A Constant Factor Approximation for the d-Hop Connected Dominating Set in Three-Dimensional Wireless Networks

Ke Li, Xiaofeng Gao, *Member, IEEE*, Fan Wu, *Member, IEEE*, and Guihai Chen, *Member, IEEE*

***Abstract*— In the past few years, wireless sensor networks (WSNs) have been widely used in many areas. In these applications, sensors are remotely deployed to gather related environmental information for further analysis. To support higher scalability and better data aggregation, sensor nodes are often grouped into disjoint and mostly non-overlapping clusters. All nodes in a cluster can send their data to the cluster head within d-hop distance, and the head should communicate with other cluster heads and pass all data to base station. For better communication between these cluster heads, lower maintenance cost, and easier management, it is necessary to make the number of the clusters as small as possible. Moreover, in many environments such as mountainous area or underwater monitoring, node deployment is often not flat, resulting in a high dimensional network. In this paper, we focus on proposing a scheme to select cluster heads for a homogeneous network in three-dimensional situation. The scheme meets two requirements: the number of cluster heads is minimum and the head nodes can communicate with each other. These requirements can be formed as an NP-complete problem named *d-hop connected dominating set*. Correspondingly, we proposed a distributed approximation algorithm and proved its approximation ratio as $(d+1)\beta$, where β is a calculated parameter with respect to d. We also analyzed the performance of our algorithm with corresponding numerical experiments.**

***Index Terms*— Wireless sensor network, d-CDS, spanning tree, cluster, distributed algorithm.**

I. Introduction

WIRELESS sensor networks (WSNs) are autonomous and self-organized communication systems consisting of many small, inexpensive, and battery-powered embedded devices called sensor nodes, each with sensing, computing, communication capabilities. These sensor nodes serve not only as mobile hosts but also as routers. Because of such characteristics, WSNs can be widely used in lots of applications such as disaster management, battlefield reconnaissance, border patrol and surveillance, etc. In these applications, sensor nodes are mainly used to gather vital information from the surrounding areas such as temperature, sound, vibration, pressure, motion or pollutants, etc. Also, sensor nodes need to transmit their collected data to base stations directly or indirectly. The related researches have been widely studied in past decades [1]–[11].

In wireless sensor networks, sensor nodes have constraints in terms of processing power. In most cases, the unattended nature of WSNs makes it quite difficult to recharge node batteries or replace sensor nodes. Therefore, energy conservation is a major design goal in these networks. To prolong the lifetime of WSNs, a considerable number of researches have been done in [4], [7], [8], [10]–[12]. In addition, sensor nodes are limited by communication bandwidth and storage space, which makes it difficult to transmit message throughout the whole network and to process a mass of data. In order to overcome such shortcomings, the efficient approach that is commonly agreed by most of researchers is clustering. We can divide the whole network into disjoint and mostly non-overlapping sets called clusters. Each cluster elects a leader called cluster head. All ordinary nodes in a cluster transmit their data to the cluster head, and the head passes all data to the base station. Additionally, each cluster header can communicate with other cluster heads. Clustering schemes offer reduced communication overheads and efficient resource allocations, thus decreasing the overall energy consumption and reducing interferences among sensor nodes.

Numbers of clustering algorithms have been proposed. Based on the fact that we can improve the system's lifetime by improving each cluster's lifetime, [4] alternates cluster head in a cluster to maximize the cluster's lifetime and finally determines each cluster and the corresponding head. In [5], the authors proposed a clustering algorithm which based on cell combination for the networks. In [6], through coordination of nodes belonging to the same cluster which can effectively avoid redundant sensing or processing, Alaei et al. proposed a clustering method to optimize the energy conservation and prolong network lifetime. Other related clustering algorithms are referred to in [8], [10], [12].

Easy to know, small size clusters lead to a large number of clusters, which will congest the area while a few number of

Manuscript received October 18, 2018; revised March 27, 2019; accepted June 4, 2019. Date of publication July 17, 2019; date of current version September 10, 2019. This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFB1004703, in part by the National Natural Science Foundation of China under Grant 61872238, Grant 61672353, and Grant 61672348, in part by the Shanghai Science and Technology Fund under Grant 17510740200, in part by the Huawei Innovation Research Program under Grant HO2018085286, in part by the State Key Laboratory of Air Traffic Management System and Technology under Grant SKLATM20180X, in part by the Tencent Social Ads Rhino-Bird Focused Research Program, in part by the Open Project Program of the State Key Laboratory of Mathematical Engineering and Advanced Computing under Grant 2018A09, and in part by the Alibaba Group through the Alibaba Innovation Research Program. The associate editor coordinating the review of this paper and approving it for publication was J. Tang. *(Corresponding author: Xiaofeng Gao.)*

The authors are with the Shanghai Key Laboratory of Scalable Computing and Systems, Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: like19971019@sjtu.edu.cn; gao-xf@cs.sjtu.edu.cn; fwu@cs.sjtu.edu.cn; gchen@cs.sjtu.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TWC.2019.2923744

clusters will exhaust the cluster head with large amounts of messages transmitted from cluster members. An effective way to control the average size of clusters is to assume that each cluster head can be at most d-hop away from the nodes within its dominating range, where d can be manually set based on the real circumstance. When sensor nodes in a network are uniformly and independently distributed, [3] gives a solution to evaluate the average hop distance. In this paper, we focus on constructing d-hop clusters for a given WSN.

Usually, the set of cluster heads can be viewed as a *connected dominating set* (CDS) of graph G. For a given graph $G = (V, E)$, a CDS of G is a subset $C \subseteq V$ such that for each vertex $v \in V \backslash C$, there exists a vertex $u \in C$ satisfying $(u, v) \in E$. Moreover, the subgraph induced by C is connected. To construct d-hop clusters for a given WSN is equivalent to choose a d-CDS for a given graph. To reduce the message redundancy, finding a minimum d-CDS is important.

In most cases, people assume that wireless nodes are on a 2D plane, and use a *unit disk graph* (UDG) to model the network. However, in many environments like mountainous areas or underwater regions, node deployment is often not flat, resulting in a 3D network. Given that communication between nodes in high dimensional space is more complicated and restricted with the environment, clustering algorithms in high dimensional space tend to have higher time complexity. What is more, many optimal algorithms in 2D space cannot directly apply to high dimensional situation. For example, some polynomial time approximation schemes (PTAS) in existing works for CDS problem in UDG cannot be directly generalized to generate a PTAS for minimum CDS in UBG because their proofs rely on some geometrical properties that hold true in the plane but are not true in space any more [13]. Therefore, it is challenging to design an efficient clustering algorithm in high dimensional space. In this paper, we propose such an algorithm with minimum d-CDS, using a *unit ball graph* (UBG) to model networks in 3D space.

Given a graph without its geometric representation, it is NP-hard to determine whether it can be represented as a UDG or a UBG [14]. Since the UDG can be regarded as a special case of UBG, we can just view wireless networks as UBG and solve problems in 3D space. As far as we know, there is not yet specialized works on d-CDS in unit ball graph before this paper. The only related work is the work done by Kim *et al.* [15] which can be seen as 1-CDS example in UBG.

Our goal in this paper is to partition a wireless sensor network into d-hop clusters by finding a minimum d-CDS for a UBG $G = (V, E)$ derived from the given network in a 3D space. In this paper, we proposed a three-phase distributed algorithm to find a minimum d-CDS. The first part of this algorithm is to construct a spanning tree from a given network. Then we select a d-MIS based on the spanning tree. Afterwards, we add some extra nodes to connect the d-MIS to make it a d-CDS.

Our contributions are threefold:

- As far as we know, this is the first work focusing on finding a minimum d-CDS in three-dimensional situation.
- Since distributed algorithms become more and more important for self-organized WSNs, we proposed a distributed algorithm for minimum d-CDS problem. As for the details of our algorithm, we first introduce concepts of "level id" and "neighbor information" to overcome some bug in designing similar distributed algorithms.
- As the minimum d-CDS problem is NP-complete which is proved by Vuong and Huynh in [16], we analyzed the performance of our algorithm and provided approximation ratio for our algorithm. Moreover, the approximation ratio is proved to be $(d + 1)\beta$, where β is a parameter given in Sec. VI.

