On the Use of the Dual Formulation for Minimum Weighted Vertex Cover in Evolutionary Algorithms

Mojgan Pourhassan
Optimisation and Logistics
School of Computer Science
The University of Adelaide
Adelaide, Australia

Tobias Friedrich
Algorithm Engineering
Hasso Plattner Institute
Potsdam, Germany

Frank Neumann
Optimisation and Logistics
School of Computer Science
The University of Adelaide
Adelaide, Australia

ABSTRACT

We consider the weighted minimum vertex cover problem and investigate how its dual formulation can be exploited to design evolutionary algorithms that provably obtain a 2-approximation. Investigating multi-valued representations, we show that variants of randomized local search and the (1+1) EA achieve this goal in expected pseudo-polynomial time. In order to speed up the process, we consider the use of step size adaptation in both algorithms and show that RLS obtains a 2-approximation in expected polynomial time while the (1+1) EA still encounters a pseudo-polynomial lower bound.

1. INTRODUCTION

The theoretical understanding of bio-inspired computing techniques lags far behind their practical success. These algorithms are very popular with practitioners from various domains such as engineering and economics. Having a good understanding of the working principles of evolutionary algorithms and other bio-inspired algorithms helps to increase the trust in these methods and leads to the design of new high performing algorithms. The area of runtime analysis of bio-inspired computing algorithms has contributed significantly to the theoretical understanding of these methods [2, 22, 27].

One of the most prominent NP-hard combinatorial optimization problems that has been studied in this context is the minimum vertex cover problem. Node-based approaches have been studied for this problem in the single-objective and multi-objective setting [18, 25]. Friedrich et al. [18] have shown that in expected polynomial time, the single-objective (1+1) EA cannot achieve a better than trivial approximation ratio. Further investigations regarding the approximation behaviour of evolutionary algorithms for the vertex cover problem have been carried out in [17, 30]. Furthermore, edge-based encodings have been investigated in [23], where an evolutionary algorithm finds a maximal matching from which a 2-approximation vertex cover can be induced.

Inspired by their approach, we investigate a different way of approximating the minimum vertex cover problem by evolutionary algorithms. While Jansen et al. [23] considered the classical vertex cover problem, we analyse the weighted version of the problem. We study an edge-based encoding together with a multi-valued representation that works on the dual of the minimum vertex cover formulation. Our investigations can be seen as a generalization of the approach based on matchings investigated by Jansen et al. [23], although no direct connection to the use of dual formulations was made in that paper. We are only aware of four previous theoretical works with multi-valued representations. Doerr et al. [11, 13] regard the optimization of multi-valued linear functions via a variant of the (1+1) EA. More recently, static and dynamically changing variants of multi-valued OneMax functions have been considered [9, 24].

We analyze edge-based approaches generalizing the edge-based encoding in conjunction with a fitness function obtaining maximal matchings investigated in [23]. Our edge-based approaches consider the dual formulation of the weighted vertex cover problem. Working with the dual formulation an encoding assigns a weight to each edge. During the evolutionary process the weight of the edges may be increased or decreased and vertices whose constraints become tight are selected as vertices for the cover. We first study the situation where each weight can only increase or decrease by 1 at each step and present pseudo-polynomial upper bounds on the expected time until our approaches have obtained a 2-approximation for the minimum vertex cover problem.

In order to deal with potentially large weights of the given graph, we incorporate *step size adaptation* into our algorithms. Step size adaptation is a popular mechanism to steer the progress of an evolutionary algorithm to the right level. Step size adaptation is a form of *parameter control* [15], where a parameter is changed during the execution of the algorithm. Adaptive parameters are very essential in continuous search spaces [4] and popularly used for covariance-matrix adaptation [19]. There are only few theoretical studies on adaptive parameters in discrete spaces. Known results are that changing the mutation rate [5, 7] and the population size [8] can reduce the asymptotic runtime. Moreover, dynamically choosing the number of parallel instances in parallel evolutionary algorithms is studied in [26], and self-adjusting of the number of bits to be flipped instead of a standard bit mutation is shown to improve the performance of the optimization process [10].

In this paper, defining $c_1 > 1$ and $c_2 > 1$ as two constants, we show that the use of step size adaptation where the step size is multiplied by c_1 in the case of a success and multiplied by $1/c_2$ in case of failure, leads to a polynomial upper bound on the expected runtime of the RLS algorithm to achieve a 2-approximation. Furthermore, we present a pseudo-polynomial lower bound for the (1+1) EA using this step size adaptation. The proof uses the insight that the considered (1+1) EA is not able to achieve a sufficiently

FOGA '17, January 12 - 15, 2017, Copenhagen, Denmark

ISBN 978-1-4503-4651-1/17/01…$15.00

DOI: http://dx.doi.org/10.1145/3040718.3040726

Algorithm 1: RLS

Initialize $s := 0^m$ and $\sigma := 1^m$;
while *termination condition not satisfied* **do**
  $s' := s$;
  Choose $i \in \{1, \cdots, m\}$ uniformly at random;
  Choose $b \in \{0, 1\}$ uniformly at random;
  if $b = 0$ **then**
    $s'_i := s'_i + \sigma_i$;
  else
    $s'_i := \max(s'_i - \sigma_i, 0)$;
  if $\sum_{i=1}^{m} s_i < \sum_{i=1}^{m} s'_i$ *and* $\sum_{j \in \{1,\cdots,m\} | e_j \cap \{v\} \neq \emptyset} s'_j \leq w(v),\ \forall v \in V$ **then**
    $s := s'$;
return $C := \{v \in V \mid w(v) = \sum_{j \in \{1,\cdots,m\} \mid e_j \cap \{v\} \neq \emptyset} s_j\}$;

Algorithm 2: (1+1) EA

Initialize $s := 0^m$ and $\sigma := 1^m$;
while *termination condition not satisfied* **do**
  $s' := s$;
  for $i := 1$ **to** m **do**
    with probability $1/m$ **do**
      Choose $b \in \{0, 1\}$ uniformly at random;
      if $b = 0$ **then**
        $s'_i := s'_i + \sigma_i$;
      else
        $s'_i := \max(s'_i - \sigma_i, 0)$;
  if $\sum_{i=1}^{m} s_i < \sum_{i=1}^{m} s'_i$ *and* $\sum_{j \in \{1,\cdots,m\} | e_j \cap \{v\} \neq \emptyset} s'_j \leq w(v),\ \forall v \in V$ **then**
    $s := s'$;
return $C := \{v \in V \mid w(v) = \sum_{j \in \{1,\cdots,m\} \mid e_j \cap \{v\} \neq \emptyset} s_j\}$;

large step size during the optimization process in order to reach a 2-approximation.

This paper is structured as follows. In Section 2, we present our edge-based approach based on a dual formulation for solving the minimum vertex cover problem. We analyze RLS and (1+1) EA with a step size of 1 in Section 3. Afterwards, we show a polynomial upper bound for RLS with Step Size Adaptation in Section 4 and a pseudopolynomial lower bound for (1+1) EA with Step Size Adaptation in Section 5. Finally, we finish with some concluding remarks.

