Approximating Minimum Multicuts by Evolutionary Multi-Objective Algorithms*

Frank Neumann[1] and Joachim Reichel[2]

[1] Max-Planck-Institut für Informatik, Saarbrücken, Germany,
firstname.lastname@mpi-inf.mpg.de
[2] Institut für Mathematik, TU Berlin, Germany,
reichel@math.tu-berlin.de

Abstract. It has been shown that simple evolutionary algorithms are able to solve the minimum cut problem in expected polynomial time when using a multi-objective model of the problem. In this paper, we generalize these ideas to the NP-hard minimum multicut problem. Given a set of k terminal pairs, we prove that evolutionary algorithms in combination with a multi-objective model of the problem are able to obtain a k-approximation for this problem in expected polynomial time.

1 Introduction

Evolutionary algorithms and other kinds of metaheuristics have become very popular for solving combinatorial optimization problems. In recent years, a lot of progress has been made in understanding this kind of algorithms with respect to their runtime behavior. Most of these results are on classical polynomially solvable problems such as minimum spanning trees [19] or shortest paths [9, 20]. One goal of such studies on easy problems is to get an understanding how the heuristics work in order to analyze difficult problems in the future. Later on, such studies have served as a basis for analyzing evolutionary algorithms on NP-hard problems [10, 16, 22].

Recently, it has been shown in [17] that the minimum cut problem cannot be solved by simple single-objective evolutionary algorithms. In contrast to this a multi-objective approach has been presented which provably solves the problem in expected polynomial time. The algorithms analyzed in this paper use the dual problem, i. e., the maximum flow problem, as a subroutine. The goal of this study was to gain new insights into the behavior of evolutionary algorithms when considering cutting problems and to express the usefulness of the original problem and its dualization with respect to the optimization by evolutionary algorithms. The study carried out in the present paper extends the mentioned results to the NP-hard minimum multicut problem. In this problem a set of k pairs of nodes (s_i, t_i), $1 \leq i \leq k$ is given and the goal is to find a cut of minimum cost such that all $(s_i,\, t_i)$ pairs are separated. This problem has been shown to

* This work was supported by the Deutsche Forschungsgemeinschaft (DFG) as part of the Collaborative Research Center “Computational Intelligence” (SFB 531).

be MAX SNP-hard [3, 7, 8, 21]. As a consequence, there is no polynomial time approximation scheme (unless P = NP) [2].

Due to the results obtained in [17], we consider multi-objective models for the multicut problem using flow computations which can be carried out in polynomial time as a subroutine. It is our aim to examine how such an approach can approximate an optimal solution for this problem. We study an evolutionary algorithm called Global SEMO (GSEMO) which has been widely used for the runtime analysis of evolutionary multi-objective algorithms (see e.g. [4, 10, 17, 18]). Our analysis points out that this algorithm achieves a factor k-approximation in polynomial time as long as the weights on the edges of the given graph are polynomially bounded in the size of the input. The requirement on the edge weights is necessary since the population size of GSEMO may become as large as a polynomial in the largest edge weight [13]. One way to deal with this circumstance is to incorporate the concept of ε-dominance [14] into the algorithm. Using this mechanism in a similar way as done in [13, 17], we prove that a k-approximation can be achieved in expected polynomial runtime even if the weights of the given graph are not polynomially bounded.

The paper is organized as follows. In Section 2, we present the model of the multicut problem and the algorithms that are analyzed in this paper. The results for GSEMO are presented in Section 3. In Section 4 we improve these results by incorporating the ε-dominance approach into the algorithm. Finally, we give some concluding remarks.

2 Problem Definition

We consider the following problem. Given a connected directed or undirected graph G = (V, E) on n vertices and m edges and a cost function $c : E \mapsto \mathbb{N}_+$ that imposes positive integer weights on the edges. Let $\{(s_1, t_1), \ldots, (s_k, t_k)\}$ be a set of k pairs with $s_i \neq t_i, 1 \leq i \leq k$. The source of commodity i is given by s_i, the target by t_i. We denote by $c_{\max} = \max_{e \in E} c(e)$ the largest cost among all edges.

A multicut $S \subseteq E$ is a set of edges such that there is no path from s_i to t_i in $(V, E \setminus S)$ for any commodity i. The cost of a subset of E is defined as the sum of the costs of its elements. The goal is to find a multicut $S \subseteq E$ of minimum cost. For $k = 1$, we obtain the minimum s-t-cut problem as a special case.

