A Study on Performance of the (1+1)-Evolutionary Algorithm [*]

Pavel A. Borisovsky
Omsk Branch of Sobolev
Institute of Mathematics
13, Pevtsov str. 644099, Omsk, Russia
borisovsky@iitam.omsk.net.ru

Anton V. Eremeev
Omsk Branch of Sobolev
Institute of Mathematics
13, Pevtsov str. 644099, Omsk, Russia
eremeev@iitam.omsk.net.ru

Abstract

The first contribution of this paper is a theoretical comparison of the (1+1)-EA evolutionary algorithm to other evolutionary algorithms in the case of so-called monotone reproduction operator, which indicates that the (1+1)-EA is an optimal search technique in this setting. After that we study the expected optimization time for the (1+1)-EA and show two set covering problem families where it is superior to certain general-purpose exact algorithms. Finally some pessimistic estimates of mutation operators in terms of upper bounds on evolvability are suggested for the NP-hard optimization problems.

1 Introduction

In this paper we study the performance of the (1+1)-EA evolutionary algorithm, also known as random hill-climbing algorithm or Metropolis algorithm at zero temperature. Our goal is to compare the (1+1)-EA to other evolutionary and deterministic algorithms and to estimate its average optimization time.

The search process in evolutionary algorithms is usually guided by the selection operators designed in such a way that individuals with greater fitness function values have greater chances to produce the offspring. This principle that "favors" the regions of higher fitness in genotypes space can be easily justified by intuition but the theoretical basis for this way of controlling the population is not completely studied yet.

In view of the "No free lunch" theorem (Wolpert and Macready, 1997) such justification is unlikely without some additional assumptions concerning the fitness function behavior and in fact analysis should encompass both the fitness function and the reproduction operators, i.e crossover and mutation. One of the ways to introduce such additional assumption was framed by Altenberg (1994, 1995) in terms of correlation between the fitness of parents and the probability of the upper tail of the fitness distribution of their offspring. The well-known "big valley" conjecture (Boese, Kahng and Muddu, 1994) has a similar flavor since one of its assumptions states that the better local optima in fitness landscape tend to be closer to the global optimum in some distance measure. A stronger form of such kind of assumption has been proposed in (Eremeev, 2000) as monotonic growth of probability of the upper tail of the offspring fitness distribution as a function on parent's fitness. Another example can be found in (Aldous and Vazirani, 1994), where the rigorous theoretical analysis reveals a significant potential of the "go with the winners" evolutionary algorithms in a situation where the mutation may only increase the parents fitness by one. More versions of assumptions ensuring good performance of the evolutionary algorithms can be found in literature dating back up to three decades (see e.g. (Holland, 1975, Lbov, 1972)).

In this paper we will investigate the implications of the monotonicity assumption which now will be extended from mutation to general reproduction operators with the same basic idea: the probability to obtain the offspring of "high" fitness does not decrease with improvement of the parents fitness.

The Introduction section contains the description of a framework (Eremeev, 2000) for evolutionary algorithm analysis which is based on the usual Markov chain approach with grouping of states. As an illustration of the monotone mutation concept here we show that the black-box complexity class $ONEMAX^{**}$ defined in (Droste, Jansen, Tinnefeld and Wegener, 2002) may be alternatively characterized through monotonicity of the standard mutation operator. In Sect. 2 we prove that the (1+1)-EA is the optimal search technique in a wide class of evolutionary algorithms in case of so-called monotone reproduction operator and show that the reproduction operator can then be reduced to mutation only. The subsequent analysis is concentrated on the time complexity of the (1+1)-EA. In Sect. 3 we obtain some upper bounds on the expected (1+1)-EA optimization time using the standard probability theory techniques and the well-known method of waiting times until a new fitness level is reached. Based on one of these upper bounds in Sect. 4 we construct a pessimistic estimate valid for any polynomial-time mutation operator for polynomially bounded NP-hard optimization problem, imposing the upper bounds on *evolvability* of some individuals (the details on the notions of evolvability and its role in evolutionary algorithms analysis can be found in (Altenberg, 1994)). The latter result is not a novelty but an instructive application of the common knowledge from complexity theory to the evolutionary algorithms analysis.

1.1 Notation and Assumptions

Let the optimization problem consist in finding a feasible solution $y \in Sol \subseteq D$, which maximizes the objective function $f : Sol \to \mathbf{R}$, where D is the *space of solutions*, $Sol \subseteq D$

is a *set of feasible solutions* and $\mathbf{R}$ is the set of real numbers. In general an evolutionary algorithm is searching for the optimal or near-optimal solutions using a population of individuals, which is driven by the principles observed in biological evolution. The (1+1)-EA mainly considered in this paper is a simplified version of the evolutionary algorithm where the population consists of a single individual.

We assume that an individual is represented by a *genotype*, which is a fixed length string g of genes $g_1, g_2, \ldots, g_n$, and all genes are the symbols of some finite alphabet A. For example, the alphabet $\Sigma = \{0, 1\}$ is used in many applications. Each genotype g represents an element $y = y(g)$ of space D, which may not necessarily be a feasible solution.

The search process is guided by evaluations of nonnegative *fitness function* $\Phi(g)$, which defines the fitness of an individual with genotype g. In case $y(g) \in Sol$, it is supposed that Φ is a monotone function of $f(y(g))$. In case $y(g) \notin Sol$, the fitness function may incorporate a penalty for violation of constraints defining the set Sol. Through this paper we will assume that if $Sol \neq D$ then for any g' such that $y(g') \in D \backslash Sol$ holds $\Phi(g') < \max_{g \in A^n} \Phi(g)$.

The genotype of the current individual on iteration t of the (1+1)-EA will be denoted by $g^{(t)}$. The initial genotype $g^{(0)}$ is generated with some a priori chosen probability distribution. Each new individual is built with the help of a random *mutation operator* $\mathcal{M} : A^n \to A^n$, which adds some random changes to the parent genotype. The mutation operator is applied to $g^{(t)}$ and if $g = \mathcal{M}(g^{(t)})$ is such that $\Phi(g) > \Phi(g^{(t)})$, then we set $g^{(t+1)} := g$; otherwise $g^{(t+1)} := g^{(t)}$. The stopping criterion is usually the limit on the maximum number of iterations.

In this paper the mutation operator will be viewed as a randomized algorithm which has an input genotype g and a random output $\mathcal{M}(g) \in A^n$ with probability distribution depending on the input genotype and the specific data of the given problem instance. So the data of the problem instance may be considered as a part of the input of mutation. An r-parent reproduction operator in general will be treated analogously – its description will be given in Sect. 2. One of the frequently used mutation operators is the *standard mutation*, which consists in changing each gene of g with a fixed mutation probability p_{mut}. Another simple example is the *1-bit-flip* mutation operator which chooses a random position i and replaces the gene g_i by a new symbol (see e.g. (Rudolph, 1998)). In general the reproduction operator may be a much more complicated problem specific randomized heuristic including recombination, mutation and local improvement heuristics. It may involve random choices dependent on the "logics" of the input problem like in (Schöning, 1999) and the repair heuristics analogous to those of Beasley and Chu (1996).

