International Journal of Foundations of Computer Science
Vol. 14 No. 2(2003) 323–333

A SIMPLE HEURISTIC FOR MINIMUM CONNECTED DOMINATING SET IN GRAPHS

PENG-JUN WAN
Department of Computer Science
Illinois Institute of Technology
Chicago, Illinois 60616, USA
Email: wan@cs.iit.edu

KHALED M. ALZOUBI
Department of Computer Science
Illinois Institute of Technology
Chicago, Illinois 60616, USA
Email: alzokha@iit.edu

and

OPHIR FRIEDER
Department of Computer Science
Illinois Institute of Technology
Chicago, Illinois 60616, USA
Email: ophir@cs.iit.edu

Received 20 August 2002
Accepted 20 November 2002
Communicated by S. Olariu

ABSTRACT

Let $\alpha_2(G), \gamma(G)$ and $\gamma_c(G)$ be the 2-independence number, the domination number, and the connected domination number of a graph G respectively. Then $\alpha_2(G) \leq \gamma(G) \leq \gamma_c(G)$. In this paper, we present a simple heuristic for Minimum Connected Dominating Set in graphs. When running on a graph G excluding K_m (the complete graph of order m) as a minor, the heuristic produces a connected dominating set of cardinality at most $7\alpha_2(G) - 4$ if $m = 3$, or at most $\left(\frac{m(m-1)}{2} + 5\right)\alpha_2(G) - 5$ if $m \geq 4$. In particular, if running on a planar graph G, the heuristic outputs a connected dominating set of cardinality at most $15\alpha_2(G) - 5$.

Keywords: connected dominating set, 2-independent set, minor, approximation algorithm

1. Introduction

A *dominating set* (DS) of a graph $G = (V, E)$ is a subset $U \subset V$ such that each node not in U has a neighbor in U, and a *connected dominating set* (CDS) is a dominating set which also induces a connected subgraph. The Minimum Dominating Set

(MDS) problem, and the Minimum Connected Dominating Set (MCDS) problem seek, for a given graph G, a least-cardinality DS and a least-cardinality CDS, respectively. MCDS in general graphs is known to be NP-hard [8]. In addition, Guha and Khuller [9] gave an approximation preserving reduction from the set-cover problem [7] to MCDS, which implied that for any fixed $0 < \epsilon < 1$, no polynomial-time algorithm can find a CDS in general graphs within $(1-\epsilon) H(\Delta)$ times the minimum unless $NP \subset DTIME\left[n^{O(\log\log n)}\right]$ [10], where Δ is the maximum degree and H is the harmonic function. They also presented two greedy heuristics for MCDS with approximation ratios of $2H(\Delta)+2$ and $ln\Delta + 3$ respectively.

Although MCDS in general graphs is hard to approximate, the restriction to certain special graph classes admits much better approximation results. MCDS in planar graphs remains NP-hard even for planar graphs that are regular of degree 4 [8]. The related problem, MDS in planar graphs, is also NP-hard even for planar graphs with maximum vertex degree 3 and planar graphs that are regular of degree 4 [8]. It is well-known that MDS in planar graphs possesses a polynomial-time approximation scheme (PTAS) based on the shifting strategy [2]. That is, there is a polynomial-time approximation algorithm with approximation factor $1+\varepsilon$, where ε is a constant arbitrarily close to 0. Thus, it is immediate to conclude that MCDS in planar graphs can be approximated within a factor $3+\varepsilon$ for any $\varepsilon > 0$ in polynomial time. However, the degree of the polynomial grows with $1/\varepsilon$ and hence, the approximation scheme is hardly practical. Furthermore, the shifting strategy and dynamic programming requires expensive distributed implementation.

Recently, MCDS in unit disk graphs has generated much interest due to its relevance to wireless ad hoc networks. MCDS in unit-disk graphs is still NP-hard [5]. But MCDS in unit-disk graphs possesses a PTAS [4]. However, such PTAS is not suitable for practical applications such as distributed construction of virtual backbone in wireless ad hoc networks [3]. A simple 10-approximation algorithm for MCDS in unit-disk graphs was first proposed in [11]. An improved 8-approximation algorithm for MCDS in unit-disk graphs together with efficient distributed implementation was recently developed in [12]. Other centralized algorithms with approximation ratios less than 8 can be found in [1].

