Information Processing Letters 116 (2016) 428–432

Contents lists available at ScienceDirect

Information Processing Letters

www.elsevier.com/locate/ipl

Black-box complexity: Advantages of memory usage

Tobias Storch

German Aerospace Center (DLR), Earth Observation Center (EOC), Oberpfaffenhofen, 82234 Wessling, Germany

ARTICLE INFO

Article history:
Received 8 June 2015
Received in revised form 19 January 2016
Accepted 24 January 2016
Available online 25 January 2016
Communicated by Ł. Kowalik

Keywords:
Analysis of algorithms
Computational complexity
Randomized algorithms
Theory of computation

ABSTRACT

This article proves for the first time the strong advantages of black-box optimizers with storage size two versus one. On the one hand we illustrate for some classes of functions that the black-box complexity for memory size one is exponential. On the other hand these classes are efficiently optimized by black-box algorithms with memory size two and even by simple genetic algorithms.

1. Introduction

The influence of memory usage is discussed since the first considerations of black-box complexity [4]. The black-box complexity of a class of functions is basically the minimal number of queries of function values until an optimum of any function of the considered class is queried. The function f is unknown with exception of all elements of the search space $x_1, \ldots, x_{t-1}$ queried so far, $t \geq 1$, and their function values. For the most general black-box algorithms the (randomized) selection of x_t is based on $(x_1, f(x_1)), \ldots, (x_{t-1}, f(x_{t-1}))$. Restrictions on the memory size are in the focus of this article. Namely, only a subset of all elements and their function values queried so far is stored. This subset of a fixed size provides the basis for the selection of the next element as well as for the next subset. Since many optimizers like evolutionary algorithms are memory-restricted the respective black-box algorithms and complexity are of particular interest. Other restrictions such as rank-based [2], elitist [3], or unbiased [6] black-box algorithms are not in the focus of this article. The black-box complexity states the difficulty of a problem to be solved by general-purpose optimizers featuring some restrictions or not.

On the one hand for some classes of functions efficient black-box algorithms with restricted memory are presented. E.g., [1] also proved upper bounds of order $n/\log n$ for memory size one on the so-called function class ONEMAX on strings of n bits. And lower bounds of order $n/\log n$ apply even for unrestricted memory. Simple evolutionary algorithms are only a bit less efficient, typically with bounds of order $n \log n$.

On the other hand for some classes of functions large lower bounds are presented. But these results are all independent of memory size, namely applicable to unrestricted storage. E.g., [5] also proved exponential lower and upper bounds on a function class REALJUMP. These results are based on similar bounds for the so-called function class NEEDLE.

The advantages and disadvantages of memory for specific evolutionary algorithms on natural and non-natural functions are already widely analyzed, see also [7]. In many situations randomized search heuristics with memory size one are actually successful. Because of this we answer a major question of Droste, Jansen, and Wegener [4]. For some classes of functions we prove that their memory-less black-box complexity is very large, i.e., optimizers with populations of one element are considered. But with stor-

E-mail address: tobias.storch@dlr.de.

http://dx.doi.org/10.1016/j.ipl.2016.01.009
0020-0190/

Algorithm 1 Black-box algorithm with memory size $k \geq 0$.

Let the multiset $\mathcal{P}_0 = \emptyset$.
Step $t \geq 1$. Depending on t and $\mathcal{P}_{t-1}$, choose some probability distribution p_t on S and create a random element $x_t \in S$ according to p_t. Query $f(x_t)$. Choose $\mathcal{P}_t \subseteq \mathcal{P}_{t-1} \cup \{(x_t, f(x_t))\}$ of size $|\mathcal{P}_t| \leq k$.

age (of size two) the black-box complexity becomes very small.

In Section 3 we prove the big gap in the efficiency for the memory-less situation versus the smallest memory-restriction, i.e., populations of two elements. To achieve these results we investigate in Section 2 the degraded memory-restriction, i.e., populations of no elements at all. We finally extend our findings to generalizations of the already considered real royal road functions by [8]. In Section 4 they are examined in combination with simple genetic algorithms necessarily using operators combining two elements.

First we have to specify black-box algorithms and complexity especially with respect to memory in Section 1.1. In Section 1.2 we recall the well-known methods for proofs of lower bounds. And to apply them we analyze in Section 1.3 the influences of memory-restrictions on the appearance of decision trees. Decision trees originally describe memory-unrestricted black-box algorithms.

