Maximizing Lifetime of CDS-Based Wireless Ad Hoc Networks

Xiaohui Wei, Yuanyuan Liu, and Xin Bai
College of Computer Science and Technology, Jilin University, Changchun 130012, China
E-mail: weixh@jlu.edu.cn, liuyuanyuan1993jl@yahoo.com, baixin13@mails.jlu.edu.cn

Abstract —Currently, Connected Dominating Sets (CDS) have been widely used to serve as virtual backbones for the topology control of wireless ad hoc networks. Lifetime is one of the most important characteristics of a wireless ad hoc network since mobile nodes are equipped with energy-limited batteries. In this paper, we propose two CDS construction algorithms, CDS-LL and Extended CDS-LL (E-CDS-LL), to prolong lifetime of a CDS-based wireless network. The CDS-LL algorithm trades CDS size for lifetime by a tunable parameter λ. When λ moves from 1 to 0, a CDS will be constructed with more energy-rich nodes. In the simulation, we show that when λ is 0.8, the CDS-LL algorithm increases lifetime at least 6 times through sacrificing around 8% of size. Moreover, since CDS nodes consume more energy than non-CDS nodes, the E-CDS-LL algorithm balances energy consumption of wireless nodes via dynamically selecting energy-rich nodes with the CDS-LL algorithm to reconstruct the CDS before the CDS is disabled. Simulation results show that network lifetime can be prolonged by 66% on average through the E-CDS-LL algorithm compared with the CDS-LL algorithm.

Index Terms—Wireless Ad Hoc Networks, connected dominating set, lifetime

I. Introduction

A wireless ad hoc network consists a number of wireless nodes usually equipped with limited energy. As an autonomous and multi-hop network without fixed or predefined infrastructure, a wireless ad hoc network is easy for these low-cost wireless nodes to implement rapid wireless communication and deploy it on many application areas, such as traffic control, environmental monitoring, military communication, and so forth [1]. Since battery recharging and replenishing are difficult, the power source of a node in a wireless ad hoc network is limited. When the size of the network grows, routing of a wireless network becomes complicated and energy efficiency gets worse. Therefore, the Virtual Backbone (VB) technology is widely used for efficient routing and energy consumption. A VB is a subset of nodes which are responsible for performing data communication tasks of the entire wireless network.

Currently, Connected Dominating Sets (CDS) have been used to serve as VBs. A wireless network is usually modeled as a graph G=(V, E), where V represents all the nodes in the network and E represents the edge set linking nodes. A connected dominating set of a graph G is a connected subset of nodes in V which are called dominators. Each node not in the CDS is denoted as a dominatee which is adjacent to at least one dominator of the CDS. When dominatee A wants to send a message to dominatee B, it first sends the message to its dominator. The CDS is responsible for delivering the message to the dominator of dominatee B. Finally dominatee B gets the message from its dominator. A CDS node usually consumes more energy than a non-CDS node because a non-CDS node can turn on a sleep mode with little energy consumption when it is idle.

Since a VB with a smaller size achieves less overhead for communications and suffers less from interference, previous works focus extensively on obtaining a small VB for more efficient energy consumption and prolonging the lifetime of a wireless network. For CDS-based VBs, this is a Minimum Connected Dominating Set (MCDS) problem which has been well studies. However, we argue that the lifetime of a wireless ad hoc network relies on not only the energy consumption efficiency, but also the energy consumption balancing. For example, if nodes with little remaining energy are selected to be CDS nodes that consume energy faster, these nodes will be disabled when other nodes still remain a lot of energy. Then the wireless network may collapse because of the losing of these nodes. Balancing the energy consumption of nodes can make better use of energy more reasonable and bring extra lifetime of the network. Therefore, in scenarios where network lifetime plays the most critical role rather than backbone size, lifetime-aware CDS algorithms are promising. We propose two CDS construction algorithms in this paper, the CDS with Longer Lifetime (CDS-LL) algorithm and the Extended CDS-LL (E-CDS-LL) algorithm, to prolong lifetimes of these CDS-based wireless ad hoc networks:

1) CDS-LL algorithm. The CDS-LL algorithm is one of the two-phased algorithms which are the majority of current CDS construction algorithms. These two-phased algorithms generate a Maximal Independent Set (MIS) as a dominating set in the first phase. Then

Manuscript received November 9, 2015; revised April 15, 2016.
Paper supported by the National Natural Science Foundation of China (NSFC) (Grant No.61170004, 41304083), Specialized Research Fund for the Doctoral Program of Higher Education (20130061110052), Key Science and Technology Research Project of Science and Technology Department of Jilin Province (20140204013GX).
Corresponding author email: baixin13@mails.jlu.edu.cn;
doi:10.12720/jcm.11.4.365-373

some additional nodes are added to connect the MIS in the second phase to construct the CDS. We present both a centralized version and a distributed version of the CDS-LL algorithm. In addition to a small size, the CDS-LL algorithm can select energy-rich nodes for the CDS to prolong the lifetime of the CDS in both of these two phases. The CDS-LL algorithm trades size for lifetime by a tunable parameter λ ($0 \leq \lambda \leq 1$). When λ is 1, the CDS-LL algorithm only concerns about generating a minimum size CDS. When λ moves from 1 to 0, more energy-rich nodes are included to construct a CDS for a longer lifetime. Simulation results show that when λ is close to 0.8, the CDS-LL algorithm generates CDSs with a much better lifetime and little loss of size.

2) E-CDS-LL algorithm. The E-CDS-LL algorithm is an extension of the CDS-LL algorithm which reconstructs a CDS similar to [2]. In each round of this algorithm, the energy-rich nodes are selected to construct a CDS by the CDS-LL algorithm. After the network runs for a certain amount of time, there may be lesser energy remains in the CDS nodes since they consume more energy than the non-CDS nodes. If we reconstruct the CDS until the CDS is disabled, some nodes would be disabled which are prejudicial to the energy consumption balancing of the network. Therefore, the E-CDS-LL algorithm dynamically reconstructs the CDS before it is disabled to balance the energy of nodes and prolong the lifetime of the network.

