. In *Proc. OR '06*, pages 197–202. Springer, 2006.
31. D. Wagner, G. R. Raidl, U. Pferschy, P. Mutzel, and P. Bachhiesl. A directed cut for the design of the last mile in real-world fiber optic networks. In *Proceedings of the International Network Optimization Conference 2007*, 2007.
32. L. A. Wolsey. *Integer Programming*. Wiley-Interscience, 1998.