Maintaining 2-Approximations for the Dynamic Vertex Cover Problem Using Evolutionary Algorithms

Mojgan Pourhassan
Optimisation and Logistics
School of Computer Science
The University of Adelaide
Adelaide, SA 5005, Australia
mojgan.pourhassan@adelaide.edu.au

Wanru Gao
Optimisation and Logistics
School of Computer Science
The University of Adelaide
Adelaide, SA 5005, Australia
wanru.gao@adelaide.edu.au

Frank Neumann
Optimisation and Logistics
School of Computer Science
The University of Adelaide
Adelaide, SA 5005, Australia
frank.neumann@adelaide.edu.au

ABSTRACT

Evolutionary algorithms have been frequently used to deal with dynamic optimization problems, but their success is hard to understand from a theoretical perspective. With this paper, we contribute to the theoretical understanding of evolutionary algorithms for dynamic combinatorial optimization problems. We examine a dynamic version of the classical vertex cover problem and analyse evolutionary algorithms with respect to their ability to maintain a 2-approximation. Analysing the different evolutionary algorithms studied by Jansen et al. [6], we point out where two previously studied approaches are not able to maintain a 2-approximation even if they start with a solution of that quality. Furthermore, we point out that the third approach is very effective in maintaining 2-approximations for the dynamic vertex cover problem.

Categories and Subject Descriptors

F.2 [**Theory of Computation**]: Analysis of Algorithms and Problem Complexity

Keywords

Dynamic Vertex Cover Problem; Local Search; (1+1) EA; Combinatorial Optimisation.

1. INTRODUCTION

Evolutionary algorithms have been frequently applied to dynamic optimization problems. With this paper, we contribute to the theoretical understanding of evolutionary algorithms for one of the most classical combinatorial optimization problems, namely the vertex cover problem. The vertex cover problem is a well-known NP-hard combinatorial optimisation problem with various applications in scheduling, networking, bioinformatics, etc [1, 4, 9]. Several algorithms are known that achieve a 2-approximation for this problem. The goal of our investigations is to contribute to the understanding of how evolutionary algorithms can maintain a 2-approximation when dynamic changes such as edge addition and deletion are applied to the current graph. We study different variants of the classical Randomised Local Search (RLS) and (1+1) EA that have already been investigated for the static vertex cover problem in the context of approximations. This includes a node-based representation examined in [3, 8, 7] as well as different edge-based representations analyzed in [6].

For both of the representations there are hard instances introduced [3, 6] in which with high probability a 2-approximation solution can not be found in less than exponential time by means of (1+1) EA. Nevertheless, inspired by the approximation algorithms for the vertex cover problem using maximal matchings, Jansen et al. [6] have suggested that evolutionary algorithms using edge-based representation instead of the node-based representation can solve the problem faster. They have shown that with edge-based representation and a fitness function that penalizes edges sharing nodes, the algorithm can find a 2-approximation solution in $O(m \log m)$ where m denotes the number of edges in the given graph.

We adapt the three approaches of Jansen et al. [6] to the dynamic vertex cover problem [5] where edges may be added or deleted from the graph. For the first two approaches, we point out where they are not able to maintain 2-approximations for the dynamic vertex cover problem. In contrast to this, we show that the third approach maintains solutions of that quality very efficiently.

The rest of the paper is structured as follows. In Section 2 the problem definition is given and the algorithms are introduced. Section 3 and 4 include hard instances of the dynamic vertex cover problem for the node-based and edge-based approaches respectively. Run time behaviour of the edge-based approach with the complex fitness function is analysed in Section 5 and the conclusion is presented in the last section.

2. ALGORITHMS AND THE DYNAMIC VERTEX COVER PROBLEM

The vertex cover problem can be defined as follows. Given a graph $G = (V, E)$ with vertex set $V = \{v_1, \ldots, v_n\}$ and edge set $E = \{e_1, \ldots, e_m\}$, the goal is to find the minimum subset of nodes, $V_C \subseteq V$, that covers all edges, i.e. $\forall e \in E, e \cap V_C \neq \emptyset$.

ACM ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

Algorithm 1 Node-Based RLS (RLS$_{NB}$)

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

Algorithm 2 Node-Based (1+1) EA ((1+1) EA$_{NB}$)

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

Algorithm 3 Edge-Based RLS (RLS$_{EB}$)

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

Algorithm 4 Edge-Based (1+1) EA ((1+1) EA$_{EB}$)

```
1: The initial solution, $s$, is given: a bit-string of size $m$ which used to be a 2-approximation solution before changing the graph.
2: Set $s' = s$
3: Flip each bit of $s'$ 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
```

In dynamic version of vertex cover problem, the given instance is subject to the addition and deletion of edges. We assume that these changes happen one by one each τ iterations where $\tau \in poly(n)$ and $poly(n)$ is a polynomial function in n.

In most of the work on the vertex cover problem using evolutionary algorithms, the natural node-based representation is used [3, 8, 7]. In this representation the search space is $\{0,1\}^n$ where n is the number of nodes in the graph. A potential solution is a search point $s \in \{0,1\}^n$ describing a selection of nodes, i. e. the 1-bits identify the nodes that are in the cover-set for that solution:

$$V_C(s) = \{v_i \in V \mid s_i = 1\}.$$

In the work of Jansen et al. [6] the edge-based representation is introduced for this problem. In this representation the search space is $\{0,1\}^m$ where m is the number of edges in the graph, and a search point $s \in \{0,1\}^m$ describes a selection of edges $E(s) = \{e_i \in E \mid s_i = 1\}$. The cover set then is the subset of all vertices that are on either side of the selected edges:

$$V_C(s) = \{v \in V \mid \exists e \in E(s) : v \in e\}.$$

Note that in the dynamic version of the problem, the size of the bit-string corresponding to a search point increases and decreases when edges are added or removed respectively. In our analysis, m is the largest number of edges in the graph at all stages.

Jansen et al. [6] have suggested that this representation can help evolutionary algorithms solve the problem faster. They first investigated the fitness function

$$f(s) = |V_C(s)| + (|V| + 1) \cdot |\{e \in E \mid e \cap V_C(s) = \emptyset\}|. \quad (1)$$

The first part of this fitness function is the cardinality of the cover set which needs to be minimised. The second part is a penalty for the edges that this set does not cover.

