Improved Runtime Analysis of RLS and (1+1) EA for Dynamic Vertex Cover Problem

Mojgan Pourhassan
Optimisation and Logistics
School of Computer Science
The University of Adelaide
Adelaide, Australia
mojgan.pourhassan@adelaide.edu.au

Vahid Roostapour
Optimisation and Logistics
School of Computer Science
The University of Adelaide
Adelaide, Australia
vahid.roostapour@adelaide.edu.au

Frank Neumann
Optimisation and Logistics
School of Computer Science
The University of Adelaide
Adelaide, Australia
frank.neumann@adelaide.edu.au

***Abstract*—In this paper, we perform theoretical analyses of the behaviour of an evolutionary algorithm and a randomised search algorithm on the dynamic vertex cover problem. The dynamic vertex cover problem has already been theoretically investigated for these two algorithms to some extent. We improve some of the existing results, i. e. we find a linear expected re-optimization time for a (1+1) EA to maintain a 2-approximation when edges are dynamically deleted from the graph. Furthermore, we investigate a different setting for the dynamic version of the problem, in which a dynamic change happens at each step with probability P_D. We prove that when $P_D \leq \frac{1}{(2+\epsilon)em}$, where m is the number of edges of the graph, in expected time $O(m \log m)$ RLS and (1+1) EA find a 2-approximate solution from an arbitrary initial solution. Furthermore, we prove that in expected time $O(m)$ after the first dynamic change, they maintain the quality of that solution.**

***Index Terms*—Dynamic Vertex Cover Problem; Local Search; (1+1) EA; Combinatorial Optimisation**

I. Introduction

Using evolutionary algorithms for solving dynamic combinatorial optimization problems have previously been theoretically analysed in a number of articles [1], [4], [6], [9], [11], [12]. Some of the classical problems that have been investigated in the dynamic context are the OneMax problem, the makespan scheduling problem and the vertex cover problem [1], [4], [6], [9]. In a recent work [11], the behaviour of evolutionary algorithms on linear functions under dynamically changing constraints is investigated.

In [9] the vertex cover problem is considered with a simple dynamic setting where the rate of dynamic changes is small enough, so that the studied algorithm can re-optimize the problem after a dynamic change, before the following change happens. The article by Droste [1] on the OneMax problem presents another setting for dynamically changing problems, where a dynamic change happens at each step with probability p'. In that article, the maximum rate of dynamic changes is found such that the expected optimization time of (1+1) EA remains polynomial for the studied problem. In his analyses the goal is to find a solution which has the minimum Hamming distance to an objective bit-string and one bit of the objective bit-string changes at each time step with a probability p'; which results in the dynamic changes of the fitness function over time. The author of the article has proved that the (1+1) EA has a polynomial expected runtime if $p' = O(log(n)/n)$, while for every substantially larger probability the runtime becomes super polynomial. The results of that article hold even if the expected re-optimization time of the problem is larger than the expected time until the next dynamic change happens. Kotzing et al. [4] have reproved some of the results of [1] by means of drift analysis, and have extended the work to search spaces with more than two values for each dimension. Furthermore, they analyse how closely their investigated algorithm can track the dynamically moving target over time.

In this paper, we consider both dynamic settings and analyse two simple randomised algorithms on the vertex cover problem. Different variants of the classical randomised local search (RLS) and (1+1) EA have previously been investigated for the static vertex cover problem in the context of approximations. This includes a node-based representation examined in [2], [5], [7], [10] as well as a different edge-based representation analysed in [3] and a generalization of that for the weighted vertex cover problem analysed in [8].

For the dynamic version of the problem, three variants of those randomised algorithms have been investigated in [9]. The investigated variants include an approach with the classical node-based representation in addition to two approaches with edge-based representation introduced in [3]: one with a standard fitness function, and one with a fitness function that gives a large penalty for adjacent edges. Among these approaches, the third one finds a 2-approximation from scratch in expected time $O(m \log m)$ [3], where m is the number of edges. Having the large penalty for adjacent edges in that approach results in finding a maximal matching, which induces a 2-approximate vertex cover. Considering the dynamic version of the problem where a solution which is a maximal matching is given before the dynamic change, Pourhassan et al. [9] proved that the RLS maintains the quality of the solution in expected time $O(m)$. They also proved that (1+1) EA manages to maintain the 2-approximation in expected time $O(m)$ when the dynamic change is adding an edge, but for edge deletion, the expected

This research has been supported by the Australian Research Council (ARC) grants DP140103400 and DP160102401.

time $O(m \log m)$ was obtained, which is the same as the expected time of finding a 2-approximation from scratch.

