Electronic Colloquium on Computational Complexity, Report No. 102 (2010)

Black-Box Search by Unbiased Variation

Per Kristian Lehre*
University of Birmingham
Birmingham, UK
p.k.lehre@cs.bham.ac.uk

Carsten Witt†
Technical University of Denmark
Kgs. Lyngby, Denmark
cfw@imm.dtu.dk

May 1, 2010

Abstract

The complexity theory for black-box algorithms, introduced by Droste et al. (2006), describes common limits on the efficiency of a broad class of randomised search heuristics. There is an obvious trade-off between the generality of the black-box model and the strength of the bounds that can be proven in such a model. In particular, the original black-box model allows polynomial complexity for certain NP-complete problems and provides for well-known benchmark problems relatively small lower bounds, which are typically not met by popular search heuristics.

In this paper, we introduce a more restricted black-box model which we claim captures the working principles of many randomised search heuristics including simulated annealing, evolutionary algorithms, randomised local search and others. The key concept worked out is an unbiased variation operator. Considering this class of algorithms, significantly better lower bounds on the black-box complexity are proved, amongst them an $\Omega(n \log n)$ bound for functions with unique optimum. Moreover, a simple unimodal function and gap functions are considered. We show that a simple (1+1) EA is able to match the runtime bounds in several cases.

*Supported by EPSRC under grant no. EP/D052785/1.
†Supported by Deutsche Forschungsgemeinschaft (DFG) under grant no. WI 3552/1-1.

ISSN 1433-8092

1 Introduction

The theory of randomised search heuristics has advanced significantly over the last years. In particular, there exist now rigorous results giving the runtime of canonical search heuristics, like the (1+1) EA, on combinatorial optimisation problems [15]. Ideally, these theoretical advances will guide practitioners in their application of search heuristics. However, it is still unclear to what degree the theoretical results that have been obtained for the canonical search heuristics can inform the usage of the numerous, and often more complex, search heuristics that are applied in practice. While there is an ongoing effort in extending runtime analysis to more complex search heuristics, this often requires development of new mathematical techniques.

To advance the theoretical understanding of search heuristics, it would be desirable to develop a computational complexity theory of search heuristics. The basis of such a theory would be a computational model that captures the inherent limitations of search heuristics. The results would be a classification of problems according to the time required to solve the problems in the model. Such a theory has already been introduced for local search problems [9]. The goal in *local search* problems is to find any solution that is locally optimal with regards to the cost function and a neighbourhood structure over the solution set. Here, we are interested in *global search* problems, i. e. where the goal is to find any globally optimal solution.

Droste et al. introduced a model for global optimisation called the *black-box model* [4]. The framework considers search heuristics, called black-box algorithms, that optimise functions over some finite domain S. The black-box algorithms are limited by the amount of information that is given about the function to be optimised and have to make queries to an oracle in order to determine fitness values. In this framework, lower bounds on the number of queries required for optimisation can be obtained. An advantage of the black-box model is its generality. The model covers any realistic search heuristic. Despite this generality, the lower bounds in the model are in some cases close or equal to the corresponding upper bounds that hold for particular search heuristics. For example, it is shown that the needle-in-the-haystack and trap problems are hard problems, having black-box complexity of $(2^n + 1)/2$ [4].

However, the black-box model also has some disadvantages. For example, it is proved that the NP-hard MAX-CLIQUE problem has polynomial black box complexity [4]. One cannot expect any realistic search heuristic to solve all instances of this problem in polynomial time. Indeed, the black-box algorithm that achieves the polynomial runtime is contrived. The algorithm first queries for the function values of all possible pairs of nodes, thus uncovering all the edges in the graph. Given knowledge about the instance, the algorithm can find the optimal solution through offline brute force search, without making any further queries to the black-box. This algorithm requires in total no more than $\binom{n}{2} + 1$ function evaluations. Similar black-box algorithms can be envisaged for other NP-hard problems. This result exposes two weaknesses: the lower bounds in the model are often obtained by black-box algorithms that do not resemble any randomised search heuristic. Secondly, the model is too unrestricted with respect to the amount of resources disposable to the algorithm. black-box algorithms can spend unlimited time in-between queries to do computation.

In order to define a more realistic black-box model, one should consider which additional restrictions to consider. Droste et al. suggested to limit the available storage space available to the black box algorithm [4], but did not prove any lower bounds in the space restricted scenario.

The remaining of this paper is organised as follows. Section 2 introduces the new black-box model along with a description of the unbiased variation operators. Section 3 provides the first lower bound in the model for a simple, unimodal problem. Then, in Section 4, we consider a function class that contains Hamming cliffs. Finally, in Section 5, we prove lower bounds that hold for any function with a single, global optimum. The paper is concluded in Section 6.

2 A Refined Black-Box Model

We now present the refined black-box model that is obtained by imposing two additional restrictions in the old black-box model. We start with a preliminary informal description and motivation.

Firstly, we limit the degree of independence between the queries. The initial query is a bitstring chosen uniformly at random. Every subsequent query must be made for a search point that is obtained by applying an unbiased variation operator to one of the previously queried search points. Unbiased variation operators will be defined in Section 2.1.

Secondly, we put an additional restriction on the information that is available to the algorithm by preventing the algorithm from observing the bit values of the search points that are queried. Hence, the only information that can be exploited by the algorithm is the sequence of fitness values obtained when making queries to the black-box, and not the search points themselves. Note that without this restriction, any black-box algorithm could be simulated as follows: For each query x made by the unrestricted algorithm, the restricted algorithm would solve the problem corresponding of minimising the Hamming distance to search point x. This simulation would have an expected overhead factor of $O(n \log n)$ function evaluations.

