2015 IEEE 35th International Conference on Distributed Computing Systems

A Novel Approximation for Multi-Hop Connected Clustering Problem in Wireless Sensor Networks

Jun Li*, Xudong Zhu*, Xiaofeng Gao*§, Fan Wu*, Guihai Chen*, Ding-Zhu Du†, Shaojie Tang†

*Shanghai Key Laboratory of Scalable Computing and Systems, Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, 200240, P.R.China
†University of Texas at Dallas, Richardson, TX 75080, USA
lijun2009@sjtu.edu.cn, nongeek.zv@gmail.com, {gao-xf, fwu, gchen}@cs.sjtu.edu.cn, dzdu@utdallas.edu, tangshaojie@gmail.com

***Abstract*—Wireless sensor networks (WSNs) have been widely used in plenty of applications. To achieve higher efficiency for data collection, WSNs are often partitioned into several disjointed clusters, each with a representative cluster head in charge of the data gathering and routing process. Such a partition is balanced and effective if the distance between each node and its cluster head can be bounded within a constant number of hops, and any two cluster heads are connected. Finding such a cluster partition with minimum number of clusters and connectors between cluster heads is defined as *minimum connected d-hop dominating set* (d-MCDS) problem, which is proved to be NP-complete.**

In this paper, we propose a distributed approximation algorithm, named *CS-Cluster*, to address the d-MCDS problem. CS-Cluster constructs a sparser d-hop maximal independent set (d-MIS), connects the d-MIS and finally checks and removes redundant nodes. We prove the approximation ratio of CS-Cluster is $(2d+1)\lambda$, where λ is a parameter related with d but is no more than 18.4. Compared with the previous best result $O(d^2)$, our approximation ratio is a great improvement. Our evaluation results demonstrate the outstanding performance of our algorithm compared with previous works.

I. Introduction

Wireless sensor network (WSN) is a kind of self-organized communication system consisting of many small-sized, inexpensive, and battery-powered sensors. Since such sensors can be remotely deployed in large numbers and operated autonomously in unattended environments, WSNs have been widely used in a lot of applications such as health-care industry, food industry, disaster management, battlefield surveillance, etc. In these applications, the task of each sensor is to collect the information in its surrounding environment and transmit the corresponding data to the base stations of WSNs. Over the past decades, a lot of related researches have studied data gathering and transmission problems [1]–[7].

Because the unattended environments of WSNs generally make it quite difficult to recharge sensor batteries, sensors in such environments are energy constrained. Therefore, energy conservation is always an important factor for extending the lifetime of WSNs. In the literature, a lot of researches have been done to extend the lifetime of WSNs [6], [8]–[11]. In addition, sensor nodes also have constraints on communication bandwidth, communication range, and storage space. The communication range restricts the transmission capacity of a sensor within a localized area in the network. Thus, a message may be transferred multiple times through several intermediate nodes along a path to its destination. This kind of flooding-like routing scheme causes huge amount of traffic collision, message redundancy, and energy consumption.

In order to overcome these shortcomings, an efficient approach named clustering has been widely used by many researchers. We can divide a given WSN into several disjointed clusters, each with a cluster head to take charge of the data gathering process and the communications inside or outside the cluster. In this mechanism, any ordinary node in the WSN only needs to collect information and send the data to its corresponding cluster head, which saves a lot of energy by avoiding many redundant message forwarding processes. The most important part for clustering scheme is to efficiently partition the given WSN into disjointed clusters and many algorithms were proposed to achieve this purpose. In [12], through coordination of nodes in the same cluster to efficiently reduce the redundant sensing area, the authors proposed a clustering algorithm to optimize the energy conservation. Load balancing is also a crucial issue for clustering mechanisms. Younis et al. [13] aimed at setting equal-sized clusters to balance the work load of each cluster. By setting a hop threshold k, researchers also discussed k-hop clustering problem [14]–[16] requiring that each node in the network should not be more than k hops far from its cluster head.

The average size of clusters is an important metric to estimate the performance of a WSN. If the cluster size is too large, then it is difficult for the cluster head to manage the cluster. In contrast, if the cluster size is too small, then there will be too many clusters in the WSN, which downgrades the performance of clustering strategy. An effective way to control the average size is to set a parameter d, and to let each cluster head only take charge of its d-hop neighbors. To achieve better communication between clusters, any pair of the cluster heads should be able to communicate with each other directly or through the relay of the other cluster heads. Thus the problem becomes finding an effective partition with minimum number of clusters w.r.t. d, while each cluster head can generate a

§X.Gao is the corresponding author.

1063-6927/15 $31.00 © 2015 IEEE
DOI 10.1109/ICDCS.2015.76

connected subgraph with the smallest number of additional connectors.

More precisely, we use a graph $G = (V, E)$ to model a WSN, where V is the set of sensors in the network while $(u, v) \in E$ iff u and v communicate with each other. In this paper, we consider a homogeneous network, where each node has the same communication range. So the graph can be formed as a *unit disk graph* (UDG). We focus on constructing a connected d-hop clusters for G. Actually, the set of cluster heads can be considered as a d-hop dominating set (d-DS). Consequently, our task is to find a connected d-DS (d-CDS) for a given graph. Moreover, we hope the cardinality of the d-CDS is minimized in order to reduce the maximum possible number of redundant messages, which corresponds to a *minimum connected d-hop dominating set* (d-MCDS) problem.

In all, our goal is to find a d-MCDS for a UDG $G = (V, E)$. In this paper, we propose a three-phase algorithm named *Connected Sparse Clustering Strategy* (CS-Cluster) to solve this problem. The first phase of CS-Cluster is to select a d-hop maximal independent set (d-MIS). Then, some extra nodes are added to connect the d-MIS into a d-CDS. In order to further reduce the size of the obtained d-CDS, the third phase is to check and remove redundant nodes. Our contributions in this paper are summarized as follows:

- We propose a novel approximation algorithm for d-MCDS problem, and prove that its approximation ratio is $(2d + 1)\lambda$, where λ is a parameter related with d but no more than 18.4. Our algorithm improves the previous best ratio of $O(d^2)$ into $O(d)$. Moreover, evaluation results also exhibit the outstanding performance of CS-Cluster compared with the closest related work.
- We theoretically analyze the upper bound of the size of d-MIS (denoted as $\alpha(G)$) w.r.t. the size of d-MCDS (denoted as $\gamma(G)$) for a UDG $G = (V, E)$. We prove that the ratio of $\alpha(G)$ and $\gamma(G)$ is bounded by λ. Compared with the previous best result $O(d)$, we reduce it to $O(1)$, which is a huge improvement.
- CS-Cluster is a distributed algorithm, which is more suitable for applications in WSNs. With thorough discussions, we conclude that CS-Cluster not only fits for UDG model, but also for general network model.

Our paper is organized as follows. Section II provides some preliminaries. In Section III, we introduce the related works. In Section IV, we describe our algorithm CS-Cluster for d-MCDS problem. In Section V, we analyze the upper bound of d-MIS for a given graph. Afterwards, we analyze the performance of CS-Cluster in Section VI and Section VII gives the simulation. Finally, Section VIII concludes this paper.

II. Preliminaries

In this section, we will introduce some definitions and then summarize the symbols used in the paper. Firstly, for a given graph $G = (V, E)$, we give several definitions.