The analysis of the (1+1)-EA in principle could be carried out by the means of Markov chains theory (see e.g. (Rudolph, 1998)). However, the size of the transition matrix of a Markov chain grows exponentially as the genotype length increases, and the applicability of this approach appears to be limited when studying the optimization problems with large cardinality of solutions space. In order to overcome this difficulty we use the grouping of the states into larger classes on the basis of fitness.

Assume that there are d level lines of the fitness function fixed so that $\Phi_0 = 0 < \Phi_1 < \Phi_2 \ldots < \Phi_d$. The number of the level lines and the fitness values corresponding to them may be chosen arbitrarily, but they should be relevant to the given problem and the mutation operator to yield a meaningful model. Let us introduce the following sequence of subsets of the set A^n:

$$H_i = \{g : \Phi(g) \geq \Phi_i\}, \quad i = 0, \ldots, d.$$

Due to the nonnegativity of the fitness function, H_0 equals the set of all genotypes.

The distribution of the current individual in the (1+1)-EA will be characterized (though not completely) by the *vector of probabilities*

$$Q^{(t)} = (q_1^{(t)}, ..., q_d^{(t)}) = (P\{g^{(t)} \in H_1\}, \ldots, P\{g^{(t)} \in H_d\}),$$

which reflects the chances to have "good enough" genotypes on iteration t.

Now suppose that for all $i = 0, ..., d$ and $j = 1, ..., d$ the a priori lower bounds α_{ij} on mutation transition probability from subset $H_i \backslash H_{i+1}$ to H_j are known, i.e. for every $g \in H_i \backslash H_{i+1}$ holds $\alpha_{ij} \leq P\{\mathcal{M}(g) \in H_j\}$, where $P\{\mathcal{M}(g) \in H_j\} = \sum_{g' \in H_j} P\{\mathcal{M}(g) = g'\}$. Here for convenience we assume that $H_{d+1} = \emptyset$ by definition. Let $\mathbf{A}$ denote the matrix with elements α_{ij} where $i = 0, ..., d$, and $j = 1, ..., d$.

If for all $i = 0, \ldots, d$, and $j = 1, \ldots, d$ the probability $P\{\mathcal{M}(g) \in H_j\}$ does not depend on the choice of $g \in H_i \backslash H_{i+1}$ then there exists a matrix $\mathbf{\Gamma}$ with $\gamma_{ij} = P\{\mathcal{M}(g) \in H_j\}$, $i = 0, ..., d$, and $j = 1, ..., d$, called *the threshold transition matrix.*

A matrix $\mathbf{A}$ will be called *monotone* if $\alpha_{i-1,j} \leq \alpha_{ij}$ for all i, j from 1 to d. In other words, the matrix of bounds on transition probabilities is monotone if for any $j = 1, \ldots, d$, the genotypes from any subset H_i have the bounds on transition probabilities to H_j not less than the bounds of the genotypes from any $H_{i'}$, $i' < i$.

Obviously, for any mutation operator the monotone lower bounds exist (for example $\mathbf{A} = \mathbf{0}$ where $\mathbf{0}$ is a zero matrix). The problem may be only with the absence of bounds which are sharp enough to evaluate the mutation operator properly.

Definition 1 *A mutation operator is monotone with respect to a set of levels* $\Phi_0, \Phi_1, ..., \Phi_d$ *if there exists a threshold transition matrix* $\mathbf{\Gamma}$ *for it and this matrix is monotone.*

In general $\mathcal{M}$ will be called monotone if it is monotone with respect to some set of levels. Three examples of situations where mutation is monotone in solving some set covering problems will be described in Sect. 3. Another interesting example of monotone mutation is discussed below.

1.2 Class of functions $ONEMAX^{**}$ and monotone mutation

Droste, Jansen, Tinnefeld and Wegener (2002) introduce a number of problem classes generalizing some of the well-known functions such as $ONEMAX$ function, number of the leading ones function, needle in the haystack function etc., and suggest the lower and upper bounds on the complexity of these classes for the black box optimization algorithms. Now we will see that one of these classes may be alternatively defined in terms of monotone mutation.

First of all $ONEMAX^*$ is defined as the class of functions

$$ONEMAX_a(x) = \sum_{i=1}^{n} ((x_i + a_i) \bmod 2),$$

where the argument x and the parameter a belong to $\{0,1\}^n$. A more general class $ONEMAX^{**}$ by definition consists of all functions $\psi \circ f$ where $f \in ONEMAX^*$ and $\psi : \mathbf{R} \to \mathbf{R}$ is a strictly increasing function. Obviously function $ONEMAX$ is an element of $ONEMAX^{**}$ with $a = (0, 0, ...0)$ and an identity mapping ψ.

Let us denote the set of all values of the fitness function Φ on the set of genotypes by $Range(\Phi)$, i.e. $Range(\Phi) = \{\Phi(g) : g \in A^n\}$.

Proposition 1 *Let $\mathcal{M}$ be the standard mutation operator with $p_{mut} < 1/2$ and $A = \{0,1\}$. Assume there is a unique maximum g^* of $\Phi(g)$ on A^n and $|Range(\Phi)| = n+1$. Then $\mathcal{M}$ is monotone iff $\Phi \in ONEMAX^{**}$.*

Proof. Naturally here we will set $d = n$, $\{\Phi_0, ..., \Phi_n\} = Range(\Phi)$ and consider the threshold transition matrix $\mathbf{\Gamma}$ with elements γ_{ij}, $i = 0, ..., d$, $j = 1, ..., d$.

In case $\Phi \in ONEMAX^{**}$, the values of the fitness function monotonically increase with the number of the correctly chosen genes and by Prop. 5 (Eremeev, 2000) the monotonicity of $\mathcal{M}$ follows.

To prove the statement in the other direction first of all note that by assumption $\Phi_n = \max_{g \in A^n} \Phi(g)$ and $H_n = \{g^*\}$.

Now we will see that no $0 \le i < n$ exists such that $\gamma_{in} = \gamma_{i+1,n}$. Indeed, since $H_n = \{g^*\}$, so for all $g \in H_i \backslash H_{i+1}$, we have $P\{\mathcal{M}(g) = g^*\} = \gamma_{i,n}$. Now there are exactly $n+1$ different values for

$$P\{\mathcal{M}(g) = g^*\} = p_{mut}^{\delta(g,g^*)}(1-p_{mut})^{n-\delta(g,g^*)}, \quad (1)$$

where $\delta(g, g^*)$ is the Hamming distance between g and g^*. So if we suppose that for some $0 \le i < n$ holds $\gamma_{in} = \gamma_{i+1,n}$ then not all $n+1$ possible values of $P\{\mathcal{M}(g) = g^*\}$ are present among $\gamma_{0n}, \gamma_{1n}, ..., \gamma_{nn}$. This would imply that for some $0 \le j < n$ there exist $g \in H_j \backslash H_{j+1}$ and $g' \in H_j \backslash H_{j+1}$ such that $\gamma_{jd} = P\{\mathcal{M}(g) = g^*\} \neq P\{\mathcal{M}(g') = g^*\} = \gamma_{jd}$ which is impossible. So $\gamma_{in} < \gamma_{i+1,n}$ due to monotonicity of $\mathcal{M}$ and each subset $H_i \backslash H_{i+1}$ consists of genotypes with equal Hamming distance to g^*.

