SIAM J. DISCRETE MATH.
Vol. 22, No. 4, pp. 1480–1487

ODD MINIMUM CUT SETS AND b-MATCHINGS REVISITED*

ADAM N. LETCHFORD†, GERHARD REINELT‡, AND DIRK OLIVER THEIS§

Abstract. The famous Padberg–Rao separation algorithm for b-matching polyhedra can be implemented to run in $\mathcal{O}(|V|^2|E|\log(|V|^2/|E|))$ time in the uncapacitated case, and in $\mathcal{O}(|V||E|^2\log(|V|^2/|E|))$ time in the capacitated case. We give a new and simple algorithm for the capacitated case which can be implemented to run in $\mathcal{O}(|V|^2|E|\log(|V|^2/|E|))$ time.

Key words. matching, polyhedra, separation

AMS subject classifications. Primary, 90C27; Secondary, 05C70, 90C57, 90C10

DOI. 10.1137/060664793

1. Introduction.

Let $G = (V, E)$ be an undirected graph, let $b \in \mathbb{Z}_+^V$ be a vector of vertex capacities, and let $u \in \mathbb{Z}_+^{|E|}$ be a vector of edge capacities. A *u-capacitated b-matching* is a family of edges, possibly containing multiple copies, such that

- for each $i \in V$, there are at most b_i edges in the family incident on i;
- at most, u_e copies of edge e are used.

If we define for each edge e the integer variable x_e, representing the number of times e appears in the matching, then the incidence vectors of u-capacitated b-matchings are the solutions to

$$\sum_{e\in\delta(i)} x_e \le b_i \quad \text{for all } i \in V, \tag{1}$$

$$0 \le x_e \le u_e \quad \text{for all } e \in E, \tag{2}$$

$$x_e \in \mathbb{Z} \quad \text{for all } e \in E. \tag{3}$$

Here, as usual, $\delta(i)$ represents the set of edges incident on i.

The convex hull in $\mathbb{R}^{|E|}$ of solutions to (1)–(3) is called the *u-capacitated b-matching polytope*. Edmonds and Pulleyblank (see [Edm65] and [Pul73]) gave a complete linear description of this polytope. It is described by the *degree inequalities* (1), the *bounds* (2), and the following *blossom inequalities*:

$$\sum_{e\in E(W)} x_e + \sum_{f\in F} x_f \le \left\lfloor \frac{b(W) + \sum_{f\in F} u_f}{2} \right\rfloor \tag{4}$$

$$\text{for all } W \subset V, F \subset \delta(W) \text{ with } b(W) + \textstyle\sum_{f\in F} u_f \text{ odd.}$$

*Received by the editors July 11, 2006; accepted for publication (in revised form) May 13, 2008; published electronically October 1, 2008. An extended abstact of this paper appeared in Integer Programming and Combinatorial Optimization 10, Lecture Notes in Comput. Sci. 3064, Springer-Verlag, Berlin, 2004, pp. 196–205.

http://www.siam.org/journals/sidma/22-4/66479.html

†Department of Management Science, Lancaster University, Lancaster LA1 4YW, UK (A.N. Letchford@lancaster.ac.uk).

‡Institute of Computer Science, University of Heidelberg, Heidelberg, Germany (Gerhard. Reinelt@informatik.uni-heidelberg.de).

§Service de Géométrie, Combinatoire et Théorie des Groupes, Département de Mathématique, Université Libre de Bruxelles, Brussels, Belgium (Dirk.Theis@ulb.ac.be). This author's research was supported by Deutsche Forschungsgemeinschaft (DFG, RE 776/9-1).

Here, $E(W)$ (respectively, $\delta(W)$) represents the set of edges with both endvertices (respectively, exactly one endvertex) in W, and $b(W)$ denotes $\sum_{i\in W} b_i$.