II. Relate Works

A. Model

In order to study the CDS construction problem, a lot of efforts have been made on the Unit Disk Graphs (UDG) model [3], [4] to abstract two-dimensional homogenous wireless ad hoc networks. A UDG assumes nodes of a wireless network as disks with the same radius. Also some other models [5], [6] such as Unit Ball Graphs (UBG), Disk Graph with Bidirectional links (DGB), Ball Graph with Bidirectional links (BGB) have been proposed to represent three-dimensional homogenous wireless networks, two-dimensional heterogeneous wireless networks and three-dimensional heterogeneous wireless networks respectively. The CDS-LL algorithm and the E-CDS-LL algorithm are able to be used on all these undirected graph models, but the performance evaluations of the algorithms are mainly given on the UDG model in this paper.

B. MCDS Approximation Algorithms

The MCDS problem has been proved NP-hard in UDGs [1]. Hence, heuristic algorithms are often used for the CDS construction problem. MCDS algorithms can be divided into two categories: centralized algorithms and decentralized algorithms. Decentralized algorithms are more practical for requiring no topology information of the entire network. Decentralized algorithms can be further divided into distributed algorithms and localized algorithms. Localized algorithms generate a CDS with only h-hop neighboring information and perform better scalability and fault tolerance.

The first distributed CDS construction algorithm under UDG model was proposed by Wan *et al*. [7] which has the constant approximation ratio of 8 and consists of two phases. In the first phase, it finds a MIS by constructing a spanning tree. And then secondly it adds more nodes to connect all nodes in MIS. The approximation ratio analysis of the MIS-based algorithms relies on the lower bound of MISs in UDGs. Wan *et al*. in [7] gave an upper bound of (4opt+1) for UDGs, where opt is the size of a MCDS. Wu Weili. in [8] improved this result to (3.8opt+1). As we know, the best-known upper bound for MISs was proposed by M. Li in [9] as (3.4306opt+4.8185). With this result, an algorithm with approximation ratio 6.075 was provided in UDGs.

C. Other Algorithms

In addition to study the MCDS problem for smaller size, some CDS construction algorithms focus on other parameters of a CDS such as diameter, Average Backbone Path Length (ABPL), load balancing and fault tolerance. The diameter of a given connected graph is the length of the longest shortest path between a pair of nodes in the graph [10]. Kim Donghyun in [2] presented two centralized algorithms with constant approximation ratios and bounded diameters. ABPL of a CDS is the sum of the hop distances between any pair of CDS nodes x and y divided by the number of all the possible pair of nodes [10]. In [11]-[13], Ding et al. proposed several Minimum routing cost CDS(MOC-CDS) algorithms for better routing path length. With these algorithms, the average length of routing paths reduces significantly compared to regular CDSs. He J.S. in [14] taken not only the size, but also the load balancing factors into account which balances traffic load on each backbone node. K-connected m-dominating sets [15], [16] are used for fault tolerance and maintenance of CDSs. A k-connected m-dominating sets ensures that between any pair of dominators there exists at least k different paths and each dominates has at least m adjacent dominator neighbors. In this way, it can provide a frame for constructing a more flexible and robust virtual backbone set to fit the dynamic adjustment of network nodes. But it can surely result in the inevitable increase of nodes in CDS.

D. Remarks

Although previous algorithms optimize different parameters of a CDS, they ultimately aim for better energy efficiency and longer lifetime of the wireless network at the same time.

Unfortunately, prior works performs unsatisfied on balancing of wireless nodes energy consumption. For example, in algorithms proposed in [7]-[12], there are no

assumptions considered about the remaining energy of nodes. That is, a node may be selected as a backbone node even if it is about to be out of power. Thus, an ad hoc network becomes invalid fast when there are still lots of energy-rich nodes. The CDS-LL algorithm and the E-CDS-LL algorithm make best use of energy on all nodes to maximize the lifetime of the network.

The rest of this paper is organized as follows: Section 3 and Section 4 introduces the CDS-LL algorithm and the E-CDS-LL algorithm respectively. In section 5 we give some simulation results of the algorithms. Finally, section 6 describes the conclusions of this paper.

III. The CDS-LL Algorithm

In this section, we first proposed the definition of CDS lifetime. To improve the lifetime of a CDS, the centralized CDS-LL algorithm is devised. Then a distributed version of the CDS-LL algorithm is presented. We also theoretically analyze the performance of the algorithms.

A. Centralized Version of the CDS-LL Algorithm

When a node in the CDS is disabled, the entire CDS is disabled. Therefore, we define the lifetime of a CDS as Definition 1:

Definition 1 (CDS lifetime). The lifetime of a CDS is the minimum lifetime of the node in it.

If a CDS is disabled, the overhead of reconstructing the wireless network is high. Prolonging the lifetime of a CDS decreases reconstruction times and extends lifetime of the network.

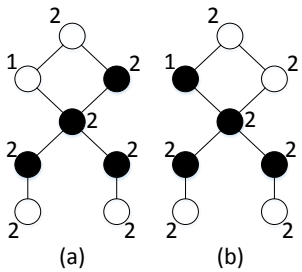


Fig. 1. Two CDSs with the same size and different lifetime. The black nodes form a CDS of the wireless network. The number next to each node represents its remaining lifetime.

Algorithms considering only size may generate a CDS with bad lifetime (Fig. 1(b)). In Fig. 1, size of the CDS in Fig. 1(a) and Fig. 1(b) are both 4 but their lifetimes are 2 and 1. The motivation of the CDS-LL algorithm is to devise an algorithm generates CDSs with longer lifetimes and acceptable sizes.

