Minimum monopoly in regular and tree graphs

S. Mishra[a,*,1], S.B. Rao[b]

[a] *Department of Mathematics, I. I. T. Guwahati, Guwahati, Assam 781 039, India*
[b] *Theoretical Statistics and Mathematics Unit, Indian Statistical Institute, 203 B. T. Road, Kolkata 700 035, India*

Abstract

In this paper we consider a graph optimization problem called minimum monopoly problem, in which it is required to find a minimum cardinality set $S \subseteq V$, such that, for each $u \in V$, $|N[u] \cap S| \geqslant |N[u]|/2$ in a given graph $G = (V, E)$. We show that this optimization problem does not have a polynomial-time approximation scheme for k-regular graphs ($k \geqslant 5$), unless $\mathsf{P} = \mathsf{NP}$. We show this by establishing two L-reductions (an approximation preserving reduction) from minimum dominating set problem for k-regular graphs to minimum monopoly problem for $2k$-regular graphs and to minimum monopoly problem for $(2k-1)$-regular graphs, where $k \geqslant 3$. We also show that, for tree graphs, a minimum monopoly set can be computed in linear time.

Keywords: Minimum monopoly; APX-complete; Algorithms

1. Introduction

In this paper we consider the following synchronous local majority process in a connected graph $G = (V, E)$ whose vertices are colored either black or white. In this local majority process, each vertex recolors itself with the major color in its neighborhood simultaneously. We shall call this process a synchronous voting process or simply voting process. Obviously, if all the vertices are colored black initially, after a voting process every vertex will get black color. A subset $S \subseteq V$ is called a monopoly of G if every vertex gets black color after one voting process, where in the initial coloring each vertex in S is colored black and remaining vertices white. In this paper we are interested for finding a monopoly with minimum cardinality. We shall denote this optimization problem as *Min-Monopoly*.

Various properties of a monopoly are studied, such as the lower bound on the cardinality of a monopoly in a graph [12,3], what is the maximum influence of a set $S \subseteq V$ on the graph and how small can a monopoly be [15]. These problems have many real life applications such as overcoming failure in distributed computing [12], data redundancy in the area of distributed database management algorithm [5], ensuring mutual exclusion in resource allocation [16] and fault-local mending in distributed network [11]. For a survey about these problems, we refer to [15].

The problem which is closely related to Min-Monopoly is minimum multi-covering problem (Min-MC) [8]. In Min-MC problem, given a ground set E, a set $\mathbb{E}$ of subsets of E and a positive integer b_i for each element of $e_i \in E$,

* Corresponding author.
E-mail addresses: sounak@iitg.ernet.in (S. Mishra), raosb@isical.ac.in (S.B. Rao).
[1] This work was done when this author was a graduate student at I.S.I. Kolkata and was visiting I.M.Sc. Chennai, India.

we are asked to find a subset T of $\mathbb{E}$ with minimum cardinality such that for each element $e_i \in E$ there are at least b_i sets in T containing e_i. Minimum set cover (Min-SC) is a special case of Min-MC with $b_i = 1$ for each $e_i \in E$. Minimum dominating set (Min-Dom-Set) is the problem of finding a dominating set of minimum cardinality in a given graph $G = (V, E)$, where a subset $D \subseteq V$ is called a dominating set of G if for each vertex $u \in V - D$ there is a vertex $v \in D$ such that $(u, v) \in E$. It is known that Min-SC and Min-Dom-Set are equivalent with respect to L-reduction (an approximation preserving reduction) [10]. Since all these problems can be seen as a special case of Min-MC and a greedy algorithm approximates Min-MC within a factor of $(\ln |E| + 1)$ [6], all these problems have such an approximation algorithm. Also it is known that, unless $\mathsf{NP} \subset \mathsf{DTIME}(n^{\log\log n})$, for any $\varepsilon > 0$ there is no polynomial-time algorithm to approximate Min-SC (also Min-Dom-Set) within a factor of $(1 - \varepsilon)\log n$ [7]. Peleg [15] has made the following conjecture: "unless $\mathsf{NP} \subset \mathsf{DTIME}(n^{\log\log n})$, for any $\varepsilon > 0$, Min-Monopoly has no polynomial-time $\log n - \varepsilon$ approximation algorithm". It seems that Peleg's conjecture is true as in [13] the authors have proved a weaker version of this conjecture. They prove that, unless $\mathsf{NP} \subset \mathsf{DTIME}(n^{\log\log n})$, for any$\varepsilon > 0$, Min-Monopoly has no polynomial-time $(\frac{1}{3} - \varepsilon)\log n$ approximation algorithm. While considering Min-Monopoly problem for bounded degree graphs, they also prove that, for every $\varepsilon > 0$, there exists positive constants $\triangle_\varepsilon$ and c such that for any $\triangle \geqslant \triangle_\varepsilon$, any Min-Monopoly instance with degree bounded by $\triangle$ cannot be approximable within a factor of $\ln \triangle - c \ln\ln \triangle$, unless $\mathsf{P} = \mathsf{NP}$. However, they show that for 3-regular graphs Min-Monopoly is APX-complete and has a 1.6154 approximation algorithm.

In the next section, we will describe few definitions which we shall use. In Section 3, we shall prove that Min-Monopoly is APX-complete for k-regular graphs. In other words, for k-regular graphs ($k \geqslant 5$), Min-Monopoly has no polynomial-time approximation scheme, unless $\mathsf{P} = \mathsf{NP}$. Also we show that for regular graphs it is approximable within a factor of 2. In Section 4, we present a linear time algorithm for Min-Monopoly for tree graphs.