An important special case is where the upper bounds u_e are not present (or, equivalently, $u_{ij} \geq \max\{b_i, b_j\}$ for all $\{i,j\} \in E$). The associated *uncapacitated* b-matching polytope is described by the degree inequalities, the nonnegativity inequalities $x_e \geq 0$ for all $e \in E$, and the *simplified blossom inequalities*

$$\sum_{e\in E(W)} x_e \leq \left\lfloor \frac{b(W)}{2} \right\rfloor \quad \text{for all } W \subset V \text{ with } b(W) \text{ odd.} \tag{5}$$

In their seminal paper, Padberg and Rao [PR82] devised an efficient combinatorial *separation algorithm* for b-matching polytopes. A separation algorithm is (see [GLS88]) a procedure which, given a point $x^* \in \mathbb{Q}^{|E|}$ lying outside of the polytope, finds a linear inequality which is valid for the polytope yet violated by x^*. Clearly, testing if a degree inequality or bound is violated can be performed in linear time; therefore, the main contribution of [PR82] is to identify violated blossom inequalities.

For *uncapacitated* b-matching, the Padberg–Rao algorithm involves the solution of up to $|V|-1$ max-flow problems on graphs with $\mathcal{O}(|V|)$ vertices and $\mathcal{O}(|E|)$ edges. Using the well-known *preflow push* algorithm [GT88] to solve the max-flow problems, this leads to an overall running time of $\mathcal{O}(|V|^2|E|\log(|V|^2/|E|))$.

The Padberg–Rao separation algorithm for *capacitated* b-matching, however, is substantially more time consuming. It involves the solution of up to $|V|+|E|-1$ max-flow problems on graphs with $\mathcal{O}(|E|)$ vertices and $\mathcal{O}(|E|)$ edges. Using the preflow push algorithm, this leads to a worst-case running time of $\mathcal{O}(|E|^3\log|V|)$.

In 1987, Grötschel and Holland [GH87] observed that the max-flow computations needed in the capacitated case can in fact be carried out on graphs with only $\mathcal{O}(|V|)$ vertices and $\mathcal{O}(|E|)$ edges. Although the idea behind this is simple, it reduces the overall running time for the capacitated case to $\mathcal{O}(|V||E|^2\log(|V|^2/|E|))$.

In this paper, we propose a new separation algorithm for the capacitated case, whose running time is the same as that for the uncapacitated case. As well as being faster than the previous approaches, the new algorithm is simpler and easier to implement. It also has a surprisingly simple proof of correctness. Like the previous algorithms, it can also be easily adapted to the case of *perfect* capacitated b-matchings, in which the inequalities (1) are changed to equations, and to certain other problems in the matching family.

The importance of blossom separation extends, of course, far beyond matching problems. This is because blossom-like inequalities can be defined for many important $\mathcal{NP}$-hard combinatorial optimization problems involving constraints on vertex degrees and bounds on the variables. The most famous problem of this type is the *traveling salesman problem*, for which certain blossom inequalities are facet-inducing and make very useful cutting planes (see [GP79a], [GP79b], [PR90]). Other problems for which blossom-like inequalities can be defined and separated more quickly using the new algorithm include, for example, the *Capacitated Vehicle Routing Problem* [LLE04], the *Rural Postman Problem* [GL00], the *Capacitated Arc Routing Problem* [BB98], the *k-edge Connected Spanning Subgraph Problem* [Mah94], and the *Network Connectivity Problem* [GMS92].

Another very interesting application of blossom separation is described in [CF96]. In that paper, a family of cutting planes for general integer linear programs is defined, called $\{0, \frac{1}{2}\}$-*Chvátal–Gomory cuts*, and it is shown that the associated separation problem can be solved in polynomial time if the constraint matrix satisfies certain

conditions. Under one of these conditions, the separation problem essentially reduces to blossom separation, and our algorithm can be used. (We omit details for the sake of brevity.)

Note: An extended abstract of this paper appeared in the proceedings of IPCO X [LRT04]. However, the proof of correctness of the algorithm has been simplified substantially.

2. Review of key concepts. To make the paper self-contained, and also for the sake of clarity, we review in this section some key concepts from the literature.

