Minimum Data-Latency-Bound k-Sink Placement Problem in Wireless Sensor Networks

Donghyun Kim, *Member, IEEE, ACM*, Wei Wang, Nassim Sohaee, Changcun Ma, Weili Wu, *Member, IEEE*, Wonjun Lee, *Senior Member, IEEE*, and Ding-Zhu Du, *Member, IEEE*

***Abstract*—In this paper, we propose a new multiple-sink positioning problem in wireless sensor networks to best support real-time applications. We formally define this problem as the k-Sink Placement Problem (k-SPP) and prove that it is APX-complete. We show that an existing approximation algorithm for the well-known k-center problem is a constant factor approximation of k-SPP. Furthermore, we introduce a new greedy algorithm for k-SPP and prove its approximation ratio is very near to the best achievable, 2. Via simulations, we show our algorithm outperforms its competitor on average.**

***Index Terms*—Graph theory, greedy approximation algorithms, network center placement problem, wireless sensor networks (WSNs).**

I. Introduction

RECENTLY, wireless sensor networks (WSNs) have been investigated for a number of real-world applications. The primary goal of WSNs is observing specified events and, once an event is detected, notifying the occurrence of the event to a designated collector, which is generally called a sink or a gateway, for further processing. Usually, a sink is assumed to have an almost unlimited energy source and computation power, and sometimes mobility. Generally, a sensor node communicates with its neighbors using radio frequency (RF) signals, which consume more energy superlinearly proportional to their travel distance. Since each sensor node has a limited power source such as a battery, energy efficiency is one of the most crucial issues of WSNs. Therefore, multihop communication is preferred over long-range direct communication in WSNs. In multihop WSNs, the amount of energy spent to deliver a message and the latency of the message are proportional to the number of hops in the path over which the message travels.

Most earlier papers in this field studied a WSN with a single sink. They assumed the sink is fixed and tried to improve the performance (i.e., lifetime, latency, etc.) of the WSN using various approaches. However, in many applications of WSNs such as battlefield monitoring or wildfire detection, the sink has mobility and can be repositioned. This fact motivated many people to study the optimal sink repositioning problem to maximize the lifetime of WSNs [1], [2]. More recently, people realized that by introducing multiple sinks, the performance of WSNs can be greatly boosted and the WSNs can be more scalable. The majority of them studied the multiple-sink placement problem to maximize the lifetime of WSNs [3]–[10]. Some others used this idea to minimize the average data latency of WSNs [11], [12].

In this paper, we consider a new multiple-sink placement problem for applications where timeliness is a critical issue. We noticed that while the "average data latency" is a good performance measurement for some applications of WSNs, there are many other applications of WSNs, such as battlefield surveillance and intrusion detection, in which it is extremely important to know the "worst-case data latency" so that the applications can schedule and execute their jobs without violating their real-time constraints (i.e., job deadline). Clearly, if we can make the worst-case data latency smaller, we can improve the performance of a real-time system. It is known that the latency of a message is proportional to the number of hops that the message travels in multihop wireless networks [12]. Therefore, we can conclude that the latency bound can be reduced by carefully placing the available sinks. Based on the observations and facts, given k available sinks due to the limitation of budget and resources, the problem of how to minimize the maximum data latency from a node to its nearest sink can be abstracted as the *k-Sink Placement Problem* (*k-SPP*), whose goal is to minimize the maximum hop distance between a node and its nearest sink. We provide the formal definition of k-SPP and its APX-completeness proof in Section V.

The following is the summary of the contributions of this paper.

1) We formally define k-SPP and prove k-SPP is *APX-complete* by showing there is no $(2-\epsilon)$-approximation algorithm unless $P = NP$, where ϵ is a small positive con-

Manuscript received October 30, 2009; revised August 28, 2010 and January 02, 2011; accepted January 15, 2011; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor K. Kar. Date of publication February 24, 2011; date of current version October 14, 2011. This work was supported in part by the National Natural Science Foundation of China under Grants 11071191, 61073174, and 61033001; the National Scientific Foundation under Grants CNS-0524429, CCF-0627233, and CCF-0514796; the National Basic Research Program of China under Grants 2007CB807900 and 2007CB807901; the Ministry of Education, Science and Technology (MEST), Korea, under WCU (R33-2008-000-10044-0); the National Research Foundation (NRF), Korea, under Grant KRF-2008-314-D00354; and the Ministry of Knowledge Economy (MKE), Korea, under ITRC NIPA-2010-(C1090-1021-0008).

D. Kim is with the Department of Mathematics and Computer Science, North Carolina Central University, Durham, NC 27707 USA (e-mail: donghyun.kim@nccu.edu).

W. Wang is with the Department of Mathematics, Xi'an Jiaotong University, Xi'an 710049, China (e-mail: wang_weiw@163.com).

N. Sohaee is with the Department of Clinical Sciences, University of Texas Southwestern Medical Center, Dallas, TX 75390-9066 USA (e-mail: nassim.sohaee@utsouthwestern.edu).

C. Ma is with the Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing 100084, China (e-mail: mcc03@mails.tsinghua.edu.cn).

W. Wu and D.-Z. Du are with the Department of Computer Science, University of Texas at Dallas, Richardson, TX 75083 USA (e-mail: weiliwu@utdallas.edu; dzdu@utdallas.edu).

W. Lee is with the Department of Computer Science and Engineering, Korea University, Seoul 136-713, Korea (e-mail: wlee@korea.ac.kr).

Digital Object Identifier 10.1109/TNET.2011.2109394

stant. This is done by showing in a subclass of unit disk graph (UDG; see Section III for its definition) that k-SPP is equal to an existing APX-complete problem, which can be approximated no better than 2 unless $P = NP$.

2) We show that an existing approximation algorithm for the k-center problem [17], GREEDY-k-CENTER, is also an approximation algorithm for k-SPP. In our analysis, we show given a k-SPP instance $\text{Cost}(S_1) \leq 18\ \text{Cost}\ (\text{OPT}_{k\text{-SPP}}) + 8$, where S_1 is an output of the approximation algorithm for the k-center problem, which is also a feasible solution of k-SPP, Cost (S_1) is the cost of a feasible solution (i.e., S_1) of the k-SPP instance (i.e., the maximum hop distance from a node to its nearest sink over the shortest hop path between them), and $\text{OPT}_{k\text{-SPP}}$ is an optimal solution of the k-SPP instance.
3) To get a better quality solution, we introduce a new simple greedy algorithm, which uses a larger, but still polynomial-size, feasible solution space $\mathcal{F}$. That is, we notice that given an input UDG $G = (V, E)$, an output of GREEDY-k-CENTER is a subset of V. In contrast, our new algorithm uses a simple graph theoretical technique to compute a new solution space $\mathcal{F}$ such that an optimal solution of k-SPP is a subset of $\mathcal{F}$. Then, the algorithm, namely GREEDY-k-SPP, applies a simple greedy strategy to get a feasible solution of k-SPP. We show given a k-SPP instance Cost $(S_2) \leq 2$ Cost $(\text{OPT}_{k\text{-SPP}}) + 1$, where S_2 is an output of GREEDY-k-SPP. This means that even in the worst case, the cost of an output of this algorithm is within three times from the cost of an optimal solution. Furthermore, as the cost of $\text{OPT}_{k\text{-SPP}}$ grows, the performance ratio of this algorithm is nearly 2, which is *the best possible performance ratio* for k-SPP.
4) Through simulations, we show that GREEDY-k-SPP outperforms its competitor on average.

The rest of this paper is organized as follows. In Section II, we introduce the related work. Section III presents the notations, definitions, and assumptions of our work. In Section IV, we present preliminaries that include an existing approximation algorithm for the k-center problem, namely GREEDY-k-CENTER, and we prove that, in fact, GREEDY-k-CENTER is a constant factor approximation for k-SPP. In Section V, we formally define k-SPP and show it is APX-complete. In Section VI, we introduce a new greedy approximation algorithm for k-SPP and its corresponding theoretical analysis. The simulation results and analysis are given in Section VII. Finally, Section VIII makes a conclusion and suggests several future research directions.

