Black-Box Complexities of Combinatorial Problems

Benjamin Doerr
Max-Planck-Institut für Informatik
Saarbrücken, Germany

Timo Kötzing
Max-Planck-Institut für Informatik
Saarbrücken, Germany

Johannes Lengler
ETH Zürich
Zürich, Switzerland

Carola Winzen
Max-Planck-Institut für Informatik
Saarbrücken, Germany

ABSTRACT

Black-box complexity is a complexity theoretic measure for how difficult a problem is to be optimized by a general purpose optimization algorithm. It is thus one of the few means trying to understand which problems are tractable for genetic algorithms and other randomized search heuristics.

Most previous work on black-box complexity is on artificial test functions. In this paper, we move a step forward and give a detailed analysis for the two combinatorial problems minimum spanning tree and single-source shortest paths. Besides giving interesting bounds for their black-box complexities, our work reveals that the choice of how to model the optimization problem is non-trivial here. This in particular comes true where the search space does not consist of bit strings and where a reasonable definition of unbiasedness has to be agreed on.

Categories and Subject Descriptors

F.2 [**Theory of Computation**]: Analysis of Algorithms and Problem Complexity

General Terms

Theory, algorithms

Keywords

Black-box complexity, running time analysis, combinatorial optimization, theory

1. INTRODUCTION

Black-box complexity is a notion trying to capture how difficult a problem is to be solved via problem-independent, possibly randomized, search heuristics. Roughly speaking, the black-box complexity of a problem (a class of functions to be optimized) is the expected number of function evaluations needed to find the optimum of an unknown member of the class (Droste, Jansen, and Wegener [11]).

This *unrestricted* black-box model sometimes gives unrealistically small complexity values (as compared with run times exhibited by standard randomized search heuristics (RSH)). A way to overcome this is to restrict the class of randomized algorithms regarded (of course, in a way that classic RSH are still included). To this aim, Lehre and Witt [12] suggested an *unbiased black-box* model, in which algorithms are only allowed to generate new solutions from existing ones, and only via so-called unbiased variation operators. Doerr and Winzen [9] regard the restriction that the algorithm has no access to the absolute objective values of solutions, but only to the ranking implied by their fitnesses. This leads to an (unrestricted or unbiased) *ranking-based black-box* model.

A number of deep and sometimes unexpected results exist for the different notions, most of them, however, only regarding artificial test problems like OneMax, LeadingOnes, or jump functions. The focus of this paper is to start an in-depth analysis of black-box complexities for combinatorial problems. As we will see in this paper, a number of additional modeling issues have to be regarded here. We start our analysis with the minimum spanning tree (MST) problem, because here it is generally agreed on that a bit-string representation is most natural. This allows to use the definition of unbiasedness as in [12]. When talking about ranking-based black-box complexity, the two-criteria fitness (total weight, number of connected components) needs attention, but the only reasonable model is to treat the two criteria separately, i.e., to assume comparability in both criteria.

We then proceed to the single-source shortest path (SSSP) problem, where current-best evolutionary approaches use representations different from bit-strings. Here it is not clear a priori what unbiasedness shall mean. Transforming the definition from Lehre and Witt [12] in a straightforward way leads to not very useful results. Taking the problem semantics into account, we find a reasonable definition for unbiasedness and prove meaningful black-box complexities.

Spanning Trees. In one of the earliest theoretical works on evolutionary algorithms for combinatorial optimization problems, Neumann and Wegener [13, 14] analyze the (expected) optimization time of the (1+1) evolutionary algorithm (EA) for the MST problem. They prove that the expected time to find one is $O(m^2 \log(nw_{\max}))$, where n is the number of vertices, m the number of edges and $w_{\max}$

is the maximum of the positive and integral edge weights. It is a major open problem whether the dependence on the maximum edge weight is necessary.

The same bound is proven for a randomized local search (RLS) variant doing one-bit and two-bit flips each with probability $1/2$. This can be easily improved to $O(m^2 \log n)$ by noting that the optimization behavior remains exactly the same if we replace the existing edge weights by the numbers from 1 to m (keeping the relative order of the edge weights unchanged) [16].

Since the MST problem has a natural representation via bit strings, for this combinatorial problem we can easily use the four black-box complexity notions. Our results can be found in Table 1. They imply that the unbiased black-box complexity is asymptotically different for the unary case and for arities ≥ 3.

In a nutshell, the results show that, on the one hand, simple algorithms based on unary operators, such as EAs and RLS can get run times very close to the theoretically optimal; on the other hand, they show how operators of higher arity can further improve on the run time (see also [3,6,8] for higher-arity operators for combinatorial optimization problems).

Shortest Paths. In another one of the earliest theoretical works on evolutionary algorithms for combinatorial optimization problems, Scharnow, Tinnefeld, and Wegener [17,18] analyze how a (1+1) EA solves the SSSP problem.

Since in the SSSP problem a shortest path between the source and any other vertex is sought for, a bit-string representation for solution candidates seems not very natural. Therefore, most works resort to trees or slightly more general structures as representations. To ease the comparison with most existing works on the SSSP problem, in this work we shall only work with the vertex-based representation employed in [18], which, roughly speaking, for each vertex stores its predecessor on the path from the source to it. We note that superior run times were recently proven for an edge-based approach [4].

In addition, also the choice of the fitness function is subtle. In [18], a *multi-criteria fitness* was suggested. For each vertex, the objective function returns the distance from the source in the current solution (infinity, if the vertex is not connected to the source). An offspring is only accepted if, in each of these $n-1$ criteria, it is not worse than the parent. For the natural (1+1) EA building on this framework, they prove an expected optimization time of $O(n^3)$. This was improved to a bound of $O(n^2 \max\{\ell, \log(n)\})$, where ℓ is the smallest height of a shortest path tree [2].