TABLE I. Notations for the CDS-LL Algorithm

$N(u)$	Neighbors of node *u*
$D(u)$	Degree of node *u*
$C(u)$	Number of black-blue components adjacent to *u*
$E(u)$	Remaining energy of node *u*

Table I lists some notations for the CDS-LL algorithm used in this paper:

Algorithm 1. CDS-LL

1: INPUT: A UDG G=(V, E), all nodes are white
2: OUTPUT: A CDS
3: I=Φ;B=Φ; $\lambda \in [0,1]$
4: **WHILE** $V \neq \phi$ **DO**
5: find a node with the biggest , color *u* black, and add *u* into the set *I*.
6: color all white nodes that belongs to gray
7: $V = V - \{u\} - N(u)$
8: **END WHILE**
9: **WHILE** $G(I \cup B)$ is disconnected **DO**
10: **IF** there is a gray node $b \in V$, that $C(b) > 1$
11: select a gray node *b* with the biggest $C(b)^{\lambda} E(b)^{1-\lambda}$
12: **ELSE**
13: select a gray node *b* with the biggest $E(b)$, and *b* must has a white neighbor that is adjacent to a different black-blue component from *b*.
14: color *b* blue
15: $B = B \cup \{b\}$
16: **END WHILE**
17: **RETURN**

The CDS-LL algorithm is actually a spectrum of algorithms with an tunable parameter $\lambda \in [0,1]$. If λ is closer to 1, the CDS-LL considers more about size of the CDS. Conversely if λ is closer to 0, the algorithm is inclined to constructs a CDS with longer lifetime.

The CDS-LL algorithm is a two-phased algorithm. All nodes are initialized with white color in the algorithm. In the first phase, the CDS-LL algorithm generates a MIS by using $D(u)^{\lambda} E(u)^{1-\lambda}$ as the priority to select nodes. When a node is chosen as a MIS node, the node is colored black. When λ is 1, the MIS is comprised of high degree nodes for a small size. When λ is 0, the MIS is comprised of nodes with more remaining energy for better lifetime. In the second phase, we color blue nodes to connect black-blue components similar to Thai's algorithm in [6]. A black-blue component is a connected component of the subgraph induced only by black and blue nodes, ignoring connections between blue nodes. After the first phase, each black node is a black-blue component. When λ is 1, the white node connected with most black-blue components is colored as a blue node. When λ is 0, the white node with most remaining energy is colored as a blue node. Step fourteen ensures that white nodes connected to the same black-blue component with its white neighbor will not be colored blue because adding this node will only increase the size of the CDS but will not reduce the number of black-blue components.

When λ changes from 1 to 0, both the first phase and the second phase trade numbers of the black or blue nodes for their lifetimes. That is, the CDS-LL algorithm can trade size for lifetime by tuning the parameter λ.

Though the CDS-LL algorithm prolongs the lifetime of a CDS by sacrificing its size, Theorem 1 shows that the CDS-LL algorithm constructs CDSs with bounded sizes. By simulation results, we discover that when λ is between 0.7 and 0.9, the CDS-LL algorithm prolongs a great deal of CDS lifetime by little loss of its size. Thus, the CDS-LL algorithm performs quite competitively on both size and lifetime than algorithms only concerns about size.

Theorem 1. In Algorithm 1, $I \bigcup B$ is a CDS.

Proof. First we prove that *I* is a MIS. After step 8, all nodes are black and gray and black nodes form the set *I*. For any black node *u* in *I*, all neighbors of *u* are gray. Thus, each pair of nodes in *I* are not adjacent so that *I* is an IS. Since any gray node is adjacent to a black node, *I* will be no longer an IS if any gray node is added into *I*. Therefore, *I* is a MIS which is also a DS. Moreover, $I \bigcup B$ is a DS.

Now we prove that $I \bigcup B$ is connected. According to the algorithm, we only need to prove that if $G(I \bigcup B)$ is disconnected then (i) there is a gray node $b \in V$, $C(b) > 1$, or (ii) there is a gray node *b* with a gray neighbor that is adjacent to a different black-blue component from *b*. Since for any gray node *b*, $C(b) \geq 1$, (i) or (ii) holds. (iii) if for any gray node *b*, $C(b)=1$, then there is a gray node *b* with a gray neighbor that is adjacent to a different black-blue component from *b*. When $C(b)=1$, all black-blue components are separated by at least two adjacent gray nodes. When $G(I \bigcup B)$ is disconnected, then there are at least two black-blue components. Hence, (iii) holds. Therefore, $I \bigcup B$ is a CDS.

Theorem 2. Let *C* be the CDS obtained from the CDS-LL algorithm, then $|C| \leq 10.2918opt + 12.4555$.

Proof. Denote *I* as a MIS generated by the CDS-LL algorithm. From [9], $|I| \leq 3.4306opt + 4.8185$ .All the black nodes are connected by at most $2(|I|-1)$ blue nodes. Hence, $|C| \leq 10.2918opt + 14.4555\text{-}2$ $= 10.2918opt + 12.4555$

B. Distributed Version of the CDS-LL Algorithm

In this section, we describe the distributed version of the CDS-LL algorithm. Phase 1 is the procedure of MIS construction and phase 2 connects these MIS nodes. In phase 1, each node firstly marks itself white and maintains an A_PRIORITY list. Upon receiving a WHITE message, each node updates it's A_PRIORITY list by sorting the value of $D(v_i)^{\lambda} E(v_i)^{1-\lambda}$. When a node with its id is at the beginning of the A_PRIORITY list, it becomes black and sends a black message with its id to turn its neighbors gray. Upon a node turns gray , it will send a gary1 message with its own id and black node id included in black message to notify other white nodes to delete id in the gray1 message from their A_PRIORITY list. Finally, all nodes turns black or gray and all black nodes form a MIS.