2. PRELIMINARIES

The weighted vertex cover problem is defined as follows. Given a graph $G = (V, E)$ with vertex set $V = \{v_1, \ldots, v_n\}$ and edge set $E = \{e_1, \ldots, e_m\}$, and a positive weight function $w\colon V \to \mathbb{N}^+$ on the vertices, the goal is to find a subset of nodes, $V_C \subseteq V$, that covers all edges and has minimum weight, i.e. $\forall e \in E, e \cap V_C \neq \emptyset$ and $\sum_{v \in V_C} w(v)$ is minimized.

Consider the standard node-based representation, in which a solution $x = (x_1, \ldots, x_n)$ is a bitstring of size n, where $x_i = 1$ iff the node v_i is chosen. With this representation, the Integer Linear Programming (ILP) formulation for this problem is:

$$\begin{aligned} & \min \sum_{i=1}^{n} w(v_i) \cdot x_i \\ s.t. \quad & x_i + x_j \geq 1 \quad \forall (i,j) \in E \\ & x_i \in \{0,1\} \quad \forall i \in \{1, \cdots, n\} \end{aligned}$$

In the linear programming relaxation of the problem, the Fractional Weighted Vertex Cover Problem, the constraint $x \in \{0, 1\}$ is relaxed to $x \in [0, 1]$. We denote the cost of the optimal solution for the original problem and the relaxed version of the problem by OPT and OPT^* respectively. Observe that $OPT^* \leq OPT$.

Any LP problem (which we refer to as the primal problem) has a dual form, which is also an LP problem. When the primal problem is a minimization problem, the dual problem helps with finding lower bounds of the optimal solution of the primal problem (or upper bounds in case the primal form is a maximization problem). Consider the following standard LP problem in which the goal is to minimize the objective function.

$$\begin{aligned} & \min \ \sum_{i=1}^{n} c_i x_i \\ s.t. \quad & \sum_{i=1}^{n} a_{ji} x_i \geq b_j \quad j = 1, \cdots, m \\ & x_i \geq 0, \quad i = 1, \cdots, n \end{aligned}$$

where c_i, a_{ji} and b_j are given rational numbers. The dual form of the above LP problem can be formulated as the following, where y_i is a variable for the ith inequality. For more explanations on primal and dual forms of an LP problem, and how to derive the dual form from the primal form refer to [31].

$$\begin{aligned} & \max \ \sum_{j=1}^{m} b_j y_j \\ s.t. \quad & \sum_{j=1}^{m} a_{ji} y_j \leq c_i \quad i = 1, \cdots, n \\ & y_j \geq 0, \quad j = 1, \cdots, m \end{aligned}$$

Considering these formulations, the Weak Duality Theorem described below, helps in finding lower bounds of any feasible solution of the primal problem. The reader can find the proof of this theorem in [31].

Theorem 1 (The Weak Duality Theorem)**.** *If $x = (x_1, \cdots, x_n)$ and $y = (y_1, \cdots, y_m)$ are feasible solutions for the primal and dual problem respectively, then*

$$\sum_{i=1}^{n} c_i x_i \geq \sum_{j=1}^{m} b_j y_j.$$

Using the concept of duality and the Weak Duality Theorem, 2-approximations of the vertex cover problem can be obtained. The dual of the relaxed covering problem is a packing problem formulated as the following, where $s_j \in \mathbb{N}^+$ denotes a weight on the edge e_j:

$$\begin{aligned} & \max \sum_{j=1}^{m} s_j \\ s.t. \quad & \sum_{j \in \{1,\cdots,m\} | e_j \cap \{v\} \neq \emptyset} s_j \leq w(v) \quad \forall v \in V \end{aligned}$$

Algorithm 3: RLS with Step Size Adaptation

Initialize $s := 0^m$ and $\sigma := 1^m$;
while *termination condition not satisfied* **do**
  $s' := s$;
  $I := \emptyset$;
  Choose $i \in \{1, \cdots, m\}$ uniformly at random;
  Choose $b \in \{0, 1\}$ uniformly at random;
  if $b = 0$ **then**
    $s'_i := s'_i + \sigma_i$;
  else
    $s'_i := \max(s'_i - \sigma_i, 0)$;
  $I := I \cup \{i\}$;
  if $\sum_{i=1}^{m} s_i < \sum_{i=1}^{m} s'_i$ *and* $\sum_{j \in \{1,\cdots,m\} | e_j \cap \{v\} \neq \emptyset} s'_j \leq w(v),\ \forall v \in V$ **then**
    $s := s'$;
    $\sigma_i := c_1 \cdot \sigma_i,\ \forall i \in I$;
  else
    $\sigma_i := \max\left(\frac{\sigma_i}{c_2}, 1\right),\ \forall i \in I$;
return $C := \{v \in V \mid w(v) = \sum_{j \in \{1,\cdots,m\} \mid e_j \cap \{v\} \neq \emptyset} s_j\}$;

Algorithm 4: (1+1) EA with Step Size Adaptation

Initialize $s := 0^m$ and $\sigma := 1^m$;
while *termination condition not satisfied* **do**
  $s' := s$;
  $I := \emptyset$;
  for $i := 1$ **to** m **do**
    with probability $1/m$ **do**
      Choose $b \in \{0, 1\}$ uniformly at random;
      if $b = 0$ **then**
        $s'_i := s'_i + \sigma_i$;
      else
        $s'_i := \max(s'_i - \sigma_i, 0)$;
      $I := I \cup \{i\}$;
  if $\sum_{i=1}^{m} s_i < \sum_{i=1}^{m} s'_i$ *and* $\sum_{j \in \{1,\cdots,m\} | e_j \cap \{v\} \neq \emptyset} s'_j \leq w(v),\ \forall v \in V$ **then**
    $s := s'$;
    $\sigma_i := c_1 \cdot \sigma_i,\ \forall i \in I$;
  else
    $\sigma_i := \max\left(\frac{\sigma_i}{c_2}, 1\right),\ \forall i \in I$;
return $C := \{v \in V \mid w(v) = \sum_{j \in \{1,\cdots,m\} \mid e_j \cap \{v\} \neq \emptyset} s_j\}$;

In other words, the dual problem is to maximize the sum of weights on all edges, provided that for each vertex, the sum of weights of edges incident to that vertex is at most equal to the weight of that vertex.

