A Faster Approximate Method to Identify Minimum Dominating Set

You Zhou , Guodong Lv, Baoxin Xiu ,Weiming Zhang and Qing Cheng
Science and Technology on Information Systems Engineering Laboratory
National University of Defense Technology
Changsha, Hunan Province, China.
zhouyou12690@gmail.com

Abstract—**The minimum dominating set (MDS) problem is known to be NP-hard. Compared with the currently fastest extract algorithm for MDS problem on graphs of n vertices using O($2^{0.59n}$)time, which could merely tackle these problems with 200 vertices scale in 8 hours, the paper presents a faster approximate layer-method to address MDS problems in O(n^2) time and the method is proved to be effective particularly for larger scale MDS. Furthermore, the programing phases are shown in detail.**

Keywords—Minimum dominating set; network; layer method

I. Introduction

More accurate and agiler exact algorithms for NP hard and NP complete problems, such as minimum dominating set (MDS), vertex cover[9-11], 3-cloring[12-13], and maximum independent set[14-17] etc. have been developed by researchers since the seminal work of Tarjan and Trojanowski [18]. Notably, about 50 years ago, Claude Berge firstly described the concept of dominating in a graph [4] and later Ore utilized the term MDS [5]. In 1979, the minimum dominating set (MDS) problem was proved to be one of the classical NP hard problems [1, 7]. But although it is difficult to improve the efficiency of algorithms to identify the MDS, the great value of MDS in tacking practical problems like virus spreading, router distribution, information searching, image processing etc. [7] has drawn heaps of studies on MDS problems.

In 2004, Fomin FV, Kratsch D and Woeginger G proposed FKW algorithm for MDS problems and it could address n-vertex MDS problem in approximately O ($2^{0.955n}$) time, prior to the original O (2^n) time [1]. However, the result was still out of satisfaction. Later, they improved the algorithm in 2005 and the time complexity was reduced to O ($2^{0.8235n}$) [5]. In 2006 a faster algorithm was developed by Grandoni [6] using the Measure and Conquer technique, and this method could solve such problem in O ($2^{0.61n}$) time. Also, Johan presented another smart algorithm with O ($2^{0.59n}$) in 2009 [2]. However, even the fastest algorithm could solve only about 200-vertex-scale problem in eight hours [7]. The result definitely cannot be practically applied to control the key dominating nodes set in traditional real networks with thousands of vertices. As the number of the vertices increases, the so-called "dimension disaster" comes about and the time used for finding the exact MDS becomes unacceptably long. It is therefore imperative to discover a generalizable and fast method to identify MDS in real networks.

In this paper, we present an O(n^2) layer-method for MDS, where n is the number of nodes on a graph.

The rest of this paper is organized as follows. In the next section, basic concepts and definition for MDS are briefly introduced. The proposed layer-method is described in Sec. 3. In Sec.4, the performance of the proposed method is evaluated through the simulation experiments. Finally, sec. 5 concludes the paper.

II. Preliminaries and Definitions

In this section, to provide a sufficient background for the remainder of the paper, we present a brief overview of the MDS problems and relevant lemma.

A. Basic Graphical Definition

Let $G=(V,E)$ be an n-vertex undirected, connected, simple graph without loops. The number of vertices of G is denoted by n and the neighborhood of a vertex v is denoted by $N(v)=\{u\in V|\{u,v\}\in E\}$. The degree of a vertex v is $|N(v)|$. For a vertex set $S\subseteq V$, we define $N[S]=\bigcup$ and $N(S)=N[S]-S$.

A set $D\subseteq V$ with $N[D]=V$ is called a dominating set for G if every vertex of G is either contained in D or adjacent to some vertices in D. The domination number $\gamma(G)$ of a graph G is the minimum cardinality of a dominating set of G. The MDS problem needs to determine $\gamma(G)$ and find a dominating set of minimum cardinality.

B. Rules of Layering a Graph

Choose a vertex with the maximum degree first (if more than one vertices have such degree, choose anyone), and put this vertex on the first layer namely $L(1)$. Obviously, $L(1)$ only contains one vertex with the maximum degree. Then we define $N(L(1))$ as the second layer $L(2)$. For $i\geq 2$ we define $L(i+1)$ as $\{v\,|\,v\in N(L(i))\bigcap$ -1)}. By this way, we could layer a graph easily.

