Runtime Analysis of Randomized Search Heuristics for the Dynamic Weighted Vertex Cover Problem

Feng Shi
Central South University
School of Info. Sci. and Engineering
Changsha, China

Frank Neumann
The University of Adelaide
School of Computer Science
Adelaide, Australia

Jianxin Wang
Central South University
School of Info. Sci. and Engineering
Changsha, China

ABSTRACT

Randomized search heuristics such as evolutionary algorithms are frequently applied to dynamic combinatorial optimization problems. Within this paper, we present a dynamic model of the classic Weighted Vertex Cover problem and analyze the performances of the two well-studied algorithms Randomized Local Search and (1+1) EA for it, to contribute to the theoretical understanding of evolutionary computing for problems with dynamic changes. In our investigations, we use an edge-based representation based on the dual formulation of the problem and study the expected runtimes that the two algorithms require to maintain a 2-approximate solution when the given weighted graph is modified by an edge-editing or weight-editing operation. Considering the weights on the vertices may be exponentially large with respect to the size of the graph, the step size adaption strategy is incorporated. Our results show that both algorithms can recompute 2-approximate solutions for the studied dynamic changes efficiently.

CCS CONCEPTS

•Mathematics of computing → **Evolutionary algorithms;** **•Theory of computation** → *Random search heuristics;* **•General and reference** → General conference proceedings;

KEYWORDS

runtime analysis, weighted vertex cover, dynamic graph-editing, evolutionary algorithm

1 INTRODUCTION

Over the past decades, randomized search heuristics such as evolutionary algorithms and ant colony optimization have been applied successfully in various areas, including engineering and economics. To understand well the behaviors of evolutionary algorithms, many theoretical techniques for analyzing their expected runtimes are presented [1, 10, 16]. And using these techniques, evolutionary algorithms designed for some classic combinatorial optimization problems have been studied. In particular, the Vertex Cover problem plays a crucial role in the area [6, 8, 11, 15, 20].

Given an instance of a considered combinatorial optimization problem, and a solution (optimal or approximated) to the instance. If an operation on the instance results in a new instance, which is not very "far away" from the original one (the "distance" between the two instances depends on the operation), then an interesting problem arises: is the solution to the new instance "far away" from the original solution? In other words, how much runtime does a specific algorithm take to get the target solution starting with the original one? This setting is referred as dynamic combinatorial optimization problems.

Studying the performances of evolutionary algorithms for dynamic combinatorial optimization problems is an emerging field [7, 14, 17, 19, 21]. Within this paper, we present a dynamic model of the Weighted Vertex Cover problem (WVC), which is named Dynamic Weighted Vertex Cover problem (DWVC). Our goal is to analyze the behaviors of the well-studied algorithms Randomized Local Search (RLS) and (1+1) EA designed for it. Specifically, we study the expected runtimes (the expected number of fitness evaluations) they need to recompute a 2-approximate solution when the given weighted graph is edited by a graph-editing operation, starting with a 2-approximate solution to the original graph.

For the Vertex Cover problem, it is well-known that under the unique games conjecture [12], there does not exist an approximation algorithm with a constant ratio $r < 2$, unless P = NP [13]. The best-known 2-approximation algorithm for the Vertex Cover problem is based on the maximal matching: construct a maximal matching by greedily adding edges, then let the vertex cover contain both endpoints of each edge in the matching. For WVC, Hochbaum [9] presented the best-known approximation algorithm, who showed that a 2-approximate solution can be found by using the Linear Programming (LP) result of the Fractional WVC. Du et al. [5] found that a maximal solution to the dual form [22] of the LP formulation (simply called dual formulation) for the Fractional WVC also directly induces a 2-approximate solution. Using this conclusion, Bar-Yehuda and Evan [2] presented a linear-time 2-approximation algorithm for WVC. The essential difference between (the primal form of) the LP formulation and the dual formulation for the Fractional WVC is: the LP formulation considers the problem from the perspective of vertices; the dual formulation considers that from the perspective of edges.

Pourhassan et al. [19] presented a dynamic model of the Vertex Cover problem, in which the graph editing operator adds (or removes) exactly one edge into (or from) the given unweighted graph, and analyzed evolutionary algorithms with respect to their abilities to maintain a 2-approximate solution (i.e., a maximal matching). They examined different variants (node-based representation and edge-based representation) of RLS and (1+1) EA. If using the

GECCO '18, Kyoto, Japan

978-x-xxxx-xxxx-x/YY/MM...$15.00
DOI: 10.1145/nnnnnnn.nnnnnnn

node-based representation, they gave classes of instances where both algorithms behave badly (the two algorithms cannot get the 2-approximate solution in polynomial time with high probability). If using the edge-based representation, they showed that RLS and (1+1) EA can maintain 2-approximations easily if the algorithms start with a maximal matching of the original unweighted graph and use the fitness function given by Jansen et al. [11], which penalizes edges sharing vertices (RLS can maintain the quality of the solution in linear time $O(m)$ when an edge is added or deleted; (1+1) EA can maintain the quality of the solution in linear time $O(m)$ when an edge is added).

Inspired by the work of Pourhassan et al. [19] and the essential difference between the LP formulation and dual formulation for the Fractional WVC, we utilize the dual formulation to analyze DWVC. Thus DWVC studied in this paper, is formulated as: given a weighted graph $G = (V, E, W)$ and a maximal solution to the dual formulation of the Fractional WVC on G, the goal is to find a maximal solution to the dual formulation of the Fractional WVC on the weighted graph $G^* = (V^*, E^*, W^*)$, where G^* is obtained by one of the following four graph-editing operations on G: (1) add a new edge-set E^+ into E; (2) remove an edge-subset E^- from E; (3) increase the weights on the vertices in $V^+ \subseteq V$; (4) decrease the weights on the vertices in $V^- \subseteq V$. To study the influences of the graph-editing operations on the performances of the algorithms, we denote the exact sizes of E^+, E^-, V^+, and V^- by variable $D \in \mathbb{N}^+$.

Recently Pourhassan et al. [18] studied WVC using the dual formulation for the Fractional WVC. Considering the weights on the vertices may be exponentially large with respect to the size of the weighted graph, they incorporated the *Step Size Adaption* strategy [3] into their (1+1) EA (see Algorithm 4 in their paper). However, their (1+1) EA was shown to take exponential expected runtime with high probability to get a 2-approximate solution. There are two reasons for the long runtime of their algorithm. First, for a mutation M constructed by their (1+1) EA, there may exist two edges selected by M whose weights are increased and decreased respectively. The randomness leads to the relatively small probability for M to be accepted. Second, for a mutation M that is rejected by their (1+1) EA, the step sizes of all the edges selected by M would be decreased. Because of the two reasons, the step sizes of the edges cannot be increased enough to overcome the exponentially large weights on the vertices. That means, the step size adaptation strategy is nearly invalid for their (1+1) EA.

Drawing on the experience of Pourhassan et al. [18], we give two algorithms (1+1) EA and RLS for DWVC, with step size adaption as well. The two algorithms employ a novel way to prevent the invalidation of the step size adaption strategy that happens in the algorithm of [18]. We show that the (1+1) EA and RLS take expected runtime $O(\alpha m \log_\alpha^2 (D \cdot W_{max}))$ to solve the four versions of DWVC, where m denotes the number of edges in G^*, $W_{max} \geq 1$ denotes the maximum weight that the vertices in G and G^* have, and $\alpha \in \mathbb{N}^+$ denotes the increasing/decreasing rate of the step size (i.e., the increment of the weight on each edge can be exponentially increased by multiplying α, or decreased by multiplying $1/\alpha$).