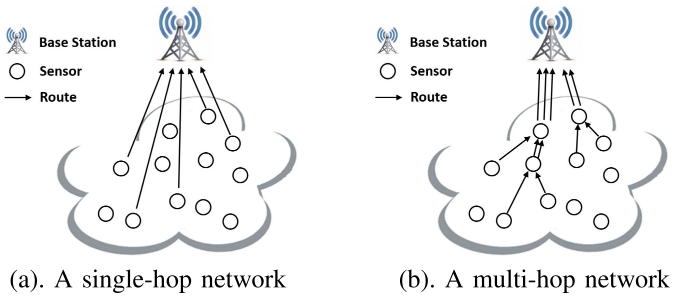


Fig. 1. Different wireless sensor networks with different hops.

The rest of this paper is organized in the following structure: Sec. II introduces some background information. In Sec. III, we describe the related work in detail. Definitions and notations that will be used in later sections are illustrated in Sec. IV. In Sec. V, we present our three-phase distributed algorithm to select a d-CDS for the given graph and discuss the redundancy of the d-CDS we choose. Afterwards, the corresponding performance analysis is described in Sec. VI and Sec. VII gives simulation settings and results. Finally, Sec. VIII concludes this paper.

II. Background Information

A. Types of Sensing Networks

As we can see in Fig. 1, sensors transmit gathered data directly to the base station in a single-hop network while some nodes in a multi-hop network function as the routers for other nodes to deliver their gathered data through several hops. In summary, in single-hop networks, sensors directly deliver their sensed data to the sink. In multi-hop networks, sensors transmit their sensed data to the sink using intermediate nodes [17]. In our paper, our clustering network belongs to multi-hop networks.

B. Unit Ball Graph

The topology of wireless sensor networks is often modeled as a unit disk graph (UDG), a geometric graph in which there is an edge between two nodes if and only if their Euclidean distance is at most 1. Fig. 2 gives an example of modeling a WSN using a UDG, where each node has the same communication range denoted as a circle and there exists an edge between two nodes if and only if the two circles are intersectant or tangent, meaning they are in the communication range of each other.

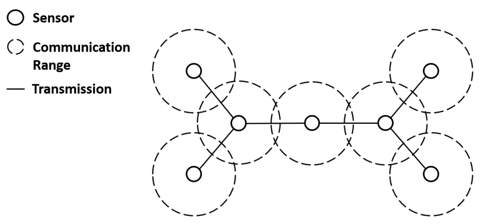


Fig. 2. Model a WSN using a unit disk graph (UDG).

When it comes to a 3D environment like mountainous areas or underwater regions, UDG is generalized to UBG with similar properties. Since UDG is a reduced form of UBG in 2D space, we adopt UBG in our paper to address the problems and the techniques can directly apply to 2D situations. We give the formal definition of UBG as follows.

*Definition 1: A given graph $G = (V, E)$ is a **UBG (unit ball graph)** if each node in V has the same communication range, denoted as a ball, and two vertices are connected to each other in E if and only if the Euclidean distance between them is less than or equal to 1.*

III. Related Work

Since our goal is to find a minimum d-CDS for a given graph $G = (V, E)$, we will introduce some related work about CDS in this section.

Clark *et al.* [18] proved that the MCDS problem is NP-hard even in UDG. Hence, a commonly used approach is to use approximation algorithms to solve such problems. In 2002, Wan et al. [19] first found that MCDS problem has polynomial-time constant-factor approximation solution and proposed a two-phase algorithm to select a CDS. Based on this design, a lot of similar two-phase algorithms were proposed in literature. All these algorithms first choose a maximal independent set (MIS), and the second phase is to add some extra nodes to connect them. An MIS for a graph $G = (V, E)$ is a subset $M \subseteq V$ such that any two vertices in M are not directly connected, and if we insert a node u from $V \backslash M$ into M, M will not be an MIS any more. Obviously, MIS is also a dominating set (DS) [20]. Besides, the related theoretical approximation ratio of such algorithms are also been widely studied [21]. On the other side, Cheng et al. [22] first found that MCDS problem has PTAS solutions. Even though PTAS has a better approximation ratio, the time cost in [22] is too high to implement in reality. Hence, the main focus of this field is still on constructing effective approximation algorithms to find a feasible solution within polynomial time. Except for those maximal independent set-based algorithms, there also exist other kinds of algorithms, such as greedy algorithms [23], Steiner tree-based algorithms [2], pruning-based algorithms [24] and connected clustering-based algorithms [25].

As for the two-phase algorithm, it is very common to apply the color-marking algorithm to first select an MIS, which is also a DS from a given graph $G = (V, E)$ [19]. After we have obtained a DS, connecting this DS into a CDS is equivalent to finding a Steiner Tree for G. The formal definition of Stenner Tree is as follows: Given a graph $G = (V, E)$, a selected subset S, a Steiner Tree is a tree in G includes all vertices in S. In those MIS-based algorithms, Steiner Tree is commonly used in the connecting part. The detailed introduction about Steiner Tree used in constructing MCDS can refer to [2].

In a wireless sensor network, implementing cluster-based hierarchical structure is much more helpful to achieve efficient routing, increase the lifetime of networks and improve the network's scalability. Furthermore, such structure can be modeled as a d-hop CDS. Most of related works on d-hop CDS were finished within recent decades. In 2006, Nguyen and Huynh [33] first proved that finding a minimum d-hop CDS in UDG is NP-complete. For a period of time since then, researchers presented some heuristic algorithms to solve d-hop CDS problem or some problems with specific d, such as TCDS (2-hop CDS) and 3-CDS. They also used several numerical examples to analyze the performance of their algorithms. These d-CDS algorithms are utilized in many fields. In 2016, Azizian *et al.* [34] adopted a d-hop clustering algorithm in vehicular ad-hoc networks (VANETs) to control resource sharing and management functions and took the relative mobility of nodes in d-hop communication range into consideration. However, all these works lacked exact approximation analysis, especially the quantitative description about the gap between optimal solution and feasible solution.

Until recently, some theoretical researches came out. In [9], Gao et al. proposed a two-phase approximation algorithm to compute a d-hop CDS in UDG with a constant-factor approximation which is relevant with $O(d^3)$. Later, Zhang et al. [28] improved the approximation ratio into $O(d^2)$ level. In 2017, Wang et al. [35] proposed a new polynomial time constant factor approximation algorithm for the minimum 4-connected m-dominating set problem in UDG with any positive integer $m \geq 1$. Coelho et al. [30] proved that letting G be a graph, if there exists an algorithm for minimum CDS problem with approximation factor $\alpha(G)$, then there will be a $d\alpha(G^d)$-approximation algorithm for minimum d-CDS problem for any $d \in Z_>$. Lately, Gao et al. [31] designed a distributed approximation with the approximation ratio $(2d+1)\lambda$, where λ is a parameter related with d but is no more than 18.4 to address the minimum d-CDS problem under UDG. Compared with the previous best result $O(d^2)$, it is a great improvement. However, their approximation ratio is proved in UDG where the d-hop neighbors of one node are much fewer and the edge relationship between nodes is much less complicated than UBG so our $(d+1)\beta$-approximation algorithm can better deal with the more complex networks in 3D space.

Tab. I summarizes a few characteristics of some existing d-hop CDS algorithms, including the sources, the number of hops, local maintenance situation, time complexity information, message complexity, and approximation ratio analysis.

By investigating, we found that no previous d-hop CDS algorithm was designed in UBG. Obviously, UBG can formulate a network environment more precisely than UDG because UBG model can reflect more details of the real world. Although the design of algorithms for MCDS in UBG seems to be similar with the design in UDG, some algorithms in UDG cannot be used to solve problems in 3D space efficiently

TABLE I

SUMMARY OF d-HOP CDS CONSTRUCTION

Source	Hop	Local Info	Time complexity	Msg. complexity	Approximation ratio
[26]	2	Distributed	$O(10)$	$O(\|V\|^2)$	None
[27]	d	Centralized	None	None	$\begin{cases} (2d+1)^3, & k \le (2d+1)^2, \\ (2d+1)((2d+1)^2+1), & k > (2d+1)^2. \end{cases}$
[27]	d	Centralized	None	None	$(2d+1)ln(\gamma)$
[28]	d	Centralized	None	None	$\begin{cases} (d+1)\beta - d, & k \le \beta, \\ (d+1)(\beta+1) - d, & k > \beta. \end{cases}$
[29]	d	Centralized	$O(\|V\|^2\|E\|log_{(1+\|E\|/\|V\|)}\|V\|)$	None	$(PD(G)-1) \times H(\gamma)$
[9]	d	Distributed	$O(\|V\|)$	$O(\|V\|log\|V\|)$	$(0.335r^3 + 1.337r^2 + 0.585r)$
[30]	d	Centralized	None	None	$d(1+\epsilon)(1+ln(\Delta(G^d)-1))$
[31]	d	Distributed	None	None	$(2d+1)\lambda$
[32]	d	Distributed	$O(\delta_1\delta_{d+1})$	$O(d\|V\|\delta_1)$	None