For $\forall v\in V$, $v\in L(i), i\geq 2$, we define the neighbors of v on its previous layer as

978-1-4799-3279-5 /14/$31.00 ©2014 IEEE

$$N^{\mathrm{U}}(v) = \{u \mid u \in N(v) \bigcap \qquad -1)\}$$

For $\forall v \in V$, $v \in L(i)$, $i \geq 2$, we define vertices in $N(v)$ except for those on the same layer of v as

$$\overline{N}(v) = \{u \mid u \in N(v) \bigcap \qquad \}$$

For $\forall v \in V$, $v \in L(i), i \geq 2$, $d'(v)$ represents the number of edges between v and its neighbors on the previous layer of v.

Based on such definitions, we could get the following lemma.

Lemma 1. Let $G = (V, E)$ be a connected, simple graph. If vertices u and v ($u, v \in V, u \in L(i), v \in L(j)$) are adjacent, then $|i - j| \in \{0,1\}$.

Lemma 1 furnishes us that the neighbors of a vertex are either on its previous layer or next layer as well as same layer. Similarly, the neighbors of all the vertices on a particular layer are either on its previous or next layer. Notably, a vertex may have no next-layer-neighbor.

III. DESCRIPTION OF THE LAYER METHOD

A. Phases of the Layer Method

The layer method to identify MDS is described in detail and it contains four phases as follows.

Phase 1: Layer a graph. The vertices of a graph should be firstly layered to different layers based on our layering rules. After layering, each vertex belongs to a fixed layer, and each vertex's neighbors could be allocated into different layers: the same layer, its previous layer, or its next layer.

Phase 2: Dominate the no-next-layer-neighbor (NNLN) vertex. Note that such vertex as vertex b in both Fig.1. (a) and Fig.1.(b) does not have next layer, but it definitely has previous layer according to our layering rules.

For every NNLN vertex v_i, we should sort them in an ascending order according to the $d'(v_i)$, and we could get the NNLN set where the first element v_1 has the minimum d'. Then we label $u \in N^U(v_i)$ as a dominating vertex of v_i, if $N^U(v)$ contains more than one vertices, we choose the one which could dominate more NNLN vertices.

If these dominating vertices could dominate the whole graph, skip phase 3 and 4. If not, go next.

Phase 3: Label dominating layer. At the end of phase 2, all the NNLN vertices have been dominated. If the vertices of a whole layer are labelled as dominating vertices, both its previous layer and its next layer could be guaranteed to be dominated.

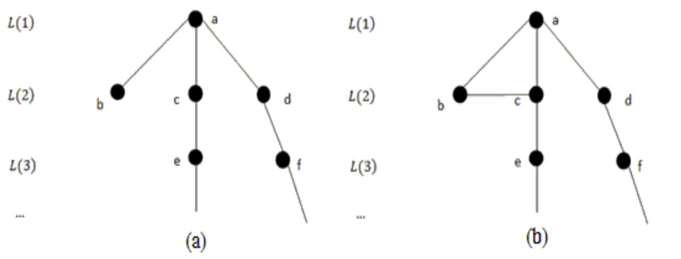

Fig. 1. Two cases of no-next-layer-neighbor

We define a layer as dominating layer if all the vertices except NNLN vertices in this layer are dominating vertices. Labeling a layer as dominating layer means that we label the vertices except for those NNLN vertices in this layer as dominating vertices. Then we label first layer and the penultimate layer as dominating layers.

Considering that two closest dominating layers are $L(m)$ and $L(n)$ (m<n) respectively,

If $n - m \leq 3$, do nothing.

Else if $n - m = 3k + 1$ (k ≥ 1), choose the layers of $\min\{\sum_{i=1}^{k}|L'(m+3i)|, \sum_{i=0}^{k\text{-}1}|L'(m+3i{+}1)|, \sum_{i=0}^{k\text{-}1}|L'(m+3i{+}2)|\}$ and label them.

Where $L'(i)$ represents such vertices of layer i except NNLN vertices and dominating vertices, and $|L'(i)|$ is the number of vertices of $L'(i)$.