The rest of the paper is structured as follows. We start by giving related definitions and the problem formulations in Section 2. Then we present the (1+1) EA and RLS for DWVC in Section 3, and analyze their expected runtimes in Section 4. Finally conclusions are presented in Section 5.

2 PRELIMINARIES

Given a weighted graph $G = (V, E, W)$ with a vertex-set $V = \{v_1, \ldots, v_n\}$, an edge-set $E = \{e_1, \ldots, e_m\}$, and a weight function $W : V \rightarrow \mathbb{N}^+$. For any vertex $v \in V$, denote by $N_G(v)$ the set containing all neighbors of v in G, and by $E_G(v)$ the set containing all edges incident to v in G. For any vertex-subset $V' \subseteq V$, let $E_G(V') = \bigcup_{v \in V'} E_G(v)$. For any edge $e \in E$, denote by $E_G(e)$ the set containing all edges in G that have a common endpoint with e. For any edge-subset $E' \subseteq E$, let $E_G(E') = \bigcup_{e \in E'} E_G(e) \setminus E'$.

A vertex-subset $V_c \subseteq V$ is a *vertex cover* of G if for each edge $e \in E$, where e can be represented by its two endpoints v and v' as $[v, v']$, at least one of v and v' is in V_c. The weight of V_c is defined as the sum of the weights on all vertices in V_c, written $\sum_{v \in V_c} W(v)$. The Weighted Vertex Cover problem (WVC) on the weighted graph G asks for a vertex cover of G with the minimum weight.

Using the node-based representation (i.e. the search space is $\{0,1\}^n$ and for any solution $x = x_1 \ldots x_n$ the node v_i is chosen iff $x_i = 1$), the Integer Linear Programming (ILP) formulation for WVC is given as follows.

$$\begin{aligned} & min \sum_{i=1}^{n} W(v_i) \cdot x_i \\ st. \quad & x_i + x_j \geq 1 \quad \forall\, [v_i, v_j] \in E \\ & x_i \in \{0,1\} \quad i = 1, \ldots, n \end{aligned}$$

By relaxing the constraint $x_i \in \{0,1\}$ to $x_i \in [0,1]$, the Linear Programming (LP) formulation for the Fractional WVC is obtained. Hochbaum [9] showed that a 2-approximate solution can be found by using the LP result of the Fractional WVC — include all vertices v_i for which $x_i \geq \frac{1}{2}$. The dual form of the LP formulation (or called dual formulation) for the Fractional WVC is given as follows, where $Y : E \rightarrow \mathbb{R}^+ \cup \{0\}$ denotes a weight assignment on the edges.

$$\begin{aligned} & max \sum_{e \in E} Y(e) \\ st. \quad & \sum_{e \in E_G(v)} Y(e) \leq W(v) \quad \forall\, v \in V \end{aligned}$$

The weight assignment Y is called a *dual-solution* of G in the paper. Vertex $v \in V$ *satisfies* the *dual-LP constraint* with respect to the dual-solution Y if $\sum_{e \in E_G(v)} Y(e) \leq W(v)$. Edge $e \in E$ *satisfies* the dual-LP constraint with respect to Y if both its endpoints satisfy the dual-LP constraint with respect to Y. The dual-solution Y of G is *feasible* if all vertices in G satisfy the dual-LP constraint with respect to Y, otherwise, *infeasible*. Vertex $v \in V$ is *tight* with respect to Y if $\sum_{e \in E_G(v)} Y(e) = W(v)$, and edge $e \in E$ is *tight* with respect to Y if at least one of its two endpoints is tight with respect to Y.

Given a dual-solution Y of G, denote by $V_G(Y)$ the set containing all vertices in G that do not satisfy the dual-LP constraint with respect to Y, and by $E_G(Y)$ the set containing all edges in G that are incident to $v \in V_G(Y)$.

A *maximal feasible dual-solution* (MFDS) of G is a feasible dual-solution such that none of the edges can be assigned with a larger weight under the dual-LP constraint. For any MFDS Y of G, it induces a vertex cover of G with ratio 2 directly, which contains all tight vertices with respect to Y (a formal proof about the ratio can be found in Theorem 8.4 of [5]).

Four versions of DWVC are studied in this paper. They are all given a weighted graph $G = (V, E, W)$, an MFDS Y_{orig} of G, and a graph-editing operation. Their aims are to find an MFDS of G^*, where G^* is the weighted graph obtained by the corresponding operation on G. Due to the space limit, their full formulations are not given, only the corresponding graph-editing operations.

(1) **DWVC-E$^+$**: add a new edge-set E^+ into E;
(2) **DWVC-E$^-$**: remove an edge-subset E^- from E;
(3) **DWVC-W$^+$**: increase the weights on the vertices in $V^+ \subseteq V$ (i.e., $W^+(v) > W(v)$ for each $v \in V^+$, and $W^+(v) = W(v)$ for each vertex $v \in V \setminus V^+$);
(4) **DWVC-W$^-$**: decrease the weights on the vertices in $V^- \subseteq V$ (i.e., $W^-(v) < W(v)$ for each $v \in V^-$, and $W^-(v) = W(v)$ for each vertex $v \in V \setminus V^-$).

3 ADAPTIVE (1+1) EA AND RLS

Given two weighted graphs $G = (V, E, W)$ and $G^* = (V^*, E^*, W^*)$, where G^* is obtained by one of the four graph-editing operations on G. We study the runtimes that the (1+1) EA and RLS considered in the paper need to find an MFDS of G^*, starting with a given MFDS Y_{orig} of G (not from scratch). The general idea of the (1+1) EA and RLS is: if the given MFDS Y_{orig} of G is also a feasible dual-solution of G^*, then the two algorithms directly increase the weights on the edges in G^* until the weight on any edge cannot be assigned with a larger value under the dual-LP constraint (i.e, an MFDS of G^* is found if the claimed condition is met); otherwise, they first decrease the weights on the edges in $E_{G^*}(Y_{\text{orig}})$, aiming to get a feasible dual-solution Y_t of G^* as soon as possible (because only the vertices in $V_{G^*}(Y_{\text{orig}})$ do not satisfy the dual-LP constraint with respect to Y_{orig}), then increase the weights on the edges in G^* to get an MFDS based on Y_t. Thus we give a sign function below, written $sign(Y)$, to judge whether the considered solution Y is a feasible dual-solution of G^*. According to $sign(Y)$, the two algorithms know what they should do at the next step, increasing or decreasing the weights on the edges.

$$sign(Y) = \begin{cases} -1 & \text{if } V_{G^*}(Y) \neq \emptyset \\ 1 & \text{otherwise} \end{cases}$$

Note that for any mutation generated by the two algorithms, the weights on the edges selected by the mutation are either all increased or all decreased. Thus we always have that

$$sign(Y) \cdot \sum_{e \in E^*} (Y'(e) - Y(e)) \geq 0 ,$$

where Y' is the dual-solution obtained by a mutation M on Y. Consequently, the function $f(Y', Y)$ comparing the fitness of Y' and Y is only required to pay more attention to the feasibilities of Y and Y', and the edges whose weights are changed by the mutation M: $f(Y', Y) \geq 0$ if Y' is not worse than Y; $f(Y', Y) < 0$ otherwise.

$$f(Y', Y) = 2|E^*| \cdot W_{max} \cdot (sign(Y) - 1) \cdot \sum_{e \in E^* \setminus E_{G^*}(Y)} (Y(e) - Y'(e)) + (sign(Y') - sign(Y))$$