In this paper, we present a simple heuristic for MCDS in general graphs. When running on graphs excluding K_m (the complete graph of order m) as a minor, the heuristic has an approximation ratio of at most 7 if $m=3$, or at most $\frac{m(m-1)}{2}+5$ if $m \geq 4$. In particular, if running on a planar graphs, the heuristic has an approximation ratio of at most 15. Because of its simplicity, our heuristic is expected to admit efficient distributed implementation, which will be the subject of our further study.

The remaining of this paper is organized as follows. In Section 2, we introduce some related graph-theoretic concepts and parameters. In Section 3, we describe our heuristic for MCDS in general graphs. In Section 4, we provide an upper bound on the cardinality of the CDS output by our heuristic. Finally, we conclude this paper in Section 5.

2. Preliminaries

Let $G = (V, E)$ be a graph. We sometimes write $V(G)$ instead of V and $E(G)$ instead of E. For $U \subseteq V$, we use $G[U]$ to denote the subgraph of G induced by U. A subset $U \subset V$ is a *dominating set* (DS) of G if each node not in U has a neighbor in U. A subset $U \subset V$ is a *connected dominating set* (CDS) of G if it is a DS of G and $G[U]$ is connected. The *domination number*, denoted by $\gamma(G)$, and the *connected domination number*, denoted by $\gamma_c(G)$, are the smallest cardinalities of a dominating set and a connected dominating set, respectively. The distance $dist_G(u, v)$ in G of two vertices $u, v \in V(G)$ is the length of a shortest path between u and v in G. The distance between a vertex v and a set $U \subseteq V(G)$ is $\min_{u \in U} dist_G(u, v)$. The distance between two subsets U and W of $V(G)$ is $\min_{u \in U, w \in W} dist_G(u, w)$. A vertex set $U \subseteq V(G)$ is a k-independent set (k-IS) of G if the distance between any pair of vertices in U is greater than k. The *k-independence number* of G,denoted by $\alpha_k(G)$, is the largest cardinality of a k-IS. Note that a 1-IS is a usual IS and $\alpha_1(G)$ is the usual independence number $\alpha(G)$. The parameters $\gamma(G), \gamma_c(G), \alpha(G)$ and $\alpha_2(G)$ are related by the following inequalities [6].

$$\begin{aligned} \gamma(G) &\leq \gamma_c(G) \leq 3\gamma(G) - 2; \\ \gamma_c(G) &\leq 2\alpha(G) - 1; \\ \alpha_2(G) &\leq \gamma(G) \leq \alpha(G). \end{aligned}$$

To see why $\alpha_2(G) \leq \gamma(G)$, let $U \subseteq V(G)$ be a maximum 2-IS of G. For each $u \in U$, let $N[u]$ denote the closed neighborhood of u in G. Then the closed neighborhoods $N(u)$ for all $u \in U$ are pairwise disjoint. Thus each DS of G must contain at least one vertex from each $N[u]$. This implies that $\gamma(G) \geq \alpha_2(G)$.

A *contraction* of an edge (u, v) in G is made by identifying u and v with a new vertex whose neighborhood is the union of the neighborhoods of u and v (with resulting multiple edges and self-loops deleted). A *contraction* of G is a graph obtained from G by a sequence of edge contractions. A graph H is a *minor* of G if H is the contraction of a subgraph of G. G is *H-free* if G has no minor isomorphic to H. For example, by Kuratowski's theorem, a graph is planar if and only if it is both K_5 and $K_{3,3}$-free. In this paper, we focus on K_m-free graphs. Our algorithm would find a CDS of size at most

$$\left(\frac{m(m-1)}{2} + 5\right)\alpha_2(G) - 5$$

of a K_m-free graph G for any $m \geq 4$. This implies that if G is K_m-free for some $m \geq 4$, then