Let $s = (s_1, \cdots, s_m)$, be a maximal feasible solution for the dual problem with a cost of $Cost_D$. Since s is a maximal solution, none of the edges can be assigned a greater weight without violating a constraint. Therefore, for at least one vertex of each edge, v, we have

$$w(v) = \sum_{j \in \{1,\cdots,m\} | e_j \cap \{v\} \neq \emptyset} s_j$$

As a result, the set of nodes for which the above equality holds, $C = \{v \in V \mid w(v) = \sum_{j \in \{1,\cdots,m\} \mid e_j \cap \{v\} \neq \emptyset} s_j\}$, is a vertex cover. The cost of this vertex cover, $Cost_P$, is at most twice the weight of all edges in the dual solution. Therefore, $Cost_P \leq 2 \cdot Cost_D$. Moreover, since s is a feasible solution, according to Weak Duality Theorem (Theorem 1), $Cost_D \leq OPT$, which results in $Cost_P \leq 2 \cdot OPT$, i. e. set C is a 2-approximation for the weighted vertex cover problem.

Constructing maximal solutions for the dual problem has been used in a number of algorithms for finding 2-approximations of the weighted vertex cover problem, e. g. Bar-Yehuda and Evan's greedy algorithm [3] and Clarkson's greedy algorithm [6]. A formal proof of the approximation ratio of the solution obtained by this approach can be found in Theorem 8.4 in [14] (represented in Theorem 2 below). There, the output of a specific algorithm is studied as the maximal dual solution, but the presented proof is valid for Theorem 2 with any given maximal solution s.

Theorem 2. *Consider s, a maximal feasible solution for the dual problem of the relaxed weighted vertex cover problem. The vertex set*

$$C = \{v \in V \mid w(v) = \sum_{j \in \{1,\cdots,m\} \mid e_j \cap \{v\} \neq \emptyset} s_j\}$$

is a 2-approximation for the original weighted vertex cover problem.

In this paper, we analyse the behaviour of four evolutionary algorithms which find a 2-approximation for the weighted vertex cover problem by means of finding a maximal solution for the dual form of the problem. A simple Randomized Local Search (RLS) is presented in Algorithm 1, where a solution $s = (s_1, \cdots, s_m)$, is represented by a string of m integers, denoting the weights of the m edges of the input graph. This algorithm starts with the initial solution $s = 0^m$, and selects one edge at each step to increase or decrease the weight corresponding to that by one. The new solution replaces the old one, if the sum of weights of edges is increased, and the weight constraint of the packing problem is not violated for any of the vertices. At the end, the algorithm returns the set of nodes for which the constraint has become tight.

One other algorithm that we analyse in this paper is the (1+1) EA, presented in Algorithm 2, which is quite similar to the RLS of Algorithm 1 except for selecting the edges for mutation. In (1+1) EA, at each step a mutation happens on the weight of all edges with probability $1/m$ for each of them, while in RLS one edge is selected and the mutation takes place on the weight of that edge. Note that in (1+1) EA more than one mutation may happen on the current solution.

In both RLS and (1+1) EA (Algorithms 1 and 2) the increment size of one on the weights of the edges might be too small and make the algorithm slow. Motivated by step size adaptation in evolution strategies [4] in RLS with Step Size Adaptation and (1+1) EA with Step Size Adaptation (Algorithms 3 and 4), a step size for each edge is kept in an auxiliary vector $\sigma = (\sigma_1, \cdots, \sigma_m)$. The initial step size for all edges is set to 1. The algorithms work with two constant parameters $c_1 > 1$ and $c_2 > 1$. If a mutation with that size is accepted, the step size is increased by a factor of c_1; otherwise, it is decreased by a factor of c_2 with a minimum accepted size of one.

Analysing the runtime of our algorithms, we find the number of iterations of the while loop, until a maximal packing solution is found, which induces a complete vertex cover. We call this the expected time of obtaining the desired goal by the considered algorithm. It should be noted that the edge-based approach for the unweighted minimum vertex cover investigated by Jansen et al. [23] can be seen as a special case of our formulation as the use of maxi-

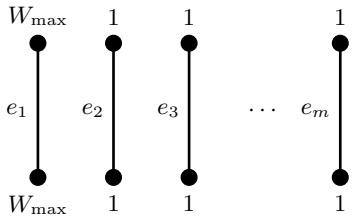

Figure 1: G, a hard instance for RLS and (1+1) EA

mal matchings is equivalent to the dual problem if all edges have a weight of 1.

3. RLS AND (1+1) EA

In this section, we present the analysis on finding 2-approximations for the weighted vertex cover problem by RLS and (1+1) EA.

Theorem 3. *The expected time of RLS and (1+1) EA (Algorithms 1 and 2) to find a 2-approximation is $\mathcal{O}(m \cdot OPT)$.*

Proof. In order to prove this theorem, we show that the algorithms find a maximal solution for the dual problem in expected time $\mathcal{O}(m \cdot OPT)$. Having achieved that maximal solution, the algorithms return the set

$$C := \{v \in V \mid w(v) = \sum_{j \in \{1,\cdots,m\} \mid e_j \cap \{v\} \neq \emptyset} s_j\}$$

as the solution for the weighted vertex cover problem which, according to Theorem 2, is a 2-approximation of the optimal solution.

If a solution s is not a maximal solution for the dual problem, then there exists at least one edge for which the assigned weight can be increased. The probability of selecting only that edge for mutation and choosing $b = 0$ is at least $\frac{1}{2 \cdot m}$ for RLS and $\frac{1}{2 \cdot e \cdot m}$ for (1+1) EA at each step, and according to the Weak Duality Theorem (Theorem 1), the cost of any maximal solution is upper bounded by OPT. Therefore, using the method of Fitness Based Partitions [32], we find the expected time $\mathcal{O}(m \cdot OPT)$ for finding a maximal solution for the dual problem by both algorithms. □

Note that the presented upper bound in Theorem 3 is a pseudo polynomial time, because OPT can be exponentially large with respect to the input size. In the remainder of this section, we introduce an instance of the problem for which a pseudo polynomial time is required for finding a 2-approximation. This instance is also used in Section 5, as a hard instance for the (1+1) EA with Step Size Adaptation.

The hard instance of the problem, G, illustrated in Figure 1, contains m edges, $e_1, \cdots, e_m$, none of which share a node with another. One of the edges, e_1, is adjacent to two nodes of weight $W_{\max}$ while all other edges are adjacent to vertices of weight 1. The dual problem of this instance has only one maximal solution: $s_1 = W_{\max}$ and $s_i = 1,\ 2 \leq i \leq m$. In this instance, we assume that $W_{\max} > 2^m$.