Algorithm 2. CDS-LL-D

Phase 1:

1: **INPUT:** A UDG G=(V,E).

2: **OUTPUT:** A CDS

3: Each white node v_i marks itself white and broadcasts a WHITE message with item $< ID_i, D(v_i)^{\lambda} E(v_i)^{\lambda} >$.

4: Each white node maintains an A_PRIORITY list to sort the priority of nodes.

5: Find a node v_i with its maximal priority in A_PRIORITY list and marks itself black, send a BLACK message with ID_i to all its neighbors.

6: When a white node v_i receives a BLACK message with ID_j , v_i marks itself gray and broadcasts a GRAY1 message with ID_i and ID_j.

7: When a white node receives a GRAY1 message, it deletes the black and gray nodes from its A_PRIORITY list.

Phase 2:

8: For each gray node, set $CID_i = -1$ and SAME_NEIGHBOR=0. For each black node v_i, set $CID_i = ID_i$ and send a COMPONENT message with CID_i.

9: Each gray node maintains a C_NEIGHBOR list and a B_PRIORITY list. When a gray node receives a COMPONENT message with CID_i, updating its C_NEIGHBOR list by adding CID_i.

10: Each gray node send a GRAY2 message with its C_NEIGHBOR list. When a gray node v_i receives a GRAY2 message that contains same CID in the C_NEIGHBOR list of v_i , set SAME_NEIGHBOR=1.

11: Each gray node v_i broadcasts a GRAY3 message with item: $< ID_i, |C_{NEIGHBOR}|^{\lambda} E(v_i)^{1-\lambda},$ $|C_{NEIGHBOR}|, SAME\_NEIGHBOR >$

12: Upon receiving a GRAY3 message, each gray node update its B_PRIORITY list sorted by the second element of items in the GRAY3 message.

13: v_i marks itself blue and set CID_i to the smallest element in its C_NEIGHBOR list if the following conditions are satisfied:

a) ID_i is at the beginning of its B_PRIORITY list and $C(v_i) > 1$;

b) All items before v_i includes that |C_NEIGHBOR| equals to 1 and |SAME_NEIGHBOR|=1. v_i contains |C_NEIGHBOR| equals to 1 and

|SAME_NEIGHBOR|=0;

Then v_i sends a BLUE1 message with $<CID_i, CID_i>$ to all its neighbors and send a BLUE2 message with ID_i.

14: When a non-gray node v_j receives a BLUE1 message with CID_i, updates CID_j to CID_i and send a BLUE1 message with $<CID_j, CID_i>$ to all its neighbors.

15: When a gray node receives a BLUE2 message with ID_i, update its B_PRIORITY list by deleting item with ID_i.

16: When a gray node v_k receives a BLUE1 message with $<CID_j, CID_i>$, update its C_NEIGHBOR list by adding CID_i and deleting CID_j. If C_NEIGHBOR list contains the same CID with CID_i and its SAME_NEIGHBOR equals to 0, set SAME_NEIGHBOR=1. v_k broadcasts a GRAY3 message as described in the 11th step and send a GRAY4 message with its C_NEIGHBOR list.

17: When a gray node v_i with a SAME_NEIGHBOR=0 receives a GRAY4 message that has a C_NEIGHBOR list containing the same CID as CID in the C_NEIGHBOR list of v_i, set SAME_NEIGHBOR to 1 and broadcast a GRAY3 message as described in the 11th step.

18: If all nodes in the B_PRIORITY list include that |C_NEIGHBOR| equals to 1 and |SAME_NEIGHBOR|=1, then do nothing.

Based on the MIS generated in phase 1, phase2 connects black nodes in a distributed manner. Firstly, each gray node sets a parameter CID initialed to -1 and SAME_NEIGHBOR=0 used to judge the condition in the 14th step in centralized algorithm, and each black node also sets a CID initialed to its own id and sends a component message. Each gray node maintains a C_NEIGHBOR list and a B_PRIORITY list and receives a component message to update its C_NEIGHBOR list by adding the CID of its neighbor black nodes.

After all gray nodes have been initialized, each gray node sends a GRAY2 message to update SAME_NEIGHBOR and a GRAY3 message to update its B_PRIORITY list including the parameter $|C_{NEIGHBOR}|^{\lambda} E(v_i)^{1-\lambda}$. Then each gray node use the conditions described in the 13th step to find which node should mark itself blue and notify other neighbor nodes. From step 14 to step 17, each non-gray node updates its CID to the CID of its neighbor blue node v_i if v_i turns blue in the previous step, each gray node updates its B_PRIORITY list by deleting item of v_i.

Furthermore, if non-gray nodes v_i modifies its CID, the C_NEIGHBOR list of a gray node that includes the CID of v_i should be changed as described in step 16. Then the modified gray nodes re-broadcast GRAY3 message.

Theorem 3. The CDS-LL-D algorithm has an $O(n^2)$ time complexity and $O(n^2)$ message complexity.

Proof. In the first phase, since each node sends a white message and each node needs to sort its own A_PRIORITY list, the time and message complexity are both . $O(n^2)$. In the second phase, because the number of gray nodes is less than *n*, the time and message complexity of the sorting of B_PRIORITY list are $O(n^2)$.

IV. The Extend CDS-LL Algorithm

In a CDS-based wireless ad hoc network, only nodes in the CDS are responsible for relaying messages. The non-CDS nodes can go to a low-powered sleep mode if there are no communication tasks. Therefore, a CDS node usually consumes much more energy than a non-CDS node. Hence, the lifetime of a wireless network can be further prolonged by reconstructing the CDS before the CDS is disabled which balances the energy consumption of nodes. The final mapping state of all physical nodes in a data center can be constructed. Then the problem is how to migrate virtual machines from initial mapping state to get the final mapping stated. A migration plan that causes minimum migration cost is optimal.