From (1) we conclude that for all $0 \le i \le n$ a genotype g belongs to $H_i \backslash H_{i+1}$ iff $\delta(g, g^*) = n - i$. Consequently $\Phi(g)$ is a monotonically decreasing function of $\delta(g, g^*)$, i.e. $\Phi \in ONEMAX^{**}$. □

The requirement for the uniqueness of the global optimum cannot be simply omitted in this proposition, because outside $ONEMAX^{**}$ the functions with several optima exist satisfying the rest of conditions of Prop 1. In case $n = 3$ e.g. one of such functions is $\Phi(g) = \min\{k(g), 7 - k(g)\}$, where $k(g) = g_1 + 2g_2 + 4g_3$.

2 The (1+1)-EA compared to other evolutionary algorithms

In order to define the general scheme of an evolutionary algorithm we will assume that the reproduction operator $\mathcal{R}$ is a randomized algorithm which has a set of genotypes $a^1, a^2, ..., a^r$ on its input and computes a random output $\mathcal{R}(a^1, a^2, ..., a^r) \in A^{ns}$ i.e. s

offspring genotypes. The probability distribution of the output depends on the input genotypes and the specific data of the problem being solved.

Let us consider an evolutionary algorithm EA which corresponds to the following scheme: the initial set of genotypes $a^{(0,1)}, \dots, a^{(0,N)}$ is given, and on each iteration t a new group of genotypes $a^{(t,1)}, \dots, a^{(t,s)}$ is produced by applying $\mathcal{R}(b^1, \dots, b^r)$ where each $b^k, k = 1, ..., r$ is some genotype which had been already generated before (i.e. $b^k \in \mathcal{A}^{(t-1)}$ where $\mathcal{A}^{(t-1)} = \{a^{(0,l)} : l = 1 \dots, N\} \cup \{a^{(\tau,j)} : \tau = 1, \dots, t-1, j = 1, \dots, s\}$). It is easy to see that most of all of the evolutionary algorithms such as the genetic algorithms (Holland, 1975), "go with the winners" algorithms (Aldous and Vazirani, 1994), many versions of genetic programming algorithms (Koza, 1992), (μ, λ)-EA and $(\mu + \lambda)$-EA – see e.g. (Bäck, 1993) satisfy this scheme. Due to the finiteness of set A^n we can fix the set of all level lines $\{\Phi_0, \dots, \Phi_d\} = Range(\Phi)$. Denote by $\tilde{a}^{(t)}$ the best genotype in set $\mathcal{A}^{(t)}$ and let $P^{(t)}$ be the vector of probabilities for $\tilde{a}^{(t)}$: $P^{(t)} = (P\{\tilde{a}^{(t)} \in H_1\}, \dots, P\{\tilde{a}^{(t)} \in H_d\})$.

Let $\tilde{a}_{\mathcal{R}}(b^1, \dots, b^r)$ denote the best genotype among all s genotypes generated by $\mathcal{R}$ on input $b^1, \dots, b^r$. Now the definition of monotonicity has to be generalized for arbitrary reproduction procedure. Informally this new definition will require that substitution of parent genotypes by genotypes with greater or equal fitness should never decrease the components of vector of probabilities:

Definition 2 *Operator $\mathcal{R}$ is monotone if for arbitrary r-element sets of genotypes $b^1, \dots, b^r$ and $h^1, \dots, h^r$ such that*

$$\Phi(b^1) \le \Phi(h^1), ..., \Phi(b^r) \le \Phi(h^r) \tag{2}$$

the following conditions hold for all $j = 1, \dots, d$:

$$P\left\{\Phi(\tilde{a}_{\mathcal{R}}(b^1, \dots, b^r)) \ge \Phi_j\right\} \le P\left\{\Phi(\tilde{a}_{\mathcal{R}}(h^1, \dots, h^r)) \ge \Phi_j\right\}. \tag{3}$$

Note that this definition implies that if all conditions (2) are equalities for the sets of parents $b^1, ..., b^r$ and $h^1, ..., h^r$ then the probability distributions of $\Phi(\tilde{a}_{\mathcal{R}}(b^1, \dots, b^r))$ and $\Phi(\tilde{a}_{\mathcal{R}}(h^1, \dots, h^r))$ must coincide. Obviously the monotone mutation operator defined in Sect. 1 is a special case of monotone reproduction operator with $r = s = 1$.

Let us consider now a simple example of monotone reproduction and postpone another example till the end of Sect. 3. Suppose we have an arbitrary function $\Phi \in ONEMAX^{**}$ and $\mathcal{R}$ is the standard 1-point or uniform crossover operator, but the genes in one of the parent genotypes are randomly permuted before the crossover. It is not difficult to see that in this case $\mathcal{R}$ is monotone and it would remain monotone if after this crossover the 1-bit flip mutation were applied.

Sometimes in our analysis the monotonicity condition may be relaxed in the following way. We will call $\mathcal{R}$ *weakly monotone* if the inequality (3) holds at least for all j such that $\Phi_j > \max\{\Phi(h^k) : k = 1, \dots, r\}$.

Let us define a one-parent mutation operator corresponding to $\mathcal{R}$ as

$$\mathcal{M}_{\mathcal{R}}(g) = \text{ argmax } (\Phi(g), \Phi(g')),$$

where $g' = \tilde{a}_{\mathcal{R}}(g, \dots, g)$, i.e. in $\mathcal{M}_{\mathcal{R}}(g)$ firstly the reproduction $\mathcal{R}$ is applied to a set of identical parent genotypes and then the output is chosen as the fittest among the parent and the offspring.

Now we can formulate the main result of this section:

Theorem 2 *Suppose that a monotone reproduction operator $\mathcal{R}$ is used in the algorithm EA and the operator $\mathcal{M}_{\mathcal{R}}$ is used in the (1+1)-EA. Let the algorithm (1+1)-EA always start from the best genotype among $a^{(0,1)}, \dots, a^{(0,N)}$. Then for all $t \geq 0$*

$$P^{(t)} \leq Q^{(t)}. \tag{4}$$

Here and below the vectors are compared coordinate-wise, i.e. (4) is equivalent to the set of inequalities:

$$P\{\tilde{a}^{(t)} \in H_1\} \leq P\{g^{(t)} \in H_1\}, ..., P\{\tilde{a}^{(t)} \in H_d\} \leq P\{g^{(t)} \in H_d\}.$$

It is clear that if $\mathcal{R}$ is monotone then $\mathcal{M}_{\mathcal{R}}$ is monotone too. The following natural property of $\mathcal{M}_{\mathcal{R}}$ will be useful for us.

