More Effective Crossover Operators for the All-Pairs Shortest Path Problem

Benjamin Doerr[1], Daniel Johannsen[1], Timo Kötzing[1], Frank Neumann[1], and Madeleine Theile[2]

[1] Max-Planck-Institut für Informatik, Campus E 1 4, 66123 Saarbrücken, Germany
[2] Technische Universität Berlin, Straße des 17. Juni 136, 10623 Berlin, Germany

Abstract. The all-pairs shortest path problem is the first non-artificial problem for which it was shown that adding crossover can significantly speed up a mutation-only evolutionary algorithm. Recently, the analysis of this algorithm was refined and it was shown to have an expected optimization time of $\Theta(n^{3.25}(\log n)^{0.25})$.

In this work, we study two variants of the algorithm. These are based on two central concepts in recombination, *repair mechanisms* and *parent selection*. We show that repairing infeasible offspring leads to an improved expected optimization time of $O(n^{3.2}(\log n)^{0.2})$. Furthermore, we prove that choosing parents that guarantee feasible offspring results in an optimization time of $O(n^3 \log n)$.

1 Introduction

One of the important issues when designing successful evolutionary algorithms is to choose a suitable representation of possible solutions together with good variation operators. Different representations and variation operators have been discussed for a wide range of combinatorial optimization problems (see e. g. [18,11,19]). Often, variation operators (such as crossover or mutation) are designed to produce feasible offsprings. For mutation this is easy to achieve, as a mutation operator usually only applies a small number of local changes to a given feasible solution.

However, the design of crossover operators, producing from two feasible solutions a new feasible one, is usually more complicated (see e. g. [14] for different crossover operators for the traveling salesman problem). Whenever a crossover operator produces an infeasible solution, one option is to discard it. However, this typically does not lead to efficient methods, as time is wasted on producing infeasible solutions and evaluating them. To deal with this situation, one can use repair mechanisms, which produce from an infeasible solution a feasible one based on properties of both parents [22]. Another way of dealing with the problem of infeasible solutions is to use specific selection methods and/or more problem specific crossover operators that are likely to produce promising solutions [2,15].

R. Schaefer et al. (Eds.): PPSN XI, Part I, LNCS 6238, pp. 184–193, 2010.

The goal of this paper is to point out the effect of repair mechanisms and parent selection for crossover on the runtime of evolutionary algorithms in combinatorial optimization. Analyzing the runtime behavior of evolutionary algorithms has become a major part in their theoretical analysis. Based on results for different kinds of pseudo-Boolean functions [6,10], results have been obtained for different kinds of combinatorial optimization problems. Starting with some results for classical combinatorial optimization problems that are solvable in polynomial time such as the computation of minimum spanning trees [17] or maximum matchings [8], different results have been obtained for NP-hard problems [16,7,12,23]. One cannot expect to beat the best known algorithms if the problem under consideration can be solved in polynomial time. With such studies we want to gain new insights on how evolutionary algorithms behave on natural optimization problems and give insights into the important modules that make such algorithms successful.

We carry out theoretical studies on evolutionary algorithms for the computation of shortest paths. Computing shortest paths is one of the basic problems in computer science and has already been considered in various theoretical studies of evolutionary algorithms. There are different results for the single-source shortest path (SSSP) problem [1,21,3].

We investigate the all-pairs shortest path (APSP) problem which is a generalization of the SSSP problem. Given a strongly connected directed graph $G = (V, E)$ with $|V| = n$ and $|E| = m$ and a weight function $w : E \to \mathbb{R}$ that assigns weights to the edges. We distinguish between the *weight* of a path (the sum of the weight of all its edges) and its *length* (the *number* of edges in the path). The task is to compute from each vertex $v \in V$ a weight-shortest path to every other vertex $u \in V \setminus \{v\}$. Throughout this paper, we assume that G does not contain cycles of negative weight. The APSP problem can be solved by the Floyd-Warshall algorithm; using appropriate data structures, APSP can be computed in time $O(nm + n^2 \log n)$ (see, e. g. [13]). Our aim is to study how general purpose algorithms can deal with the APSP problem. In particular, we want to examine the usefulness of crossover operators in evolutionary computation.