The dual of this problem is the maximum-value multicommodity flow problem. This problem asks for an s_i-t_i-flow for each commodity i such that the sum of all flow values is maximum. The flow for each commodity i has to satisfy the flow conservation constraints at every node except s_i and t_i, and the sum of all k flows has to obey the capacities given by the cost function c.

Let F_i denote the value of a maximum s_i-t_i-flow in G and define $F := \sum_i F_i$. Let F^* denote the sum of all flow values of a maximum multicommodity flow in G and let C^* denote the cost of a minimum multicut of G. Note that $F^* \leq C^* \leq C := m \cdot c_{\max}$. Furthermore we have $F^* \leq F = \sum_i F_i \leq k \cdot F^* \leq k \cdot C^* \leq k \cdot C$.

For the undirected case it has been shown [11] that $C^* \in O(\log(k) \cdot F^*)$. The proof is constructive and leads to an $O(\log k)$-approximation algorithm for the minimum multicut problem. This bound is tight, i. e., there are graph classes for which $C^* \in \Omega(\log(k) \cdot F^*)$ holds. In the directed case, the gap between F^* and C^* can be as large as $\tilde{\Omega}(n^{1/7})$ [6] and is at most $O(\sqrt{n \log(k+1)})$ [5]. In the special case $k = 1$ we have $F^* = C^*$ by the max-flow-min-cut theorem [1].

Based on the results in [17], we consider an edge-based approach. We work with bit strings of length $m = |E|$. For a search point $x \in \{0,1\}^m$, the set $E(x) := \{e_i \in E \mid x_i = 1\}$ denotes the subset of E corresponding to the 1's in x. Note, that not every search point represents a multicut, i. e., not every search point is a feasible solution.

Due to the results for the special case of the minimum s-t-cut problem [17], we do not consider single-objective evolutionary algorithms at all. Instead we focus on multi-objective evolutionary algorithms. Examples for simple multi-objective evolutionary algorithms that have been analyzed before are SEMO and GSEMO [12, 15, 18]. The GSEMO algorithm can be described as follows. Note that the fitness function f is vector-valued and the $\leq$-comparison is to be understood component-wise.

Algorithm 1 *GSEMO (Global Simple Evolutionary Multi-objective Optimizer)*

```
1. Choose $x \in \{0,1\}^m$ uniformly at random.
2. Determine $f(x)$ and initialize $P := \{x\}$.
3. Repeat
   – choose $x \in P$ uniformly at random.
   – create an offspring $y$ by flipping each bit of $x$ independently with probability $1/m$.
   – let $P$ unchanged, if there is an $z \in P$ such that $f(z) \leq f(y)$ and $f(z) \neq f(y)$.
   – otherwise, exclude all $z$ with $f(y) \leq f(z)$ and add $y$ to $P$.
```

We consider the fitness function $f : \{0,1\}^m \mapsto \mathbb{N}^2$, $f(x) = (cost(x), flow)$, where $cost(x) = \sum_{e \in E(x)} c(e)$, $flow(x) := \sum_i flow_i(x)$ and $flow_i(x)$ denotes the value of a maximum s_i-t_i-flow in $G(x) := (V, E \setminus E(x))$.

Note that the values of both components $cost(\cdot)$ and $flow(\cdot)$ of the fitness function can be exponential in the input size, which implies that GSEMO has to cope with a Pareto front of exponential size.

Therefore, we also investigate a variation of GSEMO which uses the concept of ε-dominance [14]. It has already been shown in [13, 17] that such an approach may be provably helpful when dealing with exponentially large Pareto fronts. We consider the DEMO algorithm (Diversity Evolutionary Multi-objective Optimizer) which differs from GSEMO by using a function b that assigns the same function value to search points with similar objective vectors. During the run of the algorithm at most one search point for any fixed function value of b is present in the population.

We examine DEMO partitioning the objective space into boxes by using the function $b : \{0,1\}^m \mapsto \mathbb{N}^2$ with $b_1(x) := \left\lfloor \frac{\log(1+cost(x))}{\log(1+\varepsilon)} \right\rfloor$ and $b_2(x) :=$

$\left\lfloor \frac{\log(1+flow(x))}{\log(1+\varepsilon)} \right\rfloor$, where $\varepsilon > 0$ is a parameter that determines the size of the boxes. The algorithm has the following description.

