Approximating the Minimum Connected Dominating Set in Stochastic Graphs Based on Learning Automata

J. Akbari Torkestani
Department of Computer Engineering
Islamic Azad University- Arak Branch
Arak Iran
j-akbari@iau-arak.ac.ir

M. R. Meybodi
Department of Computer Engineering
Amirkabir University
Tehran Iran
mmeybodi@aut.ac.ir

Abstract— **The minimum connected dominating set (MCDS) of a given graph G is the smallest sub-graph of G such that every vertex in G belongs either to the sub-graph or is adjacent to a vertex of the sub-graph. Finding the MCDS in an arbitrary graph is a NP-Hard problem, and several approximation algorithms have been proposed for solving this problem in deterministic graphs, but to the best of our knowledge no work has been done on finding the MCDS in stochastic graphs. In this paper, the MCDS problem in the stochastic graphs is first introduced, and then a learning automata-based approximation algorithm called SCDS is proposed for solving this problem when the probability distribution function of the vertex weight is unknown. It is shown that by a proper choice of the parameters of the proposed algorithm, the probability with which the proposed algorithm find the MCDS is close enough to unity. The simulation results show the efficiency of the proposed algorithm in terms of the number of samplings.**

Keywords-component; ***Dominating set, minimum connected dominating set, stochastic graph, learning automata***

I. INTRODUCTION

The dominating set (DS) problems are a class of optimization problems which are widely used in wireless ad hoc networks. A dominating set of a graph $G=(V,E)$ is a subset $S \subseteq V$, such that every vertex $v \subseteq V$ is either in S or adjacent to a vertex of S. A vertex of S is said to dominate itself and all adjacent vertices. Finding the dominating set is a well-known approach, proposed for clustering the wireless ad hoc networks. A minimum DS (MDS) is a DS with the minimum cardinality. A dominating set is also an independent dominating set, if no two vertices in the set are adjacent. A connected dominating set (CDS) S of a given graph G is a dominating set whose induced sub-graph, denoted $< S >$, is connected, and a minimum CDS (MCDS) is a CDS with the minimum cardinality. A MCDS forms a virtual backbone in the network graph by which the routing overhead can be significantly reduced, where the number of hosts responsible for the route discovery and data transmission can be reduced to the number of vertices in the MCDS of the network topology graph. The MDS and MCDS problems have been shown to be NP-Hard [1], and even for a unit disk graph, the problem of finding a MCDS is still NP-Hard.

In all existing methods proposed for CDS formation in vertex-weighted graphs, it is assumed that the graph is deterministic and thus the weight of its vertices fixed. This assumption can not hold true in almost all the applications in wireless ad hoc networks, so in this paper, the MCDS problem in stochastic graphs is first introduced, and then a learning automata-based approximation algorithm is proposed for solving this problem, when the probability distribution function of the weight of the vertices is unknown. The proposed algorithm significantly reduces the number of samples needs to be taken from the stochastic graph to find the MCDS. It is shown that the MCDS is found with a probability as close as to unity by proper choice of the parameters of the proposed algorithm. To evaluate the performance of the proposed algorithm, the number of samples needs to be taken by it from the weighted vertices of the stochastic graph is compared to that of the standard sampling method. The simulation results show the efficiency of the proposed algorithm in terms of the number of samplings. The reminder of the paper is organized as follows. In the next section, the learning automata and stochastic graphs are described. In section III, the proposed learning automata-based algorithm is presented. The performance of the proposed algorithm is evaluated through the simulation experiments in section IV. Section V concludes the paper.