We now turn our considerations into a formal definition of a refined black-box model, as stated in Algorithm 1. Let us first pick up the unrestricted black-box scenario [4]. The black-box algorithm A is given a class of pseudo-Boolean functions $\mathcal{F}$. An adversary selects a function f from this class and presents it to the algorithm as a black-box. At this point, the only information available to the algorithm about the function f is that it belongs to function class $\mathcal{F}$. The black-box algorithm can now start to query an oracle for the function values of any search points. The runtime $T_{A,f}$ of the algorithm on function f is the number of function queries on f until the algorithm queries the function value of an optimal search point for the first time. The runtime $T_{A,\mathcal{F}}$ on the class of functions is defined as the maximum runtime over the class of functions. In the

Algorithm 1 Unbiased Black-Box Algorithm

$t \leftarrow 0$.
Choose $x(t)$ uniformly at random from $\{0,1\}^n$.
repeat
  $t \leftarrow t+1$.
  Compute $f(x(t-1))$.
  $I(t) \leftarrow (f(x(0)), ..., f(x(t-1)))$.
  Depending on $I(t)$, choose a probability distribution p_s on $\{0, ..., t-1\}$.
  Randomly choose an index j according to p_s.
  Depending on $I(t)$, choose an unbiased variation operator $p_v(\cdot \mid x(j))$.
  Randomly choose a bitstring $x(t)$ according to p_v.
until termination condition met.

unbiased black-box model, queries of new search points must be made according to Algorithm 1. I.e., the initial search point is chosen uniformly at random by an oracle, and subsequent search points are obtained by asking the oracle to apply a given unbiased variation operator to a previously queried search point.

The *unbiased black-box complexity* of a function class $\mathcal{F}$ is the minimum worst case runtime $T_{A,\mathcal{F}}$ among all unbiased black-box algorithms A satisfying the framework of Algorithm 1. Hence, any upper bound on the worst case of a particular unbiased black-box algorithm, also implies the same upper bound on the unbiased black-box complexity. To prove a lower bound on the unbiased black-box complexity, it is necessary to prove that the lower bound holds for any unbiased black-box algorithm. Note that since unbiased black-box algorithms are a special case of black-box algorithms, all the lower bounds that hold for the general black-box model also hold for the unbiased black-box model.

2.1 Unbiased Variation Operators

We formalise variation operators as conditional probability distributions over the search space. Given k search points $x_1, \ldots, x_k$, a variation operator p produces a search point y with probability $p(y \mid x_1, \ldots, x_k)$. We put some restrictions on the probability distribution to capture the essential characteristics of the variation operators that are used by the common randomised search heuristics. Firstly, one can limit the number k of search points that are used to produce the new search point. The number k determines the *arity* of the variation operator. Here, we will only consider *unary* variation operators, i.e. when $k = 1$. Furthermore, we will impose the following two unbiasedness-conditions on the operators:

$$1)\ \forall x, y, z,\ p(y \mid x_1, \ldots, x_k) = p(y \oplus z \mid x_1 \oplus z, \ldots, x_k \oplus z),$$
$$2)\ \forall x, y, \sigma,\ p(y \mid x_1, \ldots, x_k) = p(\sigma_b(y) \mid \sigma_b(x_1), \ldots, \sigma_b(x_k)),$$

where $\oplus$ denotes the exclusive or-operation on bitstrings, and for any permutation σ over $[n]$, σ_b is an associated permutation over the bitstrings defined as

$$\sigma_b(x_1 x_2 \cdots x_n) := x_{\sigma(1)} x_{\sigma(2)} \cdots x_{\sigma(n)}.$$

Variation operators that satisfy the first condition are called $\oplus$-invariant, while variation operators that satisfy the second condition are called σ-invariant. In this paper, unbiased variation operators is defined as variation operators that satisfy both conditions, i.e. $\oplus$-σ-invariant operators. Note that this is a special case of the framework by Rowe, Vose and Wright [18], who study invariance from a group-theoretical point of view.

We claim that the two conditions are natural. Firstly, the variation operators used by common randomised search heuristics are typically $\oplus$-σ-invariant, including flipping a randomly chosen bit position, bitwise mutation with any mutation probability, and uniform sampling of bitstrings. Furthermore, the black-box algorithm is allowed to select a different variation operator in each iteration, as long as the variation operators are statistically independent. This generality allows adaptive variation operators to be covered, including rank-based mutation [16]. Secondly, the two conditions on the variation operators can also be motivated by practical concerns. In applications, the set of bitstrings typically encode the variable settings of candidate solutions, and is rarely the optimisation domain *per se*. Hence, the encoding from variable setting to bit value, and from variable to bitstring position, can be arbitrary. E.g., whether the user encodes a "high temperature" variable setting as 0 or 1, or decides to encode the temperature variable by the first instead of the last variable in the bitstring, should not influence the behaviour of a search heuristic.

Droste and Wiesmann recommended that all search points that are within the same distance of the originating search point should have the same probability of being produced [5]. This unbiasedness criterion, which we call *Hamming-invariance*, can be formalised as follows:

$$3)\ \forall x, y, z, \quad H(x, y) = H(x, z) \Longrightarrow p(y \mid x) = p(z \mid x),$$

where $H(x, y)$ denotes the Hamming distance between x and y. We now show that these criteria are related.

Proposition 1. *Every unary variation operator that is $\oplus$-σ-invariant, is also Hamming-invariant.*

Proof. Assume that p is an $\oplus$-σ-invariant variation operator. Let x, y, u and v be any search points such that $d(x, y) = d(u, v) =: d$. Given these assumptions, we will prove that $p(y \mid x) = p(v \mid u)$ holds. There must exist a permutation σ such that $\sigma(y \oplus x) = v \oplus u$. We then have

$$\begin{aligned} p(y \mid x) &= p(y \oplus x \mid 0^n) = p(\sigma(y \oplus x) \mid 0^n) \\ &= p(v \oplus u \mid 0^n) = p(v \mid u). \end{aligned}$$

By setting $u = x$, it is clear that p is Hamming-invariant. $\square$

When referring to unbiased black-box algorithms in the following, we mean any algorithm that follows the framework of Algorithm 1. In particular, by *unary, unbiased black-box algorithms*, we mean such algorithms that only use unary, unbiased variation operators. The class of unary, unbiased black-box algorithms is general, and includes many well known algorithms, including simulated annealing [11], random local search (RLS) [14], (μ+λ) EA [3, 8, 20], and many other population-based EAs that do not use crossover. Note that the restriction to unary, unbiased variation operators excludes some randomised search heuristics. In particular, the model does not cover EAs that use crossover. Many of the commonly used diversity mechanisms are excluded. Also, estimation of distribution algorithms [12], ant colony optimisation [2] and particle swarm optimisation [10] are not covered by the model.

3 Simple Unimodal Functions

As an initial example, we consider the simple unimodal function LeadingOnes $(x) = \sum_{i=1}^{n} \prod_{j=1}^{i} x_j$. (The function Onemax is covered by the results in Section 5.) The expected runtime of the (1+1) EA on this function is $\Theta(n^2)$ [3], which seems optimal among commonly analysed EAs. Increasing either the offspring or parent population sizes does not reduce the runtime. For (μ+1) EA, the runtime is $\Theta(n^2 + \mu n \log n)$ [20], and for (1+λ) EA, the runtime is $\Theta(n^2 + \lambda n)$ [8].