As the general idea of the two algorithms given above, if Y is a feasible dual-solution of G^*, then the two algorithms directly increase the weights on the edges, aiming to get an offspring Y' of Y that is a feasible dual-solution of G^* such that $\sum_{e \in E^*} Y'(e) \geq \sum_{e \in E^*} Y(e)$. Thus, if Y' is infeasible, then let

$$f(Y', Y) = sign(Y') - sign(Y) = -2 < 0 ,$$

otherwise,

$$f(Y', Y) = sign(Y') - sign(Y) = 0 .$$

If Y is an infeasible dual-solution of G^*, then they decrease the weights on the edges firstly, aiming to get a feasible dual-solution of G^*. Remark that the edges in $E^* \setminus E_{G^*}(Y)$ satisfy the dual-LP constraint with respect to Y, so the weights on these edges do not need to be decreased. Thus the first term of $f(Y', Y)$,

$$2|E^*| \cdot W_{max} \cdot (sign(Y) - 1) \cdot \sum_{e \in E^* \setminus E_{G^*}(Y)} (Y(e) - Y'(e)) ,$$

penalizes the mutation that decreases the weights on the edges in $E^* \setminus E_{G^*}(Y)$, which hence guides the mutation to decrease only the weights on the edges in $E_{G^*}(Y)$ (if we do not restrict that the weights on the edges in $E^* \setminus E_{G^*}(Y)$ cannot be decreased, then the feasible dual-solution we get may be further away from the MFDS of G^*). Once a feasible dual-solution Y' is found by a mutation on the unfeasible dual-solution Y, then

$$f(Y', Y) = sign(Y') - sign(Y) = 2 \geq 0 .$$

The two algorithms for DWVC, (1+1) EA and RLS, are given in Algorithm 1 and 2 respectively. They run in a similar way, except the mechanism selecting edges for mutation. The (1+1) EA selects each edge in E^* with probability $1/m$ at each iteration ($m = |E^*|$), resulting an edge-subset I containing all selected edges (see step 8 of the (1+1) EA), and increases (or decreases) the weights on the edges in I. RLS differs from the (1+1) EA by selecting exactly one edge in $e \in E^*$ in each round.

Algorithm 1: (1+1) EA

```
1  Initialize Y and σ ;
2  Determine sign(Y) ;
3  while the termination not satisfied do
4      Y' := Y and I := ∅ ;
5      for each edge e ∈ E* do
6          with probability 1/m do
7              Y'(e) := max{Y(e) + σ(e) · sign(Y), 0} ;
8              I := I ∪ {e} ;
9      Determine sign(Y') and f(Y', Y) ;
10     if f(Y', Y) ≥ 0 then
11         Y := Y' ;
12         σ(e) := α · σ(e) for all e ∈ I ;
13     else
14         if sign(Y) > 0 then
15             Let I' be the subset of I such that each edge e ∈ I'
               has an endpoint that violates the dual-LP
               constraint in Y', and no other edge in I shares the
               endpoint with e ;
16             and σ(e) := max{σ(e)/α, 1} for all e ∈ I' ;
```

Algorithm 2: RLS

1 Initialize Y and σ;
2 Determine $sign(Y)$;
3 **while** *the termination not satisfied* **do**
4     $Y' := Y$;
5   Choose an edge $e \in E^*$ uniformly at random ;
6     $Y'(e) := \max\{Y'(e) + \sigma(e) \cdot sign(Y), 0\}$;
7   Determine $sign(Y')$ and $f(Y', Y)$;
8   **if** $f(Y', Y) \geq 0$ **then**
9         $Y := Y'$ and $\sigma(e) := \alpha \cdot \sigma(e)$;
10   **else**
11   **if** $sign(Y) > 0$ **then**
12             $\sigma(e) := \max\{\frac{\sigma(e)}{\alpha}, 1\}$;

To deal with the case that the weights on the vertices are exponentially large with respect to the size of the graph, the Step Size Adaption strategy [3] is incorporated into the two algorithms (see steps 10-16 of the (1+1) EA and steps 8-12 of RLS): the increment on the weights of the edges can exponentially increase or decrease. Let $\sigma : E^* \to \mathbb{N}^+$ be the step size function that keeps the step size for each edge in E^*, and let σ be initialized as $\sigma : E^* \to 1$.

For a mutation of RLS, if it is accepted, then the step size of the chosen edge e is multiplied by a factor $\alpha \in \mathbb{N}^+$; otherwise, multiplied by $1/\alpha$ if $sign(Y) > 0$. For a mutation of the (1+1) EA, if it is accepted, then the step size of each edge $e \in I$ is multiplied by α, otherwise, the step size of each edge $e \in I'$ is multiplied by $1/\alpha$ if $sign(Y) > 0$, where I' is a subset of I such that each edge $e \in I'$ has an endpoint that violates the dual-LP constraint with respect to the dual-solution Y', and no other edge in I shares the endpoint with e (see step 15 of the (1+1) EA). The reason why we define the subset I' of I is that we can ensure that the step size of each edge in I' is unfit for Y. For each edge e in $I \setminus I'$, there are two cases: (1) neither its two endpoints violates the dual-LP constraint with respect to Y'; (2) there is another edge $e' \in I \setminus \{e\}$ that has a common endpoint with e such that the common endpoint of e and e' violates the dual-LP constraint with respect to Y'. For case (1), we should not decrease its step size. For case (2), the step size of e may be fit for Y if it is considered independently. If we adopt a "radical" strategy decreasing the step sizes of all the edges in I if the mutation is rejected, then the algorithms take much time on increasing the step sizes of the edges (in some extreme case, the step size cannot be exponentially increased, resulting that the expected runtime to get an MFDS is exponential [18]). Thus, we adopt a "conservative" strategy: only decrease the step sizes of the edges in I'.

Note that for any mutation of the (1+1) EA or RLS that is rejected, the step sizes of the edges selected by the mutation are not decreased if $sign(Y) < 0$. Because the rejection of the mutation is caused by the selection of the edges, not the violation of the dual-LP constraint.

The selection mechanism implies the following lemma.

Lemma 3.1. *For any dual-solution Y' obtained by a mutation of the* (1+1) EA *or* RLS *on a given dual-solution Y, $sign(Y') \geq sign(Y)$ if Y' is accepted.*

Proof. Since $sign(Y') \geq -1$, $sign(Y') \geq sign(Y)$ obviously holds if $sign(Y) = -1$. For the case that $sign(Y) = 1$, we assume that $sign(Y') = -1$. By the definition of $f(Y', Y)$,

$$f(Y', Y) = sign(Y') - sign(Y) = -2 < 0 ,$$

and neither the (1+1) EA nor RLS would accept Y'. Thus the assumption $sign(Y') = -1$ is incorrect, and $sign(Y') = 1 = sign(Y)$. □

The following lemmata show that once a feasible solution has been obtained, all later accepted solutions will be feasible. Lemma 3.2 can be directly derived from Lemma 3.1.