Note: k: every node v not in CDS has at least k neighbors within d hops in CDS,
$r = d + 0.5$,
$PD(G) = max\{dist(u : v, w) | u, v, w$ are three distinct vertices in $V\}$,
λ: related with d but no more than 18.4,
$N^d(v)$: neighbors of node v within d hops,
$\gamma = \max\{|N^d(v)| : v \in V\}$,
$H(\gamma) = \sum_{i=1}^{\gamma} \frac{1}{i}$,
$\Delta(\cdot)$: the maximum vertex-degree of the input graph,
$\delta_d = \max\{|N^d(v) \setminus N^{d-1}(v)| : v \in V\}$,

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

ϵ: any fixed positive constant satisfying $0 < \epsilon \le 1$.

because their proofs depend on specific geometrical properties that only hold true in the plane and the analysis part of these approximation algorithms could be much harder in UBG than that in UDG. Because of such difficulty, few papers study MCDS approximation in 3D space. Actually, in most cases, when people begin to study MCDS in UBG, they usually modify and generalize the corresponding approximation algorithms in 2D space, such as the two-phase algorithms mentioned above. In [36], Butenko and Ursulenko first proved that the ratio between the size of MIS and MCDS is at most 11 which leads to an approximation ratio of 22 for MCDS in UBG. Later, Kim *et al.* [15] improved that ratio into 14.937.

As far as we know, there is few work studying minimum d-hop CDS in 3D space. This paper is the first relevant research.

IV. PRELIMINARIES

As mentioned before, for any given graph, we hope to find a minimum d-CDS to gather data and cluster the network. When designing algorithms for CDS, many researchers adopt a two-step algorithm as follows:

1) Construct a maximal independent set (MIS). It is easy to see that an MIS for a graph G is also a DS for G.
2) Connect this MIS into a CDS.

We follow a similar pattern in the design of our d-CDS algorithm. We first select a d-MIS, then connect this d-MIS into a d-CDS. The following are several definitions we will use in solving the d-MCDS problem. Besides, we give the relation between d-MIS and d-DS.

*Definition 2: A d-**DS (dominating set)** for a given graph $G = (V, E)$ is a vertex set C such that any node in V is either in set C or connected to a node in set C within d hops.*

*Definition 3: A d-**IS (independent set)** for a given graph $G = (V, E)$ is a vertex set I such that for any pair of nodes in I, the distance between them is greater than or equal to d hops.*

*Definition 4: A d-**MIS (maximal independent set)** for a graph $G = (V, E)$ is a d-IS such that if we insert any vertex from $V \setminus I$, I is no longer a d-IS.*

It has been recognized by many researchers that it is more efficient and convenient to find a d-MIS rather than a d-DS for a given graph G. Thus, we need to find the relation between d-MIS and d-DS. By definition, it is easy to get Lem. 1.

Lemma 1: A d-MIS for a graph G is also a d-DS for G.

Proof: We prove it by contradiction. Suppose I is a d-MIS for $G(V, E)$ and there exists a node v which is not d-hop dominated by I, meaning the distance from v to any node of I is greater than d. Then we can insert v into I, contradicting the fact that I is a d-MIS. Thus I is a d-DS for G. □

*Definition 5: A d-**CDS (connected dominating set)** for a given graph $G = (V, E)$ is a vertex set C such that C is a d-DS and the subgraph induced by C is connected.*

Our target is to find a minimum d-CDS for a given graph G to reduce the message redundancy, lower maintenance costs and obtain better communication between nodes in the d-CDS.

Next we introduce some useful notations and terminologies. In the following sections, the distance between any two nodes u, v always means the smallest number of hops needed from u to v, and we use $dis(u, v)$ to denote this distance. For any node u, we use d-hop neighbors for u to denote the set of nodes within d hops from u (except itself), i.e.,

$$N^d(u) = \{v \in V \mid 1 \le dis(u, v) \le d\}.$$

V. A DISTRIBUTED ALGORITHM

In this section, we will introduce our three-phase distributed algorithm for minimum d-CDS problem in 3D space. Our distributed algorithm has three subroutines, which is a generalization of the algorithm described in [9], [19], [37]. Different from traditional two-phase algorithms, we first construct a spanning with some initialization of each node in Subsec. V-A before finding a d-MIS. Based on the spanning tree, we construct a d-MIS in Subsec. V-B, and then insert additional nodes to connect the selected nodes as a d-CDS in Subsec. V-C. What is more, we discuss the redundancy of vertices in the d-CDS produced by our algorithm in Subsec. V-D and give an example in Subsec. V-E.

Specifically, all previous works failed to clarify the details of message interchange processes in their distributed designs, which might easily bring deadlock and termination problems. Our algorithm design avoids such problems, which can be implemented in any asynchronous systems.

A. Constructing a Spanning Tree

Given a UBG $G = (V, E)$, we select an arbitrary root r (usually with maximum node degree and locates at the center of the network) and design a spanning tree construction algorithm with time complexity $O(n)$ and message complexity $O(n \lg n)$, inspired by the distributed leader-election algorithm mentioned in [38]. Reference [38] uses a special form of information propagation with feedback (PIF) to select a leader from several fragments. We borrow the PIF process to construct a spanning tree and exchange information among d-hop neighbors.

There are many spanning tree construction algorithms studied in existing works, but most of them are designed to achieve minimum energy or minimum latency and few can obtain knowledge of d-hop neighbors while constructing the spanning tree. Thus, our algorithm is novel and $O(n)$ time complexity performs better than many traditional spanning tree algorithms, such as DFS, BFS, Prim algorithm, and Kruskal algorithm.

In our spanning tree algorithm, every node n_i has a distinct id, and uses a variable $parent$ to store the id of its parent node in the spanning tree. In the meanwhile, based on this spanning tree, we calculate the $level$ of each node, which is the number of hops from root to this node. As for the root r, $level = 0$. Then we can give a rank to every node by using the pair of its $level$ and its id, $(level, id)$. For any two node n_x, n_y, suppose $n_x.level$, $n_y.level$ is their $level$ information, $n_x.id$, $n_y.id$ is their id information, then n_x has a lower rank than n_y if and only if one of the following conditions holds:

1) $n_x.level < n_y.level$; or
2) $n_x.level = n_y.level$ and $n_x.id < n_y.id$.

Besides, each n_i has a variable $n_i.children$ for counting its children and maintains an intermediate variable $n_i.x$ to count its unterminated children. It also uses a set $n_i.nb$ to record its d-hop neighbors. Each entry of nb is the in the form of $(id_u, level_u, hop_u)$, representing a d-hop neighbor u, hop_u is the smallest number of hops from u to this node n_i itself.

Then the detailed description of determining the $level$ of each node can be shown in Alg. 1. The intermediate variable $n_i.x$ counts the number of children which have not broadcast $complete$ message and is thus initialized to the $children$ of n_i. The set $n_i.nb$ is used to record its d-hop neighbors' id, $level$, and how many hops they are from the node n_i at least. As a result, $n_i.nb$ is initially empty.

First, once the rooted spanning tree mentioned above is constructed, the root r announces its $level$ 0 by broadcasting a $level$ message. Second, a node n_i will record the sender's id, $level$, and the smallest number of hops from the sender to the node n_i itself into $n_i.nb$ after receiving a $level$ message. If the sender is n_i's parent in the spanning tree, it sets its own $level$ as 1 plus the sender's $level$, and broadcasts a $level$ message to announce its $level$. If n_i is a leaf in the spanning tree, which means $n_i.children$ is equal to 0 and its own $level$ has been determined, it will send a $complete$ message to its parent and terminate its level construction process. Third, upon receiving a $complete$ message, n_i decreases the number of its unterminated children $n_i.x$ by 1 if the message is transmitted from its children. Once $n_i.x$ is reduced to 0 and n_i is not the root r, it transmits a $complete$ message to its parent as well and terminates its own process. Finally, when the intermediate variable $n_i.x$ is equal to 0 at the root, the whole algorithm comes to an end and the level construction process is finished.

So far, each node n_i knows its own $level$ in the spanning tree and records information of some of its d-hop neighbors in $n_i.nb$. Afterwards, we will complement the rest of n_i's d-hop neighbors into nb in Alg. 2.