Our aim in this paper is to improve the expected time that (1+1) EA with the third approach requires to maintain a 2-approximation when edges are dynamically deleted from the graph. Moreover, we investigate the other setting for a dynamic version of the problem, in which a dynamic change happens with a certain probability, P_D, at each step. We prove that when P_D is small enough, RLS and (1+1) EA with the third approach, find a 2-approximate solution from an arbitrary initial solution in expected polynomial time, and maintain the quality of 2-approximation in expected linear time after a dynamic change happens.

The rest of the paper is structured as follows. The problem definition and the investigated algorithm are given in Section II. Section III includes the analysis for improving the re-optimization time of (1+1) EA with the third approach for dynamic delete of an edge. The other dynamic setting for the promlem is investigated in Section IV and the conclusion is given in Section V.

II. Algorithms and the Dynamic Vertex Cover Problem

In this section we present the definition of vertex cover problem, the dynamic vertex cover problem and the algorithms that we investigate in this paper. For a given graph $G = (V, E)$ with set of vertices $V = \{v_1, \ldots, v_n\}$ and set of edges $E = \{e_1, \ldots, e_m\}$, the vertex cover problem is to find a subset of nodes $V_C \subset V$ with minimum cardinality, that covers all edges in E, i.e. $\forall e \in E, e \cap V_C \neq \emptyset$.

In the dynamic version of the problem, an arbitrary edge can be added to or deleted from the graph. We investigate two different settings for applying the dynamism on the problem. In the first setting, which has previously been analysed in [9], the changes on the instance of the problem take place every $\tau = poly(n)$ iterations where $poly(n)$ is a polynomial function in n. We improve some results that were obtained in [9] for this setting. In the second setting that we investigate, a dynamic change happens at each step with a probability P_D; therefore, on expectation, a dynamic change happens on the graph each $\frac{1}{P_D}$ steps.

For solving the vertex cover problem by means of evolutionary algorithms, two kinds of representation have been suggested: the node-based representation and the edge-based representation. While the node-based representation is the natural one for this problem, and is used in most of the relevant works [2], [5], [7], the edge-based representation, introduced by Jansen et al. [3], has been suggested to speed up the approximation process. In their work [3], they have proved that an evolutionary algorithm using the edge-based representation and a specific fitness function, can find a 2-approximate solution in expected time $O(m \log m)$ where m is the number of edges in the graph.

In this representation, each solution is a bit string $s \in \{0,1\}^m$, describing a selection of edges $E(s) = \{e_i \in E | s_i = 1\}$. Then the cover set of s, denoted by $V_C(s)$, is the set of nodes on both side of each edge in $E(s)$. It should be noticed that the size of the solution may change according to the dynamic changes of the graph. In our analysis m is the maximum number of edges in the graph.

Algorithm 1 Edge-Based RLS (RLS$_e$) [9]

1: The initial solution, s, is given: a bit-string of size m which used to be a 2-approximate solution before changing the graph.
2: Set $s' = s$
3: Select $i \in \{1, \ldots, m\}$ uniformly at random and flip ith bit of s'
4: If $f(s') \leq f(s)$ then $s := s'$
5: If stopping criteria not met continue at line 2

The specific fitness function that Jensen et al. [3] have suggested for this representation is:

$$f(s) = |V_C(s)| + (|V| + 1) \cdot |\{e \in E | e \cap V_C(s) = \emptyset\}| + (|V| + 1) \cdot (m + 1) \cdot |\{(e, e') \in E(s) \times E(s) | e \neq e', e \cap e' \neq \emptyset\}|. \quad (1)$$

The goal of the studied evolutionary algorithm is to minimize $f(s)$ which consists of three parts. The first part is the size of the cover set that we want to minimize. The second part is a penalty for edges that solution s does not cover, and the third part is an extra penalty inspired from the fact that a maximal matching induces a 2-approximate solution for the vertex cover problem.

