Math. Program., Ser. A 91: 307–348 (2002)

Digital Object Identifier (DOI) 10.1007/s101070100209

P. Bauer · J.T. Linderoth · M.W.P. Savelsbergh

A branch and cut approach to the cardinality constrained circuit problem

Received: April 1998 / Accepted: October 2000
Published online October 26, 2001 –

Abstract. The Cardinality Constrained Circuit Problem (CCCP) is the problem of finding a minimum cost circuit in a graph where the circuit is constrained to have at most k edges. The CCCP is NP-Hard. We present classes of facet-inducing inequalities for the convex hull of feasible circuits, and a branch-and-cut solution approach using these inequalities.

Key words. cardinality constrained circuit problem – circuit polytope – branch and cut

1. Introduction

In the knapsack constrained circuit problem (KCCP), we are given an undirected graph $G = (V, E)$, a cost c_e for each edge $e \in E$, a weight $w_v \geq 0$ for each vertex $v \in V$, and an integer k. The objective is to find a minimum cost circuit (i.e., a simple cycle) subject to the constraint that the sum of the weights on the vertices in the circuit is at most k. The KCCP is easily seen to be NP-hard, because when we subtract a sufficiently large constant from the cost of each edge and set $k = \sum_{v \in V} w_v$, we obtain a traveling salesman problem.

Although the KCCP is an interesting optimization problem, its importance to us stems from the fact that it can be used to model the pricing problem in branch-and-price algorithms for the vehicle routing problem. For a comprehensive discussion of branch-and-price algorithms for vehicle routing problems the reader is referred to [12].

In branch-and-price algorithms for vehicle routing problems, the pricing problem is usually solved by dynamic programming, i.e., multi-label shortest path algorithms. Solving the pricing problem by a branch-and-cut algorithm, rather than a dynamic programming algorithm, may have several computational advantages. First, good feasible solutions, corresponding to columns with a negative reduced cost, may be found quicker because multi-label shortest path algorithms find feasible solutions only when the sink node is labeled, which may take a long time if the underlying network is large. Secondly, using a branch-and-cut algorithm, it is not necessary to solve the problem to optimality to show that no negative reduced column exists. As soon as the global lower bound becomes nonnegative, we know the optimal solution will be nonnegative as well. Finally,

P. Bauer: Siemens AG, Corporate Research and Development, ZT SE 4, 81730 Munich, Germany
e-mail: Petra.Bauer@mchp.siemens.de

J.T. Linderoth: Axioma, Inc., 501-F Johnson Ferry Road, Marietta, GA 30068, USA
e-mail: jlinderoth@axiomainc.com

M.W.P. Savelsbergh: School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0205, USA, e-mail: mwps@isye.gatech.edu

if the state space cannot be pruned by restrictions such as time windows, a dynamic programming approach begins to resemble exhaustive search.

We are interested in investigating the advantages and disadvantages of using a branch-and-cut algorithm, rather than using dynamic programming, to solve the pricing problem within a branch-and-price algorithm for vehicle routing problems.

Developing a branch-and-cut algorithm for the KCCP is also interesting from another perspective. The polytope defining the set of feasible solutions to the KCCP is the intersection of two other polyhedra, namely the knapsack polytope and the circuit polytope. We know a lot about the structure of both these polyhedra, and it is interesting to learn more about the value of this knowledge when it comes to developing a branch-and-cut algorithm for the KCCP.

To facilitate our investigation, we have decided to start with the special case of unit weights, i.e., the cardinality constrained circuit problem (CCCP). The CCCP models the pricing problem that arises in branch-and-price algorithms for the vehicle routing problem with unit demands.

Before we discuss problems that are related to the KCCP and the CCCP, we first present a transformation that converts node related information to edge related information. This transformation allows us to look at the problems from different perspectives. The weight w_v for $v \in V$ can be placed on the edges by introducing an edge weight $w_e = 0.5(w_u + w_v)$ for all $e = \{u, v\} \in E$ and requiring that the sum of the weights on the edges in the circuit be at most k.

In the capacitated prize collecting traveling salesman problem (CPCTSP) [9], we are given an undirected graph $G = (V, E)$, a travel cost t_e for $e \in E$, a reward p_v and weight w_v for $v \in V$, a depot node $v_0 \in V$, and a capacity $W \in \mathbb{Z}_+$. The objective is to find a route, or set of edges, R starting and ending in v_0 that maximizes the collected rewards minus the incurred travel costs, i.e., $\sum_{v \in V(R)} p_v - \sum_{e \in R} t_e$, subject to the constraint that the total weight on the route, i.e., $\sum_{v \in V(R)} w_v$, does not exceed the capacity W.

In the orienteering problem (OP), we are given a graph $G = (V, E)$, rewards p_v for $v \in V$, a depot node $v_0 \in V$, travel costs t_e for $e \in E$, and an upper bound $Q \in \mathbb{Z}_+$. The objective is to find a route R starting and ending at v_0 that maximizes the total collected reward ($\sum_{v \in V(R)} p_v$) subject to the constraint that the total travel cost ($\sum_{e \in R} t_e$) is less than Q. Heuristics for solving the OP are given by Golden et al. [17] and Ramesh et al. [34]; and polyhedral approaches for OP are given by Fischetti et al. [14] and Leifer and Rosenwein [26].

Applying the transformation presented above, we see that the CPCTSP and OP are equivalent problems. Furthermore, if we remove the requirement that the route goes through the depot node, then the CPCTSP and OP are also equivalent to the KCCP.

In the prize collecting traveling salesman problem (PCTSP) [3], there is a reward for visiting a node and penalty for not visiting a node. The objective is to minimize the sum of the penalties and the travel costs subject to the constraint that the total collected reward should be greater than or equal to a given minimum. A heuristic for the PCTSP in which the prize requirement is not considered is given by Bienstock et al. [8]. Polyhedral based approaches to the PCTSP are presented by Balas [3] and Pillai [32].

If we remove the knapsack constraint from the KCCP, we are left with the weighted girth problem (WGP) or circuit problem (CP), where we are trying to find a minimum cost circuit in a graph. Bauer [5,6] studies the WGP problem in great detail, presents

facet defining inequalities for its underlying polyhedron, and provides a branch-and-cut approach for its solution. Wang [39] examines both the CP and the closely related Eulerian subtour problem.

Nguyen and Maurras [28,29] study the k-cycle polyhedron. Given a complete undirected graph, the k-cycle polyhedron is the the convex hull of the incidence vectors of the cycles of length exactly k. The k-cycle polyhedron is a face of the CCCP polyhedron. Hartmann and Ozluk [21] give a polyhedral analysis of the directed variant of the k-cycle polyhedron.

The remainder of the paper is organized as follows. In Sect. 2, we give two integer programming formulations for the CCCP and present basic results on the facial structure of the polyhedra associated with the convex hulls of feasible solutions for both formulations. In Sect. 3, we show that many facet inducing inequalities for the WGP polyhedron are also facet inducing for the CCCP polyhedron. In Sect. 4, we derive new classes of facets for the CCCP polyhedron. Section 5 discusses the complexity of the separation problem for the various classes of facet inducing inequalities and presents separation algorithms and heuristics. In Sect. 6, we provide details and computational results of a branch-and-cut algorithm for the CCCP.

2. Integer programming formulations of the CCCP

In order to ease the exposition, we will first introduce a few definitions. For $V' \subseteq V$, define

$$E(V') \equiv \{(i, j) \in E : i \in V', j \in V'\}, \text{ and}$$
$$\delta(V') \equiv \{(i, j) \in E : i \in V', j \notin V'\}.$$

In addition, we will write $\delta(v)$ instead of $\delta(\{v\})$ for $v \in V$. For a given subset of edges $E' \subseteq E$, we use the notation

$$V(E') \equiv \{v \in V : E' \cap \delta(v) \neq \emptyset\}$$

to define the set of nodes spanned by E'.

To formulate the CCCP, we use decision variables $x_e, e \in E$, and $y_v, v \in V$, to describe a circuit C with the following meanings:

$$x_e = \begin{cases} 1 & \text{if } e \in C, \\ 0 & \text{otherwise,} \end{cases}$$

and

$$y_v = \begin{cases} 1 & \text{if } v \in V(C), \\ 0 & \text{otherwise.} \end{cases}$$

For notational convenience, we often write $x(E')$ to denote $\sum_{e \in E'} x_e$ for a set of edges $E' \subseteq E$, and $y(V')$ to denote $\sum_{v \in V'} y_v$ for a set of vertices $V' \subseteq V$. Also, for two

subsets $S, T \subseteq E$, $S \cap T = \emptyset$, we let

$$x(S:T) = \sum_{s \in S, t \in T} x_{s,t}.$$

An integer programming formulation of the CCCP can be given as follows:

Minimize

$$\sum_{e \in E} c_e x_e \tag{2.1}$$

subject to

$$\begin{aligned}
x(\delta(v)) &= 2y_v && \forall v \in V, && (2.2)\\
x(\delta(S)) &\geq 2(y_u + y_v - 1) && \forall S \subset V,\ 3 \leq |S| \leq n-3, \\
& && u \in S,\ v \in V \setminus S, && (2.3)\\
x(E) &\geq 3, && && (2.4)\\
x(E) &\leq k, && && (2.5)\\
x_e &\in \{0,1\} && \forall e \in E, && (2.6)\\
y_v &\in \{0,1\} && \forall v \in V. && (2.7)
\end{aligned}$$

In this formulation, the *degree equations* (2.2) ensure that a feasible solution goes exactly once through each visited node, and the *disjoint circuit elimination constraints* (2.3) make sure that our solution is a connected circuit. Constraint (2.4) eliminates the null circuit, constraint (2.5) is the *cardinality constraint*, and constraints (2.6) and (2.7) give the integrality conditions on our variables.

Let $\mathcal{C}_n$ be the set of circuits of K_n, the complete graph on n nodes, and let χ^C be the incidence vector of a circuit C. We are interested in studying the *cardinality constrained circuit polytope*

$$\begin{aligned}
P_C^{n,k} &= \text{conv}\{(\chi^C, \chi^{V(C)})^T \in \mathbb{R}^{|E|+|V|} | C \in \mathcal{C}_n, |C| \leq k\} \\
&= \text{conv}\{(x,y)^T \in \mathbb{R}^{|E|+|V|} | (x,y) \text{ satisfies } (2.2) - (2.7)\}.
\end{aligned}$$

$P_C^{n,k}$ is contained in the intersection of the two polytopes:

$$\begin{aligned}
P_C^n &= \text{conv}\{(x,y) \in \mathbb{R}^{|E|+|V|} | (x,y) \text{ satisfies } (2.2) - (2.4), (2.6) - (2.7)\} \text{ and} \\
P_k^n &= \text{conv}\{x \in \mathbb{R}^{|E|} | x \text{ satisfies } (2.5) \text{ and } (2.6)\}.
\end{aligned}$$

Hence, any valid inequality for P_C^n or P_k^n is also valid for $P_C^{n,k}$. We will show that in many cases facet defining inequalities for P_C^n and P_k^n are also facet defining for $P_C^{n,k}$.

By substituting out the node variables $y_v (v \in V)$ using (2.2), we obtain a formulation of the CCCP that uses only edge variables x_e $(e \in E)$.

Minimize

$$\sum_{e \in E} c_e x_e$$

subject to

$$x(\delta(v)) \le 2 \qquad \forall v \in V, \tag{2.8}$$

$$x(\delta(v) \setminus e) - x_e \ge 0 \qquad \forall v \in V, e \in \delta(v), \tag{2.9}$$

$$x_e + x((u:T)) + x((v:S)) - x((S:T)) \le 2 \qquad \forall e = (u,v) \in E \text{ such that } S, T \text{ is a partition of } V \setminus \{u, v\}, |S|, |T| \ge 2, \tag{2.10}$$

$$x(E) \ge 3, \tag{2.11}$$

$$x(E) \le k, \tag{2.12}$$

$$x_e \in \{0, 1\} \ \forall e \in E. \tag{2.13}$$

The *degree constraints* (2.8) and the *parity constraints* (2.9) ensure that every vertex has degree zero or two. The *disjoint circuit elimination constraints* (2.10) ensure our circuit is connected. Since there are no node variables in this formulation, the associated cardinality constrained circuit polytope and circuit polytope have different definitions:

$$\begin{aligned} \tilde{P}_C^{n,k} &= \text{conv}\{\chi^C \in \mathbb{R}^{|E|} \mid C \in \mathcal{C}_n, |C| \le k\} \\ &= \{x \in \mathbb{R}^{|E|} | x \text{ satisfies } (2.8) - (2.13)\} \end{aligned}$$

and

$$\tilde{P}_C^n = \text{conv}\{\chi^C \in \mathbb{R}^E \mid C \in \mathcal{C}_n\}.$$

Bauer [5,6] and Wang [39] have studied the facial structure of $\tilde{P}_C^n$ and we will frequently use their results.

Our next goal is to show some properties of the polytopes introduced above and to establish relations between them. Similar proofs appear in [9].

Theorem 1. *For* $4 \le k \le n$, $dim(\tilde{P}_C^n) = dim(P_C^n) = dim(\tilde{P}_C^{n,k}) = dim(P_C^{n,k}) = |E| = n(n-1)/2$.

Proof. Bauer [5] and Wang [39] establish that $\dim(\tilde{P}_C^n) = |E|$. Their proofs use circuits of at most length four, so $\dim(\tilde{P}_C^{n,k}) = |E|$. The rank of the set of equalities $(x(\delta(v)) = 2y_v \ \forall v \in V)$ is $|V|$; thus, $\dim(P_C^n) \le |E|$ and $\dim(P_C^{n,k}) \le |E|$. The proofs of Bauer and Wang show that there are $|E| + 1$ circuits (of length at most 4) whose incidence vectors $x \in \mathbb{R}^{|E|}$ are affinely independent. For these same circuits, the incidence vectors in terms of edge and node variables $(x, y)^T \in \mathbb{R}^{|E|+|V|}$ are also affinely independent, so $\dim(P_C^n) \ge |E|$ and $\dim(P_C^{n,k}) \ge |E|$.

□

Since the two formulations of the CCCP describe the same set of feasible circuits, we would also suspect that their polyhedra have the same facets.

Theorem 2. *If the inequality* $a^T x \le a_0$ *is facet defining for* $\tilde{P}_C^{n,k}$, *it is also facet defining for* $P_C^{n,k}$. *If* $b^T x + d^T y \le b_0$ *is facet defining for* $P_C^{n,k}$, *then* $h^T x \le b_0$, *where* $h = (h)_{ij} \equiv b_{ij} + \frac{1}{2}(d_i + d_j)$ *is facet defining for* $\tilde{P}_C^{n,k}$.

Proof. If $a^T x \leq a_0$ is facet defining for $\tilde{P}_C^{n,k}$, there are $|E|$ affinely independent circuits that satisfy $a^T x = a_0$. These same $|E|$ circuits written in terms of edge and node variables are also affinely independent and satisfy the equality $a^T x = a_0$. If $b^T x + d^T y \leq b_0$ is facet defining for $P_C^{n,k}$, then there are $|E|$ affinely independent circuits $(x^1, y^1)^T, (x^2, y^2)^T, \ldots, (x^{|E|}, y^{|E|})^T \in \mathbb{R}^{|E|+|V|}$ such that $b^T x^j + d^T y^j = b_0 \; \forall j = 1, 2, \ldots |E|$. As $y_v = x(\delta(v))/2 \; \forall v \in V$, we find by substitution that these circuits also satisfy $h^T x = b_0$. Further, the incidence vectors of circuits $x^1, x^2, \ldots x^{|E|}$ are affinely independent, for if not, we would have

$$|E| > \text{rank} \begin{bmatrix} x^1 & x^2 & \ldots & x^{|E|} \\ -1 & -1 & \ldots & -1 \end{bmatrix} = \text{rank} \begin{bmatrix} x^1 & x^2 & \ldots & x^{|E|} \\ y^1 & y^2 & \ldots & y^{|E|} \\ -1 & -1 & \ldots & -1 \end{bmatrix} \tag{2.14}$$

which would imply that the original circuits $(x^1, y^1)^T, (x^2, y^2)^T, \ldots, (x^{|E|}, y^{|E|})^T \in \mathbb{R}^{|E|+|V|}$ were not affinely independent.