We take the APSP problem as a prominent example to show in a rigorous way how different crossover operators influence the runtime of evolutionary algorithms. Recently, it has been shown that the use of crossover operators provably leads to better evolutionary algorithms than evolutionary algorithms that are just based on mutation [4,5]: The runtime for the mutation-and-crossover approach is $\Theta(n^{3.25}(\log n)^{0.25})$, which is better than the expected optimization time of $\Theta(n^4)$ of the algorithm just using mutation. In addition, [9] studied the runtime behavior of ant colony optimization for this problem and proved an upper bound of $O(n^3(\log n)^3)$. However, we will see that the evolutionary approach examined in this paper solves the APSP problem in expected optimization time $O(n^3 \log n)$.

In the next section, Section 2, we introduce the algorithms that are subject to our analyses. In Section 3, we show how repair mechanisms can speed up the

```
1  P = {P_{u,v} = (u,v) | (u,v) ∈ E};
2  while true do
3      Choose r ∈ [0,1] uniformly at random;
4      if r ≤ p_c then
5          choose two individuals P_{x,y} and P_{x',y'} from P u. a. r.; perform crossover
           on P_{x,y} and P_{x',y'} to obtain an individual P'_{s,t} ;
6      else
7          choose one individual P_{x,y} uniformly at random from P and mutate
           P_{x,y} to obtain an individual P'_{s,t};
8      if P'_{s,t} is a path from s to t then
9          if there is no individual P_{s,t} ∈ P then P = P ∪ {P'_{s,t}};
10         else if w(P'_{s,t}) ≤ w(P_{s,t}) then P = (P ∪ {P'_{s,t}}) \ {P_{s,t}};
```

Algorithm 1: Steady State $\mathrm{GA}_{\mathrm{APSP}}$

optimization process to $O(n^{3.2}(\log n)^{0.2})$. In Section 4, we analyze a crossover selecting two matching individuals, and show that this leads to an optimization time of $O(n^3 \log n)$.

In order to meet space constraints, some proofs had to be left out.

2 Algorithms

For the APSP problem we examine the population-based approach introduced in [4], where each individual in the population is a path. Our goal is to evolve an initial population consisting of a set of paths into a population which contains, for each pair of vertices (u, v) with $u \neq v$, a shortest path from u to v.

We investigate two evolutionary algorithms for the APSP problem that differ on how they apply crossover. The algorithms start with a population $\mathcal{P} := \{P_{u,v} = (u, v) | (u, v) \in E\}$ of size $|E|$, containing all paths corresponding to the edges of the given graph G. The variation operators produce in each iteration one single offspring.

Our algorithm, called Steady State $\mathrm{GA}_{\mathrm{APSP}}$ (see Algorithm 1), decides in each iteration whether the offspring is produced by crossover or mutation. With probability p_c a crossover operator is applied to two randomly chosen individuals of $\mathcal{P}$ or otherwise (with probability $1 - p_c$) mutation is used to produce the offspring. To make sure that both operators, mutation and crossover, are used we require $p_c \notin \{0, 1\}$. For all investigations in this paper, we assume that p_c is chosen as an arbitrary constant, i. e. $p_c \in]0, 1[$.

The mutation operator takes an individual $P_{x,y}$ from the population and applies sequentially $S + 1$ local operations. Here, S is a parameter that is chosen according to the Poisson distribution with parameter $\lambda = 1$. In a local operation, the current path is either lengthened or shortened by a single edge. Assume that the current individual represents a path $P_{x,y} = (x = v_0, v_1, \ldots v_{\ell-1}, y = v_\ell)$ from x to y consisting of ℓ edges, and denote by $E^-(v)$ and $E^+(v)$ the set of incoming and outgoing edges of a vertex v in G, respectively. Then an edge