Pourhassan et al. [9] proved that an RLS with the edge-based representation and the fitness function given in 1 maintains the quality of the 2-approximation solution if the initial solution is a maximal matching in expected time $\mathcal{O}(m)$ and such result holds for (1+1) EA only if changes are limited to adding edges. For (1+1) EA and dynamic delete of an edge, the expected time $\mathcal{O}(m \log m)$ was obtained there, which is not tight. This bound is improved in this paper. The two studied algorithms of [9] for the edge-based representation are presented in Algorithms 1 and 2. In the dynamic setting that was studied in that paper, a large gap of $\tau = poly(n)$ iterations was assumed in which no dynamic changes happened. We use this setting for our analysis in Section III. In Section IV, we consider the second setting for the dynamic vertex cover problem where a dynamic change happens at each step with a certain probability. We investigate the behaviour of the two edge-based algorithms (Algorithms 1 and 2) in that section. We perform the runtime analysis with respect to the number of fitness evaluations of the algorithms.

III. Improving Re-optimisation Time of the (1+1) EA for Dynamic Vertex Cover Problem

In [9], using (1+1) EA with the edge-based representation (Algorithm 2) and the fitness function given in Equation 1, it was shown that if a 2-approximate solution is given as the initial solution, after a dynamic delete happens on the graph, the re-optimization process takes expected time $O(m \log m)$

Algorithm 2 Edge-Based (1+1) EA ((1+1) EA$_e$) [9]

1: The initial solution, s, is given: a bit-string of size m which used to be a 2-approximate solution before changing the graph.
2: Set $s' = s$
3: Flip each bit of s' independently with probability $\frac{1}{m}$
4: If $f(s') \leq f(s)$ then $s := s'$
5: If stopping criteria not met continue at line 2

to find a 2-approximate solution. However, this upper bound is not tight, and is the same as the expected time of finding a 2-approximation from an arbitrary solution. In this section, we improve the upper bound on the expected time of maintaining 2-approximation with this algorithm.

Consider a solution s that is a matching but not a maximal matching. The cover set, $V_C(s)$, derived from this solution is not a complete cover. Let k be the number of uncovered edges of solution s and C be the minimum number of vertexes that are required to be added to $V_C(s)$ to make it a complete cover. The goal is to find a maximal matching, which induces a 2-approximation. If s is a matching with $C = 0$, then it is also maximal; therefore, the goal is to find a solution that is a matching and has $C = 0$. Also let C_t denote the value of C at step t of the algorithm, and $E[\Delta_t] = E[C_t - C_{t+1}|C_t]$ denote the drift on the value of C.

Each covered edge of the graph, is either covered by one node or two nodes of the induced node set of s. Let the number of edges that are covered from both ends be $D(s)$. Moreover, let $E_i(s), 1 \leq i \leq m$, be the set of selected edges in solution s, that deselecting each of them uncovers i covered edges. Using these definitions and the following lemmata, we prove in Theorem 3 that (1+1) EA with the edge-based representation and fitness function $f(s)$ maintains a 2-approximation for the dynamic vertex cover problem in expected time $O(m)$.

According to definitions of $D(s)$ and $E_i(s)$ and the total number of covered edges, the following lemma gives us an equation that helps us in the proof of Lemma 2.

Lemma 1. *For any solution s, $|D(s)| + \sum_{i=1}^{m} i \cdot |E_i(s)| \leq m - k$, where k is the number of uncovered edges of solution s and m is the total number of edges.*

Proof. Let us first consider all covered edges except those that are in $D(s)$. By definition of $E_i(s)$, $1 \leq i \leq m$, deselecting each edge of $E_i(s)$ uncovers i edges. This implies that all of these i edges are only covered by the deselected edge and none of them is uncovered by deselecting another edge. Therefore, each covered edge that is not in $D(s)$, is counted at most once in $\sum_{i=1}^{m} i \cdot |E_i(s)|$.

On the other hand, by definition of $D(s)$, none of the edges of $D(s)$ are uncovered when one of the edges of $E_i(s)$, $1 \leq i \leq m$ is deselected. Therefore, edges of $D(s)$ are not counted in $\sum_{i=1}^{m} i \cdot |E_i(s)|$. Moreover, the number of covered edges is $m - k$, which completes the proof. □

The following lemma finds the drift on the value of C at each step of the (1+1) EA$_e$. It shows that the expected improvement of C by the algorithm at each step is lower bounded by a value depending on C.