II. Related Work

In a WSN with a single sink, the hop distance between each node and the sink increases as the network size grows, which results in longer data latency and shorter network lifetime. To make sensor networks more scalable, more than one sink is employed [3]–[12]. There are several research issues regarding this approach. In this paper, we focus on how to find the best positions of k sinks given an optimization goal.

Oyman and Ersoy showed that by having multiple sinks in a large-scale WSN, the manageability of the network increases and the energy dissipation at each node decreases [5]. They also found that by properly placing the sinks, their benefit can be maximized, which coincides with the conclusion of [4]. Naturally, this research topic attracted lots of researchers. Especially, many efforts are made to extend the lifetime of WSNs by deploying multiple sinks carefully [3]–[10]. Given that the traffic generated by each node, current energy level of each node, the maximum number of available sinks, and the set of candidate positions to place the sinks are known, the majority of the papers tried to extend the lifespan of WSNs by exploiting linear programming-based techniques [3], [7], [8], [10]. The authors in [6] used the same assumptions and incorporated an interference model and fault tolerance consideration into their linear programming formulation separately. Interestingly, in [9], an electrostatic model was proposed to find the best positions for k sinks. They observed that nodes closer to a sink are exhausted faster for relaying messages and suggested to move the sinks near to those nodes with sufficient energy. In detail, they assign positive charge to all sinks and those nodes whose energy levels are below average. On the other hand, negative charge is given to the rest of nodes. In their algorithm, the sinks are distributed through the electrostatic field and relocated somewhere close to those nodes with higher energy level. In [11] and [12], the authors tried to find the locations of k sinks such that the average euclidean distance between nodes and their nearest sink is minimized. Solutions for this formulation can be applied to the WSNs, in which each node communicates with its nearest sink (in terms of euclidean distance) directly regardless of how far they are apart. Note that those problems are variations of the facility location problem (FLP).

The FLP is one of the very popular research topics in the operations research and management science societies. It has a number of variations, and most of them are NP-complete. Therefore, considerable attention has been made to design approximation algorithms (see [13] for a survey). Generally, the following parameters are given as an input of FLP: a set $\mathcal{F}$ to locate facilities, a set $\mathcal{C}$ of cities (or clients or nodes), a cost function f_i for opening facility $i \in \mathcal{F}$, and a connection cost $c_{i,j}$ for connecting client j to facility i. Then, the goal of FLP is to find a subset $S \in \mathcal{F}$ such that the *total cost* (or equivalently average cost) to open a facility in each location in S and connect each city to a facility is minimized. The metric (or euclidean) k-center problem [14] has an objective function similar to that of k-SPP's. That is, the object of the k-center problem is to *minimize the maximum hop distance* between a node and its nearest sink. However, unlike k-SPP, the k-center problem requires choosing the centers from existing nodes in the graph. Since in k-SPP, a sink can be located at any place, the k-center problem can be considered as a special case of k-SPP, and thus k-SPP is harder. In the rest of this paper, we denote the k-center problem by k-CENTER. Due to its gravity, we revisit k-CENTER and introduce an existing 2-approximation algorithm for it in Section IV-A.

III. Notations, Definitions, and Assumptions

Now, we introduce some notations and definitions that are frequently used in the rest of this paper. $\mathrm{Eucdist}(u,v)$ is the *euclidean distance* between two nodes u and v. $\mathrm{Hopdist}(u,v)$ is the *hop distance* between two nodes u and v over the shortest path between them. A graph is a *unit disk graph* (UDG) if for each pair of nodes u and v, there is a bidirectional edge between them if and only if the euclidean distance between u and v is no more than 1. In this paper, we use $G=(V,E)=(V(G),E(G))$ to represent a UDG. $\mathrm{Cost}(S)$ is the *cost of a feasible solution S* of k-SPP. In Section V, we formally introduce this function. $N_d(v)=\{u|\forall u\in V \text{ such that } u\neq v, \text{ and } \mathrm{Hopdist}(u,v)\leq d\}$ is the set of *d-hop neighbors* of a node v. For simplicity, $N(v)=N_1(v)$ (i.e., the set of the direct neighbors of v). Similarly, for a node set S, $N_d(S)=\cup_{\forall v\in S}N_d(v)$. For a node v, $N_d[v]=N_d(v)\cup\{v\}$. We will also use $N[v]$ and $N_1[v]$ interchangeably. A subset $S\subseteq V$ is a *dominating set* (DS) of V only if, $\forall v\in V$, either $v\in S$ or $v\in N(S)$. A subset $S\subseteq V$ is an *independent set* (IS) of V only if $\forall v,u\in S$, $(v,u)\notin E$. A subset $S\subseteq V$ is a *maximal independent set* (MIS) only if S is an IS and $\forall v\in V\setminus S$, $\{v\}\cup S$ is not an IS anymore (i.e., v is neighboring to some node in S). $\mathrm{MIS}(S)$ is an MIS of S. Note that an MIS of G is also a DS of G. d-DS is a subset $S\subseteq V$ such that $\forall v\in V$, either $v\in S$ or $v\in N_d(S)$.

In this paper, we have the following assumptions. 1) WSNs are homogeneous networks (every node has the same physical characteristics such as maximum communication range) on a 2-D plane. 2) Both nodes and sinks have equal transmission range. 3) The global coordinate of each node and each sink can be obtained using some existing methods (i.e., localization algorithm [15]) or a hardware [i.e., global positioning system (GPS)]. 4) Input graphs are connected. The reasons for these assumptions are given in Section VII.

IV. Preliminaries

A. (Metric/Euclidean) k-Center Problem and Its 2-Approximation

Given a graph $G=(V,E)$, k-CENTER is to find a size k subset $S\subseteq V$ such that the maximum hop distance between a node in $V\setminus S$ and its nearest center in S is as small as possible [14]. In [16], it is proved that k-CENTER is APX-complete for general graphs such that there exists no approximation algorithm whose performance ratio is better than 2. The proof is done by reducing k-CENTER from the minimum DS problem. That is, given a graph $G=(V,E)$, determining if we have a DS of G of size no more than k is equivalent to the existence of k centers of $G'=(V,E')$ with cost 1, where G' is a complete graph constructed from G such that for each $u,v\in V$, $(u,v)\in E'$ and its weight is 1 only if $(u,v)\in E$. Otherwise, $(u,v)\in E'$ and its weight is 2.

There are many existing approximation algorithms for k-CENTER. For example, we can use the simple greedy algorithm in [17] (Algorithm 1), which achieves the best possible approximation ratio 2 for k-CENTER. Originally, k-CENTER requires the cost to connect two nodes u and v to satisfy the triangular inequality. However, in the rest of this paper, we will assume the cost to be specifically $\mathrm{Hopdist}(u,v)$ since this is the metric that k-SPP is using and it also satisfies the triangular inequality due to the fact that each edge incurs a uniform cost. The idea behind Algorithm 1 is quite simple: Select a center one by one such that the hop distances between the newly selected center and existing centers can be maximized. In detail, we first arbitrarily select a node v_1 as the first center. Next, choose another node v_2 among the rest of nodes of V that is the farthest from v_1. Generally, if we have chosen $\{v_1,v_2,\ldots,v_s\}$, then the $(s+1)$th center v_{s+1} is chosen from $V\setminus\{v_1,v_2,\ldots,v_s\}$, such that the minimum hop distance between v_{s+1} and each $v_i, 1\leq i\leq s$ is maximized. Continue this process until all k centers are placed. Fig. 1 is an example illustrating how this algorithm works.

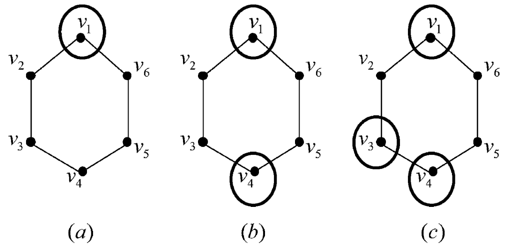


Fig. 1. This example illustrates how Algorithm 1 works. Suppose $k=3$. First, the algorithm picks a node. (a) Suppose v_1 is chosen. (b) Second, the algorithm selects the farthest node from v_1, which is v_4. At last, the algorithm picks any of $\{v_2,v_3,v_5,v_6\}$ since each of them is adjacent to one of the selected nodes. (c) Suppose v_3 is selected. At the end, this algorithm puts three sinks very near to v_1, v_3, and v_4.