Lemma 3.2. *If* (1+1) EA *or* RLS *starts with a feasible dual-solution Y of $G^* = (V^*, E^*, W^*)$, then during the process from Y to the output* MFDS*, all dual-solutions accepted by the algorithm are feasible.*

Lemma 3.3. *If* (1+1) EA *or* RLS *starts with an infeasible dual-solution Y of $G^* = (V^*, E^*, W^*)$, where $Y(e) \leq W_{max}$ for each $e \in E^*$, then during the process from Y to the output* MFDS*, there exists an accepted feasible dual-solution Y_t such that all dual-solutions accepted after Y_t are feasible.*

Proof. Let Y' be the dual-solution obtained by a mutation on Y that selects an edge $\overline{e} \in E^* \setminus E_{G^*}(Y)$. Since Y is an infeasible dual-solution of G^*, $Y'(e) \leq Y(e)$ for each $e \in E^* \setminus E_{G^*}(Y)$. Thus,

$$\sum_{e \in E^* \setminus E_{G^*}(Y)} (Y(e) - Y'(e)) \geq Y(\overline{e}) - Y'(\overline{e}) \geq 1 \text{ and}$$

$$2|E^*| \cdot W_{max} \cdot (sign(Y) - 1) \cdot \sum_{e \in E^* \setminus E_{G^*}(Y)} (Y(e) - Y'(e)) \leq -4|E^*| \cdot W_{max} ,$$

where $|E^*|$ and W_{max} are assumed not less than 1.

Combining the obvious fact that $0 \leq sign(Y') - sign(Y) \leq 2$, we have that if the mutation decreases the weight on some edge in $E^* \setminus E_{G^*}(Y)$, then $f(Y', Y)$ is always less than 0, and Y' cannot be accepted by the algorithm. That is, to get a feasible dual-solution of G^*, only the weights on the edges in $E_{G^*}(Y)$ can be decreased.

It is easy to see that if the weights on the edges in $E_{G^*}(Y)$ are decreased to 0, then a feasible dual-solution of G^* can be found. Thus if the algorithm starts with Y, and keeps decreasing the weights on the edges in $E_{G^*}(Y)$, then it can find a feasible dual-solution Y_t, which can be accepted. By Lemma 3.2, all dual-solutions accepted after Y_t are feasible. □

4 RUNTIME ANALYSIS

Given an edge e in the weighted graph $G^* = (V^*, E^*, W^*)$, a mutation of the (1+1) EA or RLS is a *valid* mutation on e if it results an increase or decrease on the weight of e, or on the step size $\sigma(e)$. In this section, we first study the behaviors of the (1+1) EA and RLS on a specific edge $\mathfrak{e} = [v_1, v_2]$ in $G^* = (V^*, E^*, W^*)$. Then using the results obtained for the edge $\mathfrak{e}$, we study the runtimes of the (1+1) EA and RLS for the four versions of DWVC seperately.

Lemma 4.1. *Starting with a feasible dual-solution Y of G^*,* (1+1) EA *(or* RLS*) takes expected runtime $O\left(\alpha m \log_\alpha (Y^*(\mathfrak{e}) - Y(\mathfrak{e}))\right)$ to get a feasible dual-solution Y^* of G^* with respect to which $\mathfrak{e}$ is tight.*

Proof. We first consider the (1+1) EA. Since Y is a feasible dual-solution of G^*, by Lemma 3.2, the sign function $sign()$ keeps to be equal to 1 during the process from Y to Y^*, indicating that the weight on $\mathfrak{e}$ is increased from $Y(\mathfrak{e})$ to $Y^*(\mathfrak{e})$.

Let Y' be the offspring obtained by a mutation M on Y, and let I be the set containing all the edges selected by M. In the following discussion, we first analyze how the mutation M influences the step size $\sigma(\mathfrak{e})$ of $\mathfrak{e}$. Apparently if $\mathfrak{e} \notin I$, then M cannot influence $\sigma(\mathfrak{e})$. Thus we assume that $\mathfrak{e} \in I$.

Case (1). $\sigma(\mathfrak{e}) \leq Y^*(\mathfrak{e}) - Y(\mathfrak{e})$. If M is accepted by the (1+1) EA, then $\sigma(\mathfrak{e})$ is multiplied by the factor α. If M is rejected by the (1+1) EA, then the analysis is divided into the following three subcases.

Case (1.1). An endpoint of $\mathfrak{e}$ violates the dual-LP constraint with respect to Y'. Since $\sigma(\mathfrak{e}) \leq Y^*(\mathfrak{e}) - Y(\mathfrak{e})$, there exists an edge $\mathfrak{e}' \in E_{G^*}(\mathfrak{e}) \cap I$ such that the common endpoint of $\mathfrak{e}$ and $\mathfrak{e}'$ violates the dual-LP constraint with respect to Y'. According to the definition of the edge-set I' (see step 15 of the (1+1) EA), we have that $\mathfrak{e} \notin I'$, and M does not influence $\sigma(\mathfrak{e})$.

Case (1.2). No endpoint of $\mathfrak{e}$ violates the dual-LP constraint with respect to Y'. According to the definition of the edge-set I', we also have that $\mathfrak{e} \notin I'$, and M does not influence $\sigma(\mathfrak{e})$.

By the above analysis, any mutation of the (1+1) EA cannot decrease $\sigma(\mathfrak{e})$ to $\sigma(\mathfrak{e})/\alpha$ for Case (1).

The mutation that only selects the edge $\mathfrak{e}$ can be generated by the (1+1) EA with probability $\Omega(1/m)$, which can be accepted under Case (1). Thus for Case (1), the (1+1) EA takes expected runtime $O(m)$ to increase the weight on edge $\mathfrak{e}$ from $Y(\mathfrak{e})$ to $Y(\mathfrak{e}) + \sigma(\mathfrak{e})$, and increase the step size $\sigma(\mathfrak{e})$ of $\mathfrak{e}$ to $\alpha \cdot \sigma(\mathfrak{e})$.

Case (2). $\sigma(\mathfrak{e}) > Y^*(\mathfrak{e}) - Y(\mathfrak{e})$. Since $\mathfrak{e} \in I$, then the mutation M would be rejected by the (1+1) EA.

Case (2.1). There is an edge $\mathfrak{e}' \in I$ sharing the endpoint of $\mathfrak{e}$ that violates the dual-LP constraint with respect to Y'. Because of the existence of $\mathfrak{e}'$, $\mathfrak{e} \notin I'$ and M does not influence $\sigma(\mathfrak{e})$.

Case (2.2). There is no edge in I that shares an endpoint with $\mathfrak{e}$ such that their common endpoint violates the dual-LP constraint with respect to Y'. Apparently, for this subcase, $\mathfrak{e} \in I'$, and $\sigma(\mathfrak{e})$ is decreased to $\sigma(\mathfrak{e})/\alpha$.

The mutation that only selects the edge $\mathfrak{e}$ belongs to Case (2.2), which can be generated by the (1+1) EA with probability $\Omega(1/m)$. Thus for Case (2), the (1+1) EA takes expected runtime $O(m)$ to decrease step size $\sigma(\mathfrak{e})$ of $\mathfrak{e}$ to $\sigma(\mathfrak{e})/\alpha$.

Now we are ready to analyze the expected runtime that the (1+1) EA takes to increase the weight on $\mathfrak{e}$ from $Y(\mathfrak{e})$ to $Y^*(\mathfrak{e})$, using the above results. The process is divided into two phases: (I). the $\sigma(\mathfrak{e})$-increasing phase; (II). the $\sigma(\mathfrak{e})$-decreasing phase.

During the $\sigma(\mathfrak{e})$-increasing phase, $\sigma(\mathfrak{e})$ can only increase. The $\sigma(\mathfrak{e})$-decreasing phase follows the $\sigma(\mathfrak{e})$-increasing phase, during which $\sigma(\mathfrak{e})$ may increase or decrease, but the general trend is decreasing. Assume that $\sigma(\mathfrak{e})$ is initialized as α^p, where $p \geq 0$. If $\alpha^p > Y^*(\mathfrak{e}) - Y(\mathfrak{e})$, then we are already at the $\sigma(\mathfrak{e})$-decreasing phase. In the following discussion, we assume that $\alpha^p \leq Y^*(\mathfrak{e}) - Y(\mathfrak{e})$.