Else if $n - m = 3k + 2$ (k ≥ 1), choose the layer of

$$\min\{\sum_{i=1}^{k}|L'(m+3i)|, \sum_{i=0}^{k}|L'(m+3i+1)|, \sum_{i=0}^{k-1}|L'(m+3i+2)|\}$$

and label them.

Else if $n - m = 3k$ (k ≥ 2), choose the layer of

$$\min\{\sum_{i=1}^{k-1}|L'(m+3i)|, \sum_{i=0}^{k-1}|L'(m+3i+1)|, \sum_{i=0}^{k-1}|L'(m+3i+2)|\}$$

and label them.

After phase 3, the selected dominating vertices could already dominate the whole graph.

Phase 4: Remove redundant dominating vertices. Since there might exist redundant vertices in the dominating set, we need to reduce the redundancy as much as possible to achieve the MDS.

In each dominating layer, check out whether there exist such case as follows:

For vertex x and vertex y, if $\overline{N}(x) \subseteq \overline{N}(y)$ and $\exists v \in N(x)$, where v is a dominating vertex, then remove the vertex x from our dominating set.

With this method, we could identify the approximate minimal dominating set of a graph.

B. Complexity Analysis

Apparently, there is no complex computation during the four phases of the layer method. Assume a fundamental operation (comparison, labeling, canceling labeling, judgment, and laminating vertex) takes t_0 time, for a graph with n vertices, the process that each vertex is layered takes $n * t_0$ time during phase 1. On phase 2, the worst case of dominating NNLN vertices is that each dominating vertex just dominates one NNLN vertex, therefore, the phase takes $\frac{n}{2} * t_0$ time since the maximum number of NNLN vertices is $n/2$. Simultaneously, the dominating layers could be labelled on phase 3 almost at the same time. On phase 4, the process checking each couple of vertices in every dominating layer makes the biggest contribution to the time complexity and the worst case is $l = 3$ i.e. $|L(2)| = \left\lfloor \frac{n-1}{2} \right\rfloor$, so that we have to check $(\left\lfloor \frac{n-1}{2} \right\rfloor - 1)(\left\lfloor \frac{n-1}{2} \right\rfloor)$ couples (see Fig.2).

In conclusion, the whole method needs $\frac{n^2+2n+3}{4} * t_0$ time at most to identify the MDS, therefore, time complexity of our layer method is $O(n^2)$, which is identical to that of greedy method [19].

IV. EXPERIMENTS AND RESULTS

A. The effectiveness of layer method

As shown in Fig.3, we take the graph in article [4] for experiment. Fig.3.a shows the strong advice-seeking ties in global consulting company.

According to phase 1, vertex 5 maintains the biggest degree of 9. So node 5 should be placed on the first layer. Vertices 1, 2, 3, 4, 6, 8, 9, 10, 11 are adjacent to vertex 5, so they could be layered to the second layer. Fig.4 furnishes us with the insight into the layering result calculated by the layer method. Vertices 1, 2, 3, 4, 7, 8, 9, 11, 12, 15, 16, 19, 21, 17, 24, 28, 30, 31, 32, 29 are NNLN vertices, so according to phase 2, vertices 5, 6, 13, 20, 18, 23, 25, 26, 28 are dominating vertices as shown in Fig.5. As these 9 vertices could dominate the whole network, step 3 and 4 could be skipped. By the layer method, the black vertices shown in Fig.3.b are the dominating vertices and the result indicates the number of the dominating vertices of MDS is 9. The exact number of the MDS for the graph is also 9 given by document [4].