Theorem 4. *With probability $1 - e^{-\Omega(2^m)}$, the required time for RLS and the (1+1) EA (Algorithm 1 and 2) to find a 2-approximation of G is lower bounded by $\Omega(m \cdot W_{\max})$.*

Proof. Consider a phase of $\frac{m \cdot W_{\max}}{4}$ steps. Let X be the number of times that e_1 is selected for mutation by RLS or (1+1) EA in this phase. Since the probability of selecting e_1 is $\frac{1}{m}$ for both algorithms, the expected value of X is $\frac{W_{\max}}{4}$. As these probabilities are independent of each other at each step, by Chernoff bounds we get

$$\Pr(X > \frac{W_{\max}}{2}) \leq e^{-\frac{W_{\max}}{12}} = e^{-\Omega(2^m)}$$

At each step that e_1 is selected for mutation, s_1 can be increased by at most 1. Therefore, with probability $1 - e^{-\Omega(2^m)}$, in a phase of $\frac{m \cdot W_{\max}}{4} = \Omega(m \cdot W_{\max})$ steps, we have $s_1 \leq \frac{W_{\max}}{2}$, i.e. s_1 does not reach its maximal value of $W_{\max}$. Therefore, with probability $1 - e^{-\Omega(2^m)}$, the RLS and the (1+1) EA find a 2-approximation of G in time $\Omega(m \cdot W_{\max})$. □

4. RLS with Step Size Adaptation

In this section, we analyse the behaviour of RLS with Step Size Adaptation for finding 2-approximations of the weighted vertex cover problem. We prove that the RLS with Step Size Adaptation finds a 2-approximation for the weighted vertex cover problem in expected polynomial time with respect to the input size, provided that $c_1 = c_2$. This also holds for $c_1 \geq c_2$, which is stated in Corollary 8. The two lemmata below are used in the proof of the main result stated later.

Lemma 5. *If $c_1 = c_2$, the step size σ_i for each edge e_i in RLS with Step Size Adaptation, can only take a value from*

$$\{{c_1}^k \mid 0 \leq k \leq \lceil \log_{c_1} W_{\max} \rceil\},$$

where $W_{\max}$ is the largest weight assigned to any vertex.

Proof. The algorithm starts with initial value of $\sigma_i = 1$ for all edges. This value is increased by a factor of c_1 each time a mutation is accepted for edge e_i, and is divided by the same factor with a minimum accepted value of one if the mutation is rejected (lines 14 and 16 of Algorithm 3). Therefore σ_i is always a power of c_1. Moreover, in order to fulfil the constraints on the vertices, none of the edges can be assigned a weight larger than $W_{\max}$. Therefore, any mutation that increases the current weight of an edge by at least $W_{\max}$, is rejected. Therefore, σ_i can be increased to at most c_1^k where $k = \lceil \log_{c_1} W_{\max} \rceil$. □

Lemma 6. *For an edge e_i, let $D(s_i) = MAX_i - s_i$ where s is the solution obtained so far by the algorithm and MAX_i is the maximum acceptable value for s_i in the current solution s. In expected time $\mathcal{O}(m \log_{c_1}^2 W_{\max})$ a solution s' with $D(s'_i) \leq \frac{c_1 \cdot D(s_i)}{c_1 + 1}$ is found by RLS with Step Size Adaptation when $c_1 = c_2$.*

Proof. Note that since at any step only one mutation happens, for any solution s' obtained after s, we have $D(s'_i) \leq D(s_i)$, otherwise the algorithm would have rejected s'. We divide the analysis into two phases. The first phase, consists of all steps until the algorithm reaches a situation in which s_i is selected for an increasing mutation and $\sigma_i \leq D(s_i)$. In this phase σ_i decreases. The second phase begins when σ_i starts increasing. We show that by the end of the second phase, we have reached a solution s' with $D(s'_i) \leq c_1 \cdot D(s_i)/(c_1 + 1)$.

In the first phase, whenever s_i is selected for an increase, we have $\sigma_i > D(s_i)$; therefore, σ_i is decreased. If $\sigma_i \leq D(s_i)$ at a step in which s_i is selected for an increase, then we are already in the second phase and σ_i is added to s_i, resulting in decreasing $D(s_i)$. Note that it is not only increasing s_i that decreases $D(s_i)$. Instead, increasing the weight of other edges that are adjacent to e_i can also decrease $D(s_i)$. If we reach a solution s' where $D(s'_i) = 0$ in Phase 1, then we already have $D(s'_i) \leq \frac{c_1 \cdot D(s_i)}{c_1 + 1}$ (stated in the lemma) without going to Phase 2.

Here we show that Phase 1 is over in expected time $\mathcal{O}(m \cdot \log_{c_1} W_{\max})$. At each step, with probability $\frac{1}{m}$, s_i is mutated. Since $\sigma_i > D(s_i)$, increasing mutations on s_i are rejected as well as decreasing mutations, and σ_i is divided by c_1 with each rejection. This needs to be done at most $\log_{c_1} W_{\max}$ times until we reach $\sigma_i \leq D(s_i)$, which in expectation takes $\mathcal{O}(m \cdot \log_{c_1} W_{\max})$.

The second phase starts when we reach a step with an increasing mutation on s_i in which $1 \leq \sigma_i \leq D(s_i)$. This move is accepted and σ_i is increased by a factor of c_1. Note that $D(s_i)$ might be far larger than σ_i. Since σ_i is always a power of c_1, we define $a \in \mathbb{N}^+$ as $a = \log_{c_1} \sigma_i$ to make the proof easier. Due to Lemma 5, we have $0 \leq a \leq \lceil \log_{c_1} W_{\max} \rceil$. Here, an increase on s_i is accepted by the algorithm and a is increased to $a+1$, while a decrease is rejected and a is decreased to $a-1$. The increase and decrease happen with equal probability; therefore, a fair random walk happens for a on integer values in $[0, \lceil \log_{c_1} W_{\max} \rceil]$, with initial value of at least 0.

It is proved that the expected number of required steps for a fair random walk to visit all vertices in a graph with v vertices and e edges is bounded by $2e(v-1)$ [1]. In the fair random walk that happens on a, there are $\lceil \log_{c_1} W_{\max} \rceil + 1$ vertices to visit with $\lceil \log_{c_1} W_{\max} \rceil$ edges between them. This gives us the expected number of steps $k = 2\lceil \log_{c_1} W_{\max} \rceil^2 = \mathcal{O}(\log^2_{c_1} W_{\max})$ for our random walk, to reach any possible value of a. As a result, as long as $\sigma_i \leq D(s_i)$ holds, in k mutations on s_i, a reaches its maximal possible value which is upper bounded by $\lceil \log_{c_1} W_{\max} \rceil$ after which the inequality does not hold. This implies that k is an upper bound on the number of mutations that can happen on s_i before this phase ends, which is in expectation done in time $\mathcal{O}(m \cdot \log^2_{c_1} W_{\max})$. At the end of this phase, $\sigma_i > D(s'_i)$, whereas the last accepted mutation has increased s'_i by at least $\frac{1}{c_1}\sigma_i$. This implies that

$$D(s'_i) \leq D(s_i) - \frac{1}{c_1}\sigma_i \leq \frac{c_1}{c_1+1} D(s_i),$$

which completes the proof. □

Theorem 7. *The RLS with Step Size Adaptation with $c_1 = c_2$ and the initial solution $s = 0^m$, finds a vertex cover that is at least a 2-approximation in expected time $\mathcal{O}(m \cdot \log^3_{c_1} W_{\max})$.*

Proof. Similar to the proof of Theorem 3, we show that the algorithm finds a maximal solution for the dual problem in expected time $\mathcal{O}(m \cdot \log^3_{c_1} W_{\max})$.

