The Minimum Generalized Vertex Cover Problem

REFAEL HASSIN

Tel-Aviv University

AND

ASAF LEVIN

The Hebrew University

Abstract. Let $G = (V, E)$ be an undirected graph, with three numbers $d_0(e) \geq d_1(e) \geq d_2(e) \geq 0$ for each edge $e \in E$. A solution is a subset $U \subseteq V$ and $d_i(e)$ represents the cost contributed to the solution by the edge e if exactly i of its endpoints are in the solution. The cost of including a vertex v in the solution is $c(v)$. A solution has cost that is equal to the sum of the vertex costs and the edge costs. The minimum generalized vertex cover problem is to compute a minimum cost set of vertices. We study the complexity of the problem with the costs $d_0(e) = 1$, $d_1(e) = \alpha$ and $d_2(e) = 0$ $\forall e \in E$ and $c(v) = \beta$ $\forall v \in V$, for all possible values of α and β. We also provide 2-approximation algorithms for the general case.

Categories and Subject Descriptors: G.2.2 [**Discrete Mathematics**]: Graph Theory

General Terms: Algorithms

Additional Key Words and Phrases: Vertex cover, local-ratio, complexity classification

1. *Introduction*

Let $G = (V, E)$ be an undirected graph. The MINIMUM VERTEX COVER PROBLEM is to find a minimum size vertex set $S \subseteq V$ such that for every $(i, j) \in E$ at least one of i and j belongs to S.

In the minimum vertex cover problem, it makes no difference if we cover an edge by both its endpoints or by just one of its endpoints. In this article, we generalize the problem and an edge incurs a cost that depends on the number of its endpoints that belong to S.

For every edge $e \in E$, we are given three numbers $d_0(e) \geq d_1(e) \geq d_2(e) \geq 0$, and for every vertex $v \in V$, we are given a number $c(v) \geq 0$.

An extended abstract version of this article appeared in *Proceedings of the 11th Annual European Symposium on Algorithms, 2003*.

Authors addresses: R. Hassin, Department of Statistics and Operations Research, Tel-Aviv University, Tel-Aviv 69978, Israel, e-mail: hassin@post.tau.ac.il; A. Levin, Department of Statistics, the Hebrew University, Jerusalem 91905, Israel, e-mail: levinas@mscc.huji.ac.il.

For a subset $S \subseteq V$, denote $\bar{S} = V \setminus S$, $E(S)$ is the set of edges whose both end-vertices are in S, $E(S, \bar{S})$ is the set of edges that connects a vertex from S with a vertex from $\bar{S}$, $c(S) = \sum_{v \in S} c(v)$, and for $i = 0, 1, 2$ $d_i(S) = \sum_{e \in E(S)} d_i(e)$ and $d_i(S, \bar{S}) = \sum_{e \in E(S,\bar{S})} d_i(e)$.

The MINIMUM GENERALIZED VERTEX COVER PROBLEM (GVC) is to find a vertex set $S \subseteq V$ that minimizes the cost $c(S) + d_2(S) + d_1(S, \bar{S}) + d_0(\bar{S})$. Thus, the value $d_i(e)$ represents the cost of the edge e if exactly i of its endpoints are included in the solution, and the cost of including a vertex v in the solution is $c(v)$.

Note that GVC generalizes the unweighted MINIMUM VERTEX COVER PROBLEM that is the special case with $d_0(e) = 1,\ d_1(e) = d_2(e) = 0\ \forall e \in E$ and $c(v) = 1\ \forall v \in V$.

A motivation for this problem is the following (see Paik and Sahni [1995] and Krumke et al. [1999]); Let $G = (V, E)$ be an undirected graph. For each vertex $v \in V$ we can upgrade v at a cost $c(v)$. For each edge $e \in E$ $d_i(e)$ represents the cost of the edge e if exactly i of its endpoints are upgraded. The goal is to find a subset of upgraded vertices, such that the sum of upgrading and edge costs, is minimized.

Using this illustration, we use the term *upgraded vertex*, to denote a vertex that is included in the solution, and *nonupgraded vertex* to denote a vertex that is not included in the solution.

Paik and Sahni [1995] and Krumke, et al. [1999] considered bicriteria problems where we can upgrade each vertex at some cost, and the cost of each edge depends on the number of its endpoints that are upgraded. In Paik and Sahni [1995], the objective is to minimize the size of the set of upgraded vertices (equal upgrading costs), such that a given set of performance criteria will be met (e.g., for every pair of vertices u, v, the shortest path between u and v has length at most Δ). In Krumke et al. [1999], the problem is the following: given a budget that can be used to upgrade vertices and the goal is to upgrade a vertex set such that in the resulting network the minimum cost spanning tree is minimized.

When $d_0(e) = 1, d_1(e) = \alpha, d_2(e) = 0\ \forall e \in E$ and $c(v) = \beta\ \forall v \in V$, we obtain the MINIMUM UNIFORM COST GENERALIZED VERTEX COVER PROBLEM (UGVC). Thus, the input to UGVC is an undirected graph $G = (V, E)$ and a pair of constants α (such that $0 \leq \alpha \leq 1$) and β. The cost of a solution $S \subseteq V$ for UGVC is $\beta|S| + |E(\bar{S})| + \alpha|E(S, \bar{S})|$.

The PROVISIONING PROBLEM is a related problem that we use. Suppose there are n items to choose from, where item j costs $c_j \geq 0$. Also suppose there are m sets of items $S_1, S_2, \ldots, S_m$. If all the items in set S_i are chosen, then a benefit of $b_i \geq 0$ is gained. The objective is to maximize the net benefit, that is, total benefit gained minus total cost of items purchased. The problem was shown in Lawler [1976, pp. 125–127) to be solvable in polynomial time.

1.1. OUR RESULTS. We study the complexity of UGVC for all possible values of α and β. The shaded areas in Figure 1 illustrate the polynomial-time solvable cases, whereas all the other cases are NP-hard. The numbers in each region refers to the lemma that provides a polynomial algorithm or proves the hardness of the problem in that region. We provide two 2-approximation algorithms for GVC, one is based on linear programming relaxation, and the other one runs in linear time and is based on the local-ratio technique.