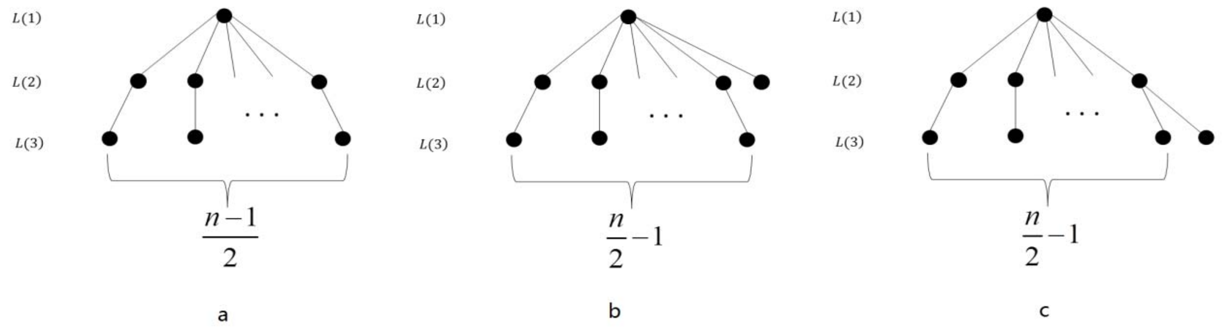

Fig. 2. Worst case of phase 4 (a) n is even (b),(c) n is odd

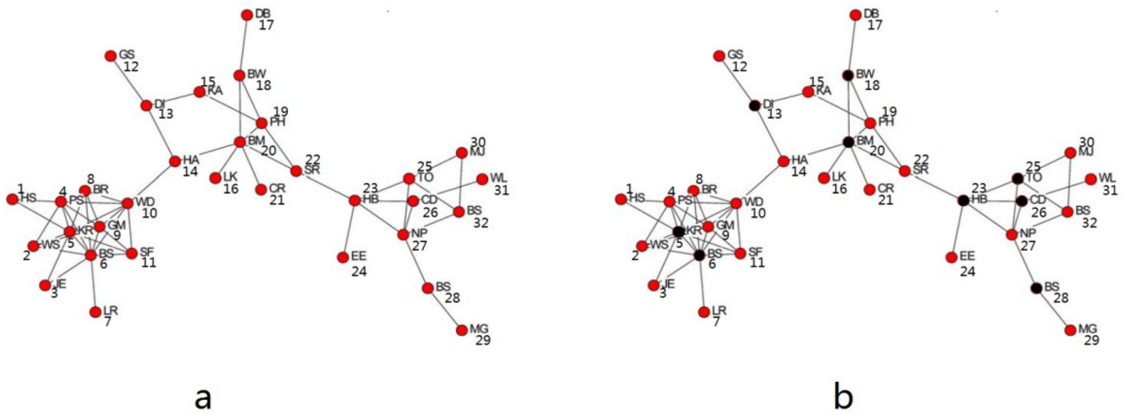

Fig. 3. Result of layer method (a). Strong advice-seeking ties in global consulting company. (b). Solution of MDS problem.

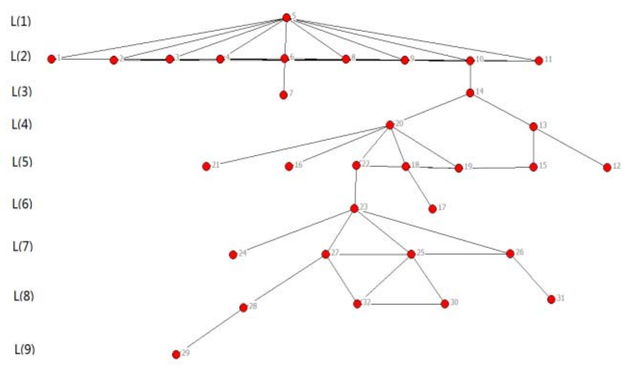

Fig. 4. The result of Phase 1.

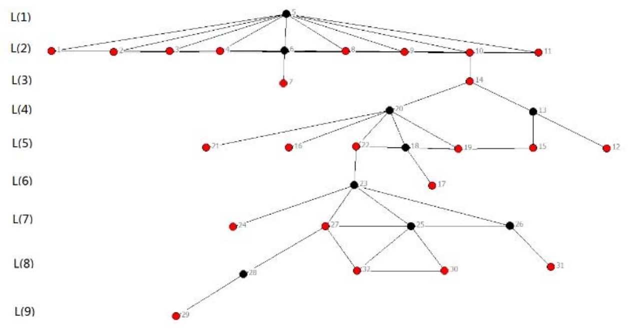

Fig. 5. The result of Phase 2.

B. *Comparative analysis of relevant methods*

As WWW, P2P network and social network are scale-free (SF) networks, several SF networks could be generated to experiment according to the principle proposed by Barabasi et al [8]. The results are the average performance of each test.