For $f(s)$ of Equation 1, they have managed to show that a (1+1) EA performs equally poor in worst cases for both defined representations. Furthermore, Jansen et al. [6] have shown that with adding a penalty for adjacent edges to the fitness function, the (1+1) EA with edge-based representation can solve the vertex cover problem in $O(m \log m)$. The fitness function with an extra penalty that they have used is defined as

$$\begin{aligned} f_e(s) &= f(s) + (|V| + 1) \cdot (m + 1) \\ &\cdot |\{(e, e') \in E(s) \times E(s) \mid e \neq e', e \cap e' \neq \emptyset\}| \quad (2) \end{aligned}$$

The fitness function $f_e(s)$ is inspired by the well-known approximation algorithm that finds a 2-approximation for the vertex cover problem based on a maximal matching [2].

In this paper, we analyse the behaviour of RLS and the (1+1) EA on the dynamic vertex cover problem with similar approaches that were studied in [6] on vertex cover problem. These algorithms are supposed to modify the given solution if needed to make it keep its quality of 2-approximation, after an edge has been added to or deleted from the graph. In our runtime analysis, we measure runtime by the number of fitness evaluations to reach a certain goal. The expected runtime refers to the expected number of fitness evaluations to reach the desired goal. In our case, the goal is to recompute a 2-approximation after one or more sequentially applied dynamic changes have occurred.

The Node-Based RLS (RLS$_{NB}$) and Node-Based (1+1) EA ((1+1) EA$_{NB}$) are defined in Algorithm 1 and Algorithm 2, respectively. Similarly, Edge-Based RLS (RLS$_{EB}$) and Edge-Based (1+1) EA ((1+1) EA$_{EB}$) using the fitness function $f(s)$ are presented in Algorithm 3 and 4, respectively. The definition of the algorithms for the third approach is quite similar to the second approach except that for comparing two solutions in line 4, where instead the fitness function with the extra penalty of Formula 2 is used. In the following sections, we denote the RLS and (1+1) EA variants of the third approach by RLS$_e$ and (1+1) EA$_e$, respectively.

3. HARD INSTANCE FOR NODE-BASED APPROACH

In this section, we introduce a bipartite graph for which it is hard to maintain a 2-approximation solution by means of RLS$_{NB}$ and (1+1) EA$_{NB}$. We go even further by showing that for our instance and a sequence of dynamic changes, only a very bad approximation will be found by these two algorithms. In our instance, both of the algorithms stick to a local optimum with a bad approximation ratio of $\Omega(n^{1-\epsilon})$,

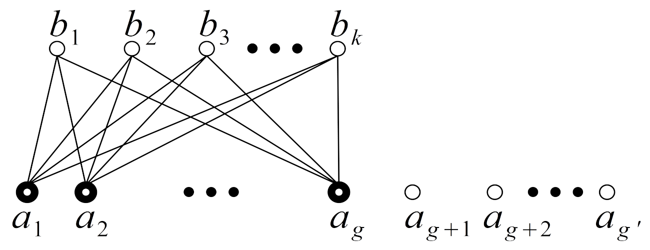

Figure 1: G_1, a hard instance for node-based approach

$\epsilon > 0$ a small constant, if the graph is subject to a polynomial number of changes. In this section, we assume that $\tau \geq n^{(3+\delta)}$, $\delta > 0$ a small constant. An Illustration of our instance, G_1, is given in Figure 1.

$G_1 = (V, E)$ is a bipartite graph and the set of nodes, V, is partitioned into two sets $W = \{a_1, \ldots, a_{g'}\}$ and $U = \{b_1, \ldots, b_k\}$. If n denotes the total number of nodes, then we assume that $k = \frac{1}{3}n^\epsilon$ and $g' = n - \frac{1}{3}n^\epsilon$. Initially, $g = 2k$ nodes from part W are connected to all the nodes of part U, i. e. the sub-graph consisting of nodes $U \cup \{a_1, \ldots, a_g\}$ and all edges between them, which we denote by ${G_1}'$, is a complete bipartite graph. The other nodes, $\{a_i \mid g+1 \leq i \leq g'\}$, are initially not adjacent to any edge, but will be connected to the nodes of part U one by one. In other words, the dynamic changes that the graph is subject to, are adding edges $\{\{a_i, b_j\} \mid g+1 \leq i \leq g', 1 \leq j \leq k\}$. Among these, edges $\{e \mid a_i \in e\}$ are added prior to edges $\{e \mid a_{i+1} \in e\}$; and edge $\{a_i, b_j\}$, is added prior to edge $\{a_i, b_{j+1}\}$. The number of these edges is $(g' - g) \cdot k = (n - n^\epsilon) \cdot (\frac{1}{3}n^\epsilon) = O(n^{1+\epsilon})$.

Property 1. *The optimal solution for G_1 at all stages of dynamic changes, is the set U of size k.*

Proof. At all stages, the sub-graph G_1' is a complete bipartite graph; and a vertex cover for a complete bipartite graph, contains at least all of the nodes of one of the parts. Therefore, either all the nodes of U need to be in the solution or all the nodes $\{a_i \mid 1 \leq i \leq g\}$. Since $g = 2k$, any cover set containing $\{a_i \mid 1 \leq i \leq g\}$, has a size of at least $2k$; whereas any cover set containing U has a size of at least k. Therefore, the optimal solution has a size of at least k.

On the other hand, the set U is a complete cover for G_1 at all the stages because it is one of the partite sets of G_1. Therefore, the optimal cover set at all the stages of the graph is U with a size of k. $\square$

The initial 2-approximation solution that is given consists of $\{a_i \mid 1 \leq i \leq g\}$. This is a 2-approximation solution because $g = 2k$ and the optimal solution has a size of k as we saw in Property 1. In Sections 3.1 and 3.2 we show that algorithms RLS$_{NB}$ and (1+1) EA$_{NB}$ find a locally optimal solution consisting of all nodes of W when the dynamic changes are done and G_1 is a complete bipartite graph.

Based on the fitness function $f(s)$ that is used in the node-based approach, we bring 6 lemmata here that hold for both RLS$_{NB}$ and (1+1) EA$_{NB}$, and help us with the proofs in Section 3.1, and 3.2.