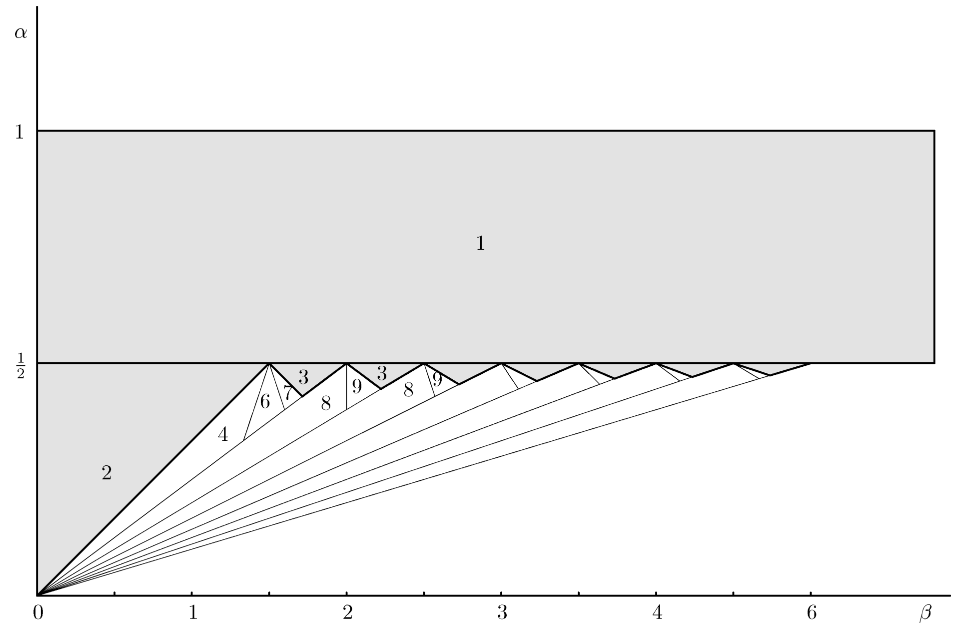

FIG. 1. The complexity of UGVC.

2. *The Complexity of UGVC*

In this section, we study the complexity of UGVC.

LEMMA 1. *If $\frac{1}{2} \leq \alpha \leq 1$, then UGVC can be solved in polynomial time.*

PROOF. If $\frac{1}{2} \leq \alpha \leq 1$, then UGVC is reducible to the provisioning problem as follows: The items are the vertices of the graph each has a cost of β. The sets are of two types: a single item $\{v\}$ for every vertex $v \in V$, and a pair $\{u, v\}$ of vertices for every edge $(u, v) \in E$. A set of a single vertex $\{v\}$ has a benefit of $(1 - \alpha) deg(v)$ and a set that is a pair of vertices has a benefit of $2\alpha - 1 \geq 0$. Then, the sum of costs of the provisioning solution and the corresponding UGVC solution, is always m, that is, the number of edges in E. To see this fact, note that we can allocate the benefit of choosing a vertex v_i among its adjacent edges so each edge gain a benefit of $1 - \alpha$. In case both vertices of an edge (u, v) were chosen, then the sets of the singletons $\{u\}$ and $\{v\}$ already results a benefit of $2(1 - \alpha)$ and the set $\{u, v\}$ causes an additional gain of $2\alpha - 1$ and thus the total gain for such an edge is exactly 1. So if we choose k vertices, then by paying βk in the provisioning problem (this is exactly as their cost in UGVC) we gain $1 - \alpha$ for each edge that is covered once, and 1 for each edge that is covered twice.

Therefore, optimizing the first objective (provisioning) is equivalent to optimizing the second objective (UGVC). $\square$

For a graph G, a *leaf* is a vertex with degree 1.

LEMMA 2. *If $\alpha < \frac{1}{2}$ and $\beta \leq 3\alpha$, then UGVC can be solved in polynomial time.*

PROOF. We first observe that upgrading one end of an edge saves $1 - \alpha$, and if also the other end of the edge is upgraded then the additional saving is α. By assumption, $\alpha < \frac{1}{2}$ and therefore $\alpha < 1 - \alpha$.

By assumption $\beta \leq 3\alpha$, and therefore it is optimal to upgrade all the vertices whose degree is greater than or equal to 3. We will analyze the solution for the leaves and vertices with degree 2.

If two leaves u and v are connected by an edge, then if $\beta > 1 - \alpha$ it is optimal not to upgrade u and v, if $1 - \alpha \geq \beta > \alpha$ it is optimal to upgrade u and not to upgrade v, and if $\beta \leq \alpha$ it is optimal to upgrade u and v.

If $\beta \leq \alpha$, then it is optimal to upgrade all the vertices.

If $\alpha < \beta \leq 2\alpha$, then it is optimal to upgrade all the vertices that have degree 2 and not to upgrade the leaves.

If $\beta > 2(1 - \alpha)$, then all the vertices that have degree 1 or 2 are not upgraded.

Assume that $2\alpha < \beta \leq 3\alpha$ and that $\beta \leq 2(1 - \alpha)$. The vertices with degree 1 or 2 induce disjoint cycles, paths that connect a pair of vertices with degree at least 3, paths that connect a vertex with degree at least 3 and a leaf, and paths that connect a pair of leaves.

We first assume that $\beta \geq 1$. In this case, it is not optimal to upgrade adjacent vertices with degree 1 or 2. Therefore, the upgraded set is an independent set.

—A cycle of k vertices causes a cost of $t(2\alpha + \beta) + k - 2t = k + t(\beta - 2(1 - \alpha))$ if we upgrade an independent set of size t. This cost is minimized when t is maximized. Number the vertices along the cycle $v_1, v_2, \ldots, v_k$. Then, the vertices with even index constitute a maximum independent set and upgrading them is optimal.

—For a maximal path of $k + 1$ vertices $v_0, v_1, \ldots, v_k$ with degree 1 or 2, such that, if $k = 1$, then either v_0 or v_1 has degree 2, we upgrade all the odd index vertices (not including v_k, if k is odd). This is an optimal solution due to the following: The cost of this path if we upgrade an independent set with t inner vertices is exactly $t(2\alpha + \beta) + k - 2t - 2 = k - 2 + t[\beta - 2(1 - \alpha)]$, which is minimized when t is maximized.