For each edge e_i, the distance of s_i to its maximal value, D_i, is decreased by at least $\frac{D_i}{c_1+1}$ by RLS with Step Size Adaptation, in expected time $O(m \log^2_{c_1} W_{\max})$ according to Lemma 6. Since the initial value of D_i is upper bounded by $W_{\max}$, according to Multiplicative Drift Theorem [12], s_i reaches its maximal value in expected time $O(m \log^3_{c_1} W_{\max})$. □

In the proof of Lemma 6, setting $c_1 > c_2$, is in favour of increasing the value of σ_i; therefore, the lemma holds in that situation as well, resulting in the following corollary.

Corollary 8. *The RLS with Step Size Adaptation with $c_1 \geq c_2$ and the initial solution $s = 0^m$, finds a vertex cover that is a 2-approximation in expected time $\mathcal{O}(m \cdot \log^3_{c_1} W_{\max})$.*

5. (1+1) EA with Step Size Adaptation

In this section we prove a pseudo polynomial lower bound on the time that (1+1) EA with Step Size Adaptation requires for finding a 2-approximation of the weighted vertex cover problem, when $c_1 \leq c_2$. To prove this lower bound, we investigate the behaviour of (1+1) EA with Step Size Adaptation on G (Figure 1), the hard instance of the problem presented in Section 3, with the assumption that $W_{\max} \geq {c_1}^m$. We show that with high probability, the (1+1) EA with Step Size Adaptation needs exponential time with respect to the input size for finding a maximal dual solution for G.

In the following, $A(s) = \{s_i \mid s_i = 1,\ 2 \leq i \leq m\}$. Moreover, Phase 1 indicates the steps starting from the initial step until finding a solution s, with $|A(s)| \geq \frac{3m}{4}$, and Phase 2 consists of ${c_1}^{m^{\varepsilon/2}}$ steps, where $0 < \varepsilon \leq \frac{1}{3}$, starting by the end of Phase 1. We also define Property 9 below, which is used in Lemmata 12 and 14, and Theorem 16.

Property 9. *For current solution s, we have $|A(s)| \geq \frac{m}{2}$.*

In order to prove the main theorem of this section, we make use of Lemmata 10, 12, 13 and 14, which follow.

Lemma 10. *For sufficiently large m, with probability $1-e^{-\Omega(m^\varepsilon)}$, Phase 1 needs at most $m^{1+\varepsilon}$ steps, where $\varepsilon > 0$ is a constant.*

Proof. Let $Z(s) = \{s_i \mid s_i = 0,\ 2 \leq i \leq m\}$. Note that $|Z(s)| + |A(s)| = m-1$. At each step, if one of the edges of set $Z(s)$ is selected for a mutation of increase, and no other mutations happen, the new solution is accepted by the algorithm. Therefore, the probability of producing a solution s' with $|A(s')| = |A(s)|+1$ is at least

$$\frac{|Z(s)|}{2 \cdot e \cdot m} = \frac{m-1-|A(s)|}{2 \cdot e \cdot m}.$$

This implies that, the positive drift on $|A(s)|$, denoted by Δ_+, is at least $\frac{m-1-|A(s)|}{2 \cdot e \cdot m}$ at each step.

Moreover, to obtain a solution s' with $|A(s')| = |A(s)|-k$ from s, k mutations should happen on edges of A, and in order to make these changes acceptable, a mutation of increase should happen on s_1. The probability of increasing s_1 at each step is $\frac{1}{2m}$, and the probability of k other mutations to happen at the same step is upper bounded by

$$\binom{m-1}{k} \cdot \left(\frac{1}{m}\right)^k \left(1-\frac{1}{m}\right)^{m-1-k} \leq \frac{1.06}{k!e},$$

for sufficiently large m. Here, it suffices if we assume $m \geq 20$. Overall, the probability of finding a solution s' with $|A(s')| = |A(s)| - k$ is at most $\frac{1.06}{k!e \cdot 2m}$. As a result, for the negative drift on $|A(s)|$, denoted by Δ_-, we have

$$\begin{aligned}
\Delta_- &\leq \sum_{k=1}^{|A(s)|} k \cdot \frac{1.06}{k!e \cdot 2m} \\
&= \frac{1.06}{e \cdot 2m} \sum_{k=1}^{|A(s)|} \frac{1}{(k-1)!} \\
&\leq \frac{1.06}{e \cdot 2m} \cdot 3 = \frac{3.18}{e \cdot 2m}.
\end{aligned}$$

Summing up, the total drift on $|A(s)|$ is

$$\Delta = \Delta_+ - \Delta_- \geq \frac{m - 4.18 - |A(s)|}{2 \cdot e \cdot m}.$$

We now analyse the time to find a solution with $|A(s)| \geq \frac{3m}{4}$. For any solution s with $|A(s)| < \frac{3m}{4}$, we have $\Delta \geq \frac{\frac{m}{4}-4}{2 \cdot e \cdot m} \geq 0.0075$, since we have assumed $m > 20$. By additive drift argument [21], we can see that a solution with $|A(s)| \geq \frac{3m}{4}$ is found in expected time $\frac{1}{0.0075} \cdot \frac{3m}{4} = 100m$. By Markov's inequality, with

probability at least $\frac{1}{2}$, the time until finding that solution is at most $200m$. Therefore, in a phase of $m^{1+\varepsilon}$ steps, the probability of not finding that solution is $(\frac{1}{2})^{\frac{m^\varepsilon}{200}} = e^{-\Omega(m^\varepsilon)}$. □

In the proof of the next lemma, we use the Simplified Drift Theorem (Theorem 11) presented in [28, 29]. In this theorem, F_t denotes a filtration on states. In the proof of Lemma 12, we analyse the changes on the size of $A(s)$, and no filtration is applied on the steps.

Theorem 11. *(Simplified Drift Theorem [29]) Let X_t , $t \geq 0$, be real-valued random variables describing a stochastic process over some state space. Suppose there exist an interval $[a, b] \subseteq \mathbb{R}$, two constants $\delta, \varepsilon > 0$ and, possibly depending on $l := b - a$, a function $r(l)$ satisfying $1 \leq r(l) = o(l/\log(l))$ such that for all $t \geq 0$ the following two conditions hold:*

1. $E[X_{t+1} - X_t \mid F_t \wedge a < X_t < b] \geq \varepsilon$,

2. $\Pr(|X_{t+1} - X_t| \geq j \mid F_t \wedge a < X_t) \leq \frac{r(l)}{(1+\delta)^j}$ for $j \in \mathbb{N}$.

Then there is a constant $c^* > 0$ such that for $T^* := min\{t \geq 0 : X_t \leq a | F_t \wedge X_0 \geq b\}$ it holds $\Pr(T^* \leq 2^{c^* l/r(l)}) = 2^{-\Omega(l/r(l))}$.