(I). The $\sigma(\mathfrak{e})$-increasing phase. Let q be the integer such that

$$\sum_{i=p}^{q} \alpha^i \leq Y^*(\mathfrak{e}) - Y(\mathfrak{e}) \text{ , } and \sum_{i=p}^{q+1} \alpha^i > Y^*(\mathfrak{e}) - Y(\mathfrak{e}) \text{ .}$$

It is easy to see that $\sigma(\mathfrak{e})$ can be increased from α^p to α^{q+1} during this phase. Thus the number of valid mutations on $\mathfrak{e}$ during this phase is $q - p + 1$, where

$$q - p + 1 = \left\lfloor \log_\alpha \left(\frac{(Y^*(\mathfrak{e}) - Y(\mathfrak{e}))(\alpha - 1)}{\alpha^p} + 1 \right) \right\rfloor \text{ .}$$

Combining the analysis for Case (1), the $\sigma(\mathfrak{e})$-increasing phase takes expected runtime $O\left(m \log_\alpha (Y^*(\mathfrak{e}) - Y(\mathfrak{e}))\right)$ (because p may be 0).

(II). The $\sigma(\mathfrak{e})$-decreasing phase. During this phase, the weight on $\mathfrak{e}$ is increased from $Y(\mathfrak{e}) + \sum_{i=p}^{q} \alpha^i$ to $Y^*(\mathfrak{e})$, and $\sigma(\mathfrak{e})$ is decreased from α^{q+1} to 1. Similar to the analysis for the $\sigma(\mathfrak{e})$-increasing phase, we analyze the number T of valid mutations on $\mathfrak{e}$ during the $\sigma(\mathfrak{e})$-decreasing phase.

To simplify the analysis, we consider the number t_i of valid mutations on $\mathfrak{e}$ with $\sigma(\mathfrak{e}) = \alpha^i$ $(0 \leq i \leq q+1)$ among the T valid mutations on $\mathfrak{e}$ (since $\sigma(\mathfrak{e})$ can increase or decrease during the $\sigma(\mathfrak{e})$-decreasing phase, there may be more than one valid mutation on $\mathfrak{e}$ with $\sigma(\mathfrak{e}) = \alpha^i$). Obviously $T = \sum_{i=0}^{q+1} t_i$.

We first consider t_{q+1}. Since the valid mutation on $\mathfrak{e}$ with $\sigma(\mathfrak{e}) = \alpha^{q+1}$ cannot be accepted, $\sigma(\mathfrak{e})$ will be decreased to α^q. Observe that if a valid mutation on $\mathfrak{e}$ with $\sigma(\mathfrak{e}) = \alpha^q$ is accepted, then $\sigma(\mathfrak{e})$ would be increased to α^{q+1}. Thus $t_{q+1} \leq 1 + (\alpha - 1) = \alpha$, because there are at most $\alpha - 1$ valid mutations on $\mathfrak{e}$ with $\sigma(\mathfrak{e}) = \alpha^q$ among the T valid mutations on $\mathfrak{e}$ that can be accepted by the algorithm.

For any $1 \leq i \leq q$, since there are at most $\alpha - 1$ valid mutations on $\mathfrak{e}$ with $\sigma(\mathfrak{e}) = \alpha^i$ among the T valid mutations on $\mathfrak{e}$ that can be accepted, and at most α valid mutations on $\mathfrak{e}$ with $\sigma(\mathfrak{e}) = \alpha^i$ that can be rejected (use the analysis given above for $\sigma(\mathfrak{e}) = \alpha^{q+1}$), we can get that $t_i \leq 2\alpha - 1$.

If $\sigma(\mathfrak{e}) = 1$, then the weight on $\mathfrak{e}$ is between $Y^*(\mathfrak{e}) - \alpha + 1$ and $Y^*(\mathfrak{e})$. If the weight on $\mathfrak{e}$ equals $Y^*(\mathfrak{e})$, then the $\sigma(\mathfrak{e})$-decreasing phase is over, and $t_0 = 0$. If the weight on $\mathfrak{e}$ is between $Y^*(\mathfrak{e}) - \alpha + 1$ and $Y^*(\mathfrak{e}) - 1$, then $t_0 \leq \alpha - 1$.

The above analysis gives

$$T = \sum_{i=0}^{q+1} t_i \leq (2\alpha - 1)(q + 1) \text{ .}$$

Combining the analysis for Case (1-2), the $\sigma(\mathfrak{e})$-decreasing phase takes expected runtime $O\left(\alpha m \log_\alpha (Y^*(\mathfrak{e}) - Y(\mathfrak{e}))\right)$.

Therefore, there are at most $2\alpha(q+1)$ valid mutations on $\mathfrak{e}$ during the process from Y to Y^*, for which the (1+1) EA takes expected runtime $O\left(\alpha m \log_\alpha (Y^*(\mathfrak{e}) - Y(\mathfrak{e}))\right)$.

Since we only consider the mutation selecting exactly one edge in the analysis for the (1+1) EA, it is easy to get that the conclusions for the (1+1) EA also apply to RLS. □

Now we turn to analyze the expected runtimes that the two algorithms take to make the edge $\mathfrak{e}$ satisfy the dual-LP constraint, if they start with an infeasible dual-solution with respect to which $\mathfrak{e}$ does not satisfy the dual-LP constraint.

Lemma 4.2. *Starting with an infeasible dual-solution Y of G^* with respect to which edge $\mathfrak{e}$ does not satisfy the dual-LP constraint, the* (1+1) EA *(or* RLS*) takes expected runtime $O\left(m \log_\alpha (Y(\mathfrak{e}) - Y^*(\mathfrak{e}))\right)$ to get the first dual-solution Y^* of G^* with respect to which $\mathfrak{e}$ satisfies the dual-LP constraint.*

Proof. We first consider the (1+1) EA. Since Y is infeasible, the (1+1) EA decreases the weight on $\mathfrak{e}$ to get Y^*.

Assume that $\sigma(\mathfrak{e})$ is initialized as α^p ($p \geq 0$). Note that the value of $\sigma(\mathfrak{e})$ will not be decreased during the process from Y to Y^*, hence there exists an integer q such that $\sum_{i=p}^{q} \alpha^i = Y(\mathfrak{e}) - Y^*(\mathfrak{e})$. Thus during the process, $\sigma(e)$ is increased from α^p to α^{q+1}, and there are $q - p + 1$ valid mutations on $\mathfrak{e}$, where

$$q - p + 1 = \log_\alpha \left(\frac{(Y(\mathfrak{e}) - Y^*(\mathfrak{e}))(\alpha - 1)}{\alpha^p} + 1 \right) .$$

The mutation that only selects the edge $\mathfrak{e}$ can be generated by the (1+1) EA with probability $\Omega(1/m)$, which is a obviously valid mutation on $\mathfrak{e}$. Hence the (1+1) EA takes expected runtime $O(m(q+1)) = O\left(m \log_\alpha (Y(\mathfrak{e}) - Y^*(\mathfrak{e}))\right)$ to get Y^* (because p may be 0). The above conclusions for the (1+1) EA also apply to RLS. □

4.1 Analysis for DWVC-E$^+$

In this subsection, we study the performances of the two algorithms for DWVC-E$^+$.

Theorem 4.3. *The expected runtime of the* (1+1) EA *(or* RLS*) for* DWVC-E$^+$ *is* $O(\alpha m \log_\alpha^2 (D \cdot W_{max}))$.