□

Since the polyhedra are the same, we use only the notation $P_C^{n,k}$ when referencing either polyhedron for the remainder of the paper.

Let us introduce the following two polytopes which are closely related to P_C^n and $P_C^{n,k}$. For a node set $K \subseteq V$, we let

$$P_C^K = \text{conv}\{(\chi^C, \chi^{V(C)})^T \in \mathbb{R}^{|E(K)|+|K|} \mid C \text{ is a circuit in } G(K) = (K, E(K))\}$$

and

$$\begin{aligned} P_C^{n,K} &= \text{conv}\{(\chi^C, \chi^{V(C)})^T \in \mathbb{R}^{|E|+|V|} \mid C \in \mathcal{C}_n, \; C \text{ covers only nodes in } G(K)\} \\ &= \text{conv}\{(\chi^C, \chi^{V(C)})^T \in \mathbb{R}^{|E|+|V|} \mid C \in \mathcal{C}_n, \; x_e = 0 \text{ for all } e \notin E(K), \\ &\qquad y_v = 0 \text{ for all } v \notin K\}. \end{aligned}$$

The following lemma will be useful in helping us characterize when facet defining inequalities for P_C^n are also facet defining for $P_C^{n,k}$.

Lemma 1. *Let $4 \leq k < n$ and let $ax + fy \leq a_0$ be facet defining for P_C^n. Suppose there is a set $K \subseteq V$ with $|K| \leq k$ such that the restriction $\tilde{a}x + \tilde{f}y \leq a_0$ of $ax + fy \leq a_0$ to $G(K) = (K, E(K))$ is facet defining for P_C^K. Moreover, assume that for every $e \in E \setminus E(K)$ there is a circuit $C \in \mathcal{C}_n$ with $e \in C$, $|C| \leq k$ and $a\chi^C + f\chi^{V(C)} = a_0$. Then $ax + fy \leq a_0$ is facet defining for $P_C^{n,k}$.*

Proof. Since $\tilde{a}x + \tilde{f}y \leq a_0$ defines a facet of P_C^K, we can conclude that $ax + fy \leq a_0$ defines a facet of $P_C^{n,K}$. With $P_C^{n,K} = P_C^{n,k} \cap \{(x, y)^T \in \mathbb{R}^{E+V} \mid x_e = 0 \text{ for all e} \in E \setminus E(K), \; y_v = 0 \text{ for all } v \in K\}$, the claim follows from the existence of a circuit $C \in \mathcal{C}_n$ with $e \in C$, $|C| \leq k$, and $a\chi^C + f\chi^{V(C)} = a_0 \; \forall e \in E \setminus E(K)$.

□

Bauer [5] has characterized when the basic inequalities defining P_C^n are facet inducing. Because of Theorem 2 and the fact the proofs require only circuits of lengths 3 and 4, we have

Theorem 3.

(i) *The* **trivial inequalities** $x_e \geq 0$, $e \in E$, *define facets of* $P_C^{n,k}$ *for* $n \geq 5$ *and* $k \geq 4$.

(ii) *The* **degree constraints** $x(\delta(v)) \leq 2$, $v \in V$, *define facets of* $P_C^{n,k}$ *for* $n \geq 5$ *and* $k \geq 4$.

(iii) *The* **parity constraints** $x(\delta(v) \setminus x_e) - x_e \geq 0$, $v \in V, e \in \delta(v)$, *define facets of* $P_C^{n,k}$, $n \geq 5$ *and* $k \geq 4$.

(iv) *Let* $e = (u, v) \in E$, *and let* S, T *be a partition of* $V \setminus \{u, v\}$ *with* $S, T \geq 2$. *Then the* **disjoint circuit elimination constraint**

$$x_e + x((u : T)) + x((v : S)) - x((S : T)) \leq 2$$

define facets of $P_C^{n,k}$, $n \geq 6$ *and* $k \geq 4$.

(v) *The inequality* $x(E) \geq 3$ *defines a facet of* $P_C^{n,k}$, $n \geq 5$ *and* $k \geq 4$.

Next, we show that the cardinality constraint, which is facet inducing for P_k^n is also facet inducing for $P_C^{n,k}$.

Theorem 4. *Let* $4 \leq k < n$. *Then the* **cardinality constraint**

$$x(E) \leq k$$

is facet defining for $P_C^{n,k}$.

Proof. Assume that there is an inequality $bx \leq b_0$, $b \in \mathbb{R}^{|E|}$, $b \neq 0$, which is valid for $P_C^{n,k}$ and satisfies $\{x \in P_C^{n,k} \mid x(E) = k\} \subseteq \{x \in P_C^{n,k} \mid bx = b_0\}$.

Let $f = (u, v)$ and $g = (w, z)$ be any two nonadjacent edges in E, define $h = (v, w)$, $l = (u, z)$, and let C be a circuit of cardinality k containing the edges f and h, but not the node z. With $C' = C \setminus \{f, h\} \cup \{g, l\}$, we have $b\chi^C = b\chi^{C'} = b_0$ and thus

$$b_f + b_h = b_g + b_l.$$

Analogously, we can derive

$$b_g + b_h = b_l + b_f$$

and get $b_g = b_f$. Since for any two adjacent edges, we can find an edge which is not adjacent to either of them, we get the same coefficient for all edges $e \in E$. This immediately yields that $bx \leq b_0$ is a positive multiple of $x(E) \leq k$.

□

In the next sections, we introduce valid inequalities and facets for the polyhderon $P_C^{n,k}$ that do not explicitly appear as inequalities in the integer programming formulation. In Sect. 3, we give conditions under which some facets of P_C^n are also facets of $P_C^{n,k}$. In Sect. 4, we derive valid inequalities and facets for $P_C^{n,k}$ which are not valid inequalities for P_C^n. Some of the these are obtained from lifting facets of P_C^n that define only lower dimensional faces of $P_C^{n,k}$, and others are obtained independently.

3. Inequalities from the circuit polytope

This section, included for completeness, notes some known facets of P_C^n that are also facets for $P_C^{n,k}$. The *cut inequalities* are facet inducing for both P_C^n and $P_C^{n,k}$ for $n \geq 7$. The *forest inequalities* are facet defining for P_C^n for $n \geq 7$, and facet inducing for $P_C^{n,k}$ if an easy to check condition is satisfied.

3.1. The cut inequalities

The *cut inequalities*, introduced by Seymour [38], generalize the parity constraints (2.9). They are shown to be facet defining for P_C^n, $n \geq 2$, by Bauer [5]. Her proof involves only circuits of length 3 and 4 and thus immediately yields the following theorem.

Theorem 5. *Let $n \geq 7$ and $k \geq 4$. For $S \subseteq V$, $3 \leq |S| \leq n-3$, and $e \in \delta(S)$, the* **cut inequality**

$$x(\delta(S) \setminus e) - x_e \geq 0$$

is facet defining for $P_C^{n,k}$.

3.2. The forest inequalities

Bauer [5,6] gives several classes of facet defining inequalities for P_C^n that are derived using the fact that the traveling salesman polytope is a face of the circuit polytope. Among them are the *forest inequalities*, which are obtained from the facet inducing clique tree inequalities of Grötschel and Pulleyblank [19]. It turns out that, if an easy to check condition on the "size" of the inequality is satisfied, the inequality is also a facet for $P_C^{n,k}$.

A *clique tree* is a connected graph composed of cliques which satisfy the following properties (in the following a clique tree is always considered a subgraph of K_n):

(1) The cliques are partitioned into two sets, the set of *handles* and the set of *teeth*.
(2) No two teeth intersect.
(3) No two handles intersect.
(4) Each tooth contains at least two nodes, at most $n-2$ nodes, and at least one node not belonging to any handle.
(5) For each handle, the number of teeth intersecting it is odd and at least three.
(6) If a tooth and a handle intersect, then their intersection is an articulation set of the clique tree.

Suppose we are given a clique tree with handles $H_1, \ldots, H_r$ and teeth $T_1, \ldots, T_s$. For every tooth T_j, we denote by t_j the number of handles which intersect T_j. Choose from every tooth T_j, $1 \leq j \leq t$, a node $r(j)$ not belonging to any handle, referred to as the *root* of T_j. Choose from every nonempty intersection $H_i \cap T_j$, $1 \leq i \leq r$, $1 \leq j \leq s$, of a tooth and a handle a node $u(i, j)$, which we call the *link* of H_i and T_j. Define R to be the set of all roots $r(j)$, $1 \leq j \leq s$, and U to be the set of all links $u(i, j)$, where $1 \leq i \leq r$, $1 \leq j \leq s$, and $H_i \cap T_j \neq \emptyset$. Using the result of Bauer [5] and performing the substitution $x(\delta(v)) = 2y_v \; \forall v \in V$, the following can be shown to be true.

Theorem 6. *Let a clique tree of K_n, $n \geq 6$, be given by a set of handles $H_1, \ldots, H_r$, $r \geq 1$, and a set of teeth $T_1, \ldots, T_s$. Let R be a set of roots and U a set of links. Then the* **forest inequality**

$$\sum_{i=1}^{r} x(E(H_i)) + \sum_{j=1}^{s} x(E(T_j)) \leq \sum_{i=1}^{r} y(H_i) + \sum_{j=1}^{s} (y(T_i) - t_i) - y(U) - y(R) + \frac{3}{2}(s-1)$$

is facet defining for P_C^n.

Using Lemma 1, we can find a sufficient condition for a forest inequality to be facet defining for $P_C^{n,k}$.

Theorem 7. *Let $n \geq 9$, $4 \leq k < n$, and let $ax + fy \leq a_0$ be a forest inequality with set of roots R and set of links U. Then $ax + fy \leq a_0$ is facet defining for $P_C^{n,k}$ whenever $|R| + |U| + 2 \leq k$.*

Proof. We let $K = U \cup R$ and apply Lemma 1. Let $e = (u, v)$ be any edge not contained in $E(K)$. The inequality $\tilde{a}x + \tilde{f}y \leq a_0$, where $(\tilde{a}, \tilde{f})$ is the restriction of (a, f) to $G_K^{uv} = (K \cup \{u, v\}, E(K \cup \{u, v\}))$ is also a forest inequality and hence there is a circuit C of length at most $|K \cup \{u, v\}| \leq k$ containing e and satisfying $\tilde{a}\chi^C + \tilde{f}\chi^{V(C)} = a\chi^C + f\chi^{V(C)} = a_0$.

□

If $r = 1$ and $|T_j \cap H_1| = 1 \; \forall j = 1, \ldots, s$, we call the inequality a *simple forest inequality*. Simple forest inequalities correspond to the *simple comb inequalities* of the TSP polytope. If, in addition, $|T_j| = 2 \; \forall j = 1, \ldots, s$, we call the resulting inequality a *2-forest inequality*. The 2-forest inequalities correspond to *2-matching inequalities* of the TSP polytope.

4. Inequalities specific to the cardinality constrained circuit polytope

In this section, we derive valid inequalities for $P_C^{n,k}$ that are not valid for P_C^n. The first two, the *cardinality-path inequalities* and the *k-partition inequalities* are derived from first principles. The second two, the *cardinality-tree inequalities* and the *maximal set inequalities* are obtained by strengthening inequalities known to be valid and facet inducing for P_C^n.

All classes of inequalities presented are facet inducing under certain conditions. However, the proofs are standard, tedious, and long. Therefore, in the text, we discuss only the validity of the classes of inequalities and refer the interested reader to the technical report for details on the facet proofs [7].

4.1. The cardinality-path inequalities

If P is a path with k edges and C is a circuit of cardinality at most k, then the cardinality-path inequality corresponding to P says that C never uses more edges of P than inner nodes of P.

Theorem 8. *Let $4 \leq k < n$, P be a path in K_n consisting of k edges, and $\dot{P}$ denote the set of inner nodes of P. Then the* **cardinality-path** *inequality*

$$x(P) \leq y(\dot{P})$$

defines a facet of $P_C^{n,k}$.

Proof of validity. To see that the cardinality-path inequalities are valid for $P_C^{n,k}$, suppose for a contradiction that C is a feasible circuit satisfying $x(P) \geq y(\dot{P}) + 1$ for some path P consisting of k edges. By definition $y(\dot{P}) = k - 1$, so $x(P) \geq k$. Since C is a circuit, $x(C \setminus P) \geq 1$, but then $x(C) = x(C \setminus P) + x(P) > k$, so C was not feasible.

□

4.2. The k-partition inequalities

The k-partition inequalities ensure that we use enough edges across a partition of V into sets of size $k - 1$.

Theorem 9. *Let $4 \leq k < n$, $s = \lceil \frac{n}{k-1} \rceil$ and $V = \bigcup_{i=1}^{s} V_i$ be a partition of V with $|V_i| = k - 1$ for $1 \leq i \leq s - 1$ and $|V_s| \leq k - 1$. Then*

$$2 \sum_{i=1}^{s} x(E(V_i)) + \sum_{i=1}^{s-1} \sum_{j=i+1}^{s} x((V_i : V_j)) \leq 2(k-1)$$

is facet defining for $P_C^{n,k}$.

Proof of validity. To see that the k-partition inequalities are valid, note that all circuits of length at most $k - 1$ satisfy the inequality. Further, a circuit of length k must use at least two edges in $\cup_{i=1}^{s-1} \cup_{j=i+1}^{s} (V_i : V_j)$, so the inequality is satisfied in this case as well.

□

4.3. Maximal set inequalities

Wang [39] introduces a class of inequalities he calls the *multipartition inequalities* and shows them to be facet defining for P_C^n. His theorem is stated below.

Theorem 10 (Wang). *Let $K_n = (V, E)$, $4 \leq k < n$ and a partition of V be given by $V = \bigcup_{i=1}^{s} V_i$ where $|V_i| \geq 2$ for all $1 \leq i \leq s$. Moreover, let $T_i \subseteq E(V_i)$, $1 \leq i \leq s$, be a spanning tree of the complete graph induced by V_i and $\overline{T}_i$ be its complement with respect to $E(V_i)$. Then a facet of P_C^n is induced by the inequality*

$$2 \sum_{i=1}^{s} x(\overline{T}_i) + \sum_{i=1}^{s-1} \sum_{j=i+1}^{s} x((V_i : V_j)) \geq 2.$$

These inequalities do not in general induce facets of $P_C^{n,k}$ but can be strengthened by replacing the sets $\overline{T}_i$, which are the complements of maximal sets in $E(V_i)$ not containing any circuit, by complements of maximal sets in $E(V_i)$ not containing any circuit of cardinality less than or equal to k.

Theorem 11. *Let $K_n = (V, E)$, $4 \leq k < n$, and let a partition of V be given by $V = \bigcup_{i=1}^{s} V_i$ where $|V_i| \geq 2$ for all $1 \leq i \leq s$. Moreover, let $M_i \subseteq E(V_i)$, $1 \leq i \leq s$, be a maximal edge set with respect to $E(V_i)$ not containing any circuit of cardinality less than or equal to k. Let $\overline{M}_i = E(V_i) \setminus M_i$ be its complement with respect to $E(V_i)$. Then a facet of $P_C^{n,k}$ is induced by the* **maximal set inequality**

$$2\sum_{i=1}^{s} x(\overline{M}_i) + \sum_{i=1}^{s-1}\sum_{j=i+1}^{s} x((V_i : V_j)) \geq 2.$$

Proof of validity. The inequality is valid, since every feasible circuit either uses at least one edge in $\cup_{i=1}^{s} \overline{M}_i$ or two edges in $\cup_{i=1}^{s-1} \cup_{j=i+1}^{s} (V_i : V_j)$.