Regarding a graph with different vertices, we could get comparative results of relevant methods as Table 1 shows. (The experiments were conducted using a computer with the following characteristics: Intel(R) Pentium(R) Dual E2200, 2.20 and RAM:1.99GB, OS: Win XP; MATLAB).

In Table 1, the first column is the number of vertices of the experimental SF networks; column 2 to column 4 shows $\gamma(G)$ identified by different algorithms; column 5 to column 7 represents the time consumption of relevant methods. As an exact algorithm to solve MDS problem, Grandioni algorithm gets the accurate $\gamma(G)$, however it takes quite a long time. For instance, when n grows to 50, it will take more than 6 days to get solve the MDS problem and this is unacceptable for application. Compare with this accurate algorithm, greedy algorithm could save much time to get approximate results. From column 2 and 4, our layer method could identify the similar results as Grandioni method. Furthermore, with the increase of n, layer method becomes more time-saving than greedy method.

As shown in Fig.6, when average degree increases, $\gamma(\text{greedy})$ may be more accurate but the ratio of $\gamma(\text{greedy}) / \gamma(\text{layer})$ is astringent. With the dramatic increase of the scale of real networks, the time to identify MDS by

TABLE I. RESULTS OF MDS PROBLEM BY DIFFERENT ALGORITHMS.

n	$\gamma(G)$			**Time consumption (s)**		
	Grandioni	**greedy**	**layer**	**Grandioni**	**greedy**	**layer**
32	9	11	9	922	0.031	0.15
40	12	13	13	24900	0.032	0.153
45	14	17	14	212786	0.034	0.151
50		15	14	＞6 days	0.034	0.152
100		25	24		0.044	0.18
200		55	55		0.125	0.21
500		164	152		1.89	0.33
1000		318	303		18.66	0.934
2000		662	617		194.34	2.87

greedy or layer algorithm also rises. Actually, greedy algorithm is faster than layer algorithm when the scale of SF network is small because the phase 1 of layer algorithm practically is a pre-process which is the foundation of ensuing phases and consume a little drop of time. Nevertheless, when the number of vertices of a network grows beyond 200, calculation-time cost by greedy algorithm increases dramatically, but that of layer algorithm still maintains the efficiency.

More experiments have been done on random network and small world network to prove the accuracy and agility of our layer method. Similarly, we generate several ER random networks and WS small world networks with different vertices number and different average degree to conduct experiments with our method. Fig.7 and Fig.9 show that the ratio of $\gamma(\text{greedy})$ and $\gamma(\text{layer})$ decreases due to the increase of the average degree. When the average degree is small, $\gamma(\text{greedy})$ is bigger than $\gamma(\text{layer})$ i.e. layer algorithm is more accurate. But with the increase of average degree, $\gamma(\text{greedy})$ becomes smaller than $\gamma(\text{layer})$ and the ratio declines but it is astringent to 0.6 in Fig.7. That means the ratio fluctuates in a limited range.

Fig.8 and Fig.10 show the ratio of time consumption by greedy and layer algorithms. The ratio of $t(\text{greedy}) / t(\text{layer})$ increases as the scale of network become larger. When the network contains 2000 vertices the ratio reaches more than 60 in Fig.8 and more than 40 in Fig.10. This means that layer method is more time-saving. Furthermore, with the networks expanding, the layer algorithm turns to be more advantage.

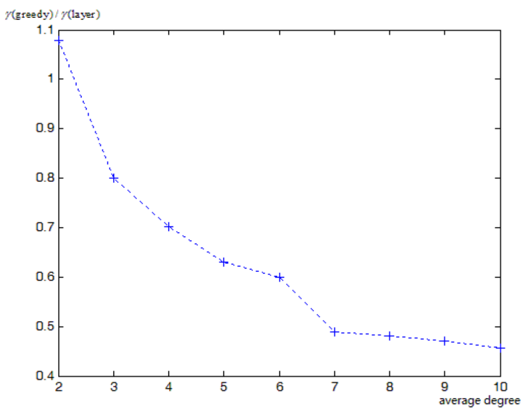

Fig. 6. $\gamma(\text{greedy}) / \gamma(\text{layer})$ at different average degree in SF network