$$\gamma_c(G) \leq \left(\frac{m(m-1)}{2} + 5\right)\alpha_2(G) - 5.$$

In particular, for a planar graph G,

$$\gamma_c(G) \leq 15\alpha_2(G) - 5.$$

3. Algorithm Description

We first give a brief overview on our algorithm design. Our algorithm is presented as a color-marking process. All vertices are white initially, and will be marked with either black or gray eventually. In the end, all black vertices form a CDS and all gray vertices are dominatees. A white vertex remains white until either it is selected as a dominator, in which case it is marked black, or one of its neighbors is marked black, in which case it is marked gray. A gray vertex may be remarked black in the future. But once a vertex is marked black, it will stay black. The color-marking process proceeds in iterative phases. Each phase produces additional black nodes, which together with the black nodes from the previous phases, induce a connected subgraph. When a node is marked black, all its non-black neighbors will receive a time-stamp which is equal to the current phase number. New time-stamp does not overwrite old time-stamps, if there is any. Thus, a gray vertex may have multiple time-stamps. At the end of a phase k, each white vertex, if there is any left, has a gray neighbor with time-stamp j for every $1 \leq j \leq k$.

For the simplicity of description, we introduce some new terms and notations. Given a coloring marking of all vertices of G, the *residue graph* is the graph obtained from G by first removing all black vertices and those gray vertices without white neighbors, and then removing edges between gray vertices. Thus, each vertex of a residue graph is either white or gray, and each connected component of a residue graph must have at least one white vertex.

Consider a connected graph H and a positive integer k which satisfy the following properties: Each vertex of H is either white or gray and at least one vertex is white. If $k = 1$, then all vertices are white; and otherwise, every white vertex is adjacent to a gray vertex stamped with j for every $1 \leq j \leq k-1$. Such pair (H, k) is referred to as a *residue pair*. A *restricted 2-connected dominating set* (R2CDS) for a residue pair (H, k) is a subset of vertices U of H such that

- $H[U]$ is connected;
- every white vertex not in U, if there is any, is at a distance of exactly two from U;

- and for every $1 \leq j \leq k-1$, U contains at least one gray vertex stamped with j.

We propose a simple procedure, called R2CDS(H, k), which takes as input a residue pair (H, k) and outputs a R2CDS U for (H, k). All vertices in U are marked black, and all vertices dominated by U are marked gray and stamped with k. Given a vertex v and a positive integer k, we use MarkStamp(v, k) to denote the basic operation which marks v black and all white neighbors of v gray, and stamps all non-black neighbors of v with k. The procedure R2CDS(H, k) consists of four steps:

1. Initialization: If $k \geq 2$, let $a_j = 0$ for $j = 1, \cdots, k-1$.
2. Sorting: Build a spanning arborescence T of H rooted at a white vertex. For each node v of H, assign a rank $(\ell(v), v)$ where $\ell(v)$ is the level of node v in T. Sort all white vertices in the increasing lexicographic order of ranks. Let $v_1, v_2, \cdots, v_s$ denote the ordering.
3. Coloring and Stamping: MarkStamp(v_1, k). For $i = 2, \cdots, s$, if v_i is white and has no gray neighbors stamped with k, proceed as follows:
   (a) Set $l = 1$, $u_1 = v_i$. Repeat the following iteration until u_l is black: If $k \geq 2$ and u_l is gray, set $a_j = 1$ for each stamp $j < k$ of u_l. If u_l has a neighbor black, set u_{l+1} to any such neighbor; otherwise, if u_l has a gray neighbor stamped with k, set u_{l+1} to any such neighbor; otherwise, set u_{l+1} to its parent in T. Set l to $l+1$.
   (b) Repeat the following iteration until $l = 1$: Set l to $l-1$ and invoke MarkStamp(u_l, k).
4. Post-processing: If $k \geq 2$, perform the following processing. For $j = 1, \cdots, k-1$, if $a_j = 0$ choose a neighbor u of v_1 stamped with j, set $a_t = 1$ for each stamp $t < k$ of u, and then MarkStamp(u, k).

The $k-1$ boolean variables a_j for $j = 1, \cdots, k-1$ indicates whether the R2CDS U contains at least one gray vertex stamped with j for $j = 1, \cdots, k-1$ (the third bullet of the properties of U). They are initialized to zero in the first step. Whenever a gray vertex with stamp j is marked black within step 3 or step 4, a_j is set to one. Step 4 ensures all these boolean variables are one eventually, so that the third bullet is satisfied.