Algorithm 1 Level Construction (Run on Each n_i)

```
1  $n_i.level = 0$;                                    ▷ Initialization
2  $n_i.x = n_i.children =$ number of $n_i$'s children;
3  $n_i.nb = \varnothing$;
4  if $n_i$ is root then
5      $n_i.level = 0$;
6      broadcast $level(n_i, n_i.level)$;              ▷ Marking Levels
7  if receive $level(n_j, n_j.level)$ then
8      push $(n_j, n_j.level)$ into $n_i.nb$;
9      record the smallest number of hops from $n_j$ to $n_i$ into
       $n_i.nb$;
10     if $n_i.parent = n_j$ then
11         $n_i.level = n_j.level + 1$;
12         broadcast $level(n_i, n_i.level)$;
13     if $n_i.children = 0$ and $n_i.level \neq 0$ then
14         broadcast $complete(n_i, n_i.parent)$;
15         terminate;                                  ▷ Termination
16 if receive $complete(n_j, n_j.parent)$ then
17     if $n_i = n_j.parent$ then $n_i.x = n_i.x - 1$;
18     if $n_i.x = 0$ and $n_i \neq r$ then
19         broadcast $complete(n_i, n_i.parent)$;
20         terminate;                                  ▷ Termination
```

Alg. 2 is used to exchange the d-hop information among nodes in the network. In this process, we use a simple distributed algorithm to discover the d-hop neighbors for each node. The key idea is, suppose each node has already recorded its k-hop neighbors in nb, then if we let all nodes exchange this information with its 1-hop neighbors, then each node can discover its $(k+1)$-hop neighbors. Since each node has already recorded the information of its 1-hop neighbors in nb after Alg. 1, if all nodes exchange their nb for $(d-1)$ rounds, then each node can get its all d-hop neighbors.

The detailed description is in Alg. 2. $n_i.received$ includes the id of all the nodes received by n_i in this round while

Algorithm 2 d-hop Neighbor Discovering

```
1 $n_i.received = \varnothing$;                                ▷ Initialization
2 broadcast $discover(n_i.id, n_i.nb)$;
3 if receive $discover(n_j.id, n_j.nb)$ then
4     if $n_j.id$ is not in $n_i.received$ then
5         push $n_j.id$ to $n_i.received$;
6         forall the $(n_k.id, n_k.level)$ in $n_j.nb$ do
7             if $(n_k.id, n_k.level)$ is not in $n_i.nb$ then push $(n_k.id, n_k.level)$ to $n_i.nb$ ;
8         record new $(n_k.id, n_k.level)$'s $n_k.hop$ in $n_j.nb$ to $n_i.nb$;
9 if $length(n_i.received) = n_i.children + 1$ then terminate ;
```

$n_k.hop$ denotes the smallest number of hops that n_k is away from n_i. We run Alg. 2 ($d-1$) times to get d-hop neighborhood information for our three-phase distributed algorithm.

After Alg. 2, we can calculate the number of lower-ranked d-hop neighbors of each node n_i from $n_i.nb$, which will be used later in our distributed algorithm.

By this time, all nodes know the $level$ and $rank$ of its own and all its d-hop neighbors. Hence, we can move on to the construction of a d-MIS by a color-marking process in the next subsection with all the initialization completed above.

B. Constructing a d-MIS

In this subsection, we will introduce subroutine 2 of our three-phase distributed algorithm. Subroutine 2 is a coloring process to select a d-MIS (also a d-DS) for the given network. All the nodes are initially colored white, and will be colored black or grey eventually. We use three special variables $WHITE$, $BLACK$, $GREY$ to denote these three colors.

First, when a node u colors itself black, it will broadcast a $black$ message, which will reach all nodes in $N^{d+1}(u)$, and each node v in $N^d(u)$ will color itself grey and broadcast a $grey$ message which will also reach all nodes in $N^d(v)$. Each node in $N^{d+1}(u) \setminus N^d(u)$ will record the path to u in $blackpath$ if it is still white and its $blackpath$ has not been determined. Second, once a node knows that all its lower-ranked d-hop neighbors have been colored grey already, it will color itself black and broadcast a $black$ message. Third, when a leaf node has already been colored black or grey, it transmits a $colored$ message to its parent. After receiving a $colored$ message, n_i will decrease its $n_i.x$ by 1 if the message is broadcast from its children. If $n_i.x$ is equal to 0 and n_i is not the root r, the node n_i will transmit a $colored$ message to its parent as well. Finally, by the time the variable $n_i.x$ is 0 at the root, the whole algorithm comes to an end and all nodes are colored black or grey eventually.

All the black nodes will make up a d-MIS for G after this subroutine. The detailed description is shown in Alg. 3. We still use $n_i.x$ to count the unterminated children of n_i. With a d-MIS constructed, we then insert additional nodes to connect the selected nodes as a d-CDS in subroutine 3 in Subsec V-C.

Algorithm 3 d-MIS Coloring

```
1 $n_i.color = WHITE$;                                        ▷ Initialization
2 $n_i.x = n_i.children$;
3 if $n_i$ is root then broadcast $black(\{n_i.id\})$;
4 if receive $black(path)$ then
5     if $length(path) \le d$ then
6         push $n_i.id$ to $path$;
7         broadcast $black(path)$;
8         if $n_i.color = WHITE$ then
9             $n_i.color = GREY$;
10            broadcast $grey(n_i.id, d)$;
11    else if $length(path) = d + 1$ and $n_i.color = WHITE$ and $n_i.blackpath$ is not set then $n_i.blackpath = path$ ;
12 if receive $grey(n_j.id, k)$ and $k > 1$ then
13     mark $n_j.id$ as colored in $n_i.nb$;
14     broadcast $grey(n_j.id, k - 1)$;
15 if all lower rank $d$-hop neighbors are colored grey and $n_i.color = WHITE$ then
16     $n_i.color = BLACK$;
17     broadcast $black(\{n_i.id\})$;
18 if $n_i.children = 0$ and $n_i.color \neq WHITE$ then
19     broadcast $colored(n_i.parent)$;
20     terminate;                                              ▷ Termination
21 if receive $colored(n_j.parent)$ and $n_i.id = n_j.parent$ then
22     $n_i.x = n_i.x - 1$;
23     if $n_i.x = 0$ then
24         if $n_i$ is not root then broadcast $colored(n_i.parent)$ ;
25         terminate;                                          ▷ Termination
```

C. Connecting a d-MIS Into a d-CDS

In subroutine 3, we will connect d-MIS into a d-CDS. For each black node except the root, we color the nodes in its $blackpath$ as black to connect it with another black node.

The function $pop(blackpath)$ means to take out the last entry of $blackpath$. After this subroutine, all black nodes will make up a d-CDS for G. The detailed description is in Alg. 4.

The three subroutines form our algorithm and we will refer to it as the d-CDS algorithm in the following sections. Alternatively, instead of connecting each black dominator directly via $blackpath$, we may use distributed Steiner tree algorithm to connect black nodes (denoted as Steiner nodes). A comparison between the solution produced by our algorithm and the optimal solution is shown in Fig. 3 (notice that we use blue nodes in Fig. 3(c) to denote the nodes added in Alg. 4).

D. Discussion on Redundancy

It is trivial to notice that the d-CDS produced by our algorithm suffers from some redundancy, i.e. we could remove some vertices safely and the remaining set is still a d-CDS, but such redundancy is hard to avoid in a distributed algorithm.

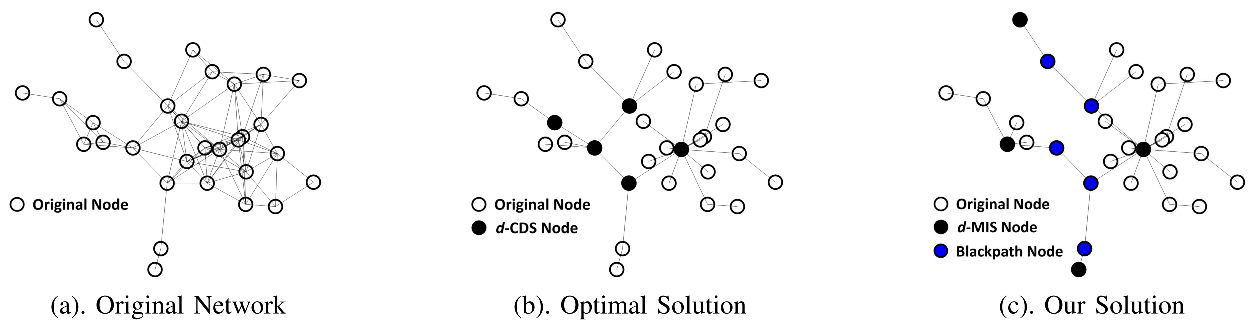


Fig. 3. Optimal solution and our solution.