2. Definitions

Let $G = (V, E)$ be an undirected graph with $V = \{1, 2, \ldots, n\}$. We define neighborhood $N(v)$ of a vertex $v \in V$ as $N(v) = \{u | u \in V \text{ and } (u, v) \in E\}$ and the closed neighborhood $N[v]$ of a vertex $v \in V$ as the set $N[v] = N(v) \cup \{v\}$. We say that a set $S \subseteq V$ is a monopoly (or satisfies a majority rule) if $|N[v] \cap S| \geqslant |N[v]|/2$, at each vertex $v \in V$, i.e. majority of the vertices of $N[v]$ are in S. In this mathematical formulation of a monopoly set, we assume that if there are equal number of black and white vertices in $N[v]$, for some $v \in V$, then vertex v gets black color after the voting process.

Following [2], we next recall some basic concepts regarding approximation algorithms for NP-optimization problems. The class NPO is the set of all NPO problems and the class PO is the set of all NPO problems that are solvable in polynomial time.

For an instance x of a problem $\pi \in \mathsf{NPO}$, $m^*(x)$ denotes the measure of an optimal solution of x, i.e. $m^*(x) = \text{goal}_{y \in \text{sol}(x)} m(x, y)$ where $\text{sol}(x)$ denotes the finite set of feasible solutions of x and $m(x, y)$ denotes the nonnegative measure of the solution y of the instance x of π. Given an instance x of a $\pi \in \mathsf{NPO}$ and $y \in \text{sol}(x)$, the performance ratio of y with respect to x is defined by

$$R_\pi(x, y) = \max\left\{\frac{m(x, y)}{m^*(x)}, \frac{m^*(x)}{m(x, y)}\right\}.$$

A polynomial-time algorithm A for a problem $\pi \in \mathsf{NPO}$ is an ε-*approximate* algorithm for π, if $R_\pi(x, A(x)) \leqslant \varepsilon$, for some $\varepsilon \geqslant 1$ and for any instance x of π, where $A(x)$ is the solution for x given by A. The class APX is the set of all $\pi \in$ NPO which have ε-approximate algorithm for some constant $\varepsilon \geqslant 1$.

Clearly, $\mathsf{PO} \subseteq \mathsf{APX} \subseteq \mathsf{NPO}$ and it is known that these inclusions are strict if and only if $\mathsf{P} \neq \mathsf{NP}$ [2,4]. In order to introduce the notion of completeness for the class APX, several approximation preserving reductions are introduced. Among them is L-reduction [14] which is most commonly used and is as follows:

π_1 is said to be *L-reducible* to π_2 [14], in symbols $\pi_1 \leqslant_L \pi_2$, if there exists a function f from instances of π_1 to instances of π_2 and two positive constants α, β such that:

1. $m^*_{\pi_2}(f(x)) \leqslant \alpha \cdot m^*_{\pi_1}(x)$.
2. For any $x \in I_{\pi_1}$ and for any $y \in \text{sol}_{\pi_2}(f(x))$ we can in polynomial time find a solution $y' \in \text{sol}_{\pi_1}(x)$ such that

$$|m^*_{\pi_1}(x) - m_{\pi_1}(x, y')| \leqslant \beta \cdot |m^*_{\pi_2}(f(x)) - m_{\pi_2}(f(x), y)|.$$

A problem $\pi \in$ NPO is APX-hard if, for any $\pi' \in$ APX, $\pi' \leqslant_L \pi$, and problem π is APX-complete if π is APX-hard and $\pi \in$ APX.

3. APX-hardness

In this section, we shall denote Min-Monopoly (Min-Dom-Set) problem for k-regular graphs by Min-Monopoly-k (Min-Dom-Set-k). We will show that Min-Monopoly problem is APX-complete for k-regular graphs, for $k \geqslant 5$. But before that we will prove that Min-Dom-Set-k is APX-complete for all k-regular graphs, $k \geqslant 3$.

First, we would like to note that Min-Dom-Set for graphs with maximum degree bounded above by a constant has a constant factor approximation algorithm and is known to be APX-complete [14]. Alimonti and Kann [1] have proved that Min-Dom-Set-3 is APX-complete. From their proof it also follows that Min-Dom-Set is APX-complete for graphs in which degree of a vertex is either 2 or 6 (see the proof of Theorem 3.3 in [1]). We shall denote this problem as Min-Dom-Set-(2, 6). Next we will show that Min-Dom-Set-4 is APX-complete.

Lemma 1. *Min-Dom-Set-4 is APX-complete.*

Proof. To show that Min-Dom-Set-4 is APX-hard, we will establish the following two reductions:

$$\text{Min-Dom-Set-}(2,6) \leqslant_L \text{Min-Dom-Set-}(2,4) \leqslant_L \text{Min-Dom-Set-4}.$$

For the first reduction, given a graph $G=(V,E)$ in which degree of a vertex is either 2 or 6 construct a graph $G'=(V',E')$ in which degree of a vertex either 2 or 4 in the following manner. To each vertex $v \in V$ of degree 6, split v into two vertices v_1 and v_2 of degree 3 each. Then add 5 new vertices v_3, v_4, v_5, v_6, v_7, and edges (v_1, v_3), (v_3, v_4), (v_4, v_5), (v_5, v_6), (v_6, v_7), (v_7, v_2), see Fig. 1.