The For-loop in step 3 guarantees that in the end, every white vertex is adjacent to a gray vertex stamped with k, and thus is exactly two hops away from some black vertex. So the second bullet of the properties of U is satisfied.

The inner loop in step 3(a) establishes a path from a white vertex v_i without gray neighbors stamped with k to some black vertex. The inner loop in step 3(b) marks all vertices of the path black and marks/stamps the other vertices dominated by them. This ensures the connectedness of the black vertices, so that the first bullet of the properties of U is satisfied. The path (excluding the black vertex) consists of either three or four vertices. Indeed, u_1 is v_i, and since v_i is white and has no gray neighbors stamped with k, u_2 is always set to the parent of v_i. Depending on the color of u_2, the other vertices are selected as follows:

1. In the first case, u_2 is white. Then u_2 must have a gray neighbor stamped with k as early as when u_2 is examined, for otherwise, it would have been marked black. Thus, u_3 is a gray neighbor of u_2 stamped with k. The path thus consists of the three vertices u_1, u_2 and u_3.

2. In the second case, u_2 is gray. Then every stamp of u_2 is less than k. If u_2 has a gray neighbor stamped with k, then u_3 is one of such gray neighbors, and the path just consists of the three vertices u_1, u_2 and u_3. So we assume u_2 is not adjacent to any gray neighbor stamped with k. As u_2 is not adjacent to any black vertex (for otherwise, u_2 would have the stamp k), u_3 is set to the parent of u_2. Since no gray vertices with stamps less than k are adjacent in H, u_3 must be white. Then u_3 must have a gray neighbor stamped with k as early as when u_3 is examined, for otherwise, u_3 would have been marked black. Thus, u_4 is set to one of such gray neighbors, and the path just consists of the four vertices u_1, u_2, u_3 and u_4.

In summary, the path consists of either three vertices or four vertices. And if the path consists of four vertices, then k must be greater than one and at least one a_j is set to one for some $1 \leq j \leq k-1$ in step 3(a).

Now we are ready to describe our heuristic, denoted by CDS(G), for finding a CDS of G. Initially, $k = 0$ and all vertices of G have white colors. Repeat the following iteration while there are some white vertices left:

- Let $k = k + 1$ and G_k denote the residue graph.
- For each connected component H of G_k, apply R2CDS(H, k).

Let B denote the set of black vertices marked by CDS(G). It is easy to see that B is a CDS of G. In the next section, we will provide an upper bound on $|B|$ if the graph G is free of K_m-minor for some $m \geq 3$.

4. Performance Analysis

The main theorem of this section is given below.

Theorem 1 *Suppose that G is free of K_m-minor for some $m \geq 3$. If $m = 3$, then*

$$|B| \leq 7\alpha_2(G) - 4.$$

If $m \geq 4$, then

$$|B| \leq \left(\frac{m(m-1)}{2} + 5\right)\alpha_2(G) - 5.$$

By Kuratowski's theorem, a planar graph has no K_5-minor. So we have the following corollary of Theorem 1.

Corollary 1 *If G is a planar graph, then*

$$|B| \leq 15\alpha_2(G) - 5.$$

Since

$$\alpha_2(G) \leq \gamma(G) \leq \gamma_c(G),$$

Theorem 1 implies that when running on graphs excluding K_m (the complete graph of order m) as a minor, our heuristic has an approximation ratio of at most 7 if $m = 3$, or at most $\frac{m(m-1)}{2} + 5$ if $m \geq 4$. In particular, if running on a planar graphs, our heuristic has an approximation ratio of at most 15. The remaining of this section is dedicated to the proof for Theorem 1.

Let H be a graph in which every vertex is either white or gray and there is at least one white vertex. A *restricted 2-independent* set (R2IS) is a 2-IS of H which consists of only white vertices. The *restricted 2-independence number* of H, denoted by $\alpha_2'(H)$, is the largest cardinality of a R2IS of H. Obviously, $\alpha_2'(H) \leq \alpha_2(H)$. The next lemma presents the "monotonic" properties of the residue graphs.