□

4.4. The cardinality-tree inequalities

Wang [39] introduces the following generalization of the degree inequalities.

Theorem 12 (Wang). *Let T be a spanning tree of K_n. For each $e = (u, v) \in E \setminus T$, define l_e^T as the length of the unique (u, v) path in T. The* **tree inequality**

$$\sum_{e \in T} x_e + \sum_{e \notin T} (2 - l_e^T) x_e \leq 2$$

is a valid inequality for P_C^n. If T is such that every edge $e \in T$ is in a star $K_{1,3} \subseteq T$, then the inequality is also facet defining for P_C^n.

Using the fact that all circuits must be of length at most k, this inequality can be strengthened.

Theorem 13. *Let T be a spanning tree of K_n. For each $e = (u, v) \in E \setminus T$, define l_e^T as the length of the unique (u, v) path in T. Define*

$$w_{l_e}^T = \begin{cases} 2 - l_e^T & \text{if } 2 \leq l_e^T \leq k - 1, \\ 4 - 2k + l_e^T & \text{if } k \leq l_e^T \leq \lceil 3k/2 \rceil - 2, \\ 2 - k/2 & \text{if } k \text{ is even and } l_e^T \geq 3k/2 - 1, \\ (3 - k)/2 & \text{if } k \text{ is odd and } l_e^T = \lceil 3k/2 \rceil - 1 + 2i \text{ for some } i \in \mathbb{Z}_+, \\ (5 - k)/2 & \text{if } k \text{ is odd and } l_e^T = \lceil 3k/2 \rceil + 2i \text{ for some } i \in \mathbb{Z}_+. \end{cases}$$

The **cardinality-tree inequality**

$$\sum_{e \in T} x_e + \sum_{e \in E \setminus T} w_{l_e}^T x_e \leq 2$$

is a valid inequality for $P_C^{n,k}$.

If $n \geq 5$, $k \geq 4$, *and* T *is a spanning tree of* K_n *with the following properties:*

- *If* v *is a leaf node of* T *with adjacent node* u, *then* $|\delta(u)| \geq 3$,
- *If* $e \in E \setminus T$ *is such that* $l_e^T \geq \lceil 3k/2 \rceil - 1$, *then there exists and edge* $f \in E \setminus T$ *with* $l_f^T = l_e^T - 1$, *and a circuit* C, $|C| = k$, *consisting of* e, f *and edges of* T,

then the inequality

$$\sum_{e \in T} x_e + \sum_{e \in E \setminus T} w_{l_e}^T x_e \leq 2$$

is facet inducing for $P_C^{n,k}$.

Proof of validity. The proof of validity of the cardinality-tree inequalities relies on the concept of coefficient improvement and is fairly long. We sketch the proof here and refer the interested reader to the appendix for a complete proof.

We will improve the coefficients of the variables corresponding to edges $e \in E \setminus T$. These coefficients w_e start out at their initial values $2 - l_e^T$, and it is our goal to show that they can be increased to at least $w_{l_e}^T$, thereby strengthening the inequality. Note first that the coefficients of edges $f \in E \setminus T$ with $l_f^T \leq k - 1$ are not changed, so the "new" coefficients $w_{l_f}^T$ are certainly valid in this case. For an edge $f \in E \setminus T$, define the coefficient improvement problem (CIP) for f as

$$\begin{aligned} \max_x \quad & z_f \equiv \sum_{e \in T} x_e + \sum_{e \in E \setminus T} w_e x_e \\ \text{subject to:} \quad & x \text{ is the incidence vector of a circuit } C, \\ & |C| \leq k, \\ & x_f = 1. \end{aligned}$$

With the above definitions, the coefficient $w_f^T = 2 - l_f^T$ can be increased by $2 - z_f$. Now, suppose for a contradiction that the coefficient for edge $f \in E \setminus T$ cannot be increased to $w_{l_f}^T$. This implies that

$$2 - l_f^T + 2 - z_f \leq w_{l_f}^T - 1. \tag{4.15}$$

We now proceed to prove the validity of the coefficient improvements on a case by case basis by considering an optimal circuit C defined by an incidence vector x and showing that inequality (4.15) does not hold.

□

Using techniques found in the proof of Theorem 13, we can show that the condition that if v is a leaf node of T with adjacent node u, then $|\delta(u)| \geq 3$ is both necessary

and sufficient for the tree inequalities to be facet defining for P_C^n, thus strengthening Theorem 12 of Wang [39].

5. Separation results

In this section, we investigate the complexity of the separation problems associated with the classes of facet inducing inequalities discussed above. Given an arbitrary point, the separation problem associated with a class of inequalities is to determine an inequality in the class that is violated by this point, or to show that no violated inequality in the class exists.

5.1. Parity constraints

Given a point $(x^*, y^*)^T \in \mathbb{R}^{|E|+|V|}$, we can determine if it violates any of the parity constraints

$$x^*(\delta(v) \setminus x_e) - x_e^* \geq 0, \ v \in V, \ e \in \delta(v)$$

by simple substitution. This requires computation time $O(n^2)$.

5.2. Disjoint circuit elimination constraints

For $e = (u, v) \in E$ and a partition S, T of $V \setminus \{u, v\}$, we have a disjoint circuit elimination constraint

$$x_e + x((u : T)) + x((v : S)) - x((S : T)) \leq 2.$$

This constraint is equivalent to

$$x((S \cup v) : (T \cup u)) \geq x(\delta(u)) + x(\delta(v)) - 2.$$

We can therefore solve the separation problem for the disjoint circuit elimination constraints by finding a minimum (u, v)-cut for all pairs of nodes $u, v \in V$.

This can be done efficiently by creating a Gomory-Hu cut tree T for G. A Gomory-Hu cut tree T of G is a weighted tree T on the same node set as G with the property that for any two nodes u and v of G, the value of the minimum (u, v)-cut is equal to the smallest weight of an edge e on the path from u to v in T. The cut with this minimum value is given by the components of $T \setminus e$. Gomory-Hu cut trees can be built by performing $n - 1$ maximum flow computations. If the shortest augmenting path algorithm of Edmonds and Karp [13] is used to compute the maximum flows, the cut tree can be built and the check for violated inequalities can be performed in time $O(|V|^3|E|)$. For a description of an algorithm to build a Gomory-Hu cut tree, the reader is refered to Gusfield [20].

5.3. Cut inequalities

A point $(x^*, y^*)^T \in \mathbb{R}^{|E|+|V|}$ satisfies all the cut inequalities of Theorem 5 if and only if for all $\emptyset \neq S \subseteq V,\ e \in \delta(S)$, we have

$$x^*(\delta(S)) \geq 2x_e^*.$$

We can therefore solve the separation problem for the cut inequalities by calculating a minimum (s, t)-cut for each edge $e = (s, t) \in E$ and checking whether the corresponding cut inequality is satisfied. This can be done efficiently using a Gomory-Hu cut tree.

5.4. Forest inequalities

Bauer [5] shows that the forest inequalities are valid for P_C^n by subtracting appropriate positive multiples of the degree constraints from the clique tree inequalities. Therefore, if we solve the separation problem for the class of clique tree inequalities and find a violated inequality, it must be the case that the corresponding forest inequality is also violated. This gives us a heuristic for separating for forest inequalities.

Little is known about the separation problem for clique tree inequalities. Carr [10] has shown that it is possible to separate clique tree inequalities having a fixed number of handles and teeth in polynomial time, and Fleischer and Tardos [15] have shown that separating comb inequalities in planar graphs can also be done in polynomial time. We separate only for the 2-matching and simple comb subclasses of the clique tree inequalities. The corresponding forest inequalities are the *2-forest* and *simple forest* inequalities, respectively.

For the 2-matching inequalities, an exact separation procedure is known [30], but it requires solving an odd minimum cut problem, which is very time consuming. Instead, we use a heuristic presented by Padberg and Rinaldi [31] to find violated 2-matching (2-forest) inequalities. The procedure is given as *Procedure 4.10* in [31] and has worst case complexity of $O(n^4)$. In general, the running time is much faster.

We also heuristically identify violated simple forest inequalities using the *Procedure 5.3* of Padberg and Rinaldi [31] for finding violated simple comb inequalities.

5.5. Cardinality-path inequalities

Consider the separation problem for the cardinality-path inequalities.

PROBLEM:	Cardinality-path separation problem (CPSEP)
INSTANCE:	Complete graph $G = (V, E)$, weights $x^* \in Q_+^{|E|},\ y^* \in Q_+^{|V|},\ k \in \mathbb{Z}_+$.
QUESTION:	Does there exist a path P of cardinality k in G such that $x^*(P) - y^*(\dot{P}) > 0$?

Theorem 14. *CPSEP is NP-Complete.*

Proof. CPSEP is obviously in NP. We show it is NP-Complete by a reduction from the longest path problem (LPP) [16]:

PROBLEM: Longest Path Problem (LPP)
INSTANCE: Complete graph $G = (V, E)$, weights $w_e \in Q_+^{|E|}$, $B \in Q$.
QUESTION: Does there exist a path P with $w(P) > B$ in G?

Since $w_e \geq 0$, it suffices to consider paths P of cardinality $|V| - 1$ with $w(P) > B$ in LPP. Given an instance of LPP, we construct an instance of CPSEP by letting the two graphs be the same, $x^* = w$, $k = |V| - 1$, and $y_v^* = B/(|V| - 2)\ \forall v \in V$. This is a polynomial transformation.

By construction there exists a path P with $w(P) > B$ for LPP if and only if there exists a path P of cardinality $|V| - 1$ such that $x^*(P) - y^*(\dot{P}) > 0$ for CPSEP.

□

Since the separation problem is NP-Complete, we look for violated cardinality-path inequalities by enumerating the paths of length k and checking if $x(P) - y(\dot{P}) > 0$ for each path. The effectiveness of this approach can be greatly enhanced by a simple observation. During the enumeration, we will be considering (partial) paths P' (of length $< k$). Suppose we orient the path so that vertex b is the first in the path. Then, whenever the partial path P' is such that $x(P') - (y(V(P')) - y_b) \leq -1$, the path needs no longer to be considered, since it will never lead to a violated cardinality-path constraint.

With this simple device, we have greatly reduced the amount of work from simple total enumeration. We may even ask whether the resulting algorithm has polynomial running time. The following theorem shows that the number of paths considered may still be exponential.

Theorem 15. *The enumeration algorithm with the pruning condition described for solving CPSEP has a nonpolynomial running time.*

Proof. Let $n = |V|$ and consider an instance of CPSEP with $y_v = 3/n\ \forall v \in V$, $x_e = 6n/(n-1)\ \forall e \in E$, and $k = n$. The weights satisfy all the necessary conditions in an LP-solution to the CCCP.

Let $b \in V$ be the vertex from which we begin the path enumeration procedure. For a (partial) path P^l of length l, $x(P^l) = 6l/n(n-1)$ and $y(V(P^l)) - y_b = 3l/n$. In this instance, the value of $x(P^l) - (y(V(P^l)) - y_b)$ depends only on l, and $x(P^l) - (y(P^l) - y_b) \leq -1$ only when

$$\frac{6l}{n(n-1)} - \frac{3l}{n} \leq -1 \quad \Leftrightarrow \quad l \geq \frac{n(n-1)}{3(n-3)}.$$

Thus, our enumeration algorithm will consider all paths until

$$l = \left\lceil \frac{n(n-1)}{6(n-2)} \right\rceil.$$

The number of such paths is

$$N = \frac{(n-1)!}{(n-1-\left\lceil \frac{n(n-1)}{3(n-3)} \right\rceil)!}.$$

As n grows large this value approaches

$$N = \frac{(n-1)!}{(\frac{2}{3}(n-1))!},$$

which is not polynomially bounded.

□

5.6. *k-partition inequalities*

Given a point $x^* \in \mathbb{R}_+^{|E|}$, the separation problem for the k-partition inequalities is to find a partition of the vertices $V = \bigcup_{i=1}^{s} V_i$, with $|V_i| = k-1$ for $i = 1, \ldots, s$ and $|V_s| \leq k-1$ such that

$$2\sum_{i=1}^{s} x^*(E(V_i)) + \sum_{i=1}^{s-1}\sum_{j=i+1}^{s} x^*((V_i : V_j)) > 2(k-1).$$

Through some simple algebraic manipulation, this condition can be shown to be equivalent to finding a partition of the prescribed form with

$$\sum_{i=1}^{s}\sum_{j=i+1}^{s} x^*((V_i : V_j)) < 2(x^*(E) - (k-1)).$$

Theorem 16. *The separation problem for the k-partition inequalities is NP-Complete.*

Proof. The separation problem is clearly in NP. We will show that the separation problem for k-partition inequalities is at least as hard as the following NP-Complete graph partitioning problem (GPP) [23]:

PROBLEM: Graph Partitioning Problem (GPP)
INSTANCE: Graph $G = (V, E)$, weights $l(e) \in Q^+$ $\forall e \in E$, and positive integers K, J.
QUESTION: Is there a partition of $V = \bigcup_{i=1}^{s} V_i$ such that $|V_i| \leq K$ $\forall i = 1, \ldots, s$ and such that $\sum_{i=1}^{s}\sum_{j=i+1}^{s} l((V_i : V_j)) \leq J$?

Given an instance of GPP, we construct an instance of the separation problem for k-partition inequalities by letting the graphs be the same, $k = K + 1$, and

$$x_e = \frac{2(k-1)}{2l(E) - J} l(e).$$

We may, without loss of generality, assume that $2l(E) - J \geq 0$ and hence $x_e \geq 0$ $\forall e \in E$. With these definitions, there exists a partition $V = \bigcup_{i=1}^{s} V_i$ with $|V_i| \leq K$ and $\sum_{i=1}^{s}\sum_{j=i+1}^{s} l((V_i : V_j)) \leq J$ in GPP if and only if there exists a partition

$V = \bigcup_{i=1}^{s} V_i$ with $|V_i| = k-1$ and $\sum_{i=1}^{s}\sum_{j=i+1}^{s} x((V_i : V_j)) \leq 2(x(E) - (k-1))$ in the corresponding k-partition inequalities separation problem.

$$\sum_{i=1}^{s}\sum_{j=i+1}^{s} x((V_i : V_j)) \leq 2(x(E) - (k-1)) \Leftrightarrow$$

$$\frac{2(k-1)}{2l(E) - J}\sum_{i=1}^{s}\sum_{j=i+1}^{s} l((V_i : V_j)) \leq 2\left(\frac{2(k-1)}{2l(E) - J}l(E) - (k-1)\right) \Leftrightarrow$$

$$\sum_{i=1}^{s}\sum_{j=i+1}^{s} l((V_i : V_j)) \leq 2l(E) - (2l(E) - J) \Leftrightarrow$$

$$\sum_{i=1}^{s}\sum_{j=i+1}^{s} l((V_i : V_j)) \leq J.$$

□

Since the separation problem is NP-Complete, in order to search for violated k-partition inequalitites, we use a heuristic. The heuristic begins by greedily partitioning the vertices, followed by two-exchanges.

5.7. Maximal set inequalities

Given $x^* \in \mathbb{R}_+^{|E|}$, and an integer s, the separation problem for the maximal set inequalities is to find a partition of the vertices $V = \bigcup_{i=1}^{s} V_i$ and in each V_i a maximal edge set M_i not containing any circuit of cardinality less than or equal to k such that

$$2\sum_{i=1}^{s} x^*(\overline{M}_i) + \sum_{i=1}^{s-1}\sum_{j=i+1}^{s} x^*((V_i : V_j)) < 2.$$

In V_i, there may be many types of maximal sets not containing a circuit of cardinality less than or equal to k. One example is a spanning tree, where all fundamental circuits in the tree have cardinality at most k. If we choose such a tree for all $1 \leq i \leq s$, we get an inequality of the type given in Theorem 10.

For the special case $s = 1$, the separation problem for this particular maximal set is the following.