Lemma 2. *When the solution s at step t of (1+1) EA$_e$ is a matching, the drift on C is $E[\Delta_t] \geq \frac{C}{em}$.*

Proof. We first investigate the situations where C can be increased or decreased. Then we find the expected change on the value of C that happens in each situation and use it for finding its total drift.

The value of C may only change in an step of the algorithm with an accepted move. The assumption is that we are starting with a matching, and according to Lemma 22 and Lemma 23 of [9] a matching is never replaced by a non-matching or a matching with a greater number of uncovered edges. Therefore, only solutions with at most the same number of uncovered edges are accepted by the algorithm. We here define conditions on the accepted steps to help us with investigating the changes on C.

- $Condition\ Q$: Mutations happen on uncovered edges only.
- $Condition\ R$: Mutations happen on both covered and uncovered edges.

Let $E^*_C(s)$ and $E^*_U(s)$ be the set of mutating edges that are covered and the set of mutating edges that are uncovered by solution s, respectively. For both conditions we have $|E^*_U(s)| \geq 1$. Furthermore, when condition Q holds, we have $|E^*_C(s)| = 0$, while for condition R we have $|E^*_C(s)| \geq 1$. We show that the value of C decreases at a step of the algorithm where condition Q holds, and it can only increase if condition R holds. At each accepted step of the algorithm, if the mutation(s) only include one (or more) of the uncovered edges (i. e. $|E^*_U(s)| \geq 1$ and $|E^*_C(s)| = 0$), the value of C decreases by at least one; because each uncovered edge includes at least one of the required nodes for making a complete cover. Therefore, if condition Q is satisfied, C decreases by at least one. In other words we have

$$E[\Delta_t|Q] \geq 1.$$

Moreover, since the probability of condition Q to be satisfied is at least the probability of having only one mutation on one of the uncovered edges, for solutions that are not complete covers we have

$$P(Q) \geq \frac{k}{m}\left(1 - \frac{1}{m}\right)^{m-1} \geq \frac{k}{em}. \tag{2}$$

The number of uncovered edges of a solution s is always greater than or equal to the minimum number of nodes that have to be added to $V_C(s)$ to make it a complete cover. This implies that

$$P(Q) \geq \frac{C}{em}.$$

Now we analyse the steps where the value of C may be increased. Any mutation that removes a required node from the cover set, uncovers some edges and needs to happen together

with one (or more) mutation(s) that covers at least the same number of uncovered edges (Lemma 23 in [9]), otherwise it is rejected by the algorithm. Therefore, C can only increase at the steps where at least one of the uncovered edges is selected together with another mutation, which is the definition of condition R. Note that here we have $|E^*_C(s)| \geq 1$ and $|E^*_U(s)| \geq 1$. Similar to what we explained for condition Q, since we have $|E^*_U(s)| \geq 1$, C decreases by at least one. Furthermore, mutations on covered edges that are not selected in the current solution make the solution a non-matching; therefore, in order to be accepted, all mutations on covered edges must happen on selected edges.

Now we find the expected increase on C that happen as a result of mutating these edges (i.e. $E^*_C(s)$). Deselecting each edge happens with probability $\frac{1}{m}$ at each step. Moreover, for $1 \leq i \leq m$ we have $|E_i(s)|$ edges that uncover i edges when they are deselected. Therefore, an expected number of $\frac{|E_i(s)|}{m}$ edges of $E_i(s)$ are mutated at each step. Furthermore, deselecting an edge from $E_1(s)$ increases C by at most one, because only one edge is uncovered by that mutation. Deselecting any edge from $E_i(s)$, $2 \leq i \leq m$ increases C by at most 2, because including both nodes of the deselected edge covers all adjacent edges. Moreover, there are $|D(s)|$ edges that can only be uncovered if the selected edges of their both ends are deselected. Since the probability of deselecting two edges at one step is $\frac{1}{m^2}$, there are an expected number of $\frac{|D(s)|}{m^2}$ edges of this kind. Each of them increase the value of C by at most one, because one node is enough to cover an uncovered edge. As a result, the drift on the value of C when condition R holds is:

$$E[\Delta_C|R] \geq 1 - \frac{|D(s)|}{m^2} - \frac{|E_1(s)|}{m} - 2\sum_{i=2}^{m}\frac{|E_i(s)|}{m}.$$