$e = (u, v) \in E^-(x) \cup E^+(y) \cup \{(x, v_1), (v_{\ell-1}, y)\}$ is chosen uniformly at random. If $e \in \{(x, v_1), (v_{\ell-1}, y)\}$, the edge is removed. This means that either the first edge or the last edge in the path is removed leading to an individual $P'_{v_1,y}$ or $P'_{x,v_{\ell-1}}$ consisting of $\ell - 1$ edges. If $e \in (E^-(x) \cup E^+(y)) \setminus \{(x, v_1), (v_{\ell-1}, y)\}$, the edge is added and the path is lengthened. Here, a new individual $P'_{u,y}$ or $P'_{x,v}$ is produced that contains $\ell + 1$ edges. Note that a local operation applied to a valid path always leads to a new valid solution which implies that the mutation operator only constructs solutions which are paths.

Crossover takes two individuals and combines them into a valid path if the end vertex of $P_{x,y}$ and the start vertex of $P_{x',y'}$ match. Choosing both individuals uniformly at random from $\mathcal{P}$, as it was done in [4,5], often does not lead to a recombined offspring that represents a path in the given graph. In the next section, we discuss how repair mechanisms can lead to more efficient evolutionary algorithms. Later on, we discuss how selection methods that select promising pairs of individuals for crossover lead to evolutionary algorithms that are almost as fast as classical algorithms for the APSP problem.

The selection operator only accepts individuals that are paths in the graph. In addition, it ensures diversity with respect to the different pairs of vertices. For this reason, each individual $P_{u,v}$ is indexed by the start vertex u and the end vertex v. In the selection step an offspring is only compared to an individual of the current population that has the same start and end vertex. It is ensured that, for each pair of vertices (u, v) with $u \neq v$, at most one individual $P_{u,v}$ is contained in the population. This implies that the population size of our algorithms is always at most $n(n - 1)$.

For our theoretical investigations, we measure the optimization time of the algorithm by the number of fitness evaluations until an optimal population has been reached for the first time. A population is optimal if it represents, for each pair of vertices, a shortest path.

Finally, the term w. h. p. (with high probability) denotes a result that holds with probability at least $(1 - O(n^{-c}))$ for some $c > 0$ independent of n.

3 Crossover with Repair

In this section, we present a simple way to increase the success probability of the crossover operator used in previous work. This result, as we shall prove rigorously, is an improved optimization time of $O\big(n^{3.2}(\log n)^{0.2}\big)$.

The main reason why previous crossover operators for the APSP problem have a relatively small success probability is the fact that very often the two parent individuals simply do not fit together. That is, the end-point of the first is not equal to the starting point of the second path. Since this is a rather obvious way of failing, one might think of simple solutions.

One natural way is the following. If end-point of first and starting point of second path are different, we try to bridge this gap by the (if existent, unique) path from one point to the other which is contained in our population. If the population does not contain such a bridging path, then the crossover operator still fails. This is what we shall call *crossover with repair*.

Definition 1 (Crossover with repair). *Let $\otimes_r$ denote the crossover operator with repair as follows (compare Figure 1).*

Input: $P_{x,y} = (x, \dots, y)$ and $P_{x',y'} = (x', \dots, y')$ taken u. a. r. from $\mathcal{P}$
1 **if** $y = x'$ **then**
2 $\quad P'_{s,t} = (s = x, \dots, y = x', \dots, t = y')$ merging $P_{x,y}$ and $P_{x',y'}$ at vertex y ;
3 **else**
4 $\quad$ **if** *there is a path $P_{y,x'}$ from y to x' in $\mathcal{P}$* **then**
5 $\quad\quad P'_{s,t} = (s = x, \dots, y, \dots, x', \dots, t = y')$ merging $P_{x,y}$, $P_{y,x'}$ and $P_{x',y'}$ at their common endpoints;
6 $\quad$ **else**
7 $\quad\quad \otimes_r$ fails and returns a dummy individual with fitness worse than all other possible individuals;

The individual $P_{y,x'}$ from Line 4 is called repair-path.

Note that this operation is inserted in Line 5 of Algorithm 1.