Algorithm 4 d-CDS Connecting

```
1 if $n_i.color = BLACK$ then
2     $nexthop = pop(n_i.blackpath)$;
3     broadcast $join(nexthop, n_i.blackpath)$;
4 if $receive\ join(nexthop, blackpath)$ then
5     if $n_i.id = nexthop\ and\ blackpath \neq \varnothing$ then
6         $nexthop = pop(blackpath)$;
7         broadcast $join(nexthop, blackpath)$;
8         $n_i.color = BLACK$;
```

We could add one more subroutine to remove all such redundant vertices, but the cost is too high. To confirm a vertices v in a d-CDS C is redundant, we must check two conditions:

- The vertices dominated by v can be dominated by other vertices of C.
- Removing v does not change the connectivity of C.

The first condition is relatively easy to check, and can be accomplished in a distributed manner. The hard part is how to check the second condition. Unfortunately, there is still no efficient distributed algorithms concerning it, so we didn't provide a redundancy-removing subroutine in this paper.

E. An Example

In this subsection, we use an example to illustrate our algorithm. To keep it simple and precise, we just plot the example in two-dimensional space as is shown in Fig. 4.

Originally, we have a given UBG $G = (V, E)$ with 17 nodes, as is shown in Fig. 4(a). We want to find a 2-hop CDS for G. First, we need to construct a spanning tree T for G. As Fig. 4(b) shows, a spanning tree is formed according to the original graph with its root node labeled as 1. All other nodes are labeled from 2 to 17 as their unique id numbers to distinguish themselves from others. In this spanning tree, there exist two kinds of lines between two adjacent nodes, which are marked as solid lines or dashed lines. If there is a solid line or dashed line between two nodes u and v, it means there is an edge $(u, v) \in E$. A solid line means an edge in the spanning tree T, and a dashed line means an edge in the original graph G, but this edge is not included in T. Before actually processing coloring subroutine, each node should have already got its level and 2-hop neighbors' information.

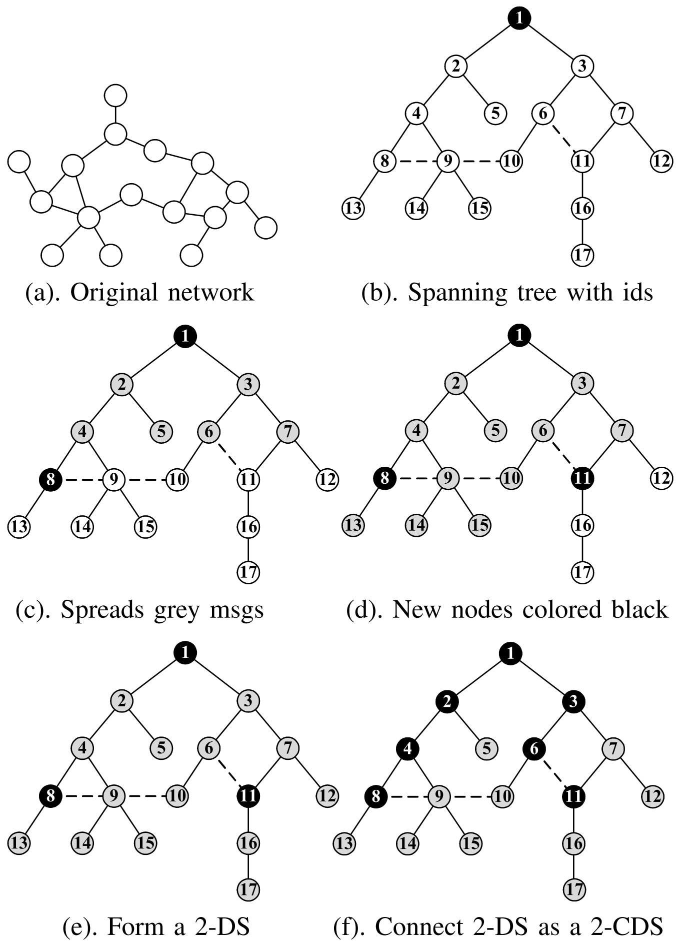

(a). Original network
(b). Spanning tree with ids
(c). Spreads grey msgs
(d). New nodes colored black
(e). Form a 2-DS
(f). Connect 2-DS as a 2-CDS

Fig. 4. An example to construct a 2-CDS with 17 nodes.

Second, we can start to color the network. In the coloring procedure as is shown in Alg. 3, initially all nodes are white. In Fig. 4(b), root r colors itself black, and broadcasts a $black$ message. In Fig. 4(c), upon receiving a $black$ message and detecting that there is a black node within its 2-hop distance, nodes 2, 3, 4, 5, 6 color themselves grey, and broadcast a $grey$ message to inform all their 2-hop neighbors that they have already been colored. At this time, node 8 notices that all its lower-ranked 2-hop neighbors have been colored grey (they are nodes 2, 4, 5, respectively), so node 8 colors itself black and broadcasts a $black$ message. In Fig. 4(d), nodes 9, 10, 13, 14, and 15 receive the $black$ message initiated from node 8, and thus color themselves as grey and broadcast $grey$ messages. Now node 11 can color itself black and broadcast a $black$ message, which will lead nodes 12, 16, and 17 to color themselves as grey in Fig. 4(e). Finally, the set of nodes $\{1, 8, 11\}$ is our selected 2-MIS for G, also a 2-DS for G.

Finally, Fig. 4(f) shows a possible way to connect these black nodes into a 2-CDS, since both node 8 and node 11 record the path from node 1 as their blackpaths. As a result, the set of nodes $\{1, 2, 3, 4, 6, 8, 11\}$ is our selected 2-CDS.

VI. Performance Analysis

We first prove the correctness of our d-hop CDS algorithm.

Theorem 1: d-CDS algorithm can successfully terminate with consistent agreement.

Proof: First, we prove the termination for the d-CDS algorithm above. In Alg. 1, from Line 4-12, we can see that "level" messages could be transmitted from every parent to its all children. Then, according to Line 13-15, when "level" messages reach to a leaf node, the leaf will terminate and tell its parent that it has terminated. When a parent receive all its children's "complete" messages, it turns to terminate and tell its own parent that it has terminated. Eventually, when the root terminates, all the nodes has terminated. As for Alg. 2, when each node exchanges neighbor information with all its one-hop neighbors, it turns to terminate. According Line 1-17 in Alg. 3, after enough time, all nodes will be colored either black or grey. From Line 18-20, leaf nodes will first terminate. Then leaf nodes will feed the messages of termination to their parents. Their parents will also terminate when all their children terminate according to Line 21-25. As for Alg. 4, the entire network will terminate when all $blackpaths$ pop out.

Next, we will prove the property of agreement. Since each message in d-CDS algorithm except for Alg. 3 is broadcasted only once by each node, it is easy to understand the agreement in this situation. Besides, in Alg. 3, Line 15 ensures the sequence of the chosen black nodes is deterministic. Hence, the algorithm must meet the requirement of agreement. □

Theorem 2: All black nodes after Alg. 3 form a d-MIS of G.

Proof: We noticed that every node must be colored as black or grey after Alg. 3, otherwise Alg. 3 cannot terminate.

Then we prove that the distance between any two black nodes is greater than or equal to d. We prove it by contradiction. Suppose there are two black nodes u and v with distance less than d. Because all the nodes are totally ordered by rank, without loss of generality we assume u has a higher rank than v. Then u cannot color itself unless it knows that v has been already colored. However, since the distance between u and v is less than d, in all $black$ messages initiated from v and reached u, there must be one traveling less than d hops. Then by Alg. 3, u will be colored grey, which yields a contradiction.

We also cannot insert any more black nodes because if we change any grey node into black, there must exist a black node within d hops. Thus all black nodes form a d-MIS for G. □

Theorem 3: All black nodes after Alg. 4 form a d-CDS of G.

Proof: Denote the set of black nodes after Alg. 4 as C. According to Lem. 1 and Lem. 2, the set of black nodes after Alg. 3 is a d-MIS, so it is a d-DS for G by Lem. 1. Hence we only need to prove the connectivity of C. In Alg. 4, we use the nodes in $blackpath$ to connect each black node (except the root) to another black node, the connectivity just follows. □

Then we move on to the discussion of the approximation ratio. First we find an upper bound for the number of independent vertices for in a node's d-hop neighbors. Then we use it to establish a connection between IS and DS for a graph. Finally, we prove the approximation ratio of our algorithm. During our derivation, we use Lem. 2 which is proved in [39].

Lemma 2: [39] For any vertex u in a UBG G, the neighborhood $N(u)$ contains at most 12 independent vertices.