Lemma 12. *For sufficiently large m, with probability $1 - e^{-\Omega(m)}$, Property 9 holds during Phase 2.*

Proof. Phase 2 starts with the solution s with $|A(s)| \geq \frac{3m}{4}$, found by the end of Phase 1. Using Simplified Drift Theorem (Theorem 11) with parameters $\delta = 1$, $r(l) = 1$ and interval $[a, b] = [\frac{m}{2}, \frac{3m}{4}]$, we show that with high probability, a solution s' with $|A(s')| \leq \frac{m}{2}$ is not found by the algorithm until end of Phase 2.

Let $X_t = |A(s)|$, where s is the solution obtained at time t. The total drift on the value of X_t is Δ of the proof of Lemma 10, which is at least 0.0075 when $X_t \leq \frac{3m}{4}$. Therefore, the two conditions of the Simplified Drift Theorem hold:

1. $\mathrm{E}(X_{t+1} - X_t \mid a \leq X_t \leq b) = \mathrm{E}(X_{t+1} - X_t \mid \frac{m}{2} \leq X_t \leq \frac{3m}{4}) \geq 0.0075$, and
2. $\Pr(|X_{t+1} - X_t| \geq j \mid a \leq X_t) \leq \frac{1}{j!e} \leq \frac{1}{2^j} = \frac{r(l)}{(1+\delta)^j}$

The inequality regarding the second condition holds, because the probability of mutating j edges at one step follows the Poisson distribution and is $\frac{1}{j!e}$. Having these two conditions satisfied, the Simplified Drift Theorem says that the probability of finding a solution with $|A(s)| \leq \frac{m}{2}$ in time $2^{\frac{c^* m}{4}}$, $c^* > 0$ a constant, is at most $2^{-\Omega(\frac{m}{4})}$. This implies that with probability $1 - e^{-\Omega(m)}$, such a solution is not found by the end of Phase 2 which consists of ${c_1}^{m^{\varepsilon/2}} = 2^{\log_2 c_1 \cdot m^{\varepsilon/2}}$ steps. □

Lemma 13. *Let $\varepsilon \leq 1/3$ be a positive constant. In Phase 1, with probability $1 - e^{-\Omega(m^\varepsilon)}$, the (1+1) EA with Step Size Adaptation does not reach a solution where $s_1 > 2 \cdot m^\varepsilon \cdot {c_1}^{2 \cdot m^\varepsilon}$. Moreover, the step size of s_1 does not exceed ${c_1}^{2 \cdot m^\varepsilon}$, i. e. $\sigma_1 \leq {c_1}^{2 \cdot m^\varepsilon}$.*

Proof. From Lemma 10, we know that this phase is at most $m^{1+\varepsilon}$ steps. Let X be the number of times that the first edge is selected for mutation during Phase 1. Since the probability of selecting each edge at each step is $\frac{1}{m}$, the expected value of X is at most m^ε. Moreover, since probability of selecting edges are independent of each other, by Chernoff bounds we have:

$$\Pr(X \geq 2 \cdot m^\varepsilon) \leq e^{-m^\varepsilon/3}.$$

Therefore, with probability $1 - e^{-\Omega(m^\varepsilon)}$ the first edge is not selected for mutation more than $2 \cdot m^\varepsilon$ times, which means that the step size for that edge is at most ${c_1}^{2 \cdot m^\varepsilon}$ after that phase. This implies that $2 \cdot m^\varepsilon \cdot {c_1}^{2 \cdot m^\varepsilon}$ is an upper bound for the value of s_1 by the end of Phase 1. Note that s_1 and σ_1 have not reached their maximal values, since $\varepsilon \leq 1/3$. □

In the following lemma, we show that when $|A(S)| \geq m/2$, the probability of decreasing the step size σ_1 is larger than the probability of increasing it. This lemma is used in Theorem 16 to show that we do not reach large values of σ_1 in polynomial time.

Lemma 14. *Assuming that Property 9 holds, and also assuming that $\sigma_1 > m$ and $s_1 \leq W_{\max}$, at any step where σ_1 is changed by (1+1) EA with Step Size Adaptation, it is increased with probability $P_{inc} < 0.4$ and decreased with probability $P_{dec} > 0.6$.*

Proof. The value of σ_1 changes at the steps where e_1 is selected for mutation. All other steps make no change on σ_1. Here we only consider the steps at which e_1 is selected for a mutation.

The value of σ_1 increases when a mutation on e_1 is accepted. Since $\sigma_1 > m$ and $s_1 \leq W_{\max}$, any mutation that decreases the value of e_1 is rejected. Since we have assumed that Property 9 holds, there are at least $\frac{m}{2}$ edges other than e_1, with a weight of one. A mutation of increase on these edges is rejected. Therefore, an increase on e_1 is also rejected if one of those edges is selected for an increase in addition to e_i at the same step. The probability that an increase is selected to be done on e_1, while none of those edges are selected for increase, is:

$$\frac{1}{2} \cdot \left(1 - \frac{1}{2m}\right)^{m - \frac{m}{2}} \leq \frac{1}{2} \cdot \left(\frac{1}{e}\right)^{\frac{1}{4}} < 0.4$$

This probability is an upper bound for the probability that an acceptable increase on e_1 happens, which is denoted by P_{inc}. In other words:

$$P_{inc} < 0.4$$

Since $P_{inc} + P_{dec} = 1$ at steps where a mutation happens on e_1, we have $P_{dec} > 0.6$. □

In order to prove the main theorem of this section, we use the Gambler's Ruin Theorem, introduced by Feller [16]. We use the parameter settings of a variant of this theorem (Theorem 15) presented in [20].

Theorem 15 (Gambler's Ruin Theorem)**.** *[20]*
Let p be the probability of winning one dollar and $q = 1 - p$ be the probability of loosing one dollar in a single bet and let $\delta = q/p$. Starting with x dollars, the probability of reaching $z > x$ dollars before attaining zero dollars is

$$P_x = \frac{\delta^x - 1}{\delta^z - 1}$$

Theorem 16. *For sufficiently large m and a positive constant $\varepsilon \leq \frac{1}{3}$, with probability $1 - e^{-\Omega(m^{\varepsilon/2})}$, the required time for (1+1) EA with Step Size Adaptation (Algorithm 4) to find a 2-approximation on G with $W_{\max} = {c_1}^m$ is lower bounded by $2^{m^{\varepsilon/2}}$, when $c_1 = c_2$.*

Proof. According to Lemma 13, during Phase 1, with probability $1 - e^{-\Omega(m^\varepsilon)}$, we have $\sigma_1 \leq {c_1}^{2 \cdot m^\varepsilon}$. Using Lemma 14 and the Gambler's Ruin Theorem, we prove that with high probability, in Phase 2, we always have $\sigma_1 \leq {c_1}^{m^{2\varepsilon}}$.