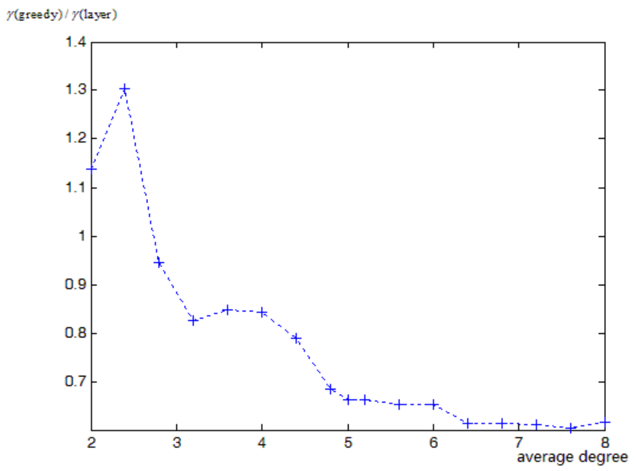

Fig. 7. $\gamma(\text{greedy}) / \gamma(\text{layer})$ at different average degree in random network

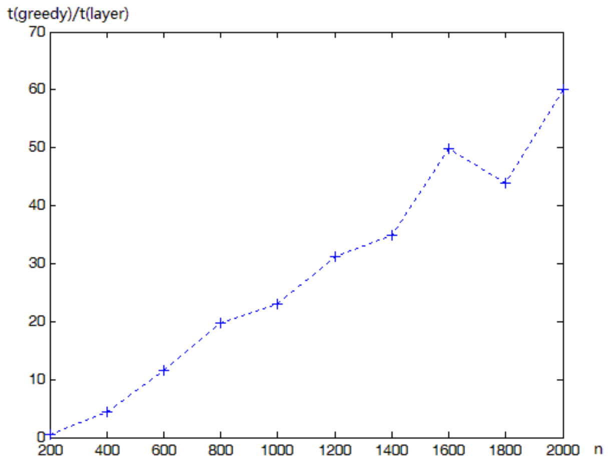

Fig. 8. $t(\text{greedy}) / t(\text{layer})$ at different scale of random network

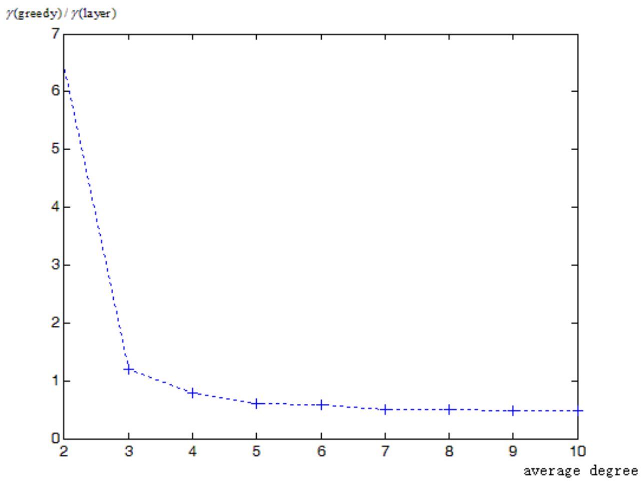

Fig. 9. $\gamma(\text{greedy}) / \gamma(\text{layer})$ at different average degree in small world network

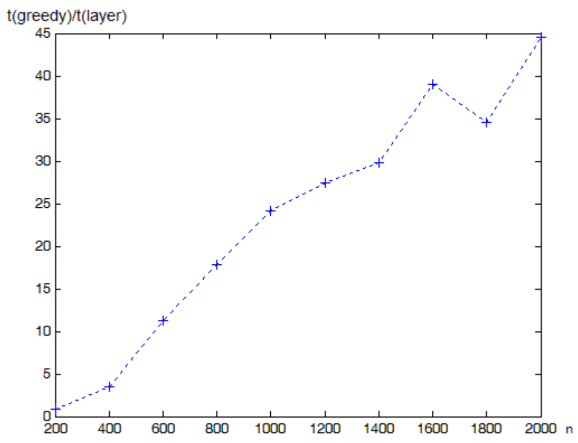

Fig. 10. $t(\text{greedy}) / t(\text{layer})$ at different scale of small world network

V. CONCLUSIONS AND DISCUSSIONS

In this paper, we represent a layer method to tackle the minimum dominating set problem using $O(n^2)$ time. The layer method can solve MDS problem fast at the cost of the accuracy. As the scale of network increases, the advantage of efficiency becomes obvious. Therefore, layer algorithm matches such cases when the scale of network is large and average degree is small. Particularly, the layer method is a smart option to the requirement and demand for high speed.