Proof. Given a DWVC-E$^+$ instance $\{G = (V, E, W), Y_{\text{orig}}, E^+\}$. We first consider the expected runtime of the (1+1) EA to obtain an MFDS of $G^* = (V, E \cup E^+, W)$, starting with the MFDS Y_{orig} of $G = (V, E, W)$. For each edge $e \in E^+$, $Y_{\text{orig}}(e)$ and $\sigma(e)$ are initialized as 0 and 1 respectively. Apparently Y_{orig} is a feasible dual-solution of G^*. Thus any mutation on Y_{orig} would be rejected if Y is an MFDS of G^*, and the algorithm would keep Y forever.

In the following, we assume that Y_{orig} is not an MFDS of G^*. Observe that the weights on the edges in E cannot be increased. Thus by Lemma 3.2, we have that $Y^*(e) = Y_{\text{orig}}(e)$ for each edge $e \in E$, and $Y^*(e) \geq Y_{\text{orig}}(e)$ for each edge $e \in E^+$, where Y^* is an arbitrary MFDS of G^* obtained by the (1+1) EA starting with Y_{orig}.

Now we analyze the expected runtime that the (1+1) EA takes to get Y^*. Denote $Y^*(e) - Y_{\text{orig}}(e)$ by $\Delta(e)$ for each $e \in E^+$, and denote the number of valid mutations on e required to increase the weight on e from $Y_{\text{orig}}(e)$ to $Y^*(e)$ by $\beta(e)$. Let $E_{\beta\neq 0} = \{e \in E^+ | \beta(e) \neq 0\}$. For the $\sum_{e \in E_{\beta\neq 0}} \beta(e)$ valid mutations on the edges in $E_{\beta\neq 0}$, we have that $\sum_{e \in E_{\beta\neq 0}} Y_{\text{orig}}(e)$ is increased by $\sum_{e \in E_{\beta\neq 0}} \Delta(e)$. Thus the expected increment for each valid mutation on the edges in $E_{\beta\neq 0}$ is

$$\frac{\sum_{e \in E_{\beta\neq 0}} \Delta(e)}{\sum_{e \in E_{\beta\neq 0}} \beta(e)} .$$

The mutation selecting exactly one edge in $E_{\beta\neq 0}$ is an obviously valid mutation on the edges in $E_{\beta\neq 0}$, which can be generated by an iteration of the while-loop with probability $\Omega(\frac{|E_{\beta\neq 0}|}{e \cdot m})$. Thus the expected increment of $\sum_{e \in E_{\beta\neq 0}} Y_{\text{orig}}(e)$ made by each iteration of the while-loop is at least

$$\frac{|E_{\beta\neq 0}|}{e \cdot m} \cdot \frac{\sum_{e \in E_{\beta\neq 0}} \Delta(e)}{\sum_{e \in E_{\beta\neq 0}} \beta(e)} .$$

Now we analyze the lower bound of $\frac{|E_{\beta\neq 0}| \cdot \sum_{e \in E_{\beta\neq 0}} \Delta(e)}{e \cdot m \cdot \sum_{e \in E_{\beta\neq 0}} \beta(e)}$. By Lemma 4.1, for each edge $e \in E_{\beta\neq 0}$, we have that

$$\begin{aligned} \beta(e) &\leq 2\alpha \left\lfloor \log_\alpha ((\alpha - 1) \cdot \Delta(e) + 1) \right\rfloor \\ &\leq 2\alpha \lfloor \log_\alpha (\alpha \cdot \Delta(e)) \rfloor = 2\alpha \left(\lfloor \log_\alpha \Delta(e) \rfloor + 1 \right) . \end{aligned}$$

Thus,

$$\begin{aligned} \sum_{e \in E_{\beta\neq 0}} \beta(e) &\leq 2\alpha \left(|E_{\beta\neq 0}| + \sum_{e \in E_{\beta\neq 0}} \log_\alpha \Delta(e) \right) \\ &\leq 2\alpha \left(|E_{\beta\neq 0}| + |E_{\beta\neq 0}| \cdot \log_\alpha \sum_{e \in E_{\beta\neq 0}} \Delta(e) \right) . \end{aligned}$$

The maximum value and minimum value that $\sum_{e \in E_{\beta\neq 0}} \Delta(e)$ can take are $D \cdot W_{max}$ and 1 respectively, hence

$$\frac{|E_{\beta\neq 0}| \cdot \sum_{e \in E_{\beta\neq 0}} \Delta(e)}{e \cdot m \cdot \sum_{e \in E_{\beta\neq 0}} \beta(e)} \geq \frac{\sum_{e \in E_{\beta\neq 0}} \Delta(e)}{2e \cdot \alpha m \left(1 + \log_\alpha (D \cdot W_{max})\right)} .$$

The Multiplicative Drift Analysis [4] gives that the (1+1) EA takes expected runtime $O(\alpha m \log_\alpha^2 (D \cdot W_{max}))$ to find an MFDS Y^* of G^*.

Since we only consider the mutations selecting exactly one edge in the above analysis for the (1+1) EA, the above conclusions for the (1+1) EA also apply to RLS. □

4.2 Analysis for DWVC-E$^-$

Given an instance $\{G = (V, E, W), Y_{\text{orig}}, E^-\}$ of DWVC-E$^-$. Obviously, the endpoints of the edges in E^- may not be tight with respect to the MFDS Y_{orig} of G once the edges in $E^- \subseteq E$ are removed (note that the domain of definition for Y_{orig} and the weight function W would be modified as $E \setminus E^-$ after the edges in E' are removed). Thus the weights on the edges in $E_{G^*}(E^-)$ have the room to be increased.

Theorem 4.4. *The expected runtime of the* (1+1) EA *(or* RLS*) for* DWVC-E$^-$ *is* $O(\alpha m \log_\alpha^2 (D \cdot W_{max}))$.

Proof. We consider the expected runtime of the (1+1) EA (or RLS) to obtain an MFDS of $G^* = (V, E \setminus E^-, W)$, starting with the MFDS Y_{orig} of $G = (V, E, W)$. Apparently Y_{orig} is a feasible dual-solution of G^*. If Y_{orig} is an MFDS of G^*, then any mutation of the (1+1) EA (or RLS) on Y_{orig} would be rejected, and the algorithm keeps Y_{orig} forever. Thus in the following discussion, we assume that Y_{orig} is not an MFDS of G^*.

Let Y^* be an arbitrary MFDS of G^* obtained by the (1+1) EA (or RLS) starting with Y_{orig}. Observe that the weights on the edges in $E \setminus (E^- \cup E_{G^*}(E^-))$ cannot be increased. Thus we have that $Y^*(e) = Y_{\text{orig}}(e)$ for each edge $e \in E \setminus (E^- \cup E_{G^*}(E^-))$, and $Y^*(e) \geq Y_{\text{orig}}(e)$ for each edge $e \in E_{G^*}(E^-)$.

Since all the edges in $E_{G^*}(E^-)$ are incident to the endpoints of the edges in E^-, where the number of the endpoints of the edges in E^- can be bounded by $2D$, $\sum_{e \in E_{G^*}(E^-)} Y^*(e)$ can be upper bounded by $2D \cdot W_{max}$, and

$$\sum_{e \in E_{G^*}(E^-)} (Y^*(e) - Y_{\text{orig}}(e)) \leq \sum_{e \in E_{G^*}(E^-)} Y^*(e) \leq 2D \cdot W_{max} .$$

Using the reasoning similar to that for Theorem 4.3, the (1+1) EA (or RLS) can obtain Y^* in expected runtime $O(\alpha m \log^2_\alpha (D \cdot W_{max}))$. □

4.3 Analysis for DWVC-W$^+$

Given an instance $\{G = (V, E, W), Y_{\text{orig}}, V^+, W^+\}$ of DWVC-W$^+$. The following lemma shows that the sum of the increments of the weights on the edges can be bounded.