II. LEARNING AUTOMATA AND STOCHASTIC GRAPH

A. Learning Automata

A learning automaton [5-6] is an adaptive decision-making unit that improves its performance by learning how to choose the optimal action from a finite set of allowed actions through repeated interactions with a random environment. Learning automata can be classified into two main families: fixed structure learning automata and variable structure learning automata. Variable structure learning automata are represented by a triple $< \underline{\beta}, \underline{\alpha}, T >$, where $\underline{\beta}$ is the set of inputs, $\underline{\alpha}$ is the set of actions, and T is learning algorithm. The learning algorithm is a recurrence relation which is used to modify the action probability vector. Let $\alpha(k)$ and $\underline{p}(k)$ denote the action chosen at instant k and the action probability vector on which the

978-0-7695-3595-1/09 $25.00 © 2009 IEEE
DOI 10.1109/ICIME.2009.133

chosen action is based, respectively. The recurrence equation shown by (1) and (2) is a linear learning algorithm by which the action probability vector $\underline{p}$ is updated. Let $\alpha_i(k)$ be the action chosen by the automaton at instant k.

$$p_j(n+1) = \begin{cases} p_j(n) + a[1 - p_j(n)] & j = i \\ (1-a)p_j(n) & \forall j \quad j \neq i \end{cases} \quad (1)$$

When the taken action is rewarded by the environment (i.e., $\beta(n) = 0$) and

$$p_j(n+1) = \begin{cases} (1-b)p_j(n) & j = i \\ (b/r-1) + (1-b)p_j(n) & \forall j \quad j \neq i \end{cases} \quad (2)$$

When the taken action is penalized by the environment (i.e., $\beta(n) = 1$). r is the number of actions can be chosen by the automaton, $a(k)$ and $b(k)$ denote the reward and penalty parameters and determine the amount of increases and decreases of the action probabilities, respectively.

B. Stochastic graph

A vertex-weighted graph can be described by a triple $G =< \mathbf{V}, \mathbf{E}, W >$, where $V = \{v_1, v_2, ..., v_n\}$ denotes the vertex set, $\mathbf{E} \subseteq \mathbf{V} \times \mathbf{V} = \{e_1, e_2, ..., e_m\}$ denotes the edge set and $W = \{w_1, w_2, ..., w_n\}$ denotes the set of vertex weights such that $W(v_i) = w_i$, for all $i \in \{1,2,...,n\}$. In most scenarios, the weight of the vertices is assumed to be fixed, but this is not always true and it varies with time. If the weight of the vertices in a graph is stochastic, then it is a stochastic vertex-weighted graph, hereafter in this paper is called stochastic graph, in which w_i for all $i \in \{1,2,...,n\}$ denotes the probability distribution function of vertex v_i. Let $G =< \mathbf{V}, \mathbf{E}, W >$ and $S = \{\delta_1, \delta_2, ..., \delta_r\}$ be a stochastic graph and the set of all its CDSs, respectively. If $\overline{W}(v_i)$ and $\overline{W}(\delta_i) = \sum_{v_i \in \delta_i} \overline{W}(v_i)$ denote the expected weight of vertex v_i and the expected weight of the connected dominating set δ_i, respectively, therefore the stochastic connected dominating set $\delta^* \in S$ of graph G is minimum, if and only if $\overline{W}(\delta^*) = \min_{\forall \delta_i \in S} \overline{W}(\delta_i)$. That is, The MCDS of a given stochastic graph G is defined as the stochastic CDS with the minimum expected weight.

III. The Proposed MCDS Formation Method

In this section, an intelligent algorithm based on learning automata is proposed for solving the MCDS problem in stochastic graphs, as described in section 1. It is assumed that the weights of the vertices of the graph are random variables and the parameters of the underlying probability distribution of the vertex weight are unknown. So they need to be estimated by a statistical method. A network of learning automata isomorphic to the stochastic graph is formed by assigning to each vertex of the graph a learning automaton. The resulting network of automata can be described by a triple $< \underline{A}, \underline{\alpha}, W >$, where $\underline{A}$ denotes the set of the learning automata, $\underline{\alpha}$ denotes the set of actions in which α_i defines the set of actions can be taken by learning automata A_i, for each $\alpha_i \in \underline{\alpha}$, and $W = \{w_1, ..., w_n\}$ denotes the set of weights such that w_i (for all $i \in \{1,...,n\}$) is the random weight associated with automata A_i. The action set of the learning automaton assigned to each vertex v_i of the stochastic graph, referred to as α_i, has two actions α_0 and α_1. Choosing the action α_0 declares the vertex v_i as a dominatee vertex and choosing the action α_1 declares it as a dominator vertex. At each stage of the algorithm, the set of vertices which are declared as dominators are candidates may succeed in forming the MCDS. The learning automata iteratively construct the CDSs and update the action probability vectors until they find a near optimal solution to the MCDS problem. The steps of the proposed algorithm are described in more detail below.