Any dominating set D' of G' can be transformed back to a dominating set D of G as follows. D is the set consisting of set $D' \cap V$ and for each vertex $v \in V$ of degree 6: $v \in D$ if $|\{v_1, \ldots, v_7\} \cap D'| \geqslant 3$ and $v \notin D$ if $|\{v_1, \ldots, v_7\} \cap D'| = 2$. D is a dominating set of G because, if $|D' \cap \{v_1, \ldots, v_7\}| \geqslant 3$ then, without loss of generality, we can assume that $D' \cap \{v_1, \ldots, v_7\} = \{v_1, v_5, v_2\}$; and if $|D' \cap \{v_1, \ldots, v_7\}| = 2$ then there can be three cases (i) $D' \cap \{v_1, \ldots, v_7\} = \{v_3, v_6\}$; (ii) $D' \cap \{v_1, \ldots, v_7\} = \{v_4, v_6\}$ and (iii) $D' \cap \{v_1, \ldots, v_7\} = \{v_4, v_7\}$) and in all these cases D' must contain at least one vertex from $D' - \{v_1, \ldots, v_7\}$ to dominate either v_1 or v_2. Also, it is not difficult to see that $|D| \leqslant |D'| - 2s$, where s is the number of vertices of degree 6 in G.

Let D be a dominating set of G. Let D_2 be the set of vertices of degree 2 in D. From D, construct the set $D' = D_2 \cup \left[\bigcup_{v \in D \text{ and } d(v)=6} \{v_1, v_2, v_5\}\right] \cup \left[\bigcup_{v \in V-D \text{ and } d(v)=6} \{v_4, v_6\}\right]$. Since D is a dominating set of G, it is easy to see that D' is a dominating set of G'. Also $|D'| = |D| + 2s$. From these, it can be shown that if D_o is a minimum dominating set of G then D'_o defined as above is also a minimum dominating set of G'. Since G has bounded degree 6, $|D| \geqslant |V|/7$. Therefore $|D'_o| \leqslant |D_o| + 2|V| \leqslant 15|D_o|$. This establish the first L-reduction with $\alpha = 15$ and $\beta = 1$.

For the second reduction, given a graph $G=(V,E)$ in which degree of a vertex is either 2 or 4 construct a 4-regular graph $G'=(V',E')$ as follows. For each vertex $v \in V$ of degree 2 construct the graph $H_5(v)$ which has $\{v_1, \ldots, v_5\}$ as vertex set and has all the edges except the edge (v_1, v_5). Then connect $H_5(v)$ to G with the edges (v, v_1) and (v, v_5). Clearly, G' is 4-regular.

Let D' be a dominating set of G'. For any vertex $v \in V$ of degree 2, D' must contain at least one vertex from $H_5(v)$ irrespective of whether the vertex v is in D' or not. Hence, without loss of generality, we will assume that any dominating set D' of G' contains only one vertex, say v_2, from $H_5(v)$, for each degree 2 vertex $v \in V$.

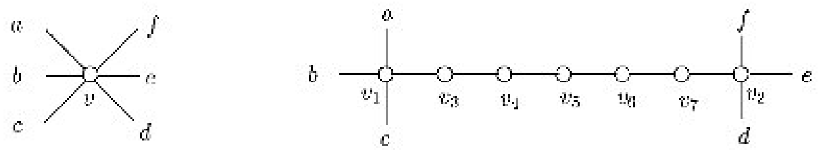

Fig. 1. The transformation of a degree 6 vertex in the reduction Min-Dom-Set-(2, 6) $\leqslant_L$ Min-Dom-Set-(2, 4).

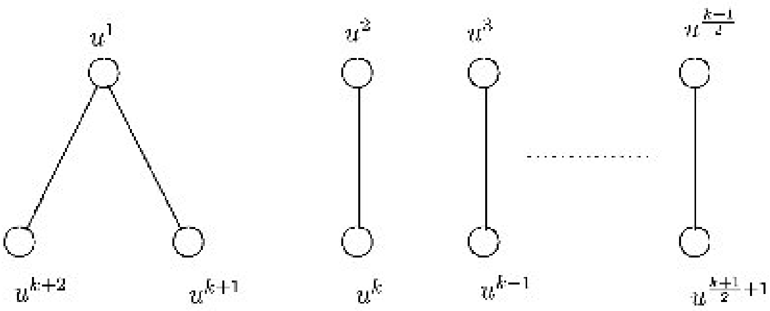

Fig. 2. The graph $H_k^c(u)$.

Now it is easy to see that if D' is a dominating set of G' then $D = D' \cap V$ is a dominating set of G and conversely, if D is a dominating set of G then the set $D' = D \cup \{v_2 | v \in V \text{ and } d(v) = 2\}$ is a dominating set of G'. Also $|D'| = |D| + t$, where t is the number of vertices of degree 2 in G. Since G is of bounded degree 4, this is an L-reduction with $\alpha = 5$ and $\beta = 1$. □

By using Lemma 1, it is easy to prove the following theorem.

Theorem 2. *Min-Dom-Set-k is* APX*-complete, for* $k \geqslant 3$.

Proof. Since Min-Dom-Set is APX-complete for 3 and 4-regular graphs, it is sufficient to show that Min-Dom-Set-k $\leqslant_L$ Min-Dom-Set-$(k+2)$, for $k \geqslant 3$.

Given a k-regular graph $G = (V, E)$ construct a $k+2$-regular graph as follows. For each vertex $v \in V$, let $H_k(v)$ be the graph obtained by removing an edge from the complete graph $K_{(k+3)}$. Let the vertex set of $H_k(v)$ be $\{v_1, \ldots, v_{k+3}\}$ and let edge set of $H_k(v)$ contains all the edges except the edge (v_1, v_{k+3}). For each vertex $v \in V$ of degree k, connect $H_k(v)$ with G by the pair of edges (v, v_1) and (v, v_{k+3}). Clearly, G' is $(k+2)$-regular.