2.1. Cut-trees. The notion of a *cut-tree* was introduced by Gomory and Hu [GH61]. Let $G = (V, E)$ be an undirected graph, $w \in \mathbb{Q}_+^{|E|}$ a vector of edge weights, and $X \subset V$ a set of *terminal vertices*. A cut-tree is an edge-weighted tree, spanning X, that compactly represents the minimum cut in G between every pair of vertices in X. More formally, the cut-tree consists of a mapping $\pi\colon V \to X$ with $\pi(x) = x$ for all $x \in X$, and an adjacency relation $\sim$ on the set X. The adjacency relation shall make the set of terminal vertices into a tree, i.e., $x \sim y$ means that x and y are connected by an edge of the cut-tree. Deleting an edge $x \sim y$ of the cut-tree partitions the set X into two sets X_x and X_y, and thus defines a cut $(U, \bar{U})$ in G by letting $U := \pi^{-1}(X_x)$ and $\bar{U} := \pi^{-1}(X_y)$. We call this the cut *induced* by the edge $x \sim y$. Now, the following condition must hold:

(6) for all pairs $x, y \in X$ with $x \sim y$, the cut induced by the edge $x \sim y$ is a minimum (x, y)-cut in G with respect to the weights w.

The Gomory–Hu algorithm for producing a cut-tree involves solving $|X| - 1$ max-flow/min-cut problems on a series of graphs obtained from G by suitable contractions. Using the preflow push algorithm as a subroutine, the Gomory–Hu algorithm runs in $\mathcal{O}(|X||V||E| \log(|V|^2/|E|))$ time.

2.2. Minimum odd cuts. We now move on to consider *minimum odd cuts*. As before, let $G = (V, E)$ be an undirected graph with edge weights $w \in \mathbb{Q}_+^{|E|}$. Let $T \subset V$, with $|T|$ even, be a set of vertices that are labelled *odd*. A cut $\delta(U)$ is called *T-odd*, or just *odd* for short, if $|T \cap U|$ is an odd number. The *minimum odd cut problem* asks for an odd cut $\delta(U)$ with minimum weight $w(\delta(U))$.

In [PR82], Padberg and Rao gave a surprisingly simple algorithm to compute minimum odd cuts. It involves simply invoking the Gomory–Hu algorithm with terminal set T and then checking each of the $|T| - 1$ cuts produced; see Algorithm 1. It is easy to show that the bottleneck of the algorithm is the Gomory–Hu call, and therefore the algorithm can be implemented to run in $\mathcal{O}(|T||V||E| \log(|V|^2/|E|))$ time.

2.3. The Padberg–Rao separation algorithms. Padberg and Rao then showed how to reduce the blossom separation problem to the minimum odd cut problem.

First, consider the uncapacitated case. We introduce, for each $i \in V$, the term $s_i := b_i - \sum_{e \in \delta(i)} x_e$, which can be viewed as the *slack* of the corresponding degree inequality (1). The simplified blossom inequality (5) can then easily be rewritten to take the following form:

$$\sum_{i \in W} s_i + \sum_{e \in \delta(W)} x_e \geq 1. \tag{7}$$

Now, let $x^* \in \mathbb{R}_+^{|E|}$ be a given vector satisfying the degree inequalities. Take the graph G and assign the weight x_e^* to each $e \in E$. Add a new *dummy* vertex v and, for each

Algorithm 1. Minimum odd cut [PR82]

Input:

Graph G, set $T \subset V$ of odd vertices, weight vector $w \in \mathbb{Q}_+^{|E|}$.

Output:

A minimum odd cut.

1: Let $r = \infty$.
2: Compute a cut-tree for the graph G with weights w and terminal vertex set T.
3: **For** each of the $|T| - 1$ edges of the cut-tree
4: Let $\delta(U)$ denote the cut induced by the cut-tree edge.
5: *Check the cut:*
If $|T \cap U|$ is odd and $w(\delta(U)) < r$, then store U and set $r := w(\delta(U))$.
6: **End for**
7: Output the cut $\delta(U)$.