Lemma 1 *Suppose that CDS(G) runs in l iterations. Then*

$$\begin{aligned} G &= G_1 \supset G_2 \supset \cdots \supset G_l; \\ \alpha_2(G) &= \alpha_2'(G_1) \geq \alpha_2'(G_2) \geq \cdots \geq \alpha_2'(G_l). \end{aligned}$$

Proof. It is obvious that $G_1 = G$ and $\alpha_2'(G_1) = \alpha_2(G)$. Fix a k between 1 and $l-1$. We prove that $G_{k+1} \subset G_k$ and $\alpha_2'(G_{k+1}) \leq \alpha_2'(G_k)$.

We first show that $V(G_{k+1}) \subset V(G_k)$. Note that all white vertices of G_{k+1} must be have been white in the previous iteration and thus are white vertices of G_k as well. In addition, all gray vertices of G_{k+1} which are white in the previous iteration must be white vertices of G_k. So it is sufficient to show that each gray vertex of G_{k+1} which is also gray in the previous iteration is also a vertex of G_k. Let v be a gray vertex of G_{k+1} which is also gray in the previous iteration. Then v has a white neighbor, denoted by u, in G_{k+1}. Since u is also a white vertex of G_k, v must be also a gray vertex of G_k.

Next, we show that $E(G_{k+1}) \subset E(G_k)$. Consider any edge uv of G_{i+1}. Then at least one of its endpoints is white. By symmetry, assume v is white. If u is also white, then the edge uv also appears in G_k. If u is gray, then u is either white or gray in G_k. In either case, the edge uv appears in G_k.

Finally, we show that $\alpha_2'(G_{k+1}) \leq \alpha_2'(G_k)$. Let w_1 and w_2 be any pair of white nodes of G_{k+1}. As G_{k+1} is a subgraph of G_k,

$$dist_{G_{k+1}}(w_1, w_2) \geq dist_{G_k}(w_1, w_2).$$

We claim that, however, if $dist_{G_k}(w_1, w_2) \leq 2$, then

$$dist_{G_{k+1}}(w_1, w_2) = dist_{G_k}(w_1, w_2).$$

The claim is true if $dist_{G_k}(w_1, w_2) = 1$. So we assume that $dist_{G_k}(w_1, w_2) = 2$. Then $dist_{G_{k+1}}(w_1, w_2) \geq 2$. Let v be a common neighbor of w_1 and w_2 in G_k. Then v must remain as a vertex of G_{k+1}, for otherwise, v would have been marked black in the previous iteration and both w_1 and w_2 would have become gray in G_{k+1}. Thus, $dist_{G_{k+1}}(w_1, w_2) = 2$. So our claim is true. From the claim, we conclude that if $dist_{G_{k+1}}(w_1, w_2) > 2$, then $dist_{G_k}(w_1, w_2) > 2$. This implies that $\alpha_2'(G_{k+1}) \leq \alpha_2'(G_k)$. □

The lemma below gives an upper bound on the total number of iterations if the graph G is free of K_m-minor.

Lemma 2 *If G is free of K_m-minor for some $m \geq 3$, then CDS(G) runs in at most $m-1$ iterations.*

Proof. We prove the lemma by contradiction. Assume that G is free of K_m-minor but CDS(G) runs in at least m iterations. Let H_m^* be an arbitrary connected component of G_m. By Lemma 1, for each $1 \leq k \leq m-1$, G_k has a unique connected component, denoted by H_k^*, which contains H_m^* as a subgraph. Obviously,

$$H_1^* \supset H_2^* \supset \cdots \supset H_m^*.$$

For each $1 \leq k \leq m$, let B_k^* be the black vertices of H_k^* marked by the procedure R2CDS(H_k^*, k). Then for any $1 \leq i < j \leq m$, B_i^* and B_j^* are disjoint and separated by one hop. Since each B_k^* is connected, the m sets $B_1^*, B_2^*, \cdots, B_m^*$ give rise to a K_m-minor in G, which is a contradiction. □

The next lemma provides an upper bound on the number of black vertices output by the procedure R2CDS(H, k).