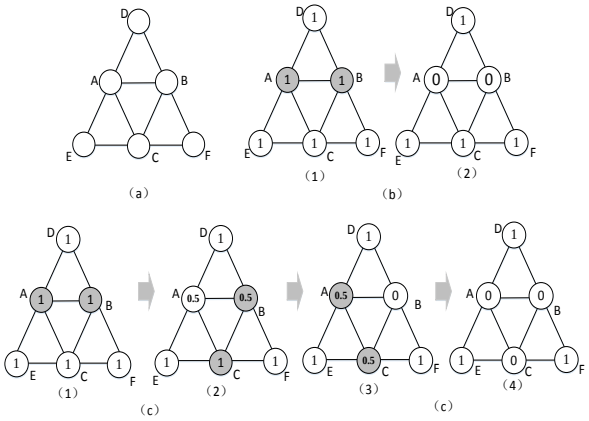


Fig. 2. (a) Any two nodes among A, B and C can work as a CDS in the network. (b) Lifetime of the network is 1 time unit by selecting A and B as a CDS. (c) Lifetime of the network can be 1.5 time unit if the CDS is reconstructed before it is disabled.

Fig. 2 is an example illustrates how the reconstruction works. In Fig. 2(a), any two nodes among nodes A, B and C can form a CDS. Without loss of generality, we suppose a non-CDS nodes consume no energy and a CDS node consume 1 unit energy in 1 time unit. At first, all nodes have 1 unit energy. If we choose A and B as a CDS and run the network until the CDS is disabled, the network will be disabled as Fig. 2(b) after 1 unit time. In Fig. 2(c), we reconstruct the CDS every 0.5 time unit. We first use nodes A and B as a CDS. After the network runs 0.5 time unit, the state of the network is as Fig. 2(c)(2). Then we use node B and C as a CDS. After another 0.5 time unit (Fig. 2(c)(3)), node B is disable but node A and C can still form a CDS. At last we use node A and C as a CDS that the network can still work for 0.5 time unit.

Thus, we can prolong the lifetime of the network from 1 time unit to 1.5 time unit by the reconstruction.

Algorithm 3. E-CDS-LL （G=(V,E)）

1:. Execute CDS-LL to construct a CDS. T_c denotes the lifetime of the CDS.

2:. Run the network for time

$$T_0 = \begin{cases} T_c * R, & if\ T_c > T_{threshold}; \\ T_c, & if\ T_c > T_{threshold}. \end{cases}$$

Where R and $T_{threshold}$ are constants, $R \in (0,1)$.

3:. **IF** G is connected

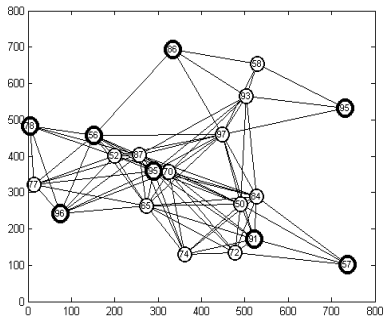
4:. go to step 1

5:. **ELSE**

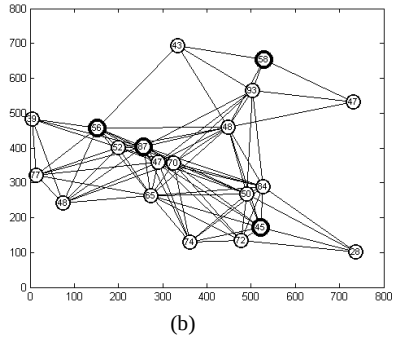
6:. end

In the E-CDS-LL algorithm, we make the reconstruction after the network run for T_0 instead of T_c in case the CDS nodes consume too much energy to be disabled, which affects the connectivity of G. When the remaining energy of nodes are running low and the lifetime of the CDS is small, $T_{threshold}$ guarantees the reconstruction will not happen too frequently. Simulation results in section 5 show that when $R \in (0.3, 0.6)$ and $T_{threshold}$ can be set between 10% and 20% of the initial lifetime of the network, the E-CDS-LL algorithm usually achieves better performance.

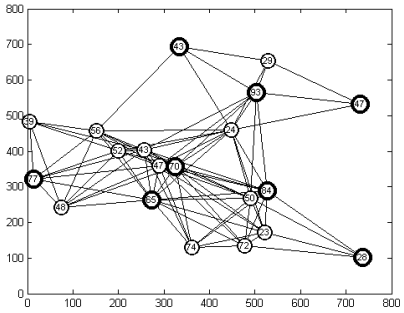
The E-CDS-LL algorithm cannot decrease the energy consumption of nodes. However, since the energy consumption of CDS nodes are much more than the non-CDS nodes, the E-CDS-LL algorithm balances the energy consumption of nodes in the network by dynamically selecting energy-rich nodes as CDS nodes for reconstruction. Thus, the E-CDS-LL algorithm can further prolongs the lifetime of the network than the CDS-LL algorithm.



(a)



(b)



(c)

Fig. 3. An example to illustrate E-CDS-LL algorithm(a) consrtuct a CDS at the first time. (b) reconstrcut a CDS after (a). (c) reconstruct a CDS after (b).

Fig. 3 shows a reconstruction example generated by the E-CDS-LL algorithms in an 800×800 area. There are 20 nodes random constructed to compose a connected graph. The parameter λ is set 0.8 and transmission range is 250. Assuming that we reconstruct a CDS when R is 0.5, from Fig. 3 (a), we can see that the selection of CDS nodes considers both lifetime and the degree of nodes. In Fig. 3 (a), most nodes with larger lifetime can be selected into dominating set. And Fig. 3 (b) and (c) shows that other nodes different from the last time also with relatively larger lifetime composed a CDS. Through choosing nodes with good degrees and large lifetimes as possible, we can effectively balance the energy consumption of nodes.