Assuming that all individuals used in the operation have a length of at most k, the application of the $\otimes_r$-operator produces a new individual of size at most $3k$. If all individuals considered for the operation are shortest paths (w. r. t. to their weight) then it is possible to produce shortest paths (w. r. t. to their weight) of length up to $3k$ due to the optimal substructure property of shortest paths. However, later on we will merely consider the case that if all optimal individuals of length k are present in the population the new individual will have a length of $\frac{3}{2}k$.

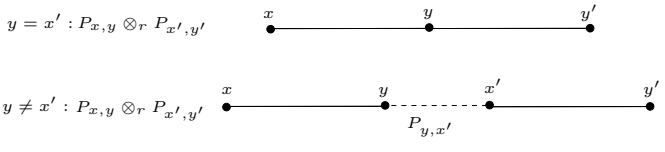

Fig. 1. Effect of the repair crossover applied to two paths $P_{x,y}$ and $P_{x',y'}$.

To analyze our crossover operator, we use the gap concept introduced in [5]. The key observation is that it suffices that crossover finds a path that sufficiently well approximates a sought-after path, because mutation is fast enough to fill the gaps.

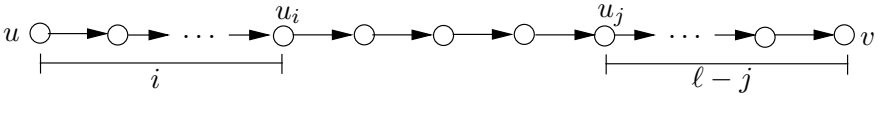

Fig. 2. Approximating path P_{u_i,u_j} with a gap of $g := i + \ell - j$

Definition 2 (Gap). *Consider a path $P_{u,v} = (u = u_0, u_1, \ldots, u_\ell = v)$ and an arbitrary sub-path $P_{u_i,u_j} = (u_i, u_{i+1}, \ldots, u_j)$ with $0 \leq i \leq j \leq \ell$, compare Figure 2. We call the integral value $g := i + \ell - j$ the* gap *of the path P_{u_i,u_j} (w. r. t. P). We also call P_{u_i,u_j} an approximating path of $P_{u,v}$. If $P_{u,v}$ is a shortest path between vertex u and v, we call P_{u_i,u_j} an approximating shortest path.*

For a simplification of the proofs we make use of the following definition which takes into account all pairs of vertices for which there is a shortest path containing at most k edges.

Definition 3. *Let $G = (V, E)$ be a graph and let $k \in \mathbb{R}$. We let V_k^2 be the set of all $(u, v) \in V^2$ with $u \neq v$ such that there exists a shortest path $P_{u,v}$ from u to v consisting of at most k edges.*

Note that we allow for a fractional k in order get rid of some delicate case distinctions later on.

The main statement of this section is captured by the following theorem. It shows that the use of the introduced repair mechanism leads provably to a better optimization time.

Theorem 1. *The Steady State GA_{APSP} with any constant rate $0 < p_c < 1$ using crossover with repair (Definition 1) has an optimization time of $O\big(n^{3.2}(\log n)^{0.2}\big)$ with high probability.*

For the proof of the theorem we need to analyze the success probability of the $\otimes_r$-operator, analyze the success probability of the mutation operator and describe the interplay between mutation and crossover.

We start by investigating the success probability for the crossover operator with repair. Using the crossover operator with repair gives us an additional factor of k for the success probability compared to the corresponding results in [5]. This is made precise in the following lemma.

Lemma 1 (Analysis of Crossover). *Let $k > 1$ and let $\mathcal{P}$ be an arbitrary but fixed population. Assume that $\mathcal{P}$ contains for each pair $(u, v) \in V_k^2$ a shortest path connecting them. Let $\ell := \frac{3}{2}k$ and $g \leq \frac{k}{4}$. Then the following holds.*

(1) A single step of the $\otimes_r$-operator generates a shortest path from u to v with $(u, v) \in V_\ell^2 \backslash V_k^2$ with probability $\Omega\Big(\frac{k^2}{n^4}\Big)$.