1.1. Black-box optimization and memory

We regard a black-box algorithm A on a function $f : S \to \mathbb{N}$ for all $f \in F$ of a class F. Let S and F be finite and without loss of generality we consider maximization of f. If $k = 0$ the algorithm is called *degraded* as commented below. If $k = 1$ it is called *memory-less*. For $1 < k < \infty$ it is called *memory-restricted* and *memory-unrestricted* if $k = \infty$. And if p_t is a deterministic distribution for all t, the algorithm is called *deterministic*. Otherwise it is called *randomized*. We call the multiset $\mathcal{P}_t$ the *population* and x_t the *offspring*.

For the upper bounds in this article we always define even deterministic algorithms and even without considering the step counter t. However, most types of optimizers like evolutionary algorithms do not have access to t. And the lack of access to t strongly challenges some proofs of upper bounds, see also [1].

Let $T(f, A)$ denote the expected number of steps of A that are needed until the function value of an element $x \in S$ with $f(x) \geq f(y)$ for all $y \in S$ is queried, namely an optimum. With $T(F, A) := \max\{T(f, A) \mid f \in F\}$ we have defined the worst-case expected number of steps of A concerning F. And with $T(F) := \min\{T(F, A) \mid A\}$ the *black-box complexity of* F is defined, namely the number of function evaluations necessary to find the maximum of any member of F.

1.2. Methods for proofs of lower bounds

For lower bounds on black-box complexity we do not apply any restrictions. For the proofs we define a maximal number of function evaluations $t_{\max}$ and we assume that latest at step $t_{\max}$ an optimum is evaluated. This does not increase the number of function evaluations. For the memory-unrestricted situation $t_{\max}$ is limited by $|S|$, because multiple queries of the same element are easily avoided.

We observe that the set of black-box algorithms is finite, if the search space S and the class F of functions are finite. And every randomized algorithm is equivalent to a probability distribution on deterministic algorithms. The dependency also on t for the choice of the offspring is essential, because otherwise for the deterministic algorithms the offspring is necessarily the same for the same population and for all steps. This is unrepresentable for randomized situations as discussed in detail by [3].

Hence, it is possible to apply the method for proofs of lower bounds known as the minimax principle by Yao [10].

Theorem 1 *(Yao's minimax principle). Let F be a finite class of functions on a finite search space S, and let $\mathcal{A}$ be a finite set of deterministic algorithms on the set F. For every probability distribution p on F and every probability distribution q on $\mathcal{A}$ it holds* $\min_{A \in \mathcal{A}} T(f_p, A) \leq \max_{f \in F} T(f, A_q)$.

1.3. Ordered directed acyclic decision graphs

A deterministic memory-unrestricted black-box algorithm is equivalently expressed by an ordered directed decision tree [9]. The root at level $t = 1$ represents the population $\emptyset$ and offspring x_1. Let v be a node at level $t \geq 1$ representing the population after step $t - 1$ and offspring x_t. There is a directed edge for each potentially obtained $f(x_t)$ from v to a node at level $t + 1$ representing the population after step t. Only a subset $F(v) \subseteq F$ describes the functions which are consistent with all queries and answers on a path from the root to v. At node v it is sufficient to consider all $f(x_t)$, where $f \in F(v)$. For each function f the algorithm A follows a unique path and at level $T(f, A)$ an optimal element is queried.

In the memory-unrestricted situation the population consists of all offspring on the unique path from the root to v. And it therefore forms an ordered directed tree. In memory-restricted situations, as well as in degraded and memory-less situations, the population consists of a subset of all offspring. And two nodes at level t representing different populations or offspring may therefore result in the same node at level $t + 1$. Thus, there may be more than one path from the root to v. And the tree collapses to an ordered directed acyclic graph in memory-restricted, memory-less, and degraded situations.

2. Degraded vs. memory-less black-box algorithms

A class of functions is defined in the following, where on the one hand each degraded black-box algorithm is inefficient and on the other hand an efficient memory-less algorithm exists. We first look at the class N_ℓ, $\ell \in \mathbb{N}^{>0}$, of so-called NEEDLE (in the haystack) functions. The class N_ℓ consists of all functions $N_{\ell,i}$, $i \in \{1, \ldots, \ell\} =: S_\ell$, where $N_{\ell,i}(i) := 1$ (optimum) and $N_{\ell,i}(x) := 0$, if $x \neq i$. Even for unrestricted-memory the black-box complexity of N_ℓ is known to equal $(\ell + 1)/2$ which is proved by [4]. All black-box algorithms for N_ℓ are inefficient, because there is no