Definition 1 (UDG)**.** *G is a **Unit Disk Graph (UDG)** if $\forall u, v \in V$, there is an edge $(u, v) \in E$ if and only if $dis(u, v) \leq 1$. Here $dis(u, v)$ is the Euclidean distance between u and v.*

Definition 2 (d-DS)**.** *$D \subseteq V$ is a **d-hop Dominating Set (d-DS)** of G if $\forall v \in V$, either $v \in D$ or $\exists u \in D$ such that there exists a path within d-hop between u and v.*

Definition 3 (d-IS)**.** *A **d-hop Independent Set (d-IS)** of G is a subset $I \subseteq V$ such that $\forall u, v \in I$, there does not exist a path within d-hop between u and v.*

Definition 4 (d-MIS)**.** *A d-IS I is a **d-hop Maximal Independent Set (d-MIS)** if $\forall v \in V \backslash I$, $I \cup \{v\}$ is no longer a d-IS.*

Many researchers consider finding a d-MIS instead of a d-DS for graph G, since the former is easier and more efficient. Thus we need the following lemma.

Lemma 1. *For any given graph G, a d-MIS is also a d-DS.*

Proof: For a d-MIS I and any node $v \in V \backslash I$, $I \cup \{v\}$ is no longer a d-MIS, which means v is dominated by some node in I in d hops. As a result, I is a d-DS. ■

Definition 5 (d-CDS)**.** *A subset $C \subseteq V$ is a **Connected d-hop Dominating Set (d-CDS)** of G if C is a d-DS and the subgraph induced by C is connected.*

In addition, we need to define the Voronoi Division which will be referred in next sections.

Definition 6 (Voronoi Division)**.** *Let S be a set of nodes in Euclidean space. For each node $v \in S$, the corresponding Voronoi cell $V(v)$ is the set of points that are closer to v than to other nodes of S, which means*

$$V(v) = \{w \mid \text{for every } u \in S \backslash \{v\}, dis(v, w) \leq dis(u, w)\}.$$

The Voronoi diagram is the partition induced by Voronoi cells.

In order to simplify our problem, we make some assumptions. First, we focus on WSNs located at 2-dimensional space. Second, each sensor in the network has the same communication range and implements efficient scheduling strategy with multiple available frequencies, such that no collision occurs in the procedure of message transmitting. Under these assumptions, we define d-MCDS problem as follows.

Definition 7 (d-MCDS)**.** *For a given UDG $G = (V, E)$, finding a d-CDS with minimum size is called d-MCDS problem.*

Table I introduces and summarizes the symbols, functions, and notations that will be used in the following sections.

III. Related Works

The minimum connected dominating set problem (namely, 1-MCDS problem) is a classic NP-complete problem. Over the past decades, there are a lot of researches related to it. In [17], Clark et al. first proved that MCDS problem is NP-complete even in UDG. To obtain a better feasible solution, Wan et al. [18] devised a two-phase algorithm with constant-factor approximation ratio. The first part of their algorithm is

TABLE I
SYMBOLS, NOTATIONS, AND FUNCTIONS IN THIS PAPER

Notation	Explanation
$dis(u,v)$	the Euclidean distance between u and v.
$\|S\|$	the cardinality of set S.
$N^r(v)$	the r-hop neighbor set of v with nodes which are at most r-hop away from v.
$N^r(S)$	the r-hop neighbor set of S with nodes which are at most r-hop away from some node in S.
$disk_r(v)$	the disk with center v and radius r.
$A_r(v)$	the area of $disk_r(v)$.
$A_r(S)$	the area of $\cup_{v\in S} disk_r(v)$
$Area(P)$	the area of geometrical shape P.

to select an MIS. Then, it adds some extra nodes to connect the MIS into a CDS. Later, a lot of related works were done to improve the approximation ratio of this two-phase algorithm [19]–[21]. The ratio of the size of MIS and the size of MCDS in graph G is crucial to estimate the algorithm's performance. The upper bound of such ratio is also called the theoretical bound to approximation CDS. Up to now, the best result for the theoretical bound is 3.399 by Du et al. [22].

The problem of d-MCDS is an extension of 1-MCDS problem. In 2000, Vuong et al. [23] proved that d-MCDS is NP-complete in general graph by a reduction from 3-SAT problem. Later, Nguyen et al. [24] proved that d-MCDS is NP-complete even in UDG. A lot of heuristic algorithms were proposed to find a feasible solution of d-MCDS [25], [26]. However, these algorithms all lacked approximation analysis.

In 2010, Li et al. [27] proposed an approximation algorithm on finding minimum two-connected d-hop dominating set. They gave an approximation ratio of $O(\log|V|)$. Later, Gao et al. [28] presented a two-phase distributed algorithm to compute d-CDS in UDG, and they gave a constant-factor approximation ratio of $O(d^3)$. Zhang et al. [29] improved the approximation ratio into $O(d^2)$. In 2014, Zhu et al. [30] proposed the first constant-factor approximation for d-MCDS in 3-dimensional space.

There are many other related works for the clustering problem. Wang et al. [31] proposed a PTAS to minimize the average hop distance from any nodes to cluster heads in 2D sensor networks without connectivity property. Kim et al. [32] discussed how to find the locations of k sinks such that the maximum distance from other nodes to sinks is minimized.

IV. CS-CLUSTER FOR d-MCDS PROBLEM

In this section, we introduce our algorithm named *Connected Sparse Clustering Strategy* (CS-Cluster) for d-MCDS problem in $G=(V,E)$. CS-Cluster consists of three phases. First, we will construct a sparse d-MIS. Second, we will connect the d-MIS into a d-CDS by adding some extra nodes called connectors. Finally, we will remove redundant nodes from the obtained d-CDS to further reduce the size of d-CDS.

A. Constructing a d-MIS

According Lemma 1, we can select a d-MIS as d-DS for a given graph. And we usually select d-MIS nodes one by one. For example, assume the current d-IS is S, then we select a node from $V\setminus\cup_{v\in S}N^d(v)$ and add it to S. We continue this process until S is a d-MIS. In literature such as [28]–[30], researchers often restrict that the new node is $d+1$ hops away from S. However, if we remove such restriction, we can form a sparser d-MIS. An example can be shown in Fig. 1 where $d=2$. If we implement traditional algorithms, we will select nodes $\{4,9,12,15,16\}$ with size 5. However, if we ignore such restriction, we can select a set $\{4,11,16\}$ with size 3.

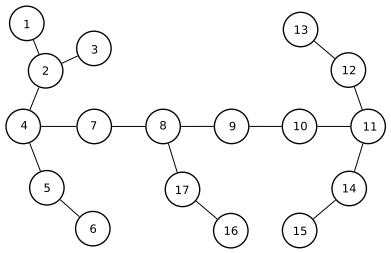


Fig. 1. The input graph of a 2-MCDS problem with 17 nodes.

Based on this observation, we propose an efficient distributed algorithm to select a sparser d-MIS as Alg. 1 shows. The main idea is to select a d-hop dominator as far as possible to the current d-IS. This strategy is also helpful to select connectors more flexibly in the next step.

In Alg. 1, we use $dominatee(u)$ to denote the number of u's white neighbors within d hops in the current round. This algorithm is processed round by round. Obviously, for any white node u, if $dominatee(u)$ is the maximum compared with those values of nodes in $N^d(u)$, selecting node u can cover the most white nodes compared with selecting any node in $N^d(u)$. Since Alg. 1 is a distributed algorithm, there may be more than one nodes to be selected in each round. Once a white node is selected and is colored black, it may affect the states of nodes within its d hops, including the value of $dominatee(\cdot)$. Considering that, for any node u, we restrict that only one white node in $N^{2d}(u)$ is selected in one round.

Besides, we use set $dominator(u)$ to denote the nodes which can dominate node u. These variable will be used in later. The pseudo-code of this procedure is presented in Alg. 1.

Algorithm 1: Constructing d-MIS

In each round, for every white node v **in** V**:**
1 Compare the value of $dominatee(v)$ with $dominatee(u)$, where $u\in N^{2d}(v)$. Use node id to break ties in the procedure of comparison;
2 **if** $dominatee(v)$ *is the largest* **then**
3 Color v black;
4 Send message to each node u in $N^d(v)$, and tell it to add v's id to $dominator(u)$. If u is white, color it grey;
5 Send message to each node $w\in N^{2d}(v)$, and tell it to check and update the value of $dominatee(w)$;

Fig. 2 shows the corresponding 2-MIS after processing Alg. 1 for the 2-hop MCDS problem in Fig. 1.