Now any dominating set D' of G' contains at least one vertex from $H_k(v)$ irrespective of whether $v \in D'$ or not. Hence, without loss of generality, we can assume that, for every vertex $v \in V$ of degree k, D' contains the vertex v_2 from $H_k(v)$. From this it is clear that if D' is a dominating set of G' then $D = D' \cap V$ is a dominating set of G and conversely, if D is a dominating set of G then $D' = D \cup \{v_2 | v \in V\}$ is a dominating set of G'. Also $|D'| = |D| + n$. Since G is k-regular, it follows that it is an L-reduction with $\alpha = k+2$ and $\beta = 1$. □

Next we shall show that Min-Monopoly-k is APX-complete, for $k \geqslant 5$. For this we shall establish two L-reductions from Min-Dom-Set, one is for even regular graphs and other is for odd regular graphs. First, we prove two lemmas which will be used in these two reductions.

Lemma 3. *Let* $k \geqslant 5$ *be an odd integer and let* $H_k(u)$ *be the graph consisting of* $(k+2)$ *vertices with exactly one vertex of degree* $(k-1)$ *and all other* $(k+1)$ *vertices of degree k each. Then, any monopoly set S of* $H_k(u)$ *must contain at least* $((k+1)/2+1)$ *vertices. Moreover*, $H_k(u)$ *has a monopoly set of size* $((k+1)/2+1)$ *and it contains the vertex of degree* $(k-1)$.

Proof. Clearly, the complement $H_k^c(u)$ of $H_k(u)$ has a vertex of degree 2 and all other vertices has degree 1 each. Also it has $(k+1)/2$ connected components. Out of these one is a path of length 2 and others have two vertices each. We assume that the vertices are numbered as shown in Fig 2. Assume that in $H_k(u)$, the vertex u^1 has degree $k-1$.

It can be seen easily that the vertex set $S_u = \{u^1, u^2, \ldots u^{(k+1)/2+1}\}$ is a monopoly set of $H_k(u)$, because every connected component of $H_k^c(u)$ has at least one vertex from S and $|S| = (k+1)/2+1$.

Now, we show that any vertex set S of $H_k(u)$ with $|S| = (k+1)/2$, cannot be a monopoly set of $H_k(u)$. We prove this by considering two cases, one is for $k = 5$ and other is for $k > 5$.

Let $k = 5$. Then, $H_5^c(u)$ has three components and $|S| = 3$. There can be two cases again; either all the three vertices in S are from the larger connected component of $H_5^c(u)$ (note this component has three vertices) or at least one vertex

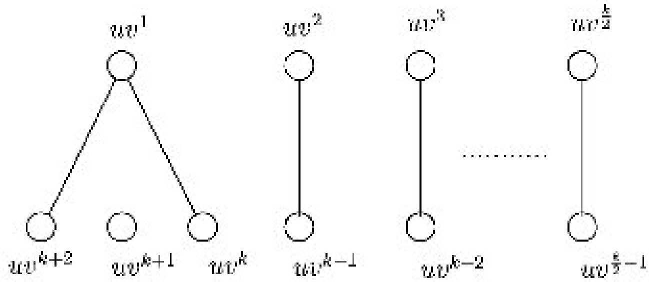

Fig. 3. The graph $F_k^c(u, v)$.

from any one of the other two components of $H_5^c(u)$. Suppose S is consisting of all the three vertices from the larger component of $H_5^c(u)$. Note that it is a path of length 2 and let its end vertices be w and x. Then, in $H_5(u)$, $|N[x] \cap S| = 2$ as u^1 and x are adjacent in $H_5^c(u)$. But we need at least 3 vertices from S in the closed neighborhood of x. Hence, such an S cannot be a monopoly set. In the other case, let the edge (w, x) be a component of $H_5^c(u)$ and $w \in S$. Then, in $H_5(u)$, $|N[x] \cap S| \leqslant 2$; implying that S cannot be a monopoly set of $H_5(u)$.

Let $k > 5$ be any odd integer and S be a subset of the vertex set in $H_k(u)$ with $|S| = (k+1)/2$. Obviously, $|S| > 3$ and S contains at least one vertex from a component consisting of two vertices in $H_k^c(u)$. Let w, x be the vertices in one such component and $w \in S$. Then in $H_k(u)$, $|N[x] \cap S| \leqslant (k+1)/2 - 1$ as w and x are adjacent in $H_k^c(u)$ and $|S| = (k+1)/2$. Hence, such an S cannot be a monopoly of $H_k(u)$. □

Lemma 4. *Let $k \geqslant 6$ be an even integer and let $F_k(u, v)$ be the graph consisting of $(k+2)$ vertices and having exactly one vertex of degree $(k-2)$ and all other vertices are of degree k each. Then, any monopoly set S of $F_k(u, v)$ must contain at least $\lceil (k+1)/2 \rceil + 1$ vertices. Moreover, $F_k(u, v)$ has a monopoly set of size $\lceil (k+1)/2 \rceil + 1$ and it contains the vertex of degree $(k-2)$.*

Proof. Clearly, $F_k^c(u, v)$ has exactly one vertex of degree 3 and all other $(k+1)$ vertices of degree 1. Also it has $k/2$ connected components. Clearly, one component has 4 vertices and 3 edges; and others have 2 vertices each. We assume that the vertices of $F_k^c(u, v)$ are numbered as shown in Fig. 3. Here note that, in $F_k(u, v)$, that the vertex uv^1 has degree $k-2$ and all other vertices have degree k.