When analyzing the black-box complexity of this formulation of the SSSP problem, we first note that both unbiased and ranking-based complexities make little sense. Since the multi-objective fitness explicitly distinguishes the vertices, treating vertices equally here (as done by unbiased operators) or making individual distances incomparable (as done by component-wise ranking) is ill-natured.

Hence for the multi-criteria fitness, we shall only regard the unrestricted black-box complexity. Interestingly, this problem is also among the few combinatorial problems for which black-box complexity results exist. Droste, Jansen, (Tinnefeld,) and Wegener [10, 11] showed that the unrestricted black-box complexity of the SSSP in the multi-criteria formulation is at least $n/2$ and at most $2n-3$.[1] We first improve these bounds to exactly $n-1$ for both the upper and the lower bound.[2] Surprisingly, if we may assume that the input graph is a complete graph, we obtain a black-box complexity of at most $n/2 + O(1)$, see Table 3 in Section 4.1. That is, the SSSP problem becomes easier (in the black-box complexity sense) if we transform an arbitrary instance to one on a complete graph (but adding expensive dummy edges).

The natural *single-criterion* formulation of the SSSP problem takes as objective simply the sum of the distances of all vertices to the source in the current solution. This approach was dismissed in [18] for the reason that then all solutions with at least one vertex not connected to the source form a huge plateau of equal fitness.

In [1], it was observed that this (artificial) problem dissolves if each unconnected vertex only contributes a large value (e.g., larger than the sum of all edge weights) to the objective value. This is the common way to implement the ∞-value in most algorithms. In this setting, also the single-criterion EA is efficient and finds the optimum, on average, in $O(n^3 \log(nw_{\max}))$ iterations.

For the single-criterion version of the SSSP problem, there is no reason to not regard unbiased black-box complexities. However, we shall see that finding a good notion for unbiasedness is a crucial point here. From the representation point of view, since individuals are nothing more than certain mappings from the vertex set into itself, unbiasedness in the sense of Lehre and Witt would mean that we treat all possible images of each vertex symmetrically. From the problem view-point, unbiased should mean that we treat all vertices (apart from the source) equal. This is a substantial difference, as we shall discuss in detail in Section 4.2. Both approaches lead to different black-box complexities, cf. Table 2.

Due to space limitations, most proofs are omitted in this extended abstract and are available from the authors.

2. THE FOUR BLACK-BOX MODELS

In this section we present two black-box models, the *unrestricted* black-box model by Droste et al. [11] and the *unbiased* model by Lehre and Witt [12]. Furthermore, we shall introduce the unrestricted and the unbiased *ranking-based* black-box models in Section 2. For reasons of space we give only a short presentation of the models. For a more detailed exposure confer [9, 11, 12]. The description below follows closely the one in [7].

Throughout this work, we use the following notations. We denote the positive integers by $\mathbb{N}$ and the positive reals by $\mathbb{R}^+$. For any $k \in \mathbb{N}$, we abbreviate $[k] := \{1, \ldots, k\}$. Analogously, we define $[0..k] := [k] \cup \{0\}$. For $x = x_1 \cdots x_n \in \{0,1\}^n$ we denote by $\bar{x}$ the bitwise complement of x (i.e., for all $i \in [n]$ we have $\bar{x}_i = 1 - x_i$). The bitwise exclusive OR is denoted by $\oplus$. For any set S we denote by 2^S the power set of S, i.e., the set of all subsets of S. By S_n we denote the set of all permutations of $[n]$. Lastly, with $\log_a$ we denote the logarithm to

[1]The only other result published on the black-box complexity of a combinatorial problem is the proof that the NP-complete MaxClique problem has a polynomial black-box complexity, see again [11].

[2]Note that the upper bound in [11] still holds in a more restricted setting, cf. Section 4.1.

	(rb) unrestr.	$*$-ary unb.	unary unb.	rb unary unb.	(rb) binary unb.	(rb) 3-ary unb.
upper bound	$2m+1$	$O(m)$	$O(mn\log n)^{\dagger}$	$O(mn(\log n)^2)$	$O(m\log n)$	$O(m)$
lower bound	$(1-o(1))n$	$\Omega(n)$	$\Omega(m\log n)$	$\Omega(m\log n)$	$\Omega(m/\log n)$	$\Omega(m/\log n)$

Table 1: Upper and lower bounds for the black-box complexity of MST in the different models. Abbreviations: unrestr.= unrestricted, rb = ranking-based, unb. = unbiased.
† $O(mn\log(m/n))$ **if all edge weights are distinct.**

	unrestr.	rb unrestr.	unary struct	binary struct	3-ary struct	unary redir
upper bound	$n(n-1)/2$	$(n-1)^2$	$O(n^3\log n)$	$O(n^2\log n)$	$O(n^2)$	$O(n^3)$

Table 2: Upper bounds for the black-box complexity of SSSP with single-criteria fitness function in the different models. Abbreviations: unrestr.= unrestricted, rb = ranking-based, struct = structure preserving unbiased, redir = redirecting unbiased.

base a, and with log we denote the natural logarithm to base $e := \exp(1)$.

The Unrestricted and Unbiased Black-Box Models. We are interested in measuring the complexity of a problem's *optimizability by randomized search heuristics*. Black-box complexity follows the usual approach to take as a measure the *performance of the best algorithm* out of some class of algorithms. As our main interest is in the performance of RSH, we restrict our attention to the class of algorithms which obtain information about the problem to be solved only by learning the objective value of possible solutions. The objective function is given by an oracle, or as a *black-box*. Using this oracle, the algorithm may query the objective value of all possible solutions, but any such query does only return this solution's objective value and no other information about the objective function.