Algorithm 1: GREEDY-k-CENTER $(G(V,E),k)$

```
1: Set $S \leftarrow \emptyset$ and $V \leftarrow V(G)$.
2: Randomly select a vertex $u \in V$ and set $S \leftarrow \{u\}$ and
   $V \leftarrow V \setminus \{u\}$.
3: while $|S| < k$ do
4:     Select $u \in V$ such that
           $u = \arg\max_{v_j \in V} \min_{v_i \in S} \mathrm{Hopdist}(v_j, v_i)$.
5:     $S \leftarrow S \cup \{u\}$ and $V \leftarrow V \setminus \{u\}$.
6: end while
```

Here, we provide a brief sketch of the proof to show that the performance ratio of GREEDY-k-CENTER is 2. Any interested reader can find more details from [18]. We would like to emphasize that as the algorithm chooses more centers, the maximum distance from a node to a center does not increase (remain the same or decrease).

Now, we prove this by contradiction. Suppose that after the algorithm selected k centers

$$\mathrm{Cost}\ (S) > 2\ \mathrm{Cost}\ (\mathrm{OPT}_{k\text{-CENTER}}) \tag{1}$$

is true, where $S=\{s_1,\ldots,s_k\}$ is an output of the algorithm and $\mathrm{OPT}_{k\text{-CENTER}}$ is an optimal solution. Let $s_{k+1}\in V\setminus S$ be the node that is farthest from S. Now, assign each element in S' to its nearest center in $\mathrm{OPT}_{k\text{-CENTER}}$. By the Pigeonhole Principle, there has to be at least one pair of nodes, $s_a,s_b\in S'$, that share the same nearest center in $c \in \mathrm{OPT}_{k\text{-CENTER}}$. By the assumption (1), the hop distance between any pair of nodes in $S' = S \cup \{s_{k+1}\}$ is greater than $2\mathrm{Cost}(\mathrm{OPT}_{k\text{-CENTER}})$ [i.e., Hopdist $(s_a, s_b) > 2$ Cost $(\mathrm{OPT}_{k\text{-CENTER}})$]. However, if Hopdist $(s_a, s_b) > 2$ Cost $(\mathrm{OPT}_{k\text{-CENTER}})$ is true, then by the triangular inequality, both s_a and s_b cannot be within $\mathrm{Cost}(\mathrm{OPT}_{k\text{-CENTER}})$ hops from c at the same time, which contradicts: 1) our initial assumption (1); and 2) the assertion that c is the nearest member of an optimal solution to s_a and s_b. As a result, $\mathrm{Cost}(S) \leq 2$ Cost $(\mathrm{OPT}_{k\text{-CENTER}})$ is true, and the performance ratio of GREEDY-k-CENTER is 2. Hochbaum and Shmoys also proposed another 2-approximation algorithm for k-CENTER using a power graph technique [19]. However, since this is not close to our work, we will not describe the details.

B. Performance Analysis of GREEDY-k-CENTER for k-SPP

In k-SPP, a sink can be placed anywhere on the euclidean plane. Therefore, by putting a sink on each node selected by Algorithm 1, we can have a feasible solution of k-SPP, which is not necessarily an optimal solution of k-SPP. In this section, we prove Algorithm 1 is a constant factor approximation for k-SPP. Let S be a set of nodes chosen by Algorithm 1. Then, from [18], we know

$$\mathrm{Cost}(S) \leq 2 \text{ Cost } (\mathrm{OPT}_{k\text{-CENTER}}) \quad (2)$$

where $\mathrm{OPT}_{k\text{-CENTER}}$ is an optimal solution of k-CENTER. To see the performance ratio of Algorithm 1 for k-SPP, we need to find the relationship between $\mathrm{Cost}(\mathrm{OPT}_{k\text{-CENTER}})$ and $\mathrm{Cost}(\mathrm{OPT}_{k\text{-SPP}})$, where $\mathrm{OPT}_{k\text{-SPP}}$ is an optimal solution of k-SPP. For example, if we can prove that $\mathrm{Cost}(\mathrm{OPT}_{k\text{-CENTER}}) \leq c \cdot \mathrm{Cost}(\mathrm{OPT}_{k\text{-SPP}})$, for some c, we can combine this with (2) and prove that $\mathrm{Cost}(S) \leq 2c \cdot \mathrm{Cost}(\mathrm{OPT}_{k\text{-SPP}})$. Now, we present two important lemmas and bound c to a constant based on the lemmas. As a result, we can prove that GREEDY-k-CENTER is a constant factor approximation for k-SPP.

Lemma 4.1: Suppose G is d-hop dominated by a subset $S = \{x_1, \ldots, x_k\}$ of $V(G)$. Then, G is also d-hop dominated by $Y = \bigcup_{\forall i} \mathrm{MIS}(N_1(x_i))$

Proof: Let v be a node in G. Now, suppose v is d-hop dominated by another node, say $x_i \in V(G)$. Then, there has to be a path $v \ldots w x_i$ of length at most d connecting v and x_i, where w is a neighbor of x_i. It follows w is connected with some node in $\mathrm{MIS}(N_1(x_i))$ or included in $\mathrm{MIS}(N_1(x_i))$. Hence, v is d-hop dominated by some node in Y. ■

Lemma 4.2: Let G be a connected graph with size $5k$. Then, there is a 4-DS of G whose size is no more than k, which implies that there is a 4-DS of G whose size is exactly k (by adding some redundant nodes).

Proof: We prove this lemma by induction on the size of 4-DS. The basis step of this lemma is very easy to prove since when $k = 1$, a connected graph with size 5 is easily 4-hop dominated by one node. As the induction hypothesis, consider a connected graph of size $5k$ and a 4-DS of the graph whose size is less than or equal to k. Now, we need to show that for a connected graph G with $5k+5$ nodes, there is a 4-DS whose size is no more than $k+1$. Suppose $T(G)$ is a spanning tree over G and P is the longest path in $T(G)$. Let $\{v_1, v_2, \ldots, v_p\}$ be a set of nodes in P, and sorted by the order of their appearance in P and v_1 and v_p are two leaves. Now, remove an edge between v_5 and v_6 and partition $T(G)$ into $T_1(G)$ and $T_2(G)$. Without loss of generality, we assume $|T_1(G)| \leq |T_2(G)|$. Then, $|T_1(G)| \geq 5$ and $T_1(G)$ can be 4-hop dominated by one node, v_5. On the other hand, $|T_2(G)| \leq 5k$, and by the induction hypothesis, $T_2(G)$ can be 4-hop dominated by k nodes. Therefore, $T(G)$ can be 4-hop dominated by $k+1$ nodes, and so does G. ■

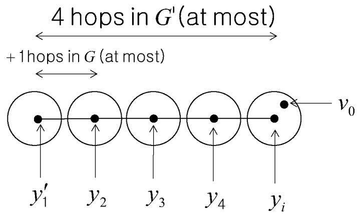


Fig. 2. Suppose we have a 4-DS for G' (big circles). Then, y_i which d^*-hop dominates v_0, in G is at most 4 hops far from some y_1' in G'. The hop distance between two neighboring nodes y_i and y_j in G' is at most $2d^*+1$ in G. Therefore, each node in G must have at least one node in Y' such that their distance is at most $9d^* + 4$ hops.

Theorem 4.3: Given a k-SPP instance, $\mathrm{Cost}(S) \leq 18$ Cost $(\mathrm{OPT}_{k\text{-SPP}}) + 8$ is true, where S is an output of Algorithm 1 over the problem instance.