We now show that the runtime of the (1+1) EA on LeadingOnes is asymptotically optimal in the unbiased black-box model. We define the potential of the algorithm at time step t as the largest number of leading 1- or 0-bits obtained so far, i.e. $k := \max_{0 \leq j \leq t}\{\text{Lo}(x(j)), \text{Lz}(x(j))\}$. The number of 0-bits must be considered because flipping every bit in a bitstring with i leading 0-bits will produce a bitstring with i leading 1-bits. The increase of the potential will be studied using drift analysis [6, 7]. To lower bound the drift, it is helpful to proceed as in the analysis of (1+1) EA on LeadingOnes [3], i.e. to first prove that the substring after the first 0-bit in a given time step is uniformly distributed.

Lemma 1. *If the potential of the algorithm in step $t-1$ is k, then the bits from position $k+2$ to position n in search point $x_i(t)$ are independent and uniformly distributed.*

Proof. We first prove the claim that for all $t \geq 0$, $b \in \{0, 1\}$ and $i, k+2 \leq i \leq n$ it holds that $\mathbf{Pr}[x_i(t) = b] = 1/2$. The proof is by induction over the time step t. The initial search point is sampled uniformly at random, so the claim holds for $t = 0$. Assume that the claim holds for $t = t_0 \geq 0$. The fitness profile $I(t_0)$ does not depend on any of the bits in position i in the previously sampled search points. The choice of search point $x(j)$ the algorithm makes is therefore independent of these bits. By the induction hypothesis, search point $x(j)$ has a bit value of b in position i with probability $p := 1/2$. Letting r be the probability that the variation operator flips bit position i, we have

$\mathbf{Pr}[x_i(t_0+1) = b] = p(1-r) + (1-p)r = 1/2$. By induction, the claim now holds for all t.

We now prove the lemma by induction over the time t. The lemma holds for time step $t = 0$. Assume that the lemma holds for $t = t_0 \geq 0$. For $m := n-k-1$, let $Z \in \{0,1\}^m$ be a random vector where the elements $Z_i, 1 \leq i \leq m$, take the value $Z_i = 1$ if and only if bit position $k+i+1$ flipped in step t_0. Then for all bitstrings $y \in \{0,1\}^m$,

$$\mathbf{Pr}[x_{k+2}(t_0+1) \cdots x_n(t_0+1) = y] = \sum_{z \in \{0,1\}^m} \mathbf{Pr}[Z = z]\, \mathbf{Pr}[x_{k+2}(t_0) \cdots x_n(t_0) = y \oplus z],$$

which by the induction hypothesis (more precisely, the statement about independence) equals

$$\sum_{z \in \{0,1\}^m} \mathbf{Pr}[Z = z] \prod_{i=1}^{m} \mathbf{Pr}[x_{k+i+1}(t_0) = y_i \oplus z_i]$$
$$= 2^{-m} = \prod_{i=1}^{m} \mathbf{Pr}[x_{k+i+1}(t_0+1) = y_i],$$

where the last equality follows from the claim above. This proves the independence at time $t_0 + 1$, and, therefore, the induction step. □

Theorem 1. *The expected runtime of any unary, unbiased black-box algorithm on* LeadingOnes *is* $\Omega(n^2)$.

Proof. We first prove the claim that w.o.p., the potential of the algorithm will at some time be in the interval between $n/2$ and $3n/4$, before the optimum has been found. With probability $1 - 2^{-n/2}$, the initial search point will have potential less than $n/2$. Let integer i be the number of 0-bits in the interval from $n/2$ and $3n/4$ in the bitstring that the algorithm selects next. By Lemma 1 and Chernoff bounds, there is a constant $\delta > 0$ such that with overwhelming probability, this integer satisfies $(1-\delta)n/8 < i < (1+\delta)n/8$. In order to increase the potential from less than $n/2$ to at least $3n/4$, it is necessary to flip every 0-bit in the interval from $n/2$ to $3n/4$ and no other bits. We optimistically assume that exactly i bits are flipped in this interval. However due to the unbiasedness condition, every choice of i among $n/4$ bits to flip is equally likely. The probability that only the 0-bits are flipped is therefore at most $\binom{n/4}{i}^{-1} \leq (4i/n)^i \leq ((1+\delta)/2)^{n/4(1-\delta)}$. The claim therefore holds.

We now apply drift analysis according to the potential k of the algorithm, only counting the steps starting from a potential in the interval $n/2 \leq k < 3n/4$. In order to find the optimum, the potential must be increased by at least $n/4$. Assume that the selected search point has r 0-bits in the first $k+1$ bit positions. In order to increase the potential, it is necessary to flip all these 0-bits, and none of the 1-bits within this interval. This corresponds to consecutively selecting all the r red balls from an urn containing $k+1$ balls. The probability of this event is less than

$$\frac{r}{k+1} \cdot \frac{r-1}{k} \cdots \frac{2}{k-r+2} \cdot \frac{1}{k-r+1} \leq \frac{1}{k+1}.$$

The drift in each step is bounded from above by $\Delta_i(t) \leq (1+\mathbf{E}[Y_t])/(k+1)$, where random variable Y_t is the number of free-riders [3] in step t. Applying Lemma 1 gives $\mathbf{E}[Y_t] \leq \sum_{i=1}^{\infty} 2^{-i} \leq 1$. The polynomial drift theorem [7] now implies that $\mathbf{E}[T] \geq (n/4)/\Delta_t(i) = (k+1)n/4 = \Omega(n^2)$. □

Note that the complexity of LeadingOnes in the unrestricted black-box model[1] is bounded above by $n/2 - o(n)$ [4]. This illustrates that the complexity of a function class can be significantly higher in the unbiased black-box model than in the unrestricted black-box model.

4 Enforcing Expected Runtimes

In this section, we are interested in situations where a black-box algorithm is confronted with a Hamming cliff of size m, i.e., has to change at least m bits in order to arrive at a better search point. Such situations typically arise in combinatorial optimisation, e.g., in the analysis of the (1+1) EA on the minimum spanning tree problem by Neumann and Wegener [14], where at least two edges have to be flipped for an improvement. In the latter case, this is accounted for by an $\Omega(n^2)$ term in the expected runtime. More generally, there exists combinatorial problems that for any m have instances with Hamming cliffs of size m [13]. The (1+1) EA needs $\Omega(n^m)$ steps to overcome a Hamming cliff of size m, and there is an example called $\textsc{Jump}_m$ [3] where also an upper bound $O(n^m)$ holds, i.e., an expected runtime can be enforced. This corresponds to a hierarchy consisting of a class of functions with increasing difficulty.