Stage k of the proposed algorithm is briefly described as follows:

Step 1. Each automaton A_i (for all $1 \le i \le n$) chooses one of its actions according to its action probability vector. If the chosen action declares the vertex v_i as a dominator, then

- Vertex v_i is added to the set of dominator vertices being selected in this stage.
- Vertex v_i and all its neighbors are added to the set of dominate vertices (if they have not already been added).
- The random weight associated with the selected action, is added to the weight of the selected dominator vertices set.

Step 2. If the sub-graph induced by the selected dominator vertices is a connected sub-graph of G, and the cardinality of the dominatee set is equal to the number of vertices of graph G, then

- The CDS constructed in this stage is specified. Let us assume that δ_i has been selected at stage k.
- The average weight of the CDS δ_i until stage k is computed as

$$\overline{W}(\delta_i^k) = \frac{1}{k_i} \sum_{j=1}^{k_i} W(\delta_i(j)) \quad (3)$$

where k_i denotes the number of times the CDS δ_i is constructed until stage k, and $W(\delta_i(j))$ denotes the weight of the j th sampled CDS δ_i, which is defined as

$$W(\delta_i(j)) = \sum_{v_s \in \delta_i} W(v_s(j)) \quad (4)$$

where $W(v_s(j))$ denotes the weight of the vertex v_s in the j th sample taken from δ_i.

Step 3. All learning automata reward their chosen actions if the average weight of the selected CDS δ_i is less than or equal to the dynamic threshold T_{k-1}, and penalize them otherwise. Each learning automaton updates its action probability vector using L_{R-I} reinforcement scheme. At each stage $k > 1$, the value of the dynamic threshold T_k is calculated as

$$T_k = \frac{1}{r}\sum_{i=1}^{r} \overline{W}(\delta_i^k) \quad (5)$$

where r denotes the number of all CDSs of G.

Step 4. The process of constructing the CDSs and updating the action probabilities are repeated until the product of the probability of choosing the vertices of the constructed CDS called PCDS is greater than a certain threshold or the number of constructed CDS exceeds a pre-specified threshold. The CDS which is formed last before stopping the algorithm is the CDS with the minimum expected weight among all CDSs of the stochastic graph.

IV. Experimental Results

To study the performance of the proposed algorithm, we have conducted a group of simulation on two stochastic graphs borrowed from [2] and shown in figures 1 and 2. In all experiments presented in this paper, the number of samples taken by the proposed algorithm from the stochastic graph to construct the optimal CDS is compared with that of the standard sampling method quoted in appendix A. The updating scheme for action probability vectors is L_{R-I}, and each algorithm is terminated when the probability of the constructed CDS (*PCDS*) is 0.95 or greater, or the number of constructed CDSs exceeds a pre-defined threshold. In these experiments, the proposed algorithm is tested on graphs 1 and 2 and the results are summarized in tables II and 4 for the different values of learning rate a which varies from 0.01 to 0.10. Then, the obtained results in terms of the number of samples taken from the graph are compared with those of the standard sampling method given in tables I and III. The aim of the standard sampling method used in our experiments is to obtain the number of samples needs to be taken from the stochastic graph such that the average weight of the samples taken from the optimal CDS be in the interval $(\mu-\delta, \mu+\delta)$ with probability $1-\varepsilon$, where ε is the error parameter and δ is a small enough positive number. Graph1 which is shown in figure 1 has 8 vertices, 12 edges and its minimal CDS is $\delta_1^* = \{v_3, v_6\}$, and graph2 which is shown in figure 2 has 10 vertices, 21 edges and its minimal CDS is $\delta_3^* = \{v_3, v_7\}$. The probability distributions of the vertex weight of these two stochastic graphs are given in tables shown in figures 1 and 2. The simulation results given in tables II and IV are averaged over 5000 runs and the first column of these tables includes the average size of the MCDS. The second and third columns include the total number of samples taken from the graph and from the MCDS, respectively. The fourth column includes the number of constructed CDSs. The fifth column includes the average weight of the MCDS, and the last column of these tables includes the percentage of the converged runs (the percentage of runs converged to the MCDS).