It can be seen easily that the vertex set $S_{uv} = \{uv^1, uv^2, \ldots, uv^{k/2+2}\}$ is a monopoly set containing the vertex of degree $k-2$.

By similar arguments as given in the proof Lemma 3, it can be proved that no vertex set S of size $\lceil (k+1)/2 \rceil$ can be a monopoly set of $F_k(u, v)$. □

Now we shall establish two L-reductions Min-Dom-Set-$k \leqslant_L$ Min-Monopoly-$(2k-1)$ and Min-Dom-Set-$k \leqslant_L$ Min-Monopoly-$2k$, for $k \geqslant 3$. From this it will follow that Min-Monopoly-k set is APX-hard, for $k \geqslant 5$. In the remaining part of this section, without loss of generality, we will assume that any instance of Min-Dom-Set-k has even number of vertices.

Theorem 5. *Min-Dom-Set-$k \leqslant_L$ Min-Monopoly-$(2k-1)$, for $k \geqslant 3$.*

Proof. Let $p = 2k - 1$. Let $G = (V, E)$ be a k-regular graph (an instance of Min-Dom-Set-k). From G we construct an instance $G' = (V', E')$ of Min-Monopoly-p, which is a p-regular graph, as follows: for each vertex $u \in V$, make $(k-1)$ many copies of the graph $H_p(u)$ as described in the Lemma 3, say $H_p(u_1), \ldots, H_p(u_{k-1})$, and join them to the graph G with the $k-1$ edges $(u, u_1^1), (u, u_2^1), \ldots, (u, u_{k-1}^1)$. Clearly, G' is a $(2k-1)$-regular graph. Since G' is $(2k-1)$-regular graph, a monopoly set of G' must contain at least k vertices from each closed neighborhood.

To a dominating set S of G, we associate a set $S' \subseteq V'$, defined by $S' = S \cup [\bigcup_{u \in V} [S_{u_1} \cup S_{u_2} \cdots \cup S_{u_{k-1}}]]$. S' is a minimal monopoly set of G' because, for each vertex $u \in V$, S_{u_i} is a monopoly set of $H_p(u_i)$, for $1 \leqslant i \leqslant k-1$ (by Lemma 3); and for a vertex $u \in V$, $|N_{G'}[u] \cap S'| = |\{u_1^1, u_2^1, \ldots, u_{k-1}^1\}| + |N_G[u] \cap S| \geqslant k - 1 + 1 = k$ (as S is a dominating set of G). Also $|S'| = (k-1)(k+3)/2|V| + |S|$. Because of Lemma 3, we assume that a monopoly set

S' of G' is of the form as defined above. For any minimal monopoly set S' of G', $S = S' \cap V$ is a dominating set of G, and $|S'| = (k-1)(k+3)/2|V| + |S|$; because, for each $u \in V$, $N_{G'}[u]$ contains at least k vertices from S' and it contains exactly $k-1$ vertices from $V' - V$. From this it follows, that S_o is a minimum dominating set of G iff corresponding set S'_o is a minimum monopoly function of G'. Since G is k-regular, we have $|S_o| \geqslant |V|/(k+1)$ and hence $|S'_o| \leqslant (k-1)(k+3)/2(k+1)|S_o| + |S_o| = ((k-1)(k+1)(k+3)/2+1)|S_o|$. Also for any monopoly set S' of G', $|S'| - |S'_o| = |S| - |S_o|$. Hence, this is an L-reduction with $\alpha = (k-1)(k+1)(k+3)/2 + 1$ and $\beta = 1$. □

Theorem 6. *Min-Dom-Set-k* $\leqslant_L$ *Min-Monopoly-2k, for* $k \geqslant 3$.

Proof. Let $G = (V, E)$ be an instance of Min-Dom-Set-k. Since $|V|$ is even, we can express V as $V = \bigcup_{i=1}^{t} P_i$, where P_is are pairwise disjoint and each has cardinality 2. From G we construct an instance $G' = (V', E')$ of Min-Monopoly-$2k$ as follows: for each set $P_i = \{u, v\}$, construct k copies of the graph $F_{2k}(u, v)$ (as defined in Lemma 4), say $F_{2k}(u_1, v_1), \ldots, F_{2k}(u_k, v_k)$, and connect them to G by $2k$ edges $(u, uv_1^1), (v, uv_1^1), \ldots, (u, uv_k^1), (v, uv_k^1)$. This completes the construction of G'.

To a dominating set S of G we associate a set $S' = S \cup \left[\bigcup_{i=1}^{t} [S_{u_1v_1} \cup S_{u_2v_2} \cdots \cup S_{u_kv_k}]\right]$. Note that $|S'| = k(k+4)/4|V| + |S|$. By similar arguments as given in the proof of Theorem 5, we can show that S_o is a minimum dominating set of G iff the associated set S'_o is a minimum monopoly set of G'. Since G is k-regular, we have $|S'_o| \leqslant (k(k+1)(k+4)/4+1)|S_o|$. Also for any monopoly set S' of G' we have $|S'| - |S'_o| = |S| - |S_o|$. This completes the L reduction with $\alpha = k(k+1)(k+4)/4+1$ and $\beta = 1$. □

Next we will show that for, k-regular graphs, Min-Monopoly problem is approximable within a factor of 2. First, we shall prove an inequality for arbitrary graphs.