We are aiming at generalising this result to black-box algorithms with unary unbiased variation. We pick up the general idea of the $\textsc{Jump}_m$ function in [3], but modify the fitness values of the "gaps" (see below) and make the function symmetrical compared to the original definition by introducing a second gap. This prevents black-box algorithms from extracting information from the gap and from reaching the optimum by flipping a very large number of bits. The formal definition follows. Given $2 \leq m < n/2$,

$$\textsc{Jump}_m(x) := \begin{cases} |x|_1 & \text{if } m < |x|_1 \leq n-m \text{ or } |x|_1 = n, \\ 0 & \text{otherwise.} \end{cases}$$

[1] Unlike unbiased black-box algorithms, unrestricted black-box algorithms (the general model studied in [4]) can optimise function classes containing a single function in constant time by querying the optimum in the first iteration. To obtain meaningful results in the unrestricted model, it is therefore necessary to consider the generalised class of functions containing $\textsc{LeadingOnes}_z(x) := \textsc{LeadingOnes}\,(x \oplus z)$ for every bitstring z.

Hence, the search points with at most m or more than $n-m$ (but less than n) 1-bits each form a region/a gap of inferior fitness such that from search points outside at least m bits have to be flipped in order to jump over the gap to the optimum.

In this section, we prove two different lower bounds and an upper bound. For the lower-bound proofs, we apply drift analyses of different complexity. The first drift analysis goes back to the following theorem.

Theorem 2 (Simplified Drift Theorem [17]). *Let X_t, $t \geq 0$, be the random variables describing a Markov process over the state space $S := \{0, 1, ..., N\}$, and denote $\Delta(i) := (X_{t+1} - X_t \mid X_t = i)$ for $i \in S$ and $t \geq 0$. Suppose there exists an interval $[a, b]$ of the state space and three constants $\beta, \delta, \kappa > 0$ such that for all $t \geq 0$*

1. *$\mathbf{E}[\Delta(i)] \geq \beta$ for $a < i < b$, and*
2. *$\mathbf{Pr}[\Delta(i) = -j] \leq 1/(1+\delta)^{j-\kappa}$ for $i > a$ and $j \geq 1$,*

then there exists a constant $c^* > 0$ such that for

$$T^* := \min\{t \geq 0 : X_t \leq a \mid X_0 \geq b\}$$

it holds that $\mathbf{Pr}\left[T^* \leq 2^{c^*(b-a)}\right] = 2^{-\Omega(b-a)}$.

For a second bound, we use a more general drift theorem.

Lemma 2 (Hajek [6]). *Let $X_0, X_1, X_2, \ldots$ be the random variables describing a Markov process over a state space S and $g\colon S \to \mathbb{R}_0^+$ a function mapping each state to a non-negative real number. Pick two real numbers $a(\ell)$ and $b(\ell)$ depending on a parameter ℓ such that $0 \leq a(\ell) < b(\ell)$ holds. Let $T(\ell)$ be the random variable denoting the earliest point in time $t \geq 0$ such that $g(X_t) \leq a(\ell)$ holds. If there are $\lambda(\ell) > 0$ and $p(\ell) > 0$ such that for all $t \geq 0$ the condition*

$$\mathbf{E}\left[e^{-\lambda(\ell)\cdot(g(X_{t+1})-g(X_t))} \mid a(\ell) < g(X_t) < b(\ell)\right] \leq 1 - \frac{1}{p(\ell)}$$

holds then for all time bounds $L(\ell) \geq 0$

$$\mathbf{Pr}\left[T(\ell) \leq L(\ell) \mid g(X_0) \geq b(\ell)\right] \leq e^{-\lambda(\ell)\cdot(b(\ell)-a(\ell))} \cdot L(\ell) \cdot D(\ell) \cdot p(\ell),$$

where $D(\ell) = \max\left\{1, \mathbf{E}\left[e^{-\lambda(\ell)(g(X_{t+1})-b(\ell))} \mid g(X_t) \geq b(\ell)\right]\right\}$.

In the proof of the forthcoming theorems, we also need upper bounds on the tail of the hypergeometric distribution. The following result due to Chvátal [1] is an analogue to the Chernoff bounds for the binomial distribution.

Lemma 3 (Chvátal, [1]). *If X is a hypergeometrically distributed random variable with parameters n (number of balls), m (number of red balls) and r (number of samples), then $\mathbf{Pr}[X \geq \mathbf{E}[X] + r\delta] \leq \exp(-2\delta^2 r)$, where $\mathbf{E}[X] = \frac{rm}{n}$.*

We are ready to prove a lower bound that is $2^{\Omega(m)}$, which means that the size of the gap also increases the lower bound.

Theorem 3. *For any $m \leq n(1-\varepsilon)/2$ with ε an arbitrary positive constant $0 < \varepsilon < 1$, the runtime of any unary, unbiased black-box algorithm on* JUMP_m *is at least 2^{cm} iterations with probability $1 - 2^{-\Omega(m)}$ for a constant $c > 0$.*

Proof. We partition the set of non-optimal search points into three sets: A_1 is all search points x with $|x|_1 \leq m$, A_2 is all search points x with $m < |x|_1 \leq n - m$, and A_3 is all search points x with $n - m < |x|_1 < n$. We distinguish between two cases. Either, the optimum is obtained by varying a search point in region A_2, or the optimum is obtained by entering one of the two plateau regions A_1 and A_3.

In the first case, we assume that the chosen search point x in region A_2 has $i, m \leq i \leq n-m$, 0-bits. The probability of obtaining the optimal solution from x is bounded using the arguments presented in the proof of Theorem 1. We obtain an upper bound of no more than

$$\frac{i}{n} \cdot \frac{i-1}{n-1} \cdots \frac{2}{n-i+2} \cdot \frac{1}{n-i+1} = \frac{1}{\binom{n}{i}} \leq \frac{1}{\binom{n}{m}} \leq (m/n)^m \leq ((1+\varepsilon)/2)^m.$$

By union bound, the probability that less than 2^{cm} applications of the variation operator to points in region A_2 will produce the optimum is $2^{-\Omega(m)}$ for a small enough constant c.