Lemma 3 *If $\mathcal{M}_{\mathcal{R}}$ is monotone and g and h are the random genotypes distributed so that for all $i = 1, \dots, d$ it holds that $P\{g \in H_i\} \leq P\{h \in H_i\}$, then for all $j = 1, \dots, d$*

$$P\{\mathcal{M}_{\mathcal{R}}(g) \in H_j\} \leq P\{\mathcal{M}_{\mathcal{R}}(h) \in H_j\}. \tag{5}$$

Proof. By means of the total probability formula and Abel transform we see that

$$P\{\mathcal{M}_{\mathcal{R}}(g) \in H_j\} = \sum_{i=0}^{j-1} \gamma_{ij} P\{g \in H_i \setminus H_{i+1}\} + P\{g \in H_j\} =$$

$$\sum_{i=0}^{j-1} \gamma_{ij} \left(P\{g \in H_i\} - P\{g \in H_{i+1}\}\right) + P\{g \in H_j\} =$$

$$\gamma_{0j} + \sum_{i=1}^{j-1} (\gamma_{ij} - \gamma_{i-1,j}) P\{g \in H_i\} + (1 - \gamma_{j-1,j}) P\{g \in H_j\}.$$

The probability $P\{\mathcal{M}_{\mathcal{R}}(h) \in H_j\}$ may be represented similarly, and from the monotonicity of $\mathcal{M}_{\mathcal{R}}$ it follows that $\gamma_{ij} - \gamma_{i-1,j} \geq 0$ so inequality (5) holds. □

Proof of Theorem 2. For $t = 0$ inequality (4) is obvious. By induction on t assume that $P^{(t-1)} \leq Q^{(t-1)}$. We have $\tilde{a}^{(t)} = \text{ argmax } \{\Phi(a'), \Phi(\tilde{a}^{(t-1)})\}$, where $a' = \tilde{a}_{\mathcal{R}}(b^1, \dots, b^r)$ and each b^k is chosen from $\mathcal{A}^{(t-1)}$ somehow. Then for any $j = 0, \dots, d$ it holds:

$$P\{\tilde{a}^{(t)} \in H_j\} = P\{a' \in H_j \text{ or } \tilde{a}^{(t-1)} \in H_j\} =$$

$$P\{\tilde{a}^{(t-1)} \in H_j\} + P\{a' \in H_j, \tilde{a}^{(t-1)} \notin H_j\}.$$

Let us apply the total probability formula to the last summand:

$$P\{a' \in H_j, \tilde{a}^{(t-1)} \notin H_j\} =$$

$$\sum_{i=0}^{j-1} P\{a' \in H_j | \tilde{a}^{(t-1)} \in H_i \setminus H_{i+1}\} P\{\tilde{a}^{(t-1)} \in H_i \setminus H_{i+1}\}.$$

The monotonicity of operator $\mathcal{R}$ and the definition of $\mathcal{M}_{\mathcal{R}}$ yield:

$$P\{a' \in H_j | \tilde{a}^{(t-1)} \in H_i \setminus H_{i+1}\} \leq$$

$$P\{\tilde{a}_{\mathcal{R}}(\tilde{a}^{(t-1)}, \ldots, \tilde{a}^{(t-1)}) \in H_j | \tilde{a}^{(t-1)} \in H_i \setminus H_{i+1}\} \leq$$

$$P\{\mathcal{M}_{\mathcal{R}}(\tilde{a}^{(t-1)}) \in H_j | \tilde{a}^{(t-1)} \in H_i \setminus H_{i+1}\}$$

(recall that $\mathcal{M}_{\mathcal{R}}$ returns the best of input and mutated genotypes). Thus

$$P\{\tilde{a}^{(t)} \in H_j\} \leq P\{\tilde{a}^{(t-1)} \in H_j\}+$$

$$+\sum_{i=0}^{j-1} P\{\mathcal{M}_{\mathcal{R}}(\tilde{a}^{(t-1)}) \in H_j | \tilde{a}^{(t-1)} \in H_i \setminus H_{i+1}\} P\{\tilde{a}^{(t-1)} \in H_i \setminus H_{i+1}\} =$$

$$P\{\mathcal{M}_{\mathcal{R}}(\tilde{a}^{(t-1)}) \in H_j\}.$$

Now using the inductive assumption and the claim of Lemma 3 we obtain (4). □

It is easy to see that if $\mathcal{R}$ is weakly monotone then $\mathcal{M}_{\mathcal{R}}$ is still monotone. Thus the following corollary holds.

Corollary 4 *The monotonicity condition on $\mathcal{R}$ in Theorem 2 may be relaxed to weak monotonicity.*

The monotone reproduction operators appear to be a convenient theoretical construction since for any given problem and operator $\mathcal{R}$, a monotone reproduction operator $\mathcal{R}'$ can be considered with the transition probabilities $P\{\Phi(\tilde{a}_{\mathcal{R}'}(b^1, \ldots, b^r)) \geq \Phi_j\}$ equal to some monotone lower bounds on $\mathcal{R}$ but here the definition of the lower bounds has to be understood in more general sense[1] than the bounds for mutation operator in Sect. 1.

Then $\mathcal{R}'$ may be viewed as a realization of the worst-case situation for a given set of lower bounds. In this respect, Theorem 2 shows that the best possible lower bound on $P^{(t)}$ for the EA within our framework is not better than the best lower bound on $Q^{(t)}$ for the (1+1)-EA with the corresponding mutation operator $\mathcal{M}_{\mathcal{R}}$.

Note that Theorem 2 could be generalized to a continuous case where instead of discrete genotypes the elements of some continuous search space D are used. Instead of vectors $P^{(t)}$ and $Q^{(t)}$ then we would compare the corresponding families of tails of distributions,

[1]Let $I = (i_1, ..., i_r)$ be the vector of integers, and $b = (b^1, \ldots, b^r) \in A^{nr}$. Then by $b \in H_I \setminus H_{I+1}$ we mean that $b^1 \in H_{i_1} \setminus H_{i_1+1}, ..., b^r \in H_{i_r} \setminus H_{i_r+1}$. Now we may introduce the lower bounds as

$$\alpha_{Ij} \leq P\{\tilde{a}_{\mathcal{R}}(b) \in H_j\},\ I \in \{0, ..., d-1\}^r,\ j = 1, ..., r,$$

for all $b \in H_I \setminus H_{I+1}$. Then we have a partial order: $I \preceq I'$ iff $i_1 \leq i'_1, ..., i_r \leq i'_r$. The set of bounds $\{\alpha_{Ij}\}$ will be called monotone if $I \preceq I'$ implies that $\alpha_{Ij} \leq \alpha_{I'j}$ for all $I \in \{0, ..., d-1\}^r,\ j = 1, ..., r$.

besides that the summations in the proofs of the theorem and lemma would have to be turned into integrals (see e.g. (Borovkov, 1998); p.84). In order to ensure legitimacy of integration by parts (instead of Abel transformation) one can impose some practically insignificant conditions on operator $\mathcal{R}$.