Let $G = (V, E)$ be a graph with minimum degree δ and maximum degree $\triangle$. Let S be a monopoly of G. Then $\sum_{v \in S} |N[v] \cap (V - S)| = \sum_{v \in (V-S)} |N[v] \cap S|$ as this the number of edges in G between the vertex sets S and $V - S$. Since S is a monopoly set of G we have $\sum_{v \in S} (d(v)+1)/2 \geqslant \sum_{v \in (V-S)} (d(v)+1)/2$. From this inequality, it can be seen easily that $|S| \geqslant |V|(\delta+1)/(\delta+\triangle+2)$.

In particular, if G is a k-regular graph and S_o is an optimal monopoly then $|S_o| \geqslant |V|/2$. From this it follows that for any monopoly S of G, $|S|/|S_o| \leqslant 2$. Hence we have proved the following theorem.

Theorem 7. *For* $k \geqslant 5$, *Min-Monopoly-k is approximable within a factor of* 2 *and is* APX*-complete.*

4. Linear algorithm for trees

In this section we shall present a linear time algorithm for Min-monopoly for tree graphs. First, we shall give some definitions. Given a graph $G = (V, E)$, a function $f : V \to \{-1, 1\}$ is called a monopoly function of G if, for all $v \in V$, $f(N[v]) \geqslant 0$. It can be observed that f is a monopoly function of G if $S_f = \{v | f(v) = 1\}$ is a monopoly set of G, conversely if S is a monopoly set of G then

$$f(v) = \begin{cases} -1 & \text{if } v \notin S, \\ 1 & \text{if } v \in S \end{cases}$$

is a monopoly function of G. A monopoly function f of G is minimal if and only if for every vertex $v \in V$ with $f(v) = 1$ there exists a vertex $u \in N[v]$ with $f(N[u]) \in \{0, 1\}$. We define cost of a monopoly function f as $f(V) = \sum_{v \in V} f(v)$. Here, we are interested in finding a monopoly f for G of minimum cost. Though this objective function is different from the objective function in Min-Monopoly, an optimal solution of Min-Monopoly for a graph G defines uniquely a minimum monopoly function of G and conversely.

Next we are going to present a linear time algorithm for finding a minimum monopoly function in a tree graph T. The algorithm roots the tree and associates a few variables with the vertices of T. In a rooted tree, $\mathrm{Ch}(v)$ and $\mathrm{Ch}[v]$ denote the set of children of v and $\mathrm{Ch}(v) \cup \{v\}$, respectively. For any vertex $v \in V$, the variable $\mathrm{MinSum}(v)$ denotes the minimum possible sum of the values that may be assigned to v and its children; $\mathrm{ChildSum}(v)$ is the sum of values assigned to the children of v and $\mathrm{Sum}(v) = \mathrm{ChildSum}(v) + f(v)$.

Monopoly-Tree

Without loss of generality we can assume that, the vertices of tree are numbered from 1 to n so that $u > v$ if the level of the vertex u is less than the level of vertex v. Note that root is the nth vertex in T.

for $i = 1$ **to** $(n-1)$ **do**
  if i is a leaf node **then** $f(i) = -1$ and $Sum(i) = -1$;
  else
    compute $ChildSum(i)$;
    if ($ChildSum(i) > 0$ and $Sum(j) > 0$, for all $j \in Ch(i)$)
    then $f(i) = -1$; compute $MinSum(i)$; $Sum(i) = MinSum(i)$;
    else
      $f(i) = 1$; compute $MinSum(i)$; $Sum(i) = MinSum(i)$;
      while ($Sum(i) < -1$) **do**
        choose a vertex $j \in Ch(i)$ with $f(j) = -1$ and set $f(j) = 1$;
        recompute $Sum(i)$;
if ($ChildSum(n) > 0$ and $Sum(j) > 0$, for all $j \in Ch(n)$)
  then $f(n) = -1$; compute $MinSum(n)$; $Sum(n) = MinSum(n)$;
  else
    $f(n) = 1$; compute $MinSum(n)$; $Sum(n) = MinSum(n)$;
    while($Sum(n) < 0$) **do**
      choose a vertex $j \in Ch(n)$ with $f(j) = -1$ and set $f(j) = 1$;
      recompute $Sum(n)$;

Now we shall verify the correctness of the above algorithm. But before that we shall prove few lemmas which will be used for proving the following theorem. The method used for proving the following theorem is borrowed from [9].

Theorem 8. *Algorithm Monopoly-Tree produces a minimum cost monopoly function for a tree graph in linear time.*

Lemma 9. *When above algorithm assigns a value to the root r' of a subtree (or tree) T', the following three conditions will hold*:

(i) *For any vertex $v \in T' - \{r'\}$, $f(N[v]) \geqslant 0$.*
(ii) *$Sum(r') \geqslant MinSum(r')$.*
(iii) *The initial value assigned to r' is the minimum value it can receive given the values of its descendants under f.*

Proof. Proof is by induction on the order in which the vertices are numbered. First, a leaf node i is processed and $f(i) = -1$. The first condition is satisfied trivially (as i has no child). At this state $\mathrm{Sum}(r') = \mathrm{MinSum}(r') = -1$. Hence second and third conditions are also satisfied. This completes the base case for the induction proof.

Next we assume that the algorithm assigns values to the first k vertices so that these three conditions holds at these k vertices. Now it requires to show that these conditions hold after the $(k+1)$st vertex is assigned a value.