In the second case, we consider any search path $x(t_0)$, $x(t_1)$, $x(t_2), \ldots$ that enters region A_3, i.e. $x(t_0) \notin A_3$ and $x(t_s) \in A_3$ for $s > 1$. Let $X_{t \geq 0}$ be the Markov process, where X_t corresponds to the number of 0-bits in the t-th step of the search path. Let $r \geq 1$ be the number of bits that was flipped by the variation operator in step t. The number of 0-bits that is flipped by the variation operator is a hypergeometrically distributed random variable Z_t with parameters n (number of balls), X_t (number of red balls) and r (number of samples). We will now show that the drift of this Markov process which is defined as $\Delta(i) := (X_{t+1} - X_t \mid X_t = i) = r - 2Z_t$ satisfies the two conditions of Theorem 2 on the interval $[0, m)$. By noting that $\mathbf{E}[Z_t \mid X_t = i] = ri/n \leq r(1-\varepsilon)/2$, we have

$$\mathbf{E}[\Delta(i) \mid X_t = i] = r - 2\mathbf{E}[Z_t \mid X_t = i] = r(1 - 2i/n) \geq \varepsilon,$$

and the first condition of the drift theorem holds for the parameter $\beta := \varepsilon$. The probability of reducing the distance by $j \leq r$ can be bounded using Lemma 3 as

$$\begin{aligned}
\mathbf{Pr}[\Delta(i) = -j] &= \mathbf{Pr}[r - 2Z_t = -j] = \mathbf{Pr}[Z_t = r/2 + j/2] \\
&\leq \mathbf{Pr}\left[Z_t \geq r \cdot \left(\frac{i}{n} + \frac{\varepsilon}{2} + \frac{j}{2r}\right)\right] \\
&= \mathbf{Pr}\left[Z_t \geq \mathbf{E}[Z_t] + r \cdot \left(\frac{\varepsilon}{2} + \frac{j}{2r}\right)\right] \\
&\leq \exp(-\varepsilon^2 r/2) \leq \exp(-\varepsilon^2 j/2).
\end{aligned}$$

The second condition of the drift theorem now holds for the parameters $\delta := \exp(\varepsilon^2/2) - 1$ and $\kappa := 0$. As both conditions hold, the probability that a given search path reaches the optimum within $2^{c^* m}$ steps is $2^{-\Omega(m)}$ for some constant c^*. Note that all search paths that enter region A_3 are indistinguishable with regards to their fitness. Hence, the probability that the algorithm selects a search path that leads to the optimum in less than $2^{c^* m}$ steps is $2^{-\Omega(m)}$.

Finally, it can be proven in the same way that the probability that search paths that enter region A_1 will lead to a global optimum within $2^{c^* m}$ steps is $2^{-\Omega(m)}$. □

We can show a partly stronger bound if $rm \leq \kappa n$ for some $\kappa < 1/2$. This time, the order of growth is $\left(\frac{n}{rm}\right)^{\Omega(m)}$, i. e., there is also a trade-off between the size of r and m in the bound. If $rm = \omega(n)$, the bound is useless. However, e. g., for constant r, we have the runtime behaviour $(n/m)^{\Omega(m)}$. This has for constant m the same asymptotics as the runtime of the (1+1) EA, but is different for larger m. For $m = \Omega(n)$, we again obtain $2^{\Omega(m)}$.

Theorem 4. *There is a constant $0 < \kappa < 1/2$ such that under the assumption $rm \leq \kappa n$, the runtime of any unary, unbiased black-box algorithm on* JUMP_m *is at least $\left(\frac{n}{rm}\right)^{cm}$ iterations with probability $1 - 2^{-\Omega(m \ln(n/(rm)))}$ for some constant $c > 0$.*

Proof. Note that $r \leq n/2$ must hold. We reuse terminology and arguments from the proof of Theorem 4. Search points in the region A_2 can be handled in the same way as before because we already have proved an upper bound of $(m/n)^m$ on the probability of obtaining an optimal solution from such a search point.

The main difference of this proof is that we apply Hajek's drift theorem (Lemma 2) for the interval A_3 (and, analogously, also the region A_1). The distance function will be the identity, we choose $b := m$ and $a := 0$ and we will use a λ that is $\Omega(\ln(n/(rm)))$ (note that λ needs not be constant). For $i \leq m$, we derive a better bound on $\mathbf{Pr}[\Delta(i) = -j]$ for $j \geq 1$ compared to Theorem 4. The aim is to bound the expression by $(rm/n)^{\Omega(j)}$. Observe that it is necessary to flip at least j 0-bits. Even when already r bits have been flipped, there are $n - r$ unflipped 1-bits left and the probability of choosing a zero-bit as next flipping bit is at most $m/(n-r)$. If already some 0-bits have been flipped, the probability is only smaller. Note that $m/(n-r) \leq m/(n-n/2) \leq (m/n) \cdot 2\kappa/n < 1$ since $r \leq n/2$ and $\kappa < 1/2$. There are at most $\binom{r}{j}$ ways of flipping the j 0-bits within r trials. Hence,

$$\mathbf{Pr}[\Delta(i) = -j] \leq \binom{r}{j}\left(\frac{2m}{n}\right)^j \leq \left(\frac{2mr}{n}\right)^j \leq e^{-c' j \ln\left(\frac{n}{rm}\right)}$$

for a constant $c' > 0$ that gets the closer to 1 the closer the constant κ is to 0 (note that $\kappa \leq 1/2$ is necessary for $c' > 0$). If we choose $\lambda := (c'/d)\ln(n/(rm))$ for some $d > 1$, we can estimate the contribution of the distance-decreasing steps to the expectation $\mathbf{E}\left[e^{-\lambda(X_{t+1}-X_t)} \mid X_t = i\right]$ as follows:

$$
\begin{aligned}
Q &:= \sum_{j=1}^{\infty} \mathbf{Pr}\left[\Delta(i) = -j\right] \cdot e^{\lambda j} \\
&= \sum_{j=1}^{\infty} e^{-c' j \ln(n/(rm)) + (c'/d) j \ln(n/(rm))} \;=\; \frac{\left(\frac{rm}{n}\right)^{c'(1-1/d)}}{1 - \frac{rm}{n}^{c'(1-1/d)}}.
\end{aligned}
$$

Recalling that c' and d are constants (and c' approaches 1 if κ decreases), we can choose κ so small that the denominator is at least $1/2$. Then $Q \leq 2(rm/n)^{c'(1-1/d)}$.