Assume that $\beta < 1$. In this case, a solution is not optimal if it does not upgrade two adjacent vertices. Therefore, an optimal solution upgrades a vertex cover of the edges with an endpoint with degree at most 2.

—For a cycle of k vertices, we number the vertices along the cycle $v_1, v_2, \ldots, v_k$ and we upgrade all the vertices that have an odd index. The upgraded set is a minimum size vertex cover. The cost of the cycle if we upgrade a vertex cover with t vertices is $t\beta + (2k - 2t)\alpha = t(\beta - 2\alpha) + 2k\alpha$, and it is minimized when t is minimized.

—For a maximal path of $k + 1$ vertices, $v_0, v_1, \ldots, v_k$ that have degree 1 or 2, such that, if $k = 1$, then either v_0 or v_1 has degree 2, we upgrade all the odd index vertices. If k is odd and v_k has degree 1, then we upgrade v_{k-1} instead of v_k. The upgraded set is a minimum size vertex cover. The cost of the path if we upgrade a vertex cover with t inner-vertices is $t\beta + (2k - 2t)\alpha = t(\beta - 2\alpha) + 2k\alpha$, and it is minimized when t is minimized. □

LEMMA 3. *If $\alpha < \frac{1}{2}$ and there exists an integer $d \geq 3$ such that $d(1 - \alpha) \leq \beta \leq (d + 1)\alpha$ then UGVC can be solved in polynomial time.*

PROOF. Simply upgrade a vertex if and only if its degree is at least $d+1$. We now argue that this is an optimal solution. First, given a solution that does not upgrade a vertex with degree at least $d+1$, then we improve its cost if we decide to upgrade it (the change is improvement by at least $(d+1)\alpha - \beta \geq 0$). Second, given a solution that upgrades a vertex v with degree at most d, then the solution resulting from the previous one such that it does not upgrade v, we improve its cost by at least $\beta - d(1-\alpha) \geq 0$. $\square$

If Lemma 1, Lemma 2, and Lemma 3 can not be applied, then UGVC is NP-hard. We will divide the proof into several cases.

LEMMA 4. *If $\alpha < \frac{1}{2}$ and $3\alpha < \beta \leq 1+\alpha$, then UGVC is NP-hard even when G is 3-regular.*

PROOF. Assume that G is 3-regular and assume a solution to UGVC that upgrades k vertices. Because of the lemma's assumptions, if there is an edge $(u, v) \in E$ such that both u and v are not upgraded, then it is better to upgrade u (resulting in an improvement of at least $1-\alpha+2\alpha-\beta = 1+\alpha-\beta \geq 0$). Therefore, without loss of generality, the solution is a vertex cover (if $\beta = 1-\alpha$, then not all the optimal solutions are vertex covers; however, it is easy to transform a solution into a vertex cover without increasing the cost). Since there are $2|E|-3k$ edges such that exactly one of their endpoints is upgraded, the cost of the solution is $\beta k + \alpha(2|E|-3k) = k(\beta-3\alpha)+2\alpha|E|$. Since $\beta > 3\alpha$, the cost of the solution is a strictly monotone increasing function of k. Therefore, finding an optimal solution to UGVC for G is equivalent to finding a minimum vertex cover for G. The MINIMUM VERTEX COVER PROBLEM restricted to 3-regular graphs is NP-hard (see problems [GT1] and [GT20] in Garey and Johnson [1979]). $\square$

LEMMA 5. *If $\alpha = 0$ and $\beta > 0$, then UGVC is NP-hard.*

PROOF. If $\beta \leq 1$, then by the proof of Lemma 4, UGVC is NP-hard. Assume that $\beta > 1$. Let G be an input graph to MINIMUM VERTEX COVER PROBLEM. Replace each edge by $\lceil\beta\rceil$ copies and denote the resulting multigraph by G'. Then, an optimal solution to UGVC on G' is an optimal solution to the vertex cover problem on G. $\square$

LEMMA 6. *If $\alpha < \frac{1}{2}$ and $1+\alpha < \beta < 2-\alpha$, then UGVC is NP-hard even when G is 3-regular.*

PROOF. Assume that the input to UGVC with α, β satisfying the lemma's conditions, is a 3-regular graph $G = (V, E)$. By local optimality of the optimal solution for a vertex v, v is upgraded if and only if at least two of its neighbors are not upgraded: If v has at least two nonupgraded neighbors, then upgrading v saves at least $2(1-\alpha)+\alpha-\beta = 2-\alpha-\beta > 0$; if v has at least two upgraded neighbors then upgrading v adds to the total cost at least $\beta-2\alpha-(1-\alpha) = \beta-(1+\alpha) > 0$.

We will show that the following decision problem is NP-complete: Given a 3-regular graph G and a number K, is there a solution to UGVC with cost at most K. The problem is clearly in NP. To show completeness we present a reduction from NOT-ALL-EQUAL-3SAT PROBLEM.

The NOT-ALL-EQUAL-3SAT is defined as follows (see Garey and Johnson [1979]): given a set of clauses $S = \{C_1, C_2, \ldots, C_p\}$ each with exactly 3 literals, is there a truth assignment such that each clause has at least one true literal and at least one false literal.

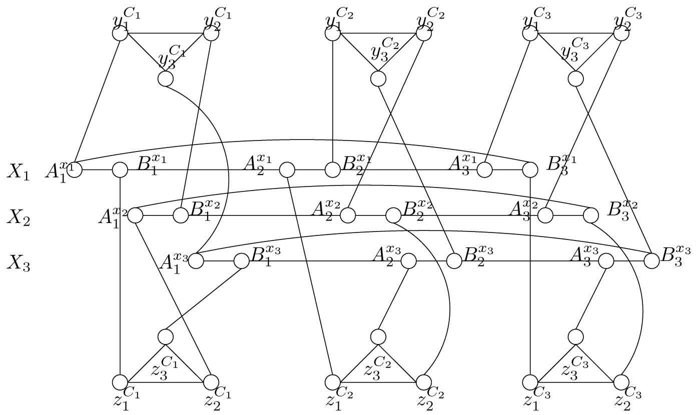

FIG. 2. The graph G obtained for the clauses $C_1 = x_1 \vee \bar{x_2} \vee x_3$, $C_2 = \bar{x_1} \vee x_2 \vee \bar{x_3}$, and $C_3 = x_1 \vee x_2 \vee \bar{x_3}$.