In this inequality, 1 is the minimum decrease on C because we have $|E^*_U(s)| \geq 1$. Moreover, $\frac{|E_1(s)|}{m}$, $2\sum_{i=2}^{m}\frac{|E_i(s)|}{m}$ and $\frac{|D(s)|}{m^2}$ are upper bounds on expected increase of C as a result of deselecting edges in $E_1(s)$, $E_i(s)$, $2 \leq i \leq m$ and uncovering edges of $D(s)$, respectively. By replacing 2 with i and m^2 with m in the inequality above we get:

$$\begin{aligned} E[\Delta_t|R] &\geq 1 - \frac{|D(s)|}{m^2} - \frac{|E_1(s)|}{m} - 2\sum_{i=2}^{m}\frac{|E_i(s)|}{m} \\ &\geq 1 - \frac{|D(s)|}{m} - \sum_{i=1}^{m} i \cdot \frac{|E_i(s)|}{m}. \end{aligned} \tag{3}$$

By Lemma 1 we have:

$$E[\Delta_t|R] \geq 1 - \frac{m-k}{m} \geq \frac{1}{m}.$$

The last inequality holds when at least one edge is uncovered, i.e. $k \geq 1$.

The value of C can only change if either condition Q is satisfied or condition R; therefore, the drift on the value of C is

$$\begin{aligned} E[\Delta_t] &= E[\Delta_t|Q] \cdot P(Q) + E[\Delta_t|R] \cdot P(R) \\ &\geq 1 \cdot \frac{C}{em} + \frac{1}{m} \cdot P(R) \;\geq\; \frac{C}{em}. \end{aligned} \tag{4}$$

□

We now prove the main theorem of this section. A dynamic change affects the graph by either deleting an edge or adding it. However, it is already shown that (1+1) EA$_e$ maintains the quality of the solution when a new edge is added dynamically in expected time $O(m)$ [9]. Here we prove that the expected re-optimisation time of (1+1) EA$_e$ after a dynamic delete is also $O(m)$. Using the drift on C which is proven in Lemma 2, Theorem 3 shows that after an edge is dynamically deleted from a maximal matching , (1+1) EA$_e$ turns the matching into a maximal matching in linear time.

Theorem 3. *Starting with a 2-approximate solution s, which is a maximal matching, (1+1) EA$_e$ maintains the quality of the solution when one edge is dynamically deleted from the graph in expected time $O(m)$.*

Proof. Let $e = \{v_1, v_2\}$ be the edge that is deleted from the graph. If $e \notin E(s)$ then s is still a maximal matching and corresponds to a 2-approximate vertex cover. If $e \in E(s)$, then it is deleted from the solution as well. The new s is still a matching but may not be a maximal matching. The number of uncovered edges of s after the dynamic delete can be in $O(m)$, but all of them can be covered by including the two nodes of the deleted edge; therefore, $C \leq 2$ holds just after the dynamic change. Moreover, according to Lemma 2, the drift on C is at least $\frac{C}{em} \geq \frac{1}{em}$. Therefore, by additive drift analysis, we find expected time $\frac{2}{1/em} = O(m)$ to reach a solution s where $C = 0$; which implies a complete cover. Moreover, by Lemma 22 of [9], we know that a non-matching solution is never accepted by the algorithm; therefore, s is a maximal matching, which induces a 2-approximate solution. □

IV. Complexity Analysis for the Second Setting of Dynamic Vertex Cover Problem

In this section we consider the second setting for the dynamic vertex cover problem, in which a dynamic change happens on the graph at each step of the algorithm with probability $P_D \leq \frac{1}{(2+\epsilon)em}$, where $0 \leq \epsilon \leq 1$ is an arbitrary small constant. Similar to the previous section, we assume that the maximum number of edges in the graph is m. Firstly, we prove a lower bound for the performance of RLS$_e$. After that, using multiplicative drift analysis, in Theorem 5 we prove that RLS$_e$ and (1+1) EA$_e$ find a 2-approximate solution for this problem in expected time $O(m \log m)$ even if they start from an arbitrary solution. Moreover, in Theorem 6 we find the expected required time of RLS$_e$ and (1+1) EA$_e$ to maintain the quality of a 2-approximate solution after the first dynamic change happens on the graph.