V. Simulation

In this section, some simulation experiments are conducted to evaluate CDS-LL with different network parameters. In CDS-LL, CDS sizes are traded for lifetime by setting a smaller λ. Therefore, we test the effects of λ with different node numbers, network density and also make a performance comparison with well-known Wan's algorithm [7] in UDG models. Then, some parameters used in E-CDS-LL are also tested to balance the cost of reconstruction and the network lifetime.

A. Effects of λ with Different Node Numbers

To evaluate the effects of λ with different node numbers, we randomly deploy certain number of nodes to a fixed area of 800*800 for UDG. Each node is set a transmission range 200. The lifetime of each node is randomly chosen as an integer from 0 to 100. For each combination of network parameters, we test 1000 times and use the average as the final result.

Fig. 4 shows the lifetimes and CDS sizes where 20, 40 and 60 nodes are deployed. Fig. 4(a) indicates that CDS size obtained from CDS-LL is smaller if a bigger λ is selected. That is because when λ trends to 1, the degrees of nodes is a main factor influencing the sizes of constructed backbone nodes. However, as can be seen from Fig. 4(b), lifetime decreases obviously when λ changes from 0 to 1 in UDG. Especially when λ is closed to 1, lifetime reduces quite fast that the system

would be disabled in a short time if λ equals to 1. And in Fig. 4 (a), there are few changes on CDS size if λ increases from 0.5 to 1. By contrast, an acceptable lifetime and a better CDS size can be obtained if λ is in the range of 0.7 and 0.9 that will prolong lifetime at least 6 times only sacrificing around 8% of size.

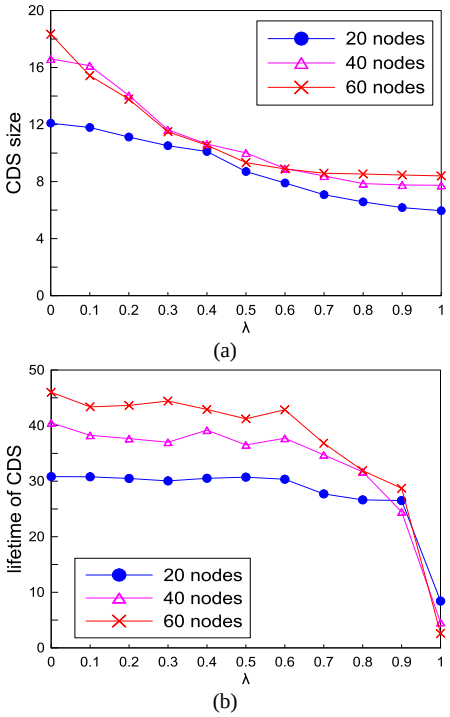


Fig. 4. Effect of λ with different node numbers (a) CDS size (b) lifetime of CDS

B. *Effects of λ with Different Node Densities*

Simulations are also carried out for comparing CDS-LT performance with different network densities. We fix number of nodes to 60 and increased the area size. For UDG, areas of 800*800, 1000*1000, 1200*1200 are tested and areas of 800*800*800, 1000*1000*1000 and 1200*1200*1200 are tested for BGB. Transmission range is set 400 for each node.

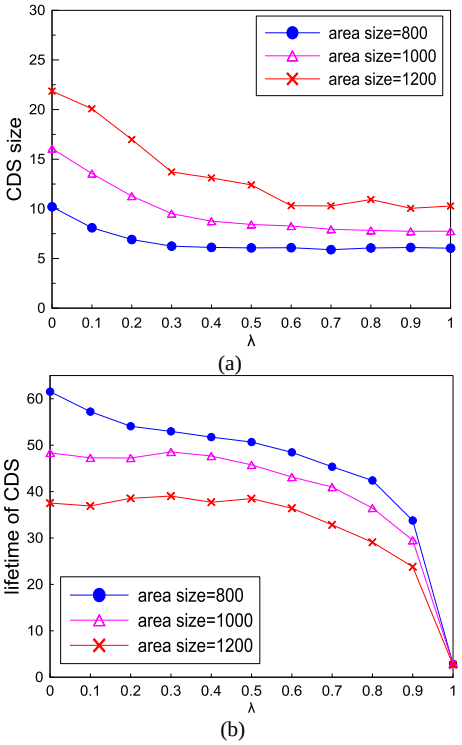


Fig. 5. Effects of λ with different network densities (a) CDS size (b) lifetime of CDS

Fig. 5 also shows an obvious decreasing trend as the same as in Fig. 4 when λ changes from 0 to 1. Therefore, the results can further illustrate the effects of λ on the lifetime of CDS especially set between 0.5 and 1. Fig. 5 indicates that when the network density decreases, CDS size gets bigger and lifetime gets smaller. This is because when the network density decreases, there are less neighbors for each node. To dominate all nodes of the network, CDS size needs to be larger. Since there are more nodes in CDS when network density decreases, nodes with lower lifetime have to be added in the CDS. Therefore, lifetime gets smaller.