3 Expected hitting time for the optimum

In this section we will consider the average number of iterations the (1+1)-EA spends in search of the optimal genotype. Of course the optimal solution to the problem may not be represented in the set of all genotypes, however for simplicity in this section we will assume that $\max\{f(y(g)) : g \in A^n, y(g) \in Sol\} = \max_{y\in Sol} f(y)$.

Let us denote by t_j the expected number of iterations until level j or greater is reached, i.e. the expected hitting time of H_j and put $T = (t_1, t_2, ..., t_d)$.

Given bounds matrix $\mathbf{A}$ let us introduce the following matrix $\mathbf{W}$ with elements $w_{ij}, i = 1, ..., d, j = 1, ..., d$ and vector α:

$$w_{ij} = \begin{cases} \alpha_{i,j} - \alpha_{i-1,j}, & i < j \\ 1 - \alpha_{j-1,j}, & i = j \\ 0, & i > j, \end{cases} \qquad \alpha = (\alpha_{01}, \dots, \alpha_{0d}).$$

By reasoning similar to Theorem 5 in (Eremeev, 2000) for arbitrary matrix norm $||\cdot||$ (see e.g (Lankaster, 1969)) we obtain:

Proposition 5 *If* $\mathbf{A}$ *is monotone and* $||\mathbf{W}^t|| \overset{t\to\infty}{\longrightarrow} 0$ *then*

$$Q^{(t)} \geq Q^{(0)}\mathbf{W}^t + \alpha(\mathbf{I} - \mathbf{W})^{-1}(\mathbf{I} - \mathbf{W}^t), \tag{6}$$

which is an equality in case $\mathcal{M}$ *is monotone with* $\mathbf{\Gamma} = \mathbf{A}$.

For example it can be shown that $||\mathbf{W}^t|| \overset{t\to\infty}{\longrightarrow} 0$ for any matrix norm if $\alpha_{i,i+1} > 0, i = 1, 2, ..., d-1$, which is true for many mutation operators. Note that the right-hand part of (6) tends to $\alpha(\mathbf{I} - \mathbf{W})^{-1}$ when $t \to \infty$ and by definition of $\mathbf{W}$ and α it follows that $\alpha(\mathbf{I} - \mathbf{W})^{-1} = \mathbf{1}$. Thus we conclude that for convergence of the (1+1)-EA (in probability) to the optimum it suffices that operator $\mathcal{M}$ have a non-zero probability of improvement in every non-optimal genotype (obviously one can easily guarantee this using the traditional Markov chain approach – see e.g. (Rudolph, 1998)). The next theorem provides a qualitative estimate of the average time till convergence to optimal and near-optimal solutions.

Theorem 6 *If* $\mathbf{A}$ *is monotone and* $||\mathbf{W}^t|| \overset{t\to\infty}{\longrightarrow} 0$ *then*

$$T \leq (\mathbf{1} - Q^{(0)})(\mathbf{I} - \mathbf{W})^{-1} \tag{7}$$

which is an equality in case $\mathcal{M}$ *is monotone with* $\mathbf{\Gamma} = \mathbf{A}$.

Proof. Using the properties of expectation (see e.g. (Borovkov, 1998); Chap.4, §4) we conclude that $t_j = \sum_{t=0}^{\infty}(1 - q_j^{(t)}), j = 1, ..., d$ and thus

$$T = \sum_{t=0}^{\infty}(\mathbf{1} - Q^{(t)}) \leq \sum_{t=0}^{\infty}(\mathbf{1} - Q^{(0)}\mathbf{W}^t - \alpha(\mathbf{I} - \mathbf{W})^{-1}(\mathbf{I} - \mathbf{W}^t))$$

Since $\alpha(\mathbf{I} - \mathbf{W})^{-1} = \mathbf{1}$ so

$$T \leq \sum_{t=0}^{\infty}(\mathbf{1} - Q^{(0)}\mathbf{W}^t - \mathbf{1}(\mathbf{I} - \mathbf{W}^t)) = (\mathbf{1} - Q^{(0)})(\mathbf{I} - \mathbf{W})^{-1}. \quad \square$$

Bound (7) has a simple matrix form but in its applications the necessary lower bounds on mutation operator may be hard to find. The next more simple estimate is based on the idea of summing the waiting times until a new fitness level is reached – this idea has been used already in the similar bounds by Bäck (1993), Garnier, Kallel and Schoenauer (1999) and Mühlenbein (1993).

Proposition 7 *The expected hitting time t_d of set H_d for the (1+1)-EA is estimated as follows:*

$$t_d \leq \sum_{i=0}^{d-1} \frac{1}{\alpha_{i,i+1}}.$$

Let t^* denote the expected number of iterations of the (1+1)-EA till the optimum is reached and consider the 1-bit-flip mutation operator.

For function $\Phi \in ONEMAX^{**}$ we naturally assume $d = n$ and $\{\Phi_0, ..., \Phi_n\} = Range(\Phi)$. Then it is easy to see that for any genotype $g \in H_i \backslash H_{i+1}$ holds $P\{\Phi(\mathcal{M}(g)) \geq \Phi_{i+1}\} = 1 - i/n$. So by Prop. 7 we have

$$t_d \leq n \sum_{i=0}^{n-1} \frac{1}{n-i} = n \sum_{k=1}^{n} \frac{1}{k} \leq n + n \int_1^n \frac{dx}{x},$$

which leads to a corollary analogous to the well-known result for $ONEMAX$ function (see e.g. (Garnier, Kallel and Schoenauer, 1999, Rudolph, 1998)):

Corollary 8 *For the (1+1)-EA with 1-bit-flip mutation operator applied to a function $\Phi \in ONEMAX^{**}$ holds $t^* \leq n(1 + \ln n)$, where n is the string length.*

A similar approach can be applied to some families of covering problems with regular structure. Let us first consider a family of the set covering problems (SCP) suggested by Balas (1984). In general the SCP is NP-hard and can be formulated as follows.

Given: $M = \{1, ..., \mathbf{m}\}$ and a set of subsets $M_j \subseteq M$, $j \in N = \{1, \ldots, \mathbf{n}\}$. A subset $J \subseteq N$ is called a *cover* if $\mathbf{U}_{j \in J} M_j = M$. The goal is to find a cover of minimum cardinality.

Assume that $N_i = \{j : i \in M_j\}$. In the Balas SCP family $B(\mathbf{n}, \mathbf{p})$ it is assumed that $m = C_{\mathbf{n}}^{\mathbf{p}-1}$ and the set $\{N_1, N_2, ..., N_{\mathbf{m}}\}$ consists of all $\mathbf{n} - \mathbf{p} + 1$-element subsets of N. Thus $J \subseteq N$ is an optimal cover iff $|J| = \mathbf{p}$.