Lemma 4.5. *For any* MFDS Y^* *obtained by the* (1+1) EA *(or* RLS*) for instance* $\{G = (V, E, W), Y_{\text{orig}}, V^+, W^+\}$,

$$\sum_{e \in E} \left(Y^*(e) - Y_{\text{orig}}(e)\right) \leq \sum_{v \in V^+} \left(W^+(v) - W(v)\right).$$

Proof. Since Y_{orig} is an obviously feasible dual-solution of $G^* = (V, E, W^+)$, $Y^*(e) \geq Y_{\text{orig}}(e)$ for each edge $e \in E$. Let E_{W^+} be the set containing all edges $e \in E$ where $Y^*(e) > Y_{\text{orig}}(e)$. Observe that the weights on the edges in $E \setminus E_{G^*}(V^+)$ cannot be increased. Thus

$$\sum_{e \in E} \left(Y^*(e) - Y_{\text{orig}}(e)\right) = \sum_{e \in E_{W^+}} \left(Y^*(e) - Y_{\text{orig}}(e)\right). \quad (1)$$

For each $e \in E_{W^+}$, let $\tau(e)$ be the endpoint of e that is tight with respect to Y_{orig} (if both endpoints are tight, then arbitrarily choose one). Obviously $\tau(e) \in V^+$. For any $v \in V^+$, we have

$$\sum_{e \in E_{W^+} | \tau(e) = v} \left(Y^*(e) - Y_{\text{orig}}(e)\right) \leq W^+(v) - W(v),$$

and

$$\sum_{e \in E_{W^+}} \left(Y^*(e) - Y_{\text{orig}}(e)\right) \leq \sum_{v \in V^+} \left(W^+(v) - W(v)\right). \quad (2)$$

Combining (1) and (2) gives the claimed inequality. □

Using the reasoning similar to that for Theorem 4.3 and the bound proved by Lemma 4.5, we have the following theorem.

Theorem 4.6. *The expected runtime of the* (1+1) EA *(or* RLS*) for* DWVC-W$^+$ *is* $O(\alpha m \log^2_\alpha (D \cdot W_{max}))$.

4.4 Analysis for DWVC-W$^-$

Given an instance $\{G = (V, E, W), Y_{\text{orig}}, V^-, W^-\}$ of DWVC-W$^-$. Since Y_{orig} may be an infeasible dual-solution of $G^* = (V, E, W^-)$, it is necessary to consider the process from the infeasible dual-solution Y_{orig} of G^* to a feasible dual-solution of G^* in the following discussion.

Theorem 4.7. *The expected runtime of the* (1+1) EA *(or* RLS*) for* DWVC-W$^-$ *is* $O(\alpha m \log^2_\alpha (D \cdot W_{max}))$.

Proof. We first consider the expected runtime of the (1+1) EA to obtain an MFDS of $G^* = (V, E, W^-)$, starting with the MFDS Y_{orig} of $G = (V, E, W)$. Obviously if Y_{orig} is a feasible dual-solution of G^*, then Y_{orig} is also an MFDS of G^*, and any mutation on Y_{orig} would be rejected. In the following discussion, we assume that Y_{orig} is an infeasible dual-solution of G^*.

Let Y_t be the first feasible dual-solution accepted by the (1+1) EA. By Lemma 3.3, to get a feasible dual-solution of G^*, only the weights on the edges in $E_{G^*}(Y_{\text{orig}})$ can be decreased by the (1+1) EA, where $E_{G^*}(Y_{\text{orig}}) \subseteq E_{G^*}(V^-)$. Thus $Y_t(e) \leq Y_{\text{orig}}(e)$ for each edge $e \in E_{G^*}(Y)$, and $Y_t(e) = Y_{\text{orig}}(e)$ for each edge $e \in E \setminus E_{G^*}(Y_{\text{orig}})$.

Denote $Y_{\text{orig}}(e) - Y_t(e)$ by $\Delta(e)$ for each $e \in E_{G^*}(Y_{\text{orig}})$, and denote the number of valid mutations on e required to decrease the weight on e from $Y_{\text{orig}}(e)$ to $Y_t(e)$ by $\beta(e)$. Let $E_{\beta \neq 0} = \{e \in E_{G^*}(Y_{\text{orig}}) \mid \beta(e) \neq 0\}$.

Since $\sum_{e \in E_{\beta \neq 0}} Y_{\text{orig}}(e)$ is decreased by $\sum_{e \in E_{\beta \neq 0}} \Delta(e)$ for the $\sum_{e \in E_{\beta \neq 0}} \beta(e)$ valid mutations on the edges in $E_{\beta \neq 0}$, and each iteration of the while-loop generates a mutation selecting exactly one edge in $E_{\beta \neq 0}$ with probability $\Omega(\frac{|E_{\beta \neq 0}|}{e \cdot m})$, the expected decrement of $\sum_{e \in E_{\beta \neq 0}} Y_{\text{orig}}(e)$ made by each iteration of the while-loop is at least

$$\frac{|E_{\beta \neq 0}|}{e \cdot m} \cdot \frac{\sum_{e \in E_{\beta \neq 0}} \Delta(e)}{\sum_{e \in E_{\beta \neq 0}} \beta(e)}.$$

Now we analyze the lower bound of $\frac{|E_{\beta \neq 0}| \cdot \sum_{e \in E_{\beta \neq 0}} \Delta(e)}{e \cdot m \cdot \sum_{e \in E_{\beta \neq 0}} \beta(e)}$. By Lemma 4.2, for each edge $e \in E_{\beta \neq 0}$, we have that

$$\beta(e) \leq \log_\alpha ((\alpha - 1)\Delta(e) + 1) \leq \log_\alpha \Delta(e) + 1.$$

$$\text{Thus,} \sum_{e \in E_{\beta \neq 0}} \beta(e) \leq \sum_{e \in E_{\beta \neq 0}} \log_\alpha \Delta(e) + |E_{\beta \neq 0}| \leq \sum_{e \in E_{\beta \neq 0}} \log_\alpha \sum_{e \in E_{\beta \neq 0}} \Delta(e) + |E_{\beta \neq 0}| = |E_{\beta \neq 0}| \cdot \log_\alpha \sum_{e \in E_{\beta \neq 0}} \Delta(e) + |E_{\beta \neq 0}|.$$

Therefore, we have

$$\frac{|E_{\beta \neq 0}| \cdot \sum_{e \in E_{\beta \neq 0}} \Delta(e)}{e \cdot m \cdot \sum_{e \in E_{\beta \neq 0}} \beta(e)} \geq \frac{\sum_{e \in E_{\beta \neq 0}} \Delta(e)}{e \cdot m \cdot \left(\log_\alpha \left(\sum_{e \in E_{\beta \neq 0}} \Delta(e)\right) + 1\right)}.$$

Since $\sum_{e \in E_{G^*}(V^-)} Y_{\text{orig}}(e) \leq D \cdot W_{max}$ and $E_{G^*}(Y_{\text{orig}}) \subseteq E_{G^*}(V^-)$, we have $\sum_{e \in E_{G^*}(Y_{\text{orig}})} Y_{\text{orig}}(e) \leq D \cdot W_{max}$, and the maximum value and minimum value that $\sum_{e \in E_{\beta \neq 0}} \Delta(e)$ can take are $D \cdot W_{max}$ and 1 respectively. Therefore,

$$\frac{|E_{\beta \neq 0}| \cdot \sum_{e \in E_{\beta \neq 0}} \Delta(e)}{e \cdot m \cdot \sum_{e \in E_{\beta \neq 0}} \beta(e)} \geq \frac{\sum_{e \in E_{\beta \neq 0}} \Delta(e)}{e \cdot m \left(\log_\alpha (D \cdot W_{max}) + 1\right)}.$$

The Multiplicative Drift Analysis [4] gives that the (1+1) EA takes expected runtime $O(m \log^2_\alpha (D \cdot W_{max}))$ to get Y_t.