Lemma 3 *The number of black vertices output by the procedure R2CDS(H, k) is at most $3\alpha_2'(H) - 2$ if $k = 1$, and at most $4\alpha_2'(H) + k - 4$ if $k \geq 2$.*

Proof. Let $v_1, v_2, \cdots, v_s$ be the ordering of the white vertices of H produced by Step 2 of the procedure R2CDS(H,k). Let I be the set of integers i in $\{2,\cdots,s\}$ such that when v_i is examined in the for-loop of Step 3, v_i is white and has no gray neighbors stamped with k. It is obvious that $\{v_i : i \in \{1\} \cup I\}$ form a R2IS of H. Thus,

$$1 + |I| \le \alpha_2'(H).$$

Next, we count the number of vertices marked black during each iteration i with $i \in I$ in the for-loop of Step 3. Fix an $i \in I$. From the explanation after the procedure R2CDS(H,k) in the previous section, either three or four vertices are marked black during iteration i. In addition, if four vertices are marked black in this iteration, then k must be greater than one and at least one a_j is set to one for some $1 \le j \le k-1$.

Finally, we count the total number of black vertices. Note that v_1 is always marked black. If for each $i \in I$, the iteration i of the for-loop at Step 3 marks exactly three vertices black, then Step 4 marks at most $k-1$ additional vertices black. So the total number of black vertices is at most

$$\begin{aligned} 1 + 3|I| + k - 1 &= 3(1+|I|) + k - 3 \\ &\le 3\alpha_2'(H) + k - 3. \end{aligned}$$

If for some $i \in I$, the iteration i of the for-loop at Step 3 marks four vertices black, then $k > 1$ and Step 4 marks at most $k-2$ additional vertices black. So the total number of black vertices is at most

$$\begin{aligned} 1 + 4|I| + k - 2 &= 4(1+|I|) + k - 5 \\ &\le 4\alpha_2'(H) + k - 5. \end{aligned}$$

Thus, if $k = 1$, the total number of black vertices is at most

$$3\alpha_2'(H) + 1 - 3 = 3\alpha_2'(H) - 2.$$

If $k \ge 2$, the total number of black vertices is at most

$$\begin{aligned} &\max\{3\alpha_2'(H) + k - 3, 4\alpha_2'(H) + k - 5\} \\ \le 4 \quad &\alpha_2'(H) + k - 4 \end{aligned}$$

□

The next lemma gives upper bounds on the number of black vertices produces in each iteration of CDS(G).

Lemma 4 *Let B_k be the set of black vertices produced in the k-th iteration of CDS(G). Then*

$$\begin{aligned} |B_1| &\le 3\alpha_2(G) - 2, \\ |B_2| &\le 4\alpha_2(G) - 2, \\ |B_3| &\le 4\alpha_2(G) - 1, \\ |B_k| &\le k\alpha_2(G), k \ge 4. \end{aligned}$$

Proof. ¿From Lemma 3, $|B_1| \leq 3\alpha_2(G) - 2$. So we assume that $k > 1$. Suppose that G_k has t connected components, denoted by $H_{k,1}, \cdots, H_{k,t}$. Since each connected component contains at least one white vertex,

$$1 \leq t \leq \sum_{i=1}^{t} \alpha_2'(H_{k,i}) = \alpha_2'(G_k).$$

For each $1 \leq i \leq t$, let $B_{k,i}$ be the vertices of $H_{k,i}$ marked by the procedure R2CDS$(H_{k,i}, k)$. Then

$$B_k = B_{k,1} \cup \cdots \cup B_{k,t};$$

and by Lemma 3,

$$|B_{k,i}| \leq 4\alpha_2'(H_{k,i}) + k - 4$$

for each $1 \leq i \leq t$. Thus, if $k = 2$ or 3,

$$\begin{aligned} |B_k| &= \sum_{i=1}^{t} |B_{k,i}| \leq 4\sum_{i=1}^{t} \alpha_2'(H_{k,i}) + (k-4)t \\ &= 4\alpha_2'(G_k) + (k-4)t \\ &\leq 4\alpha_2(G) + (k-4). \end{aligned}$$

If $k \geq 4$,

$$\begin{aligned} |B_k| &= \sum_{i=1}^{t} |B_{k,i}| \leq 4\sum_{i=1}^{t} \alpha_2'(H_{k,i}) + (k-4)t \\ &\leq 4\alpha_2'(G_k) + (k-4)\alpha_2'(G_k) = k\alpha_2'(G_k) \\ &\leq k\alpha_2(G). \end{aligned}$$