PROBLEM: Maximal set separation problem (MAXSEP1):
INSTANCE: Graph $G = (V, E)$, weights $x_e \in Q^+$ $\forall e \in E$, positive integer k.
QUESTION: Does there exist a spanning tree T of G such that $x(T) > x(E) - 1$ and the longest path in T has at most $k-1$ edges?

Theorem 17. *MAXSEP1 is NP-Complete.*

Proof. MAXSEP1 is obviously in NP. We will prove that MAXSEP1 is at least as hard as the NP-Complete bounded diameter spanning tree problem [16]:

PROBLEM: Bounded diameter spanning tree (BDST):
INSTANCE: Graph $G = (V, E)$, weight $w_e \in Q^+ \; \forall e \in E$, positive integer $D \leq |V|$, positive integer B.
QUESTION: Does there exist a spanning tree T of G such that $w(T) < B$ and the longest path in T has at most D edges?

Let $w^* \equiv \max_{e \in E} w_e$. Given an instance of BDST, we construct an instance of MAXSEP1 by letting the graphs be the same, $k - 1 = D$, and

$$x_e = \frac{w^* - w_e}{(|E| - |V| + 1)w^* - w(E) + B} \quad \forall e \in E.$$

We may assume that our constructed instance has $(|E| - |V| + 1)w^* - w(E) + B > 0$ (and hence $x_e \geq 0 \; \forall e \in E$), for if

$$(|E| - |V| + 1)w^* - w(E) + B \leq 0,$$

we have

$$(|E| - |V| + 1)w^* - w(E \setminus T) - w(T) + B \leq 0,$$

which implies that $w(T) \geq B$.

With these definitions, a tree T is such that $x(T) > x(E) - 1$ if and only if $w(T) < B$:

$$\begin{aligned} x(T) &> x(E) - 1 \Leftrightarrow \\ \frac{(|V| - 1)w^* - w(T)}{(|E| - |V| + 1)w^* - w(E) + B} &> \frac{|E|w^* - w(E)}{(|E| - |V| + 1)w^* - w(E) + B} - 1 \Leftrightarrow \\ (|V| - 1)w^* - w(T) &> |E|w^* - w(E) - (|E| - |V| + 1)w^* \\ &\quad + w(E) - B \Leftrightarrow \\ w(T) &< B, \end{aligned}$$

which completes the proof.

□

Since the separation problem is NP-Complete, we consider a heuristic to find violated maximal set inequalities of this form. The heuristic has the following steps:

0. $s = 1$.
1. Find a partition $\cup_{i=1}^{s} V_i$ of the vertices such that $\sum_{i=1}^{s-1} \sum_{j=i+1}^{s} x^*((V_i : V_j))$ is small.
2. For each V_i, $i = 1, \ldots, s$, construct a spanning tree T_i of large weight such that all paths in T have length at most $k - 1$.
3. If $s = |V|/2$, then stop. Else $s = s + 1$. Go to 1.

Note that in our heuristic, we do not require that $|V_i| \geq 2$, $i = 1, \ldots, s$. The inequality is still valid in this case, but may not be facet defining. Step 1 of the heuristic is equivalent to finding a minimum s-cut. This problem is NP-Complete for arbitrary s [18]. Saran and Vazirani [36] show how to compute s-cuts within a factor of $2 - 2/s$ of optimal using a Gomory-Hu cut tree. Our separation heuristic uses this method to find a light weight partition for each $s \in \{1, 2, \ldots |V|/2\}$. Step 2 of the heuristic is equivalent to

solving the NP-Complete problem **BDST**. To construct trees of large weight which have no paths of length more than k, we use a heuristic modification of Prim's greedy algorithm to find a maximum weight spanning tree [33]. At each step of the algorithm, we add the heaviest edge to the tree that will not create a circuit and will not induce a path of length k in the tree.

5.8. Cardinality-tree inequalities

Given some fractional LP solution x^*, the separation problem seeks a violated cardinality-tree inequality, that is a tree T such that

$$\sum_{e\in T} x_e^* + \sum_{e\in E\setminus T} w_{l_e}^T x_e^* > 2. \tag{5.16}$$

Since the cardinality-tree inequalities are a strengthening of the tree inequalities, if we are able to find a violated tree inequality, then the corresponding cardinality-tree inequality must also be violated. Now we show how to solve the separation problem for tree inequalities.

In the Optimum Requirement Spanning Tree problem (ORST), we are given a complete graph $K_n = (V, E)$ and "requirements" $r_e \in Q_+ \ \forall e \in E$, and we ask for a spanning tree T that minimizes the quantity $\sum_{e\in E} l_e^T r_e$. (Recall that for $e = (u, v)$, l_e^T is the length of the unique (u, v) path in T). The ORST problem was introduced by Hu [22]. He also shows that the optimal solution to ORST is a Gomory-Hu cut tree using the "requirements" r_e as weights.

Theorem 18. *Tree inequalities can be separated in polynomial time.*

Proof. Let x^* be the solution to the linear programming relaxation. We first show that

$$\sum_{e\in T} x_e^* + \sum_{e\in E\setminus T} (2 - l_e^T) x_e^* > 2 \Leftrightarrow \sum_{e\in E} l_e^T x_e^* < 2\sum_{e\in E} x_e^* - 2.$$

$$\sum_{e\in T} x_e^* + 2\sum_{e\in E\setminus T} x_e^* - \sum_{e\in E\setminus T} l_e^T x_e^* > 2 \Leftrightarrow$$

$$\sum_{e\in E} x_e^* + \sum_{e\in E\setminus T} x_e^* - \sum_{e\in E\setminus T} l_e^T x_e^* > 2 \Leftrightarrow$$

$$\sum_{e\in E\setminus T} l_e^T x_e^* - \sum_{e\in E\setminus T} x_e^* < \sum_{e\in E} x_e^* - 2 \Leftrightarrow$$

$$\sum_{e\in E\setminus T} l_e^T x_e^* - \sum_{e\in E\setminus T} x_e^* + \sum_{e\in T} x_e^* + \sum_{e\in E\setminus T} x_e^* < 2\sum_{e\in E} x_e^* - 2 \Leftrightarrow$$

$$\sum_{e\in E} l_e^T x_e^* < 2\sum_{e\in E} x_e^* - 2.$$

Thus, using the result of Hu for the ORST problem, we solve the separation problem for tree inequalities by computing a Gomory-Hu tree using the solution x^* as requirements and comparing the optimal objective value to the constant $2\sum_{e\in E} x_e^* - 2$.

□

Wang does not discuss the separation problem for the tree inequalities, so Theorem 18 is a new result. The complexity of separating the cardinality-tree inequalities is unknown. If the "weights" w_{le}^T in (5.16) are arbitrary, then the problem is equivalent to the NP-Complete Optimum Communication Spanning Tree problem [24].

5.9. Summary

In the preceeding sections, six classes of valid inequalities for $P_C^{n,k}$ have been presented. Table 5.1 shows a summary of relevant information for each of these classes.

Table 5.1. Characteristics of the classes of inequalities for $P_C^{n,k}$

Inequality Class	Facet Defining?	Separation Complexity	Separation Routine
cut	Yes	Polynomial	min (s, t) cut $\forall e = (s, t) \in E$.
forest	If $\lvert R\rvert + \lvert U\rvert + 2 \leq k$	NP-Complete (polynomial for certain clique trees)	Heuristic procedures of Padberg and Rinaldi [31]
cardinality-path	Yes	NP-Complete	Limited enumeration heuristic
k-partition	Yes	NP-Complete	Greedy and 2-opt heuristic
maximal set	If $\lvert V_i\rvert \geq 2\ \forall i$	NP-Complete	Heuristic combining min s-cut and max bounded diameter spanning tree
cardinality-tree	For certain trees	Unknown (polynomial for tree inequalities)	Exact separation for tree inequalities. Uses Gomory-Hu tree

6. Computations

In this section, we describe the design and implementation of a branch-and-cut algorithm for the CCCP based on the classes of valid inequalities introduced in the previous sections. The initial formulation consists of the degree constraints, the constraint $x(E) \geq 3$, and the cardinality constraint $x(E) \leq k$. The other classes of valid inequalities are handled implicitly, i.e., constraints are added on-the-fly only when they are violated.

Our current implementation of the branch-and-cut algorithm does not use any sparse graph representation or column generation techniques. Therefore, the size of the instances we are able to solve is limited by the amount of memory on our computer. We had to restrict ourselves to instances with graphs of up to 100 vertices. Fortunately, these instances are large and difficult enough to thoroughly investigate the performance of the algorithm. Furthermore, as mentioned in the introduction, the main motivation for studying branch-and-cut algorithms for the CCCP is to see if they might provide a viable alternative to dynamic programming to solve the pricing problems arising in branch-and-price algorithms for the vehicle routing problem. In that context, being able to solve pricing problems with up to a 100 customers is more than enough.

We have used the integer programming software MINTO v2.0 [27] to manage the branch-and-bound tree and to interface with the linear programming solver. The linear progams were solved with CPLEX v4.0 [11].

6.1. A primal heuristic for the CCCP

In order for a branch-and-cut algorithm to be effective, good upper as well as lower bounds on the value of the optimal solution must be obtained. Upper bounds are obtained by finding feasible solutions to the problem. LP-based heuristics have been shown to be quite effective [25,37]. Such heuristics are based on the intuition that the solution to the LP relaxation of an integer program is suggestive of what good feasible solutions look like. For example, if the value of an edge variable in the LP solution is large, then it is likely that feasible solutions containing this edge are of good quality. An edge-oriented LP-based heuristic for the CCCP is given in Algorithm 1.

Algorithm 1 LP Edge-value based heuristic for the CCCP

1. Using the LP values x^* as weights, greedily create a set of paths $P = \{P_1, P_2, \ldots, P_{|\mathcal{P}|}\}$ (with index set $\mathcal{P}$) such that $|\cup_{j\in\mathcal{P}} V(P_j)| \leq k$.
2. For each of $P_1, P_2, \ldots, P_{|\mathcal{P}|}$, find the edge that, when added, creates the cheapest circuit. The edge need not connect the endpoints of the path; if it does not, the superflous edges are removed. Let the cheapest circuit created this way be the current incumbent C^*.
3. Let $t = \arg\min_{j\in\mathcal{P}}\{c(P_j)\}$. Find the cheapest way to connect a path in P to P_t and call the resulting path P'_t.
4. If P'_t can be closed as in Step 2 creating a circuit C' such that $c(C') < c(C^*)$, let $C^* = C'$, $P = P \cup P'_t \setminus P_t$, and goto 3. Else output C^* and end.

6.2. A branching scheme

If we are unable to generate violated inequalities, then we branch on a fractional integer variable. Because fixing a node variable has a much stronger impact than fixing an edge variable, we choose to branch on the most fractional node variable. If no fractional node variable can be found, we use the default branching scheme in MINTO v2.0. We note that this may not be the most efficient branching scheme for this problem class. For example, the *strong branching* method introduced by Applegate et al. [1] may very well be effective for the CCCP. We leave such investigations as a line of future computational research.

6.3. A cut management scheme

When designing a branch-and-cut algorithm, a number of choices have to be made regarding when to generate cuts and how to use generated cuts. Recent studies have shown that these choices can significantly impact the overall effectiveness of the branch-and-cut algorithm [4].

Some of the choices that need to be considered are the following:

- Should we generate cuts at every node of the search tree?
- When we decide to generate cuts at a node, should we keep generating cuts as long as we can find them?

- Should we add all the generated violated cuts for a given fractional LP point?
- Should we try to generate violated cuts for all classes of cuts?
- Should we delete inactive cuts from the active linear program?

A cut management scheme tries to find the right balance between the various components of a branch-and-cut algorithm, e.g., LP solving, cut generation, and branching. In this section, we discuss a number of experiments that we have conducted to develop an effective cut management scheme for our branch-and-cut algorithm.

We have taken 4 instances from the TSPLIB [35] and modified them to create instances of the CCCP in the following manner. First, a positive constant was subtracted from the weight of each edge. This results in optimal solutions having more than a small number of edges in the optimal solution. This strategy was also used by Bauer to construct instances of WGP [5]. For each of these modified TSPLIB instances, we created three instances of the CCCP by varying the value of k. Let q be the cardinality of the optimal circuit for the modified TSPLIB instance without a cardinality constraint. The values of k were chosen as follows:

- $k = q$,
- $k = \lfloor 2q/3 \rfloor$,
- $k = \lfloor q/3 \rfloor$.

This gives us 12 instances. This is admittedly a small set of instances, but at the moment we are only looking for trends to help us decide on a cut management scheme. When we have decided on a cut management scheme and are ready to test the complete branch-and-cut algorithm, we will use a much larger set of instances.

Our code was compiled with the IBM xlc compiler with code optimization -O2 and run on an RS6000 Model 390 with 512MB of memory.

6.3.1. Cut deletion. Preliminary computational experiments showed that because many violated cuts are generated, the size of the active linear program increases rapidly. This leads to linear programs that require a prohibitive amount of memory. Therefore, we chose to delete all cuts from the active linear program that have not been binding in the last two LP solutions. This reduces the memory requirements and speeds up the LP solution times. It may happen that we generate the same cut several times.

6.3.2. Cuts per round. Given a fractional LP solution x^*, we solve the separation problem for each class of valid inequalities introduced in the previous sections. We call this a *round* of cut generation. In each round, we typically find many violated inequalities. To conserve memory and to speed up the solution of the active linear program, it may be more efficient to add only a subset of the generated violated inequalities, for example the η "best" cuts. We have used the distance d from the LP solution x^* to the hyperplane $a^T x = b$ defining a cut as the measure of quality of a cut. The distance d is given by

$$d = \frac{b - a^T x^*}{a^T a}.$$

We have conducted a small experiment to determine a good value for η. In the experiment, we generated cuts at every node of the search tree for each class of inequalities,

and we used a "tailing-off detection percentage" (see Sect. 6.3.3) of 0.1%. Table 6.2 shows the performance of the branch-and-cut algorithm for various values of η on our twelve test instances.

Table 6.2. The effect of the maximum number of cuts per round

η	Total Nodes	Total Time (sec.)
1	1680	(N/A)
10	359	4898
25	261	5968
50	288	5093
all	274	4966

If $\eta = 1$, i.e, when we add only the deepest cut in a single round of cut generation, then the branch-and-cut code fails to solve three instances. In each case, it exceeds the memory limit. If $\eta > 1$, it is hard to draw definite conclusions. We have chosen to work with $\eta = 50$ in all remaining experiments.

6.3.3. Tailing-off. The change in the value of the objective function of the LP relaxation has a tendency to decrease as the number of rounds of cut generation increases at a node in the search tree. This is known as the *tailing-off* effect. If the change in the value of the objective function from one round of cut generation to the next becomes "too small", we may decide to branch rather than to generate additional cuts, since it appears that spending more time on cut generation is not worthwhile at the moment. We have implemented tailing-off detection as follows. If the change in the value of the objective function is less than α% over the last two rounds of cut generation, we say tailing-off has occurred.

We have conducted a small experiment to determine a good value for α. In the experiment, we generated cuts at every node of the search tree for each class of inequalities. Table 6.3 shows the performance of the branch-and-cut code for various values of α.

Table 6.3. The effect of the tailing off percentage

α	Total Nodes	Total Time (sec.)
0.1	274	4966
1.0	403	3432
2.5	426	3356
5.0	486	3185
10.0	480	3196

Based on these results, we decided to use $\alpha = 5.0$ in all remaining experiments.

6.3.4. Cutting or branching. Since generating cuts takes time, and is not guaranteed to be successful, it may not be wise to try to generate cuts at every opportunity. A simple scheme to realize this is to generate cuts at the root node and then only at every κth

node that is evaluated. Balas et al. call κ the *skip factor* [4]. The motivation behind this scheme is that after a number of branchings, we have moved to a different part of the solution space and we are more likely to find deep cuts again. (Note that we have to make one exception. If the LP relaxation at a node is integral but not feasible, i.e., there are violated parity or disjoint circuit elimination constraints, then we have to add cuts.)