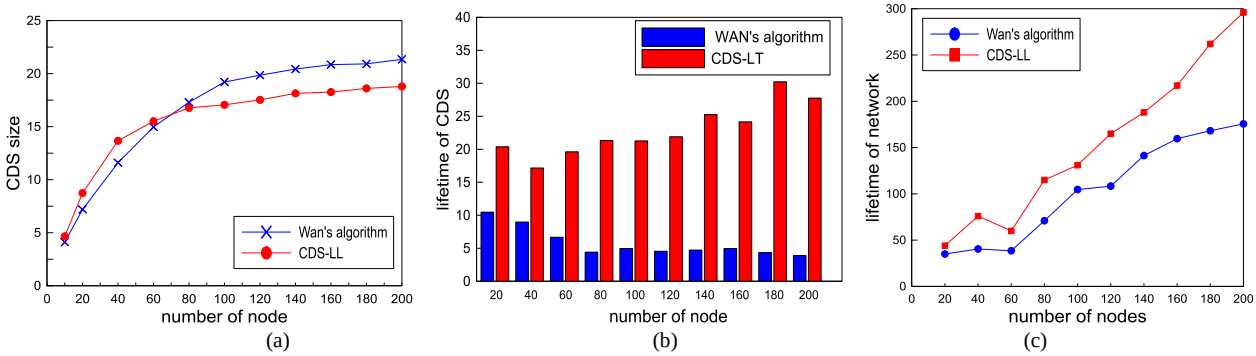


Fig. 6. Performance comparison between CDS-LT and Wan's algorithm (a) Compare the CDS size (b) Compare the lifetime of CDS.(c)Compare the network lifetime

C. *Performance Comparison between CDS-LT and Wan's Algorithms*

To test the size of CDS generated by the new algorithm, we do the simulation to compare the performance with Wan's Algorithms. We fix the area of 800*800 and vary the number of deployed nodes from 20 to 200. Transmission range is set 200 for each node. For each number of nodes, 100 network instances are tested and the results are averaged.

In Fig. 6(a), the size of a CDS obtained from new proposed program is a little larger than Wan's algorithm

[7] when the number of nodes is smaller. As network size increases, a smaller CDS size is computed by CDS-LT compared with Wan's algorithm. Due to the fact that the performance ratio of Wan's algorithm is good in related research, the size of a CDS generated by CDS-LT is acceptable. Moreover, the lifetime of a CDS using these two algorithms has a huge difference as can be seen in Fig. 6(b). The lifetime of a CDS computed by CDS-LT is much longer than that of the Wan's algorithm no matter how many nodes in a wireless network. The results indicate that the CDS-LT algorithm can greatly improve the lifetime of a CDS and meanwhile will not cause too much increased nodes in CDS.

To compare the lifetime of wireless network, we repeatedly execute CDS-LL algorithm to get a total network lifetime. We fix the area of 800*800 and vary the number of deployed nodes from 20 to 200. Transmission range is set 200 for each node. Compared with Wan's algorithm, Fig. 6(c) shows the comparison of the network lifetime with these two algorithms. These results further indicate that our new proposed algorithm can greatly prolong the lifetime of network and it is more energy efficient when the network density increases.

D. Simulation Results of Reconstruction with E-CDS-LL Algorithm

Simulation results provide the performance E-CDS-LL with different $R \in (0,1)$ values in Fig. 7. We set the initial energy value of each node in the network to 100 and tested the lifetime of network and the reconstruction times when the values of R varied from 0 to 0.9.When R initially is 0, the value of reconstruction times is a little larger since that it exactly illustrates the above-mentioned example shown in Fig 3(a). Then, as expected, the network lifetime and reconstruction times of CDS increase consistently when the parameter R increases. In wireless ad hoc networks, the cost of reconstruction cannot be ignored and even is an important effect on network overhead. So the parameter R cannot be set not closely to 1 considering the spending of reconstruction. From these simulation results, the parameter R can be approximately set in range of 0.3 to 0.6. Therefore, the CDS can be reconstructed when the remaining lifetimes of CDS nodes reduce to $R*T_c$ as shown in algorithm 3 to further extend the lifetime of wireless network.

Fig. 8 shows how $T_{threshold}$ in E-CDS-LL affects the times of reconstructing a CDS until the network is unconnected. The result shows that reconstruction times drop significantly when the range of $T_{threshold}$ is 0 to 10 and $T_{threshold}$ should be set larger than 10% of the initial energy in order to guarantee the reconstruction will not happen too frequently. Meanwhile, it cannot be set too large that will lead to more nodes directly running to failure. In Fig. 8, $T_{threshold}$ is set between 0 and 30% of the initial lifetime. In this range, $T_{threshold}$ has no influence on network lifetime showed in our experimental data. Therefore, considering only the number of reconstruction, we present a suggested value about $T_{threshold}$ which can be set between 10% and 20% of the initial lifetime of the network.

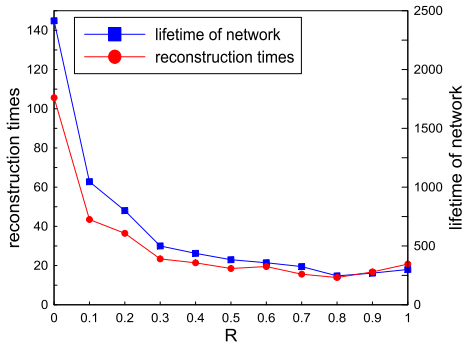


Fig. 7. lifetime of network and reconstruction times with different $R(R \in (0,1))$

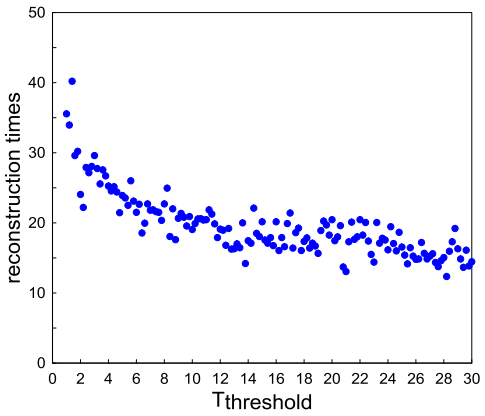


Fig. 8. Effects of $T_{threshold}$ on reconstruction times when $T_{threshold} \in [0,30]$ when the initial lifetime of each node is 100.