$i \in V$, an edge $\{i, v\}$ with weight $s_i^* := b_i - \sum_{e \in \delta(i)} x_e^*$. Denote the resulting graph by $G^+ = (V^+, E^+)$. Now, label the vertices $i \in V$ that have an odd value of $b(i)$ *odd*. Also label the dummy vertex v *odd* if $b(V)$ is odd.

Clearly, there is a violated blossom inequality in the form (7) if and only if there exists an odd cut in G^+ of weight less than 1. Thus, one can use the minimum odd cut algorithm described in the previous subsection. As a result, blossom separation can be performed in $\mathcal{O}(|V|^2|E| \log(|V|^2/|E|))$ time in the uncapacitated case. This is $\mathcal{O}(|V|^4)$ in the worst case.

The capacitated case is a little more complicated. First, the blossom inequality (4) needs to be rewritten in the following form:

$$\sum_{i \in W} s_i + \sum_{e \in \delta(W) \setminus F} x_e + \sum_{e \in F} (u_e - x_e) \geq 1. \tag{8}$$

Now, let x^* be a given vector satisfying the degree inequalities and bounds. Padberg and Rao begin by constructing the so-called split graph, which is obtained from the graph G^+ mentioned above by subdividing each edge $\{i, j\} \in E$ into two halves. (That is, each edge $\{i, j\} \in E$ is replaced with two edges, say $\{i, k\}$ and $\{j, k\}$.) One of the two halves (which we shall call the *normal* half) is given a weight of x_e^*, whereas the other half (which we shall call the *complemented* half) is given a weight of $u_e - x_e^*$.

Parities are then assigned to the edges and vertices of the split graph. The normal halves are labelled *even*, the complemented halves are labelled *even* or *odd* according to whether the corresponding upper bound u_e is even or odd, and the edge $\{i, v\}$ is labelled *even* or *odd* according to whether the upper bound b_i is even or odd. Finally, a vertex is labelled *even* or *odd* according to whether it is incident on an even or odd number of odd edges.

It can then be shown that there is a violated blossom inequality in the form (8) if and only if there exists an odd cut in the split graph of weight less than 1. Unfortunately, the split graph is substantially larger than G, with $|V|+|E|+1$ vertices and $2|E| + |V|$ edges. Thus, the minimum odd cut computation now involves $\mathcal{O}(|E|)$ max-flow computations in graphs with $\mathcal{O}(|E|)$ vertices and edges. Using the preflow push algorithm, this leads to an overall running time $\mathcal{O}(|E|^3 \log |V|)$, or $\mathcal{O}(|V|^6 \log |V|)$ in the worst case.

2.4. The Grötschel–Holland modification. As mentioned in the introduction, a simple but significant improvement to the algorithm for the capacitated case

was made by Grötschel and Holland [GH87]. (In fact, they were only interested in 2-matchings, but the same argument applies in the general case.) Consider an arbitrary edge $e = \{i, j\} \in E$. It is obvious that the maximum amount of flow which can be directly sent from i to j in the split graph is equal to the minimum of x^*_e and $u_e - x^*_e$. Consequently, at each iteration of the Gomory–Hu algorithm, for each $e \in E$, one of the two halves can be contracted before computing a max-flow. As a result, the max-flow problems can be conducted on graphs which are of a similar size to G. This reduces the overall running time to $\mathcal{O}(|V||E|^2 \log(|V|^2/|E|))$, or $\mathcal{O}(|V|^5)$ in the worst case.

3. The new algorithm. We now present a new, faster algorithm for blossom separation in the capacitated case. In fact, our algorithm solves a slightly more general problem. Given a graph $G = (V, E)$ and a set $T \subset V$ of odd vertices, let us say that a *blossom* is a pair (U, F) consisting of a set of vertices $U \subset V$ and a set of edges $F \subset \delta(U)$ with the property that $|T \cap U| + |F|$ is an odd number. Now suppose that we are given, in addition, two nonnegative weight vectors $c, c' \in \mathbb{Q}_+^{|E|}$. We define the *blossom minimization problem* as the problem of finding a blossom whose *value*