Given a set $S = \{C_1, C_2, \ldots, C_p\}$ each with exactly 3 literals, construct a 3-regular graph $G = (V, E)$ as follows (see Figure 2, see the max-cut reduction in Yannakakis [1981] for similar ideas): For a variable x that appears in $p(x)$ clauses, G has $2p(x)$ vertices $A_1^x, \ldots, A_{p(x)}^x, B_1^x, \ldots, B_{p(x)}^x$ connected in a cycle $A_1^x, B_1^x, A_2^x, B_2^x, \ldots, A_{p(x)}^x, B_{p(x)}^x, A_1^x$. In addition, for every clause C let G have six vertices $y_1^C, y_2^C, y_3^C, z_1^C, z_2^C, z_3^C$ connected in two triangles y_1^C, y_2^C, y_3^C and z_1^C, z_2^C, z_3^C. Each set of 3 vertices corresponds to the literals of the clause. If x appears in a clause C, and let y_j^C and z_j^C correspond to x then we assign to this appearance of x a distinct pair A_i^x, B_i^x (distinct i for each appearance of x or $\bar{x}$) and we connect y_j^C to A_i^x and z_j^C to B_i^x. If $\bar{x}$ appears in a clause C, and let y_j^C and z_j^C correspond to x then we assign to this appearance of $\bar{x}$ a distinct pair A_i^x, B_i^x and we connect y_j^C to B_i^x and z_j^C to A_i^x.

Note that G is 3-regular.

For a 3-regular graph, we charge the upgrading cost of an upgraded vertex to its incident edges. Therefore, the cost of an edge such that both its endpoints are upgraded is $\frac{2\beta}{3}$, the cost of an edge such that exactly one of its endpoints is upgraded is $\frac{\beta}{3} + \alpha$, and the cost of an edge such that none of its endpoints is upgraded is 1. Note that by the conditions on α and β, $\frac{\beta}{3} + \alpha < \frac{2\beta}{3}$ because by assumption $\beta > 1 + \alpha \geq 3\alpha$. Also, $\frac{\beta}{3} + \alpha < \frac{2-\alpha}{3} + \alpha = \frac{2}{3}(1 + \alpha) < 1$. Therefore, the cost of an edge is minimized if exactly one of its endpoints is upgraded.

We will show that there is an upgrading set with total cost of at most $(|E| - 2p)(\frac{\beta}{3} + \alpha) + p\frac{2\beta}{3} + p$ if and only if the NOT-ALL-EQUAL-3SAT instance can be satisfied.

Assume that S is satisfied by a truth assignment T. If $T(x) = \mathit{TRUE}$, then we upgrade B_i^x $i = 1, 2, \ldots, p(x)$ and do not upgrade A_i^x $i = 1, 2, \ldots, p(x)$. If $T(x) = \mathit{FALSE}$, then we upgrade A_i^x $i = 1, 2, \ldots, p(x)$ and do not upgrade B_i^x $i = 1, 2, \ldots, p(x)$. For a clause C, we upgrade all the y_j^C vertices that correspond to TRUE literals and all the z_j^C vertices that correspond to FALSE literals. We note that the edges with either both endpoints upgraded or both not upgraded are all triangle's edges. Note also that, for every clause, there is exactly one edge connecting a pair of upgraded vertices and one edge connecting a pair of non-upgraded vertices. Therefore, the total cost of the solution is exactly $(|E| - 2p)(\frac{\beta}{3} + \alpha) + p\frac{2\beta}{3} + p$.

Assume that there is an upgrading set U whose cost is at most $(|E|-2p)(\frac{\beta}{3}+\alpha)+p\frac{2\beta}{3}+p$. Let $\bar{U}=V\setminus U$. Denote an upgraded vertex by U-vertex and a non-upgraded vertex by $\bar{U}$-vertex. Without loss of generality, assume that U is a local optimum. Therefore, a U-vertex has at most one U-neighbor as otherwise, if a U-vertex w has at least two neighbors in U, the solution $U\setminus\{w\}$ has a reduced cost because the cost decrease by at least $\beta-(1-\alpha)-2\alpha<0$ where the last inequality holds because $\beta>1+\alpha$, and this contradicts the fact that U is a local optimum. Similarly, a $\bar{U}$-vertex has at most one $\bar{U}$-neighbor. To see this last claim, note that otherwise if a $\bar{U}$-vertex w has at least two neighbors in $\bar{U}$, the solution $U\cup\{w\}$ has a reduced cost because the cost decrease by at least $-\beta+2(1-\alpha)+\alpha>0$ where the last inequality holds because $\beta<2-\alpha$, and this contradicts the fact that U is a local optimum.

Therefore, for a triangle y_1^C, y_2^C, y_3^C (z_1^C, z_2^C, z_3^C) at least one of its vertices is in U and at least one of its vertices is in $\bar{U}$. Therefore, in the triangle, there is exactly one edge that connects either two U-vertices or two $\bar{U}$-vertices and the two other edges connect a U-vertex to a $\bar{U}$-vertex.

We will show that in G there are at least p edges that connect a pair of U-vertices and at least p edges that connect a pair of $\bar{U}$-vertices. Otherwise, there is a clause C such that, for some j, either y_j^C, z_j^C are both in U or both in $\bar{U}$. Without loss of generality, assume that y_j^C is connected to A_i^x and z_j^C is connected to B_i^x. Assume $y_j^C, z_j^C \in U$ $(y_j^C, z_j^C \in \bar{U})$, then, by the local optimality of the solution, $A_i^x, B_i^x \in \bar{U}$ $(A_i^x, B_i^x \in U)$, as otherwise y_j^C or z_j^C will have two U-($\bar{U}$-)neighbors and therefore we will not upgrade (will upgrade) them. Therefore, the edge (A_i^x, B_i^x) connects a pair of $\bar{U}$ (U) vertices. We charge every clause for the edges in the triangles corresponding to it that connect either two U-vertices or two $\bar{U}$-vertices, and we also charge the clause for an edge (A_i^x, B_i^x) as in the above case. Therefore, we charge every clause for at least one edge that connects two U-vertices and for at least one edge that connects two $\bar{U}$-vertices. These charged edges are all disjoint. Therefore, there are at least p edges that connect two U-vertices and at least p edges that connect two $\bar{U}$-vertices.