According to the standard sampling method, to obtain a confidence level $1-\varepsilon$ for the CDS, we need to build a confidence with level $1-\varepsilon_i$ for each vertex v_i such that $\sum_{i=1}^{k} \varepsilon_i = \varepsilon$. We assume that the vertices of the stochastic graph all have the same confidence level $1-\varepsilon_0$. Therefore, selecting $\varepsilon_0 = \varepsilon/k$, where k denotes the cardinality of the CDS, guarantees a confidence level $1-\varepsilon$ for the CDS. For instance, when a confidence level 80% is desired for the CDS of graph 1, we need to build a confidence with level 90% for each of the 8 vertices [4]. The results of the standard sampling method for graphs 1 and 2 are given in tables I and III, respectively.

The experiments have shown that the total number of converged runs (convergence rate) and the total number of samples taken from the stochastic graph and the MCDS increases as the learning rate of the proposed algorithm decreases. The average weight of the constructed CDS also converges to the average weight of the optimal CDS as the learning rate decreases. For instance, the average weight of the CDS constructed by the proposed algorithm is exactly the average weight of the optimal CDS when the selected learning rate is less than 0,020 (see tables IV). Comparing the results given in table I with table II and table III with table IV, we find that in all experiments the number of samples taken by the proposed algorithm is much less than that of the standard sampling method.

V. Conclusion

In this paper, a learning automata-based algorithm is proposed to solve the MCDS problem in a stochastic graph when the probability distribution functions of the weight of the vertices are unknown. As the algorithm proceeds, the process of sampling from the graph is concentrated on the vertices by which the MCDS is formed, and by a proper choice of the parameters for the algorithm, the probability with which the given algorithm find the optimal CDS is

close enough to unity. The simulation results show the efficiency of the proposed algorithm in terms of the number of samplings.

VI. Appendix-Description of the Standard Sampling Method

Theorem 1. To obtain a confidence level not smaller than $1-\varepsilon$ for the CDS, it is sufficient to build a confidence with level $1-\varepsilon_i$ for every vertex v_i such that $\sum_{i=1}^{k}\varepsilon_i=\varepsilon$, where k denotes the cardinality of the CDS.

Proof. The required sample size for each of the vertices to satisfy a confidence level $1-\varepsilon_i$ is obtained by using the vertex sampling method described below.

Vertex sampling method Let $(x_1,x_2,...,x_N)$ be a random sample of a random variable X having unknown mean μ and variance σ^2. If $\bar{x}\pm\sigma/\sqrt{N\varepsilon_i}$, where $\bar{x}=\frac{1}{N}\sum_{j=1}^{N}x_j$, is a $1-\varepsilon_i$% confidence interval for mean μ, then for each sufficiently small value of δ, there exists a positive number N_0 such that

$$p\{|\bar{x}_N-\mu|\prec\delta\}\succ 1-\varepsilon_i \qquad (A.1)$$

for all $N\geq N_0$.

The problem is then to find a confidence region for each of the vertices under which a desired confidence level $1-\varepsilon$ is guaranteed for the CDS. The confidence region for the CDS is defined as the intersection $\bigcap_{i=1}^{k}C_i(\varepsilon_i)$ of the confidence regions for the vertices, where $C_i(\varepsilon_i)=1-\varepsilon_i$ denotes the confidence region for vertex v_i and k denotes the cardinality of the CDS. Using Booles-Bonferroni inequality [3], we have

$$Min_{1\leq i\leq k}(1-\varepsilon_i)>p(\mu_i\in\bigcap_{i=1}^{k}C_i(\varepsilon_i))\geq 1-\sum_{i=1}^{k}p[\mu_i\notin C_i(\varepsilon_i)] \qquad (A.2)$$

and so

$$Min_{1\leq i\leq k}(1-\varepsilon_i)>p(\mu_i\in\bigcap_{i=1}^{k}C_i(\varepsilon_i))\geq 1-\sum_{i=1}^{k}\varepsilon_i \qquad (A.3)$$