Proof: It is easy to see $\mathrm{Cost}(\mathrm{OPT}_{k\text{-SPP}}) \leq \mathrm{Cost}(\mathrm{OPT}_{k\text{-CENTER}})$ since in k-SPP, we have more choices to put sinks. We first show $\mathrm{Cost}(\mathrm{OPT}_{k\text{-CENTER}}) \leq 9\mathrm{Cost}(\mathrm{OPT}_{k\text{-SPP}}) + 4$, and combine this with (2) to prove this theorem. Suppose we have $\mathrm{OPT}_{k\text{-SPP}} = \{x_1, x_2, \ldots, x_k\}$. Then, we can compute a subset $Y = \bigcup_{\forall i} \mathrm{MIS}(N_1(x_i)) = \{y_1, \ldots, y_s\} \subseteq V(G)$. Let $d^* = \mathrm{Cost}(\mathrm{OPT}_{k\text{-SPP}})$. Then, there is one fact we would like to emphasize.

- **Fact 1**: Since $\mathrm{OPT}_{k\text{-SPP}}$ is an optimal solution of k-SPP, G is d^*-hop dominated by $\mathrm{OPT}_{k\text{-SPP}}$, and by Lemma 4.1, G is also d^*-hop dominated by Y.

Now, we construct a new graph $G' = (Y, E')$ from G as follows: The vertex set of G' corresponds to the set Y, and there is an edge between two vertices y_i and y_j in G' if there exist two nodes u and v in G such that $u \in N_{d^*}[y_i], v \in N_{d^*}[y_j]$, and $(u, v) \in E(G)$. We would like to recall the following facts.

- **Fact 2**: y_i and y_j exist in both G and G'.
- **Fact 3**: If y_i and y_j are two neighboring nodes in G', in G
$$\text{Hopdist } (y_i, y_j) \leq 2d^* + 1. \quad (3)$$
- **Fact 4**: $|V(G')| = |Y| = s \leq 5k = 5|\mathrm{OPT}_{k\text{-SPP}}|$ because for any x_i, $|\mathrm{MIS}(N_1(x_i))| \leq 5$ in any UDG [20].

By Lemma 4.2 and Fact 4, there exists a 4-DS of G' whose size is k. Let $Y' = \{y_1', y_2', \ldots, y_k'\} \subseteq Y \subseteq V(G)$ be such 4-DS of G'. In G, what is the maximum hop distance from any sensor node v_0 to its nearest node in Y'? By Fact 1, there should be $y_i \in Y$ that d^*-hop dominates v_0 in G for some i, which implies $\mathrm{Hopdist}(v_0, y_i) \leq d^*$ in G. In G', y_i has to be 4-hop dominated by some node in Y', say y_1', which implies $\mathrm{Hopdist}(y_i, y_1') \leq 4(2d^*+1) = 8d^* + 4$ [from (3)]. As a result, we have $\mathrm{Hopdist}(v_0, y_1') \leq d^* + 4(2d^*+1) = 9d^* + 4$ (see Fig. 2). Now, suppose we place a sink on each node in Y' to solve k-SPP. Then, we have

$$\begin{aligned}\mathrm{Cost}(\mathrm{OPT}_{k\text{-CENTER}}) &\leq \max_{\forall v_j \in V} \min_{1 \leq i \leq k} \text{Hopdist } (v_j, y_i') \\ &\leq 9 \text{ Cost } (\mathrm{OPT}_{k\text{-SPP}}) + 4. \quad (4)\end{aligned}$$

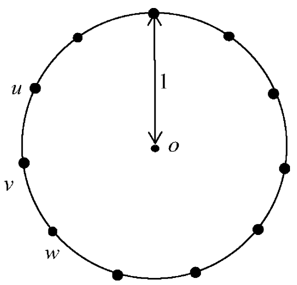


Fig. 3. For any two adjacent nodes u and v on the circle, their distance $\text{Eucdist}(u,v) = 2\sin(\pi/11) < 2\sin(\pi/6) = 1$. For any two nonadjacent nodes u and w, their distance $\text{Eucdist}(u,w) \geq 2\sin(2\pi/11) > 2\sin(\pi/6) = 1$.

Finally, from (2) and (4), we have $\text{Cost}(S) \leq 18\,\text{Cost}\,(\text{OPT}_{k\text{-SPP}}) + 8$. ■

C. Lower Bound of Approximation Ratio of GREEDY-k-CENTER for k-SPP

In Section IV-B, we showed that the approximation ratio (the worst-case performance ratio) of GREEDY-k-CENTER for k-SPP is 26. Here, we show the approximation ratio of GREEDY-k-CENTER for k-SPP has to be at least 5. Suppose a circle whose radius is exactly 1 and is centered at o (Fig. 3). Put 11 nodes dividing the circle evenly. Then, for any two adjacent nodes u and v on the circle, their distance $\text{Eucdist}(u,v) = 2\sin(\pi/11) < 2\sin(\pi/6) = 1$, and for any two nonadjacent nodes u and w, their distance $\text{Eucdist}(u,w) \geq 2\sin(2\pi/11) > 2\sin(\pi/6) = 1$. Now, consider a k-SPP problem instance, where $k = 1$. Then, the cost of an optimal solution for k-SPP over such a graph (Fig. 3) is 1, and this can be archived by putting the unique sink on the center (o). However, if we are forced to put the center on one of the 11 nodes on the border as with GREEDY-k-CENTER, then the cost of the solution should be 5 in this example. As a result, the lower bound of the approximation ratio of GREEDY-k-CENTER for k-SPP is 5.

V. Formal Definition of k-SPP and Its APX-Completeness

Now, we formally define k-SPP and prove its APX-completeness by showing that there is no constant factor approximation algorithm whose performance ratio is lower than 2 unless $P = NP$.

Definition 5.1 [k-Sink Placement Problem (k-SPP)]: Given a UDG G representing a WSN, k-SPP is to determine the positions of k sinks such that $\text{Cost}(S) = \max_{\forall i} \min_{\forall j} \text{Hopdist}(v_i, s_j)$ is minimized, where $\{v_1, v_2, \dots, v_{|V(G)|}\} = V(G)$ and $S = \{s_1, \dots, s_k\}$ is the set of k sinks.

Lemma 5.1 [21]: Suppose G is a planar graph with maximum degree 4. Then, G can be embedded in the plane using $O(|V|)$ area such that each element of $V(G)$ is at an integer coordinate and each element of $E(G)$ is drawn so that it is made up of a line segment of the form $x = i$ or $y = j$, for integers i and j.

Theorem 5.2: There is no $(2-\epsilon)$-approximation algorithm for k-SPP unless $P = NP$, where ϵ is a positive constant.

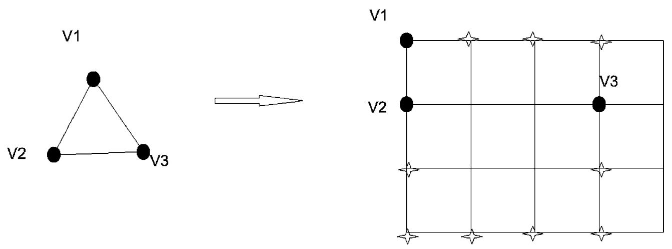


Fig. 4. Grid embedding of the graph K_3, where the stars denote the additional added nodes that make the obtained grid graph G_{p} (which is isomorphic to K_3) into a UDG G_{g}.

Proof: We prove that k-SPP is APX-complete in grid graphs with some special properties, which are in a subclass of UDGs. Clearly, if k-SPP is APX-complete for these grid graphs, then it is also APX-complete for UDGs. Let $\mathcal{G}_{\text{p}}$ be a family of planar graphs with maximum degree 3. By Lemma 5.1, we can embed graphs in $\mathcal{G}_{\text{p}}$ into a grid space to get a family of grid graphs. There can be more than one way to implement this idea. For our proof purpose, we are going to use the embedding scheme introduced by Clark *et al.* [22]. In detail, they embed each graph $G_{\text{p}} \in \mathcal{G}_{\text{p}}$ into the plane and have a grid graph G_{g} in such a way that we have the following.