Since the total cost is at most $(|E|-2p)(\frac{\beta}{3}+\alpha)+p\frac{2\beta}{3}+p$, there are exactly p edges of each such type. Therefore, for every clause C for every j there is exactly one of the vertices y_j^C or z_j^C that is upgraded. Also note that for every variable x either $A_i^x \in U$, $B_i^x \in \bar{U}$ $\forall i$ or $A_i^x \in \bar{U}$, $B_i^x \in U$ $\forall i$. If $B_i^x \in U$ $\forall i$, we assign to x the value TRUE and otherwise we assign x the value FALSE. We argue that this truth assignment satisfies S. In a clause C, if $y_j^C \in U$, then its nontriangle neighbor is not upgraded and therefore, the literal corresponding to y_j^C is assigned a TRUE value. Similarly, if $y_j^C \in \bar{U}$, the literal is assigned a FALSE value. Since in every triangle at least one vertex is upgraded and at least one vertex is not upgraded, there is at least one FALSE literal and at least one TRUE literal. Therefore, S is satisfied. $\square$

LEMMA 7. *If* $\alpha<\frac{1}{2}$, $2-\alpha\le\beta<3(1-\alpha)$, *then UGVC is NP-hard even when G is 3-regular.*

PROOF. Assume that G is 3-regular and assume a solution to UGVC that upgrades k vertices. Let $v\in V$. Because of the lemma's assumptions if any of v's neighbors is upgraded then not upgrading v saves at least $\beta-2(1-\alpha)-\alpha=\beta-(2-\alpha)\ge 0$. Therefore, without loss of generality, the solution is an independent set (if $\beta = 2-\alpha$, then not all the optimal solutions are independent sets; however, it is easy to transform a solution into an independent set without increasing the cost). The cost of the solution is exactly $\beta k + 3k\alpha + (|E| - 3k) = |E| - k[3(1-\alpha) - \beta]$. Since $3(1-\alpha) > \beta$, the cost of the solution is strictly monotone decreasing function of k. Therefore, finding an optimal solution to UGVC for G is equivalent to finding an optimal independent set for G. The MAXIMUM INDEPENDENT SET PROBLEM restricted to 3-regular graphs is NP-hard (see problem [GT20] in Garey and Johnson [1979]). $\square$

LEMMA 8. *If $\alpha < \frac{1}{2}$ and $d\alpha < \beta \leq \min\{d\alpha + (d-2)(1-2\alpha), (d+1)\alpha\}$ for some integer $d \geq 4$, then UGVC is NP-hard.*

PROOF. Let $G = (V, E)$ be a 3-regular graph that is an input to the MINIMUM VERTEX COVER PROBLEM. Since $d\alpha < \beta \leq d\alpha + (d-2)(1-2\alpha)$, there is an integer k, $0 \leq k \leq d-3$, such that $d\alpha + k(1-2\alpha) < \beta \leq d\alpha + (k+1)(1-2\alpha)$.

We generate from G a graph $G' = (V', E')$ by adding k new neighbors (new vertices) to every vertex $v \in V$. From G', we generate a graph G'' by repeating the following for every vertex $v \in V$: add $d-k-3$ copies of a star centered at a new vertex with $d+1$ leaves such that v is one of them and the other leaves are new vertices.

Since $\beta \leq (d+1)\alpha$, without loss of generality, in an optimal solution of UGVC on G'' every such center of a star is upgraded. Consider a vertex $u \in V'' \setminus V$, then u is either a center of a star or a leaf. If u is a leaf, then since $\beta > \alpha$ then an optimal solution does not upgrade u.

In G'' every vertex from V has degree $3 + k + (d-k-3) = d$ and in an optimal solution for the upgrading problem, at least one of the endpoints of every edge $(u, v) \in E$ is upgraded as otherwise u will have at least $k+1$ nonupgraded neighbors, and since $\beta \leq d\alpha + (k+1)(1-2\alpha)$, it is optimal to upgrade u.

Assume the optimal solution upgrades l vertices from V. The total cost of upgrading the l vertices and the cost of edges incident to vertices from V is $l\beta + lk\alpha + (n-l)k + (n-l)(d-k-3)\alpha + (2|E| - 3l)\alpha = l[\beta + \alpha(k - d + k) - k] + n(k + (d-k-3)\alpha) + 2|E|\alpha$. Since $\beta > k(1-\alpha) + (d-k)\alpha$, the cost is strictly monotone increasing function of l. Therefore, to minimize the upgrading network cost is equivalent to finding a minimum vertex cover for G. Therefore, UGVC is NP-hard. $\square$

LEMMA 9. *If $\alpha < \frac{1}{2}$ and $d\alpha + (d-2)(1-2\alpha) \leq \beta < \min\{d\alpha + d(1-2\alpha), (d+1)\alpha\}$ for some integer $d \geq 4$, then UGVC is NP-hard.*

PROOF. Let $G = (V, E)$ be 3-regular graph that is an input to the MAXIMUM INDEPENDENT SET PROBLEM. Since $d\alpha + (d-2)(1-2\alpha) \leq \beta < d\alpha + d(1-2\alpha)$, $d\alpha + (d-k-1)(1-2\alpha) \leq \beta < d\alpha + (d-k)(1-2\alpha)$ holds for either $k = 0$ or for $k = 1$.

If $k = 1$, we add to every vertex $v \in V$ a star centered at a new vertex with $d+1$ leaves such that v is one of them. Since $\beta \leq (d+1)\alpha$, in an optimal solution the star's center is upgraded.

For every vertex in V, we add $d-k-3$ new neighbors (new vertices). Consider a vertex $u \in V'' \setminus V$ then u is either a center of a star or a leaf. If u is a leaf, then since $\beta \geq d\alpha + (d-2)(1-2\alpha) > 1-\alpha$, an optimal solution does not upgrade u.