Algorithm 2 *DEMO (Diversity Evolutionary Multi-objective Optimizer)*

```
1. Choose $x \in \{0,1\}^m$ uniformly at random.
2. Determine $f(x)$ and initialize $P := \{x\}$.
3. Repeat
   – choose $x \in P$ uniformly at random.
   – create an offspring $y$ by flipping each bit of $x$ independently with probability $1/m$.
   – let $P$ unchanged, if there is an $z \in P$ such that $b(z) \leq b(y)$ and ($b(z) \neq b(y)$ or $cost(z) + flow(z) < cost(y) + flow(y)$).
   – otherwise, exclude all $z$ with $b(y) \leq b(z)$ and add $y$ to $P$.
```

The DEMO algorithm discards a new search point y if the corresponding box $b(y)$ is dominated by the box $b(z)$ of some search point $z \in P$ (and y and z do not fall into the same box). If $b(y) = b(z)$, the algorithm discards y if its sum of cost and flow value is larger than that of z. Otherwise, all search points in dominated boxes are removed from the population and y is included into the population.

Due to Laumanns et al. [14], the following upper bound on the population size can be given.

Lemma 1. *The population size $|P|$ of DEMO is upper bounded by*

$$B := \frac{\log(1+C)}{\log(1+\varepsilon)} = O(\varepsilon^{-1} \log C) = O(\varepsilon^{-1}(\log n + \log c_{\max})).$$

The algorithms described in this section do not use any stopping criteria. For theoretical investigations it is common to consider the algorithms as infinite stochastic processes and to use the number of fitness evaluations as a measure of the runtime. Our goal is to bound the expected number of fitness evaluations (also called expected runtime) until the algorithms have obtained a good approximation for the multicut problem.

We point out that we use the sum of single-commodity flow values instead of the value of a multi-commodity flow as second component of the fitness function. While the multi-commodity flow value would probably lead to a stronger approximation bound, the computation of a multi-commodity flow requires linear programming as there is no combinatorial algorithm known. On the other hand, single-commodity flows can be efficiently computed using different well-studied algorithms [1]. Furthermore, an oracle for the multi-commodity flow value is a rather strong oracle as it provides the value of the dual problem.

3 Analysis of GSEMO

The goal of this section is to prove a pseudopolynomial upper bound on the runtime of GSEMO until it has achieved an F/C^*-approximation for the multicut

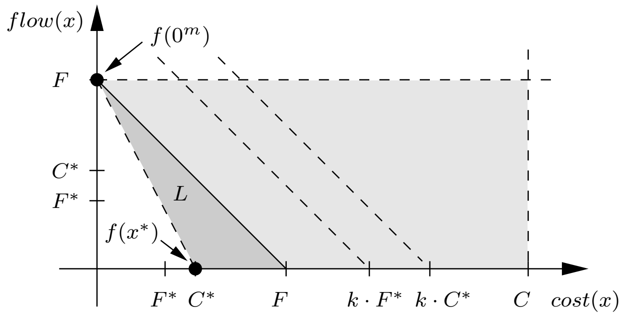

Fig. 1. Objective space of the fitness function $f(x) = (cost(x), flow(x))$. The sketch depicts the case that the sequence F^*, C^*, F, $k \cdot F^*$, $k \cdot C^*$ is strictly increasing. Note that subsequent values may coincide and that C can be as small as C^*. Optimal multicuts x^* have objective vector $(C^*, 0)$, k-approximations lie on the the segment from $(C^*, 0)$ to $(\min\{k \cdot C^*, C\}, 0)$.

problem. Note that $F/C^* \leq k$, hence in the worst case we get a k-approximation. We denote by $L = \{x \in \{0,1\}^m \mid cost(x) + flow(x) \leq F\}$ the set of search points whose objective vectors lie on or below the line given by the two objective values $(0, F)$ and $(F, 0)$. Figure 1 shows a graphical representation of the objective space. The following proposition shows that the search points of L represent subsets of F/C^*-approximations of minimum multicuts.