1) For each edge of G_{p}, G_{g} has one or more corresponding line segments that are parallel to the x-axis or y-axis.
2) No two parallel lines in G_{g} are closer than three.
3) Each line segment in G_{g} has an integer length.
4) The sum of line segments' length in G_{g} for an edge $e = (x,y)$ in G_{p} is $3k_e + 1$ for some integer $k_e \geq 0$.

In this method, they add $3k_e$ additional nodes at grid point, which are equally spaced on each line segment representing an edge (x,y) of G_{p}. This makes the grid graph G_{p} into a UDG G_{g} (see Fig. 4 for an example). From now on, we denote the set of all such G_{g} obtained from $G_{\text{p}} \in \mathcal{G}_{\text{p}}$ by $\mathcal{G}_{\text{g}}$. It is known that finding a DS with minimum cardinality for graphs in $\mathcal{G}_{\text{p}}$, planar graphs with maximum degree 3 is NP-complete [23]. Using this fact and the grid embedding above, [22, Theorem 5.1] showed that finding a minimum DS in $\mathcal{G}_{\text{g}}$, so called the grid dominating set problem, is also NP-complete. However, [22] only provides a sketch of the proof of the theorem. Therefore, we give a formal proof of this fact in Lemma 5.3 for completeness.

In [16], it was shown that k-CENTER is APX-complete for general graphs. Now, using a similar argument, we show that k-CENTER remains to be APX-complete for graphs in $\mathcal{G}_{\text{g}}$. For each $G_{\text{g}} \in \mathcal{G}_{\text{g}}$, we construct an abstract edge-weighted complete graph K^{g} as follows: Set $V(K^{\text{g}}) = V(G_{\text{g}})$ and establish an edge between every pair of nodes. For each pair of node $u, v \in V(G_{\text{g}})$, use one as the edge weight of $(u,v) \in E(K^{\text{g}})$ if $(u,v) \in E(G_{\text{g}})$. Otherwise, assign two as the edge weight. Then, in K^{g}, the existence of a solution of k-CENTER with minimum cost one is equivalent to the existence of a DS in G_{g} whose size is less than or equal to k (the decision version of the dominating set problem in G_{g}), which is NP-complete according to our previous assertion. Therefore, we cannot have a $(2-\epsilon)$-approximation algorithm for k-CENTER for graphs $G_{\text{g}} \in \mathcal{G}_{\text{g}}$ unless $P = NP$.

To complete the proof, we show that k-SPP is equivalent to k-CENTER for graphs in $\mathcal{G}_{\text{g}}$. Suppose we have an optimal solution OPT to k-SPP for $G_{\text{g}} \in \mathcal{G}_{\text{g}}$ and $\text{Cost}(\text{OPT}) = d$. If

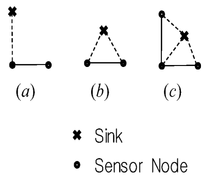


Fig. 5. All possible cases of a sink and its neighbors grid nodes are shown in Cases (a)–(c) in a G_{g}. In Cases (a) and (b), we move the sink to any of its neighboring grid points. In Case (c), we move the sink to the grid point in the middle. After the operations, if a sink is d-hop dominating a set of nodes in G_{g}, it still d-hop dominates the same set of nodes.

some sinks in OPT do not lie on a grid point, then we can always move each of them to one of its neighboring grid points to obtain another solution OPT$'$ due to the followings reasons. First of all, in OPT, each sink has at most three neighbors by our second and third construction rules for G_{g}. All possible situations for the comparative positions of a sink and its adjacent sensors are given in Fig. 5(a)–(c). In the cases of Fig. 5(a) and (b), we can move the sink to one of its neighboring grid points; in the case of Fig. 5(c), we can move the sink to its neighboring grid point in the middle. Note that in the case that a sink dominates two nodes whose distance is exactly two, we do not need to relocate the sink since it is on a grid point already. Given that the operations described above are performed, now we show that OPT$'$ still d-hop dominates the same set of nodes as OPT did. Let v be any node in G_{g} that is d-hop dominated by a sink $s \in$ OPT that is not on a grid point. In the cases of Fig. 5(a) and (b), v must be $(d-1)$-hop dominated by one of the grid points adjacent to s. Thus, after moving s onto one of the neighboring grid points, s still d-hop dominates v. Similarly, in the case of Fig. 5(c), v must be $(d-1)$-hop dominated by one of the three grid points adjacent to s. If v is $(d-1)$-hop dominated by the middle grid point (on the bottom left corner), after moving s to this point, v is $(d-1)$-hop far from s, and hence v is d-hop dominated by s. On the other hand, if v is $(d-1)$-hop dominated by one of the other two grid points, after moving s to the middle grid point, v is still d-hop dominated by s. This shows the equivalence of k-SPP and k-CENTER in $\mathcal{G}_{\mathrm{g}}$. Therefore, k-SPP is also APX-complete, and there is no $(2-\epsilon)$-approximation algorithm for it either. ■

Lemma 5.3: The problem of computing minimum cardinality dominating set in $\mathcal{G}_{\mathrm{g}}$ is NP-complete.

Proof: We show that $G_{\mathrm{p}} \in \mathcal{G}_{\mathrm{p}}$ has a DS D with $|D| \le k$ if and only if G_{g}, which is a grid graph computed from G_{p} following the construction rules, has a DS D' with

$$|D'| \le k + \sum_{e \in E(G_{\mathrm{p}})} k_e. \tag{5}$$

Suppose that $G_{\mathrm{p}} \in \mathcal{G}_{\mathrm{p}}$ has a DS D with $|D| \le k$. First, we claim that we can construct a DS D' of G_{g} satisfying (5). Assume that G_{p} is already embedded into the grid space. Let $e = (x, y)$ be an edge of G_{p}, the total length of the line segments representing which is $3k_e + 1$ for some integer k_e. Note that G_{g} is obtained from G_{p} by replacing the edge $e = (x, y)$ with a path $P = (x, w_1, w_2, \ldots, w_{3k_e}, y)$, where each w_i is the newly added grid point. Now, we construct D' as follows.

Rule 1-1: Include all nodes in D to D'.

Rule 1-2: For each $e \in E(G_{\mathrm{p}})$, if neither x nor y is in D, put nodes $w_2, w_5, \ldots, w_{3k_e-1}$ into D'. Note that all nodes in P except x and y are dominated by the selected nodes.

Rule 1-3: For each $e \in E(G_{\mathrm{p}})$, if either x or y is in D, say $x \in D$. Put $w_3, \ldots, w_{3k_e}$ into D'. Note that all nodes in path P are dominated by $x, w_3, \ldots, w_{3k_e}$.

Then, we have

$$|D'| = |D| + \sum_{e \in E(G_{\mathrm{p}})} k_e \le k + \sum_{e \in E(G_{\mathrm{p}})} k_e.$$

Next, we show that every node in G_{g} is dominated by D'. In fact, let v be any nodes in G_{g}. Suppose that v lies on a path $(x, w_1, w_2, \ldots, w_{3k_e}, y)$. Consider the following cases.

Case 1-1: If $v \ne x$ and $v \ne y$ and $x, y \notin D$, then v is dominated (or coincides with) by some nodes in $w_2, w_5, \ldots, w_{3k_e-1}$, according to Rule 1-2.

Case 1-2: If $v = x$ or y, say $v = x$, and $x, y \notin D$, then $v = x$ is adjacent to a node $u \in D$ in graph G_p. Assume that $e' = (u, v)$ is replaced by a path $(u, u_1, u_2, \ldots, u_{3k_{e'}}, v)$ in G_{g}. According to Rule 1-3, v is dominated by the node $u_{3k_{e'}}$ in graph G_{g}.

Case 1-3: If either $x \in D$ or $y \in D$, say $x \in D$, then v is either dominated by (or coincides with) x or some nodes in $w_3, \ldots, w_{3k_e}$.

Combining the analysis above, our first claim holds.

On the other hand, suppose that G_{g} has a DS D' of size $|D'| = k' + \sum_{e \in E(G_{\mathrm{p}})} k_e$ with $k' \le k$. We claim that G_{p} has a DS D of size $|D| \le k$. Let $e = (x, y)$ be any edge in G_{p} and $P = (x, w_1, w_2, \ldots, w_{3k_e}, y)$ the corresponding path in G_{g}. A key fact that we are going to use is that we always have $|D' \cap (P \setminus \{x, y\})| \ge k_e$. That is, there are at least k_e nodes in D' that lie on the interior of the path P. Now, construct D as follows.