Denote the resulting graph G'. The optimal upgrading set S in G' induces an independent set over G because if $u, v \in S \cap V$ and $(u, v) \in E$, then u has at least

$k+1$ upgraded neighbors and therefore since $d\alpha + (d-k-1)(1-2\alpha) \le \beta$, it is better not to upgrade u.

Assume the optimal solution upgrades l vertices from V. The total cost of upgrading the l vertices and the cost of edges incident to vertices from V is: $nk\alpha + (d-3-k)n + \frac{3n}{2} - l[k\alpha + (d-k)(1-\alpha) - \beta]$. Since $\beta < d\alpha + (d-k)(1-2\alpha)$, the cost is strictly monotone decreasing function of l, and therefore, it is minimized by upgrading a maximum independent set of G. Therefore, UGVC is NP-hard. □

We summarize the results:

THEOREM 10. *In the following cases, UGVC is polynomial:*

(1) *If* $\alpha \ge \frac{1}{2}$.
(2) *If* $\alpha < \frac{1}{2}$ *and* $\beta \le 3\alpha$.
(3) *If* $\alpha < \frac{1}{2}$ *and there exists an integer* $d \ge 3$ *such that* $d(1-\alpha) \le \beta \le (d+1)\alpha$.

Otherwise, UGVC is NP-hard.

3. *Approximation Algorithms*

In this section, we present two 2-approximation algorithms for the GVC problem. We present an approximation algorithm to GVC based on LP relaxation. We also present another algorithms with reduced time complexity based on the local-ratio technique.

Remark 11. If $d_0(i,j) - d_2(i,j) \ge 2[d_0(i,j) - d_1(i,j)]$ holds for every $(i,j) \in E$, then GVC can be solved in polynomial time.

PROOF. We use the PROVISIONING PROBLEM: each vertex $i \in V$ is an item with cost $\max\{0, c(i) - \sum_{j:(i,j)\in E}[d_0(i,j) - d_1(i,j)]\}$, and each edge $\{i,j\}$ is a set with benefit $d_0(i,j) - d_2(i,j) - 2[d_0(i,j) - d_1(i,j)] = 2d_1(i,j) - d_0(i,j) - d_2(i,j) \ge 0$. □

3.1. 2-APPROXIMATION FOR GVC.

For the following formulation, we explicitly use the fact that every edge $e \in E$ is a subset $\{i,j\}$ where $i, j \in V$. Consider the following integer program (**GVCIP**):

$$\min \sum_{i=1}^{n} c(i)x_i + \sum_{\{i,j\}\in E} (d_2(i,j)z_{ij} + d_1(i,j)(y_{ij} - z_{ij}) + d_0(i,j)(1 - y_{ij}))$$

subject to:

$$\begin{aligned}
y_{ij} &\le x_i + x_j && \forall \{i,j\} \in E\\
y_{ij} &\le 1 && \forall \{i,j\} \in E\\
z_{ij} &\le x_i && \forall i \in V, \{i,j\} \in E\\
x_i &\le 1 && \forall i \in V\\
x_i, y_{ij}, z_{ij} & \quad \text{integers} && \forall \{i,j\} \in E.
\end{aligned}$$

In this formulation, x_i is an indicator variable that is equal to 1 if we upgrade vertex i; y_{ij} is an indicator variable that is equal to 1 if at least one of the vertices i and j is upgraded; z_{ij} is an indicator variable that is equal to 1 if both i and j are upgraded; $y_{ij} = 1$ is possible only if at least one of the variables x_i or x_j is equal to 1; $z_{ij} = 1$ is possible only if both x_i and x_j equal 1; If y_{ij} or z_{ij} can be equal to 1, then there exists an optimal solution such that they will be equal to 1 since $d_2(i, j) \leq d_1(i, j) \leq d_0(i, j)$. Denote by **GVCLP** the continuous (LP) relaxation of **GVCIP**.

In order to define a basic solution of **GVCLP**, one first transforms the problem into the standard form of linear programming. This transformation adds one new slack variable for each constraint. So in the resulting problem the number of variables is larger than the number of constraints. Denote the number of independent constraints of **GVCLP** by K, then a basic solution is identified by setting all variables beside K variables to zero and then solving the resulting system of equations.

The following theorem generalizes a theorem by Nemhauser and Trotter [1975] for the minimum unweighted vertex cover problem.

THEOREM 12. *Let* (x, y, z) *be an optimal basic solution of* **GVCLP**. *Then,* $x_i \in \{0, \frac{1}{2}, 1\}\ \forall i$.

Hochbaum [2002] presented a set of integer programs denoted as **IP2** that contains **GVCIP**. For **IP2**, Hochbaum showed that the basic solutions to the LP relaxations of such problems are half-integral, and the relaxations can be solved using network flow algorithm in $O(mn)$ time. The following is a direct proof of Theorem 12:

PROOF. Denote by $S_1 = \{i \in V | x_i = 0\}$, $S_2 = \{i \in V | 0 < x_i < \frac{1}{2}\}$, $S_3 = \{i \in V | x_i = \frac{1}{2}\}$, $S_4 = \{i \in V | \frac{1}{2} < x_i < 1\}$, and $S_5 = \{i \in V | x_i = 1\}$. We prove that $S_2 \cup S_4 = \emptyset$. Assume otherwise, and define an $\epsilon > 0$ such that $\epsilon < x_i < \frac{1}{2} - \epsilon\ \forall i \in S_2$, and $\frac{1}{2} + \epsilon < x_i < 1 - \epsilon\ \forall i \in S_4$. Since (x, y, z) is optimal, $y_{ij} = \min\{x_i + x_j, 1\}$ and $z_{ij} = \min\{x_i, x_j\}$. We will show that if $S_2 \cup S_4 \neq \emptyset$, then there are two feasible points (x', y', z') and (x'', y'', z'') such that $x' \neq x''$ and (x, y, z) is their middle point, and this leads to a contradiction to the assumption that (x, y, z) is a basic solution. Define:

$$x'_i = \begin{cases} x_i - \epsilon & i \in S_4 \\ x_i + \epsilon & i \in S_2 \\ x_i & \text{otherwise} \end{cases} \qquad x''_i = \begin{cases} x_i + \epsilon & i \in S_4 \\ x_i - \epsilon & i \in S_2 \\ x_i & \text{otherwise.} \end{cases}$$

Define $y'_{ij} = \min\{x'_i + x'_j, 1\}$, $y''_{ij} = \min\{x''_i + x''_j, 1\}$, $z'_{ij} = \min\{x'_i, x'_j\}$, and $z''_{ij} = \min\{x''_i, x''_j\}$.