Let w be the $(k+1)$th vertex. By induction hypothesis we can assume that all its descendants, other than its children, have closed neighborhood sum at least zero. These vertices will have closed neighborhood sums at least zero even after assigning values to the vertex w. This is because, the algorithm may increase the values to the children while working for the vertex w (in any one of the while loops in the algorithm). If $f(w) = -1$, then $\mathrm{Sum}(j) > 0$, for all $j \in \mathrm{Ch}(w)$, and hence $f(N[j]) = \mathrm{Sum}(j) + f(w) \geqslant 0$, for all $j \in \mathrm{Ch}(w)$. For other cases, $f(w) = 1$; and since $\mathrm{Sum}(j) \geqslant -1$, for all $j \in \mathrm{Ch}(w)$, it follows that $f(N[j]) = \mathrm{Sum}(j) + f(w) \geqslant 0$, for all $j \in \mathrm{Ch}(w)$. Thus, $(k+1)$th vertex satisfies the condition 1.

It is easy to see proof of condition 2. It remains to consider condition 3. Let $v = r'$. Initially, if the algorithm sets $f(v) = -1$ then there is nothing to prove as -1 is the minimum possible value that can be assigned to v. Suppose initial value assigned to v be 1. Then either $\mathrm{ChildSum}(v) \leqslant 0$ or there exists a vertex $j \in \mathrm{Ch}(v)$ such that $\mathrm{Sum}(j) \leqslant 0$. Then the values assigned to its children are increased until $\mathrm{Sum}(v) \geqslant 0$ or -1 if v is root or not, respectively. If v is root of T, then $\mathrm{ChildSum}(v) = 0$ or -1 if degree of v is even or odd, respectively. If v is not root of T then $\mathrm{ChildSum}(v) = -1$ or

0 if degree of v is even or odd, respectively. It follows that f to be a monopoly function $f(v)$ must be 1. This completes the proof of lemma. □

From the proof of Lemma 9, the following corollary follows:

Corollary 10. *The function f produced by the algorithm Monopoly-Tree is a monopoly function for T.*

In order to show that the monopoly function f obtained by the algorithm Monopoly-Tree is of minimum cost, let g be any minimum cost monopoly function for the same rooted tree T. If $f \neq g$, then we will show that g can be transformed to a new minimum monopoly function g', for the same rooted tree T, that differs from f in fewer values than g did. This process will continue until $f = g$. Suppose $f \neq g$. Let v be the lowest indexed vertex for which $f(v) \neq g(v)$. Then all descendants of v are assigned the same value under g as under f. An immediate corollary to Lemma 9 is the following corollary.

Corollary 11. *If $g(v) < f(v)$, then the initial value assigned to the vertex v was increased in any one of two while loops in the algorithm.*

Lemma 12. *Let $g(v) < f(v)$, then the function g' defined as follows is a minimum monopoly function for T that differs from f in lesser values than that of g. If either parent(v) is the root of T or d(parent(v)) is even and parent(v) is not root of T then*

$$g'(u) = \begin{cases} f(u) & \text{if } u \in N[parent(v)], \\ g(u) & \text{if } u \notin N[parent(v)]. \end{cases}$$

If parent(v) is not root of T and d(parent(v)) is odd then

$$g'(u) = \begin{cases} f(u) & \text{if } u \in Ch[parent(v)], \\ g(u) & \text{if } u \notin Ch[parent(v)]. \end{cases}$$

Proof. Let $w = parent(v)$. By Corollary 11, initial value assigned to the vertex v was increased in any one of the while loops of the algorithm while the parent of v was being assigned a value. Hence $f(w) = 1$, where $w = parent(v)$.

Let w be the root of T. Then $f(N[w]) = 1$ or 0 depending on whether $d(w)$ is even or odd, respectively. If $d(w)$ is even then $g(N[w]) \geqslant 1 = f(N[w])$; and if $d(w)$ is odd then $g(N[w]) \geqslant 0 = f(N[w])$. Hence $g(N[w]) \geqslant f(N[w])$. Further, all vertices in $V - N[w]$ have same values under g as under f. Hence $g' = f$. Now $f(V) \geqslant g(V) = g(V - N[w]) + g(N[w]) = f(V - N[w]) + g(N[w]) \geqslant f(V - N[w]) + f(N[w]) = f(V)$. Hence $g(V) = f(V)$; so $g' = f$ is a minimum monopoly function of T.

Let w be not root of T. Let $d(w)$ be even. Then $f(\mathrm{Ch}[w]) = 0$ and since f is a monopoly function $f(p(w)) = 1$. Also as g is a monopoly function $g(N[w]) \geqslant 1 = f(N[w])$. In this case

$$g'(u) = \begin{cases} f(u) & \text{if } u \in N[w], \\ g(u) & \text{if } u \notin N[w]. \end{cases}$$

Since all the descendants of w, other than its children, have same values under f as under g, $g'(N[u]) = f(N[w])$, for $u = w$ or u is a descendant of w. Hence g' is a monopoly function of T. Now $g'(V) = g(V - N[w]) + f(N[w]) \leqslant g(V - N[w]) + g(N[w]) = g(V)$. Hence $g'(V) = g(V)$. Thus, g' is a minimum monopoly of T and differs from f in fewer vertices than g.