Rule 2-1: Include all nodes $D' \cap V(G_{\mathrm{p}})$ into D.

Rule 2-2: For each edge $e = (x, y)$, if $|D' \cap (P \setminus \{x, y\})| \ge k_e + 1$, then include either x or y into D.

It is clear that $|D| \le k' \le k$. Next, we show that D is a DS of G_{p}. Let u be any nodes in G_{p} and $e = (u, v)$ be an edge in G_{p}. If either u or v is in D, then u is either in D or dominated by D. If neither u nor v is in D, we consider the following cases.

Case 2-1: There are at least $k_e + 1$ nodes in D' lying on the interior of the path with u, v as two endpoints. In this case, either u or v is in D—a contradiction. Thus, this case does not happen.

Case 2-2: There are exactly k_e nodes in D' lying on the interior of the path with u, v as two endpoints. Let $e' = (u', u)$ be an edge in G_{p}, and let the corresponding paths in G_{g} be $P' = (u', u_1, u_2, \ldots, u_{3k_{e'}}, u)$ such that u is dominated by $u_{3k_{e'}} \in D'$ in G_{g}. If $|D' \cap (P' \setminus \{u', u\})| \ge 1 + k_{e'}$, then u or u' is selected to D. Since it is assumed that $u \notin D$, we must have $u' \in D$, i.e., u is dominated by $u' \in D$. Otherwise, if $|D' \cap (P' \setminus \{u', u\})| = k_{e'}$, then we must have $u' \in D'$, and hence $u' \in D$ by our construction. Thus, u is still dominated by $u' \in D$.

Thus, in any case, u is either in D or dominated by D. Our second claim holds. Since finding a DS with minimum cardinality is NP-complete for graphs in $\mathcal{G}_{\mathrm{p}}$ [23], we can conclude that finding a minimum DS for graphs in $\mathcal{G}_{\mathrm{g}}$ is NP-complete, either. ■

VI. Better Approximation Algorithm for k-SPP

Previously, we showed that GREEDY-k-CENTER, an existing 2-approximation algorithm for k-CENTER, is a constant factor approximation algorithm for k-SPP. This algorithm is very simple and therefore it is fast and easy to be implemented in both centralized and distributed environments. However, this approach has a limitation. In fact, we can easily imagine a situation in which by allowing to put each sink on any place in euclidean space, we may find a better solution.

Algorithm 2: Get-Feasible-Solution-Space $(G(V,E))$

```
1: Set $\mathcal{F} \leftarrow \emptyset$.
2: For each pair of nodes $u$ and $v$ in $V(G)$ such that
   $\mathrm{Eucdist}(u,v) < 2$, suppose $w_1$ and $w_2$ are the centers of
   two unit disks whose borders are passing both $u$ and $v$.
   Set $\mathcal{F} \leftarrow \mathcal{F} \cup \{w_1, w_2\}$. In case that $\mathrm{Eucdist}(u,v) = 2$,
   we have only one such unit disk whose center is $w_3$. Then,
   set $\mathcal{F} \leftarrow \mathcal{F} \cup \{w_3\}$.
3: Return $\mathcal{F}$.
```

In this section, we introduce a better approximation algorithm for k-SPP. One reason that k-SPP seems very difficult to deal with is the feasible solution space of this problem is infinite. To overcome its high complexity, we construct a set of positions to locate the sinks such that an optimal solution is a subset of this set. We show the size of this set is actually finite and can be computed in a polynomial time: For each pair of sensor nodes within euclidean distance two, find at most two unit disks whose boundaries are passing through these two nodes. Then, the centers of all of such unit disks are potential candidates to place sinks (for a formal description, see Algorithm 2). Among all optimal solutions, we can always find a canonical one included in the candidate solution space (see Lemma 2.2).

Next, we apply out new approximation algorithm (Algorithm 3) on this set to find an approximate solution. The spirit of the algorithm is similar to the greedy algorithm for k-CENTER (Algorithm 1). However, here we fully employ the geometry of UDG to get a better result. As a result, we can show given a k-SPP instance, $\mathrm{Cost}(V_S) \leq 2\ \mathrm{Cost}(\mathrm{OPT}_{k\text{-SPP}}) + 1$, where V_S is an output of GREEDY-k-SPP. This means that even in the worst case, the cost of an output of this algorithm is within three times from the cost of an optimal solution. Furthermore, as the cost of $\mathrm{OPT}_{k\text{-SPP}}$ grows, the performance ratio of this algorithm is nearly 2.

Lemma 6.1 [24]: If a unit disk covers a set S of at least two sensor nodes, we can always move it to a position where it still covers all sensor nodes in S and its boundary passes through at least two sensor nodes in S (See Fig. 6).

Lemma 6.2: The output by Algorithm 2 always includes at least one optimal solution of k-SPP.

Proof: Let $\mathrm{OPT}_{k\text{-SPP}} = \{x_1, x_2, \ldots, x_k\}$ be an optimal solution for k-SPP. By Lemma 6.1, for each unit disk centered at x_i and a set O of nodes inside the disk such that $|O| \geq 2$, there has to be another unit disk whose boundary passes two nodes in O and still covers (i.e., contained in the disk or lay on the boundary of the disk) all other nodes in O. On the other hand,

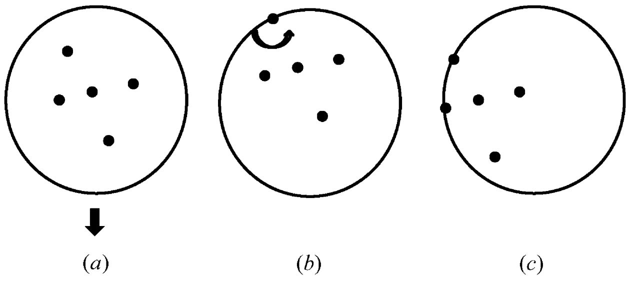


Fig. 6. (a) Move the unit disk down until its boundary meets a node in S. (b) Using the node on the boundary as a center, rotate the unit disk until its boundary meets another node in S. (c) The unit disk now has a position where it still covers S and its boundary passes through two nodes in S.

suppose $|O| = 1$, say $O = \{v\}$. Since the UDG G is connected, there has to be another sensor node u such that $\mathrm{Eucdist}(u,v) \leq 1$. Then, again, by Lemma 6.1, we can move the disk such that it passes through u and v. Suppose each x_i is moved to x'_i as described. Then, $\mathrm{OPT}'_{k\text{-SPP}} = \{x'_1, x'_2, \ldots, x'_k\} \subseteq \mathcal{F}$ (the output of Algorithm 2) is true, and $\mathrm{OPT}'_{k\text{-SPP}}$ is still an optimal solution for k-SPP. ■

Lemma 6.3: The time complexity of Algorithm 2 is $O(n^2)$, and the size of its output is bounded by $O(n^2)$.

Proof: In Algorithm 2, we add two nodes to S for each pair of nodes in $V(G)$. Therefore, $|\mathcal{F}| = O(n^2)$, and this can be done in a polynomial time. ■

Algorithm 3: GREEDY-k-SPP $(G(V,E), k)$

```
1: $\mathcal{F}$ = Get-Feasible-Solution-Space $(G)$
2: Set $V_C \leftarrow \emptyset$ and $V_S \leftarrow \emptyset$.
3: Randomly select $c_1 \in V(G)$ and set $V_C \leftarrow \{c_1\}$.
4: Select $s_1 \in \mathcal{F}$, which is adjacent to $c_1$, and set
   $V_S \leftarrow \{s_1\}$. A tie can be broken by randomly selecting
   one node.
5: **For** $i = 2$ to $k$ **do**
6:     Select $c_i \in V(G) - V_C$ such that
       $$\max_{\forall v_k \in V(G)\setminus V_C} \min_{\forall v_j \in V_S} \mathrm{Hopdist}(v_j, v_k) = \mathrm{Hopdist}(v_j, c_i).$$
7:     Select $s_i \in \mathcal{F} \setminus V_S$, which is adjacent to $c_i$. A tie can
       be broken arbitrarily.
8:     Set $V_C \leftarrow V_C \cup \{c_i\}$ and $V_S \leftarrow V_S \cup \{s_i\}$.
9: **end for**
10: For each node $v \in V_S$, place a sink $u$ very on each node $v$.
    If $v \in V(G)$, place a sink $u$ very near to each node $v$ such
    that $u$ can be connected to all neighbors of $v$.
```

Now, we prove that performance ratio of Algorithm 3.