Naturally, we allow that the algorithms are adaptive and that they use random decisions. However, the only type of action the algorithm may perform is, based on the objective values learned so far, deciding on a probability distribution on the search space $\mathcal{S}$, sampling from it a solution ("search point") $x \in \mathcal{S}$, and querying its objective value ("fitness") from the oracle. This leads to the black-box model by Droste et al.'[11] which contains all algorithms following the scheme of an *unrestricted black-box algorithm*, cf. Algorithm 1.

In typical applications of RSH, the evaluation of the fitness of a search point is more costly than the generation of a new one. Thus, we take as performance measure of a black-box algorithm the number of queries to the oracle until the algorithm first queries an optimal solution. Since we mainly talk about randomized algorithms, we regard the *expected* number of such queries and call this value the *run time* of the black-box algorithm.

For a class $\mathcal{F}$ of functions, the *complexity* of an algorithm A for $\mathcal{F}$ is the worst-case run time, i.e., the maximum run time of A on a function $f \in \mathcal{F}$. The complexity of $\mathcal{F}$ with respect to a class $\mathcal{A}$ of algorithms is the minimum ("best") complexity among all $A \in \mathcal{A}$ for $\mathcal{F}$. Hence, the *unrestricted black-box complexity* of $\mathcal{F}$ is the complexity of $\mathcal{F}$ with respect to the class of all unrestricted black-box algorithms.

Already the authors of [11] noted that the unrestricted black-box is very powerful. As an example, consider a single objective function f. Clearly, the unrestricted black-box

Algorithm 1: Scheme of an Unrestricted Black-Box Algorithm

```
1 Initialization: Sample $x^{(0)}$ according to some
    probability distribution $p^{(0)}$ on $\mathcal{S}$. Query $f(x^{(0)})$.
2 Optimization: for $t = 1, 2, 3, \ldots$ until termination
    condition met do
3     Depending on
      $\big((x^{(0)}, f(x^{(0)})), \ldots, (x^{(t-1)}, f(x^{(t-1)}))\big)$ choose a
      probability distribution $p^{(t)}$ on $\mathcal{S}$.
4     Sample $x^{(t)}$ according to $p^{(t)}$, and query $f(x^{(t)})$.
```

complexity of $\{f\}$ is 1 — the algorithm which queries an optimal solution of f as first action shows this bound.

This motivated Lehre and Witt [12] to introduce a more restrictive black-box model, where algorithms may generate new solution candidates only from random or previously generated search points and only by using *unbiased* variation operators. Note already here that Lehre and Witt formulated their model only for the hypercube $\{0,1\}^d$ as search space. In Section 4, we propose two ways to carry over the notion to a different setting. Next is a brief presentation of the model by Lehre and Witt.

Definition 1. *Let* $k \in \mathbb{N}$. *A* k-ary unbiased distribution *is a family of probability distributions* $\big(D(\cdot \mid y^{(1)}, \ldots, y^{(k)})\big)_{y^{(1)},\ldots,y^{(k)} \in \{0,1\}^d}$ *over* $\{0,1\}^d$ *such that for all* inputs $y^{(1)}, \ldots, y^{(k)} \in \{0,1\}^d$ *the following two conditions hold.*

(i) $\forall x, z \in \{0,1\}^d$:

$$D(x \mid y^{(1)}, \ldots, y^{(k)}) = D(x \oplus z \mid y^{(1)} \oplus z, \ldots, y^{(k)} \oplus z)\,;$$

(ii) $\forall x \in \{0,1\}^d \, \forall \sigma \in S_n$:

$$D(x \mid y^{(1)}, \ldots, y^{(k)}) = D(\sigma(x) \mid \sigma(y^{(1)}), \ldots, \sigma(y^{(k)}))\,,$$

where $\sigma(x) := x_{\sigma(1)} \cdots x_{\sigma(d)}$.

We refer to the first condition as $\oplus$-invariance *and to the second as* permutation invariance. *An operator sampling from a* k*-ary unbiased distribution is called a* k-ary unbiased variation operator.

1-ary (i.e., mutation only) operators are also called *unary* and we refer to 2-ary (i.e., crossover type) operators as *bi-*

nary ones. If arbitrary arity is considered, we call the corresponding model the $*$-ary unbiased black-box model.

A k-ary unbiased black-box algorithm [12] can now be described via the scheme of Algorithm 2. The *k-ary unbiased black-box complexity* of some class of functions $\mathcal{F}$ is the complexity of $\mathcal{F}$ with respect to all k-ary unbiased black-box algorithms.

Algorithm 2: Scheme of a k-ary Unbiased Black-Box Algorithm

```
1 Initialization: Sample $x^{(0)} \in \{0,1\}^d$ uniformly at random and query $f(x^{(0)})$.
2 Optimization: for $t = 1, 2, 3, \dots$ until termination condition met do
3     Depending on $\big(f(x^{(0)}), \dots, f(x^{(t-1)})\big)$ choose up to $k$ indices $i_1, \dots, i_k \in [t-1]$ and a $k$-ary unbiased distribution $D(\cdot \mid x^{(i_1)}, \dots, x^{(i_k)})$.
4     Sample $x^{(t)}$ according to $D(\cdot \mid x^{(i_1)}, \dots, x^{(i_k)})$ and query $f(x^{(t)})$.
```

Contrary to the unrestricted model, Lehre and Witt [12] could show that all functions with a single global optimum have a unary unbiased black-box complexity of $\Omega(n \log n)$, a bound which, for several standard test problems, is met by different unary randomized search heuristics, such as the $(1+1)$ EA or RLS. For results on higher arity models we refer to the work of Doerr et al. [5].

Ranking-Based Black-Box Models. Another possible restriction of the black-box model was introduced by Doerr and Winzen [9]. The authors observed that many standard RSH do not take advantage of knowing *exact* objective values. Rather, for creating the next search points, many RSH always select those individuals with largest fitness values, examples are given below.