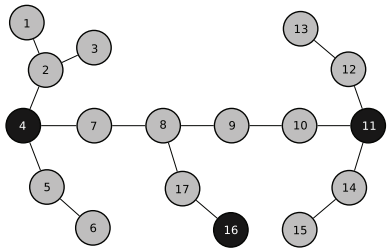


Fig. 2. The result after processing Alg. 1. Initially, $dominatee(4)$ and $dominatee(11)$ are the largest respectively, compared with nodes within their 4-hop neighbors. Thus, we color node 4 and 11 black and all their neighbors within 2 hops {1, 2, 3, 5, 6, 7, 8, 9, 10, 12, 13, 14, 15} grey. In the second round, both of two white nodes (node 16 and node 17) have one white neighbor. We break tie with smaller id, color node 16 black and color node 17 gray. Since there are not more white nodes, Alg. 1 terminates.

B. Connecting a d-MIS

After Alg. 1, we get a d-MIS and denote it as M, which is also a d-DS. In this subsection, we discuss how to connect it into a d-CDS. Before introducing the algorithm, we first introduce some notations and definitions that will be used in this section.

For a given graph $G = (V, E)$, any two nodes $u, v \in V$ are "k-hop connected" to each other if there exists a path in graph G between them and the length of the path is at most k. A subset $S \subseteq V$ is a "k-hop connected d-hop dominating set" iff:

- S is a d-hop dominating set of graph G.
- For any vertex $u \in S$, there exists at least one vertex $v \in S$ such that u and v are k-hop connected.

In addition, for a graph $G = (V, E)$, a subset $S \subseteq V$ is a "k-hop connected component" iff:

- For any vertex $u \in S$, there exist at least one vertex $v \in S$ such that u and v are k-hop connected.
- For any other vertex $v \in V \backslash S$, there does not exist any vertex $u \in S$ such that u and v are k-hop connected.

For a vertex subset $C \subseteq V$, let $f_k(C)$ be the number of k-hop connected components of the induced subgraph $G[C]$. $\forall v \in V$, define

$$-\Delta_v f_k(C) = f_k(C) - f_k\left(C \cup \{v\}\right),$$

representing the reduced number of k-hop connected components in C when v is inserted into C.

Next, we will start to introduce our algorithm which can connect M into a d-CDS. The algorithm in this subsection is processed round by round. In the i^{th} round, a subset C_i is selected from $V \backslash \bigcup_{j<i} C_i \cup M$ such that $\bigcup_{j \le i} C_j \cup M$ is a $r(i)$-hop connected d-hop dominating set. The definition of $r(i)$ is:

- $r(0) = 2d + 1$,
- $r(i) = \left\lfloor \frac{r(i-1)+1}{2} \right\rfloor$.

Obviously, after Alg. 1, M is an $r(0)$-hop connected d-hop dominating set, which means any two nodes in M are $(2d + 1)$-hop connected in graph G. When the algorithm in this subsection terminates, we should get a 1-hop connected d-dominating set, namely d-CDS. Alg. 2 depicts the detailed steps.

Algorithm 2: Connecting d-MIS

$r_0 = 2d + 1$;
In the i^{th} round:
$r_i = \lfloor \frac{r_{i-1}+1}{2} \rfloor$;
Initialize $componentNum.(v)$;
For every grey node $v \in V$:
Send messages to nodes in $N^{r_i}(v)$ to compute $cost(v)$;
After $cost(v)$ is figured out, inform this value to nodes in $N^{r_i}(v)$;
For each iteration in this round:
Compare $cost(v)$ with elements in $connector(u)$ where node u is a black or blue node and $u \in N^{r_i}(v)$. Use node id to break ties in the procedure of comparison;
if $cost(v)$ *is the largest* **then**
    Color v blue;
    Send messages to nodes in $N^d(v)$, inform them to add v's id to their $dominator(\cdot)$;
    Send messages to nodes in $N^{r_i}(v)$, inform them to change $componentNum.(\cdot)$ to be the id of node v, or to recompute the value of $cost(\cdot)$;
When receiving a message informing to recompute $cost(\cdot)$**:**
Send messages to nodes in $N^{r_i}(v)$ to compute $cost(v)$;
After $cost(v)$ is figured out, inform this value to nodes in $N^{r_i}(v)$;
For any black or blue node $u \in V$:
When receiving a message from node v to compute $cost(v)$**:**
Reply with $componentNum.(u)$ and the hop distance between u and v;
When receiving a message from node v informing the value of $cost(v)$**:**
Store $cost(v)$ in $connnector(u)$;
When receiving a message from a grey node v to compare $cost(v)$**:**
If $cost(v)$ is the largest among all elements in $connector(u)$, reply with "YES". Use node id to break ties in the procedure of comparison. Otherwise, reply with "NO";
When receiving a message from node v informing to change $componentNum.(u)$**:**
Change $componentNum.(u)$ as it is required;
Inform u's black and blue neighbors within r_i hops, not including v, to change their $componentNum.(\cdot)$ to be $componentNum.(u)$;
Inform nodes in $N^{r_i}(u)$ to recompute their $cost(\cdot)$;

To make the size of C_i as small as possible, we should choose the most efficient nodes that makes $\bigcup_{j \le i} C_j \cup M$ be an $r(i)$-hop connected d-hop dominating set. An intuitive idea is to iteratively choose nodes that can reduce the most number of $r(i)$-hop connected components. In order to make our algorithm more economical, we also add another requirement in this procedure.

To better understand our algorithm, we first assume that each node knows the global information. At the beginning of the t^{th} iteration of the i^{th} round, C_i contains $t - 1$ nodes. Assume $C = \bigcup_{j \le i} C_i \cup M$. For a node v in $V \backslash C$, it reduces the number of $r(i)$-hop connected components by $-\Delta_v f_{r(i)}(C)$. Consequently, there exist $-\Delta_v f_{r(i)}(C)$ shortest

paths that connect node v to those $-\Delta_v f_{r(i)}(C)$ components respectively. We use $no.(v)$ to denote the total number of nodes in those shortest pathes except two end points (but includes v). Moreover, we use $cost(v)$ to denote the cost of v under the current state and its definition is as follows.

$$cost(v) = \frac{-\Delta_v f_r(C \cup C_i)}{no.(v)}.$$

In every iteration, we select the node with the largest cost. Note that the additional requirement of $cost(v)$ is much crucial to guarantee and improve the performance of our algorithm.

Since our algorithm here is a distributed algorithm, each node run the algorithm locally. As a consequence, in each iteration of some round, there may be more than one node to be selected. Besides, once a node is selected, its state, such as its color, will changed. Such change will further affect states of its neighbors. Hence, nodes selected in a same iteration of some round may cause collisions. In order to avoid that, we need to do more than the simpler case above. First, each node has a local variable $componentNum.(\cdot)$, which records the component the node belongs to. The component here is actually r_i-hop connected component. Initially, at the beginning of each round in Alg. 2, for any node u, the value of $componentNum.(u)$ equals to the id of node u. Then, each node informs its value of $componentNum.(\cdot)$ within r_i hops. For any two nodes u, v, which are within r_i hops to each other, their $componentNum.(\cdot)$ are set to be the smaller id of them.

Before the first iteration of the i^{th} round, for each grey node v, node v sends messages to nodes in $N^{r_i}(v)$ to compute $cost(v)$. Once a black or blue node receives such a message, it replies that with its value of $componentNum.(\cdot)$ and the hop distance between itself and the sender of such message. With these information, node v can figure out $cost(v)$. Then v broadcasts $cost(v)$ to nodes in $N^{r_i}(v)$. All black and blue nodes in $N^{r_i}(v)$ store $cost(v)$ in their local variables $connector(\cdot)$.