(x', y', z') and (x'', y'', z'') are feasible solutions. If $S_2 \cup S_4 \neq \emptyset$, then these are two feasible points which are different from (x, y, z), such that (x, y, z) is their middle point. This contradicts the assumption that (x, y, z) is a basic solution. □

Theorem 12 provides a 2-approximation algorithm:

(1) Solve **GVCLP** using Hochbaum's [2002] algorithm, and denote by x^*, y^*, z^* its optimal solution.
(2) Upgrade vertex i if and only if $x_i^* \geq \frac{1}{2}$.

THEOREM 13. *The above algorithm is an* $O(mn)$*-time* 2*-approximation algorithm for GVC.*

PROOF. Denote by $x_i^a = 1$, if we upgrade vertex i and $x_i^a = 0$; otherwise, $y_{ij}^a = \min\{x_i^a + x_j^a, 1\} = \max\{x_i^a, x_j^a\}$, and $z_{ij}^a = \min\{x_i^a, x_j^a\}$. The performance guarantee of the algorithm is derived by the following argument:

$$\begin{aligned}
&\sum_{i=1}^{n} c(i)x_i^a + \sum_{(i,j)\in E} \left(d_2(i,j)z_{ij}^a + d_1(i,j)\left(y_{ij}^a - z_{ij}^a\right) + d_0(i,j)\left(1 - y_{ij}^a\right)\right) \\
&\le 2\sum_{i=1}^{n} c(i)x_i^* + \sum_{(i,j)\in E} \left(d_2(i,j)z_{ij}^a + d_1(i,j)\left(y_{ij}^a - z_{ij}^a\right) + d_0(i,j)\left(1 - y_{ij}^a\right)\right) \\
&\le 2\sum_{i=1}^{n} c(i)x_i^* + \sum_{(i,j)\in E} \left(d_2(i,j)z_{ij}^* + d_1(i,j)\left(y_{ij}^* - z_{ij}^*\right) + d_0(i,j)\left(1 - y_{ij}^*\right)\right) \\
&< 2\Big(\sum_{i=1}^{n} c(i)x_i^* + \sum_{(i,j)\in E} \left(d_2(i,j)z_{ij}^* + d_1(i,j)\left(y_{ij}^* - z_{ij}^*\right) + d_0(i,j)\left(1 - y_{ij}^*\right)\right)\Big).
\end{aligned}$$

The first inequality holds because we increase x_i by a factor which is at most 2. The second inequality holds because the second sum is a convex combination of $d_0(i, j)$, $d_1(i, j)$, and $d_2(i, j)$. Since $d_0(i, j) \ge d_1(i, j) \ge d_2(i, j)$, $z_{ij}^a = \min\{x_i^a, x_j^a\} \ge \min\{x_i^*, x_j^*\} \ge z_{ij}^*$, and $1 - y_{ij}^a = \max\{1 - x_i^a - x_j^a, 0\} \le \max\{1 - x_i^* - x_j^*, 0\} = 1 - y_{ij}^*$, the second inequality holds. □

3.2. A LINEAR-TIME 2-APPROXIMATION FOR GVC-BASED ON THE LOCAL-RATIO TECHNIQUE. We next show a different 2-approximation algorithm whose analysis is based on the local-ratio technique [Bar-Yehuda and Even 1981; Bar-Yehuda et al. 2004][1]. Without loss of generality, we assume that $d_2(e) = 0$ for all edges e. Our algorithm extends the algorithm of Bar-Yehuda and Even [1981] that applies for the vertex cover problem and the algorithm of Bar-Yehuda and Rawitz [2001] that applies to the special case of GVC where $d_1(e) = d_2(e) = 0$ for all edges e (this is a generalization of vertex cover in which we pay $d_0(e)$ for not covering an edge e).

We define cost functions for the GVC problem where the cost of a vertex w is $C(w)$ and for an edge e' the cost of covering it i times is $C_i(e')$. In particular, for an edge $e = \{u, v\}$ and a positive number $\epsilon > 0$ we define the following functions:

$$C^1(w) = \begin{cases} \epsilon & w \in \{u, v\} \\ 0 & \text{otherwise} \end{cases} \qquad C_1^1(e') = 0\ \forall e' \qquad C_0^1(e') = \begin{cases} \epsilon & e' = e \\ 0 & \text{otherwise} \end{cases}$$

$$C^2(w) = \begin{cases} \epsilon & w = u \\ 0 & \text{otherwise} \end{cases} \qquad C_0^2(e') = C_1^2(e') = \begin{cases} \epsilon & e' = e \\ 0 & \text{otherwise.} \end{cases}$$

We also let $C_2^i(e) = 0$ for all edges e and for $i \in \{1, 2\}$. In order to use the local-ratio technique we note that any solution is a 2-approximation with respect to any edge e and the associated the cost functions C^1 and C^2 for all $\epsilon > 0$. This is so since any solution costs with respect to these cost functions either ϵ or 2ϵ, and therefore it is a 2-approximation.

[1] Another 2-approximation algorithm that runs in linear time and is based on the primal-dual scheme, appears in Hassin and Levin [2003].

We use the following Local-Ratio Theorem:

THEOREM 14 [BAR-YEHUDA AND EVEN 1981]. *If a feasible solution is an r-approximation with respect to a pair of weight functions W_1 and W_2 then it is also an r-approximation with respect to the weight function $W_1 + W_2$.*

Our algorithm is as follows:

1. Initialize the cost function C so that $C(v) = c(v)$ for all $v \in V$, and $C_2(e) = d_2(e)$, $C_1(e) = d_1(e)$, and $C_0(e) = d_0(e)$ for all $e \in E$.
2. **While** there is an edge $e = \{u, v\}$, $\epsilon > 0$ and $i \in \{1, 2\}$ such that $C - C^i \geq 0$ and the cost of each edge is monotone nondecreasing in the number of its end-vertices that belong to the solution. (i.e., for each edge e' $C_0(e') - C_0^i(e') \geq C_1(e') - C_1^i(e') \geq C_2(e') - C_2^i(e')$) **do:**
   $C \leftarrow C - C^i$.
3. Return the set of zero cost vertices.