However, the accuracy of layer method may decrease when the average degree is big. During the ensuring work, we would focus on improving the accuracy of our method and relevant application.

ACKNOWLEDGMENT

This work was supported by the National Natural Science Foundation of China (Grants No. 71031007), National University of Defense Technology Master Research Funding No. S130503.

REFERENCES

[1] Fomin F V, Kratsch D, Woeginger G J. Exact (exponential) algorithms for the dominating set problem/ / LNCS 3353.Berlin: Springer, 2004: 245-256.

[2] Johan M M van Rooij, Jesper Nederlof , Thomas Cvan Dijk. Inclusion/ Exclusion meets measure and conquer: Exact algorithms for counting doinating sets/ / LNCS 5757, 2009:554- 565.

[3] M. R. Garey and D. S. Johnson. Computers and intractability. A guide to the theory of NP-completeness. W.H. Freeman and Co., San Francisco, 1979.

[4] Stephen P. Borgatti. Identifying sets of key players in a social network. Computational & Mathematical Organization Theory, 2006, 12:21-34 –Springer.

[5] Fomin F V, Grandoni F, Pyatkin A V, Stepanov A A. Bounding the number of minimal dominating sets: A measure and conquer approach//LNCS 3827. Berlin: Springer, 2005:573-582.

[6] Granoni F. A note on the complexity of minimum dominating set. Journal of Discrete Algorithms, 2006,4(2):209-214.

[7] LU Gang, ZHOU Ming-Tian, TANG Yong, WU Zhen-Qiang, QIU Guo-Yong, YUAN Liu. A Survey on Exact Algorithms for Dominating Set

Related Problems in Arbitrary Graphs. Chinese Journal of Computers, 2010, 33(6):1073-1087.

[8] Barabasi. A.-L., R.Albert, Emergence of scaling in random network, Scince 286:509-512(1999).

[9] R.Balasubramanian, M.Fellows, V.Raman, An improved fixed-parameter algorithm for vertex cover,Inform. Process. Lett. 65(1998) 163-168.

[10] J.Chen, I.Kanj, W.Jia, Vertex cover: further observations and further imporvements, J.Algorithm 41(2001)280-301.

[11] R.Niedermeier, P.Rossmanith, On efficient fixed-parameter algorithms for weighted vertex cover, J.Algorithms 47(2) (2003) 63-77.

[12] R.Beigel, D.Eppstein, 3-coloring in time O(1.3446^n): a no-MIS algorithm, in: IEEE Symposium on Foundations of Computer Science (FOCS), 1995, pp. 444-452.

[13] D.Eppstein, Improved algorithms for 3-coloring, 3-edge-coloring, and constraint satisfaction, in: ACM-SIAM Symposium on Discrete Algorithms (SODA), 2001, pp. 329-337.

[14] R.Beigel, Finding maximum independent sets in sparse and general graphs, in: ACM-SIAM Symposium on Discrete Algorithms (SODA), 1999, pp. 856-857.

[15] T.Jian, An O($2^{0.304n}$) algorithm for solving maximum independent set problem, IEEE Trans. Comput. 35(9) (1986) 847-851.

[16] J.M. Robson, Algorithms for maximum independent sets, J.Algorithms 7(3) (1986) 425-440.

[17] J.M. Robson, Finding a maximum independent set in time O($2^{n/4}$), Technical Report 1251-01, LaBRI, University Bordeaux I,2001.

[18] R.Tarjan, A.Trojanowski, Finding a maximum independent set, SIAM J.Comput. 6(3) (1977) 537-546.

[19] Ellis Horowitz, Sartaj Sahni, Susan Anderson-Freed. Fundamentals of Data Structures in C (ISBN: 0-7167-8250-2).