In each iteration of the i^{th} round, each grey node v compares $cost(v)$ with elements in $connector(w)$ where w is a black or blue node and $w \in N^{r_i}(v)$. If $cost(v)$ is the largest, v is selected as a connector and colored blue. Once a node v is selected as a connector, all the black and blue nodes in $N^{r_i}(v)$ set their $componentNum.(\cdot)$ to be the id of node v. Of course, once a black or blue node u changes the value of $componentNum.(u)$, it informs all those nodes whose $componentNum.(\cdot)$ are the same with $componentNum.(u)$ to set their $componentNum.(\cdot)$ as $componentNum.(u)$. Besides, the new component informs its all grey neighbors within r_i hops to update their $cost(\cdot)$ and further update some values of $connector(\cdot)$.

An example of Alg. 2 is shown in Fig. 3.

Next, we prove the correctness of Alg. 2. Initially, the input M is a d-MIS, so $f_{d+1}(M) = |M|$ and $f_{2d+1}(M) \leq |M|$. According to the definition of d-MIS, the nodes in d-MIS can be ordered in a way such that each node is at most $(2d+1)$-hop away from one of its predecessors. We call such a property as

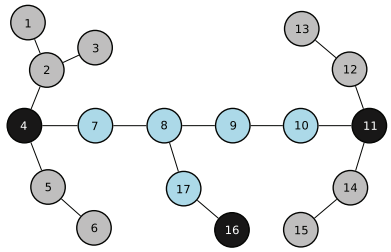


Fig. 3. The result after processing Alg. 2. At the beginning of the first round, there are three 3-hop connected components ({4}, {11}, {16}). We should 3-hop connect these three components into one. At this time, the values of nodes 7, 8, 9, 17 are 1/3, 2/5, 2/5, 1/3 respectively, while all other grey nodes have value 0. Since node 8 has smaller id than node 9, we color node 8 blue. Now the 3-hop connected component is only {4, 8, 11, 16}, and thus the first round terminates. Next, in the second round there are two 2-hop connected components ({4, 8, 16}, {11}). At this time nodes 9, 10 have the same value of 1/2 while other grey nodes have value 0. Similarly we color node 9 blue and terminate the second round. When the third round starts, there are four 1-hop connected components ({4}, {8, 9}, {11}, {16}), while nodes 7, 17, 10 have the same value of 1 and other grey nodes have value 0. Thus, we select node 7 in the first iteration of this round. Then, select 10 in the second iteration, and 17 in the third iteration. After that, Round 3 terminates. At this time $r = 1$. Consequently, Alg. 2 completes its task.

"$(2d+1)$-hop connection property". Hence, easy to see that $f_{2d+1}(M) = 1$. Actually we have Lemma 2 as follows.

Lemma 2. *For any subset S, it has k-hop connection property if and only if $f_k(S) = 1$.*

Proof: It is clear that S has k-hop connection property if and only if the subgraph induced by S is k-hop connected, which is true when $f_k(S) = 1$. ■

Lemma 3. *In Alg. 2, the i^{th} round will terminate when $f_{r(i)}(\bigcup_{j \leq i} C_j \cup M) = 1$.*

Proof: When $i = 0$, Lemma 3 holds obviously as we have shown above. Assume Lemma 3 holds for the i^{th} round where $i \leq 0$, we show that it also holds for the $(i+1)^{th}$ round.

Obviously, the $(i+1)^{th}$ round starts with $f_{r(i)}(\bigcup_{j \leq i} C_j \cup M) = 1$. In this round, we need to connect all those $r(i+1)$-hop connected components into only one $r(i+1)$-hop connected component. It is clear that as long as $f_{r(i+1)}(\bigcup_{j \leq i+1} C_j \cup M) > 1$, there must exist more than one $r(i+1)$-hop connected components. And these $r(i+1)$-hop connected components are $r(i)$-hop connected because the set $\bigcup_{j \leq i} C_j \cup M$ has the $r(i)$-hop connection property. Thus, there must exist two $r(i+1)$-hop connected components which are $r(i)$-hop connected, and a path of length at most $r(i)$ exists between them. The middle node in this path is at most $\lfloor \frac{r(i)+1}{2} \rfloor = r(i+1)$ away from these two components. So it can connect these components into one. Thus, at each iteration in the $(i+1)^{th}$ round, the algorithm can always find a satiable node that can reduce the number of $r(i+1)$-hop connected components by at least 1. Therefore, Alg. 2 will continually execute until the round reaches the final state $f_{r(i+1)}(\bigcup_{j \leq i+1} C_j \cup M) = 1$. Then Lemma 3 holds. ■

Theorem 1. *Alg. 2 connects a d-MIS into a d-CDS.*

Proof: According to Lemma 3, Alg. 2 ends up at

$f_1(\bigcup_j C_j \cup M) = 1$, which means the subgraph induced by $\bigcup_j C_j \cup M$ is connected. On the other side, set M alone can dominate V. Thus, $\bigcup_j C_j \cup M$ is a d-CDS. ■

C. Removing the Redundant Nodes

After the first two phases, we can obtain a d-CDS for a given graph and denote it as C. However, there may exist a number of redundant nodes. A node v in d-CDS is redundant iff:

- Every node that v dominates have at least one alternative dominator.
- The subgraph induced by $C - \{v\}$ is connected.

The second requirement above can be analyzed in details. For any node $v \in C$, there exist two situations to discuss. First, in the subgraph $G[C]$ induced by C, if v is a leaf node, namely v's degree in $G[C]$ is one, then removing v has no effects on the connectivity of the subgraph $G[C-\{v\}]$ induced by $C - \{v\}$. Secondly, if v is connected by more than one connectors, namely v's degree in $G[C]$ is more than 1, then the subgraph $G[C-\{v\}]$ is connected only when these connectors are connected.

Based on this, we can further reduce the size of d-CDS by Alg. 3. Obviously, for Alg. 3, the most important work is to decide whether a black node in C is redundant. According to the analysis above, we can determine that by checking the corresponding two requirements.

To check the first requirement, for any black node v, v sends messages to its grey neighbors within d hops and asks whether they have alternative dominators. Once a grey node u receives such a message, it will check its local variable $dominator(u)$ which stores ids of nodes that can dominate u in d hops. If $dominator(u)$ contains the id of the sender of the message, u sends a positive reply. Otherwise, u sends a negative reply.

Next, we discuss how to valid the second requirement. Since C is already a connected set, all nodes in C have the same value of $componentNum.(\cdot)$. And we call the corresponding component as the final component. For any black v, v sends messages to its black and blue neighbors in $N^1(v)$ and asks whether they can connect the final component by some alternative nodes. Once another black or blue node u receives such a message, it will check whether u's black or blue neighbors in $N^1(d)$ have the same value of $componentNum.(u)$. If there exists at least one such neighbor, u sends a positive reply. Otherwise, it sends a negative reply.

The pseudo-code of this pruning algorithm is presented in Alg. 3.

Fig. 4 is an example to illustrate the performance of Alg. 3 after processing Alg. 1 and Alg. 2 for Fig. 1.