Lemma 3: I is a d-IS of a UBG G, then for any vertex u, $N^d(u)$ contains at most β vertices from I, where

$$\beta = \begin{cases} 12 & \text{if } d = 1, \\ 125 & \text{if } d = 2, \\ 12 + \dfrac{8d^3 + 12d^2 + 6d}{\left\lceil \frac{1}{2} \left\lfloor \frac{d-1}{2} \right\rfloor \right\rceil} & \text{if } d \geq 3. \end{cases}$$

Proof: It is easy to see that when $d = 1$, the result is valid by Lem. 2. For $d = 2$, if we place a ball centered at z with radius 0.5 for each vertex z from I, then all these balls are mutually disjoint. Next, based on the knowledge that node u's any 2-hop neighbor (as a ball with radius 0.5) should located within the ball centered at u with radius 2.5, we can calculate the upper bound of β for $d = 2$ as

$$\beta \leq \frac{\frac{4}{3}\pi \cdot 2.5^3}{\frac{4}{3}\pi \cdot 0.5^3} = 125.$$

Now suppose $d \geq 3$. For any $u \in V$, let

$$A = N^{\lceil d/2 \rceil}(u) \cap I = \{a_1, a_2, \ldots, a_t\}$$

denote the set of u's d-hop independent neighbor within $\lfloor \frac{d}{2} \rfloor$ hops, and we will show that $t \leq 12$.

Suppose $t > 12$, then for any two vertices w, $v \in A$, denote the shortest path from w to u and from v to u as P_w and P_v, and denote w_0, v_0 as the last vertices on P_w and P_v. Because $N^1(u)$ contains at most 12 independent vertices, if $t > 12$, we could always choose two vertices w and v such that w_0 and v_0 are within each other's transmission range (could be the same vertex). Then we could construct a path from w to v by travel in this order: w, w_0, v_0, v, and the length of this path is at most $2\lceil d/2 \rceil - 1 \leq d$, contradicting with the fact that w, v are d-hop independent. Thus

$$\left| N^{\lceil d/2 \rceil}(u) \cap I \right| = t \leq 12. \tag{1}$$

Then consider the rest part of u's d-hop neighbors. Let

$$B = N^d(u) \setminus N^{\lceil d/2 \rceil}(u) \cap I = \{b_1, b_2, \ldots, b_m\}.$$

For each $1 \leq i \leq m$, let Q_i be a shortest path from b_i to u. Denote D_b as a ball centered at b with radius 0.5 and define a region C_i such that

$$C_i = \bigcup_{b \in N^{\lfloor (d-1)/2 \rfloor}(b_i) \cap V(Q_i)} D_b.$$

Here $V(Q_i)$ is the set of nodes on path Q_i. We then claim that C_i cannot intersect with C_j for any $i \neq j$. If it happens, then we can construct a path between b_i and b_j, which has length at most $2\lfloor (d-1)/2 \rfloor + 1 \leq d$, contradicting the fact that b_i, b_j are d-hop independent. Thus we have that C_i does not intersect with each other.

Suppose $Q_i = w_1 w_2 w_3 \ldots$. Since Q_i is a shortest path between b_i and u, then $D_{w_1}, D_{w_3}, \ldots$ are disjoint with each other, and the volume of C_i is at least $\left\lceil \frac{1}{2} \left\lfloor \frac{d-1}{2} \right\rfloor \right\rceil \frac{4}{3}\pi \frac{1}{2^3}$.

Next, notice that $b_i \notin N^{\lceil d/2 \rceil}$, so the distance between u and b_i is greater than $\lceil \frac{d}{2} \rceil$, i.e. $dis(u, b_i) > \lceil \frac{d}{2} \rceil$. For $b \in N^{\lfloor (d-1)/2 \rfloor}(b_i) \cap V(Q_i)$, according to the triangle inequality, we have

$$\begin{aligned} dis(u,b) &\geq dis(u,b_i) - dis(b,b_i) \\ &> \lceil d/2 \rceil - \lfloor (d-1)/2 \rfloor = 1, \end{aligned}$$

which means C_i does not intersect with the ball at center u with radius 0.5. Since every C_i locates within the ball at center u with radius $d + 0.5$, we have

$$m \leq \frac{\frac{4}{3}\pi(d+\frac{1}{2})^3 - \frac{4}{3}\pi\frac{1}{2^3}}{\left\lceil \frac{1}{2} \left\lfloor \frac{d-1}{2} \right\rfloor \right\rceil \frac{4}{3}\pi \frac{1}{2^3}} = \frac{8d^3 + 12d^2 + 6d}{\left\lceil \frac{1}{2} \left\lfloor \frac{d-1}{2} \right\rfloor \right\rceil} \tag{2}$$

From Equation (1) and Equation (2), we can get our final result. □

Lemma 4: Suppose D is a d-DS of G and I is a d-IS of G. Then $|I \setminus D| \leq \beta |D \setminus I|$.

Proof: Denote $X = I \setminus D$ and $Y = D \setminus I$. Construct a bipartite graph $H = (X, Y, E)$. Here $(x, y) \in E$ if and only if $x \in X$, $y \in Y$, and $dis(x, y) \leq d$. It is easy to see

$$\sum_{x \in X} deg_H(x) = \sum_{y \in Y} deg_H(y), \tag{3}$$

where $deg_H(x)$ represents the degree of x in the bipartite graph H. Since any vertex in X is d-hop dominated by some vertices, so for any $x \in X$,

$$deg_H(x) \geq 1. \tag{4}$$

Next, By Lem. 3, for any $y \in Y$,

$$deg_H(y) \leq \beta. \tag{5}$$

Then our lemma follows from Equation (3), (4), and (5). □

Lemma 5: Suppose I is a d-MIS for G, then we need at most $d(|I| - 1)$ nodes to connect I in Alg. 3.

Proof: In Alg. 3, for every black node except the root, there exists a black node at d hops away, just follow the $blackpath$. Thus, total nodes needed to connect I is at most $d(|I| - 1)$. □

Next we prove our main result.

Theorem 4: Our d-CDS algorithm has an approximation ratio of $(d + 1)\beta$.

Proof: Denote the set of black nodes after Alg. 3 as I, and denote the set of all black nodes after Alg. 4 as C. Let C^* be the optimal solution of d-hop CDS in G. By Lem. 4,

$$\begin{aligned} |I \setminus C^*| &\leq \beta |C^* \setminus I| \\ |I| - |I \cap C^*| &\leq \beta |C^*| - \beta |C^* \cap I| \\ |I| &\leq \beta |C^*| - (\beta - 1)|C^* \cap I| \end{aligned} \tag{6}$$

TABLE II

SIMULATION PARAMETERS

System Parameter	Description
number of nodes in WSN	100-500
number of samples in each round	50
step length	2
number of hops	1-3

Also by Lem. 5, we have

$$\begin{aligned} |C| &\leq d(|I| - 1) + |I| \\ &\leq (d+1)|I| - d \\ &\leq (d+1)\beta |C^*| \end{aligned}$$

Then the theorem holds. □

VII. SIMULATION AND EVALUATION

A. Experiment Settings

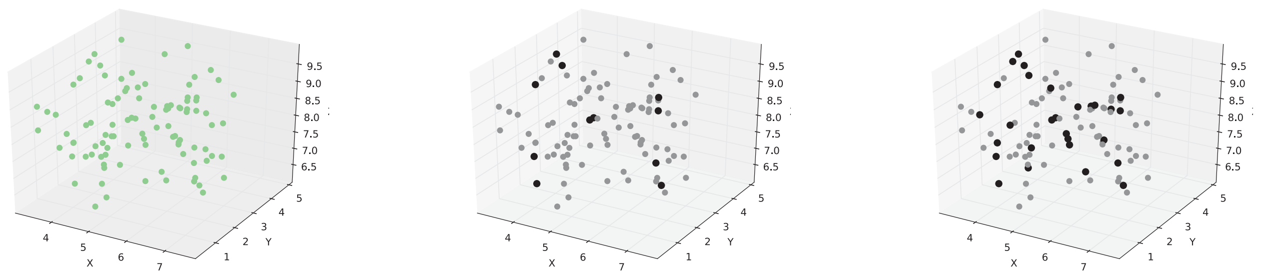
To evaluate the performance of our algorithm, we randomly generate different sets of nodes (with different node numbers and coordinates) on a space of size $100 \times 100 \times 100$ units. For each fixed hop setting ($d = 1$, $d = 2$, $d = 3$), we repeatedly run the algorithm for 50 different cases to achieve an average performance. We focus on the following impacts to evaluate the performance of our algorithm under different network sizes and different hop settings.