Proposition 1. *Let $x \in L$. Then $E(x)$ is a subset of an F/C^*-approximation of a minimum multicut of G.*

Proof. Since $x \in L$ we have $cost(x) + flow(x) \leq F$. Let S denote a minimum multicut of $G(x)$. Then $E(x) \mathbin{\dot\cup} S$ is a multicut of G with $cost(E(x) \mathbin{\dot\cup} S) = cost(x) + cost(S)$. Since S is a minimum multicut of $G(x)$, its cost is not larger than the sum of the cost of the individual minimum s_i-t_i-cuts, i. e., $cost(S) \leq flow(x)$. Hence, we have $cost(E(x) \mathbin{\dot\cup} S) \leq cost(x) + flow(x) \leq F \leq k \cdot F^* \leq k \cdot C^*$, which implies that $E(x) \mathbin{\dot\cup} S$ is an F/C^*-approximation of a minimum multicut of G. □

The preceding proposition implies the following condition for F/C^*-approximate solutions which will be essential for the analysis of the algorithms.

Corollary 1. *Let $x \in \{0,1\}^m$ such that $flow(x) = 0$. Then $E(x)$ is an F/C^*-approximation of a minimum multicut of G if and only if $x \in L$.*

We remark that the converse of Proposition 1 is not true in general, in contrast to the single-commodity case $k = 1$.

For $x \in \{0,1\}^m$ and $e \in E$ define $x^{+e} \in \{0,1\}^m$ by $x^{+e}(e) = 1$ and $x^{+e}(e') = x(e')$ for $e' \neq e$. We can bound $flow(x^{+e})$ in terms of $flow(x)$ as follows.

Proposition 2. *Let $x \in \{0,1\}^m$ and $e \in E$. Then $flow(x^{+e}) \geq flow(x) - kc(e)$.*

Proof. By the (single-commodity) max-flow min-cut theorem we have $flow_i(x^{+e}) \geq flow_i(x) - c(e)$ for each commodity i. Summation over i yields the claimed result. □

Proposition 3. *Let $x \in \{0,1\}^m$ such that $flow_i(x) > 0$ for some commodity i. Let $e \in E \setminus E(x)$ an edge of a minimum s_i-t_i-cut of $G(x)$. Then $flow(x^{+e}) \leq flow(x) - c(e)$ and $cost(x^{+e}) + flow(x^{+e}) \leq cost(x) + flow(x)$.*

Proof. Since $flow_i(x) > 0$ the minimum s_i-t_i-cut of $G(x)$ is not the empty set. Let $x \in E \setminus E(x)$ an edge from such a minimum s_i-t_i-cut. By the (single-commodity) max-flow min-cut theorem we have $flow_i(x^{+e}) = flow_i(x) - c(e)$. Furthermore, $flow_j(x^{+e}) \leq flow_j(x)$ holds for $j \neq i$. Summation over i yields the first claim.

Since $cost(x^{+e}) = cost(x) + c(e)$, the second claim follows directly from the first one. □

The following corollary is an immediate consequence of the preceding proposition and the definition of L.

Corollary 2. *Let $x \in L$ a search point such that $flow(x) > 0$. Then there exists a 1-bit flip leading to a search point $x' \in L$ with $flow(x') < flow(x)$.*

Now we are able to prove the following theorem which shows that the expected runtime of GSEMO is pseudopolynomial with respect to the given input.

Theorem 1. *The expected time until GSEMO working on the fitness function f constructs an F/C^*-approximation of a minimum k-commodity multicut is $O(Fm(\log n + \log c_{\max}))$.*

Proof. The size of the population P is at most F as GSEMO keeps at each time at most one solution per fixed flow value. First we consider the time until $0^m \in L$ has been included into the population. Note that $cost(0^m) = 0$. Afterwards we study the time until $x \in L$ with $flow(x) = 0$ has been included. By Corollary 1 the edge set $E(x)$ is an F/C^*-approximation of a minimum multicut.

The expected time until GSEMO working on the fitness function f constructs 0^m is $O(Fm(\log n + \log c_{\max}))$. This can be proved using the technique of the expected multiplicative cost decrease with respect to $\min_{x \in P} cost(x)$. The proof is analogue to the single-commodity case $k = 1$ (see proof of Theorem 3 in [17]).