$$\beta(U, F) := \sum_{e \in \delta(U) \setminus F} c_e + \sum_{f \in F} c'_f$$

is as small as possible. To cast the blossom separation problem in this form, it suffices to construct the graph G^+ used by Padberg and Rao for the uncapacitated case (see subsection 2.3), and set

$$(c_e, c'_e) := \begin{cases} (x^*_e, u_e - x^*_e) & \text{if } u_e \text{ is odd,} \\ (\min(x^*_e, u_e - x^*_e), \infty) & \text{if } u_e \text{ is even} \end{cases}$$

for all $e \in E$. For the edges $\{i, v\}$ of G^+, we set $c_{iv} := s^*_i$ and $c'_{iv} := \infty$ (v is the dummy vertex in G^+). With this construction, it is an easy exercise to verify the following proposition.

PROPOSITION 3.1. *Let (U, F^+) be a blossom in G^+ with the weight vectors (c, c') just described. W.l.o.g. we may assume that the dummy vertex v is not in U. Let*

$$W := U \quad \textit{and} \quad F := F^+ \cup \{e \in \delta(W) \mid u_e \textit{ even and } u_e - x^*_e x^*_e < x^*_e\}.$$

Then W and F satisfy the conditions required in the definition of a blossom inequality, and therefore we have

$$\sum_{i \in W} s_i + \sum_{e \in \delta(W) \setminus F} x_e + \sum e \in F(u_e - x_e) = \beta(U, F^+). \tag{9}$$

(The left-hand side of the equation is the left-hand side of inequality (8)*.) Conversely, every blossom inequality for which $F \subset \delta(W)$ is chosen such that the left-hand side in* (8) *is minimized defines a blossom (U, F^+) by letting $U := W$ and $F^+ := \{e \in F \mid u_e \textit{ odd}\}$, for which* (9) *holds.*

Our algorithm for the blossom minimization problem is displayed as Algorithm 2.

Remark 3.2. For fixed $U \subset V$, it has been observed by Padberg and Rinaldi [PR90] that

$$\beta(U) := \min\Big\{\beta(U, F) \;\Big|\; F \subset \delta(U), |T \cap U| + |F| \text{ odd}\Big\} \tag{10}$$

Algorithm 2. Blossom minimization

Input:

Graph G, set $T \subset V$ of odd vertices, weight vectors $c, c' \in \mathbb{Q}_+^{|E|}$.

Output:

A minimum blossom.

1: Let $r = \infty$.
2: Compute a cut-tree for G with weights $\min(c, c')$ and terminal vertex set V.
3: **For** each of the $|V| - 1$ edges of the cut-tree
4: Let $\delta(U)$ denote the cut induced by the cut-tree edge.
5: *Check the cut:*
Compute $\beta(U)$ as in (10).
6: If $\beta(U) < r$, then store U along with the arg-min F and set $r := \beta(U)$.
7: **End for**
8: Output the best blossom (U, F).

can be computed in time $\mathcal{O}(|\delta(U)|)$ by first tentatively taking $F := \{e \in \delta(U) \mid c'_e < c_e\}$. Now if $|T \cap U| + |F|$ is odd, then we have found a minimizing F. Otherwise, find $f \in \delta(U)$ minimizing $|c_f - c'_f|$ over $f \in \delta(U)$, because then the symmetric difference of F and $\{f\}$ minimizes $\beta(U, \cdot)$.

This implies that the loop in steps 2–6 in Algorithm 2 runs in time $\mathcal{O}(|V|^2)$, and that the running time of Algorithm 2 is dominated by the computation of the cut-tree in step 1, which amounts to $\mathcal{O}(|V|^2 |E| \log(|V|^2/|E|))$.

The similarity between the Padberg–Rao minimum odd cut Algorithm 1 and our blossom minimization Algorithm 2 is striking. Moreover, in the next section, we give a short and elegant proof of correctness of Algorithm 2, which is similar to a proof of correctness of Algorithm 1 given by Rizzi [Riz02].