(2) Consider a gap g, then a single step of the $\otimes_r$-operator generates an approximating shortest path from u to v with $(u, v) \in V_\ell^2 \backslash V_k^2$ with probability $\Omega\Big(\frac{k^2 g^2}{n^4}\Big)$.

For the progress made by mutation, we use the following result given in [5].

Lemma 2 (Analysis of Mutation). *Let $\ell > 0$ and $\mathcal{P}$ be an arbitrary but fixed population. Assume that there is a shortest path for every $(u, v) \in V_\ell^2$ in $\mathcal{P}$ and the Steady State GA_{APSP} is allowed to use only mutations an no crossover-operations. Then the following holds.*

(1) *The success probability to get an arbitrary but fixed shortest path in* $V^2_{\ell+1} \setminus V^2_\ell$ *is* $\Omega(n^{-3})$. *Hence the expected waiting time for generating such a path is* $O(n^3)$.
(2) *Let* $\lambda > 0$ *and* $c > 0$ *with* $c\lambda \geq 24 \ln n$. *The Steady State* GA_{APSP} *finds all shortest paths* $(u, v) \in V^2_{\ell+c}$ *with probability at least* $n^{2-\frac{c\lambda}{8 \ln n}}$ *in* $O(c\lambda n^3)$ *iterations.*

Combining the analysis of the mutation operator and the crossover operator with repair, we obtain the following key lemma. It nicely shows the interplay between the two operators from the point on when we have shortest paths connecting all pairs in V^2_k for $k = \Omega((n \log n)^{0.2})$.

Lemma 3. *Assume that the Steady State* GA_{APSP} *uses mutation as well as the* $\otimes_r$*-operator with constant probability. Let* $k \geq (n \log n)^{0.2}$ *and* $\Delta := \frac{(n \log n)^{0.2}}{k}$. *Assume that there is a shortest path in the population for each pair* $(u, v) \in V^2_k$. *Let* $\ell := \frac{3}{2}k$.

(1) *For all pairs* $(u, v) \in V^2_\ell \setminus V^2_k$ *an approximating shortest path* (u_i, u_j) *with gap at most* $g \leq (n \log n)^{0.2}\Delta$ *is found in* $t = O(n^3(n \log n)^{0.2}\Delta)$ *iterations with high probability.*
(2) *Assume that for each pair* $(u, v) \in V^2_\ell \setminus V^2_k$ *there is an approximating shortest path* (u_i, u_j) *with a gap of at most* $g \leq (n \log n)^{0.2}\Delta$. *Then the algorithm finds all shortest paths with end-vertices in* V^2_ℓ *in* $t = O(n^3(n \log n)^{0.2}\Delta)$ *iterations with high probability.*
(3) *Let* $k = (n \log n)^{0.2}(1.5)^i$ *for some* $i \in \mathbb{N}_0$. *Then with high probability,* $O((1.5)^{-i}n^{3.2}(\log n)^{0.2})$ *iterations suffice to have all shortest paths of up to* $1.5k$ *edges in the population (where the hidden constant in the time does not depend on* i*).*

Now we are in the position to prove our main theorem.

Proof (of Theorem 1). Both the crossover and the mutation operator have constant probability to be applied in an iteration, and neither can decrease the fitness of an individual. Hence we may occasionally only regard the effect of one of the two. Applying Lemma 2 with $c\lambda := (n \log n)^{0.2}$, we see that after $O(n^3 \cdot (n \log n)^{0.2})$ iterations, with high probability all shortest paths having up to $\ell = (n \log n)^{0.2}$ edges are in the population.

We now repeatedly apply Lemma 3(3). In time $O(n^{3.2} \log_{1.5} n(1.5)^{-i})$, with high probability we construct all shortest paths connecting vertices in $V^2_{(1.5)^{i+1}\ell}$ out of a population containing all shortest paths for vertices in $V^2_{(1.5)^i\ell}$. Hence the run-times form a geometric series and the less than $\log_{1.5} n$ such stages needed to find all shortest paths still take time $O(n^{3.2}(\log n)^{0.2})$. Since each stage works fine with high probability, our algorithm finds all shortest paths with high probability as well.