To complete a proof, we need a good lower bound on $\mathbf{Pr}\left[\Delta(i) \geq 1\right]$. To this end, observe that Markov's inequality (using $\mathbf{E}\left[Z_t\right] = ri/n \leq rm/n$) yields

$$
\begin{aligned}
\mathbf{Pr}\left[\Delta(i) \geq 1\right] = \mathbf{Pr}\left[r - 2Z_t \geq 1\right] \;&=\; \mathbf{Pr}\left[Z_t \leq \frac{r-1}{2}\right] \\
= 1 - \mathbf{Pr}\left[Z_t > \frac{r-1}{2}\right] \;&\geq\; 1 - \frac{ri}{n(r-1)/2} \;\geq\; 1 - \frac{4m}{n}
\end{aligned}
$$

for $r \geq 2$. For $r = 1$, the bound $\mathbf{Pr}\left[\Delta(i) \geq 1\right] = 1 - i/n \geq 1 - m/n$ follows by standard arguments. Altogether, we use the bound $\mathbf{Pr}\left[\Delta(i) \geq 1\right] \geq 1 - 4m/n$ for any r. Furthermore, we pessimistically assume that $\mathbf{Pr}\left[\Delta(i) = 1\right] = 1 - 4m/n$ (the terms for $j > 1$ get a smaller weight in the moment-generating function) and obtain

$$
\begin{aligned}
\mathbf{E}\left[e^{-\lambda(X_{t+1}-X_t)} \mid X_t = i\right] &\leq \frac{2rm}{n} + \left(1 - \frac{4m}{n}\right) \cdot e^{-\lambda} + Q \\
&\leq \frac{2rm}{n} + \left(\frac{rm}{n}\right)^{c'/d} + 2\left(\frac{rm}{n}\right)^{c'(1-1/d)}.
\end{aligned}
$$

We consider the last bound as a function of $rm/n < 1$. In fact, after having fixed a lower bound on c' (recalling that c' grows with decreasing κ) and having chosen a large enough d, a constant small choice of κ bounds the expectation by a constant less than 1.

Bounding $D = \max\left\{1, \mathbf{E}\left[e^{-\lambda \cdot (X_{t+1}-m)} \mid X_t \geq m\right]\right\}$ by $O(1)$ works in the same way. It is enough to use the upper bound on $\mathbf{Pr}\left[\Delta(i) = -j\right]$ for $i = m$ since the probability of making a step of size $j+k$ towards the optimum from a current number of $m+k$ 0-bits cannot be bigger than the probability of making a step of size j from a current number of m 0-bits. Altogether, Lemma 2 bounds the probability of reaching the optimum within some $e^{c \ln(n/(rm)) \cdot m} = (n/(rm))^{cm}$ steps by $2^{-\Omega(\ln(n/(rm)) \cdot m)}$ for a small enough constant $c > 0$. □

We supplement an upper bound that contains the discussed $(n/m)^{O(m)}$ term.

Theorem 5. *There exists a unary, unbiased black-box algorithm whose expected runtime on* $\textsc{Jump}_m$ *is no more than* $O(n \log n + m(en/m)^m)$.

Proof. Our black-box algorithm, called (1+1) EA-RW, contains a hill-climbing component as the (1+1) EA, and a random-walk component. From the best-so-far search point, it creates an offspring by bitwise mutation. If this is an improvement, the best-so-far is updated and nothing else happens. Otherwise, the algorithm will start a random walk from the offspring until a search point with different fitness value is encountered. If this improves on the best-so-far, the best-so-far is updated. This procedure is repeated infinitely.

Algorithm 2 (1+1) EA-RW

```
Choose x uniformly at random.
while true do
  y ← x.
  Flip each bit of y independently with probability 1/n.
  if f(y) ≥ f(x) then
    x ← y
  else
    f* ← f(y).
    repeat
      Flip a bit of y chosen uniformly at random.
    until f(y) ≠ f*.
    if f(y) ≥ f(x) then x ← y.
  end if
end while
```

For the analysis, we reuse the partition of the search space into sets A_1, A_2 and A_3 from the proof of Theorem 3. As long as no search point with $n-m$ 1-bits has been found, the black-box algorithm (1+1) EA-RW behaves like the (1+1) EA. Given that the current search point of the (1+1) EA-RW is in the set A_2, we can use the analysis of the (1+1) EA on ONEMAX by [3], whence it follows that a search point with $n-m$ 1-bits is found in an expected number of $O(n \log n)$ steps in this situation. We will show that the set A_1 and A_3 are left in favour of A_2 after $O(m(en/m)^{m-1})$ steps in expectation, altogether resulting in an expected time $O(n \log n + m(en/m)^{m-1})$ until a search point with $n-m$ 1-bits is found (pessimistically assuming that the optimum is not found before). From that time on, we will analyse the behaviour of the random-walk component (the else branch).

Given a search point x in A_3, let k, $1 \leq k < m$, denote the current number of 0-bits. Note that all search points in A_3 have minimal fitness such that every offspring produced by standard bit mutation is accepted until the set is left. In order to reach the set A_2, it is sufficient to have $m-k$ consecutive steps flipping only a 1-bit. The probability of this joint event is bounded from below by

$$\prod_{i=k}^{m-1} \frac{n-i}{n} \cdot \left(1 - \frac{1}{n}\right)^{n-1} \geq \prod_{i=1}^{m-1} \frac{n-i}{en} \geq \left(\frac{m}{en}\right)^{m-1}$$

using that $n \geq 2m$. The expected number of phases of length $m-k \leq m$

until this event happens is bounded by $(en/m)^{m-1}$, resulting altogether in an expected time of no more than $m(en/m)^{m-1}$ until the set A_2 (or the optimum) is reached. A symmetrical analysis reveals the same bound when starting from the set A_1.

Starting with m 0-bits, the algorithm can make a random walk. The remaining process is divided into phases, each of length $k, 1 \leq k \leq m$, where k is the number of 0-bits by the start of the phase. We focus on the event of having k consecutive steps, each flipping only a 0-bit, where the first step may be made according to bit-wise mutation and the rest flips single bits. The probability of this event is bounded from below by

$$\frac{m}{n}\left(1-\frac{1}{n}\right)^{n-1}\prod_{i=1}^{m-1}\frac{i}{n} \;\geq\; \frac{m!}{en^m} \;\geq\; \frac{(m/e)^m}{en^m} \;=\; \frac{1}{e}\left(\frac{m}{en}\right)^m.$$

Each phase lasts at most m steps, so the expected time until the random-walk component finds the optimum is therefore no more than $em \cdot (en/m)^m$. $\square$

It is an open problem to determine the exact order of growth of the complexity of unary, unbiased black-box algorithms on $\textsc{Jump}_m$. We conjecture $(n/m)^{\Theta(m)}$, i. e., the lower bounds should be improved.