Definition 2. *Let S be a finite set, let $f : S \to \mathbb{R}$ be a function, and let C be a subset of S. The* ranking *ρ of C with respect to f assigns to each element $c \in C$ the number of elements in C with a smaller f-value plus 1, formally, $\rho(c) := 1 + |\{c' \in C \mid f(c') < f(c)\}|$.*

Note that two elements with the same f-value are assigned the same ranking.

Following [9], we restrict the two black-box models which we introduced in the previous section to black-box algorithms that use no other information than this ranking.

Unrestricted Ranking-Based Black-Box Model. The unrestricted ranking-based black-box model can be described via the scheme of Algorithm 1 where we replace the third line by "Depending on the ranking of $\{x^{(0)}, \dots, x^{(t-1)}\}$ with respect to f, choose a probability distribution $p^{(t)}$ on $\mathcal{S}$."

Unbiased Ranking-Based Black-Box Model. For the definition of the unbiased ranking-based model we consider the scheme of Algorithm 2 and replace the third line by "Depending on the ranking of $\{x^{(0)} \dots, x^{(t-1)}\}$ with respect to f, choose up to k indices $i_1, \dots, i_k \in [t-1]$ and a k-ary unbiased distribution $D(\cdot \mid x^{(i_1)}, \dots, x^{(i_k)})$."

Both ranking-based black-box models capture many common search heuristics, such as $(\mu + \lambda)$ evolutionary algorithms, some ant colony optimization algorithms, and RLS. They do not include algorithms like simulated annealing algorithms, threshold accepting algorithms, or evolutionary algorithms using fitness proportional selection.

Doerr and Winzen [9] could show that while the basic and the ranking-based models yield the same asymptotic bounds for some problems (e.g., the OneMax function class), it does matter not to consider exact fitness values for other problems, e.g., the BinaryValue function class.

3. MINIMUM SPANNING TREES

Definition 3 (MST). *The* Minimum Spanning Tree *(MST) problem consists of a connected graph $G = (V, E)$ on $n := |V|$ vertices and $m := |E|$ weighted edges. The edge weights $w(e)$, $e \in E$, are positive real numbers. The objective is to find an edge set $E' \subseteq E$ of minimal weight that connects all vertices.*

We encode this problem in binary representation as follows. First, we enumerate the edges in E in arbitrary order $\nu : E \to [m]$. For every bit string $x \in \{0,1\}^m$ we then interpret x as the subset of edges $E_x := \{\nu^{-1}(i) \in E \mid x_i = 1\}$. In the following, we assume that the enumeration of the edges is *not known* to the algorithm. So the algorithm only knows the numbers n and m, but neither knows the geometry of the graph nor which bit corresponds to which edge. However, it may assume that the graph is connected since otherwise no solution of MST exists.

For $E' \subseteq E$ let $c(E')$ be the number of connected components induced by E', and let $w(E') = \sum_{e \in E'} w(e)$ be the total weight of E'. The book [15] argues that the objective function $f(E') = (c(E'), w(E'))$ is "appropriate in the black-box scenario". Thus, if an algorithm *queries* some $x \in \{0,1\}^m$, it receives $f(E_x) = (c(E_x), w(E_x))$ as answer.

In the ranking-based models, the objective value consists of a ranking of both components. That is, the oracle reveals two rankings of the search points, based on the first and the second component, respectively.

We obtain the following upper bounds by modifying Kruskal's algorithm to fit the black-box setting at hand.

Theorem 4 (Upper Bounds for MST). *The* (ranking-based) unrestricted *black-box complexity of the MST problem is at most $2m + 1$. The* unary unbiased *black-box complexity is $O(mn \log(m/n))$ if there are no duplicate weights and $O(mn \log n)$ if there are. The* ranking-based unary unbiased *black-box complexity is $O(mn \log n)$. The* (ranking-based) binary unbiased *black box-complexity is $O(m \log n)$. The* (ranking-based) 3-ary unbiased *black-box complexity is $O(m)$.*

We conjecture that it is not an artifact of our methods that we obtain different upper bounds, but that all four complexity classes of the MST problem are different.

Theorem 5 (Lower Bounds for MST). *The unrestricted black-box complexity of MST for complete graphs is at least $(1 - o(1))n$.*

Proof. We apply Yao's minimax principle [19]. To this end, we show that there exists a probability distribution on the input set of all weighted complete graphs such that every deterministic algorithm needs at least $(1 - o(1))n$ queries to compute a MST. More precisely, we consider the distribution

p on the set of all inputs where we sample uniformly at random a spanning tree, and give weight 1 to all of its edges. All other edges receive weight 2. We call edges of weight 1 "cheap", and all other edges "expensive".

Let us now consider a fixed deterministic algorithm A. We assume that the algorithm already knows which bit in the vector corresponds to which edge in the graph. This assumption makes life only easier for the algorithm. Then for each query the algorithm knows in advance how many connected component its query has. So the first component of the objective function does not contain any information. If the algorithm makes a query consisting of k edges, then the total weight of all these edges is contained in the interval $[2k-n+1, 2k]$, depending of how many cheap edges the query contains. Therefore, each query gives at most $\log_2(n)$ bits of information.

Obviously, the algorithm A needs to learn the set of all cheap edges. It is well known that the number of spanning trees on n vertices is n^{n-2} (so-calles Cayley's formula). Therefore A needs to learn $(n-2)\log_2(n)$ many bits, so it has in the worst case a run time of $T := n-2$. Moreover, for every $0 \le t \le T$, after $T-t$ many queries the probability to find the correct solution is at most n^{-t}. Therefore, the probability that A needs at least T steps is at least

$$\Pr[T(I_p, A) \ge T] \ge 1 - \sum_{t=1}^{T-1} n^{-t} \ge 1 - \frac{n^{-1}}{1-n^{-1}} = 1 - o(1).$$

Note that the hidden constant in $o(1)$ does not depend on the algorithm A. By Markov's inequality, the expected run time is at least

$$\begin{aligned} \mathrm{E}[T(I_p, A)] &\ge T \cdot \Pr[T(I_p, A) \ge T] \\ &\ge T \cdot (1 - o(1)) \\ &= (1 - o(1))n. \end{aligned}$$

Since this holds for all deterministic algorithms A, Yao's minimax principle implies the statement. □

In order to prove a lower bound in the unbiased setting, we compare MST with the auxiliary problem $\textsc{OneMax}_m$. The search space of $\textsc{OneMax}_m$ is the space $\{0,1\}^m$, and for each vector $x \in \{0,1\}^m$ the objective value is given by $\textsc{OneMax}_m(x) := \sum_{i=1}^m x_i$, the number of 1-bits in x.