Lemma 2. *If the number of uncovered edges by the current solution s is A, then any solution s' with B uncovered edges is rejected by RLS$_{NB}$ and (1+1) EA$_{NB}$ if $B > A$.*

Proof. Recalling Equation 1, $f(s) = |V_C(s)| + (n+1) \cdot A$. Since $B > A$, $f(s') \geq |V_C(s')| + (n+1) \cdot (A+1)$. Moreover, the maximum and minimum value of $|V_C(s)|$ and $|V_C(s')|$ are n and 0 respectively. As a result $f(s) \leq n + (n+1) \cdot A$ and $f(s') \geq 0 + (n+1) \cdot (A+1)$. Since $(n+1) \cdot (A+1) > (n+1) \cdot A$, this upper and lower bounds on $f(s)$ and $f(s')$ imply that $f(s') > f(s)$ and s' will be rejected by RLS$_{NB}$ and (1+1) EA$_{NB}$. $\square$

Lemma 3. *If the current solution s, is a cover, any solution s' where $|V_C(s')| > |V_C(s)|$, is rejected by RLS$_{NB}$ and (1+1) EA$_{NB}$.*

Proof. For any solution s' the following inequality holds:

$$f(s') \geq |V_C(s')|$$

Since solution s is a cover, we have $f(s) = |V_C(s)|$. Therefore, $|V_C(s')| > |V_C(s)|$ implies that $f(s') > f(s)$, which results in rejecting s' by RLS$_{NB}$ and (1+1) EA$_{NB}$. $\square$

Lemma 4. *If the solution s is a cover, with probability $1 - e^{-\Omega(n^\epsilon)}$, RLS$_{NB}$ and (1+1) EA$_{NB}$ find a solution s' which is a minimal cover, in time $O(n^{1+\epsilon} \log n)$.*

Proof. According to Lemma 2, any solution that is accepted by RLS$_{NB}$ or (1+1) EA$_{NB}$ after solution s, is a cover. Assume s' to be a cover with $|V_C(s')| < |V_C(s)|$. According to Lemma 3, s' is better in terms of fitness and will replace solution s. Since s is not a minimal cover, there are some extra nodes in it, removing which does not uncover any edges while reduces the size of the cover set. The process of removing extra nodes from the solution is similar to optimizing OneMax [6] and is expected to be done in time $O(n \log n)$ by RLS and (1+1) EA since $|V_C(s)| \leq n$.

If the expected time until all of extra nodes are removed is $Cn \log n$, where C is a constant, and if X is the first time that they are removed from the solution, then by Markov's inequality

$$Prob(X \geq 2Cn \log n) \leq \frac{1}{2}$$

Considering n^ϵ phases of $2Cn \log n$ steps, then

$$Prob(X \geq 2Cn^{1+\epsilon} \log n) \leq \left(\frac{1}{2}\right)^{n^\epsilon}$$

As a result, with a probability in $1 - e^{-\Omega(n^\epsilon)}$ a minimal cover will be found in time $O(n^{1+\epsilon} \log n)$. $\square$

Lemma 5. *If the solution s is a cover before the new edge e is added, then with probability $1 - e^{-\Omega(n^\epsilon)}$ starting with s, (1+1) EA$_{NB}$ and RLS$_{NB}$ find a solution s' which is also a cover after e is added, in time $O(n^{1+\epsilon})$.*

Proof. Since s had been a cover before e was added, the only edge that might not be covered by s is e. Therefore, the number of uncovered edges is at most 1. This number does not increase according to Lemma 2 during the process of RLS$_{NB}$ and (1+1) EA$_{NB}$. On the other hand, there are always two nodes (included in the uncovered edge itself) that adding at least one of them to the solution s, results in a cover. This move is accepted according to Lemma 2 and has the probability of $\Omega(n^{-1})$ for both RLS$_{NB}$ and (1+1) EA$_{NB}$. Therefore, the expected time until this improvement is found is Cn, C a positive constant, and using Markov's inequality and n^ϵ phases of $2 \cdot Cn$ steps (similar to proof of Lemma 4), with probability $1 - e^{-\Omega(n^\epsilon)}$, a cover will be found in time $O(n^{1+\epsilon})$. $\square$

LEMMA 6. *Consider the given solution s, a cover which does not include $b_{j'}$; $3 \leq j' \leq k$, before new edge $e = \{a_i, b_j\}$ is added to the graph. With probability $1 - e^{-\Omega(n^\epsilon)}$, the resulting solution of (1+1) EA$_{NB}$ and RLS$_{NB}$ after e is added, includes at least all nodes $a_{i'}$; $1 \leq i' \leq i-1$.*

PROOF. As s does not include $b_{j'}$; $3 \leq j' \leq k$, it must contain all $a_{i'}$; $1 \leq i' \leq i-1$; otherwise, it is not a cover before $e = \{a_i, b_j\}$ is added. Therefore, the only edge that might not be covered after e is added, is e itself. The number of uncovered edges does not increase during the process of (1+1) EA$_{NB}$ and RLS$_{NB}$ (Lemma 2); therefore, none of a-nodes can be removed from the solution unless all b-nodes are added at the same step. This move is not possible with RLS$_{NB}$ because only single-bit flips can be done in that algorithm. (1+1) EA$_{NB}$ also needs to flip at least $k-2$ bits at one step which has the probability of at most $e^{-\Omega(k)} = e^{-\Omega(n^\epsilon)}$. As a result, starting from s, with probability $1 - e^{-\Omega(n^\epsilon)}$, any solution accepted by (1+1) EA$_{NB}$ and RLS$_{NB}$ after e is added, includes at least all nodes $a_{i'}$; $1 \leq i' \leq i-1$. □

LEMMA 7. *Consider $g < i \leq g'$ and two stages of the graph G_1: stage X at which $\{a_i, b_2\}$ is added, and stage Y which is before $\{a_{i+1}, b_1\}$ is added. At all stages from X to Y, a solution consisting of $\{a_1, \dots, a_i\}$ is a locally optimal solution for RLS$_{NB}$. Moreover, with probability of $1 - e^{-\Omega(n^\epsilon)}$, (1+1) EA$_{NB}$ can not improve this solution in a polynomial number of steps.*

PROOF. Since edges connected to $a_{i'}$; $i' > i$ have not been added to the graph yet, s, consisting of $\{a_1, \dots, a_i\}$, is a cover. Therefore, any solution that has at least one uncovered edge (Lemma 2) or has a larger number of nodes in the cover set(Lemma 3) is rejected by RLS$_{NB}$ and (1+1) EA$_{NB}$.