□

Now we are ready to give the proof of Theorem 1. By Lemma 2, the total number of iterations is at most $m - 1$. If $m = 3$, then by Lemma 4

$$|B| \leq (3\alpha_2(G) - 2) + (4\alpha_2(G) - 2) \leq 7\alpha_2(G) - 4.$$

If $m = 4$, then by Lemma 4,

$$\begin{aligned} |B| &\leq (7\alpha_2(G) - 4) + (4\alpha_2(G) - 1) \\ &= 11\alpha_2(G) - 5 \\ &= \left(\frac{m(m-1)}{2} + 5\right)\alpha_2(G) - 5. \end{aligned}$$

If $m > 4$, by Lemma 4,

$$\begin{aligned} |B| &\leq (11\alpha_2(G) - 5) + \sum_{k=4}^{m-1} k\alpha_2(G) \\ &= 11\alpha_2(G) - 5 + \left(\frac{m(m-1)}{2} - 6\right)\alpha_2(G) \\ &= \left(\frac{m(m-1)}{2} + 5\right)\alpha_2(G) - 5. \end{aligned}$$

This completes the proof of Theorem 1.

5. Conclusion

In this paper, we present a simple heuristic for MCDS in graphs. When running on a graph G excluding K_m (the complete graph of order m) as a minor, the heuristic produces a CDS of cardinality at most $7\alpha_2(G) - 4$ if $m = 3$, or at most $\left(\frac{m(m-1)}{2} + 5\right)\alpha_2(G) - 5$ if $m \geq 4$. In particular, if running on a planar graph G, the heuristic outputs a CDS of cardinality at most $15\alpha_2(G) - 5$. We are currently developing an efficient distributed implementation of this heuristic.

Acknowledgements

Authors would like to thank the reviewers for their valuable comments.

References

1. K. M. Alzoubi, *Virtual Backbone in Wireless Ad Hoc Networks*, PhD dissertaion, Department of Computer Science, Illinois Institute of Technology, 2002.
2. B. S. Baker, "Approximation algorithms for NP-complete problems on planar graphs", *Journal of the ACM*, 41(1):153–180, 1994.
3. V. Bharghavan, and B. Das, "Routing in Ad Hoc Networks Using Minimum Connected Dominating Sets", *International Conference on Communications'97*, Montreal, Canada. June 1997.
4. X. Cheng, X. Huang, D. Li, and D.-Z. Du, "Polynomial-Time Approximation Scheme for Minimum Connected Dominating Set in Ad Hoc Wireless Networks", *Technical Report* 02-003, Department of Computer Sceince and Engineering, University of Minnesota.
5. B. N. Clark, C. J. Colbourn, and D. S. Johnson, "Unit Disk Graphs", *Discrete Mathematics*, 86:165–177, 1990.
6. P. Duchet, and H. Meyniel, "On Hadwiger's number and stability numbers", *Annal. Disc. Math.* 13 (1982), 71-74.
7. U. Feige, "A threshold of $\ln n$ for approximating set-cover", *28th ACM Symposium on Theory of Computing*, pp. 314–318, (1996).
8. M. R. Garey, and D. S. Johnson, *Computers and Intractability: a Guide to the Theory of NP-Completeness*, W. H. Freeman and Company, 1979.
9. S. Guha, and S. Khuller, "Approximation algorithms for connected dominating sets", *Algorithmica*, 20(4):374–387, April 1998.
10. C. Lund, and M. Yannakakis, "On the hardness of approximating minimization problems", *Journal of the ACM*, 41(5): 960–981, (1994).
11. M. V. Marathe, H. Breu, H. B. Hunt III, S. S. Ravi, and D. J. Rosenkrantz, "Simple Heuristics for Unit Disk Graphs", *Networks*, Vol. 25, 1995, pp. 59–68.
12. P.-J. Wan, K. M. Alzoubi, and O. Frieder, "Distributed Construction of Connected Dominating Set in Wireless Ad Hoc Networks", *IEEE INFOCOM* 2002.