Theorem 6. *The k-ary unbiased black-box complexity of MST for n vertices and m edges is at least as large as the k-ary unbiased black-box complexity of* $\textsc{OneMax}_m$.

It has been shown in [12] that $\textsc{OneMax}_m$ has a unary unbiased complexity of $\Theta(m \log m) = \Theta(m \log n)$ and in [5] it was proven that the $*$-ary unbiased complexity of $\textsc{OneMax}_m$ is $\Theta(m/\log m) = \Theta(m/\log n)$. This yields the following.

Corollary 7. *The unary unbiased black-box complexity of MST is in $\Omega(m \log n)$; for all other arities, the unbiased black-box complexity of MST is in $\Omega(m/\log n)$.*

4. SINGLE-SOURCE SHORTEST PATH

Definition 8 (SSSP). *The* Single-Source Shortest Path *(SSSP) problem consists of a connected graph $G = (V, E)$ on $n := |V|$ vertices and $m := |E|$ edges. The edge weights $w(e)$, $e \in E$, are positive real numbers. There is a distinguished* source vertex *$s \in V$. The objective is to find for all vertices $v \in V$ a path p_v in G from s to v such that the total weight of p_v, $\sum_{e \in p_v} w(e)$, is minimal among all paths from s to v.*

For the SSSP problem, it is less clear what a good choice of the search space and the objective function is. Two approaches have been regarded, which we discuss in the following subsections.

We assume without loss of generality that the nodes are labeled by $1, \dots, n$ and that $s = 1$ is the source for which we need to compute the shortest path tree. Let $w : E \to \mathbb{R}^+$ be the weight function of the edges.

4.1 SSSP with Multi-Criteria Fitness

The paper [11] argues for a multi-criteria objective function, where any algorithm may query arbitrary trees on $[n]$ and the objective value of any such tree is an $n-1$ tuple of the distances of the $n-1$ non-source vertices to the source $s = 1$ (if an edge is traversed which does not exist in the input graph, the entry of the tuple is ∞).

The paradigm underlying the unbiased black-box complexity is that the algorithm should not be allowed to exploit knowledge about solution candidates stemming from their representation, but only information stemming from their fitness and the population history. This is why only unbiased variation operators are admitted, which fulfill certain symmetry properties.

For the multi-objective formulation of the SSSP problem, we thus feel that there is little room for unbiasedness. With the fitness explicitly distinguishing the vertices, imposing certain symmetry conditions among the vertices makes little sense. A similar argument makes us not regard ranking-based black-box complexities for this problem.

[11] shows that the unrestricted black-box complexity of this problem is lower bounded by $n/2$ and upper bounded by $2n-3$.[3]

In this section, we first improve the bounds from [11] and match them. Then we restrict the problem instances to complete graphs, which will avoid objective values of ∞ for the different objectives.

	arbitrary connected graph	complete graph
upper	$n-1$	$\lfloor (n+1)/2 \rfloor + 1$
lower	$n-1$	$n/4$

Table 3: Upper and lower bounds for the unrestricted black-box complexity of SSSP with multi-criteria objective function,

Theorem 9. *The unrestricted black-box complexity of SSSP with arbitrary input graphs is $n-1$.*

Surprisingly, if we may assume that the input graph is complete, we obtain a lower complexity. Note that this includes the case where the complete graph is obtained from

[3]Note that the upper bound holds in a restricted setting where the algorithm may only store up to two previous data points. However, the algorithm witnessing our upper bound for the unrestricted black-box setting does also not require the full storage granted by the unrestricted setting, but merely needs to store a linear number of pointers at any given time.

on arbitrary one by adding dummy edges with artificially high weight. This shows, again, that even small changes in modeling the combinatorial problem can lead to substantial changes in the complexity.

For the upper bound, we cover the graph with $\lfloor (n+1)/2 \rfloor$ spanning trees, and query each of them. From the objective values, it is possible to compute all edge weights in the graph, and thus to compute the optimal solution. For the lower bound, we apply the same techniques as in the proof of 9, but use a more intricate distribution on the set of all spanning trees. Skipping the detais, we obtain the following.

Theorem 10. *The unrestricted black-box complexity of SSSP with* complete *input graphs is bounded from above by $\lfloor (n+1)/2 \rfloor + 1$ and bounded from below by $n/4$.*

We do not regard unbiased models for this setting. As mentioned before, it seems inappropriate to use an objective function that gives vertex-specific information and to still require unbiased variation operators.

4.2 SSSP with Single-Criterion Fitness

It is an interesting question how the bounds of Section 4.1 change if we require the algorithms to be unbiased. The unbiased model of Lehre and Witt in [12] has only been formulated for pseudo-Boolean functions, cf. Section 2. As we are dealing with a different representation here, our first step is to generalize the unbiasedness conditions to the setting of SSSP. As we discuss below, there is no unique best way to generalize unbiasedness to a more general class of problems. Even more, the upper bounds which we obtain for the different unbiasedness models differ by a factor of $\log n$, cf. Theorem 15 and Corollary 13. We conjecture that this difference is not an artifact of our analysis but that there actually exists an asymptotic difference between the two models.