Algorithm 3: Removing Redundant Dominators

In each round, for every black node v**:**

1. v sends requests to grey nodes in $N^d(v)$ and asks whether they have some alternative dominators;
2. v sends requests to black and blue nodes in $N^1(v)$ and asks whether they can connect to the final component by some alternative nodes;
3. If all the replies are positive, it decides that it is redundant;
4. **if** *v is redundant* **then**
5. Color v as grey;
6. **foreach** *u **in** v's 1-hop blue neighbor set* **do**
7. **if** *u connects to only one blue or black node* **then**
8. Color u as black;
9. Update the local information for nodes in $N^d(v)$;

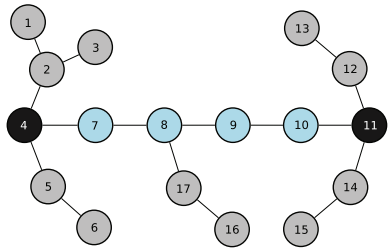


Fig. 4. The result after processing Alg. 3. In the first iteration, there are three black nodes, nodes 4, 11, 16. By checking, we find every node dominated by node 16 has alternative dominators (themselves). Besides, node 16 is connected to only one blue node (node 17). Thus, node 16 is redundant. Remove it from C and color it grey. Moreover, the degree of node 17 in $G[C]$ at this time is one. Thus, color node 17 black. In the second iteration, we have three black nodes, nodes 4, 17, 11. Similarly, we find that node 17 is redundant and remove it. Since the blue neighbor of node 17 is node 8 and the degree of node 8 is not 1, we do not change the color of node 8. Consequently, we have two black nodes, nodes 4, 11. By checking, neither of them is redundant. Thus, Alg. 3 terminates.

V. An Upper bound of d-MIS Size for a UDG G

To analyze the performance of CS-Cluster, we need to firstly discuss the upper bound of d-MIS size for a given graph. Given a UDG $G = (V, E)$, let $\alpha(G)$ denote the size of d-MIS in G and $\gamma(G)$ the size of d-MCDS in G. Since the ratio of $\alpha(G)$ and $\gamma(G)$ is crucial to estimate the performance of our algorithm, in this section we will discuss the upper bound of this ratio from two perspectives.

A. Voronoi Division and Euler's Formula

Assume that the d-MCDS of G is M and the d-MIS with maximum size is I. Obviously for any two nodes $u, v \in I$, the Euclidean distance between them is greater than 1. Thus, $disk_{0.5}(u)$ and $disk_{0.5}(v)$ do not intersect with each other. Moreover, for any node $u \in I$, $disk_{0.5}(u)$ can be completely contained in $\bigcup_{v\in M} disk_{(d+0.5)}(v)$. Then the value of $\alpha(G)$ should be less than $A_{(d+0.5)}(M)/A_{0.5}(v)$. It is easy to figure out that

$$A_{d+0.5}(M) \leq \pi(d+\frac{1}{2})^2 + (\gamma(G)-1)\Delta S \qquad (1)$$

where

$$\Delta S = (\pi - 2\theta)(d+\frac{1}{2})^2 + (d+\frac{1}{2})\sin\theta,$$

$$\theta = \arccos\left(\frac{1}{2d+1}\right).$$

Consequently, we can get an upper bound of $\alpha(G)$. However, this upper bound is a very rough result and we can improve it by Voronoi Division.

With Voronoi Division, we can partition the region $A_{d+0.5}(M)$ into Voronoi cells. For any node u in I, there is a corresponding Voronoi cell $V(u)$ which contains $disk_{0.5}(u)$. There are two types of Voronoi cells. If a cell does not contain any boundary point of $A_{d+0.5}(M)$, then we call it non-boundary Voronoi cell. Otherwise, it is a boundary Voronoi cell. For any non-boundary Voronoi cell, it is a polygon. Let s_k denote the minimum area of Voronoi cell with k edges. For those boundary Voronoi cells, we also consider them as a special kind of polygons and let s'_k denote the minimum area of boundary Voronoi cell with k edges. Next, we will first calculate the values of s_k and s'_k.

Gao et al. [20] has proved that, if a non-boundary Voronoi cell $V(u)$ has k edges and $k \leq 6$, then the regular polygon with k edges which is inscribed with $disk_{0.5}(u)$ has the smallest area. Let P_k be such kind of polygon. If $V(u)$ is a boundary Voronoi cell with k edges, under two conditions $V(u)$ could have the minimum area. The first condition is when boundary arc cut off one edge of P_k as Fig. 5 shows, and we name it as E_k. The second condition is when boundary arc cut off one angle of P_k as Fig. 5 shows, and we name it as A_{k+1}.

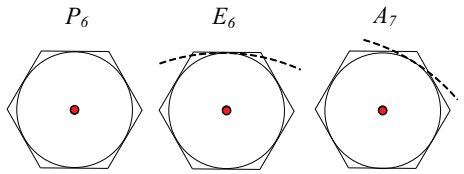


Fig. 5. An example to show the conditions of non-boundary Voronoi cells and boundary Voronoi cells having the minimum area when $k = 6$.

Since $V(u)$ with 6 adjacent neighbors is the densest situation if any two small disks does not intersect with each other, any non-boundary Voronoi cell $V(u)$ with more than 6 edges cannot be inscribed with $disk_{0.5}(u)$ any more. Thus, the areas of that kind of Voronoi cells are greater than s_6. Consequently, we can conclude that $s_i \geq s_6$ for $i \geq 7$. Similarly, for $i \geq 7$, the area of E_i is greater than the area of E_6. And the area of A_i is greater than the area of A_7 for $i \geq 8$.

Since non-boundary Voronoi cells have nothing to do with the boundary arc of $A_{d+0.5}(M)$, then the value of s_k also has nothing to do with d. It is not difficult to figure out that the area of P_k is $\frac{k}{4}\tan(\frac{\pi}{k})$ for $3 \leq k \leq 6$. And the results are as follows: $s_3 \geq s_4 \geq s_5 \geq s_6 \leq s_7 \leq s_8 \ldots$ and $s_3 = 1.299, s_4 = 1, s_5 = 0.9082, s_6 = 0.8661$.

With mathematic deduction, we can figure out the values of $Area(A_k)$ and $Area(E_k)$ and the results are as follows:

$$Area(A_k) = \frac{k-1}{4}\tan(\frac{\pi}{k-1}) - \left(\frac{\sin\phi}{2} + \frac{\sin^2(\phi/2)}{\tan(\theta/2)} - \frac{\phi}{2}\right)(d+\frac{1}{2})^2 \quad (2)$$

$$Area(E_k) = \frac{k}{4}\tan\left(\frac{\pi}{k}\right) - \left(\sin\phi - \phi + \frac{\sin^2\frac{\phi}{2}}{\tan\frac{\theta}{2}(d+\frac{1}{2})}\right)(d+\frac{1}{2})^2 \quad (3)$$

According to Eqn. (2) and Eqn. (3), by numerical methods, we can conclude that $s'_i = Area(E_i)$ for $3 \leq i \leq 6$ and $s'_7 = Area(A_7)$. Moreover,

$$s'_3 \geq s'_4 \geq s'_5 \geq s'_6 \geq s'_7 \leq s'_8 \ldots$$

Specially, when applying $d = 1$ to Eqn. (2) and Eqn. (3), we can get the corresponding values of s'_k and we denote them as s'_{k1}. The results are as follows.

$$s'_{31} = 1.1781,\ s'_{41} = 0.9717,\ s'_{51} = 0.8968,$$
$$s'_{61} = 0.8601,\ s'_{71} = 0.8550.$$

On the other side, we observe that the curvature of those boundary arcs in Fig. 5 will decrease with the increasing of d. Thus, the area that is cut off by boundary arc from original regular polygon will also decrease. Consequently, the values of $Area(E_i)$ and $Area(A_i)$ increase with the increasing of d. Therefore, we have $s'_k \geq s'_{k1}$ for $3 \leq k \leq 7$.