Since the sub-graph consisting of the nodes $\{a_1, \dots, a_i\} \cup U$ and all edges between them, is a complete bipartite graph, any solution which is a cover, must contain either U or $\{a_1, \dots, a_i\}$. Similar to Lemma 6, jumping to any solution which contains U, in one step by RLS$_{NB}$ is not possible and by (1+1) EA$_{NB}$ has a probability of $e^{-\Omega(n^\epsilon)}$. Moreover, among solutions containing the set $\{a_1, \dots, a_i\}$, s has the minimum number of nodes in the cover set. Therefore, s is a local optimum for RLS$_{NB}$ and with probability $e^{-\Omega(n^\epsilon)}$, (1+1) EA$_{NB}$ can not improve it in a polynomial number of steps. □

3.1 Analysis of RLS$_{NB}$ on G$_1$

In this section, we first bring two lemmata that help us identify local optimums. Using them, we analyse the behaviour of RLS$_{NB}$ on G_1.

LEMMA 8. *Any solution s which is a minimal cover, is a locally optimal solution for RLS$_{NB}$.*

PROOF. Only single-bit flips can be preformed by RLS$_{NB}$ which can be either adding a new node to s or removing one from it. Since s is a cover, adding new nodes to it is rejected according to Lemma 3 because it increases $|V_C(s)|$. Moreover, any move that removes a node, uncovers at least one edge as s is a minimal cover. Therefore, according to Lemma 2 deleting nodes from s also is rejected; hence, s is a local optimum. □

LEMMA 9. *Consider $g < i \leq g'$ and two stages of the graph G_1: stage X at which $\{a_i, b_1\}$ is added, and stage Y which is before $\{a_{i+1}, b_1\}$ is added. If the given solution of RLS$_{NB}$ at stage X is $\{a_1, \dots, a_{i-1}\}$, then with probability $1 - e^{-\Omega(n^\epsilon)}$, the resulting solution of the algorithm at stage Y is $\{a_1, \dots, a_i\}$.*

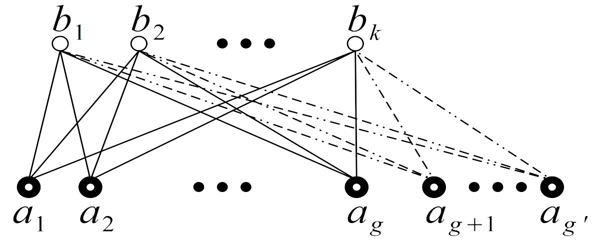

Figure 2: A solution consisting of set W

PROOF. After stage X, when the edge $\{a_i, b_1\}$ is added to the graph, the current solution, $\{a_1, a_2, \dots, a_{i-1}\}$, is not a cover any more.

The RLS$_{NB}$ flips bits of search point s, one bit at a time, to achieve an improvement on the fitness $f(s)$. Flipping a 1 to 0 indicates removing a node from the solution; which uncovers k edges and according to Lemma 2, is rejected. Flipping a 0 to 1 increases $|V_C(s)|$, but if the corresponding node covers the new edge; then the fitness is improved by n. If not, the fitness is increased by 1. Therefore, the only flips that are accepted by RLS$_{NB}$ are the ones that add a node that covers the new edge. The only such nodes are a_i and b_1. Note that in both cases the resulting solution is worse than a 2-approximation, because it contains $i \geq 2k+1$ nodes.

At each step, the probability of selecting one of these two nodes is $\frac{2}{n}$. Therefore, the expected time until one of them is selected is $\frac{n}{2}$. Let X be the first time that one of them is selected. Using Markov's inequality and n^ϵ phases of n steps (similar to proof of Lemma 4) with probability $1 - e^{-\Omega(n^\epsilon)}$, the solution is improved in $n^{1+\epsilon}$ steps.

If a_i is added, then according to Lemma 7 the solution is a local optimum until edge $\{a_{i+1}, b_1\}$ is added. Here we consider adding b_1 to the cover set.

At this stage, i.e. before the next edge is added to the graph, the solution $\{a_1, a_2, \dots, a_{i-1}, b_1\}$, is a minimal cover and a local optimum according to Lemma 8.

Similar to the first one, when the edge $\{b_2, a_i\}$ is added to the graph, the current solution is no more a complete cover. Either b_2 or a_i need to be added to the cover set. The situation is similar to what we had for the first change and will be repeated in the next stages while the b-node is selected.

For each of the k edges $\{a_i, b_j\}; 1 \leq j \leq k$, the probability that the b-node is added to the cover set instead of a_i is $\frac{1}{2}$. Therefore, with probability $1 - (\frac{1}{2})^k = 1 - e^{-\Omega(n^\epsilon)}$, at least one of these changes results in adding a_i to the selected set of nodes. Let us assume that the $\{a_i, b_j\}$ is the first edge for which a_i is added to the cover set. The new solution $\{a_1, \dots, a_i, b_1, \dots, b_{j-1}\}$ is a cover, but not a minimal cover, because removing b-nodes from the set does not uncover any edges. According to Lemma 4, they will be removed from the solution and with a probability in $1 - e^{-\Omega(n^\epsilon)}$ the locally optimal solution of $\{a_1, \dots, a_i\}$ (Lemma 7) will be found in time $O(n^{1+\epsilon} \log n)$. □

THEOREM 10. *For G_1, with probability $1 - e^{-\Omega(n^\epsilon)}$, the eventual resulting solution of RLS$_{NB}$ is the set W. The approximation ratio of this solution is in $\Omega(n^{1-\epsilon})$.*

PROOF. Since the initial solution is $\{a_1, \dots, a_g\}$, according to Lemma 9, the resulting solution of RLS$_{NB}$ before edge $\{a_{g+2}, b_1\}$ is added is $\{a_1, \dots, a_{g+1}\}$, with probability $1 - e^{-\Omega(n^\epsilon)}$. This satisfies the requirement for Lemma 9 again, and the algorithm repeats the whole process for each of $(n - n^\epsilon)$ a-nodes. As a result, after node is added, the algorithm finds a solution consisting of W with probability $1 - (n - n^\epsilon)e^{-\Omega(n^\epsilon)} = 1 - e^{-\Omega(n^\epsilon)}$ (Figure 2).

There are $g = n - \frac{1}{3}n^\epsilon$ nodes in this solution, which gives the approximation ratio of:

$$\frac{g}{k} = \frac{n - \frac{1}{3}n^\epsilon}{\frac{1}{3}n^\epsilon} = \Omega(n^{1-\epsilon})$$

□

3.2 Analysis of (1+1) EA on G_1

We here introduce a lemma using which the main theorem of this section about the behaviour of (1+1) EA$_{NB}$ on G_1 is proved.