In this section, we consider the following model for the SSSP problem. A representation of a candidate solution will be a vector $(\rho(2), \ldots, \rho(n)) \in [n]^{n-1}$ to be interpreted as follows. The predecessor of node i is $\rho(i)$. Note that we do not require that $\rho(i) \neq i$, nor do we require that the candidate solution forms a tree; RSH without repair mechanisms might generate such solutions. In order to reflect the meaning of the components, the indices of such an x will run from 2 to n, i.e., $x = (x_2, \ldots, x_n)$.

We can now formulate a first unbiasedness condition for this model. We require that, for any source-preserving permutation of the nodes, the probabilities are preserved. Intuitively, all subgraphs with the same structure but different labels are equally likely to be chosen.

Definition 11. *Let $k \in \mathbb{N}$. A* k-ary structure preserving unbiased distribution *is a family of probability distributions $\big(D(\cdot \mid y^{(1)}, \ldots, y^{(k)})\big)_{y^{(1)},\ldots,y^{(k)} \in [n]^{n-1}}$ over $[n]^{n-1}$ such that for all* inputs *$y^{(1)}, \ldots, y^{(k)} \in [n]^{n-1}$ the distribution $D(\cdot \mid y^{(1)}, \ldots, y^{(k)})$ is invariant under relabeling of the non-source nodes.*

That is, for all $\sigma \in S_n$ with $\sigma(1) = 1$ and for all $x \in [n]^{n-1}$ we have that

$$D(x \mid y^{(1)}, \ldots, y^{(k)}) = D(\hat{\sigma}(x) \mid \hat{\sigma}(y^{(1)}), \ldots, \hat{\sigma}(y^{(k)}))\,,$$

where $\hat{\sigma}(x) := \big(\sigma(x_{\sigma^{-1}(2)}), \ldots, \sigma(x_{\sigma^{-1}(n)})\big)$.

Alternatively, as search points are just mappings from the vertex set into itself, we might require that all possible images of each vertex are to be treated symmetrically. Formally, we require the following.

Definition 12. *Let $k \in \mathbb{N}$. A* k-ary redirecting unbiased distribution *$\big(D(\cdot \mid y^{(1)}, \ldots, y^{(k)})\big)_{y^{(1)},\ldots,y^{(k)} \in [n]^{n-1}}$ is a family of probability distributions over $[n]^{n-1}$ such that for all* inputs *$y^{(1)}, \ldots, y^{(k)} \in [n]^{n-1}$ the distribution $D(\cdot \mid y^{(1)}, \ldots, y^{(k)})$ is invariant under redirecting the nodes. That is, for all vectors $\sigma = (\sigma_2, \ldots, \sigma_n) \in S_n^{n-1}$ of permutations, and for all $x \in [n]^{n-1}$ we require*

$$D(x \mid y^{(1)}, \ldots, y^{(k)}) = D(\vec{\sigma}(x) \mid \vec{\sigma}(y^{(1)}), \ldots, \vec{\sigma}(y^{(k)}))\,,$$

where $\vec{\sigma}(x) := \big(\sigma_2(x_2), \ldots, \sigma_n(x_n)\big)$.

As in the hypercube model, we call an operator sampling from a k-ary structure preserving unbiased distribution a *k-ary structure preserving unbiased variation operator* and, similarly, an operator that samples from a k-ary redirecting unbiased distribution is called a *k-ary redirecting unbiased variation operator*.

To get a better understanding of the above definitions, let us investigate the restrictions for the unary case $k = 1$. We first look at the structure preserving model. Assume we have a search point z, and we would like to sample the next search point according to a probability distribution D_z on the search space. We may do so if and only if there is an unbiased family of probability distributions $(D(\cdot \mid y))_y$ such that $D(\cdot \mid z) = D_z$. A necessary condition is

For every source-preserving permutation σ with $\hat{\sigma}(z) = z$ and all $x \in [n]^{n-1}$ it holds that $D_z(x) = D_z(\sigma(x))$. (1)

This condition is also sufficient. Assume D_z satisfies (1). Then for every y there are two possibilities.

1. Either there exists a σ such that $\hat{\sigma}(y) = z$. In this case, we define $D(x \mid y) := D(\hat{\sigma}(x) \mid z)$ for all $x \in [n]^{n-1}$.
2. Or there does not exist such a σ. In this case, we let $D(\cdot \mid y)$ be the uniform distribution on the search space.

It can be checked that the family $D(\cdot \mid y)$ is a structure preserving unbiased family of distributions. The same holds for the redirecting unbiased model if we take σ from the set of all vectors in S_n^{n-1} and replace $\hat{\sigma}$ by $\vec{\sigma}$.

Now we can determine the distributions D_z satisfying condition (1). For the structure-preserving model, we need to find all source-preserving permutations that leave z unchanged. These are in one-to-one correspondence with the source-preserving automorphisms of the graph induced by z. If A denotes the group of these automorphisms, then condition (1) is that D_z must be invariant under A, i.e., for all $\alpha \in A$ and all $x \in [n]^{n-1}$ we require $D_z(x) = D_z(\alpha(x))$.

For the redirecting model, we need to determine all families $\sigma \in S_n^{n-1}$ such that $\vec{\sigma}(z) = z$. Consider any component y_i of y. Then we can choose σ_i to be any permutation of $[n]$ with $\sigma_i(z_i) = z_i$. In particular, for all $s, t \in [n] \setminus \{z_i\}$ there is such a permutation mapping s to t. Therefore, a distribution D_z is redirecting unbiased if and only if the following condition is satisfied. 'If $x^{(1)}, x^{(2)} \in [n]^{n-1}$ are vectors such that for every $i \in [2, \ldots, n]$ the equations $x_i^{(1)} = z_i$

and $x_i^{(2)} = z_i$ are either both true or are both false, then $D_z(x^{(1)}) = D_z(x^{(2)})$." Similar considerations hold for higher arity. We omit the details.