Next, with Euler's formula we can obtain a better upper bound for $\alpha(G)$ just as what Du et al. did in [22]. Let m and n respectively be the number of edges and vertices in the Voronoi diagram. Let f_i and f'_i be the the number of non-boundary Voronoi cells with i edges and boundary Voronoi cells with i edges respectively. By Euler's formula, we have

$$1 + \sum_{i=1}^{\infty}(f_i + f'_i) + n = m + 2.$$

Considering the degree of each vertex in Voronoi diagram is at least 3, we have $3n \leq 2m$. Then we have

$$\sum_{i=3}^{\infty}(f_i + f'_i) \geq 1 + \frac{m}{3}. \quad (4)$$

It is clear that each boundary Voronoi cell has at least one edge which is on the boundary of $A_{d+0.5}(M)$. Let f_{out} denote the total number of those edges in the Voronoi diagram. Then

$$\begin{aligned} 2m &= f_{out} + \sum_{i=3}^{\infty} i(f_i + f'_i) \\ &\geq \sum_{i=3}^{\infty}(if_i + (i+1)f'_i). \end{aligned} \quad (5)$$

Combining Inequalities (4) and (5), we have

$$\sum_{i=3}^{\infty}(6-i)f_i + (5-i)f'_i \geq 6.$$

Considering $\sum_{i=3}^{\infty}(s_i f_i + s'_i f'_i) \leq A_{d+0.5}(M)$, we have

$$\sum_{i=3}^{\infty}\left((s_i - (s_6 - s'_7)(6-i))\, f_i + (s'_i - (s_6 - s'_7)(5-i))\, f'_i\right) \leq A_{d+0.5}(M) - 6(s_6 - s'_7).$$

Noting that $s_6 > s'_6$, for $i \geq 7$, we have

$$s_i - (s_6 - s'_7)(6-i) \geq s_i \geq s_6$$

and

$$s'_i - (s_6 - s'_7)(5-i) \geq s'_7 + 2(s_6 - s'_7) \geq s_6.$$

For $3 \leq i \leq 6$, we can verify that

$$s_i - (s_6 - s'_7)(6-i) \geq s_i - (s_6 - s'_{71})(6-i) \geq s_6,$$

and

$$s_i' - (s_6 - s_7')(5-i) \geq s_{i1}' - (s_6 - s_{71}') \geq s_6.$$

Then,

$$\alpha(G) = \sum_{i=3}^{\infty}(f_i + f_i') \leq \frac{A_{d+0.5}(M) - 6(s_6 - s_7')}{s_6}$$

When $d \to \infty$, we have $s_6 - s_7' \to \frac{7\sqrt{3}}{12} - 1$. Hence,

$$\alpha(G) \leq \frac{A_{d+0.5}(M)}{\frac{\sqrt{3}}{2}} - 0.0718 \tag{6}$$

B. A New Upper Bound

Up to now, actually we did not make full use of the property of d-MIS. What we used for our analysis in Sec. V-A is just that for any two nodes in d-MIS, their Euclidean distance is greater than 1. In this subsection, we will analyze the properties of d-MIS more specifically so that we can get a better upper bound for $\alpha(G)$.

In [29], Zhang et al. proposed a novel idea to analyze the upper bound of $\alpha(G)$. And they gave $\alpha(G) \leq \beta\gamma(G)$, where β refers to how many independent nodes can be contained in $N^d(u)$ for any node u. The expression of β is

$$\beta = \begin{cases} 5, & \text{if } d = 1, \\ 21, & \text{if } d = 2, \\ 5 + \dfrac{4d(d+1)}{\lceil \frac{1}{2}\lfloor \frac{d-1}{2} \rfloor \rceil}, & \text{if } d \geq 3. \end{cases}$$

This result improved the previous upper bound from $O(d^2)$ into $O(d)$. Nevertheless, this result can still be greatly improved and we will show it as follows.

We divide the whole d-hop independent nodes I into two classes. They are

$$A = N^{\lceil d/2 \rceil}(M) \cap I = \{u_1, u_2, \ldots, u_t\},$$

and

$$B = N^d(M) \backslash N^{\lceil d/2 \rceil}(M) \cap I = \{v_1, v_2, \ldots, v_s\}.$$

For the first class, we are to show that $t \leq 3.399\gamma(G) + 4.874$.

For $1 \leq i \leq t$, denote the shortest path from u_i to M as P_i and let w_i be the last node on P_i before reaching M. Because $N^1(M)$ contains at most $3.399\gamma(G) + 4.874$ 1-hop independent nodes [22], if $t > 3.399\gamma(G) + 4.874$, we could always choose two nodes u_i and u_j such that w_i and w_j are adjacent. Then, $u_i P_i w_i w_j \overleftarrow{P_j} u_j$ is a path from u_i to u_j with length of at most $2(\lceil d/2 \rceil - 1) + 1 \leq d$, which contradicts that u_i and u_j are d-hop independent. Here, $\overleftarrow{P_j}$ is the reverse of P_j.

Next, we discuss the rest of d-hop independent nodes in I. For $1 \leq i \leq s$, denote the shortest path from v_i to M as Q_i and let S_i be the set of nodes which are in $N^{\lfloor (d-1)/2 \rfloor}(v_i) \cap Q_i$. Obviously, for each Q_i, its length is at least $\lceil \frac{d}{2} + 1 \rceil$. Consider any two nodes $v_i, v_j \in B$. We are to show that any node in S_i is not adjacent to any node in S_j. Suppose there are two nodes $w \in S_i$, $z \in S_j$ and w is adjacent to z. Then, there exists a path from v_i to v_j which contains w and z, and the length of this path is at most $2\lfloor (d-1)/2 \rfloor + 1 \leq d$, which contradicts that v_i and v_j are d-hop independent. Hence, $A_{0.5}(S_i)$ does not intersect with $A_{0.5}(S_j)$. Since Q_i is the shortest path from v_i to M, any node in Q_i is not adjacent to its second successor. Thus, S_i contains at least $\lceil \frac{1}{2}\lfloor (d-1)/2 \rfloor \rceil$ disjointed nodes which also means they are 1-hop independent. That is to say, one d-hop independent node corresponds to $\lceil \frac{1}{2}\lfloor (d-1)/2 \rfloor \rceil$ 1-hop independent nodes. Moreover, for any node $w \in S_i$, the hop distance from w to M is greater than $\lceil d/2+1 \rceil - \lfloor (d-1)/2 \rfloor \geq 1$, which means w is not in $A_{0.5}(M)$. According to the result in Sec. V-A, we have

$$s \leq \frac{\dfrac{A_{d+0.5}(M) - A_{0.5}(M)}{\sqrt{3}/2} - 0.0718}{\lceil \frac{1}{2}\lfloor \frac{d-1}{2} \rfloor \rceil}.$$

Combining the Inequality (1) and $A_{0.5}(M) \geq \pi/4$, we have

$$s \leq \frac{\pi(d + \frac{1}{2})^2 + (\gamma(G) - 1)\Delta S - 0.8572}{\frac{\sqrt{3}}{2}\lceil \frac{1}{2}\lfloor \frac{d-1}{2} \rfloor \rceil}.$$

Consequently, we obtain a new upper bound as follows:

$$\alpha(G) \leq \begin{cases} \dfrac{\pi(d + \frac{1}{2})^2 + (\gamma(G) - 1)\Delta S}{\frac{\sqrt{3}}{2}} - 0.0718, & \\ & \text{if } d \leq 2, \\ \dfrac{\pi(d + \frac{1}{2})^2 + (\gamma(G) - 1)\Delta S - 0.8572}{\frac{\sqrt{3}}{2}\lceil \frac{1}{2}\lfloor \frac{d-1}{2} \rfloor \rceil} & \\ +3.399\gamma(G) + 4.874, & \text{if } d \geq 3. \end{cases} \tag{7}$$