Now we bound the time until a minimum cut has been constructed. Once again we apply the method of the expected multiplicative cost decrease, now with respect to the flow value. Let x be the solution with the smallest flow value in $P \cap L$. Note that $\min_{x \in P \cap L} flow(x)$ does not increase during a run of GSEMO.

Consider a mutation step that selects x and performs an arbitrary 1-bit flip. Such a step is called a *good* step. The probability of a good step is lower bounded by $\Omega(1/F)$. By Proposition 1, $E(x)$ is a subset of an F/C^*-approximation of a minimum multicut, which can be obtained by including the remaining edges one by one. Therefore, a randomly chosen 1-bit flip decreases the minimum flow value in $P \cap L$ on average by a factor of at least $1 - 1/m$.

Hence, after N good steps, the expected minimum flow value is bounded from above by $(1-1/m)^N \cdot \mathit{flow}(x)$. Since $\mathit{flow}(x) \leq F \leq k \cdot C$, we obtain the upper bound $(1-1/m)^N \cdot k \cdot C$. Using the method of the multiplicative cost decrease the expected time until $x' \in L$ with $\mathit{flow}(x') = 0$ has been discovered is $O(Fm(\log n + \log c_{\max} + \log k))$. By Corollary 1, x' is an F/C^*-approximation of a minimum multicut. □

4 Analysis of DEMO

The upper bound given in Theorem 1 is polynomial as long as the weights are polynomially bounded with respect to the input size. For larger, i. e., exponential, weights the population size may become too large to obtain an F/C^*-approximation in expected polynomial time. To deal with this issue, we consider DEMO with an appropriate choice of ε such that the population size is always polynomially bounded with respect to the size of the given input.

To obtain the upper bound on the runtime of DEMO, we first consider the time until the search point 0^m has been included into the population and analyze the time to achieve an F/C^*-approximation afterwards.

Proposition 4. *Let $\varepsilon \leq 1/m$ and $x \in \{0,1\}^m$ a search point such that $\mathit{cost}(x) > 0$. Then there exists a 1-bit flip leading to a search point $x' \in \{0,1\}^m$ with $b_1(x') < b_1(x)$.*

Proof. Consider all 1-bit flips that remove a single edge from $E(x)$. Among all resulting search points, consider a point x' that minimizes $y' := \mathit{cost}(x')$. Let $y := \mathit{cost}(x)$.

The repeated removal of edges in $E(x)$ yields the search point 0^m. Let $\ell := |E(x)| \leq m$. Since y' was minimal, $y' \leq (1-\frac{1}{\ell})y$ holds. Since $\varepsilon \leq \frac{1}{m} \leq \frac{1}{\ell}$ and $\ell \leq y$, we have

$$\begin{aligned}(1+\varepsilon)(1+y') &\leq 1+\varepsilon+(1+\varepsilon)\left(1-\frac{1}{\ell}\right)y \\ &\leq 1+\frac{y}{\ell^2}+\left(1+\frac{1}{\ell}\right)\left(1-\frac{1}{\ell}\right)y = 1+y\,.\end{aligned}$$

This implies

$$1+\frac{\log(1+y')}{\log(1+\varepsilon)} \leq \frac{\log(1+y)}{\log(1+\varepsilon)}\ ,$$

and finally $b_1(x') < b_1(x)$. □

In the following, we bound the expected time until DEMO has produced the search point 0^m. Later on, we will show how the algorithm can proceed to obtain an F/C^*-approximation.

Lemma 2. *The expected time until DEMO working on the fitness function f includes the search point 0^m into the population is $O(m\varepsilon^{-2}(\log^2 n + \log^2 c_{\max}))$.*

Proof. The archiving strategy of DEMO guarantees that whenever a non-empty box becomes empty, another search point whose box dominates the considered box is included into the population. Therefore, $\min_{x \in P} b_1(x)$ will never increase during the run of the algorithm.

Since the population size is bounded by B, the probability of picking a search point $x \in P$ with minimal b_1-value is $\Omega(1/B)$. By Proposition 4, there exists at least one 1-bit flip leading to a search point x' with $b_1(x') < b_1(x)$. The probability to generate such a search point x' is $\Omega(1/m)$. After at most B such steps, the b_1-value is zero implying that we have found the search point 0^m. Hence, the expected time to include 0^m into the population is

$$O(B^2 m) = O(m\varepsilon^{-2} \log^2 C) = O(m\varepsilon^{-2}(\log^2 n + \log^2 c_{\max}))\,.$$

This concludes the proof. □

To come up with an upper bound for DEMO, it is necessary to examine how the algorithm may progress from a solution $x \in L$ to a solution of $x' \in L$ with $b_2(x') < b_2(x)$. The following proposition points out that this is possible by carrying out a special 1-bit flip.