The previous proof shows that the proposed repair mechanism leads provably to a better optimization time. In the next section, we will examine how selection for reproduction can influence the runtime of crossover-based evolutionary

algorithms. We will see that this even leads to bounds on the optimization time that are close to the ones of problem-specific algorithms.

4 Feasible Parent Selection

The previous section has shown that a simple repair mechanism leads to an optimization time of $O\big(n^{3.2}(\log n)^{0.2}\big)$, which is already an improvement over the optimization time of $O\big(n^{3.25}(\log n)^{0.25}\big)$ for the Steady State $\mathrm{GA}_{\mathrm{APSP}}$ in [5]. Nevertheless, the crossover operator may still produce solutions that do not constitute paths. This is the case if the start vertex of the second individual does not match the end vertex of the first individual and there is no individual in $\mathcal{P}$ for repair.

In the following, we want to make sure that the crossover operator constructs feasible solutions, i. e. individuals that represent paths. This is done by restricting the parent selection for crossover to individuals that match with respect to their endpoints. We choose the two individuals for crossover in Line 5 of the Steady State $\mathrm{GA}_{\mathrm{APSP}}$ (Algorithm 1) using the feasible parent selection procedure given in Algorithm 2.

1 Choose $P_{x,y} \in \mathcal{P}$ uniformly at random.
2 Choose $P_{x',y'} \in \{P_{u,v} \mid P_{u,v} \in \mathcal{P} \wedge u = y \wedge v \neq x\}$ uniformly at random.

Algorithm 2: Feasible Parent Selection

It chooses the first individual $P_{x,y}$ uniformly at random from the population $\mathcal{P}$ and the second individual $P_{x',y'}$ uniformly at random among all individuals in $\mathcal{P}$ whose start vertex equals the end vertex y of $P_{x,y}$ but whose end vertex does not equal the start vertex of $P_{x,y}$. Afterwards, in Line 5, crossover is performed by concatenation. Note that, due to the selection of the two individuals, a path from x to y' is constructed, which implies that the crossover operator only constructs feasible solutions.

This selection operator for the two parents reduces the optimization time even further. The following theorem shows, that the optimization time of Steady State $\mathrm{GA}_{\mathrm{APSP}}$ comes close to the best known upper bound on the runtime of problem specific algorithms if Algorithm 3 is used to select the individuals for crossover.

Theorem 2. *The Steady State GA_{APSP} with any constant rate $0 < p_c < 1$ using feasible parent selection (Algorithm 3) has an optimization time of $O\big(n^3 \log n\big)$ with high probability.*

Our proof of this theorem uses an analysis similar to that of Theorem 1, but without considering gaps.

5 Conclusions

We have shown how the use of repair mechanism or appropriate selection strategies can speed up crossover-based evolutionary algorithms in the special case of the all-pairs shortest path problem. Understanding the usefulness of crossover in evolutionary computation in a rigorous way for other problems remains a challenging task for future research.