Hence, the confidence level of the CDS is not smaller than $1-\sum_{i=1}^{k}\varepsilon_i$. In this theorem, the objective is to obtain a confidence level not smaller than $1-\varepsilon$ for the CDS. To achieve this, according to the Bonferroni Correction [4] it is sufficient to build a confidence with level $1-\varepsilon_i$ for each vertex v_i such that $\sum_{i=1}^{k}\varepsilon_i=\varepsilon$, and hence the proof of the theorem■.

References

[1] B. N. Clark, C. J. Colbourn, and D. S. Johnson, "Unit Disk Graphs, " Discrete Mathematics, Vol. 86, pp. 165-177. 1990.

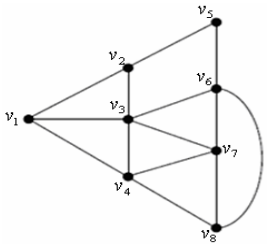
[2] K. R. Hutson, and, D. R. Shier, "Minimum Spanning Trees in Networks with Varying Edge Weights," Annals of Operations Research, Springer, Vol. 146 pp. 3-18, 2006.

[3] F. B. ALT, "Bonferroni Inequalities and Intervals," in Encyclopedia of Statistical Sciences," Vol. 1, pp. 294–301, 1982.

[4] C. E. Bonferroni, "Teoria Statistica Delle Classi e Calcolo Delle Probabilit`a," Pubblicazioni del R Istituto Superiore di Scienze Economiche e Commerciali di Firenze, Vol. 8, pp. 3–62, 1936.

[5] K. S. Narendra and K. S. Thathachar, "Learning Automata: An Introduction," New York, Printice-Hall, 1989.

[6] M. A. L. Thathachar and B. R. Harita, "Learning Automata with Changing Number of Actions," IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMG17, 1987, pp. 1095-1100.



Vertex	*Weight*	*Probability*
v_1	{2, 8, 12}	{0.9, 0.08, 0.02}
v_2	{10, 24, 35}	{0.85, 0.12, 0.03}
v_3	{6, 18, 24}	{0.88, 0.1, 0.02}
v_4	{12, 22, 30}	{0.85, 0.11, 0.04}
v_5	{17, 35, 50}	{0.75, 0.2, 0.05}
v_6	{3, 7, 10}	{0.68, 0.25, 0.07}
v_7	{4, 19, 15}	{0.75, 0.14, 0.11}
v_8	{5, 10, 12}	{0.65, 0.23, 0.12}

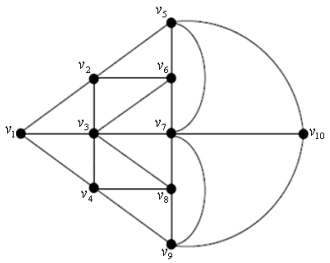
Figure1. Stochastic graph 1 and its probability distribution function

TABLE I. THE NUMBER OF SAMPLES TAKEN FROM GRAPH 1 IN STANDARD SAMPLING METHOD

Vertex	*Confidence Level for CDS*										
	0.5	*0.6*	*0.7*	*0.75*	*0.8*	*0.85*	*0.9*	*0.925*	*0.95*	*0.975*	*0.99*
v_1	317	286	259	308	243	282	269	346	310	448	369
v_2	521	518	541	466	474	468	501	517	608	454	450
v_3	353	337	373	333	381	354	353	421	415	492	465
v_4	406	395	345	377	418	309	448	309	436	391	491
v_5	590	697	642	685	630	808	753	631	610	729	699
v_6	340	265	304	273	314	241	315	356	336	371	376
v_7	311	320	277	312	315	328	377	330	354	392	339
v_8	333	369	377	395	377	471	388	423	376	387	439
Total	3171	3187	3118	3149	3152	3261	3404	3333	3445	3664	3628