LEMMA 11. *Consider $g < i \leq g'$ and two stages of the graph G_1: stage X at which $\{a_i, b_1\}$ is added, and stage Y which is before $\{a_{i+1}, b_1\}$ is added. If the given solution of (1+1) EA$_{NB}$ at stage X is $s = \{a_1, \dots, a_{i-1}\}$, then with probability $1 - e^{-\Omega(n^\delta)}$, the resulting solution of the algorithm at stage Y is $s' = \{a_1, \dots, a_i\}$.*

PROOF. The given solution $\{a_1, \dots, a_{i-1}\}$ is a cover for the graph before stage X. According to Lemma 5, after $\{a_i, b_1\}$ is added, the algorithm finds s_1 which is a cover, in polynomial time. Similarly, after $\{a_i, b_2\}$ is added, a solution s_2 that is also a cover is found in polynomial time.

On the other hand, according to Lemma 6, s_1 contains all nodes $a_{i'}$; $1 \leq i' \leq i - 1$. And according to Lemma 4, it is a minimal cover because $\tau > O(n^{1+\epsilon} \log n)$, meaning that in addition to $\{a_1, \dots, a_{i-1}\}$, it only includes one other node to cover $\{a_i, b_1\}$ i.e. either a_i or b_1. Therefore, s_1 does not include $b_{j'}$; $3 \leq j' \leq k$; which satisfies the condition of Lemma 6 for the next stage. In other words, s_2 also includes all nodes $a_{i'}$; $1 \leq i' \leq i - 1$. According to Lemma 4 it is also a minimal cover. Any minimal solution including all nodes $a_{i'}$; $1 \leq i' \leq i - 1$ which also covers $\{a_i, b_1\}$ and $\{a_i, b_2\}$, contains either a_i or b_1 and b_2. So far, we have proved that (1+1) EA$_{NB}$ finds s_2 which consists of either $V_1 = \{a_1, \dots, a_i\}$ or $V_2 = \{a_1, \dots, a_{i-1}, b_1, b_2\}$. We here show that (1+1) EA$_{NB}$ finds the solution which includes V_1.

Since both V_1 and V_2 are covers and $|V_1| < |V_2|$, the solution consisting of V_1 is less costly. Therefore, (1+1) EA$_{NB}$ does not accept a change from V_1 to V_2 but accepts the opposite move. The probability of flipping only the bits corresponding to a_i, b_1 and b_2 is

$$\left(\frac{1}{n}\right)^3 \left(1 - \frac{1}{n}\right)^{n-3} \geq \frac{1}{en^3}$$

As a result, a move from V_2 to V_1 can be performed at expected time of at most $E(T) = en^3$, where T is the first step that this move happens. Using Markov's inequality,

$$Prob(T \geq 2 \cdot en^3) \leq \frac{1}{2}$$

Considering $\frac{n^{(3+\delta)}}{2en^3}$ phases of $2en^3$ steps,

$$Prob(T \geq n^{3+\delta}) \leq \left(\frac{1}{2}\right)^{\frac{n^{(3+\delta)}}{2en^3}}$$

This implies that with probability $1 - e^{-\Omega(n^\delta)}$ the (1+1) EA$_{NB}$ finds s_2 consisting of V_1 in a phase of τ steps.

From this point according to Lemma 7, s_2 is a local optimum until $\{a_{i+1}, b_1\}$ is added. □

THEOREM 12. *For G_1, with probability $1 - e^{-\Omega(n^\delta)}$, the eventual resulting solution of (1+1) EA$_{NB}$ is the set W. The approximation ratio of this solution is in $\Omega(n^{1-\epsilon})$.*

PROOF. Since the initial solution is $\{a_1, \dots, a_g\}$, according to Lemma 11, the resulting solution of (1+1) EA$_{NB}$ before edge $\{a_{g+2}, b_1\}$ is added is $\{a_1, \dots, a_{g+1}\}$, with probability $1 - e^{-\Omega(n^\delta)}$. This satisfies the requirement for Lemma 11 again, and the algorithm repeats the whole process for each node a_i, $g < i \leq g'$. As a result, after the last change of the graph, the algorithm finds a solution consisting of all nodes of W with a probability of at least $1 - (n - n^\epsilon)e^{-\Omega(n^\delta)} = 1 - e^{-\Omega(n^\delta)}$.

Similar to Theorem 10 the approximation ratio of this solution is $\Omega(n^{1-\epsilon})$. □

4. HARD INSTANCE FOR STANDARD EDGE-BASED APPROACH

In this section, we assume that $\tau \geq m^{(4+\delta)}$, where $\delta > 0$ is a small constant. The graph of instance G_2 that we introduce in this section as a hard instance for edge-based approach is exactly the same as G_1 with one slight difference. The difference is that $g = 2k - 1$ instead of $2k$.

PROPERTY 13. *The optimal solution for G_2 at all stages of dynamic changes, is the set U of size k.*

PROOF. The proof is similar to the proof of Property 1 on G_1. □

Since we are using the edge-based representation in this section, the initial solution needs to be a search point in $\{0, 1\}^m$, representing the set of selected edges. We assume that $\{\{a_i, b_1\} \mid 1 \leq i \leq g\}$ is the given initial set of selected edges. The cover set induced from this set is $\{b_1\} \cup \{a_i \mid 1 \leq i \leq g\}$ which has a size of $2k$; therefore, is a 2-approximation because according to Property 13 the size of the optimal solution is k. In what follows, we analyse the behaviour of RLS$_{EB}$ and (1+1) EA$_{EB}$ on G_2 with the given initial solution.

Note that Lemma 2 and Lemma 3 of Section 3 are also valid in this section, because they are based on the fitness function $f(s)$ and not the representation. We bring three other lemmata here which hold for both RLS$_{EB}$ and (1+1) EA$_{EB}$.

LEMMA 14. *Starting with the given solution s which is a cover before edge e is added, with probability $1 - e^{-\Omega(m^\epsilon)}$, RLS$_{EB}$ and (1+1) EA$_{EB}$ find a solution s' which is also a cover in $O(n^{1+\epsilon})$ after e is added.*

PROOF. The proof of this lemma is similar to the proof of Lemma 5 except that we are using the edge-based approach and we should use the probability of finding the proper edge instead of the probability of finding the proper node.