Proposition 5. *Let $\varepsilon \le 1/m$ and $x \in L$ a search point such that $flow(x) > 0$. Then there exists a 1-bit flip leading to a search point $x' \in L$ with $b_2(x') < b_2(x)$.*

Proof. By Corollary 2, there exists at least one 1-bit flip leading to a search point $x' \in L$ with $flow(x') < flow(x)$. Among all such search points, consider a point x' that minimizes $y' := flow(x')$. Let $y := flow(x)$.

The repeated application of Corollary 2 yields an F/C^*-approximation $E(x^*)$ of a minimum multicut of G. Let $\ell := |E(x^*)| - |E(x)| \le m$. Since y' was minimal, $y' \le (1 - \frac{1}{\ell})y$ holds. Since $\varepsilon \le \frac{1}{m} \le \frac{1}{\ell}$ and $\ell \le y$, we have $b_2(x') < b_2(x)$ by the same calculation as in the proof of Proposition 4. □

Finally, we are able to prove the following theorem which shows that the expected runtime of DEMO with an appropriate choice of ε is always polynomially bounded with respect to the given input.

Theorem 2. *Choosing $\varepsilon \le 1/m$, the expected time until DEMO working on the fitness function f constructs an F/C^*-approximation of a minimum multicut is $O(m\varepsilon^{-2}(\log^2 n + \log^2 c_{\max}))$.*

Proof. Due to Lemma 2 the search point $0^m \in L$ has been included into the population after an expected number of $O(m\varepsilon^{-2}(\log^2 n + \log^2 c_{\max}))$ steps. Hence, it is sufficient to consider the search process after having found a search point $x \in L$.

The archiving strategy of DEMO guarantees that whenever a non-empty box becomes empty, another search point whose box dominates the considered box is included into the population. Moreover, the tie-break rule ensures that a non-empty box with a search point $x \in P \cap L$ will never exchange that search point for a search point $x' \notin L$. Therefore, $\min_{x \in P \cap L} b_2(x)$ will never increase during the run of the algorithm.

Since the population size is bounded by B, the probability of picking a search point $x \in L$ with minimal b_2-value among the search points in L is $\Omega(1/B)$. By Proposition 5, there exists at least one 1-bit flip leading to a search point $x' \in L$ with $b_2(x') < b_2(x)$. The probability to generate such a search point x' is $\Omega(1/m)$. After at most B such steps, the b_2-value is zero implying that we have found a multicut. Since $x' \in L$, this multicut is an F/C^*-approximation of a minimum cut. Hence, the expected time to obtain an F/C^*-approximation of a minimum multicut is

$$O(B^2 m) = O(m\varepsilon^{-2} \log^2 C) = O(m\varepsilon^{-2}(\log^2 n + \log^2 c_{\max}))\,.$$

This concludes the proof. $\square$

5 Conclusions

The multicut problem is an NP-hard generalization of the minimum cut problem. We have shown how the correlation between flows and cuts can be used to come up with efficient evolutionary algorithms for the approximation of the minimum multicut problem. Our multi-objective approach using flow computations and the concept of ε-dominance is able to achieve a k-approximation in expected polynomial time. Further studies will consider how the theoretical results obtained in this paper can be used to come up with good evolutionary algorithms for the multicut problem by using our model in well-known evolutionary multi-objective algorithms.

Acknowledgements

The authors thank Martin Skutella for helpful discussions on the topic of this paper.

References

1. R. K. Ahuja, T. L. Magnati, and J. B. Orlin. *Network flows: theory, algorithms, and applications*. Prentice Hall, 1993.
2. S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and the hardness of approximation problems. *Journal of the ACM*, 45(3):501–555, 1998.
3. C. Bentz, M.-C. Costa, L. Létocard, and F. Roupin. Erratum to 'M.-C. Costa, L. Létocard and F. Roupin: Minimal multicut and maximal integer maxiflow: A survey" [European Journal of Operational Research 162(1) (2005) 55–69]. *European Journal of Operational Research*, 177(2):1312, 2007.
4. D. Brockhoff, T. Friedrich, N. Hebbinghaus, C. Klein, F. Neumann, and E. Zitzler. Do additional objectives make a problem harder? In *Proc. of the 9th Genetic and Evolutionary Comp. Conference (GECCO '07)*, pages 765–772. ACM Press, 2007.