Suppose the *binary representation* of solutions (Beasley and Chu, 1996) is used, i.e. the genes $g_j \in \{0, 1\}, j \in N$ are the indicators of the elements from N, so that $y(g) = \{j \in N : g_j = 1\} \subseteq N$. If $y(g)$ is a cover then we assign its fitness $\Phi(g) = \mathbf{n} - |y(g)|$; otherwise $\Phi(g) = 0$. Setting $d = \mathbf{n} - \mathbf{p}$ and $\Phi_0 = 0, \Phi_1 = 1, ..., \Phi_d = d$ it is easy to see that $\alpha_{i,i+1} = 1 - i/\mathbf{n}$ again and by Prop. 7

$$t^* = t_{\mathbf{n}-\mathbf{p}} \le \mathbf{n} \sum_{k=\mathbf{p}+1}^{\mathbf{n}} \frac{1}{k} \le \mathbf{n} \int_{\mathbf{p}}^{\mathbf{n}} \frac{dx}{x}.$$

Corollary 9 *Let the (1+1)-EA with binary representation and 1-bit-flip mutation operator be applied to a problem from set covering family $B(\mathbf{n}, \mathbf{p})$. If the initial genotype encodes a cover then $t^* \le \mathbf{n} \ln(\mathbf{n}/\mathbf{p})$.*

In the instances of this family dimension $\mathbf{m}$ may be exponential in $\mathbf{n}$ (depending on behavior of the parameter $\mathbf{p}$), which can make the fitness evaluation procedure for SCP exponential in $\mathbf{n}$ as well. The following family of problems is free from this nuisance.

A family of set covering instances $G(k)$ consists of problems with $\mathbf{n} = \mathbf{m} = 3k$ where $|N_i| \equiv |M_j| \equiv 2$ for all $i \in M, j \in N$ and all cycles consist of 3 elements (by cycle here we mean a sequence $j(1) < \ldots < j(\nu)$, such that $M_{j(\theta)} \cap M_{j(\theta+1)} \ne \emptyset, \theta = 1, ..., \nu - 1$ and $M_{j(\nu)} \cap M_{j(1)} \ne \emptyset$). In other words we have a graph of k disjoined cliques of size 3 where set N is the set of vertices and M is the set of edges. The problem consists in selection of such subset $J \subseteq N$ that each edge has at least one endpoint in J. Obviously the optimal solution is to pick a couple of vertices from each clique.

Let us consider the *non-binary representation* (see e.g. (Beasley and Chu, 1996, Eremeev, 1999)) of the set covering problem solutions, where each gene selects one of the subsets to cover the corresponding element of M, i.e. $g_i \in N_i, i = 1, ..., \mathbf{m}$. A collection of elements defined by genotype g is $y(g) = \{j \in N : g_i = j \text{ for some } i \in M\}$, and it is always a cover. Let us assume that $\Phi(g) = \mathbf{n} - |y(g)|$ again and $d = k, \Phi_i = i, i = 0, ..., k$. Then analogously to Corollary 8 follows

Corollary 10 *For the (1+1)-EA with non-binary representation and 1-bit-flip mutation operator applied to a problem from the set covering family $G(k)$ holds $t^* = t_d \le k(1 + \ln k)$.*

It is interesting that, as shown by Zaozerskaya (1998), the SCP families $B(\mathbf{n}, \mathbf{p})$ and $G(k)$ in integer linear programming formulation are hard for the Land and Doig branch and bound algorithm (see e.g. (Schrijver, 1986); Chapt. 24) and for the L-class enumeration algorithm of Kolokolov (1996). Both of these exact algorithms make an exponential in $\mathbf{n}$ number of iterations, e.g. on the problems from $G(k)$ the Land and Doig algorithm requires 2^{k+1} branchings.

Note that both families of set covering problems discussed here may be considered as examples of monotonicity of 1-bit-flip mutation, although in case of $B(\mathbf{n}, \mathbf{p})$ this mutation operator is only weakly monotone. Besides that on family $G(k)$ the standard mutation operator with $p_{mut} < 1/2$ using the non-binary representation and the fitness function described above is monotone as well (Eremeev, 2000).

Finally let us briefly mention another example of a reproduction operator with $r = 2, s = 1$ which is monotone on the SCP family $G(k)$. This operator is a simplified version of the one used in a memetic genetic algorithm proposed by Eremeev (1999) but without the greedy heuristics of local improvement. The so-called *LP-crossover* used here is a deterministic operator aimed to find the best possible combination of the subsets given in the parent genotypes a_1 and a_2. So a problem of the optimal crossover is considered, which is a reduced version of the initial SCP but with the covering subsets restricted to the subsets present in parents. The dual simplex method is used to solve the linear relaxation of this problem taken in its integer linear programming formulation[2]. In case the solution obtained by the simplex method turns out to be integer, the LP-crossover yields the best possible offspring for a_1 and a_2; otherwise it returns genotype a_1.

On the SCP problems from family $G(k)$ it can be shown that if at least one of the parents a_1, a_2 is not optimal then the LP-crossover will return the genotype a_1 unchanged. Thus it follows that the LP-crossover with subsequent standard mutation using $p_{mut} < 1/2$ and the non-binary representation is a monotone reproduction operator.

4 Pessimistic estimates for polynomial-time mutation

In this section we investigate how large the guaranteed lower bounds for the transition probabilities can be if a "fast" i.e. polynomial-time computable mutation operator is used in solving hard optimization problems. The idea is to show that if some problem is unlikely to be efficiently solvable by the randomized algorithms in general and the process of finding the solution by EA consists of several stages then at least one of the stages should require an immense amount of computations (in the worst case). If the information about the problem instance was limited only to the outcomes of fitness evaluations made on the previous steps of EA, it would be logical to treat the problem hardness in the sense of black box complexity (see e.g. (Droste, Jansen, Tinnefeld and Wegener, 2002)) and the hard problems would probably resemble the needle in the haystack function. However in our case the reproduction operator is defined in more general setting (it may use any problem parameters given in the input problem data), thus it is more appropriate to follow the traditional approach used in the analysis of optimization problems complexity (see e.g. (Garey and Johnson, 1979)).

In what follows we shall use the notation analogous to the one in (Ausiello and Protasi, 1995). By Σ^* we denote the set of all strings $\mathbf{s}$ with symbols from $\{0, 1\}$ with arbitrary string length $|\mathbf{s}|$.

[2]The original SCP in linear programming formulation may be rewritten as

$$\min \{z_1 + ... + z_{\mathbf{n}} : \mathbf{A}z \geq e,\ z \in \{0, 1\}^{\mathbf{n}}\},$$

where $\mathbf{A}$ is an $\mathbf{m} \times \mathbf{n}$ matrix of 0s and 1s, e is the $\mathbf{m}$-vector of 1s, and $a_{ij} = 1$, iff $i \in M_j$. The linear relaxation of the SCP is obtained from this problem by replacement of the Boolean constraint $z \in \{0, 1\}^{\mathbf{n}}$ with the condition $z \geq 0$. In the LP-crossover a reduced version of the problem will normally have smaller dimensions and some submatrix of matrix $\mathbf{A}$ in its formulation.

Definition 3 *An NP maximization problem P_{max} is a triple $P_{max} = (I, Sol, f_x)$, where $I \subseteq \Sigma^*$ and $Sol(x) \subseteq \Sigma^*$ are such that:*

1. I is the set of instances of P_{max} and is recognizable in polynomial time (through this paper the term polynomial time implies the running time bounded by a polynomial on length of input instance encoding $|x|, x \in I$).