There always exists an edge (the uncovered edge itself) adding which to s results in a cover. This move has the probability of $\Omega(m^{-1})$ for both RLS$_{EB}$ and (1+1) EA$_{EB}$. Therefore, similar to Lemma 5, with probability $1-e^{-\Omega(m^{\epsilon})}$, a cover will be found in $O(m^{1+\epsilon})$. □

LEMMA 15. *For the instance G_2 starting with the given initial solution, with probability $1 - e^{-\Omega(m^{\epsilon})}$, RLS$_{EB}$ and (1+1) EA$_{EB}$ result in a solution which is a cover at all stages.*

PROOF. Using Lemma 14 as inductive steps and the initial solution as the base of induction, we can conclude that with probability $1-e^{-\Omega(m^{\epsilon})}$, RLS$_{EB}$ and (1+1) EA$_{EB}$ result in a solution which is a cover at all stages. □

LEMMA 16. *Before new edge $e = \{a_i, b_j\}$ is added to the graph, consider the given solution s, a cover which does not include $b_{j'}$; $4 \leq j' \leq k$. With probability $1-e^{-\Omega(m^{\epsilon})}$, the resulting solution of (1+1) EA$_{EB}$ and RLS$_{EB}$ after e is added, includes at least all nodes $a_{i'}$; $1 \leq i' \leq i-1$.*

PROOF. The proof is similar to proof of Lemma 6. However, notice that with edge-based representation, adding and removing nodes from the cover set of solution s can be done by adding and removing edges from s. Since G_2 is a bipartite graph, adding all nodes $b_{j'}$; $4 \leq j' \leq k$ at one step, requires at least $k = 3$ flips. □

4.1 Analysis of RLS$_{EB}$ on G_2

In this section, we analyse the behaviour of RLS$_{EB}$ on G_2 and show that there is a stage at which this algorithm fails to find a solution with a better approximation ratio than $\frac{3k-1}{k}$. We first bring a lemma that helps us with the proof.

LEMMA 17. *With probability $1-e^{-\Omega(m^{\epsilon})}$, the node b_k can only be added to the solution by RLS$_{EB}$, at a stage in which $\{a_i, b_k\}$ has been added to the graph, where $g+1 \leq i \leq g'$.*

PROOF. Consider stage X in which node b_k is added to the solution. If adding an edge $e = \{a_{i'}, b_k\}; 1 \leq i' \leq g'$ is accepted, it must cover an uncovered edge; otherwise, this move will be rejected because it increases the cardinality of the cover set. On the other hand, according to Lemma 15, the solution obtained by the algorithm before stage X, is a cover; therefore the only uncovered edge is the one that is added in stage X which we denote by e_n. As a result, e is covering e_n; i.e. $e_n = \{a_i, b_k\}$; $g+1 \leq i \leq g'$ which is what the lemma claims, or, $e_n = \{a_{i'}, b_{k'}\}$; $1 \leq k' \leq k$. In this case, $k' = k$ because otherwise, the edge $\{a_{i'}, b_k\}$ has not been added to the graph yet. This completes the proof. □

THEOREM 18. *Before edge $\{a_{3k}, b_1\}$ is added to G_2, with probability $1-e^{-\Omega(m^{\epsilon})}$, there is a stage at which RLS$_{EB}$ does not find any solution better than $\frac{3k-1}{k}$-approximation.*

PROOF. Before $\{a_{3k}, b_1\}$ is added to the graph, all edges connected to nodes a_i; $2k \leq i < 3k$, are added. We partition these stages into k phases so that in phase i ($2k \leq i < 3k$), edges $\{a_i, b_j \mid 1 \leq j \leq k\}$ are added.

We analyse the situation based on containing or not containing the node b_k in the obtained solution of at least one stage.

- If an edge containing nodes b_k is added to the solution, according to Lemma 17, it must have been added after edge $e_n = \{a_i, b_k\}$; $2k \leq i < 3k$ is added to G_2. Furthermore, adding b_k to a solution s increases $|V_C(s)|$. If s was a cover, according to Lemma 3 this move was rejected. Therefore, b_k must have covered a new edge. According to Lemma 15, before e_n is added; the algorithm has managed to find a cover; therefore, the only edge that may not be covered after adding e_n, is e_n itself. Therefore, the edge containing b_k, that is added to the solution, must cover $e_n = \{a_i, b_k\}$, to be accepted. This implies that the node a_i has not been previously added to the solution. Otherwise, e_n was already covered.

  Consider stage X, at which $\{a_i, b_{k-1}\}$ has been added. According to the above explanation, the solution obtained at this stage (before e_n is added) must not contain a_i. Moreover, according to Lemma15, this solution is a cover. Since a_i has been connected to all other b-nodes in the past stages, all the nodes b_j; $1 \leq j \leq k-1$, must be included in that solution as it does not contain a_i. Similarly, all the nodes a_l; $1 \leq l \leq i-1$ must be included in that solution because b_k is connected to all of them and is not included in the cover set yet.

  As a result, the achieved solution by the end of stage X, contains nodes a_l; $1 \leq l \leq i-1$ and b_j; $1 \leq j \leq k-1$. Since $i \geq 2k$, the total number of nodes in the cover set is at least $3k-1$.

- If b_k is not included in the solution of any stage; then the cover set must include all of the nodes that are connected to b_k. At the last stage of phase $3k-1$, all nodes $a_l, 1 \leq l \leq 3k-1$ are connected to b_k; therefore, the cover set that the algorithm finds at that stage contains at least $3k-1$ edges.

In both cases, we proved that there is a stage at which the achieved cover set contains at least $3k-1$ nodes; while the size of the optimal cover set is k at all stages. Therefore, the approximation ratio of the mentioned solutions is $\frac{3k-1}{k}$. □

4.2 Analysis of EA on G_2

Here we introduce two lemmata that helps us with the proof of the main theorem about the behaviour of (1+1) EA$_{EB}$ on G_2. Lemma 19 is obtained similar to Lemma 7.