The following lemma finds the drift on the value of C when RLS$_e$ is solving the problem without considering the dynamic changes. This lemma is used in the proof of Theorem 5 and 6.

Lemma 4. *When the solution s at step t of RLS$_e$ is a matching, the drift on C is $E[\Delta_t] \geq \frac{C}{m}$.*

Proof. The proof is similar to the Lemma 2. The only difference is that we only have one mutation at each step and that mutation has to be on an uncovered edge in order to be accepted, i.e. $E^*_U(s) = 1$ and $E^*_C(s) = 0$. Otherwise, it increases the number of uncovered edges. Therefore, the value of C decreases at least by one at each accepted step. Moreover, the probability of this event is k/m and since C is less than k, we have :

$$E(\Delta_t) \geq \frac{C}{m}.$$

□

Theorem 5. *Consider the dynamic vertex cover problem where an arbitrary edge is dynamically added to or deleted from the graph with probability $P_D \leq \frac{1}{(2+\epsilon)em}$ at each step, where $0 \leq \epsilon \leq 1$ is an arbitrary small constant. Starting with an arbitrary solution s, (1+1) EA$_e$ and RLS$_e$ find a 2-approximate solution for this problem in expected time $O(m \log m)$.*

Proof. We split the analysis into two phases. The algorithm finds a matching in the first phase, and finds a maximal matching in the second phase. Jansen et al. [3] have proved in Theorem 11 of their paper, that (1+1) EA$_e$ and RLS$_e$ with their specified fitness function, finds a matching in expected time $O(m \log m)$ for the static version of the problem. In their proof they show that deselecting an edge that shares a node with another selected edge, improves the fitness and is always accepted until the selected edges do not share nodes. They show that this problem is easier than OneMax; therefore, the algorithm finds a matching in expected time $O(m \log m)$. The changes that happen on the graph in the dynamic version of the problem (adding an edge or removing an edge) do not increase the number of shared nodes between the selected edges. Therefore, their proof holds for the dynamic vertex cover problem as well, i.e. the first phase needs expected time $O(m \log m)$.

Now we analyse the second phase. The drift on the value of C in the dynamic setting that we are analysing in this section consists of the expected changes that the algorithms makes on C in addition to the expected changes that are caused by the dynamic changes of the graph. We denote the latter by $E(\Delta_D)$. Lemma 2 and 4 give us the drift on C that is a result of running (1+1) EA$_e$ and RLS$_e$, respectively. This value is at least $\frac{C}{em}$ for both of these algorithms, because $m \leq em$. Therefore, for the total drift on C we have

$$E(\Delta_t) \geq \frac{C}{em} + E(\Delta_D). \tag{5}$$

Now we find $E(\Delta_D)$. We consider an upper bound on Δ_D which denotes the difference between C_t and C_{t+1} ($\Delta_D = C_t - C_{t+1}$) that is a result of a dynamic change in a step t. If a new edge is added to the graph by a dynamic change and one of its adjacent edges is already selected, then the new edge is covered. Otherwise, it can be covered by adding only one vertex to the cover set. The other case is that an edge is dynamically deleted from the graph. If the deleted edge had been selected in the soluion, then this change may uncover many edges. However, adding the vertices of the deleted edge to the cover set guarantees to cover the uncovered edges again. Hence, a dynamic change increases the value of C by at most 2, i.e. $\Delta_D \geq -2$. Since a dynamic change happens at each step with probability P_D, we have

$$E(\Delta_D) \geq -2 \cdot P_D \geq \frac{-2}{(2+\epsilon)em}. \tag{6}$$

Using Equations 6 in 5 we have the total drift on C as

$$E(\Delta_t) \geq \frac{C}{em} - \frac{2}{(2+\epsilon)em}.$$

Knowing that $C \geq 1$ we find

$$E(\Delta_C) \geq \frac{(2+\epsilon)C - 2}{(2+\epsilon)em} \geq \frac{\epsilon \cdot C}{(2+\epsilon)em}.$$