2. Given an instance $x \in I$, $Sol(x)$ denotes the set of feasible solutions of x. Given x and y the decision whether $y \in Sol(x)$ may be done in polynomial time, and there exists a polynomial h such that given any $x \in I$ and $y \in Sol(x)$, $|y| \leq h(|x|)$.

3. Given an instance $x \in I$ and $y \in Sol(x)$, $f_x(y)$ is a positive integer objective function (to be maximized), which is computable in polynomial time.

A similar definition may be given for NP minimization problems, and all the following statements may be properly adjusted for the minimization case as well. Here we will consider the maximization problems for convenience. Let us denote the optimal objective function value for instance x by $f_x^* = \max_{y \in Sol(x)} f_x(y)$.

We will also need the formal definitions of a randomized algorithm and of class BPP of languages recognizable with bounded probability in polynomial time (see e.g. (Ko, 1982, Motwani and Raghavan, 1995)). By a randomized algorithm we mean an algorithm which may be executed by a *probabilistic Turing machine*, i.e. the Turing machine which has a special state for "tossing a coin". When the machine enters this state it receives a bit which is 0 with probability 1/2 and 1 with probability 1/2. A *polynomial-time probabilistic Turing machine* is a probabilistic Turing machine which always halts after a polynomial number of steps. Studying the existence of mutation operators with certain lower bounds for transition probabilities we will consider the possibility of implementing the mutation by a polynomial-time randomized algorithm.

Definition 4 *BPP is the class of languages $L \subseteq \Sigma^*$ for which there exists a polynomial-time probabilistic Turing machine M, such that:*
1) *For all $x \in L$ holds $P\{M \text{gives an output } 1\} \geq 3/4$.*
2) *For all $x \notin L$ holds $P\{M \text{gives an output } 0\} \geq 3/4$.*

Note that one of the open questions in complexity theory is the relation between the classes NP and BPP. A widely believed conjecture is that $NP \not\subseteq BPP$, which is also equivalent to another conjecture $NP \neq RP$ (see e.g. (Ko, 1982)). We will use the following "folklore" result for class BPP.

Lemma 11 *Let P_{max} be an NP-hard NP maximization problem. Then unless $NP \subseteq BPP$, no randomized algorithm solves all instances x of P_{max} in polynomial time with probability more than $1/\text{poly}(|x|)$, where $\text{poly}(|x|)$ is a polynomial in the length of input x.*

Since a variable x for problem instance has been introduced, it's logical to treat all parameters of our model and the mappings y and Φ as functions of x also.

So far no specific assumptions have been made concerning the method of solutions encoding in genotype strings. Given the encoding scheme for feasible solutions of an NP maximization problem, one can use the string y as a genotype, but we will assume an arbitrary mapping $y^{(x)} : A^{n(x)} \to D$ computable in polynomial time.

Now we will consider $\Phi^{(x)} : A^{n(x)} \to \mathbf{R}$, using a set of $d(x)$ fitness function levels $0 < \Phi_1^{(x)} < \Phi_2^{(x)} \ldots < \Phi_{d(x)}^{(x)}$ and denoting the lower bounds for transition probabilities by $\alpha_{ij}(x)$, $0 \le i \le d(x)$, $1 \le j \le d(x)$. Besides that in our model we will require that the number of subsets $H_j^{(x)}$ containing the infeasible solutions i.e. max $\left\{ j : y^{(x)}\left(H_j^{(x)}\right) \not\subseteq Sol(x) \right\}$ is bounded above by some polynomial in $|x|$.

Let us recall that if f_x^* is bounded by some polynomial in $|x|$ for all instances $x \in I$, an NP maximization problem P_{max} is called *polynomially bounded.*

Proposition 12 *Suppose P_{max} is a polynomially bounded NP-hard NP maximization problem and the mappings $y^{(x)} : A^{n(x)} \to D$, $\Phi^{(x)} : A^{n(x)} \to \mathbf{R}$ are computable in polynomial time. If a polynomial-time mutation operator exists for P_{max} such that for some polynomial* poly *for all instances x*

$$\alpha_{01}(x) \ge 1/\mathrm{poly}(|x|),\ \alpha_{12}(x) \ge 1/\mathrm{poly}(|x|), ..., \alpha_{d(x)-1,d(x)}(x) \ge 1/\mathrm{poly}(|x|), \qquad (8)$$

provided that $f(y^{(x)}(g)) = f_x^*$ for all $g \in H_{d(x)}^{(x)}$, then $NP \subseteq BPP$.

Proof. Since P_{max} is polynomially bounded, there exists a polynomial poly_1 such that $d(x) \le \mathrm{poly}_1(|x|)$ for all $x \in I$. Suppose a mutation operator $\mathcal{M}$ complies with bounds (8). Then by Prop. 7 the (1+1)-EA attains the optimum after not more than $\mathrm{poly}(|x|)\mathrm{poly}_1(|x|)$ mutations on average. Denote the random number of the iteration when the (1+1)-EA finds an optimal genotype in $H_{d(x)}^{(x)}$ by $\tau^*(x)$. Then by the Markov inequality,

$$P\left\{ \tau^*(x) \ge 2 \cdot \mathrm{poly}(|x|)\mathrm{poly}_1(|x|) \right\} \le \frac{1}{2}.$$

So running the (1+1)-EA for $2 \cdot \mathrm{poly}(|x|)\mathrm{poly}_1(|x|)$ iterations with arbitrary $g^{(0)} \in A^{n(x)}$ and applying $y^{(x)}(g)$ to the final genotype we will obtain the optimum with probability at least 1/2, and by Lemma 11 the required statement follows. □

Prop. 12 shows that unless $NP \subseteq BPP$, in the setting defined in the formulation, the mutation operators with polynomial running time will necessarily fail to produce with high probability the offspring fitter than the parent at least for some parent individuals in some instances of P_{max}. Of course in practice these "degenerate" instances and individuals may be very rare and the evolvability (i.e. likelihood of parents being able to produce offspring fitter than themselves) may be high enough.

Note that the same situation takes place with any polynomial-time reproduction operator $\mathcal{R}$ because in Prop. 12 we may also consider the mutation operator $\mathcal{M}_{\mathcal{R}}$ corresponding to $\mathcal{R}$ as discussed in Sect. 2 (then $\alpha_{i,i+1}(x)$ for $\mathcal{M}_{\mathcal{R}}$ would be the lower bounds on probability that any collection of parent genotypes $b^1, ..., b^r$ such that argmax $\{\Phi(b^1), ..., \Phi(b^r)\} \in H_i \setminus H_{i+1}$, produces at least one offspring in H_{i+1}). This implies that in the evolutionary algorithms with efficient operators the non-vanishing guarantees of evolvability for the NP-hard polynomially bounded problems are unlikely to exist.