5 General Functions

In the previous sections, we provided bounds on particular pseudo-Boolean functions that are commonly considered in the runtime analysis of randomised search heuristics. In this section, we focus on finding lower bounds that hold for any function. Such bounds are only interesting when we consider functions that correspond to realistic optimisation problems, as trivial functions like constant functions can be optimised with a single function evaluation. We therefore focus on functions that have a unique global optimum.

It is of interest to compare the lower bounds in the black-box models with those bounds that have been obtained for specific EAs. Wegener proved a lower bound of $\Omega(n \log n)$ for the (1+1) EA on any function with a unique optimum [19]. This bound is significantly larger than the $\Omega(n/\log n)$ bound that holds for a generalisation of the ONEMAX problem in the black-box model [4]. Given this discrepancy, one can ask whether there is room to design better EAs which overcome the $n \log n$ barrier, or whether the black-box bound is too loose. Jansen et al. provided evidence that there is little room for improvement by showing that any EA that uses uniform initialisation, selection and bitwise mutation with probability $1/n$ needs $\Omega(n \log n)$ function evaluations to optimise functions with a unique optimum [8].

In the following, we will generalise this result further, showing that the $n \log n$-barrier for functions of a unique optimum even holds for the wider class of unary, unbiased black-box algorithms. The idea behind the proof is to show that the probability of making an improving step reduces as the algorithm approaches the optimum. To implement this idea, we use a proof technique called expected multiplicative weight decrease [14]. It will be helpful to have an estimate of the expectation of a random variable, conditional on the event that this variable takes at least a certain value. We will therefore first provide such an estimate for hypergeometrically distributed random variables.

Lemma 4. *Let Z be a hypergeometrically distributed random variable with parameters n (number of balls), r (number of samples) and m (number of red balls), then for all $k, 0 \leq k \leq r$, $\mathbf{E}[Z \mid Z \geq k] \leq k + (r-k)(m-k)/(n-k)$.*

Proof. The remaining number of trials where red balls can be obtained is maximised if already all of the first k sampled balls were red. Then the number of additionally sampled red balls is denoted by Y and hypergeometrically distributed with parameters $n-k$, $r-k$ and $m-k$. Hence,

$$\begin{aligned}
\mathbf{E}[Z \mid Z \geq k] &= k + \sum_{i=k}^{r} \left(\mathbf{Pr}[Z = i \mid Z \geq k] \cdot \mathbf{E}[Z - k \mid Z = i \geq k] \right) \\
&= k + \sum_{i=k}^{r} \left(\mathbf{Pr}[Z = i \mid Z \geq k] \cdot (i-k) \right) \\
&\leq k + \sum_{i=k}^{r} \left(\mathbf{Pr}[Y = i - k] \cdot (i-k) \right) \\
&= k + \mathbf{E}[Y] \;=\; k + \frac{(r-k)\cdot(m-k)}{n-k}.
\end{aligned}$$

□

We now state the main result of this section.

Theorem 6. *The expected runtime of any unary, unbiased black-box algorithm on any pseudo-Boolean function with a single global optimum is* $\Omega(n \log n)$.

Proof. Without loss of generality, assume that the optimum is the search point 1^n. Since this optimum can easily be obtained from search point 0^n by flipping all bits, the runtime will be bounded by the number of steps until either 1^n or 0^n is sampled for the first time. The potential $\alpha_t n$ of the population in a given iteration t is defined as the shortest Hamming distance from any previously sampled search point to either 1^n or 0^n.

The algorithm will only be charged for steps starting from a potential of $\alpha_0 n > n/3$. By Chernoff bounds, the potential in the initial generation is at least $n/3$ with exponentially high probability. The time to reduce the potential to 0 is estimated using the method of multiplicative weight decrease.

The expected weight decrease after varying a search point x is first estimated conditional on the event that the chosen variation operator p_v flips exactly $r \geq 1$ bits. Conditional on this event, the varied search point x' will be uniformly sampled among all bitstrings that have Hamming distance r to search point x. Assume that the chosen search point x has $\alpha n + cr$ number of 0-bits. In order to obtain a search point with less than αn 0-bits, it is necessary that $0 \leq c < 1$. Let X be the random variable such that the number of 0-bits in the new search point x' is $\alpha n - X$.

We claim that if the current potential is αn for any $\alpha, 0 \leq \alpha < 1/3$, then the expected reduction in potential in one iteration is bounded from above by $O(\alpha)$, independently of r and c. If the claim holds, then the expected potential in the following iteration is $\alpha n \cdot (1 - O(n^{-1}))$, and the expected number of iterations t until the optimum has been found satisfies

$$\begin{aligned}
(n/3) \cdot (1 - O(n^{-1}))^t &< 1 \\
t \cdot \log(1 - O(n^{-1})) &< -\log(n/3) \\
t \cdot O(n^{-1}) &> \log(n/3) \\
t &= \Omega(n \log n).
\end{aligned}$$

We first prove that the claim holds for all r, where $1 \leq r < n/2$. Let random variable Z denote the number of 0-bits that are flipped. The number of 0-bits in search point x' is $\alpha n - X = \alpha n + cr - Z + (r - Z)$, hence $X = 2Z - r(1+c)$. Random variable Z corresponds to the number of red balls obtained after sampling r balls without replacement from an urn containing n balls, where $\alpha n + cr$ of the balls are red. Random variable Z is therefore hypergeometrically distributed with expectation $r \cdot (\alpha + cr/n)$. The potential will only decrease when the new search point x' has at least cr fewer 0-bits than x. The probability of this event is $p_z := \mathbf{Pr}[X \geq 0] = \mathbf{Pr}[Z \geq r(1+c)/2]$. By Lemma 4, when $r < n/2$, the expected reduction in potential equals

$$\begin{aligned}
p_z \cdot \mathbf{E}[X \mid X \geq 0] &= p_z \cdot \mathbf{E}\left[2Z - r(1+c) \mid Z \geq \frac{r(1+c)}{2}\right] \\
&\leq 2p_z \cdot \frac{(r - r(1+c)/2) \cdot (\alpha n + cr - r(1+c)/2)}{n - r(1+c)/2} \\
&\leq p_z \frac{r \alpha n}{n - r} = \alpha p_z r \cdot 1/(1 - r/n) \leq 2 \alpha p_z r.
\end{aligned}$$

For $r \leq n/2$ and $\alpha < 1/3$, Lemma 3 gives

$$p_z = \mathbf{Pr}[Z > r(1+c)/2] \leq \exp(-2t^2 r)$$

where $t = (1+c)/2 - \alpha - cr/n \geq 1/6$. So, the expected weight decrease when $r < n/2$ satisfies $2\alpha r / e^{r/18} = O(\alpha)$.