Due to Lemma 12, with probability $1-e^{-\Omega(m)}$, Property 9 holds during Phase 2 which is a requirement of Lemma 14. However, Lemma 14 can only be used for the steps where $\sigma_1 > m$, while Phase 2 may start with $\sigma_1 \leq m$. Nevertheless, in order to reach large values of ${c_1}^{m^{2\varepsilon}}$ or greater, at some point of Phase 2, we need to deal with a situation where $m < \sigma_1 \leq c_1 m$, since σ_1 increases at each step at most by a factor of c_1. According to Lemma 14, at the steps in which e_1 is selected for mutation, the probability of increasing σ_1 is $p \leq 0.4$ and the probability of decreasing it is $q \geq 0.6$.

Let σ_1^0 be the value of σ_1 at the first point in Phase 2 where $m < \sigma_1 \leq c_1 m$. If $\sigma_1 \leq c_1 m$ holds, then for sufficiently large m we also have $\sigma_1 \leq {c_1}^{2 \cdot m^{\varepsilon}}$. Starting from that point where $m < \sigma_1 \leq {c_1}^{2 \cdot m^{\varepsilon}}$, we investigate whether the algorithm reaches a situation where $\sigma_1 \leq m$ earlier than a situation where $\sigma_1 \geq {c_1}^{m^{2\varepsilon}}$.

Every time σ_1 is increased, it is increased by a factor of c_1 and every time that it is decreased, it is decreased by a factor of c_2. Since we have assumed that $c_1 = c_2$, one increasing step and one decreasing step cancel each other and the problem can be mapped to the problem of Gambler's Ruin Theorem (Theorem 15) with parameters p and q described above and $\delta = \frac{q}{p} \geq \frac{0.6}{0.4} > 1$. The number of times that $\sigma_1 = \sigma_1^0$ needs to be decreased to reach $\sigma_1 \leq m$ is at most

$$\lceil \log_{c_2}(\sigma_1^0/m) \rceil \leq \log_{c_2}\left(\frac{{c_1}^{2 \cdot m^{\varepsilon}}}{m}\right) + 1 \leq 2 \cdot m^{\varepsilon} + 1$$

Also, the number of times that $\sigma_1 \leq m$ needs to be increased to reach $\sigma_1 \geq {c_1}^{m^{2\varepsilon}}$ is at least

$$\lceil \log_{c_1}({c_1}^{m^{2\varepsilon}}/m) \rceil \geq m^{2\varepsilon} - \lfloor \log_{c_1} m \rfloor$$

Therefore, other parameters of the Gambler's Ruin Theorem would be $x \leq 2 \cdot m^{\varepsilon} + 1$ and $z \geq m^{2\varepsilon} - \lfloor \log_{c_1} m \rfloor$. Using that theorem, we get P_x, the probability of reaching a state where $\sigma_1 \geq {c_1}^{m^{2\varepsilon}}$ before reaching a state where $\sigma_1 \leq m$ as:

$$P_x = \frac{(\delta)^x - 1}{(\delta)^z - 1} \leq \frac{(\delta)^{2 \cdot m^{\varepsilon}+1} - 1}{(\delta)^{m^{2\varepsilon} - \lfloor \log_{c_1} m \rfloor} - 1} = e^{-\Omega(m^{\varepsilon})}.$$

Consider a phase of $2^{m^{\varepsilon/2}}$ steps. We here show that with probability $e^{-\Omega(m^{\varepsilon/2})}$, $\sigma_1 \geq {c_1}^{m^{2\varepsilon}}$ during this phase.

We saw that with probability $1-e^{-\Omega(m^{\varepsilon})}$ we reach a state where $\sigma_1 \leq m$ before a state where $\sigma_1 \geq {c_1}^{m^{2\varepsilon}}$. If σ_1 never increases to $c_1^{m^{\varepsilon}}$ after that, then we never reach a state where $\sigma_1 \geq {c_1}^{m^{2\varepsilon}}$. Otherwise, it spends at least

$$\lceil \log_{c_1}({c_1}^{m^{\varepsilon}}/m) \rceil = m^{\varepsilon} - \lfloor \log_{c_1} m \rfloor$$

steps to reach $c_1^{m^{\varepsilon}}$. In a phase of $2^{m^{\varepsilon/2}}$ steps, there are at most

$$k = \frac{2^{m^{\varepsilon/2}}}{m^{\varepsilon} - \lfloor \log_{c_1} m \rfloor}$$

times that σ_1 increases to $c_1^{m^{\varepsilon}}$, and probability of reaching ${c_1}^{m^{2\varepsilon}}$ from there is only $e^{-\Omega(m^{\varepsilon})}$. Therefore, the probability of σ_1 to reach ${c_1}^{m^{2\varepsilon}}$ at least once in a phase of $2^{m^{\varepsilon/2}}$ steps, is at most

$$k \cdot e^{-\Omega(m^{\varepsilon})} = e^{-\Omega(m^{\varepsilon/2})}.$$

So far we have proved that with probability $1-e^{-\Omega(m^{\varepsilon/2})}$, $\sigma_1 \leq {c_1}^{m^{2\varepsilon}}$ during Phase 2 which consists of ${c_1}^{m^{\varepsilon/2}}$ steps. Moreover, according to Lemma 13, with probability $1 - e^{-\Omega(m^{\varepsilon})}$, we have $s_1 \leq 2 \cdot m^{\varepsilon} \cdot {c_1}^{2m^{\varepsilon}}$ by the end of Phase 1. Therefore, the value of s_1 during both phases is always upper bounded by

$$2 \cdot m^{\varepsilon} \cdot {c_1}^{2m^{\varepsilon}} + {c_1}^{m^{\varepsilon/2}} \cdot {c_1}^{m^{2\varepsilon}}$$

which is less than $W_{\max}$, since $\varepsilon \leq 1/3$. Therefore, with probability $1 - e^{-\Omega(m^{\varepsilon/2})}$, the (1+1) EA with Step Size Adaptation does not find a 2-approximation in time $2^{m^{\varepsilon/2}}$. □

Note that for $c_1 < c_2$, the probability of reaching a situation where $\sigma_1 \geq {c_1}^{m^{2\varepsilon}}$ before reaching $\sigma_1 < m$ is even smaller, since the number of increasing steps that are required to cancel one decreasing step is greater than one. Therefore, this situation is in favour of reaching $\sigma_1 \leq m$, resulting in the following corollary.

Corollary 17. *For sufficiently large m and a positive constant $\varepsilon \leq \frac{1}{3}$, with probability $1 - e^{-\Omega(m^{\varepsilon/2})}$, the required time for (1+1) EA with Step Size Adaptation (Algorithm 4) to find a 2-approximation of G is lower bounded by $2^{m^{\varepsilon/2}}$, when $c_1 \leq c_2$.*

6. CONCLUSION

In this paper, we have considered how to solve the minimum vertex cover problem by its dual formulation based on a multi-valued edge-based encoding. We have proven pseudo-polynomial upper bounds for RLS and the (1+1) EA until they have achieved a 2-approximation. Furthermore, we have investigated the use of step-size adaptation in both algorithms and shown that RLS with step size adaptation obtains a 2-approximation in expected polynomial time; whereas the corresponding (1+1) EA still encounters a pseudo-polynomial lower bound.