5 Discussion

The upper bounds on the (1+1)-EA complexity in Sect. 3 demonstrate that for some families of instances the (1+1)-EA is relatively competitive compared to some enumerative algorithms. However these families of problems may be easily solved using simple local search heuristics, so the random nature of (1+1)-EA is not so important here. Droste, Jansen, Tinnefeld and Wegener (2002) have shown that (1+1)-EA is a near-optimal search strategy in the class of needle in the haystack functions in black-box scenario, but at the same time the optimality was established for the trivial random search routine as well. From a formal view point a number of randomized algorithms successfully used in complexity theory may also be considered as the (1+1)-EA optimizing a needle in the haystack function of specific structure, e.g. the algorithm of C. Papadimitriou for 2-Satisfiability problem with expected running time $O(n^2)$ (see e.g. (Motwani and Raghavan, 1995)), the algorithm of U. Schöning (1999) for k-Satisfiability problem etc. However none of these methods actually exploits the full power of the (1+1)-EA since the search process is independent of fitness evaluations until the optimum is found (obviously no meaningful correlation assumption as discussed in Sect. 1 applies in such a case).

The minimum graph bisection problem with random graphs drawn from the "planted bisection" model is one of the few complicated optimization problems, for which it has been theoretically shown (Carson and Impagliazzo, 2001) that there is little difference between the "full blown" (1+1)-EA and the more sophisticated methods such as Metropolis algorithms and even a problem-specific heuristic of Boppana (1987). It would be interesting to find more non-trivial problem classes where the random evolutionary specifics of the (1+1)-EA makes it the method of choice. These further studies may involve the results presented here or the more problem-specific approaches like those in (Carson and Impagliazzo, 2001, He and Yao, 2001, Wegener, 2001).

Acknowledgements

The research was supported in part by the INTAS grant 00-217 and "Integracia" program grant.

References

Aldous, D. and U. V. Vazirani (1994). "Go with the winners" algorithms. In *IEEE Symposium on Foundations of Computer Science*, pp. 492–501.

Altenberg, L. (1994). The evolution of evolvability in genetic programming. In K. E. Kinnear (Ed.), *Advances in Genetic Programming*, pp. 47–74. Cambridge, MA: MIT Press.

Altenberg, L. (1995). The schema theorem and Price's theorem. In D. Whitley and M. D. Vose (Eds.), *Foundations of Genetic Algorithms 3*, pp. 23–49. San Mateo, CA: Morgan Kaufmann.

Ausiello, G. and M. Protasi (1995). Local search, reducibility and approximability of NP-optimization problems. *Information Processing Letters 54*, 73–79.

Bäck, T. (1993). The interaction of mutation rate, selection, and self-adaptation within a genetic algorithm. In R.Männer and B. Manderick (Eds.), *Proceedings of Parallel Problem Solving from Nature II (PPSN II)* North Holland, pp. 85–94.

Balas E. (1984). A sharp bound on the ratio between optimal integer and fractional covers. *Mathematics of Operations Research 9* (1), 1–5.

Beasley, J. E. and P. C. Chu (1996). A genetic algorithm for the set covering problem. *European Journal of Operation Research 94* (2), 394–404.

Boese, K. D., A. B. Kahng and S. Muddu (1994). A new adaptive multi-start technique for combinatorial global optimizations. *Operations Research Letters 16*, 101–113.

Boppana, R. B. (1987). Eigenvalues and graph bisection: an average case analysis. In *Proceedings of the 28th IEEE Symposium on Foundations of Computer Science*, pp. 280–285.

Borovkov, A. A. (1998). *Probability theory.* Gordon and Breach.

Carson, T. and R. Impagliazzo (2001). Hill-climbing finds random planted bisections. In *Proceedings of ACM-SIAM Symposium on Discrete Algorithms (SODA'2001)*, pp. 903–909.

Droste, S., T. Jansen, K. Tinnefeld and I. Wegener (2002). A new framework for the valuation of algorithms for black-box optimisation. In *Foundations of Genetic Algorithms 7.*

Holland, J. (1975). *Adaptation in natural and artificial systems.* University of Michigan Press.

Eremeev A. V. (1999) A Genetic Algorithm with a Non-Binary Representation for the Set Covering Problem. In P. Kall and H.-J. Lüthi (Eds.), *Proceedings of Operations Research (OR'98).* pp. 175–181. Springer Verlag.

Eremeev A. V. (2000). Modeling and analysis of genetic algorithm with tournament selection. In C. Fonlupt et al (Eds.), *Proceedings of Artificial Evolution Conference (AE'99). Lecture Notes in Computer Science, 1829.* pp. 84–95. Springer Verlag.

Garey, M. and D. Johnson (1979). *Computers and intractability. A guide to the theory of NP-completeness.* W.H. Freeman and Company.

Garnier, J., L. Kallel and M. Schoenauer (1999). Rigorous hitting times for binary mutations. *Evolutionary Computation 7* (1), 45–68.

He, J. and X. Yao (2001). Drift analysis and average time complexity of evolutionary algorithms. *Artificial Intelligence 127*, 57–85.

Ko, K. (1982). Some observations on the probabilistic algorithms and NP-hard problems, *Information Processing Letters 14,* 39–43.

Kolokolov, A. A. (1996). Regular partitions and cuts in integer programming. In A. D. Korshunov (Ed.), *Discrete Analysis and Operations Research*, pp. 59–79. Kluwer Academic Publishers.

Koza, J. R. (1992). *Genetic programming: On the programming of computers by means of natural selection.* MIT Press.

Lankaster, P. (1969). *Theory of matrices.* Academic Press.

Lbov, G. S. (1972). Training for extremum determination of function of variables measured in names scale. In *Proceedings of Second Int. Conf. on Artifical Intelligence*, London, pp. 418–423.

Motwani, R. and P. Raghavan (1995). *Randomized algorithms.* Cambridge University Press.

Mühlenbein, H. (1993). How genetic algorithms really work: I. Mutation and hillclimbing. In R. Männer and B. Manderick (Eds.), *Proceedings of Parallel Problem Solving from Nature II (PPSN II).* North Holland, pp. 15–26.

Rudolph, G. (1998). Finite Markov chain results in evolutionary computation: A tour d'horizon. *Fundamenta Informaticae 35* (1–4), 67–89.

Schöning, U. (1999). A probabilistic algorithm for k-SAT and constraint satisfaction problems. In *Proceedings of 40th IEEE Symposium on Foundations of Computer Science*, pp. 410–414.

Schrijver, A. (1986). *Theory of linear and integer programming*, Volume 2. John Wiley & Sons.

Wegener, I. (2001). Theoretical aspects of evolutionary algorithms. In *Proceedings of theTwenty-EighthInternational Colloquium on Automata, Languages, and Programming(ICALP 2001)*, Crete, Greece, pp. 64–68.

Wolpert, D. H. and W. G. Macready (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation 1*(1), 67–82.

Zaozerskaya, L. (1998). *Investigation and solving of some classes of integer programming problems on the basis of the regular partitions.* Ph. D. thesis, Omsk Branch of Sobolev Institute of Mathematics, SB RAS. (in Russian).