VI. CONCLUSIONS

In this paper, we studied the energy efficiency problem of a wireless ad hoc network in a new perspective that is balancing energy consumption of nodes. In a CDS-based wireless network, a CDS node consumes more energy than a non-CDS node. Hence, we have improved the energy efficiency and prolonged the lifetime of a wireless network in two ways: Constructing a CDS with more energy-rich nodes and reconstructing another CDS after CDS nodes work for a certain period and remain less energy. We devised two algorithms, CDS-LL and E-CDS-LL for these ideas. The CDS-LL algorithm gives a network designer a choice, generating a CDS with smaller size or with longer lifetime, by a tunable parameter λ . The E-CDS-LL algorithm extends the CDS-LL algorithm for better balancing of nodes in the wireless network. We have simulated our algorithms in wireless ad hoc networks abstracted as UDGs. In future work, we hope that the algorithms can be evaluated in real ad hoc networks.

As the most important issue, energy efficiency plays a key role to design a wireless ad hoc network in practice. We hope our algorithms to be available as considerable choices for wireless ad hoc network designers.

ACKNOWLEDGMENT

Paper supported by the National Natural Science Foundation of China (NSFC) (Grant No.61170004, 41304083), Specialized Research Fund for the Doctoral Program of Higher Education (20130061110052), Key Science and Technology Research Project of Science and Technology Department of Jilin Province (20140204013GX).

REFERENCES

[1] D. Z. Du and P. J. Wan, *Connected Dominating set: Theory and Applications*, Springer Science & Business Media, 2012, vol. 77.
[2] D. Kim, Y. Wu, Y. Li, F. Zou, and D. Z. Du, "Constructing minimum connected dominating sets with bounded diameters in wireless networks," *IEEE Trans. Parallel and Distributed Systems*, vol. 20, no. 2, pp. 147-157, Jan. 2009.
[3] J. Yu, N. Wang, G. Wang, and D. Yu, "Connected dominating sets in wireless ad hoc and sensor networks–A comprehensive survey," *Computer Communications*, vol. 36, no. 2, pp. 121-134, Jan. 2013.
[4] Y. L. Du and H. W. Du, "A new bound on maximum independent set and minimum connected dominating set in unit disk graphs," *Journal of Combinatorial Optimization*, vol. 30, no. 4, pp. 1-7, Dec. 2013.
[5] Z. Wang, W. Wang, J. M. Kim, B. Thuraisingham, and W. Wu, "PTAS for the minimum weighted dominating set in growth bounded graphs," *Journal of Global Optimization*, vol. 54, no. 3, pp. 641-648, Nov. 2012.
[6] L. Wang, P. J. Wan, and F. Yao, "Minimum cds in multihop wireless networks with disparate communication ranges," *IEEE Trans. Mobile Computing*, vol. 12, no. 5, pp. 909-916, Mar. 2013.
[7] P. J. Wan, K. M. Alzoubi, and O. Frieder, "Distributed construction of connected dominating set in wireless ad hoc networks," *Mobile Networks and Applications*, vol. 9, no. 2, pp. 141-149, Apr. 2004.
[8] W. Weili, "Minimum connected dominating sets and maximal independent sets in unit disk graphs," *Theoretical Computer Science*, vol. 352, no. 1-3, pp. 1–7, Mar. 2006.
[9] M. Li, P. J. Wan, and F. Yao, "Tighter approximation bounds for minimum CDS in wireless ad hoc networks," *Algorithms and Computation*, vol. 5878, pp. 699-709, Dec. 2009.
[10] K. Shukla and S. Sah, "Construction and maintenance of virtual backbone in wireless networks," *Wireless Networks*, vol. 19, no. 5, pp. 969-984, Jul. 2013.
[11] L. Ding, X. Gao, W. Wu, W. Lee, X. Zhu, and D. Z. Du, "Distributed construction of connected dominating sets with minimum routing cost in wireless networks," in *Proc. IEEE 30th International Conf. Distributed Computing Systems*, 2010, pp. 448-457.
[12] L. Ding, W. Wu, J. Willson, H. Du, and W. Lee, "Efficient virtual backbone construction with routing cost constraint in wireless networks using directional antennas," *IEEE Trans. Mobile Computing*, vol. 11, no. 7, pp. 1102-1112, Jul. 2012.
[13] H. Du, W. Wu, Q. Ye, D. Li, W. Lee, and X. Xu, "CDS-based virtual backbone construction with guaranteed routing cost in wireless sensor networks," *IEEE Trans. Parallel and Distributed Systems*, vol. 24, no. 4, pp. 652-661, Apr 2013.
[14] J. S. He, S. Ji, Y. Pan, and Z. Cai, "Approximation algorithms for load-balanced virtual backbone construction in wireless sensor networks," *Theoretical Computer Science*, vol. 507, pp. 2-16, Oct. 2013.
[15] Y. Li, Y. Wu, C. Ai, and R. Beyah, "On the construction of k-connected m-dominating sets in wireless networks," *Journal of combinatorial optimization*, vol. 23, no. 1, pp. 118-139, Jan. 2012.
[16] N. Ahn and S. Park, "An optimization algorithm for the minimum k-connected m-dominating set problem in wireless sensor networks," *Wireless Networks*, vol. 21, no. 3, pp. 783-792, Sep. 2014.

Xiaohui Wei is a Professor and the Dean of the College of Computer Science and Technology (CCST) at Jilin University. He is currently the Director of High Performance Computing Center of Jilin University. His current major research interests include resource scheduling for large distributed systems, infrastructure level virtualization, large-scale data processing system and fault-tolerant computing. He has published more than 50 journal and conference papers in the above areas.

Yuanyuan Liu is a master and has received her bachelor's degree from the College of Computer Science and Technology, Jilin University in 2013. Her research interests include wireless networks and resource management in cloud computing.

Xin Bai received the BS degree from the Department of Mathematics, Shanghai Jiao Tong University, China, in 2008. He received the MS degree from Department of Computer Science, Jilin University, China, in 2013, where he is currently pursuing the PhD degree. He is currently working at wireless networks.