Since we do not want the objective function to give vertex-specific information, we use the single-criterion objective function $f_G(\rho(2), \dots, \rho(n)) := \sum_{i=2}^{n} d_i$ where d_i is the distance of the i-th node to the source. If an edge – including loops – is traversed which does not exist in the input graph, we set $d_i := C$ where C is some very large constant (e.g., we could choose $C := nw_{\max}$). In all models, the constant C can be learned by the algorithm in a constant number of queries, e.g., by querying the objective value of search point $(2, \dots, 2)$ in the unrestricted model and dividing it by $n-1$ and, similarly, querying a search point with "all nodes to one non-source node" in the structure preserving unbiased model, and "all nodes to the same node" in the redirecting unbiased model. In the latter one we may perform two independent such queries to guarantee a probability of at least $1-n^{-2}$ to learn the exact value of C. Thus, we assume that the value C is known to the algorithm.

It has been argued in [1] that the RLS algorithm, which in each iteration flips exactly one bit chosen uniformly at random, solves the single-source shortest path problem with the single-criterion objective function in $O(n^3)$ iterations. Since this algorithm is contained in the ranking-based unrestricted black-box model, we immediately gain an upper bound of $O(n^3)$. RLS is also contained in the redirecting unbiased model.

Corollary 13. *The ranking-based unrestricted black-box complexity and the ranking-based unary redirecting unbiased black-box complexity of the SSSP with the single-criterion fitness function is $O(n^3)$.*

We conjecture that already for the (non-ranking-based) unary redirecting unbiased model this bound is tight. However, the following shows that we can achieve better bounds in the unrestricted black-box model.

Theorem 14. *The unrestricted black-box complexity of the SSSP with the single-criterion objective function is at most $\sum_{i=1}^{n-1} i = n(n-1)/2$ and the ranking-based unrestricted one is at most $(n-1)^2$.*

This theorem, as all the subsequent theorems, can be proven by applying some variants of Dijkstra's algorithm. It is possible to derive the edge weights in the unrestricted model from the oracle's answers, so we need only $n-1-i$ queries to add the i-th vertex to the tree. In the ranking-based model we have less information and need up to $(n-1)$ queries to add a new edge.

For the structure-preserving unbiased model, things get more involved. In the unary case we need $O(n^2 \log n)$ queries to add a new edge. In the binary and 3-nary case, some precomputations are possible that reduce the run time.

Theorem 15. *The unary structure-preserving unbiased black-box complexity of SSSP is $O(n^3 \log n)$.*

Theorem 16. *The binary structure-preserving unbiased black-box complexity of SSSP is $O(n^2 \log n)$.*

Proof (sketch). We imitate Dijkstra's algorithm. Note however, that in the structure-preserving unbiased model we are not allowed to (i) direct a node to some node of our choice, e.g., to the node that was lastly added to the search tree; and (ii) we cannot simply add a vertex to the current solution but need to construct this new solution.

To overcome the first point, we split up the algorithm in two phases. In the **search phase**, we do not actually find a search point encoding the search tree T, but rather for every leaf v of the tree we store a search point that contains the path in T from the source to v, with all other nodes pointing to themselves.

When the search phase is completed, we know the structure of the search tree, and start the **construction phase**. In this phase, we grow the desired search tree T.

Remarkably, we need binary operators only to check whether we have added the correct vertex to the tree in the construction phase.

Let us start with the search phase. For the first step, let v_1 be the source, and store the search point where every vertex points to itself. For learning how to add the $k+1$-th node, assume that for k nodes $v_1, \dots, v_k$ we have learned already how to add them to the shortest path tree. We call the other nodes "free nodes".

Consider the search point $x^{(k)}$ lastly found. As mentioned above, $x^{(k)}$ consists of a the shortest path from the source to v_k, and all nodes not on this path point to themselves. We call the nodes on the shortest path from v_1 to v_k "connected nodes". Note that a node cannot at the same time be free and connected, but there may be nodes that are neither free nor connected. We apply $O(n \log n)$ times the following unary unbiased operator. "Create the path z from $x^{(k)}$ by pointing exactly one of the unconnected nodes to v_k. Let every other unconnected node point to itself." The operator is unbiased since every source-preserving automorphism of $x^{(k)}$ must fix the path in $x^{(k)}$, and hence must fix v_k.

With high probability we have queried all possible attachments of free nodes to v_k. We compute the lowest cost for connecting one of them via v_k and compare it with the lowest cost for connecting a free node via one of the vertices $v_1, \dots, v_{k-1}$. Of all these connections, we choose the cheapest and store the corresponding search point.

Since we need to add $n-1$ vertices, the search phase needs $O(n^2 \log n)$ queries.

For the construction phase, assume that we have learned the complete shortest path tree T. I.e., for every vertex v, we have a search point encoding the path in T from the source to v. We want to construct T explicitly. We start with the empty search tree (i.e., the search point where all nodes point to themselves), and add vertices in a depth-first manner.

Now we describe how to construct T iteratively. We call the queries between adding the $(k-1)$st and k-th node the *k-th phase*. At the end of each phase, we choose an *active node*, to which the next node is to be attached. We start with the source being active.

For the $(k+1)$st phase, assume we have already constructed a tree T_k of size k, encoded in a search point $y^{(k)}$. We will make use of the unary operator `attach`$(y^{(k)})$, which creates a search point z from $y^{(k)}$ by redirecting exactly one unconnected node to the active node. In general, this operator does not need to be unbiased, since there may be automorphisms of the search tree mapping the active node somewhere else. However, it is possible to avoid such automorphisms by carefully choosing the order in which the depth first search traverses the children of the active node.

The key idea is to traverse longest search paths first. We omit the details.