We have conducted a small experiment to determine a good value for κ. Table 6.4 shows the performance of the branch-and-cut code for various values of κ.

Table 6.4. The effect of the skip factor

κ	Total Nodes	Total Time (sec.)
1	486	3185
2	1277	4753
5	6607	8779
10	11970	8138
∞	34782	(N/A)

When $\kappa = \infty$, i.e, we generate cuts at the root node only, the branch-and-cut algorithm failed to solve six instances. In each case, it exceeds the memory limit. From Table 6.4, it is clear that (combined with our other cut managent choices) it is best to add cuts at every node of the search tree, which is what we will do in all remaining experiments.

6.3.5. Useful cut classes. We have introduced several classes of valid inequalities for $P_C^{n,k}$ and have derived conditions under which they are facet inducing. However, using a class of facet inducing inequalities is by no means a guarantee for improved performance. For example, if the separation routine for a class of inequalities is computationally intensive but rarely successful, then the overall performance is likely to decrease. Even if the separation algorithm for a class of inequalities is fast and often generates a violated cut, it may still not help to improve the performance. Since we only add the η deepest cuts in any particular round of cut generation, it may be the case that the violated cuts of a particular class are seldomly added to the LP relaxation. Also, the addition of cuts, especially if they are dense, increases memory requirements and may slow down the solution of the LPs.

For each of the classes of inequalities we introduced, Table 6.5 shows the percentage of time that the separation routine was successful in generating a violated inequality, the average amount of time required by the separation routine to do so, and the percentage of time a generated cut was among the 10 deepest cuts found in a single round of cut generation.

Table 6.5 shows clearly that it is not wise to include the maximal set inequalities in our branch-and-cut algorithm. We also felt that it may not be worthwile to include the k-partition inequalities, since they are quite dense and take a relatively long time to generate, or to include the cut inequalities, since they have a tendency to cut off fractional solutions that are also cut off by the disjoint circuit elimination constraints.

Based on the results of Table 6.5 and our intuition, we conducted an experiment to test whether or not the k-partition and cut inequalities should be excluded. Table 6.6 shows

Table 6.5. Cut class statistics

Cut Type	Success %	Avg. Gen. Time (sec.)	% of Time Among 10 Deepest
Forest	22.1%	0.026	45.9%
Path	8.65%	0.308	43.9%
k-Partition	27.8%	0.135	27.2%
Cut	82.8%	0.009	24.2%
Tree	95.0%	0.007	17.3%
Maximal Set	0.434%	0.576	0.0%

the results of this experiment, in which different classes of inequalities are excluded in our branch-and-cut algorithm.

Table 6.6. Cut class statistics

Cut Classes Excluded	Total Nodes	Total Time (sec.)
None	486	3177
Maximal Set	486	2647
Maximal Set, k-Partition	507	2458
Maximal Set, Cut	706	3519
Maximal Set, k-Partition, Cut	793	3379
All	1026	3197

From Table 6.6, we see that excluding the maximal set inequalities clearly improves the performance, and that excluding the cut inequalities is detrimental to the performance. It is difficult to draw any further conclusions. For the remainder of the experiments, we decided to include all classes of cuts except the maximal set inequalities.

6.4. Computational results

With all the components of our branch-and-cut algorithm in place, we are ready to investigate its overall effectiveness. We solved many instances of the CCCP created by modifying instances from TSPLIB in the manner described in Sect. 6.3. The names of the instances are of the form `<name>m<integer>`, where `<name>` is the name of the original TSPLIB instance, and `<integer>` is the positive constant that was subtracted from each edge. The integer in the name of the TSPLIB instances is the number of nodes in the instance.

The full results can be found in Appendix B in Table B.1. For each instance, we report the best objective function value found (z_{best}), the gap between z_{best} and the best lower bound (z_{LB}), computed as $100\,|(z_{best} - z_{LB})/z_{best}|$, the gap between z_{best} and the value of the linear programming relaxation at the root node, the number of evaluated nodes (NN), the number of linear programs solved (NLP), the total computation time (t), the time spent solving linear programs (t_{LP}), and the time spent solving separation problems (t_{cut}). All times are reported in CPU seconds. A time limit of 7200 CPU seconds was imposed in the solution of each instance. Table B.2 in Appendix B shows the results when we solve the same instances using only parity and disjoint circuit elimination cuts,

i.e., cuts necessary for feasibility. If an instance was not solved to proven optimality, then either the time limit or memory limit was reached. The reader can deduce which of the limits was reached by checking the total computation time t.

There are 90 instances in our test suite, and the instances can be broken into three classes based on the computational results:

- Class I. Instances solved by both algorithms.
- Class II. Instances solved by the branch-and-cut algorithm that uses all inequalities, but not by the branch-and-cut algorithm that uses only feasibility cuts.
- Class III. Instances unsolved by both algorithms.

There are 80 instances in Class I, 5 instances in Class II, and 5 instances in class III.

Table 6.7 presents a summary of the results obtained for the "easy" instances of Class I. More specifically, we present the average number of nodes (NN), the average number of LPs solved (NLP), the average time spent on solving LPs (t_{LP}), the average time spent on cut generation (t_{CUT}), and the average solution time (t).

Table 6.7. Summary of computational results for class I instances

	NN	NLP	t_{LP}	t_{CUT}	t
Feasibility Cuts	4588	15822	21415	2072	30024
All Cuts	1995	6486	19143	4070	26768

We see that for the class of easy instances the improvement in performance of the algorithm using all cuts over the algorithm using only feasibility cuts is modest. In fact, of the 32 instances in Class I that required more than one minute to solve by either algorithm, the algorithm using all cuts performed better only in 17 cases.

However, on the more difficult instances of class II and class III the improvements are more significant. Table 6.8 presents the results obtained for the instances of Class II.

Table 6.8. Computational results for class II instances

Name	k	All Cuts?	z_{best}	Final Gap%	Root Gap%	NN	NLP	t_{LP}	t_{cut}	t
cn100am100	100	N	-7418	3.0	3.6	1400	1681	808	1020	4029
cn100am100	100	Y	-7544	0.0	1.8	221	471	352	695	2786
kroA100m200	32	N	-551	88.9	274.1	181	1150	5657	322	7200
kroA100m200	32	Y	-622	0.0	197.4	41	210	2279	411	2983
kroA100m200	48	N	-855	18.2	139.4	273	1301	5649	355	7200
kroA100m200	48	Y	-855	0.0	119.8	17	88	519	114	713
kroC100m200	52	N	-970	25.2	119.0	231	970	5667	419	7200
kroC100m200	52	Y	-970	0.0	114.3	27	100	1048	162	1325
pr76m9000	76	N	-575300	0.6	0.7	4082	4614	485	851	2666
pr76m9000	76	Y	-575841	0.0	0.5	409	790	204	446	1506

We see that for three out of the five instances, the algorithm using only feasibility cuts has not yet, after two hours, found the optimal solution. Furthermore, significant integrality gaps still remain.

Table 6.9 presents the results for the class of most difficult instances, i.e., class III.

Table 6.9. Computational results for class III instances

Name	k	All Cuts?	z_{best}	Final Gap%	Root Gap%	NN	NLP	t_{LP}	t_{cut}	t
cn100am100	33	N	-2040	40.2	47.8	153	653	3501	125	7200
cn100am100	33	Y	-2266	24.0	32.7	51	239	5242	352	7200
cn100am100	66	N	-5030	4.9	8.2	64	268	2505	249	7200
cn100am100	66	Y	-5115	2.7	8.1	51	228	2851	377	7200
cn100am30	29	N	-180	179.6	247.6	237	1291	5360	188	7200
cn100am30	29	Y	-204	115.8	192.9	42	263	5904	461	7200
cn100am30	58	N	-507	16.6	49.2	142	625	5448	408	7200
cn100am30	58	Y	-501	11.9	45.1	72	348	5111	526	7200
kroA100m300	31	N	-1839	132.9	182.4	171	808	6052	204	7200
kroA100m300	31	Y	-3522	11.0	41.1	90	404	5835	659	7200

Even though the algorithm using all cuts was unable to solve these instances within two hours, the integrality gaps at the end of the two hours are much smaller than the remaining integrality gaps for the algorithm using only feasibility cuts. With a little more time some of the instances would have been solved.

From our computaional experiments, we draw the following conclusions:

- Using the classes of inequalities introduced in this paper improves the overall performance.
- Instances with $k < q$, where q is the cardinality of the optimal circuit if the cardinality constraint is not imposed, are the most difficult to solve by a branch-and-cut algorithm.
- The addition of many cuts significantly increases the LP solution times. The average LP solution time is 2.14 seconds when only parity and disjoint circuit elimination cuts are used and 5.03 seconds when all classes of cuts (except for maximal set inequalities) are used.
- Medium-size instances can be solved effectively by a branch-and-cut approach.

Table B.3 presents detailed cut generation statistics. It shows the number of cuts of each class of inequalities that were added for each of the instances as well as the maximum size of the active linear program. Table 6.10 presents, in a slightly different form, a summary of these statistics. It shows, for several classes of inequalities, the number of cuts added to the linear program as a percentage of the total number of cuts added to the linear program for different values of k.

Table 6.10. Percentage of total cuts for various cut classes

k	Cut	Forest	Path	Tree	k-Partition
q/3	36.25	0.47	1.86	1.71	2.75
2q/3	26.62	1.35	2.88	1.56	0.19
q	18.37	5.44	0.00	1.17	0.06

For the most part, the results in Table 6.10 confirm our intuition. Path, tree, and k-partition inequalities, which are specific to the cardinality constrained circuit polytope,

are more likely to be generated for smaller values of k. Forest inequalities, which are "inherited" from the circuit polytope, are more likely to be generated for larger values of k. Also, a forest inequality with roots R and links U is facet defining only if $|R| + |U| + 2 \leq k$ (Theorem 7), which is less likely to be true for smaller values of k.

6.5. Solving pricing problems

One of the motivations for studying the CCCP is that it arises as the pricing problem in a branch-and-price algorithm for vehicle routing problems with unit demands. In this section, we give some preliminary computational results showing the effectiveness of our branch-and-cut algorithm in solving instances arising in this context.

Instances of CCCP arising in the context of vehicle routing problems with unit demands were created in the following manner. First, capacitated vehicle routing instances from TSPLIB [35] and the thesis of Augerat [2] with general demand were turned into unit demand instances. (The capacity of the vehicle in the new instances was defined to be $k = \lceil b/d \rceil$, where b was the original capacity of the vehicle and d was the average demand at a customer.) These instances were then solved using a column generation approach. As long as possible, negative reduced cost columns were generated using a heuristic consisting of a cheapest insertion construction heuristic followed by 2-exchange and 3-exchange improvement heuristics. If negative reduced cost columns were found, they were added, and the linear program resolved. The CCCP instance for which the heuristic failed to identify a negative reduced cost column was taken as a test instance for our branch-and-cut algorithm. Thus, the CCCP instances that are obtained in this way have some degree of difficulty.

In vehicle routing CCCP instances, we know that all feasible circuits have to contain the depot node v. We enforced this constraint in our branch-and-cut algorithm by setting $y_v = 1$. We also modified the primal heuristic to find a cheapest circuit of length no more than $k - 1$, and then inserted the depot node into the circuit in the cheapest way possible. This is a somewhat naive approach to solving vehicle routing CCCP instances. As pointed out by Bixby et al. [9] and Fischetti et al. [14], many classes of inequalities can be improved by using the knowledge that the depot must be included in any circuit. Also, specialized primal heuristics can be developed. We leave these improvements as a line of further research.

The results of solving these vehicle routing CCCP instances are shown in Table 6.11. The number of nodes in each instance is the first integer in the name of the instance. We report the best solution found (z_{best}), the best lower bound (z_{LB}), the number of evaluated nodes (NN), the number of linear programs solved (NLP), the time and method by which the first negative reduced cost column was found (t_{-}), the time (t_{best}) and method by which the best solution was found, and the total computation time (t). A time limit of two CPU hours was imposed on the solution of each instance.

We see that these CCCP instances are indeed difficult, but that in most instances a negative reduced cost column was identified within 10 minutes and often much faster. Again, we note that our branch-and-cut algorithm can be improved significantly by taking into account that all circuits have to include the depot.

Table 6.11. Computational results on modified TSPLIB instances

Name	z_{best}	z_{LB}	NN	NLP	t_- (Method)	t_{best} (Method)	t
eil22	-23.37	-23.37	1	7	2.4 (Heur.)	2.4 (Heur.)	2.5
eil23	-19.06	-19.06	7	48	2.5 (Heur.)	6.9 (LP)	8.8
eil30	-36.51	-36.51	3	29	2.4 (Heur.)	2.4 (Heur.)	11.0
eil33	-31.23	-31.23	17	241	79.4 (LP)	93.6 (Heur.)	151.3
att48	-3044.55	-3044.55	9	40	2.6 (Heur.)	9.3 (Heur.)	43.3
SP-n51-k4	-61.50	-158.66	182	3642	396.6 (Heur.)	5511.5 (LP)	7200.0
eil51	-11.33	-11.33	71	285	126.8 (Heur.)	320.2 (LP)	466.3
P-n55-k8	-3.36	-3.36	353	2571	401.8 (LP)	471.4 (LP)	6978.6
P-n60-k10	-2.13	-10.36	115	1558	6397.0 (LP)	6397.3 (LP)	7200.0
P-n65-k10	3.35	-11.72	86	1218	NONE	1814.7 (LP)	7200.0
eilA76	0.00	-23.68	73	546	NONE	2780.2 (LP)	7200.0
eilB76	-0.80	-15.90	85	600	183.7 (LP)	183.7 (LP)	7200.0

7. Conclusions

We have presented many classes of facet inducing inequalities for the polyhedron arising from an integer programming formulation of the cardinality constrained circuit problem. A branch-and-cut code based on these inequalities was developed and shown to be computationally effective for medium size problem instances. To be able to solve larger instances, it is necessary to use sparse graph representation techniques. Furthermore, the performance of the algorithm may be improved if better separation heuristics can be developed for some of the classes of inequalities. Our computational results show that adding a simple cardinality constraint to the circuit problem makes the problem much more difficult. This indicates that solving knapsack constrained circuit problems is likely to be even more challenging.

A. Proof of validity of cardinality tree inequalities

The proof relies on the concept of coefficient improvement. We will improve the coefficients of the variables corresponding to edges $e \in E \setminus T$. These coefficients w_e start out at their initial values $2 - l_e^T$, and it is our goal to show that they can be increased to at least $w_{l_e}^T$, thereby strengthening the inequality. Note first that the coefficients of edges $f \in E \setminus T$ with $l_f^T \leq k - 1$ are not changed, so the "new" coefficients $w_{l_f}^T$ are certainly valid in this case. For an edge $f \in E \setminus T$, define the coefficient improvement problem (CIP) for f as

$$\begin{aligned} \max_x \quad & z_f \equiv \sum_{e \in T} x_e + \sum_{e \in E \setminus T} w_e x_e \\ \text{subject to:} \quad & x \text{ is the incidence vector of a circuit } C, \\ & |C| \leq k, \\ & x_f = 1. \end{aligned}$$

With the above definitions, we can state the following lemma.

Lemma 2. *The coefficient $w_f^T = 2 - l_f^T$ can be increased by $2 - z_f$.*

The following lemma will also come in handy:

Lemma 3. *Define P_e^T as the unique path in T between the endpoints of e. If $f \in C \cap (E \setminus T)$ and $(C \setminus f) \cap (E \setminus T) \neq \emptyset$, then $P_f^T \setminus C \subseteq \bigcup_{e \in (C \setminus f) \cap (E \setminus T)} P_e^T$.*

Proof. Consider the graph G_C with edge set $\cup_{e \in C \cap (E \setminus T)} (P_e^T \bigcup C)$. G_C is planar, so draw G_C with C as its outer face, as in Fig. A.1. In this embedding, edges on interior faces are in $\cup_{e \in C \cap (E \setminus T)} P_e^T$. Let $g \in P_f^T \setminus C$. g is on the boundary of two interior faces, so g is also on some P_e^T for $e \in (C \setminus f) \cap (E \setminus T)$.