We remark that, just like the original Padberg–Rao blossom separation algorithm, our algorithm can be modified so that it outputs every violated blossom inequality found, rather than only the most violated one. We also remark that the function $\beta(U)$ defined in (10), unlike the cut function $w(\delta(U))$, is not, in general, submodular.

4. A simple proof of the correctness of Algorithm 2. The proof makes use of a well-known fact concerned with T-odd cuts and so-called T-joins, which we briefly review. Let $G = (V, E)$ be a graph, and let $T \subset V$ (with $|T|$ even) be a set of odd vertices, as before. A set $F \subset E$ is called a *T-join* if it has the following property: a vertex of G is odd if and only if there is an odd number of edges in F incident on it. The fact that we need is the following: if F is a T-join and U is an arbitrary subset of V, then $|U \cap T|$ is odd if and only if $|\delta(U) \cap F|$ is odd. That is, a cut is T-odd if and only if it meets every T-join an odd number of times.

Consider once again a graph $G = (V, E)$, a weight vector $w \in \mathbb{Q}^{|E|}$, and a set $T \subset V$ of odd vertices. Suppose that we have computed a cut-tree with terminal vertex set V. We say that an edge $x \sim y$ of the cut-tree is *T-odd* if the vertex sets on each shore of the cut defined by $x \sim y$ are T-odd. This defines a T-join in the cut-tree. The next theorem, due to Rizzi [Riz02], is an extension of a key theorem in [PR82]. For the sake of clarity, we repeat the proof of Rizzi.

THEOREM 4.1 (see [Riz02]). *One of the T-odd edges of the cut-tree induces a minimum odd cut in G.*

Proof. Let $U \subset V$ be an odd set such that $\delta(U)$ is a minimum odd cut in G. Since U is an odd set, there exists an odd number of T-odd cut-tree edges leaving U. Let

$x \sim y$ be one of them, and let $\delta(U')$ be the minimum (x,y)-cut that it induces in G (by property (6)). Since $\delta(U)$ is also an (x,y)-cut in G, we have $w(\delta(U')) \leq w(\delta(U))$. Since $x \sim y$ is a T-odd cut-tree edge, $\delta(U')$ is a minimum odd cut in G. ∎

Now we come to the proof of correctness of Algorithm 2.

THEOREM 4.2. *One of the edges of the cut-tree computed in Algorithm* 2 *induces a set* $U \subset V$ *that minimizes* $\beta(\cdot)$.

Proof. Let $E' := \{e \in E \mid c'_e < c_e\}$ and T' denote the set of vertices that are incident on an odd number of edges in E'. This way, E' is a T'-join. Define the set $T^{\triangle}$ as the symmetric difference of T with T'. Furthermore, let $w := \min(c, c')$ denote the weight function used in Algorithm 2. We note the following facts:

(a) A set U is $T^{\triangle}$-odd if and only if $|T \cap U| + |E' \cap \delta(U)|$ is odd.

(b) For all sets U, we have $\beta(U) \geq w(\delta(U))$. If U is $T^{\triangle}$-odd, then the equality holds. If U is not $T^{\triangle}$-odd, then, by Remark 3.2,

$$\beta(U) = w(\delta(U)) + \min_{e \in \delta(U)} |c'_e - c_e|.$$

Item (4) follows from the elementary fact about T-cuts and T-joins mentioned previously by taking a T-join J and noting that the symmetric difference of J and E' is a $T^{\triangle}$-join.

Now let $U \subset V$ be a set that minimizes $\beta(\cdot)$. We distinguish two cases.

Case 1: U is $T^{\triangle}$-odd. The proof of Theorem 4.1, applied to $T^{\triangle}$ and w, shows that there exists a $T^{\triangle}$-odd edge of the cut-tree which, by (4), determines a set $U' \subset V$ minimizing $\beta(\cdot)$.