Let T be the first hitting time $T = \min\{t | C_t = 0\}$, $C_0 \leq m$ the initial value of C, and $\delta = \frac{\epsilon}{(2+\epsilon)em}$ where $0 \leq \epsilon \leq 1$ is an arbitrary constant. By multiplicative drift we have

$$E(T|C_0) \leq \frac{\ln m + 1}{\frac{\epsilon}{(2+\epsilon)em}} = O(m \log m).$$

This completes the proof. □

Theorem 6. *Consider the dynamic vertex cover problem where an arbitrary edge is dynamically added to or deleted from the graph with probability $P_D \leq \frac{1}{(2+\epsilon)em}$ at each step, where $0 \leq \epsilon \leq 1$ is an arbitrary small constant. Starting with a 2-approximate solution s, which is a maximal matching, (1+1) EA$_e$ and RLS$_e$ maintain the quality of the solution in expected time $O(m)$.*

Proof. This theorem can be seen as a special case of Theorem 5. As it is proved in Theorem 5, a dynamic change increases the value of C by at most 2. Since solution is a maximal matching before the dynamic change happens, we can use the multiplicative drift analysis that we had in Theorem 6 with initial value of $C_0 = 2$. Therefore, we have:

$$E(T|C_0) \leq \frac{\ln 2 + 1}{\frac{\epsilon}{(2+\epsilon)em}} = O(m).$$

□

By Theorems 5 and 6, we have proved that (1+1) EA$_e$ and RLS$_e$ find the first 2-approximate solution for the dynamic vertex cover problem with the defined setting in expected time $O(m \log m)$, while this quality can be maintained in $O(m)$ after a dynamic change happens on the graph.

V. CONCLUSION

The expected required time for re-optimization process of the vertex cover problem after a dynamic change had been analysed in [9] for the (1+1) EA and RLS. In this paper, we improved the results of the analysis of (1+1) EA. Moreover, we have investigated another setting for the dynamic version of the problem, in which the instance of the problem is subject to a change at each step with probability P_D. We have proved that starting from an arbitrary solution, RLS and (1+1) EA achieve a maximal matching which is a 2-approximate solution in expected time $\mathcal{O}(m \log m)$, when the dynamic changes take place with probability $P_D \leq \frac{1}{(2+\epsilon)em}$. Furthermore, with the same setting, we have proved that the two investigated algorithms maintain the quality of 2-approximation, in expected time of $O(m)$.

ACKNOWLEDGMENT

This research has been supported by the Australian Research Council (ARC) grants DP140103400 and DP160102401.

REFERENCES

[1] S. Droste. Analysis of the (1+1) EA for a dynamically changing ONEMAX-variant. In *Evolutionary Computation, 2002. CEC '02. Proceedings of the 2002 Congress on*, volume 1, pages 55–60, May 2002.

[2] 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.

[3] T. Jansen, P. S. Oliveto, and C. Zarges. Approximating vertex cover using edge-based representations. In F. Neumann and K. A. D. Jong, editors, *Foundations of Genetic Algorithms XII, FOGA '13, Adelaide, SA, Australia, January 16-20, 2013*, pages 87–96. ACM, 2013.

[4] 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*, FOGA '15, pages 40–51, New York, NY, USA, 2015. ACM.

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

[6] F. Neumann and C. Witt. On the runtime of randomized local search and simple evolutionary algorithms for dynamic makespan scheduling. In *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15, pages 3742–3748. AAAI Press, 2015.

[7] 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.

[8] 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*, FOGA '17, pages 37–44, New York, NY, USA, 2017. ACM.

[9] 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*, GECCO '15, pages 903–910, New York, NY, USA, 2015. ACM.

[10] 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 of Parallel Problem Solving from Nature – PPSN XIV*, pages 729–739. Springer International Publishing, 2016.

[11] 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*, GECCO '17, pages 1407–1414, New York, NY, USA, 2017. ACM.

[12] S. A. Stanhope and J. M. Daida. Genetic algorithm fitness dynamics in a changing environment. In *Proceedings of the 1999 Congress on Evolutionary Computation (CEC1999)*, pages 1851–1858, Piscataway, NJ, 1999. IEEE.