LEMMA 19. *Consider $g < i \leq g'$ and two stages of the graph G_1: stage X at which $\{a_i, b_3\}$ is added, and stage Y which is before $\{a_{i+1}, b_1\}$ is added. At all stages from X to Y, a solution s, consisting of $\{a_1, \ldots, a_i, b_j\}$; $1 \leq j \leq k$ is a locally optimal solution for (1+1) EA$_{EB}$.*

LEMMA 20. *Consider $g < i \leq g'$ and two stages of the graph G_1: stage X at which $\{a_i, b_1\}$ is added, and stage Y which is before $\{a_{i+1}, b_1\}$ is added. If the given solution of (1+1) EA$_{EB}$ at stage X, s, includes the nodes $\{a_1, \ldots, a_{i-1}\}$, then with probability $1 - e^{-\Omega(m^{\delta})}$, the resulting solution of the algorithm at stage Y also includes $\{a_1, \ldots, a_i\}$.*

PROOF. The given solution s is a cover for the graph before stage X. According to Lemma 14, after $\{a_i, b_1\}$, $\{a_i, b_2\}$ and $\{a_i, b_3\}$ are added, the algorithm finds s_1, s_2 and s_3 which are also covering solutions, in polynomial time.

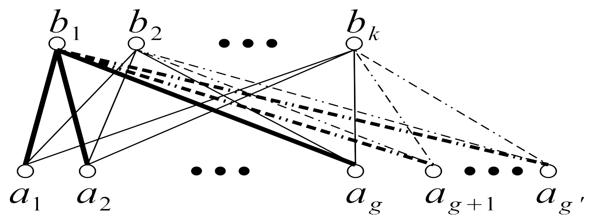

Figure 3: A solution including the set W

On the other hand, according to Lemma 16, s_1, s_2 and s_3 contain all nodes $a_{i'}$; $1 \leq i' \leq i-1$. A covering solution that contains the three new edges, must contain either a_i or b_1, b_2 and b_3. Among solutions with these properties, $s_3 = \{\{a_l, b_j\} \mid 1 \leq l \leq i\}$ has the minimum cost and is achievable from others at each step with a probability of at least $(\frac{1}{m})^4(1-\frac{1}{m})^{(m-4)}$ because at most 4 bits of s need to be flipped. Similar to proof of Lemma 11, we can conclude that with probability $1-e^{-\Omega(m^\delta)}$ the (1+1) EA$_{EB}$ finds $s_3 = \{\{a_l, b_1\} \mid 1 \leq l \leq i\}$ in a phase of τ steps which is due to Lemma 19, a local optimum until $\{a_{i+1}, b_1\}$ is added. $\square$

Similar to Theorem 12 we obtain the following result.

Theorem 21. *For G_2, with probability $1-e^{-\Omega(m^\delta)}$, the (1+1) EA$_{EB}$ finds a locally optimal solution containing the set W (Figure 3). The approximation ratio of this solution is in $\Omega(n^{1-\epsilon})$.*

5. EDGE-BASED APPROACH WITH EXTRA PENALTY

In this section, we analyse the impact of using the fitness function $f_e(s)$, defined in Equation 2, on the behaviour of RLS$_e$ and (1+1) EA$_e$. It is already proved [6] that starting from any initial solution, both of these algorithms find a maximal matching in time $O(m \log m)$, which induces a 2-approximation solution for the vertex cover problem. As a result, considering $\tau \geq m^{(1+\delta)}$, both algorithms find a 2-approximation solution for the dynamic vertex cover problem with probability $1-e^{-\Omega(\frac{m^\delta}{\log m})}$. We aim to analyse the behaviour of the two algorithms when an initial solution with that quality is given. Both kind of dynamic changes on the graph, *add* and *delete*, are analysed in this section. The following two lemmata are proved based on the fitness function $f_e(s)$ that RLS$_e$ and (1+1) EA$_e$ use.

Lemma 22. *Consider a search point $s \in \{0,1\}^m$ which is a matching. Any move that results in search point s' is rejected by RLS$_e$ and (1+1) EA$_e$ if s' is not a matching.*

Proof. We here show that for any s which is a matching and any s' which is not a matching, $f_e(s') > f_e(s)$; therefore, both algorithms reject s'.

If s' is not a matching, $f_e(s') \geq (|V|+1)\cdot(m+1) = (n+1)\cdot(m+1)$. Moreover, if s is a matching $f_e(s) = f(s) \leq n + (n+1)(m)$ because the maximum number of uncovered edges is m. On the other hand, $(n+1)\cdot(m+1) > n+(n+1)(m)$ which implies that $f_e(s') > f_e(s)$ $\square$

Lemma 23. *Consider a search point $s \in \{0,1\}^m$ which is a matching. Any move that results in search point s' is rejected by RLS$_e$ and (1+1) EA$_e$ if $|\{e \in E \mid e \cap V_C(s) = \emptyset\}| > |\{e \in E \mid e \cap V_C(s') = \emptyset\}|$.*

Proof. For any search point s', $f_e(s') \geq f(s')$ holds. Since solution s is a matching, $f_e(s) = f(s)$. Therefore, if $f(s') > f(s)$, $f_e(s') > f_e(s)$ also holds. Moreover, according to Lemma 2, if the number of uncovered edges of solution s' is larger than that of solution s, $f(s') > f(s)$ holds which completes the proof. $\square$

5.1 Analysis of RLS

In this section, using the following lemma, we prove that RLS$_e$ maintains a 2-approximation solution in $O(m)$ on expectation. This gives the probability of $1-e^{-\Omega(m^\delta)}$ for maintaining the quality of the problem with $\tau = m^{(1+\epsilon)}$.

Lemma 24. *Any search point $s \in \{0,1\}^m$ which is a maximal matching, is a locally optimal solution for RLS$_e$.*

Proof. By RLS$_e$ only single-bit flips can be performed: adding one edge to s, or deleting one edge from it. We show that both kinds are rejected; hence s is a local optimum.

Since s is a matching, removing an edge from it uncovers at least one edge, and according to Lemma 23, is rejected. And since it is a maximal matching, adding any edge to it will result in a solution which is not a matching, and according to Lemma 22, is rejected. $\square$

Theorem 25. *Starting with a 2-approximation solution s, which is also a maximal matching for an instance of the problem, RLS$_e$ maintains the quality of the solution for dynamic changes of adding or deleting an edge on the graph in expected time of $O(m)$.*

Proof. We investigate the situation for adding an edge or deleting an edge separately.

When an edge is added to the graph, s is still a matching, but it might be or not be a maximal matching. If s is still a maximal matching then it is a local optimum (Lemma 24) and a 2-approximation, because all maximal matchings induce a 2-approximation cover set.