□

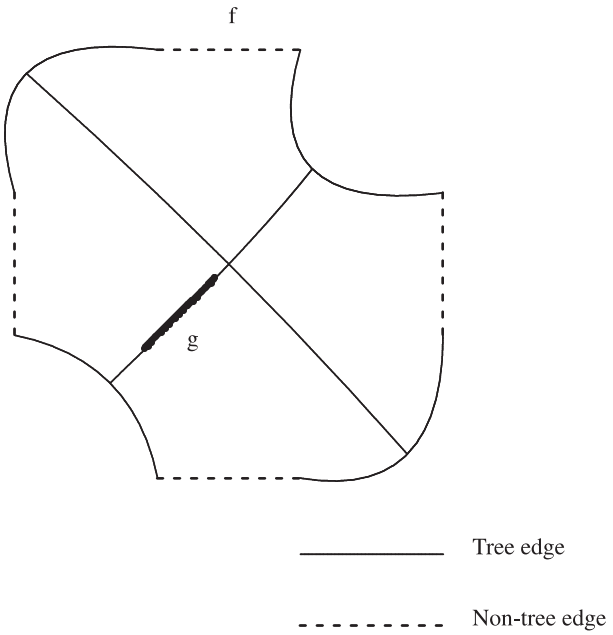

Fig. A.1. An embedding of C and P_e^T

Now, suppose for a contradiction that the coefficient for edge $f \in E \setminus T$ cannot be increased to $w_{l_f}^T$, this implies by Lemma 2 that

$$2 - l_f^T + 2 - z_f \leq w_{l_f}^T - 1. \tag{A.1}$$

Since edge f must be in an optimal solution x to problem (CIP), we have

$$z_f = \sum_{e \in T} x_e + 2 - l_f^T + \sum_{e \in (E \setminus T) \setminus f} w_e x_e. \tag{A.2}$$

Combining (A.1) and (A.2) gives that

$$w_{l_f}^T + \sum_{e\in(E\setminus T)\setminus f} w_e x_e \geq 3 - \sum_{e\in T} x_e. \tag{A.3}$$

Let C be the optimal circuit defined by incidence vector x. Define m as $m \equiv |C \cap (E \setminus T)|$. We now proceed to prove the validity of the coefficient improvements on a case by case basis.

Case I. $k \leq l_f^T \leq \lceil 3k/2 \rceil - 2$, so that $w_{l_f}^T = 4 - 2k + l_f^T$.

Subcase I.a.1. k is even and $\exists g \in (C \setminus f) \cap (E \setminus T)$ such that $l_g^T \geq k$.

In this subcase, it can be shown by definition of $w_{l_e}^T$ that $w_{l_f}^T \leq 2 - k/2$, and $w_g \leq 2 - k/2$. Also by definition of $w_{l_e}^T$, we know that $w_e \leq 0 \; \forall e \in E \setminus T$. Using these facts and (A.3), we may write

$$2 - k/2 + 2 - k/2 \geq w_{l_f}^T + w_g \geq w_{l_f}^T + \sum_{e\in E\setminus T\setminus f} w_e x_e \geq 3 - \sum_{e\in T} x_e,$$

which yields

$$\sum_{e\in T} x_e \geq k - 1.$$

Since $f, g \in C \cap (E \setminus T)$, the circuit C with incidence vector x has $|C| \geq k + 1$. But this contradicts our cardinality constraint.

Subcase I.a.2. k is odd and $\exists g \in (C \setminus f) \cap (E \setminus T)$ such that $l_g^T \geq k$.

Suppose first that one of the following occurs:

- $k \leq l_f^T < \lceil 3k/2 \rceil - 2$,
- $k \leq l_g^T < \lceil 3k/2 \rceil - 2$, or
- $l_g^T = \lceil 3k/2 \rceil - 1 + 2i$ for some $i \in \mathbb{Z}_+$.

By definitions of the weights $w_{l_e}^T$, it can be shown that either $w_{l_f}^T \leq (3 - k)/2$ and $w_g \leq (5 - k)/2$, or $w_{l_f}^T \leq (5 - k)/2$ and $w_g \leq (3 - k)/2$. Using (A.3), we can now write

$$(3 - k)/2 + (5 - k)/2 \geq w_{l_f}^T + w_g \geq w_{l_f}^T + \sum_{e\in E\setminus T\setminus f} w_e x_e \geq 3 - \sum_{e\in T} x_e,$$

which yields

$$\sum_{e\in T} x_e \geq k - 1.$$

As in Subcase I.a.1, we have a contradiction since the circuit C is too long.

Now suppose that $l_f^T = \lceil 3k/2 \rceil - 2$ and $l_g^T = \lceil 3k/2 \rceil - 2$ or $l_g^T = \lceil 3k/2 \rceil + 2i$ for $i \in \mathbb{Z}_+$. We can write $l_g^T = \lceil 3k/2 \rceil + 2(j-1)$ for some $j \in \mathbb{Z}_+$. By definition of the weights, we know that $w_{l_f}^T = w_g = (5-k)/2$. Using (A.3), we have

$$(5-k)/2 + (5-k)/2 \geq w_{l_f}^T + w_g \geq w_{l_f}^T + \sum_{e \in E \setminus T \setminus f} w_e x_e \geq 3 - \sum_{e \in T} x_e,$$

which yields

$$\sum_{e \in T} x_e \geq k - 2.$$

It must be that $\sum_{e \in T} x_e = k - 2$, or else we violated our cardinality constraint.

Suppose f, g, C, and T look as in Fig. A.2. Let l_{xy} denote the length of the path in T from x to y. Then we know that

$$l_{au_1} + l_{ab} + l_{bv_1} = (3k+1)/2, \tag{A.4}$$

$$l_{au2} + l_{ab} + l_{bv_2} = (3k+1)/2 + 2(j-1) \text{ for some } j \in \mathbb{Z}_+, \tag{A.5}$$

$$l_{au_1} + l_{au_2} + l_{bv_1} + l_{bv_2} = k - 2. \tag{A.6}$$

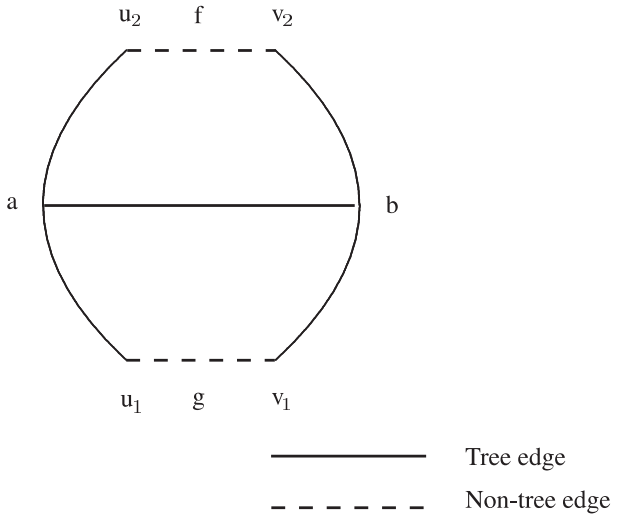

Fig. A.2. The circuit, tree, and edges in $C \cap (E \setminus T)$

Subtracting (A.4) from (A.5) and adding it to (A.6) yields:

$$2(l_{au_2} + l_{bv_2}) = k - 2 + 2(j-1).$$

The left hand side of this equation is even and the right hand side is odd, a contradiction.

Subcase I.b $\nexists g \in C \cap (E \setminus T \setminus f)$ such that $l_g^T \geq k$.

In this subcase we know that $w_{l_f}^T = 4-2k+l_f^T$ and $w_e = (2-l_e^T)\ \forall e \in C \cap (E \setminus T \setminus f)$. Recall that $m \equiv |C \cap (E \setminus T)|$. Beginning with inequality (A.3) we deduce the following:

$$\begin{aligned} 4 - 2k + l_f^T + \sum_{e \in (C \setminus f) \cap (E \setminus T)} (2 - l_e^T) &\geq 3 - \sum_{e \in T} x_e \\ 4 - 2k + l_f^T + 2(m-1) - \sum_{e \in (C \setminus f) \cap (E \setminus T)} l_e^T &\geq 3 + m - k \\ l_f^T - \sum_{e \in (C \setminus f) \cap (E \setminus T)} l_e^T &\geq k + 1 - m. \end{aligned}$$

Using Lemma 3, this is equivalent to

$$|P_f^T \cap C| - \sum_{e \in (C \setminus f) \cap (E \setminus T)} |P_e^T \setminus P_f^T| \geq k + 1 - m. \tag{A.7}$$

It is clear that

$$|C| \geq |P_f^T \cap C| + m. \tag{A.8}$$

Combining (A.7) and (A.8) gives

$$|C| \geq k + 1 + \sum_{e \in E \setminus T \setminus f} |P_e^T \setminus P_f^T| \geq k + 1,$$

a contradiction, since C is too long.

Case II. $l_f^T \geq \lceil 3k/2 \rceil - 1$.

Subcase II.a.1. k is even, and $\exists g \in (C \setminus f) \cap (E \setminus T)$ such that $l_g^T \geq k$.

In this subcase we know by definition of the weights $w_{l_e}^T$ that $w_{l_f}^T = 2 - k/2$, and $w_g \leq 2 - k/2$. A contradiction is derived in a manner similar to Subcase I.a.1.

Subcase II.a.2. k is even, and $\nexists g \in (C \setminus f) \cap (E \setminus T)$ such that $l_g^T \geq k$.

In this subcase, we know that $w_{l_f}^T = 2-k/2$ and $w_e = 2-l_e^T\ \forall e \in (C \setminus f) \cap (E \setminus T)$. Using (A.3), we can get the following inequalities:

$$\begin{aligned} 2 - k/2 + \sum_{e \in (C \setminus f) \cap (E \setminus T)} (2 - l_e^T) &\geq 3 - \sum_{e \in T} x_e \\ 2 - k/2 + 2(m-1) - \sum_{e \in (C \setminus f) \cap (E \setminus T)} l_e^T &\geq 3 + m - k \\ - \sum_{e \in (C \setminus f) \cap (E \setminus T)} l_e^T &\geq 3 - m - k/2. \end{aligned} \tag{A.9}$$

Adding the inequality $l_f^T \geq 3k/2 - 1$ to (A.9), we get

$$l_f^T - \sum_{e \in (C \setminus f) \cap (E \setminus T)} l_e^T \geq k - m + 2.$$

Using Lemma 3 we find that this is equivalent to

$$|P_f^T \cap C| - \sum_{e \in (C \setminus f) \cap (E \setminus T)} |P_e^T \setminus P_f^T| \geq k - m + 2.$$

This leads us to

$$|C| \geq |P_f^T \cap C| + m \geq k + 2 + \sum_{e \in E \setminus T \setminus f} |P_e^T \setminus P_f^T| \geq k + 2,$$

which is a contradiction, since C is too long.

Subcase II.b.1.i. k is odd, $l_f^T = \lceil 3k/2 \rceil - 1 + 2i$ for some $i \in \mathbb{Z}_+$, and $\exists g \in C \cap (E \setminus T \setminus f)$ such that $l_g^T \geq k$.

In this subcase, we know that $w_{l_f}^T = (3-k)/2$. Further, we know that $w_g \leq (5-k)/2$. Using (A.3), we have

$$(3-k)/2 + (5-k)/2 \geq w_{l_f}^T + w_g \geq w_{l_f}^T + \sum_{e \in E \setminus T \setminus f} w_e x_e \geq 3 - \sum_{e \in T} x_e,$$

which yields

$$\sum_{e \in T} x_e \geq k - 1,$$

a contradiction.

Subcase II.b.1.ii. k is odd, $l_f^T = \lceil 3k/2 \rceil - 1 + 2i$ for some $i \in \mathbb{Z}_+$, and $\nexists g \in C \cap (E \setminus T \setminus f)$ such that $l_g^T \geq k$.

We know that $w_{l_f}^T = (3-k)/2$ and $w_e = 2 - l_e^T \ \forall e \in (C \setminus f) \cap (E \setminus T)$. (A.3) implies that

$$(3-k)/2 + \sum_{e \in (C \setminus f) \cap (E \setminus T)} (2 - l_e^T) \geq 3 - \sum_{e \in T} x_e.$$

Using the fact that $\sum_{e \in T} x_e \leq k - m$, this inequality can be manipuated to

$$- \sum_{e \in (C \setminus f) \cap (E \setminus T)} l_e^T \geq (7-k)/2 - m \tag{A.10}$$

Furthermore, we know that

$$l_f^T \geq \lceil 3k/2 \rceil - 1 = (3k-1)/2 \tag{A.11}$$

Adding inequalities (A.10) and (A.11) and applying Lemma 3, we get that

$$|P_f^T \cap C| - \sum_{e \in (C \setminus f) \cap (E \setminus T)} |P_e^T \setminus P_f^T| \geq k - m + 3.$$

Therefore,

$$|C| \geq |P_f^T \cap C| + m \geq k + 3 + \sum_{e \in E \setminus T \setminus f} |P_e^T \setminus P_f^T| \geq k + 3,$$

which gives the contradiction $|C| > k$.

Subcase II.b.2.i. k is odd, $l_f^T = \lceil 3k/2 \rceil + 2i$ for some $i \in \mathbb{Z}_+$, and $\exists g \in C \cap (E \setminus T \setminus f)$ such that $l_g^T \geq k$.

Suppose first that $k \leq l_g^T < \lceil 3k/2 \rceil - 2$ or $l_g^T = \lceil 3k/2 \rceil - 1 + 2i$ for some $i \in \mathbb{Z}_+$. In this case $w_g \leq (3-k)/2$. Using (A.3) we can say

$$(5-k)/2 + (3-k)/2 \geq w_{l_f}^T + w_g \geq w_{l_f}^T + \sum_{e \in E \setminus T \setminus f} w_e x_e \geq 3 - \sum_{e \in T} x_e,$$

which yields

$$\sum_{e \in T} x_e \geq k - 1,$$

a contradiction.

Now suppose that $l_g^T = \lceil 3k/2 \rceil - 2$ or $l_g^T = \lceil 3k/2 \rceil + 2i$ for some $i \in \mathbb{Z}_+$, which can equivalently be written as $l_g^T = \lceil 3k/2 \rceil + 2(j-1)$ for some $j \in \mathbb{Z}_+$. From (A.3), we get

$$w_{l_f}^T + w_g = (5-k)/2 + (5-k)/2 \geq w_{l_f}^T + \sum_{e \in E \setminus T \setminus f} w_e x_e \geq 3 - \sum_{e \in T} x_e.$$

This inequality implies that $\sum_{e \in T} x_e \geq k - 2$, but since edges f and g are also in the circuit, it must be that $\sum_{e \in T} x_e = k - 2$, or else the cardinality constraint would be violated. Referring to Fig. A.2, we know that

$$l_{au_1} + l_{ab} + l_{bv_1} = (3k+1)/2 + 2i, \text{ for some } i \in \mathbb{Z}_+ \quad \text{(A.12)}$$

$$l_{au2} + l_{ab} + l_{bv_2} = (3k+1)/2 + 2(j-1), \text{ for some } j \in \mathbb{Z}_+ \quad \text{(A.13)}$$

$$l_{au_1} + l_{au_2} + l_{bv_1} + l_{bv_2} = k - 2. \quad \text{(A.14)}$$

Subtracting (A.12) from (A.13) and adding it to (A.14) yields:

$$2(l_{au_2} + l_{bv_2}) = k - 2 + 2(i - j + 1).$$

The left hand side of this equation is even and the right hand side is odd, a contradiction.