Acknowledgments

This research has been supported through Australian Research Council grants DP140103400 and DP160102401.

References

[1] R. Aleliunas, R. M. Karp, R. J. Lipton, L. Lovász, and C. Rackoff. Random walks, universal traversal sequences, and the complexity of maze problems. In *Proceedings of the 20th Annual Symposium on Foundations of Computer Science (FOCS '79)*, pages 218–223. IEEE Press, 1979.

[2] A. Auger and B. Doerr. *Theory of Randomized Search Heuristics: Foundations and Recent Developments*. World Scientific Publishing Co., Inc., 2011.

[3] R. Bar-Yehuda and S. Even. A linear-time approximation algorithm for the weighted vertex cover problem. *Journal of Algorithms*, 2(2):198–203, 1981.

[4] H. Beyer and H. Schwefel. Evolution strategies - A comprehensive introduction. *Natural Computing*, 1(1):3–52, 2002.

[5] S. Böttcher, B. Doerr, and F. Neumann. Optimal fixed and adaptive mutation rates for the leadingones problem. In *Conference on Problem Solving from Nature (PPSN)*, pages 1–10, 2010.

[6] K. L. Clarkson. A modification of the greedy algorithm for vertex cover. *Inf. Process. Lett.*, 16(1):23–25, 1983.

[7] D.-C. Dang and P. K. Lehre. Self-adaptation of mutation rates in non-elitist populations. In *Conference on Problem Solving from Nature (PPSN)*, 2016.

[8] B. Doerr and C. Doerr. Optimal parameter choices through self-adjustment: Applying the 1/5-th rule in discrete settings. In *Genetic and Evolutionary Computation Conference (GECCO)*, pages 1335–1342, 2015.

[9] B. Doerr, C. Doerr, and T. Kötzing. The right mutation strength for multi-valued decision variables. In *Genetic and Evolutionary Computation Conference (GECCO)*, 2016.

[10] B. Doerr, C. Doerr, and J. Yang. k-bit mutation with self-adjusting k outperforms standard bit mutation. In *Parallel Problem Solving from Nature – PPSN XIV: 14th International Conference, Edinburgh, UK, September 17-21, 2016, Proceedings*, pages 824–834. Springer International Publishing, 2016.

[11] B. Doerr, D. Johannsen, and M. Schmidt. Runtime analysis of the (1+1) evolutionary algorithm on strings over finite alphabets. In *Workshop on Foundations of Genetic Algorithms (FOGA)*, pages 119–126, 2011.

[12] B. Doerr, D. Johannsen, and C. Winzen. Multiplicative drift analysis. *Algorithmica*, 64(4):673–697, 2012.

[13] B. Doerr and S. Pohl. Run-time analysis of the (1+1) evolutionary algorithm optimizing linear functions over a finite alphabet. In *Genetic and Evolutionary Computation Conference (GECCO)*, pages 1317–1324, 2012.

[14] D.-Z. Du, K.-I. Ko, and X. Hu. *Design and Analysis of Approximation Algorithms*. Springer Publishing Company, Incorporated, 2011.

[15] A. E. Eiben, Z. Michalewicz, M. Schoenauer, and J. E. Smith. Parameter control in evolutionary algorithms. In *Parameter Setting in Evolutionary Algorithms*, volume 54 of *Studies in Computational Intelligence*, pages 19–46. Springer, 2007.

[16] W. Feller. *An Introduction to Probability Theory and Its Applications*, volume 1. Wiley, 3rd edition, 1968.

[17] T. Friedrich, J. He, N. Hebbinghaus, F. Neumann, and C. Witt. Analyses of simple hybrid algorithms for the vertex cover problem. *Evolutionary Computation*, 17(1):3–19, 2009.

[18] T. Friedrich, J. He, N. Hebbinghaus, F. Neumann, and C. Witt. Approximating covering problems by randomized search heuristics using multi-objective models. *Evolutionary Computation*, 18(4):617–633, 2010.

[19] N. Hansen, S. D. Müller, and P. Koumoutsakos. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). *Evolutionary Computation*, 11(1):1–18, 2003.

[20] E. Happ, D. Johannsen, C. Klein, and F. Neumann. Rigorous analyses of fitness-proportional selection for optimizing linear functions. In *Conference on Genetic and Evolutionary Computation (GECCO)*, pages 953–960, 2008.

[21] J. He and X. Yao. Drift analysis and average time complexity of evolutionary algorithms. *Artificial Intelligence*, 127(1):57–85, 2001.

[22] T. Jansen. *Analyzing Evolutionary Algorithms – The Computer Science Perspective*. Natural Computing Series. Springer, 2013.

[23] T. Jansen, P. S. Oliveto, and C. Zarges. Approximating vertex cover using edge-based representations. In *Workshop on Foundations of Genetic Algorithms (FOGA)*, pages 87–96, 2013.

[24] T. Kötzing, A. Lissovoi, and C. Witt. (1+1) EA on generalized dynamic onemax. In *Workshop on Foundations of Genetic Algorithms (FOGA)*, pages 40–51, 2015.

[25] S. Kratsch and F. Neumann. Fixed-parameter evolutionary algorithms and the vertex cover problem. *Algorithmica*, 65(4):754–771, 2013.

[26] J. Lässig and D. Sudholt. Adaptive population models for offspring populations and parallel evolutionary algorithms. Technical report, https://arxiv.org/abs/1102.0588, 2011.

[27] F. Neumann and C. Witt. *Bioinspired Computation in Combinatorial Optimization:Algorithms and Their Computational Complexity*. Springer-Verlag New York, Inc., New York, NY, USA, 1st edition, 2010.

[28] P. Oliveto and C. Witt. Simplified drift analysis for proving lower bounds in evolutionary computation. *Algorithmica*, 59(3):369–386, 2011.

[29] P. Oliveto and C. Witt. Erratum: Simplified drift analysis for proving lower bounds in evolutionary computation. *arXiv*, abs/1211.7184, 2012.

[30] P. S. Oliveto, J. He, and X. Yao. Analysis of the (1+1) -ea for finding approximate solutions to vertex cover problems. *IEEE Trans. Evolutionary Computation*, 13(5):1006–1029, 2009.

[31] V. V. Vazirani. *Approximation Algorithms*. Springer-Verlag New York, Inc., New York, NY, USA, 2001.

[32] I. Wegener. *Methods for the Analysis of Evolutionary Algorithms on Pseudo-Boolean Functions*, pages 349–369. Springer US, Boston, MA, 2002.