If s is not a maximal matching, then only the new edge, e, might not be covered. According to Lemma 22 and Lemma 23, s remains matching during the process of the algorithm and the number of uncovered edges does not increase. Moreover, while there is an uncovered edge, there is a probability of at least $\frac{1}{m}$ to make an improvement, because adding the uncovered edge to s reduces the number of uncovered edges to 0. This means that in expectation, it takes m steps for RLS$_e$ to find this improvement.

When an edge, $e = \{v_1, v_2\}$, is deleted from the graph, if $e \notin E(s)$ then s is still a maximal matching and corresponds to a 2-approximation solution. If $e \in E(s)$, then it is deleted from the solution as well. The new s is still a matching but might be or not be a maximal matching. If s is still a maximal matching then it is already a 2-approximation and we are done.

We examine the case where s does not constitute a maximal matching anymore. If s is not a maximal matching, then there is a non-empty set E' such that:

$$E' = \{e_1 \mid e_1 \in E \wedge (\forall e_2 \in E(s) \Rightarrow e_1 \cap e_2 = \emptyset)\}.$$

Consider the set E'':

$$E'' = \{e_1 \mid e_1 \in E \wedge (\forall e_2 \in E(s), e_1 \cap e_2 = \emptyset) \wedge e_1 \cap e \neq \emptyset\}$$

The definition implies that $E'' \subseteq E'$. Here we show that $E' = E''$. If not, $\exists e'' \in E' \setminus E''$ which means that $e'' \cap e = \emptyset$

and s was not covering e'' before removing e, and therefore was not a maximal matching which is in contrast to the given assumption on s.

Now we can define $U_1 = \{e_1 \mid e_1 \in E'' \wedge v_1 \in e_1\}$ and $U_2 = \{e_1 \mid e_1 \in E'' \wedge v_2 \in e_1\}$. We know that $U_1 \cap U_2 = \emptyset$ because the edge containing both v_1 and v_2 was e which is deleted from the graph. Therefore, U_1 and U_2 define a partition over E'' and in order to cover edges in E'', edges from both of these sets need to be covered. All edges in U_1 include the node v_1 which implies that selecting any edge from U_1 covers all other edges from this set. Similarly selecting any edge from U_2 covers all edges from this set. Therefore, using RLS$_e$, at each step there is a probability of $\frac{|U_1|}{m}$ to cover all edges of U_1 and a probability of $\frac{|U_2|}{m}$ to cover all edges of U_2.

Note that any other move is rejected: No edge can be deleted from s, because s is a matching and deleting any edge from it increases the number of uncovered edges; therefore is rejected (Lemma 23). Furthermore, all edges other than $e'' \in E''$ are covered by s and adding them to s results in a solution which is not a matching; hence, is rejected (Lemma 22).

With the mentioned probabilities of covering U_1 and U_2 in one step, each of them will be covered in expected time of Cm, where C is a constant. Using Markov's inequality and m^ϵ phases of Cm, with probability $1 - e^{-\Omega(m^\epsilon)}$ all the uncovered edges will be covered in $O(m^{1+\epsilon})$. The new solution is a maximal matching which induces a 2-approximation solution. $\square$

5.2 Analysis of (1+1) EA

In this section, we consider the (1+1) EA$_e$ and analyse maintaining a 2-approximation solution for the dynamic vertex cover problem for that. We have obtained new results for the dynamic change of adding an edge; but deleting an edge is more complicated to analyse. The reason is that the number of uncovered edges can be as large as $O(m)$ and when more than one flip can happen at each step, some uncovered edges can get covered but a smaller number of covered edges get uncovered. The best expected optimization time for this situation known so far is $O(m \log m)$ which is the same as the expected time of (1+1) EA$_e$ starting from scratch.

The following theorem, gives the result of our analysis for (1+1) EA$_e$ when edges are dynamically added to the graph.

THEOREM 26. *Starting with a 2-approximation solution s, which is also a maximal matching for an instance of the problem, (1+1) EA$_e$ maintains the quality of the solution when one new edge is dynamically added to the graph in expected time of $O(m)$.*

PROOF. The proof is similar to the proof of the first part of Theorem 25, except that the probability of flipping the bit of uncovered edge in (1+1) EA$_e$ is $\frac{1}{m}(1-\frac{1}{m})^{(m-1)}$. However, this probability also gives the expected time of $O(m)$ to make the improvement. $\square$

6. CONCLUSION

In this paper, we have carried out rigorous runtime analyses on how the different evolutionary approaches already examined by Jansen et al. [6] (for the static vertex cover problem) can deal with the dynamic vertex cover problem. For the first two examined approaches, we have presented classes of instances of bipartite graphs where adding edges lead to bad approximation behaviours even if the algorithms started with a 2-approximation. For the third approach, we have shown that 2-approximations are maintained easily by recomputing maximal matchings of the dynamically changing graph.

Acknowledgements

The research leading to these results has received funding from the Australian Research Council (ARC) under grant agreements DP130104395 and DP140103400.

7. REFERENCES

[1] N. Bansal and S. Khot. Inapproximability of hypergraph vertex cover and applications to scheduling problems. In S. Abramsky, C. Gavoille, C. Kirchner, F. Meyer auf der Heide, and P. Spirakis, editors, *Automata, Languages and Programming*, volume 6198 of *Lecture Notes in Computer Science*, pages 250–261. Springer Berlin Heidelberg, 2010.

[2] T. H. Cormen, C. Stein, R. L. Rivest, and C. E. Leiserson. *Introduction to Algorithms*. McGraw-Hill Higher Education, 2nd edition, 2001.

[3] 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.

[4] S. Guha, R. Hassin, S. Khuller, and E. Or. Capacitated vertex covering. *JOURNAL OF ALGORITHMS*, 48:257–270, 2003.

[5] Z. Ivković and E. Lloyd. Fully dynamic maintenance of vertex cover. In J. van Leeuwen, editor, *Graph-Theoretic Concepts in Computer Science*, volume 790 of *Lecture Notes in Computer Science*, pages 99–111. Springer Berlin Heidelberg, 1994.

[6] 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*, FOGA XII '13, pages 87–96, New York, NY, USA, 2013. ACM.

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

[8] 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.

[9] S. Pirzada and A. Dharwadker. Applications of graph theory. *Journal of Korean society for Inductrial and applied mathematics*, 11(4):19–38, 2007.