Subcase II.b.2.ii. k is odd, $l_f^T = \lceil 3k/2 \rceil + 2i$ for some $i \in \mathbb{Z}_+$, and $\nexists g \in C \cap (E \setminus T \setminus f)$ such that $l_g^T \geq k$.

In this subcase, we know that $w^T_{l_f} = (5-k)/2$ and $w_e = 2-l^T_e \ \forall e \in (C\setminus f)\cap(E\setminus T)$. Therefore, (A.3) implies that

$$(5-k)/2 + \sum_{e\in(C\setminus f)\cap(E\setminus T)} (2-l^T_e) \geq 3 - \sum_{e\in T} x_e.$$

Using the fact that $\sum_{e\in T} x_e \leq k-m$, this inequality can be manipulated to give

$$-\sum_{e\in(C\setminus f)\cap(E\setminus T)} l^T_e \geq (5-k)/2 - m \tag{A.15}$$

Furthermore, we know that

$$l^T_f \geq \lceil 3k/2 \rceil = (3k-1)/2 \tag{A.16}$$

Adding inequalities (A.15) and (A.16) and applying Lemma 3, we get that

$$|P^T_f \cap C| - \sum_{e\in(C\setminus f)\cap(E\setminus T)} |P^T_e \setminus P^T_f| \geq k-m+3.$$

Therefore,

$$|C| \geq |P^T_f \cap C| + m \geq k+3+\sum_{e\in E\setminus T\setminus f} |P^T_e \setminus P^T_f| \geq k+3,$$

which gives the contradiction $|C| > k$.

We have shown that in all possible cases, if the coefficient of an edge $e \in E \setminus T$ in the original tree inequality cannot be improved to $w^T_{l_e}$, then there is a contradiction.

Appendix B. Tables of computational results

Table B.1. Computational results on modified TSPLIB instances

Name	k	z_{best}	Final Gap%	Root Gap%	NN	NLP	t_{LP}	t_{cut}	t
bayg29m47	4	-17	0.0	3.9	1	9	0.2	0.1	2.6
bayg29m47	9	-17	0.0	11.9	3	15	0.4	0.3	3.2
bayg29m47	14	-32	0.0	0.0	1	8	0.2	0.1	2.8
bayg29m50	5	-26	0.0	14.3	3	17	0.4	0.3	3.1
bayg29m50	10	-45	0.0	24.0	5	26	1.6	0.8	5.0
bayg29m50	16	-78	0.0	0.0	1	8	0.3	0.1	2.9
bayg29m120	9	-661	0.0	4.9	13	40	3.4	1.0	7.2
bayg29m120	19	-1388	0.0	2.7	7	19	1.1	0.3	3.9
bayg29m120	29	-1870	0.0	0.0	1	8	0.2	0.1	2.7
berlin52m50	4	-80	0.0	0.0	1	5	0.2	0.1	2.8
berlin52m50	8	-80	0.0	30.7	7	23	1.2	0.8	4.8
berlin52m50	12	-123	0.0	0.0	1	7	0.3	0.2	3.1
berlin52m200	12	-1923	0.0	0.6	3	9	0.4	0.2	3.2
berlin52m200	25	-3233	0.0	1.8	5	14	1.1	0.6	4.6
berlin52m200	38	-3686	0.0	2.3	3	15	2.0	1.4	6.8
berlin52m500	17	-7532	0.0	2.1	9	30	2.6	1.4	7.2
berlin52m500	34	-13879	0.0	0.7	5	18	1.8	1.2	6.0
berlin52m500	52	-18458	0.0	0.0	1	2	0.3	0.1	2.9

Name	k	z_{best}	Final Gap%	Root Gap%	NN	NLP	t_{LP}	t_{cut}	t
brazil58m200	4	-418	0.0	3.1	7	19	0.6	0.5	3.8
brazil58m200	8	-589	0.0	16.5	3	16	0.9	0.8	4.7
brazil58m200	12	-607	0.0	29.0	3	19	1.7	1.7	6.9
brazil58m500	16	-5173	0.0	9.7	11	38	11.2	3.1	18.2
brazil58m500	32	-8889	0.0	13.0	9	35	19.0	3.2	27.5
brazil58m500	48	-10432	0.0	9.7	7	28	9.4	2.2	16.8
brazil58m2000	19	-34506	0.0	2.1	13	43	17.4	3.4	25.9
brazil58m2000	38	-65937	0.0	3.3	67	204	322.0	41.5	400.0
brazil58m2000	58	-90605	0.0	0.2	3	12	1.2	0.8	6.1
cn100am100	33	-2266	24.0	32.7	51	239	5242.	351.5	7200.
cn100am100	66	-5115	2.7	8.1	51	228	2851.	377.3	7200.
cn100am100	100	-7544	0.0	1.8	221	471	352.2	694.7	2786.
cn100am30	29	-204	115.8	192.9	42	263	5904.	460.9	7200.
cn100am30	58	-501	11.9	45.1	72	348	5111.	525.7	7200.
cn100am30	87	-750	0.0	2.0	3	13	69.8	16.6	99.5
cn100bm20	11	-81	0.0	92.7	65	269	414.7	172.6	675.4
cn100bm20	22	-134	0.0	59.4	35	145	494.0	132.6	716.6
cn100bm20	33	-151	0.0	45.7	11	45	131.0	37.5	192.6
cn100bm40	31	-761	0.0	11.7	29	76	295.1	41.2	367.4
cn100bm40	62	-1126	0.0	12.8	81	146	678.7	153.2	1075.
cn100bm40	94	-1450	0.0	2.4	5	18	17.3	9.3	70.6
dlo49m700	3	-352	0.0	0.0	1	12	0.3	0.3	3.1
dlo49m700	6	-1089	0.0	0.0	1	7	0.2	0.2	2.8
dlo49m700	10	-1448	0.0	0.0	1	5	0.2	0.1	2.8
dlo49m1000	6	-2889	0.0	0.0	1	7	0.2	0.1	2.9
dlo49m1000	13	-5115	0.0	4.7	12	38	4.4	2.0	10.0
dlo49m1000	20	-6377	0.0	10.7	5	23	1.8	1.4	6.3
dlo49m2000	16	-21633	0.0	2.9	11	30	3.2	1.1	7.4
dlo49m2000	32	-34400	0.0	17.5	218	705	562.2	87.2	731.7
dlo49m2000	49	-52648	0.0	0.0	1	7	0.6	0.3	3.5
dlo61m2000	19	-24719	0.0	1.8	38	106	57.7	51.3	120.8
dlo61m2000	38	-47817	0.0	2.5	54	141	86.8	32.5	139.4
dlo61m2000	58	-70797	0.0	0.4	7	23	3.2	2.5	15.6
dlo61m5000	20	-86022	0.0	0.5	15	46	15.3	22.2	42.8
dlo61m5000	40	-170338	0.0	0.7	13	52	49.4	9.9	68.7
dlo61m5000	61	-251864	0.0	0.1	9	27	3.8	4.3	25.9
eil76m6	13	-13	0.0	24.4	5	24	4.0	3.4	12.2
eil76m6	26	-15	0.0	23.3	11	42	22.4	9.2	41.9
eil76m6	39	-17	0.0	13.2	3	14	3.7	2.1	11.2
eil76m15	24	-230	0.0	3.9	21	73	56.5	16.2	124.5
eil76m15	48	-444	0.0	2.4	29	69	70.4	21.7	133.8
eil76m15	72	-612	0.0	0.0	1	5	1.3	1.1	5.8
eil76m25	25	-489	0.0	2.0	15	44	37.7	7.4	52.4
eil76m25	50	-962	0.0	0.9	7	29	18.8	6.6	31.4
eil76m25	76	-1362	0.0	0.0	1	5	1.0	1.8	6.1
kroA100m200	16	-551	0.0	168.6	93	417	5526.	757.3	6923.
kroA100m200	32	-622	0.0	197.4	41	210	2279.	411.0	2983.
kroA100m200	48	-855	0.0	119.8	17	88	519.0	114.0	713.1
kroA100m300	31	-3522	11.0	41.1	90	404	5835.	659.4	7200.
kroA100m300	62	-6898	0.0	13.3	38	139	1048.	205.2	1366.
kroA100m300	93	-9326	0.0	3.2	13	32	30.0	47.6	117.6
kroC100m200	26	-940	0.0	112.9	35	159	1753.	262.3	2222.
kroC100m200	52	-970	0.0	114.3	27	100	1048.	161.8	1325.
kroC100m200	78	-1171	0.0	71.4	21	78	554.2	107.0	738.1

Name	k	z_{best}	Final Gap%	Root Gap%	NN	NLP	t_{LP}	t_{cut}	t
kroE100m400	32	-7537	0.0	8.4	44	145	705.8	188.0	1033.
kroE100m400	64	-13277	0.0	5.6	131	369	2883.	565.7	3773.
kroE100m400	97	-18290	0.0	1.1	5	14	11.0	13.0	31.0
pr76m1500	18	-9420	0.0	23.0	48	176	110.1	48.8	190.8
pr76m1500	36	-19483	0.0	2.5	59	102	22.9	19.0	50.1
pr76m1500	54	-24376	0.0	1.9	11	23	3.2	3.7	15.4
pr76m9000	25	-201289	0.0	0.9	133	450	290.0	89.9	447.6
pr76m9000	50	-398964	0.0	0.0	5	12	2.2	1.8	7.6
pr76m9000	76	-575841	0.0	0.5	409	790	203.6	446.4	1506.
rat99m12	24	-29	0.0	56.1	63	202	351.0	235.6	708.6
rat99m12	48	-37	0.0	51.7	17	73	186.4	93.1	329.3
rat99m12	73	-40	0.0	55.6	37	119	457.4	108.0	663.8
st70m10	14	-45	0.0	18.8	13	50	39.2	16.4	63.9
st70m10	28	-72	0.0	17.4	21	83	126.7	20.3	170.6
st70m10	42	-97	0.0	9.5	3	19	12.4	6.0	24.5
st70m40	23	-742	0.0	3.8	279	822	1447.	341.4	2381.
st70m40	46	-1474	0.0	1.3	21	58	56.0	23.8	114.1
st70m40	70	-2125	0.0	0.3	3	11	2.3	2.4	12.5

Table B.2. Computational results on modified TSPLIB instances

Name	k	z_{best}	Final Gap%	Root Gap%	NN	NLP	t_{LP}	t_{cut}	t
bayg29m47	4	-17	0.0	76.5	5	22	0.2	0.1	2.7
bayg29m47	9	-17	0.0	165.9	13	46	1.1	0.2	3.9
bayg29m47	14	-32	0.0	36.5	5	14	0.3	0.1	2.8
bayg29m50	5	-26	0.0	99.2	15	41	0.6	0.1	3.4
bayg29m50	10	-45	0.0	65.9	13	52	1.8	0.3	5.0
bayg29m50	16	-78	0.0	0.0	1	8	0.2	0.0	2.9
bayg29m120	9	-661	0.0	6.4	47	117	4.9	0.3	8.6
bayg29m120	19	-1388	0.0	0.7	5	13	0.4	0.1	3.0
bayg29m120	29	-1870	0.0	0.1	3	10	0.2	0.1	2.8
berlin52m50	4	-80	0.0	18.8	3	9	0.3	0.0	3.0
berlin52m50	8	-80	0.0	54.5	23	43	1.5	0.2	4.6
berlin52m50	12	-123	0.0	0.0	1	9	0.3	0.1	3.2
berlin52m200	12	-1923	0.0	0.7	3	9	0.4	0.0	3.0
berlin52m200	25	-3233	0.0	1.9	3	13	0.7	0.2	3.7
berlin52m200	38	-3686	0.0	4.1	3	12	1.2	0.4	4.6
berlin52m500	17	-7532	0.0	2.1	11	29	1.9	0.3	6.0
berlin52m500	34	-13879	0.0	0.7	8	27	3.0	0.8	7.4
berlin52m500	52	-18458	0.0	0.0	1	2	0.4	0.1	3.0
brazil58m200	4	-418	0.0	6.9	7	19	0.5	0.1	3.2
brazil58m200	8	-589	0.0	17.7	7	24	0.9	0.1	4.2
brazil58m200	12	-607	0.0	39.8	11	36	2.8	0.4	6.9
brazil58m500	16	-5173	0.0	9.8	29	79	17.8	1.4	25.3
brazil58m500	32	-8889	0.0	13.0	11	36	13.2	1.4	19.0
brazil58m500	48	-10432	0.0	16.9	13	45	15.4	2.4	32.5
brazil58m2000	19	-34506	0.0	1.9	25	68	28.6	1.6	36.6
brazil58m2000	38	-65937	0.0	3.1	71	189	186.2	11.0	224.6
brazil58m2000	58	-90605	0.0	0.1	5	14	1.3	0.7	6.1

Name	k	z_{best}	Final Gap%	Root Gap%	NN	NLP	t_{LP}	t_{cut}	t
cn100am100	33	-2040	40.2	47.8	153	653	3501.	124.7	7200.
cn100am100	66	-5030	4.9	8.2	64	268	2505.	248.6	7200.
cn100am100	100	-7418	3.0	3.6	1400	1681	807.8	1020.	4029.
cn100am30	29	-180	179.6	247.6	237	1291	5360.	187.9	7200.
cn100am30	58	-507	16.6	49.2	142	625	5448.	407.9	7200.
cn100am30	87	-750	0.0	2.3	3	18	66.2	17.7	93.0
cn100bm20	11	-81	0.0	92.7	265	947	332.5	22.5	486.0
cn100bm20	22	-134	0.0	68.3	145	591	623.6	39.5	891.7
cn100bm20	33	-151	0.0	51.6	45	236	341.7	25.4	502.2
cn100bm40	31	-761	0.0	13.5	45	148	355.4	19.0	532.9
cn100bm40	62	-1126	0.0	6.3	45	83	202.1	36.6	293.7
cn100bm40	94	-1450	0.0	2.5	7	23	14.3	8.1	69.0
dlo49m700	3	-352	0.0	77.0	5	16	0.4	0.1	3.0
dlo49m700	6	-1089	0.0	-0.0	1	7	0.2	0.0	2.8
dlo49m700	10	-1448	0.0	-0.0	1	7	0.2	0.0	2.9
dlo49m1000	6	-2889	0.0	0.0	1	7	0.2	0.0	2.8
dlo49m1000	13	-5115	0.0	4.8	7	26	1.4	0.3	4.7
dlo49m1000	20	-6377	0.0	12.7	5	26	2.6	0.5	6.4
dlo49m2000	16	-21633	0.0	3.0	17	39	3.2	0.4	6.6
dlo49m2000	32	-34400	0.0	19.5	345	1143	746.8	51.2	970.7
dlo49m2000	49	-52648	0.0	0.0	1	8	0.6	0.3	3.6
dlo61m2000	19	-24719	0.0	1.8	39	100	24.5	3.1	36.6
dlo61m2000	38	-47817	0.0	2.3	45	150	81.0	11.3	112.1
dlo61m2000	58	-70797	0.0	1.0	11	27	4.6	1.9	28.9
dlo61m5000	20	-86022	0.0	0.5	133	392	209.8	14.9	282.8
dlo61m5000	40	-170338	0.0	0.7	17	57	38.7	4.5	51.8
dlo61m5000	61	-251864	0.0	0.2	29	50	7.3	5.5	39.6
eil76m6	13	-13	0.0	35.6	9	31	3.2	0.6	7.8
eil76m6	26	-15	0.0	31.7	17	50	23.4	3.3	36.4
eil76m6	39	-17	0.0	16.6	4	22	8.0	1.8	16.2
eil76m15	24	-230	0.0	4.0	55	167	110.0	8.4	159.7
eil76m15	48	-444	0.0	1.6	17	37	15.0	3.6	24.6
eil76m15	72	-612	0.0	0.0	1	5	1.0	0.8	4.8
eil76m25	25	-489	0.0	2.1	23	66	45.4	3.7	59.8
eil76m25	50	-962	0.0	0.9	16	38	14.4	3.7	24.1
eil76m25	76	-1362	0.0	0.1	7	15	2.5	2.6	14.9
kroA100m200	16	-551	0.0	184.7	551	2410	1651.	112.6	2274.
kroA100m200	32	-551	88.9	274.1	181	1150	5657.	322.4	7200.
kroA100m200	48	-855	18.2	139.4	273	1301	5649.	354.5	7200.
kroA100m300	31	-1839	132.9	182.4	171	808	6052.	204.4	7200.
kroA100m300	62	-6898	0.0	13.1	144	406	2092.	238.1	2895.
kroA100m300	93	-9326	0.0	3.4	117	168	129.6	100.1	450.3
kroC100m200	26	-940	0.0	122.3	279	1188	4056.	215.6	5155.
kroC100m200	52	-970	25.2	119.0	231	970	5667.	418.7	7200.
kroC100m200	78	-1171	0.0	78.9	115	491	2165.	218.8	2838.
kroE100m400	32	-7537	0.0	9.3	116	390	1504.	91.8	1926.
kroE100m400	64	-13277	0.0	8.2	69	200	945.5	115.1	1188.
kroE100m400	97	-18290	0.0	1.7	31	56	37.0	33.7	128.0
pr76m1500	18	-9420	0.0	23.2	95	293	111.0	9.0	155.0
pr76m1500	36	-19483	0.0	2.1	37	60	8.3	1.7	15.1
pr76m1500	54	-24376	0.0	1.6	23	38	4.4	2.5	16.1
pr76m9000	25	-201289	0.0	0.9	57	173	67.1	5.3	89.8
pr76m9000	50	-398964	0.0	0.0	7	16	2.4	1.0	8.0
pr76m9000	76	-575300	0.6	0.7	4082	4614	484.8	850.6	2666.