Theorem 6.4: For any output V_S of Algorithm 3, $\mathrm{Cost}(V_S) \leq 2\ \mathrm{Cost}(\mathrm{OPT}_{k\text{-SPP}}) + 1$ is true.

Proof: This proof is basically a variation of the idea used to analyze the performance of GREEDY-k-CENTER (see Section IV-A) in conjunction with our problem-specific observation (see Lemma 2.2). Let $\mathrm{OPT}_{k\text{-SPP}} = \{x_1, x_2, \ldots, x_k\}$ be an optimal solution of k-SPP, and $V_i \subseteq V(G)$ be the subset of nodes whose nearest sink is $x_i \in \mathrm{OPT}_{k\text{-SPP}}$. Also, suppose

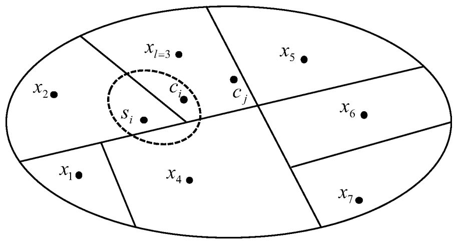


Fig. 7. Since $|X| = k$ and $|Y| = k + 1$, where $X = \{s_1, \ldots, s_k\}$ and $Y = \{c_1, \ldots, c_{k+1}\}$, there has to be at least one V_l that contains more than one element in Y.

$(s_1, c_1), (s_2, c_2), \ldots, (s_k, c_k)$ is the sequence of the pairs generated by Algorithm 3 and $V_S = \{s_1, s_2, \ldots, s_k\}$.

Clearly, $\mathrm{Cost}(V_S) = \max_{\forall i} \min_{1 \le j \le k} \mathrm{Hopdist}(v_i, s_j) = \min_{1 \le j \le k} \mathrm{Hopdist}(v, s_j)$, for some $v \in V(G) \setminus \{c_1, \ldots, c_k\}$. Let us denote v by c_{k+1} since v here is exactly the node that is going to be selected by the greedy strategy of Algorithm 3 if we want to place the $(k+1)$th sink. That is, once c_{k+1} is selected by the algorithm, a sink s_{k+1} will be positioned on one of the points in $\mathcal{F}$ that is adjacent to c_{k+1}. Since the cost of an output of Algorithm 3 is monotonically nonincreasing as the number of sinks increases, we have

$$\begin{aligned}\mathrm{Cost}(V_S) &= \max_{1 \le a \le k} \mathrm{Hopdist}(s_a, c_{k+1}) \\ &\le \mathrm{Hopdist}(s_i, c_j) \qquad (6)\end{aligned}$$

for any i and j satisfying $1 \le i < j \le k+1$. Note that the $k+1$ nodes, $c_1, c_2, \ldots, c_k, c_{k+1}$, are contained in $\bigcup_{i=1}^{k} V_i$. Then, by the Pigeonhole principle, there must exist a $V_l (1 \le l \le k)$ that contains c_i and c_j for $i \ne j$ (Fig. 7). Without loss of generality, suppose $1 \le i < j \le k+1$. Then, by (6), for any j, $\mathrm{Cost}(V_S) \le \mathrm{Hopdist}(s_i, c_j) \le \mathrm{Hopdist}(s_i, c_i) + \mathrm{Hopdist}(c_i, c_j) \le \mathrm{Hopdist}(s_i, c_i) + \mathrm{Hopdist}(c_i, x_l) + \mathrm{Hopdist}(x_l, c_j) \le 1 + 2\ \mathrm{Cost}(\mathrm{OPT}_{k\text{-SPP}})$ since $\mathrm{Hopdist}(s_i, c_i) \le 1$, $\mathrm{Hopdist}(c_i, x_l) \le \mathrm{Cost}(\mathrm{OPT}_{k\text{-SPP}})$, and $\mathrm{Hopdist}(x_l, c_j) \le \mathrm{Cost}(\mathrm{OPT}_{k\text{-SPP}})$. ■

VII. Simulation Results

In this section, we compare the average performance of GREEDY-k-SPP with its alternative, GREEDY-k-CENTER. We prepare a 100×100 space and randomly deploy a number of nodes. In this simulation, the number of nodes is 50, 60, 70, 80, 90, or 100. We check if the UDG induced from the nodes is connected. If not, we dump the graph away and generate another one until it is connected. The reason for this operation is that if the graph consists of more than k connected subgraphs and any two subgraphs are more than 2 units distant far from each other, the cost of both algorithms over such a graph will be infinite since there is no way for the k sinks to multihop dominate all nodes in the UDG. In fact, such a graph instance only prevents us from making a fair comparison between the algorithms. In practice, in many applications of WSNs, the number of sensor nodes is huge, and they are likely to be connected or can be connected by simply deploying more nodes, whose cost is very low. Due to the reason, lots of existing papers for WSNs have this assumption. After all, this is why this paper assumes the input graphs are connected. Once a valid UDG is generated, we apply both GREEDY-k-CENTER and GREEDY-k-SPP to find the locations of k sinks, where $k = 3$ or 6. We vary the maximum transmission range of nodes to 15, 20, and 25. For each parameter setting, we execute each algorithm over 100 different graphs and compute the average cost of their outputs. Table I is the summary of our simulation results. In the table, N and T represent the number of nodes and the maximum transmission range, respectively.

Let us compare Table I(a)–(c). In those tables, we fix k to 3 and T is 15, 20, and 25, respectively. By comparing the three tables, we can observe that the cost of both algorithms is decreased when we make the value of T large. This is natural since, with larger T, the diameter of an input graph is reduced and the graph is more connected, and thus and the whole graph can be dominated more efficiently by the same number of sinks. On the other hand, the cost of their output increases as the number of nodes grows. This is also easy to understand since the diameter of a graph with more nodes is expected to be larger. Most importantly, from the tables, we can see that GREEDY-k-SPP outperforms GREEDY-k-CENTER under any T and N, given $k = 3$. The performance difference between them becomes more apparent when the number of nodes and the value of T are smaller. This can be also verified by the improvement ratio, which is

TABLE I
Averaged Cost Comparison of the Outputs of GREEDY-k-CENTER and GREEDY-k-SPP, Where k Is the Number of Available Sinks, T Is the Maximum Transmission Range of Each Node, and N Is the Number of Nodes. (a) $k = 3$ and $T = 15$. (b) $k = 3$ and $T = 20$. (c) $k = 3$ and $T = 25$. (d) $k = 6$ and $T = 15$. (e) $k = 6$ and $T = 20$. (f) $k = 6$ and $T = 25$

N	50	60	70	80	90	100
GREEDY-k-CENTER	3.86	4.23	4.45	4.64	4.79	4.97
GREEDY-k-SPP	3.43	3.87	4.00	4.22	4.45	4.62
Improvement (%)	11.14	8.51	10.11	9.05	7.1	7.04

(a)

N	50	60	70	80	90	100
GREEDY-k-CENTER	3.54	3.82	3.94	4.06	4.05	4.23
GREEDY-k-SPP	3.17	3.58	3.63	3.71	3.70	3.93
Improvement (%)	10.45	6.28	7.87	8.62	8.64	7.09

(b)

N	50	60	70	80	90	100
GREEDY-k-CENTER	3.17	3.27	3.23	3.32	3.39	3.30
GREEDY-k-SPP	2.91	3.00	3.07	3.14	3.04	3.13
Improvement (%)	8.20	8.26	4.95	5.42	10.32	5.15