5. J. Cheriyan, H. Karloff, and Y. Rabani. Approximating directed multicuts. In *Proc. of the 42nd IEEE Symposium on Foundations of Computer Science (FOCS '01)*, pages 320–328, 2001.
6. J. Chuzhoy and S. Khanna. Polynomial flow-cut gaps and hardness of directed cut problems. In *Proc. of the 39th Annual ACM Symposium on Theory of Computing (STOC '07)*, pages 179–188, 2007.
7. M.-C. Costa, L. Létocard, and F. Roupin. Minimal multicut and maximal integer maxiflow: A survey. *European Journal of Operational Research*, 162(1):55–69, 2005.
8. E. Dahlhaus, D. S. Johnson, C. H. Papadimitriou, P. D. Seymour, and M. Yannakakis. The complexity of multiterminal cuts. *SIAM Journal of Computing*, 23:864–894, 1994.
9. B. Doerr, E. Happ, and C. Klein. Crossover is provably useful in evolutionary computation. In *Proc. of the 10th Genetic and Evolutionary Computation Conference (GECCO '08)*. ACM Press, 2008. to appear.
10. T. Friedrich, J. He, N. Hebbinghaus, F. Neumann, and C. Witt. Approximating covering problems by randomized search heuristics using multi-objective models. In *Proc. of the 9th Genetic and Evolutionary Computation Conference (GECCO '07)*, pages 797–804, 2007.
11. N. Garg, V. V. Vazirani, and M. Yannakakis. Approximate max-flow min-(multi)cut theorems and their applications. In *Proc. of the 25th Annual ACM Symposium on Theory of Computing (STOC '93)*, pages 698–707, 1993.
12. O. Giel. Expected runtimes of a simple multi-objective evolutionary algorithm. In *Proc. of the IEEE Congress on Evolutionary Computation (CEC '03)*, pages 1918–1925, 2003.
13. C. Horoba and F. Neumann. Benefits and drawbacks for the use of ε-dominance in evolutionary multi-objective optimization. In *Proc. of the 10th Genetic and Evolutionary Computation Conference (GECCO '08)*. ACM Press, 2008. to appear.
14. M. Laumanns, L. Thiele, K. Deb, and E. Zitzler. Combining convergence and diversity in evolutionary multi-objective optimization. *Evolutionary Computation*, 10(3):263–282, 2002.
15. M. Laumanns, L. Thiele, and E. Zitzler. Running time analysis of multiobjective evolutionary algorithms on pseudo-boolean functions. *IEEE Transactions on Evolutionary Computation*, 8(2):170–182, April 2004.
16. F. Neumann. Expected runtimes of a simple evolutionary algorithm for the multi-objective minimum spanning tree problem. *European Journal of Operational Research*, 181(3):1620–1629, 2007.
17. F. Neumann, J. Reichel, and M. Skutella. Computing minimum cuts by randomized search heuristics. In *Proc. of the 10th Genetic and Evolutionary Computation Conference (GECCO '08)*, 2008. (to appear, available as Technical Report CI-242/08, Collaborative Research Center 531, Technical University of Dortmund).
18. F. Neumann and I. Wegener. Minimum spanning trees made easier via multi-objective optimization. *Natural Computing*, 5(3):305–319, 2006.
19. F. Neumann and I. Wegener. Randomized local search, evolutionary algorithms and the minimum spanning tree problem. *Theor. Comp. Sci.*, 378(1):32–40, 2007.
20. J. Scharnow, K. Tinnefeld, and I. Wegener. The analysis of evolutionary algorithms on sorting and shortest paths problems. *Journal of Mathematical Modelling and Algorithms*, pages 349–366, 2004.
21. A. Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency*. Springer, Berlin, 2003.
22. C. Witt. Worst-case and average-case approximations by simple randomized search heuristics. In *Proc. of STACS 2005*, volume 3404 of *LNCS*, pages 44–56, 2005.