TABLE II. THE AVERAGE NUMBER OF SAMPLES TAKEN FROM GRAPH 1 AND MCDS

Learning Parameter	*Average Size of MCDS*	*Graph Sampling Rate*	*MCDS Sampling Rate*	*Number of CDSs*	*Average Weight of MCDS*	*Convergence Rate*
0.010	2.00	975.01	142.91	106.00	12.06	100.00
0.020	2.00	491.82	65.78	87.20	12.08	100.00
0.030	2.00	328.42	42.75	74.02	12.10	99.60
0.040	2.01	249.93	31.64	64.88	12.20	98.94
0.050	2.04	200.84	25.32	56.74	12.38	96.02
0.060	2.08	169.00	21.97	51.37	12.81	92.48
0.070	2.14	145.25	21.07	46.19	12.98	88.76
0.080	2.22	130.68	20.50	42.18	13.69	82.14
0.090	2.27	114.30	18.81	38.57	14.07	76.52
0.100	2.37	106.33	18.82	36.21	14.95	69.47



Vertex	*Weight*	*Probability*
v_1	{2, 8, 12}	{0.9, 0.08, 0.02}
v_2	{10, 24, 35}	{0.85, 0.12, 0.03}
v_3	{6, 18, 24}	{0.88, 0.1, 0.02}
v_4	{12, 22, 30}	{0.85, 0.11, 0.04}
v_5	{17, 35, 50}	{0.75, 0.2, 0.05}
v_6	{3, 7, 10}	{0.68, 0.25, 0.07}
v_7	{4, 19, 15}	{0.75, 0.14, 0.11}
v_8	{5, 10, 12}	{0.65, 0.23, 0.12}
v_9	{10, 19, 24}	{0.80, 0.14, 0.06}
v_{10}	{18, 27, 36}	{0.94, 0.05, 0.01}

Figure2. Stochastic graph 2 and its probability distribution function

TABLE III. THE NUMBER OF SAMPLES TAKEN FROM GRAPH 2 IN STANDARD SAMPLING METHOD

Vertex	*Confidence Level for CDS*										
	0.5	*0.6*	*0.7*	*0.75*	*0.8*	*0.85*	*0.9*	*0.925*	*0.95*	*0.975*	*0.99*
v_1	317	286	259	308	243	282	269	346	310	448	369
v_2	521	518	541	466	474	468	501	517	608	454	450
v_3	353	337	373	333	381	354	353	421	415	492	465
v_4	406	395	345	377	418	309	448	309	436	391	491
v_5	590	697	642	685	630	808	753	631	610	729	699
v_6	340	265	304	273	314	241	315	356	336	371	376
v_7	311	320	277	312	315	328	377	330	354	392	339
v_8	333	369	377	395	377	471	388	423	376	387	439
v_9	242	250	300	343	312	371	339	294	328	363	433
v_{10}	386	425	465	480	485	446	454	464	437	439	502
Total	3802	3867	3887	3977	3953	4084	4201	4096	4214	4469	4568

TABLE IV. THE AVERAGE NUMBER OF SAMPLES TAKEN FROM GRAPH 2 AND MCDS

Learning Parameter	*Average Size of MCDS*	*Graph Sampling Rate*	*MCDS Sampling Rate*	*Number of CDSs*	*Average Weight of the MCDS*	*Convergence Rate*
0.010	2.00	2147.15	914.71	296.88	13.84	100.00
0.020	2.00	1085.43	444.80	202.96	13.84	100.00
0.030	2.02	728.56	288.58	154.60	13.91	97.90
0.040	2.05	550.96	213.50	126.37	14.08	95.85
0.050	2.10	448.99	171.01	107.41	14.33	90.80
0.060	2.20	376.12	147.65	91.80	14.92	82.85
0.070	2.32	325.60	132.57	80.93	15.87	74.05
0.080	2.45	286.68	118.56	71.56	16.65	65.50
0.090	2.52	254.28	106.26	64.10	17.54	62.20
0.100	2.71	231.63	108.49	56.59	18.96	50.00