Name	k	z_{best}	Final Gap%	Root Gap%	NN	NLP	t_{LP}	t_{cut}	t
rat99m12	24	-29	0.0	71.2	139	590	654.3	65.0	986.5
rat99m12	48	-37	0.0	79.0	116	504	1111.	177.1	1629.
rat99m12	73	-40	0.0	63.3	245	892	1241.	236.2	1996.
st70m10	14	-45	0.0	21.0	83	261	94.6	8.2	155.0
st70m10	28	-72	0.0	18.8	21	79	76.7	6.2	108.9
st70m10	42	-97	0.0	8.2	7	24	9.8	2.0	16.8
st70m40	23	-742	0.0	3.8	599	2000	1854.	113.4	2723.
st70m40	46	-1474	0.0	1.3	25	48	31.6	3.4	51.3
st70m40	70	-2125	0.0	0.3	9	17	2.0	1.6	15.5

Table B.3. Cut statistics for modified WGPLIB instances

Name	k	Total Cuts	Max Loaded	Parity	DCE	Cut	2For	Simple Forest	Path	Tree	K-Part
bayg29m47	4	75	75	17	11	34	0	0	0	6	7
bayg29m47	9	223	223	16	78	99	0	0	11	10	9
bayg29m47	14	193	193	20	90	75	0	0	0	6	2
bayg29m50	5	198	198	19	53	62	0	0	27	12	25
bayg29m50	10	381	381	17	149	157	1	1	22	19	15
bayg29m50	16	230	230	22	113	85	0	0	0	6	4
bayg29m120	9	537	537	21	173	182	0	2	88	23	48
bayg29m120	19	382	382	20	200	121	3	4	20	11	3
bayg29m120	29	158	158	2	132	20	2	0	0	2	0
berlin52m50	4	29	29	10	6	8	0	0	0	3	2
berlin52m50	8	201	201	10	90	68	0	2	10	15	6
berlin52m50	12	118	118	13	54	40	3	1	0	5	2
berlin52m200	12	109	109	14	50	33	4	1	0	4	3
berlin52m200	25	325	325	24	176	111	4	4	0	6	0
berlin52m200	38	663	663	18	392	227	11	5	0	10	0
berlin52m500	17	443	443	19	265	117	8	1	17	15	1
berlin52m500	34	468	468	14	290	116	3	3	34	8	0
berlin52m500	52	65	65	1	50	11	2	0	0	1	0
brazil58m200	4	83	83	17	24	23	0	0	10	9	0
brazil58m200	8	230	230	18	108	72	0	1	0	10	21
brazil58m200	12	479	479	23	260	167	0	0	0	13	16
brazil58m2000	19	1068	1068	38	636	359	2	2	0	25	6
brazil58m2000	38	5477	1857	61	3485	1120	38	244	419	110	0
brazil58m2000	58	318	318	2	264	40	8	0	0	4	0
brazil58m500	16	784	784	29	422	289	1	0	5	19	19
brazil58m500	32	1075	1075	28	733	271	6	17	0	19	1
brazil58m500	48	1016	1016	19	697	269	17	1	0	13	0
cn100am100	33	11913	4271	79	7218	4115	0	26	0	150	325
cn100am100	66	11021	4028	101	8112	2614	13	26	2	150	3
cn100am100	100	9434	1950	6	7807	317	355	872	0	77	0
cn100am30	29	13700	5786	61	8669	4164	0	8	1	190	607
cn100am30	58	16280	4087	112	12463	3430	5	24	7	236	3
cn100am30	87	1446	1446	60	912	427	6	31	0	10	0
cn100bm20	11	5079	1672	71	2750	1538	0	3	55	169	493
cn100bm20	22	4909	1997	59	2880	1700	0	20	46	90	114
cn100bm20	33	2220	2132	64	1319	776	0	27	0	30	4
cn100bm40	31	2532	2471	53	1501	773	2	21	140	33	9
cn100bm40	62	4165	2842	94	2779	1099	33	94	16	49	1
cn100bm40	94	1234	1234	43	811	332	25	14	0	9	0
dlo49m700	3	62	62	18	4	27	0	0	7	6	0
dlo49m700	6	67	67	17	21	22	0	0	0	4	3
dlo49m700	10	92	92	22	36	30	0	0	0	3	1
dlo49m1000	6	67	67	17	21	22	0	0	0	4	3
dlo49m1000	13	721	721	41	375	235	10	22	18	19	1
dlo49m1000	20	532	532	36	321	149	7	6	0	11	2
dlo49m2000	16	495	495	42	296	132	6	4	1	14	0
dlo49m2000	32	15795	1394	210	11200	2565	129	550	722	386	33
dlo49m2000	49	202	202	2	135	51	11	0	0	3	0
dlo61m2000	19	2212	1366	62	1193	489	13	44	366	44	1
dlo61m2000	38	3300	1424	76	2303	556	18	10	271	66	0
dlo61m2000	58	502	502	5	409	62	16	3	0	7	0

Name	k	Total Cuts	Max Loaded	Parity	DCE	Cut	2For	Simple Forest	Path	Tree	K-Part
dlo61m5000	20	1155	1155	53	618	313	3	20	126	21	1
dlo61m5000	40	1654	1466	62	1155	342	16	30	18	31	0
dlo61m5000	61	627	627	1	570	22	24	4	0	6	0
eil76m6	13	675	675	33	300	276	0	0	28	15	23
eil76m6	26	1662	1662	38	836	672	10	5	72	28	1
eil76m6	39	829	829	47	423	344	2	3	0	10	0
eil76m15	24	1936	1402	49	921	698	10	19	197	41	1
eil76m15	48	2029	1767	56	1268	606	23	39	7	30	0
eil76m15	72	292	292	6	185	96	2	1	0	2	0
eil76m25	25	1312	1312	38	663	547	3	1	34	21	5
eil76m25	50	1111	1111	56	609	349	18	14	52	13	0
eil76m25	76	181	181	0	147	27	6	0	0	1	0
kroA100m200	16	17699	4670	61	8246	7530	0	18	22	285	1537
kroA100m200	32	10492	4406	62	6466	3717	0	8	36	153	50
kroA100m200	48	4967	3168	65	3070	1766	0	0	0	64	2
kroA100m300	31	19009	4979	67	11567	6734	1	77	129	258	176
kroA100m300	62	6421	3071	68	4298	1843	15	10	100	86	1
kroA100m300	93	1411	1411	38	948	342	28	43	0	12	0
kroC100m200	26	8575	4808	58	4715	3509	0	3	27	110	153
kroC100m200	52	5903	4622	57	3630	2142	0	7	0	66	1
kroC100m200	78	4637	3676	60	2709	1816	0	1	0	51	0
kroE100m400	32	6257	3913	67	3877	1770	18	163	277	74	11
kroE100m400	64	15054	4194	85	9799	3853	44	50	1051	171	1
kroE100m400	97	938	938	25	649	225	18	14	0	7	0
pr76m1500	18	3736	1248	68	1749	1598	0	7	125	105	84
pr76m1500	36	1738	956	71	583	424	1	7	634	17	1
pr76m1500	54	593	593	46	330	195	7	10	0	5	0
pr76m9000	25	8256	1194	106	3784	3168	1	13	939	242	3
pr76m9000	50	462	462	50	244	153	7	3	0	5	0
pr76m9000	76	10249	1368	0	8857	212	466	562	0	152	0
rat99m12	24	7590	2234	80	4297	2999	6	41	8	125	34
rat99m12	48	3545	2532	72	2217	1150	8	26	23	49	0
rat99m12	73	4637	3184	79	3143	1209	47	84	0	75	0
st70m10	14	1486	1486	44	705	575	0	0	89	32	41
st70m10	28	2856	1892	53	1592	1102	3	6	46	48	6
st70m10	42	1146	1146	56	658	397	8	13	0	13	1
st70m40	23	22963	2203	145	13551	7867	54	194	519	441	192
st70m40	46	1884	1669	53	1076	395	28	42	262	26	2
st70m40	70	452	452	0	362	72	11	3	0	4	0

Acknowledgements. The authors appreciate the constructive comments of the referees and associate editor, resulting in an improved presentation.

References

1. Applegate, D., Bixby, R., Cook, W., Chvátal, V. (1996): Personal communication
2. Augerat, Ph. (1995): Polyhedral study of the capacitated vehicle routing. PhD thesis, Universite Joseph Fourrier, Grenoble
3. Balas, E. (1989): The prize collecting traveling salesman problem. Networks **19**, 621–636
4. Balas, E., Ceria, S., Cornuejols, G. (1996): Mixed 0-1 programming by lift-and-project in a branch-and-cut framework. Management Science **42**, 1229–1246
5. Bauer, P. (1994): A Polyhedral Approach to the Weighted Girth Problem. PhD thesis, Universität zu Köln
6. Bauer, P. (1997): The circuit polytope: facets. Mathematics of Operations Research **22**, 110–145
7. Bauer, P., Linderoth, J.T., Savelsbergh, M.W.P. (1998): A branch and cut approach to the cardinality constrained circuit problem. Technical Report TLI-98-04, The Logistics Institute, Georgia Institute of Technology, 1998. Available from `http://tli.isye.gatech.edu/research/papers/files/TLI9804.pdf`
8. Bienstock, D., Goemans, M., Simchi-Levi, D., Williamson, D. (1993): A note on the prize collecting traveling salesman problem. Mathematical Programming **59**, 413–420
9. Bixby, A., Coullard, C., Simchi-Levi, D. (1996): The capacitated prize-collecting traveling salesman problem. Technical Report TR 96-10, Northwestern University
10. Carr, R. (1997): Separating clique trees and bipartition inequalities having a fixed number of handles and teeth in polynomial time. Mathematics of Operations Research **22**, 257–265
11. CPLEX Optimization, Inc. Using the CPLEX Callable Library, 1995

12. Desrosiers, J., Soumis, F., Desrochers, M. (1984): Routing with time windows by column generation. Networks **14**, 545–565
13. Edmonds, J., Karp, R.M. (1972): Theoretical improvements in algorithmic efficiency of network flow problems. Journal of ACM **19**, 248–264
14. Fischetti, M., González, J.J.S., Toth, P. (1998): Solving the orienteering problem through branch-and-cut. INFORMS Journal on Computing **10**, 133–148
15. Fleischer, L., Tardos, É. (1999): Separating maximally violated combs in planar graphs. Mathematics of Operations Research **24**, 130–148
16. Garey, M.R., Johnson, D.S. (1979): Computers and intractability: a guide to the theory of NP-completeness. W.H. Freeman and Company, New York
17. Golden, B.L., Wang, Q., Liu, L. (1988): A multifacted heuristic for the orienteering problem. Naval Research Logistics **35**, 359–366
18. Goldschmidt, O., Hochbaum, D. (1994): A polynomial algorithm for the k-cut problem for fixed k. Mathematics of Operations Research **19**, 24–37
19. Grötschel, M., Pulleyblank, W.R. (1986): Clique tree inequalities and the symmetric travelling salesman problem. Mathematics of Operations Research **11**, 537–569
20. Gusfield, D. (1990): Very simple methods for all pairs network flow analysis. SIAM Journal of Computing **19**, 143–155
21. Hartmann, M., Ozluk, O. (1998): Facets of the p-cycle polytope. Technical Report UNC/OR TR98-1, Department of Operations Research, University of North Carolina
22. Hu, T.C. (1974): Optimum communication spanning trees. SIAM Journal on Computing **3**, 188–195
23. Hyafil, L., Rivest, R.L. (1973): Graph partitioning and constructing optimal decision trees and polynomial complete problems. Technical Report 33, IRIA-Laboria, Rocquencourt, France
24. Johnson, D.S., Lenstra, J.K., Rinnooy Kan, A.H.G. (1978): The complexity of the network design problem. Networks **8**, 279–285
25. Jünger, M., Reinelt, G., Thienel, S. (1994): Provably good solutions for the traveling salesman problem. Zeitschrift für Operations Research **40**, 183–217
26. Leifer, A.C., Rosenwein, M.B. (1994): Strong linear programming relaxations for the orienteering problem. European Journal of Operational Research **73**, 517–523
27. Nemhauser, G.L., Savelsbergh, M.W.P., Sigismondi, G.C. (1994): MINTO, a Mixed INTeger Optimizer. Operations Research Letters **15**, 47–58
28. Nguyen, V.-H., Maurras, J.-F. (2000): The k-cycle polytope: I. Lifting theorems and lifting algorithms. Technical report 358, Laboratoire d'Informatique de Marseille, Marseille, France
29. Nguyen, V.-H., Maurras, J.-F. (2000): The k-cycle polytope: II. Facets. Technical report 359, Laboratoire d'Informatique de Marseille, Marseille, France
30. Padberg, M.W., Rao, M.R. (1982): Odd minimum cut sets and b-matchings. Mathematics of Operations Research **7**, 67–80
31. Padberg, M.W., Rinaldi, G. (1990): Facet identification for the symmetric traveling salesman problem. Mathematical Programming **47**, 219–257
32. Pillai, R. (1992): The Traveling Salesman Subset-Tour Problem with one Additional Constraint (TSSP+1). PhD thesis, University of Tennessee, Knoxville
33. Prim, R.C. (1957): Shortest connection networks and some generalizations. Bell System Technological Journal **36**, 1389–1401
34. Ramesh, R., Yoon, Y.S., Karwan, M.H. (1992): An optimal algorithm for the orienteering tour problem. ORSA Journal on Computing **4**, 155–165
35. Reinelt, G. (1991): TSPLIB – a traveling salesman problem library. ORSA Journal on Computing **3**, 376–384
36. Saran, H., Vazirani, V.V. (1995): Finding k cuts within twice the optimal. SIAM Journal of Computing **24**, 101–108
37. Savelsbergh, M.W.P., Uma, R.N., Wein, J. (1998): An experimental study of LP-based approximation algorithms for scheduling problems. In: Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 453–461
38. Seymour, P.D. (1979): Sums of circuits. In: Bondy, Murty, eds., Graph Theory and Related Topics. pp. 341–355, Academic Press, New York
39. Wang, Y. (1995): Fleet assignment, Eulerian subtours and extended Steiner trees. PhD thesis, Georgia Institute of Technology