Case 2: U is not $T^{\triangle}$-odd. Let $f = \{i, j\} \in \delta(U)$ have the minimal value of $|c_f - c'_f|$ among all edges in $\delta(U)$. On the path from i to j in the cut-tree, at least one edge $x \sim y$ has one end in U and the other not in U. Let $\delta(U')$ be the minimum (x,y)-cut in G corresponding to this edge. Notice that $f \in \delta(U')$. We then have

$$\beta(U) = w(\delta(U)) + |c_f - c'_f| \geq w(\delta(U')) + |c_f - c'_f| \geq \beta(U').$$

The first inequality holds because $\delta(U)$ is an (x,y)-cut in G. The second follows from the definition of the function $\beta(\cdot)$ and the fact that $f \in \delta(U')$. ∎

REFERENCES

[BB98] J. M. BELENGUER AND E. BENAVENT, *The capacitated arc routing problem: Valid inequalities and facets*, Comput. Optim. Appl., 10 (1998), pp. 165–187.

[CF96] A. CAPRARA AND M. FISCHETTI, $\{0, \frac{1}{2}\}$-*Chvátal-Gomory cuts*, Math. Programming, 74 (1996), pp. 221–235.

[Edm65] J. EDMONDS, *Maximum matching and a polyhedron with* 0-1 *vertices*, J. Res. Nat. Bur. Standards Sect. B, 69B (1965), pp. 125–130.

[GL00] G. GHIANI AND G. LAPORTE, *A branch-and-cut algorithm for the undirected rural postman problem*, Math. Program., 87 (2000), pp. 467–481.

[GH61] R. E. GOMORY AND T. C. HU, *Multi-terminal network flows*, J. Soc. Indust. Appl. Math., 9 (1961), pp. 551–570.

[GH87] M. GRÖTSCHEL AND O. HOLLAND, *A cutting plane algorithm for minimum perfect* 2*-matching*, Computing, 39 (1987), pp. 327–344.

[GLS88] M. GRÖTSCHEL, L. LOVÁSZ, AND A. J. SCHRIJVER, *Geometric Algorithms and Combinatorial Optimization*, Springer-Verlag, Berlin, 1988.

[GMS92] M. GRÖTSCHEL, C. L. MONMA, AND M. STOER, *Computational results with a cutting plane algorithm for designing communication networks with low-connectivity constraints*, Oper. Res., 40 (1992), pp. 309–330.

[GP79a] M. Grötschel and M. W. Padberg, *On the symmetric travelling salesman problem* I: *Inequalities*, Math. Programming, 16 (1979), pp. 265–280.

[GP79b] M. Grötschel and M. W. Padberg, *On the symmetric travelling salesman problem* II: *Lifting theorems and facets*, Math. Programming, 16 (1979), pp. 281–302.

[GT88] A. V. Goldberg and R. E. Tarjan, *A new approach to the maximum flow problem*, J. Assoc. Comput. Math., 35 (1988), pp. 921–940.

[LRT04] A. N. Letchford, G. Reinelt, and D. O. Theis, *A faster exact separation algorithm for blossom inequalities*, in Integer Programming and Combinatorial Optimization 10, D. Bienstock and G. Nemhauser, eds., Lecture Notes in Comput. Sci. 3064, Springer-Verlag, Berlin, 2004, pp. 196–205.

[LLE04] J. Lysgaard, A. N. Letchford, and R. W. Eglese, *A new branch-and-cut algorithm for the capacitated vehicle routing problem*, Math. Programming, 100 (2004), pp. 423–445.

[Mah94] A. R. Mahjoub, *Two-edge connected spanning subgraphs and polyhedra*, Math. Programming, 64 (1994), pp. 199–208.

[PR82] M. W. Padberg and M. R. Rao, *Odd minimum cut-sets and b-matchings*, Math. Oper. Res., 7 (1982), pp. 67–80.

[PR90] M. Padberg and G. Rinaldi, *Facet identification for the symmetric traveling salesman polytope*, Math. Program., 47 (1990), pp. 219–257.

[Pul73] W. R. Pulleyblank, *Faces of Matching Polyhedra*, Ph.D. thesis, University of Waterloo, Waterloo, Ontario, 1973.

[Riz02] R. Rizzi, *Minimum T-cuts and optimal T-pairings*, Discrete Math., 257 (2002), pp. 177–181.