Let $d(w)$ is odd. Then $f(\mathrm{Ch}[w]) = 1$, and

$$g'(u) = \begin{cases} f(u) & \text{if } u \in \mathrm{Ch}[w], \\ g(u) & \text{if } u \notin \mathrm{Ch}[w]. \end{cases}$$

In this case since $f(\mathrm{Ch}[w]) = 1$, $g'(N[w]) \geqslant 0$. Also Since all descendants of w, other than its children, have same values under f as under g, g' is a monopoly of T. By Lemma 9(iii), $f(\mathrm{Ch}[w]) \leqslant g(\mathrm{Ch}[w])$. Thus, $g'(V) = g(V - \mathrm{Ch}[w]) + f(\mathrm{Ch}[w]) \leqslant g(V - \mathrm{Ch}[w]) + g(\mathrm{Ch}[w]) = g(V)$. Hence g' is a minimum monopoly function of T. □

It remains to consider the case where $f(v) < g(v)$. Let v be the least indexed vertex with $f(v) < g(v)$. Here, v cannot be root of T as $f(v) < g(v)$ would imply $f(V) < g(V)$ which is impossible. If any vertex x at the same level as v has and $g(x) < f(x)$, then we can apply Lemma 12 to a minimum monopoly function g' that agrees with f in more values than under g. So, now onwards, we may assume that every vertex x at the same level as v has $f(x) \leqslant g(x)$.

Since $f(v) \leqslant g(v)$, it follows that $f(v) = -1$ and $g(v) = 1$. Let $w = \text{parent}(v)$ and $u = \text{parent}(w)$. If $f(w) \leqslant g(w)$ and $f(u) \leqslant g(u)$, then, for all $x \in N[v]$, $f(N[x]) = f(N[x] - v) + f(v) \leqslant g(N[x] - v) + g(v) - 2 = g(V) - 2$. By minimality of g, there exists a vertex $x \in N[v]$ such that $g(N[x]) \in \{0, 1\}$. In particular, for this x, $f(N[x]) \leqslant g(N[x]) - 2 < 0$. This contradicts that f is a monopoly function. Hence either $f(w) > g(w)$ or $f(u) > g(u)$.

Let $f(w) > g(w)$. In this case, let

$$g'(y) = \begin{cases} g(y) & \text{if } y \in V - \{v, w\}, \\ f(y) & \text{if } y \in \{v, w\}. \end{cases}$$

In this case the only vertex whose closed neighborhood sum is decreased is children of v. But, for any $t \in N[v]$, $g'(N[t]) = g(N[t] - v) + f(v) \geqslant f(N[t] - v) + f(v) = f(N[t])$. Hence g' is a monopoly function as f is. Further, $g'(V) = g(V)$, so that g' is a minimum monopoly of T.

For the other case, let us assume $f(w) \leqslant g(w)$. From this it follows that $f(u) > g(u)$. Define

$$g'(y) = \begin{cases} g(y) & \text{if } y \in V - \{u, v\}, \\ f(y) & \text{if } y \in \{u, v\}. \end{cases}$$

By similar argument we can show that g' is a minimum monopoly of T. This completes the proof of Theorem 8. □

References

[1] P. Alimonti, V. Kann, Hardness of approximating problems on cubic graphs, in: Proceedings of the Third Italian Conference on Algorithms and Complexity, Lecture Notes in Computer Science, vol. 1203, Springer, Berlin, 1997, pp. 288–298.

[2] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, M. Protasi, Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties, Springer, Berlin, Heidelberg, 1999.

[3] J.-C. Bermond, D. Peleg, The power of small coalitions in graphs, in: Proceedings of the Second Colloquium on Structural Information and Communication Complexity, Olympia, Greece, June 1995, Carleton University Press, pp. 173–184.

[4] P. Crescenzi, A. Panconesi, Completeness in approximation classes, Inform. Computation 93 (1991) 241–262.

[5] S.B. Davudson, H. Garcia-Molina, D. Skeen, Consistency in portioned networks, ACM Comput. Surveys 17 (1985) 341–370.

[6] G. Dobson, Worst case analysis of greedy heuristics for integer programming with nonnegative data, Math. Oper. Res. 7 (1982) 515–531.

[7] U. Feige, A threshold of ln n for approximating set cover, in: Proceedings of the ACM Symposium on the Theory of Computing, 1996.

[8] N.G. Hall, D.S. Hochbaum, A fast approximation algorithm for the multicovering problem, Discrete Appl. Math. 15 (1986) 35–40.

[9] J.H. Hattingh, M.A. Henning, P.J. Slater, The algorithmic complexity of signed domination in graphs, Australasian J. Combin. 12 (1995) 101–112.

[10] V. Kann, On the approximability of NP-complete optimization problems, Ph.D. thesis, Department of Numerical Analysis and Computing Science, Royal Institute of Technology, Stockholm, 1992.

[11] S. Kutten, D. Peleg, Fault-local distributed mending, in: Proceedings of the 36th IEEE Symposium on Foundations of Computer Science, 1995.

[12] N. Linial, D. Peleg, Y. Rabinovich, M. Saks, Sphere packing and local majorities in graphs, in: Second ISTCS, IEEE Computer Society Press, Silver Spring, MD, 1993, pp. 141–149.

[13] S. Mishra, J. Radhakrishnan, S. Sivasubramanian, On the hardness of approximating minimum monopoly problems, FSTTCS-2002, Lecture Notes in Computer Science, vol. 2556, 2002, pp. 277–288.

[14] C.H. Papadimitriou, M. Yannakakis, Optimization, approximation, and complexity classes, J. Comput. System Sci. 43 (1991) 425–440.

[15] D. Peleg, Local majority voting, small coalition and controlling monopolies in graphs: a review, in: Proceedings of Third Colloquium on Structural Information and Communication Complexity, 1996, pp. 152–169.

[16] M. Raynal, Algorithms for Mutual Exclusion, MIT Press, Cambridge, MA, 1986.