References

1. Baswana, S., Biswas, S., Doerr, B., Friedrich, T., Kurur, P.P., Neumann, F.: Computing single source shortest paths using single-objective fitness functions. In: Proceedings of the 10th International Workshop on Foundations of Genetic Algorithms (FOGA 2009), Orlando, USA, pp. 59–66. ACM Press, New York (2009)
2. Cherba, D.M., Punch, W.F.: Crossover gene selection by spatial location. In: Cattolico, M. (ed.) GECCO, pp. 1111–1116. ACM, New York (2006)
3. Doerr, B., Happ, E., Klein, C.: A tight analysis of the (1 + 1)-EA for the single source shortest path problem. In: IEEE Congress on Evolutionary Computation, pp. 1890–1895. IEEE, Los Alamitos (2007)
4. Doerr, B., Happ, E., Klein, C.: Crossover can provably be useful in evolutionary computation. In: Ryan, C., Keijzer, M. (eds.) GECCO, pp. 539–546. ACM, New York (2008)
5. Doerr, B., Theile, M.: Improved analysis methods for crossover-based algorithms. In: Rothlauf [20], pp. 247–254
6. Droste, S., Jansen, T., Wegener, I.: On the analysis of the (1+1) evolutionary algorithm. Theor. Comput. Sci. 276, 51–81 (2002)
7. Friedrich, T., Hebbinghaus, N., Neumann, F., He, J., Witt, C.: Approximating covering problems by randomized search heuristics using multi-objective models. In: Lipson, H. (ed.) GECCO 2007: Proceedings of the 9th annual conference on Genetic and evolutionary computation, pp. 797–804. ACM, New York (2007); Journal version appears in Evolutionary Computation
8. Giel, O., Wegener, I.: Evolutionary algorithms and the maximum matching problem. In: Alt, H., Habib, M. (eds.) STACS 2003. LNCS, vol. 2607, pp. 415–426. Springer, Heidelberg (2003)
9. Horoba, C., Sudholt, D.: Running time analysis of ACO systems for shortest path problems. In: Stützle, T., Birattari, M., Hoos, H.H. (eds.) SLS 2009. LNCS, vol. 5752, pp. 76–91. Springer, Heidelberg (2009)
10. Jansen, T., Wegener, I.: Evolutionary algorithms - how to cope with plateaus of constant fitness and when to reject strings of the same fitness. IEEE Trans. Evolutionary Computation 5(6), 589–599 (2001)
11. Knowles, J.D., Corne, D.: A comparison of encodings and algorithms for multiobjective spanning tree problems. In: Proceedings of the Congress on Evolutionary Computation 2001, pp. 544–551. IEEE Press, Los Alamitos (2001)
12. Kratsch, S., Neumann, F.: Fixed-parameter evolutionary algorithms and the vertex cover problem. In: Rothlauf [20], pp. 293–300
13. Mehlhorn, K., Sanders, P.: Algorithms and Data Structures: The Basic Toolbox. Springer, Heidelberg (2008)
14. Michalewicz, Z., Fogel, D.B.: How to solve it: Modern heuristics. Springer, Berlin (2004)

15. Michalewicz, Z., Nazhiyath, G., Michalewicz, M.: A note on usefulness of geometrical crossover for numerical optimization problems. In: Evolutionary Programming, pp. 305–312 (1996)
16. Neumann, F., Reichel, J.: Approximating minimum multicuts by evolutionary multi-objective algorithms. In: Rudolph, G., Jansen, T., Lucas, S., Poloni, C., Beume, N. (eds.) PPSN 2008. LNCS, vol. 5199, pp. 72–81. Springer, Heidelberg (2008)
17. Neumann, F., Wegener, I.: Randomized local search, evolutionary algorithms, and the minimum spanning tree problem. Theoretical Computer Science 378(1), 32–40 (2007)
18. Raidl, G.R., Julstrom, B.A.: Edge sets: an effective evolutionary coding of spanning trees. IEEE Trans. Evolutionary Computation 7(3), 225–239 (2003)
19. Rothlauf, F.: Representations for Genetic and Evolutionary Algorithms. Springer, Heidelberg (2003)
20. Rothlauf, F. (ed.): Genetic and Evolutionary Computation Conference, GECCO 2009, Proceedings, Montreal, Québec, Canada, July 8-12. ACM, New York (2009)
21. Scharnow, J., Tinnefeld, K., Wegener, I.: The analysis of evolutionary algorithms on sorting and shortest paths problems. Journal of Mathematical Modelling and Algorithms, 349–366 (2004)
22. Walters, T.: Repair and brood selection in the traveling salesman problem. In: Eiben, A.E., Bäck, T., Schoenauer, M., Schwefel, H.-P. (eds.) PPSN 1998. LNCS, vol. 1498, pp. 813–822. Springer, Heidelberg (1998)
23. Witt, C.: Worst-case and average-case approximations by simple randomized search heuristics. In: Diekert, V., Durand, B. (eds.) STACS 2005. LNCS, vol. 3404, pp. 44–56. Springer, Heidelberg (2005)