Let v be the node we want to attach to the active node, and let x_v be the search point storing the shortest path from s to v, i.e., $x_v = x^{(i)}$ for some $i \in [n]$. We add v to T_k as follows. Sample $z \leftarrow \texttt{attach}(y^{(k)})$. Skipping the details, we note that it is possible by binary unbiased operations to compute in a constant number of queries the number of edges in which z and x_v coincide. Hence we can decide whether the newly added node was v. We keep on sampling $z \leftarrow \texttt{attach}(y^{(k)})$ until we find a z that adds v to $y^{(k)}$, then setting $y^{(k+1)} \leftarrow z$. If v has a child in T then we make v the next active node. Otherwise, we backtrack until we find a node that has an unconnected child, and make this node active.

The expected number of queries needed to add a new node is $O(n)$. Since n nodes need to be added, the construction phase needs $O(n^2)$ queries. $\square$

By allowing 3-ary operators it is possible to imitate Dijkstra's algorithm more directly, without any need to split up the algorithm into two phases as in the proof of the previous theorem. We get the following theorem.

COROLLARY 17. *The 3-ary structure-preserving unbiased black-box complexity of SSSP is $O(n^2)$.*

5. CONCLUSIONS

This first analysis of the different black-box complexity notions for two classic combinatorial optimization problems showed the following. In general, all notions make sense for combinatorial problems as well, though some care has to be taken of how to implement unbiasedness conditions. The particular bounds we find are reasonably close to actual run times observed by existing randomized search heuristics, that is, the black-box complexities give reasonable bounds for the problem difficulties here. In cases where our bounds are smaller than those observed by current best search heuristics, further studies are needed to determine whether current heuristics can be improved, e.g., by using higher-arity variation operators, or whether additional restrictions to the black-box model are needed to exclude artificial algorithms.

Acknowledgment

Timo Kötzing was supported by the Deutsche Forschungsgemeinschaft (DFG) under grant NE 1182/5-1.

Carola Winzen is a recipient of the Google Europe Fellowship in Randomized Algorithms, and this research is supported in part by this Google Fellowship.

6. REFERENCES

[1] S. Baswana, S. Biswas, B. Doerr, T. Friedrich, P. P. Kurur, and F. Neumann. Computing single source shortest paths using single-objective fitness functions. In *Proc. of FOGA'09*, pages 59–66. ACM, 2009.

[2] B. Doerr, E. Happ, and C. Klein. A tight analysis of the (1+1)-EA for the single source shortest path problem. In *Proc. of CEC'07*, pages 1890–1895. IEEE, 2007.

[3] B. Doerr, E. Happ, and C. Klein. Crossover can provably be useful in evolutionary computation. In *Proc. of GECCO'08*, pages 539–546. ACM, 2008.

[4] B. Doerr and D. Johannsen. Edge-based representation beats vertex-based representation in shortest path problems. In *Proc. of GECCO'10*, pages 758–766. ACM, 2010.

[5] B. Doerr, D. Johannsen, T. Kötzing, P. K. Lehre, M. Wagner, and C. Winzen. Faster black-box algorithms through higher arity operators. In *Proc. of FOGA'11*, pages 163–172. ACM, 2011.

[6] B. Doerr, D. Johannsen, T. Kötzing, F. Neumann, and M. Theile. More effective crossover operators for the all-pairs shortest path problem. In *Proc. of PPSN'10*, pages 184–193. Springer, 2010.

[7] B. Doerr, T. Kötzing, and C. Winzen. Too fast unbiased black-box algorithms. In *Proc. of GECCO'11*. ACM, 2011.

[8] B. Doerr and M. Theile. Improved analysis methods for crossover-based algorithms. In *Proc. of GECCO'09*, pages 247–254. ACM, 2009.

[9] B. Doerr and C. Winzen. Towards a Complexity Theory of Randomized Search Heuristics: Ranking-Based Black-Box Complexity. *ArXiv e-prints 1102.1140*, 2011. To appear in Proc. of CSR'11, Springer.

[10] S. Droste, T. Jansen, K. Tinnefeld, and I. Wegener. A new framework for the valuation of algorithms for black-box optimization. In *Proc. of FOGA'03*, pages 253–270. Morgan Kaufmann, 2003.

[11] S. Droste, T. Jansen, and I. Wegener. Upper and lower bounds for randomized search heuristics in black-box optimization. *Theoretical Computer Science*, 39:525–544, 2006.

[12] P. K. Lehre and C. Witt. Black-box search by unbiased variation. In *Proc. of GECCO'10*, pages 1441–1448. ACM, 2010.

[13] F. Neumann and I. Wegener. Randomized local search, evolutionary algorithms, and the minimum spanning tree problem. In *Proc. of GECCO'04*, pages 713–724. Springer, 2004.

[14] F. Neumann and I. Wegener. Randomized local search, evolutionary algorithms, and the minimum spanning tree problem. *Theoretical Computer Science*, 378:32–40, 2007.

[15] F. Neumann and C. Witt. *Bioinspired Computation in Combinatorial Optimization – Algorithms and Their Computational Complexity.* Springer, 2010.

[16] J. Reichel and M. Skutella. On the size of weights in randomized search heuristics. In *Proc. of FOGA'09*, pages 21–28. ACM, 2009.

[17] J. Scharnow, K. Tinnefeld, and I. Wegener. Fitness landscapes based on sorting and shortest path problems. In *Proc. of PPSN'02*, pages 54–63. Springer, 2002.

[18] J. Scharnow, K. Tinnefeld, and I. Wegener. The analysis of evolutionary algorithms on sorting and shortest paths problems. *Journal of Mathematical Modelling and Algorithms*, 3:349–366, 2004.

[19] A. C.-C. Yao. Probabilistic computations: Toward a unified measure of complexity. In *Proc. of FOCS'77*, pages 222–227, 1977.