- The size of chosen CDS versus the total number of nodes in the network, which depicts the performance of the algorithm directly. If this percentage is smaller, then we can use a smaller size cluster heads to control the communication of the whole network, which is good for management and maintenance. It also has lower energy costs, bringing benefits to the service providers.
- The d-MIS size versus the d-CDS size. This parameter reflects the construction of two-step d-CDS algorithm, and exposes the cost in each step.
- The hop count of chosen CDS, reflecting the concentration level of CDS. If d is smaller, then more nodes need to be selected into d-MIS, while if d is larger, then more nodes are needed to connect two cluster heads.

Parameters used in our simulation are tabulated in Tab. II.

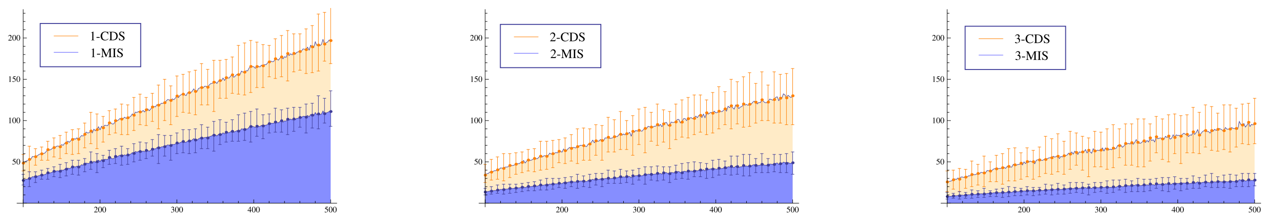
B. Experiment Results

In this part, we will provide some numerical results on our distributed algorithm. First, Fig. 5(a)-(c) show the detailed procedures of how our algorithm successfully obtains a d-CDS. In this exhibition, we set the biggest monitoring ability of a dominator as 3 hops and the number of total nodes as 100. We try to construct a 3-CDS with our algorithms. Fig. 5(a) exhibits the initial state of a WSN. In this state, every node is colored as green. After running Alg. 3, we get Fig. 5(b). In this figure, the black nodes form a 3-MIS with 11 nodes among 100 candidates. These nodes also form a dominating set which can dominate all other nodes in graph within 3 hops. Then we connect nodes in 3-MIS with some extra nodes with Alg. 4. We color that extra nodes black and form a 3-CDS shown in Fig. 5(c). Finally, 28 nodes are selected as 3-CDS for the given graph, which is only around 30% of the network.



(a). Original Network (b). A 3-DS with 11 nodes (c). A 3-CDS with 28 nodes

Fig. 5. An example to illustrate the procedure of our algorithm.



(a). MIS/CDS size w.r.t. n (b). 2-MIS/CDS size w.r.t. n (c). 3-MIS/CDS size w.r.t. n

Fig. 6. d-MIS and d-CDS size w.r.t. network size.

Next, let us compare the d-MIS size versus d-CDS size. This parameter reflects the construction cost for our algorithm. Fig. 6(a)-(c) show the error bar graph for d-MIS and d-CDS size w.r.t. network size where d is setting as 1, 2, and 3, respectively. In each case we take the best, the average, and the worst case results among 50 distinct experiments. From this figure we can see that as d increases, the size of d-MIS reduces significantly while the size of connectors is relatively increased. This phenomenon is easy to explain from the nature of our algorithm: (1) as d increases, less number of clusters are needed to dominate the whole network; (2) whereas more number of connectors are needed to connect pairwise cluster heads because the distance between them are increased according to d.

VIII. Conclusion

In this paper we propose a distributed algorithm for the clustering problem in high dimensional homogeneous WSNs, which has an approximation ratio of $(d+1)\beta$, where d is the number of hops in each cluster and β is a calculated parameter w.r.t. d. Our algorithm has 3 subroutines.

We first construct a spanning tree with some initialization of each node in the network. Then based on the spanning tree, we use coloring process to select a d-hop minimum independent set (d-MIS) for a graph G, which is also a d-hop dominating set (d-DS) for G in the second subroutine. Afterwards, we can insert some nodes to connect this d-MIS (d-DS) as a tree, making up a d-hop connected dominating set (d-CDS) for the network in the third subroutine. We propose an example to illustrate our idea, give a detailed analysis to this algorithm, and prove its approximation ratio theoretically. Finally, numerical experiments validate the efficiency of our design. To the best of our knowledge, we are the first work to design a distributed approximation algorithm for d-hop connected clustering problem in high dimensional space.

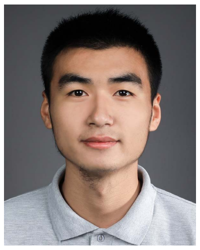
Acknowledgment

The authors would like to thank Xudong Zhu, Jun Li, and Yuanfang Xia for their contributions on the early versions of this paper.

References

[1] C.-F. Huang and Y.-C. Tseng, “The coverage problem in a wireless sensor network,” *Mobile Netw. Appl.*, vol. 10, no. 4, pp. 519–528, 2005.
[2] J. Blum, M. Ding, A. Thaeler, and X. Cheng, “Connected dominating set in sensor networks and MANETs,” in *Handbook of Combinatorial Optimization*. Berlin, Germany: Springer, 2005, pp. 329–369.
[3] N. Vlajic and D. Xia, “Wireless sensor networks: To cluster or not to cluster?” in *Proc. IEEE Int. Symp. World Wireless, Mobile Multimedia Netw. (WoWMoM)*, Jun. 2006, pp. 258–268.
[4] X. Zhu, L. Shen, and T. S. P. Yum, “Hausdorff clustering and minimum energy routing for wireless sensor networks,” *IEEE Trans. Veh. Technol.*, vol. 58, no. 2, pp. 990–997, Feb. 2009.
[5] C. Luo, Y. Zhu, X. Zhang, and Z. Zhou, “A clustering algorithm based on cell combination for wireless sensor networks,” in *Proc. IEEE Int. Workshop Educ. Technol. Comput. Sci. (ETCS)*, vol. 2, Mar. 2010, pp. 74–77.
[6] M. Alaei and J. M. Barcelo-Ordinas, “Node clustering based on overlapping FoVs for wireless multimedia sensor networks,” in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2010, pp. 1–6.
[7] V. Kumar, S. Jain, and S. Tiwari, “Energy efficient clustering algorithms in wireless sensor networks: A survey,” *Int. J. Comput. Sci. Issues*, vol. 8, no. 5, pp. 259–268, 2011.
[8] D. Wei, Y. Jin, S. Vural, K. Moessner, and R. Tafazolli, “An energy-efficient clustering solution for wireless sensor networks,” *IEEE Trans. Wireless Commun.*, vol. 10, no. 11, pp. 3973–3983, Nov. 2011.
[9] X. Gao, W. Wu, X. Zhang, and X. Li, “A constant-factor approximation for d-hop connected dominating sets in unit disk graph,” *Int. J. Sensor Netw.*, vol. 12, no. 3, pp. 125–136, 2012.
[10] M. Tripathi, R. B. Battula, M. S. Gaur, and V. Laxmi, “Energy efficient clustered routing for wireless sensor network,” in *Proc. IEEE Int. Conf. Mobile Ad-hoc Sensor Netw. (MSN)*, Dec. 2013, pp. 330–335.
[11] C. Lin, Y. Zhou, H. Dai, J. Deng, and G. Wu, “MPF: Prolonging network lifetime of wireless rechargeable sensor networks by mixing partial charge and full charge,” in *Proc. IEEE Int. Conf. Sens. Commun. Netw. (SECON)*, Jun. 2018, pp. 1–9.
[12] Z. Wang, X. Qin, and B. Liu, “An energy-efficient clustering routing algorithm for WSN-assisted IoT,” in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2018, pp. 1–6.
[13] Z. Zhao, X. Gao, W. Wu, and D. Z. Du, “A PTAS for minimum connected dominating set in 3-dimensional wireless sensor networks,” *J. Global Optim.*, vol. 45, no. 3, pp. 451–458, 2009.