By Theorem 14, to show that the resulting solution is a 2-approximation, it suffices to show that it is an optimal solution with respect to the final cost function C. To show this last claim we will show that its cost is zero (and since C is nonnegative this is an optimal solution).

LEMMA 15. *Assume that C is the cost function at the end of the algorithm. Then the resulting solution has a zero cost with respect to C.*

PROOF. The selected vertices have zero cost, and therefore do not contribute a positive amount to the cost of the solution. It remains to consider the edge costs. Consider an edge $e = \{u, v\}$.

—If both u and v are in the solution, then e contributes $C_2(e) = 0$.

—If exactly one of u and v belong to the solution, then assume it is u. The edge e contributes $C_1(e)$. We next claim that $C_1(e) = 0$. Assume otherwise and consider the cost function C^2 for $\epsilon = \min\{C_1(e), C(v)\}$. Then, by the fact that the cost of e is monotone nondecreasing in the number of its end-vertices that belong to the solution, $C - C^2$ is a nonnegative cost function. Moreover, the cost of each edge in $C - C^2$ is monotone nondecreasing in the number of its end-vertices that belong to the solution. Therefore, this contradicts the exit conditions from the **while** loop. Therefore, $C_1(e) = 0$ and the edge e does not contribute a positive amount to the cost of the resulting solution.

—If both u and v do not belong to the solution, we claim that $C_0(e) = 0$. We assume otherwise.

  —Assume that $C_1(e) > 0$. Consider the cost function C^2 for $\epsilon = \min\{C_1(e), C(u)\}$. Then, by the fact that the cost of e is monotone nondecreasing in the number of its end-vertices that belong to the solution, we conclude that $C - C^2$ is a nonnegative cost function, and the cost of each edge is monotone nondecreasing in the number of its end-vertices that belong to the solution. Therefore, this contradicts the exit conditions from the **while** loop. Therefore, the edge e does not contribute a positive amount to the cost of the resulting solution.

  —Assume that $C_1(e) = 0$. Consider the cost function C^1 for $\epsilon = \min\{C_0(e), C(u), C(v)\}$. Then, $C - C^1$ is a non-negative cost function and the cost of each edge is monotone nondecreasing in the number of its end-vertices that belong to the solution. Therefore, this contradicts the exit conditions from the **while** loop. Therefore, the edge e does not contribute a positive amount to the cost of the resulting solution. □

The linear-time implementation of the algorithm is straightforward by noting that in the second step of the algorithm, to find an appropriate $e = \{u, v\}$ and $i \in \{1, 2\}$ can be done in constant (average) time by the proof of Lemma 15, and by picking in each iteration of the **while** loop the maximum possible value of ϵ that maintains the non-negativity of the resulting cost function. Therefore, we establish the following theorem:

THEOREM 16. *There is a linear time* 2*-approximation algorithm for problem GVC that is based on the local-ratio technique.*

4. *Concluding Remarks*

In this article, we provide a complexity classification of the UGVC problem as a function of the cost parameters α and β. We are not aware of previous results concerning NP-hardness of a problem as a function of parameters of the cost coefficients. We think that such study of the complexity of problems as a function of the parameters of the cost function, is an interesting research topic, and we leave it for future research.

In Hassin and Levin [2003], we considered also the MAXIMIZATION VERSION OF GVC that is defined as follows: the input is a graph $G = (V, E)$, three profit values $0 \leq p_0(i, j) \leq p_1(i, j) \leq p_2(i, j)$ for each edge $(i, j) \in E$, and an upgrade cost $c(v) \geq 0$ for each vertex $v \in V$. $p_k(i, j)$ denotes the profit from the edge (i, j) when exactly k of its endpoints are upgraded. The objective is to maximize the net profit, that is, the total profit minus the upgrading cost. We proved that this problem is NP-hard and claimed that there is a 2-approximation algorithm for this problem. However, the proof of the last result is incorrect and we leave for future research the question of whether such an algorithm exists.

REFERENCES

BAR-YEHUDA, R., BENDEL, K., FREUND, A., AND RAWITZ, D. 2004. Local ratio: A unified framework for approximation algorithms. *ACM Comput. Surv.*, *36*, 422–463.

BAR-YEHUDA, R., AND EVEN, S. 1981. A linear time approximation algorithm for the weighted vertex cover problem. *J. Algor. 2*, 198–203.

BAR-YEHUDA, R., AND RAWITZ, D. 2001. On the equivalence between the primal-dual schema and the local-ratio technique. In *Proceedings of APPROX 2001*. 24–35.

GAREY, M. R., AND JOHNSON, D. S. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company.

HASSIN, R., AND LEVIN, A. 2003. The minimum generalized vertex cover problem. In *Proceedings of the ESA 2003*. 289–300.

HOCHBAUM, D. S. 2002. Solving integer programs over monotone inequalities in three variables: A framework for half integrality and good approximations. *Europ. J. Oper. Res. 140*, 291–321.

KRUMKE, S. O., MARATHE, M. V., NOLTEMEIER, H., RAVI, R., RAVI, S. S., SUNDARAM, R., AND WIRTH, H. C. 1999. Improving minimum cost spanning trees by upgrading nodes. *J. Algor.*, *33*, 92–111.

LAWLER, E. L. 1976. *Combinatorial Optimization: Networks and Matroids*. Holt, Rinehart and Winston.

NEMHAUSER, G. L., AND TROTTER, JR., L. E. 1975. Vertex packing: Structural properties and algorithms. *Math. Prog. 8*, 232–248.

PAIK, D., AND SAHNI, S. 1995. Network upgrading problems. *Networks*. *26*, 45–58.

YANNAKAKIS, M. 1981. Edge deletion problems. *SIAM J. Computing*. *10*, 297–309.

RECEIVED AUGUST 2004; REVISED JUNE 2005; ACCEPTED JUNE 2005