Let λ be the ratio of $\alpha(G)$ and $\gamma(G)$. Then, we have

$$\lambda = \begin{cases} \dfrac{\Delta S}{\sqrt{3}/2}, & \text{if } d \leq 2, \\ \dfrac{\Delta S}{\frac{\sqrt{3}}{2}\lceil \frac{1}{2}\lfloor \frac{d-1}{2} \rfloor \rceil} + 3.399, & \text{if } d \geq 3. \end{cases} \tag{8}$$

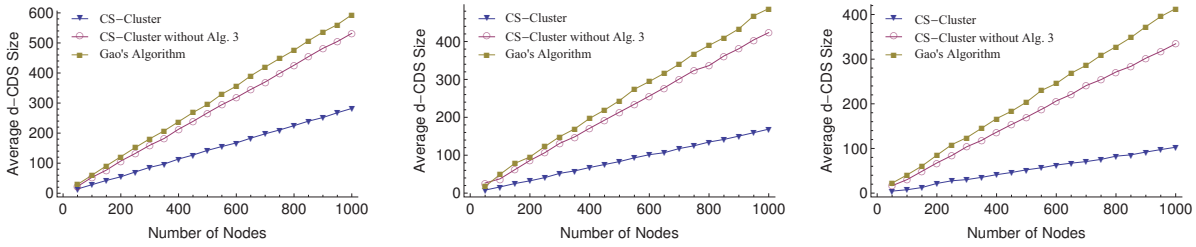
With numerical method, we can conclude that the value of λ is maximum when $d = 6$ and is equal to 18.4. Moreover, when $d \to \infty$, $\lambda \to 12.6366$. Therefore, for arbitrary value of d, the value of λ is no more than 18.4. In this sense, λ has little relation to d. Therefore, our analysis improve the previous $O(d)$ into $O(1)$. This is a huge improvement in the related area.

VI. Performance Analysis

In this section, we discuss the approximation ratio of CS-Cluster. Assume the size of an optimal d-MCDS is *opt*. Through Alg. 1, we get a d-MIS, namely the set M. According to the analysis in Sec. V, we have $|M| \leq \lambda opt$. As for Alg. 2, we have Lemma 4 below.

Lemma 4. *The total number of connectors selected in Alg. 2, namely $C \backslash M$, is at most $2d\lambda opt$.*

Proof: In the first round, as we described above, Alg. 2 is to connect all $r(1)$-hop connected components into one $r(1)$-hop connected component. Moreover, before the end of the current round, there always exists one node that can $r(1)$-hop connect two $r(1)$-hop connected components. Let v be that



(a). Comparison when $d = 2$ (b). Comparison when $d = 3$ (c). Comparison when $d = 4$

Fig. 6. Performance comparison between CS-Cluster and Gao's algorithm in [28] in different cases where $d = 2, 3, 4$ respectively.

node. Then we have

$$cost(v) \geq \frac{1}{r(0)-1} = \frac{1}{2d}.$$

Since Alg. 2 always selects the node with largest value in each iteration, the node that is chosen in this iteration has the value of at least $cost(v)$. Consequently, for each node in C_1, the value is at least $\frac{1}{2d}$ when it is chosen.

There exists an alternative algorithm to connect M. For each node $v \in C_1$, when it is chosen, assume the number of $r(1)$-hop connected components it can $r(1)$-hop connect is $comp(v)$. We select all the nodes in the shortest paths from v to the $comp(v)$ $r(1)$-hop connected components. Obviously, all those chosen nodes can connect M into a d-CDS. Besides, the size of C_i is at most $|M| - 1$ because each node in C_i can reduce at least one $r(1)$-hop component and the number of the original $r(1)$-hop connected components is at most $|M|$ when Alg. 2 begins. Thus, the size of these connectors is

$$\sum_{v \in C_1} no.(v) \leq \sum_{v \in C_1} \frac{comp(v)}{cost(v)} \leq 2d|M|.$$

Similarly, in the i^{th} round, for each node in C_i, the value when it is chosen is at least $\frac{1}{r(i-1)-1}$. The initial number of $r(i)$-hop connected components is no more than

$$|M| + \sum_{j=1}^{i-1} |C_j| \leq 2^{i-1}|M|.$$

There also exists an alternative algorithm to connect $M \cup_{j<i} C_i$ into a d-CDS, and the size of the corresponding connectors is

$$\sum_{v \in C_i} no.(v) \leq \sum_{v \in C_i} \frac{comp(v)}{cost(v)} \leq (r(i-1)-1)2^{i-1}|M|.$$

Since

$$\begin{aligned} (r(i-1)-1)2^{i-1} &= (\lfloor \frac{r(i-2)+1}{2} \rfloor - 1) \cdot 2 \cdot 2^{i-2} \\ &\leq (r(i-2)-1)2^{i-2} \\ &\cdots \\ &\leq r(0) - 1 = 2d, \end{aligned}$$

we have

$$\sum_{v \in C_i} no.(v) \leq 2d|M|.$$

Finally, in the t^{th} round where $r(t) = 1$, Alg. 2 does the same with its corresponding alternative algorithm. Therefore, $|C \backslash M| = \sum_{v \in C_t} no.(v) \leq 2d|M|$, and Lemma 4 follows. ■

From the procedure of proof above, it not difficult to conclude the following corollary.

Corollary 1. *After the i^{th} round, Alg. 2 will indicate a feasible solution F_i with size $\sum_{v \in C_i} no.(v)$ which alone can connect M into d-CDS. Moreover, the size of F_i is no more than F_{i-1}.*

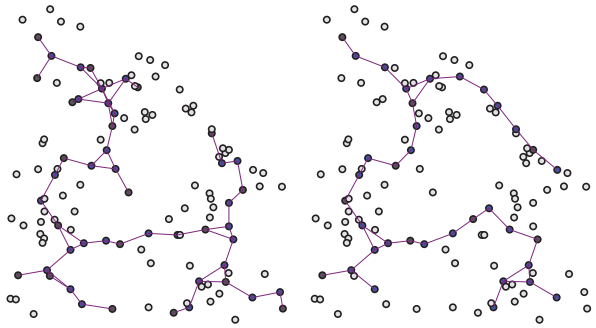
With Lemma 4, we can finally get the following theorem.

Theorem 2. *Our three-phase algorithm CS-Cluster has an approximation ratio of $(2d + 1)\lambda$.*

VII. Simulation

In this section, we compare CS-Cluster with previous literature [28] for the same problem (refer as Gao's algorithm), which is the most related work for d-MCDS problem. In our simulations, we randomly deploy sensor nodes in a 2D virtual space. The number of nodes varies from 50 to 1000 at intervals of 50. Each two nodes can connect to each other when the distance between them is at most 1. Moreover, we also ensure that the generated graph is connected.

Fig. 6(a)-(c) exhibit the performance comparisons between two algorithms with different d. From the figure, we can conclude that CS-Cluster's always performs better than Gao's algorithm under different settings. Moreover, when d increases, the superiority of CS-Cluster also increases. Considering the great influence of Alg. 3, we also compare the performance of CS-Cluster without Alg. 3 with Gao's algorithm. From Fig. 6, we find that CS-Cluster without Alg. 3 still produces



(a). 2-CDS by Gao's algorithm (b). 2-CDS by CS-Cluster

Fig. 7. A sample of UDG which contains 122 nodes. Gao's algorithm selects a 2-CDS with 49 nodes while CS-Cluster selects a 2-CDS with 36 nodes.

smaller d-CDS than Gao's algorithm. We think that the great advantage of CS-Cluster comes to the sparsity of d-MIS by Alg. 1, the flexibility of Alg. 2, and the feature of non-redundancy by Alg. 3.

Fig. 7 shows a sample comparison between two algorithms with 112 input nodes where $d = 2$. We can clearly see the superiority of CS-Cluster since Gao's algorithm constructs a 2-CDS with 49 nodes while CS-Cluster only uses 36 nodes.