[14] H. Breu and D. G. Kirkpatrick, "Unit disk graph recognition is NP-hard," *Comput. Geometry Theory Appl.*, vol. 9, nos. 1–2, pp. 9–31, 1998.
[15] D. Kim *et al.*, "A better approximation algorithm for computing connected dominating sets in unit ball graphs," *IEEE Trans. Mobile Comput.*, vol. 9, no. 8, pp. 1108–1118, Aug. 2010.
[16] T. Vuong and D. Huynh, "Adapting d-hop dominating sets to topology changes in ad hoc networks," in *Proc. IEEE Int. Conf. Comput. Commun. Netw. (ICCCN)*, Oct. 2000, pp. 348–353.
[17] S. Fedor and M. Collier, "On the problem of energy efficiency of multi-hop vs one-hop routing in wireless sensor networks," in *Proc. IEEE Int. Conf. Adv. Inf. Netw. Appl. Workshops (AINAW)*, May 2007, pp. 380–385.
[18] B. N. Clark, C. J. Colbourn, and D. S. Johnson, "Unit disk graphs," *Discrete Math.*, vol. 86, pp. 165–177, Dec. 1990.
[19] P.-J. Wan, K. M. Alzoubi, and O. Frieder, "Distributed construction of connected dominating set in wireless ad hoc networks," in *Proc. 21st Annu. Joint Conf. IEEE Comput. Commun. Soc.*, vol. 3, Jun. 2002, pp. 1597–1604.
[20] S. Bai, X. Che, X. Bai, and X. Wei, "Maximal independent sets in heterogeneous wireless ad hoc networks," *IEEE Trans. Mobile Comput.*, vol. 15, no. 8, pp. 2023–2033, Aug. 2016.
[21] J. Li and X. Gao, "Performance analysis for approximating MCDS in wireless ad-hoc network," *Inf. Jpn.*, vol. 16, no. 2, pp. 1111–1117, 2013.
[22] X. Cheng, X. Huang, D. Li, W. Wu, and D.-Z. Du, "A polynomial-time approximation scheme for the minimum-connected dominating set in ad hoc wireless networks," *Networks*, vol. 42, no. 4, pp. 202–208, 2003.
[23] B. Das and V. Bharghavan, "Routing in ad-hoc networks using minimum connected dominating sets," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Montreal, QC, Canada, vol. 1, Jun. 1997, pp. 376–380.
[24] F. Dai and J. Wu, "An extended localized algorithm for connected dominating set formation in ad hoc wireless networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 15, no. 10, pp. 908–920, Oct. 2004.
[25] M. Gerla and J. T.-C. Tsai, "Multicluster, mobile, multimedia radio network," *Wireless Netw.*, vol. 1, no. 3, pp. 255–265, 1995.
[26] D. Cokuslu and K. Erciyes, "A hierarchical connected dominating set based clustering algorithm for mobile ad hoc networks," in *Proc. IEEE Int. Symp. Modeling Anal. Simulation Comput. Telecommun. Syst. (MASCOTS)*, Oct. 2007, pp. 60–66.
[27] D. Li, L. Liu, and H. Yang, "Minimum connected r-hop k-dominating set in wireless networks," *Discrete Math. Algorithms Appl.*, vol. 1, no. 1, pp. 45–58, 2009.
[28] Z. Zhang, Q. Liu, and D. Li, "Two algorithms for connected r-hop k-dominating set," *Discrete Math. Algorithms Appl.*, vol. 1, no. 4, pp. 485–498, 2009.
[29] X. Li and Z. Zhang, "Two algorithms for minimum 2-connected r-hop dominating set," *Inf. Process. Lett.*, vol. 110, no. 22, pp. 986–991, 2010.
[30] R. S. Coelho, P. F. S. Moura, and Y. Wakabayashi, "The k-hop connected dominating set problem: Approximation and hardness," *J. Combinat. Optim.*, vol. 34, no. 4, pp. 1060–1083, 2017.
[31] X. Gao *et al.*, "A novel approximation for multi-hop connected clustering problem in wireless networks," *IEEE/ACM Trans. Netw.*, vol. 25, no. 4, pp. 2223–2234, Aug. 2017.
[32] K. Skiadopoulos, K. Giannakis, K. Oikonomou, I. Stavrakakis, and S. Aïssa, "Distributed construction of d-hop connected dominating sets for wireless sensor networks," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2018, pp. 1–7.
[33] T. Nguyen and D. Huynh, "Connected d-hop dominating sets in mobile ad hoc networks," in *Proc. IEEE Int. Symp. Modeling Optim. Mobile, Ad Hoc Wireless Netw.*, Mar. 2006, pp. 1–8.
[34] M. Azizian, S. Cherkaoui, and A. S. Hafid, "A distributed D-hop cluster formation for VANET," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2016, pp. 1–6.
[35] W. Wang *et al.*, "A new constant factor approximation to construct highly fault-tolerant connected dominating set in unit disk graph," *IEEE/ACM Trans. Netw.*, vol. 25, no. 1, pp. 18–28, Feb. 2017.
[36] S. Butenko, S. Kahruman-Anderoglu, and O. Ursulenko, "On connected domination in unit ball graphs," *Optim. Lett.*, vol. 5, no. 2, pp. 195–205, 2011.
[37] 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 *Proc. Int. Conf. Ubiquitous Inf. Manage. Commun. (IMCOM)*, 2014, pp. 1–8.
[38] I. Cidon and O. Mokryn, "Propagation and leader election in a multihop broadcast environment," in *Proc. Distrib. Comput.*, 1998, pp. 104–118.
[39] H. Huang, A. W. Richa, and M. Segal, "Approximation algorithms for the mobile piercing set problem with applications to clustering in ad-hoc networks," *ACM/Springer Mobile Netw. Appl.*, vol. 9, no. 2, pp. 151–161, 2004.



Ke Li is currently pursuing the bachelor's degree with the School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, China. His research interests include a wide range of networks and systems, including network optimization, wireless networks, crowdsourcing, and D2D communication networks. He has published a paper as the primary author studying the clustering problem in wireless networks in the 48th International Conference on Parallel Processing (ICPP 2019).

Xiaofeng Gao received the B.S. degree in information and computational science from Nankai University, China, in 2004, the M.S. degree in operations research and control theory from Tsinghua University, China, in 2006, and the Ph.D. degree in computer science from The University of Texas at Dallas, USA, in 2010. She is currently a Professor with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, China. Her research interests include data engineering, indexing, and combinatorial optimizations. She has published more than 160 peer-reviewed papers in the related area, including well-archived international journals, such as the IEEE TC, TKDE, TMC, TPDS, JSAC, and also in well-known conference proceedings, such as SIGKDD, INFOCOM, and ICDCS. She has served on the Editorial Board of *Discrete Mathematics, Algorithms and Applications*, and as the PCs and peer reviewers for a number of international conferences and journals.

Fan Wu received the B.S. degree in computer science from Nanjing University in 2004, and the Ph.D. degree in computer science and engineering from the State University of New York at Buffalo in 2009. He has visited the University of Illinois at Urbana–Champaign (UIUC) as a Post-Doctoral Research Associate. He is currently a Professor with the Department of Computer Science and Engineering, Shanghai Jiao Tong University. His research interests include wireless networking and mobile computing, algorithmic game theory and its applications, and privacy preservation. He has published more than 170 peer-reviewed papers in technical journals and conference proceedings. He was a recipient of the First Class Prize for the Natural Science Award of the China Ministry of Education, the NSFC Excellent Young Scholars Program, the ACM China Rising Star Award, the CCF-Tencent "Rhinoceros bird" Outstanding Award, the CCF-Intel Young Faculty Researcher Program Award, the Pujiang Scholar, and the Tang Scholar. He has served as the Chair of CCF YOCSEF Shanghai, on the Editorial Board of *Elsevier Computer Communications*, and as the member of the technical program committees of more than 60 academic conferences.

Guihai Chen (M'05) received the B.S. degree from Nanjing University in 1984, the M.E. degree from Southeast University in 1987, and the Ph.D. degree from The University of Hong Kong in 1997. He is a Distinguished Professor of Shanghai Jiao Tong University, China. He had been invited as a Visiting Professor by many universities, including the Kyushu Institute of Technology, Japan, in 1998, the University of Queensland, Australia, in 2000, and Wayne State University, USA, from 2001 to 2003. He has a wide range of research interests with a focus on sensor networks, peer-to-peer computing, high-performance computer architecture, and combinatorics. He has published more than 250 peer-reviewed papers and more than 170 of them are in well-archived international journals, such as the IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, the *Journal of Parallel and Distributed Computing*, *Wireless Networks*, *The Computer Journal*, the *International Journal of Foundations of Computer Science*, and *Performance Evaluation*, and also in well-known conference proceedings, such as HPCA, MOBIHOC, INFOCOM, ICNP, ICDCS, CoNEXT, and AAAI. He is a CCF Fellow. He has won several best paper awards, including the ICNP 2015 Best Paper Award. His papers have been cited for more than 10 000 times according to Google Scholar.