Obviously Y_t may not be an MFDS of G^*. Thus we also need to consider the process of the (1+1) EA to get an MFDS of G^*, starting with Y_t. To simplify the analysis, we aim to transform the process to an execution of the (1+1) EA for an instance of DWVC-W$^+$.

In the following, we first construct a weighted graph $G_t = (V, E, W_t)$ such that Y_t is an MFDS of G_t. Let $V_{\beta \neq 0}$ contain all endpoints of the edges in $E_{\beta \neq 0}$. For each vertex $v \in V \setminus V_{\beta \neq 0}$, let $W_t(v) = W(v)$, and for each vertex $v \in V_{\beta \neq 0}$, let

$$W_t(v) = W(v) - \sum_{e \in E_{\beta \neq 0} | e \cap v \neq \emptyset} \Delta(e).$$

Since Y_{orig} is an MFDS of G, Y_t is an obvious MFDS of G_t .

Since Y_t is a feasible dual-solution of G^* but an MFDS of G_t, $W_t(v) \leq W^-(v)$ for each vertex $v \in V$. Let V' be the subset of V such that for each vertex $v \in V'$, $W_t(v) < W^-(v)$. Thus the instance of DWVC-W$^+$ is $\{G_t = (V, E, W_t), Y_t, V', W^-\}$. Similar to

Lemma 4.5, we give the upper bound that the sum of the increments of the weights on the edges with respect to W^-. We have

$$\begin{aligned}\sum_{v\in V'}\left(W^-(v) - W_t(v)\right) &\leq \sum_{v\in V'}(W(v) - W_t(v))\\ &\leq \sum_{v\in V'}\left(\sum_{e\in E_{\beta\neq 0}|e\cap v\neq\emptyset}\Delta(e)\right)\\ &\leq 2\sum_{e\in E_{\beta\neq 0}}\Delta(e) \leq 2D\cdot W_{max}\,.\end{aligned}$$

By the reasoning similar to that for Theorem 4.3, the (1+1) EA takes expected runtime $O(\alpha m \log^2_\alpha(D\cdot W_{max}))$ to get an MFDS of G^* starting with Y_t.

Summarizing the above discussion, the (1+1) EA gets an MFDS of G^* in expected runtime $O(\alpha m \log^2_\alpha(D\cdot W_{max}))$. The above time complexity also applies to RLS. □

5 CONCLUSION

In this paper, we contribute to the theoretical understanding of evolutionary computing for the Dynamic Weighted Vertex Cover problem, generalizing the results obtained by Pourhassan et al. [19] for the Dynamic Vertex Cover problem. Four graph-editing operations were studied for the dynamic changes of the given weighted graph, which lead to four versions of the Dynamic Weighted Vertex Cover problem. The performances of algorithms (1+1) EA and RLS with step size adaption strategy for the four versions were analyzed seperately, which show that they can maintain the quality of the solution for these studied dynamic changes efficiently.

Pourhassan et al. [18] studied the Weighted Vertex Cover problem using the dual form of the LP formulation for the Fractional Weighted Vertex Cover problem recently. They showed that their (1+1) EA with Step Size Adaptation cannot get a 2-approximate solution in polynomial expected time with high probability. There are two main differences between their (1+1) EA and our (1+1) EA: (1). for the mutation M constructed by their (1+1) EA, there may exist two edges selected by the mutation M whose weights are increased and decreased respectively; for the mutation M constructed by our (1+1) EA, the weights on the edges selected by the mutation M are either all increased or all decreased; (2). for the mutation M that is rejected by their (1+1) EA, the step sizes of all the edges selected by the mutation M are decreased; for the mutation M that is rejected by our (1+1) EA, only the step sizes of the edges satisfying a specific condition can be decreased.

REFERENCES

[1] A. Auger and B. Doerr. *Theory of Randomized Search Heuristics: Foundations and Recent Developments*. World Scientific Publishing Co., Inc., 2011.
[2] 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.
[3] H.-G. Beyer and H.-P. Schwefel. Evolution strategies–a comprehensive introduction. *Natural computing*, 1(1):3–52, 2002.
[4] B. Doerr, D. Johannsen, and C. Winzen. Multiplicative drift analysis. *Algorithmica*, 64(4):673–697, 2012.
[5] D.-Z. Du, K.-I. Ko, and X. Hu. *Design and analysis of approximation algorithms*, volume 62. Springer Science & Business Media, 2011.
[6] 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.
[7] T. Friedrich and F. Neumann. What's hot in evolutionary computation. In *AAAI*, pages 5064–5066, 2017.
[8] 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.
[9] D. S. Hochbaum. Efficient bounds for the stable set, vertex cover and set packing problems. *Discrete Applied Mathematics*, 6(3):243 – 254, 1983.
[10] T. Jansen. *Analyzing Evolutionary Algorithms - The Computer Science Perspective*. Natural Computing Series. Springer, 2013.
[11] T. Jansen, P. S. Oliveto, and C. Zarges. Approximating vertex cover using edge-based representations. In *Proceedings of the twelfth workshop on Foundations of genetic algorithms XII*, pages 87–96. ACM, 2013.
[12] S. Khot. On the power of unique 2-prover 1-round games. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 767–775. ACM, 2002.
[13] S. Khot and O. Regev. Vertex cover might be hard to approximate to within 2-ε. *Journal of Computer and System Sciences*, 74(3):335–349, 2008.
[14] T. Kötzing, A. Lissovoi, and C. Witt. (1+ 1) ea on generalized dynamic onemax. In *Proceedings of the 2015 ACM Conference on Foundations of Genetic Algorithms XIII*, pages 40–51. ACM, 2015.
[15] S. Kratsch and F. Neumann. Fixed-parameter evolutionary algorithms and the vertex cover problem. *Algorithmica*, 65(4):754–771, 2013.
[16] 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.
[17] F. Neumann and C. Witt. On the runtime of randomized local search and simple evolutionary algorithms for dynamic makespan scheduling. In *IJCAI*, pages 3742–3748, 2015.
[18] M. Pourhassan, T. Friedrich, and F. Neumann. On the use of the dual formulation for minimum weighted vertex cover in evolutionary algorithms. In *Proceedings of the 14th ACM/SIGEVO Conference on Foundations of Genetic Algorithms*, pages 37–44. ACM, 2017.
[19] M. Pourhassan, W. Gao, and F. Neumann. Maintaining 2-approximations for the dynamic vertex cover problem using evolutionary algorithms. In *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*, pages 903–910. ACM, 2015.
[20] M. Pourhassan, F. Shi, and F. Neumann. Parameterized analysis of multi-objective evolutionary algorithms and the weighted vertex cover problem. In *Proceedings of the 14th International Conference on Parallel Problem Solving from Nature (PPSN XIV)*, pages 729–739. Springer, 2016.
[21] F. Shi, M. Schirneck, T. Friedrich, T. Kötzing, and F. Neumann. Reoptimization times of evolutionary algorithms on linear functions under dynamic uniform constraints. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1407–1414. ACM, 2017.
[22] V. V. Vazirani. *Approximation algorithms*. Springer Science & Business Media, 2013.