To prove that the claim holds when $n/2 \leq r < n$, we can use a symmetry in the hypergeometric distribution. Instead of selecting r bit positions to flip, one can select $q := n - r$ bit positions to keep and flip the other bit positions. Assume that the selected search point contains $\alpha n + cq$ 1-bits. Clearly, the constant c is bounded by $c < 1$, otherwise more than αn 1-bits will be flipped into 0-bits, and the potential would not decrease. Let random variable O denote the number of 1-bits selected not to be flipped. The number of 0-bits in search point x' is $\alpha n + cq - O + (q - O)$, hence $X = 2O - q(1+c)$. Random variable O is hypergeometrically distributed, corresponding to the number of red balls obtained after sampling q balls without replacement from an urn containing n balls, where $\alpha n + cq$ of the balls are red. This corresponds exactly to the case when $r < n/2$, where the roles of variables q and O are replaced with r and Z. It therefore follows that the expected decrease in potential is bounded from above by $O(\alpha)$.

The claim now holds for any r and c, and the theorem follows. □

6 Conclusions

This paper takes a step forward in building a unified theory of randomised search heuristics. We have defined a new black-box model that captures essential aspects of randomised search heuristics. The new model covers many of the common search heuristics, including simulated annealing and EAs commonly considered in theoretical studies. We have proved upper and lower bounds on the runtime of several commonly considered pseudo-Boolean functions. For some functions, the lower bounds coincide with the upper bounds for the (1+1) EA, implying that this simple EA is asymptotically optimal on the function class. It is shown that any search heuristic in the model needs $\Omega(n \log n)$ function evaluations to optimise functions with a unique optimum. Also, it is shown that Hamming cliffs pose a difficulty for any black-box search heuristic in the model.

This work can be extended in several ways. Firstly, it is interesting to consider more problem classes than those considered here. Secondly, the analysis should be extended to variation operators with greater arity than one. Finally, alternative black-box models could be defined that cover ant colony optimisation, particle swarm optimisation and estimation of distribution algorithms.

References

[1] V. Chvátal. The tail of the hypergeometric distribution. *Discrete Mathematics*, 25(3):285–287, 1979.

[2] M. Dorigo and T. Stützle. *Ant Colony Optimization.* MIT Press, 2004.

[3] S. Droste, T. Jansen, and I. Wegener. On the analysis of the (1+1) Evolutionary Algorithm. *Theoretical Computer Science*, 276:51–81, 2002.

[4] S. Droste, T. Jansen, and I. Wegener. Upper and lower bounds for randomized search heuristics in black-box optimization. *Theory of Computing Systems*, 39(4):525–544, 2006.

[5] S. Droste and D. Wiesmann. Metric based evolutionary algorithms. In *Proceedings of Genetic Programming, European Conference*, volume 1802 of *LNCS*, pages 29–43. Springer, 2000.

[6] B. Hajek. Hitting-time and occupation-time bounds implied by drift analysis with applications. *Advances in Applied Probability*, 13(3):502–525, 1982.

[7] J. He and X. Yao. A study of drift analysis for estimating computation time of evolutionary algorithms. *Natural Computing*, 3(1), 2004.

[8] T. Jansen, K. A. D. Jong, and I. Wegener. On the choice of the offspring population size in evolutionary algorithms. *Evolutionary Computation*, 13(4):413–440, 2005.

[9] D. S. Johnson, C. H. Papadimitriou, and M. Yannakakis. How easy is local search? *Journal of Computer and System Sciences*, 37(1):79–100, 1988.

[10] J. Kennedy and R. C. Eberhart. *Swarm intelligence.* Morgan Kaufmann Publishers Inc., 2001.

[11] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, May 1983.

[12] P. Larrañaga and J. A. Lozano. *Estimation of distribution algorithms: a new tool for evolutionary computation.* Kluwer Academic Publishers, 2002.

[13] P. K. Lehre and X. Yao. Runtime analysis of (1+1) EA on computing unique input output sequences. In *Proceedings of 2007 IEEE Congress on Evolutionary Computation (CEC'07)*, pages 1882–1889. IEEE Press, 2007.

[14] F. Neumann and I. Wegener. Randomized local search, evolutionary algorithms, and the minimum spanning tree problem. *Theoretical Computer Science*, 378(1):32–40, 2007.

[15] P. S. Oliveto, J. He, and X. Yao. Time complexity of evolutionary algorithms for combinatorial optimization: A decade of results. *International Journal of Automation and Computing*, 4(1):100–106, 2007.

[16] P. S. Oliveto, P. K. Lehre, and F. Neumann. Theoretical analysis of rank-based mutation - combining exploration and exploitation. In *Proceedings of the 10th IEEE Congress on Evolutionary Computation (CEC '09)*, pages 1455–1462. IEEE, 2009.

[17] P. S. Oliveto and C. Witt. Simplified drift analysis for proving lower bounds in evolutionary computation. In *Proceedings of Parallel Problem Solving from Nature (PPSN'X)*, number 5199 in LNCS, pages 82–91, 2008.

[18] J. E. Rowe, M. D. Vose, and A. H. Wright. Neighborhood graphs and symmetric genetic operators. In *Proceedings of Foundations of Genetic Algorithms 9*, number 4436 in LNCS, pages 110–122, 2007.

[19] I. Wegener. Methods for the analysis of evolutionary algorithms on pseudo-Boolean functions. In R. Sarker, M. Mohammadian, and X. Yao, editors, *Evolutionary Optimization*, pages 349–369. Kluwer, 2002.

[20] C. Witt. Runtime analysis of the $(\mu + 1)$ EA on simple pseudo-Boolean functions. *Evolutionary Computation*, 14(1):65–86, 2006.

ECCC ISSN 1433-8092

http://eccc.hpi-web.de