VIII. Conclusion

In this paper, we studied the multi-hop connected clustering problem for a given wireless sensor network, which can be formed as finding a minimum d-hop connected dominating set problem (d-MCDS) for a given graph. We then proposed a distributed approximation algorithm named *Connected Sparse Clustering Scheme* (CS-Cluster) to solve the problem. CS-Cluster consists of three phases: dominator selection, connector insertion, and redundancy elimination. To evaluate the performance of CS-Cluster, we estimated the upper bound λ for the dominator size in a unit disk graph, and proved that λ is no more than 18.4. As a result, we reduce the bound of $O(d)$ from previous literature [29] to $O(1)$, and achieved an approximation ratio of $(2d + 1)\lambda$ for CS-Cluster, which is the best constant-factor approximation up to now. In all, CS-Cluster is flexible to obtain a nearly optimal d-MCDS and is suitable for distributed environments. Our simulation results also exhibited the outstanding performance of CS-Cluster.

Acknowledgement

This work has been supported in part by the National Natural Science Foundation of China (Grant number 61202024, 61472252, 61133006, 61422208), China 973 project (2012CB316201), Shanghai Educational Development Foundation (Chenguang Grant No.12CG09), Shanghai Pujiang Program 13PJ1403900, the Natural Science Foundation of Shanghai (Grant No.12ZR1445000), and in part by Jiangsu Future Network Research Project No. BY2013095-1-10 and CCF-Tencent Open Fund.

References

[1] J. Wang, Z. Cao, X. Mao, and Y. Liu, "Sleep in the dins: Insomnia therapy for duty-cycled sensor networks," in *IEEE INFOCOM*, 2014.

[2] S. Guo, C. Wang, and Y. Yang, "Mobile data gathering with wireless energy replenishment in rechargeable sensor networks," in *IEEE INFOCOM*, 2013.

[3] Q. Liao, L. Shi, Y. He, R. Li, Z. Su, A. Striegel, and Y. Liu, "Visualizing anomalies in sensor networks," in *ACM SIGCOMM*, vol. 41, no. 4, 2011.

[4] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications magazine*, vol. 40, no. 8, pp. 102–114, 2002.

[5] X. Fang, H. Gao, J. Li, and Y. Li, "Approximate multiple count in wireless sensor networks," in *IEEE INFOCOM*, 2014.

[6] C. Schurgers and M. B. Srivastava, "Energy efficient routing in wireless sensor networks," in *IEEE MILCOM*, vol. 1, 2001.

[7] C.-R. Dow, P.-J. Lin, S.-C. Chen, J.-H. Lin, and S.-F. Hwang, "A study of recent research trends and experimental guidelines in mobile ad-hoc network," in *IEEE AINA*, vol. 1, 2005.

[8] S. Bandyopadhyay and E. J. Coyle, "An energy efficient hierarchical clustering algorithm for wireless sensor networks," in *IEEE INFOCOM*, vol. 3, 2003.

[9] O. Younis and S. Fahmy, "Distributed clustering in ad-hoc sensor networks: A hybrid, energy-efficient approach," in *IEEE INFOCOM*, 2004.

[10] J. Ma, W. Lou, Y. Wu, M. Li, and G. Chen, "Energy efficient TDMA sleep scheduling in wireless sensor networks," in *IEEE INFOCOM*, 2009.

[11] Y. Zhuang, J. Pan, and L. Cai, "Minimizing energy consumption with probabilistic distance models in wireless sensor networks," in *IEEE INFOCOM*, 2010.

[12] M. Alaei and J. M. Barcelo-Ordinas, "Node clustering based on overlapping fovs for wireless multimedia sensor networks," in *IEEE WCNC*, 2010.

[13] O. Younis and S. Fahmy, "Heed: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks," *IEEE Transactions on Mobile Computing*, vol. 3, no. 4, pp. 366–379, 2004.

[14] A. D. Amis, R. Prakash, T. H. Vuong, and D. T. Huynh, "Max-min d-cluster formation in wireless ad hoc networks," in *IEEE INFOCOM*, 2000.

[15] Y. Fernandess and D. Malkhi, "k-clustering in wireless ad hoc networks," in *ACM POMC*, 2002.

[16] F. G. Nocetti, J. S. Gonzalez, and I. Stojmenovic, "Connectivity based k-hop clustering in wireless networks," *Telecommunication systems*, vol. 22, no. 1-4, pp. 205–220, 2003.

[17] B. N. Clark, C. J. Colbourn, and D. S. Johnson, "Unit disk graphs," *Annals of Discrete Mathematics*, vol. 48, pp. 165–177, 1991.

[18] P.-J. Wan, K. M. Alzoubi, and O. Frieder, "Distributed construction of connected dominating set in wireless ad hoc networks," in *IEEE INFOCOM*, 2002.

[19] W. Wu, H. Du, X. Jia, Y. Li, and S. Huang, "Minimum connected dominating sets and maximal independent sets in unit disk graphs," *Theoretical Computer Science*, vol. 352, no. 1, pp. 1–7, 2006.

[20] X. Gao, Y. Wang, X. Li, and W. Wu, "Analysis on theoretical bounds for approximating dominating set problems," *Discrete Mathematics, Algorithms and Applications*, vol. 1, no. 01, pp. 71–84, 2009.

[21] M. Li, P.-J. Wan, and F. Yao, "Tighter approximation bounds for minimum CDS in unit disk graphs," *Algorithmica*, vol. 61, no. 4, pp. 1000–1021, 2011.

[22] Y. Du and H. Du, "A new bound on maximum independent set and minimum connected dominating set in unit disk graphs," *Journal of Combinatorial Optimization*, 2013, online Published.

[23] T. Vuong and D. Huynh, "Adapting d-hop dominating sets to topology changes in ad hoc networks," in *IEEE ICCCN*, 2000.

[24] T. N. Nguyen and D. T. Huynh, "Connected d-hop dominating sets in mobile ad hoc networks," in *IEEE WiOpt*, 2006.

[25] D. Cokuslu and K. Erciyes, "A hierarchical connected dominating set based clustering algorithm for mobile ad hoc networks," in *IEEE MASCOTS*, 2007.

[26] M. Q. Rieck, S. Pai, and S. Dhar, "Distributed routing algorithms for multi-hop ad hoc networks using d-hop connected d-dominating sets," *Computer networks*, vol. 47, no. 6, pp. 785–799, 2005.

[27] X. Li and Z. Zhang, "Two algorithms for minimum 2-connected r-hop dominating set," *Information Processing Letters*, vol. 110, no. 22, pp. 986–991, 2010.

[28] X. Gao, W. Wu, X. Zhang, and X. Li, "A constant-factor approximation for d-hop connected dominating sets in unit disk graph," *International Journal of Sensor Networks*, vol. 12, no. 3, pp. 125–136, 2012.

[29] Z. Zhang, Q. Liu, and D. Li, "Two algorithms for connected r-hop k-dominating set," *Discrete Mathematics, Algorithms and Applications*, vol. 1, no. 04, pp. 485–498, 2009.

[30] X. Zhu, J. Li, Y. Xia, X. Gao, and G. Chen, "An efficient distributed node clustering protocol for high dimensional large-scale wireless sensor networks," in *ACM ICUIMC*, no. 4, 2014.

[31] W. Wang, D. Kim, N. Sohaee, C. Ma, and W. Wu, "A PTAS for minimum d-hop underwater sink placement problem in 2d underwater sensor networks," *Discrete Mathematics, Algorithms and Applications*, vol. 1, no. 2, pp. 283–289, 2009.

[32] D. Kim, W. Wang, N. Sohaee, C. Ma, W. Wu, W. Lee, and D.-Z. Du, "Minimum data latency bound k-sinks placement problem in wireless sensor networks," *IEEE/ACM Transactions on Networking*, vol. 19, no. 5, pp. 1344–1353, 2011.