(c)

N	50	60	70	80	90	100
GREEDY-k-CENTER	2.44	2.64	2.80	2.97	3.19	3.20
GREEDY-k-SPP	2.09	2.25	2.40	2.56	2.79	2.91
Improvement (%)	14.34	14.77	14.29	13.8	12.54	9.06

(d)

N	50	60	70	80	90	100
GREEDY-k-CENTER	2.16	2.32	2.45	2.55	2.69	2.74
GREEDY-k-SPP	2.05	2.05	2.11	2.25	2.33	2.41
Improvement (%)	5.09	11.64	13.88	11.76	13.38	12.04

(e)

N	50	60	70	80	90	100
GREEDY-k-CENTER	2.02	2.03	2.00	2.04	2.01	2.01
GREEDY-k-SPP	1.99	2.00	2.02	2.02	2.02	2.01
Improvement (%)	1.49	1.48	-0.01	0.98	-0.49	0.0

(f)

$$\frac{(\text{cost of GREEDY-}k\text{-CENTER} - \text{cost of GREEDY-}k\text{-SPP})}{(\text{cost of GREEDY-}k\text{-CENTER})/100\%}$$

shown at the top of the page. Therefore, we can conclude that in a sparse graph, GREEDY-k-SPP works especially better than GREEDY-k-CENTER.

Now, let us look at Table I(d)–(f). In those tables, we fix k to 6 and T is set to 15, 20, and 25, respectively. In general, we can see the trend that we observed from the first three tables still exists. Interestingly, the performance gap between GREEDY-k-CENTER and GREEDY-k-SPP becomes small after k is set to 6. This is because once a certain number of sinks are used to cover the given graph, they can reduce the cost of an output sufficiently, and the last few sinks can reduce the cost of the output less significantly than the first several sinks. For example, in Table I(a), the average cost of the outputs by GREEDY-k-CENTER is 4.97 when $k = 3$ and $n = 100$. However, as we can see from Table I(d), the average cost of the outputs by GREEDY-k-CENTER is 3.20 when $k = 6$ and $n = 100$. That is, by adding three more sinks, we only reduced the cost by $4.97 - 3.20 = 1.77$.

In conclusion, our simulation results indicate that GREEDY-k-SPP outperforms GREEDY-k-CENTER on average. In the earlier section, we showed GREEDY-k-CENTER is a 26-approximation algorithm for k-SPP, and this approximation ratio cannot be improved better than 5. We also showed that the approximation ratio of GREEDY-k-SPP is 3. As a result, we can conclude that GREEDY-k-SPP outperforms GREEDY-k-CENTER completely.

VIII. Conclusion and Future Work

In this paper, we provide a basic framework to best construct WSNs for time-sensitive applications with a limited number of sinks. We recognized that: 1) for those applications, it is important to ensure that the maximum node to sink message latency is bounded; and 2) it is better to make the bound smaller to improve the performance of those real-time systems. We also realized that we can design such a WSN by employing multiple sinks and make it more effective by carefully selecting the locations of the sinks. Motivated by our observations, we introduce the k-Sink Placement Problem (k-SPP), whose goal is minimizing the maximum hop distance between each node and its nearest sink. We showed that this problem is APX-complete and proved that an existing approximation algorithm for the metric k-center problem is still an approximation of k-SPP. Then, we introduced a new greedy algorithm for k-SPP and showed its approximation ratio is very near to the best achievable, 2. In simulations, we showed our new algorithm works better than its competitor on average.

In Section IV, we showed the approximation ratio of GREEDY-k-CENTER is 26 for k-SPP, but cannot be better than 5. Our result gives a nonnegligible gap between the lower bound and upper bound of the worst-case performance ratio of GREEDY-k-CENTER for k-SPP, and we are interested in further closing this gap. We also believe that while our algorithm is useful, it can be improved in many aspects. First, we assumed homogeneous WSNs and used UDGs to model the networks. However, in reality, it is necessary to assume that WSNs are nonhomogeneous. In this case, this problem should be considered in a disk graph model. Second, we assume the topology of the input WSNs is flat and used UDGs to model the graphs. However, this may not be true in some occasions. Therefore, studying this problem in 3D-WSN could be a very interesting research direction.

Acknowledgment

The authors would like to acknowledge Prof. Z. Zhang from Xinjiang University, Urumqi, China, for her valuable comments to improve the quality of this paper. At last, the authors also acknowledge the editors and the reviewers for their time and efforts in handling this paper.

References

[1] M. Younis, M. Bangad, and K. Akkaya, “Base-station repositioning for optimized performance of sensor networks,” in *Proc. IEEE VTC*, Orlando, FL, Oct. 2003, vol. 5, pp. 2956–2960.

[2] A. Efrat, S. Har-Peled, and J. S. B. Mitchell, “Approximation algorithms for two optimal location problems in sensor networks,” in *Proc. BroadNets*, Boston, MA, Oct. 2005, vol. 1, pp. 714–723.

[3] S. R. Gandham, M. Dawande, R. Prakash, and S. Venkatesan, “Energy efficient schemes for wireless sensor networks with multiple mobile base stations,” in *Prof. IEEE GLOBECOM*, San Francisco, CA, Dec. 2003, vol. 1, pp. 377–381.

[4] A. Bogdanov, E. Maneva, and S. Riesenfeld, “Power-aware base station positioning for sensor networks,” in *Proc. IEEE INFOCOM*, Hong Kong, China, Mar. 2004, vol. 1, pp. 575–585.

[5] E. I. Oyman and C. Ersoy, “Multiple sink network design problem in large scale wireless sensor networks,” in *Proc. IEEE ICC*, Paris, France, Jun. 2004, vol. 6, pp. 3663–3667.

[6] R. Chandra, L. Qiu, K. Jain, and M. Mahdian, “Optimizing the placement of Internet TAPs in wireless neighborhood networks,” in *Proc. IEEE ICNP*, Berlin, Germany, Oct. 2004, pp. 271–282.

[7] H. Kim, Y. Seok, N. Choi, Y. Choi, and T. Kwon, “Optimal multi-sink positioning and energy-efficient routing in wireless sensor networks,” in *Proc. ICOIN*, Jeju, Korea, Jan.–Feb. 2005, pp. 350–359.

[8] W. Hua, C. T. Choua, S. Jhaa, and N. Bulusu, “Deploying long-lived and cost-effective hybrid sensor networks,” *Ad Hoc Netw.*, vol. 4, no. 6, pp. 749–767, Nov. 2006.

[9] Z. Vincze, K. Fodor, R. Vida, and A. Vidács, “Electrostatic modelling of multiple mobile sinks in wireless sensor networks,” in *Proc. IFIP Netw. Workshop Perform. Control Wireless Sensor Netw.*, Coimbra, Portugal, May 2006, pp. 30–37.

[10] H. Kim, T. Kwon, and P. Mah, “Multiple sink positioning and routing to maximize the lifetime of sensor networks,” *IEICE Trans. Commun.*, vol. E91-B, no. 11, pp. 3499–3506, 2008.

[11] Z. Vincze, R. Vida, and A. Vidacs, “Deploying multiple sinks in multi-hop wireless sensor networks,” in *Proc. IEEE ICPS*, Jul. 15–20, 2007, pp. 55–63.

[12] W. Youssef and M. Younis, “Intelligent gateways placement for reduced data latency in wireless sensor networks,” in *Proc. IEEE ICC*, Glasgow, Scotland, Jun. 2007, pp. 3805–3810.

[13] D. B. Shmoys, É. Tardos, and K. Aardal, “Approximation algorithms for facility location problems,” in *Proc. 29th Annu. ACM Symp. Theory Comput.*, 1997, pp. 265–274.

[14] D. S. Johnson, “Approximation algorithms for combinatorial problems,” *J. Comput. Syst. Sci.*, vol. 9, pp. 256–278, 1974.

[15] N. Patwari, J. Ash, S. Kyperountas, I. Hero, A. O. R. Moses, and N. Correal, "Locating the nodes: Cooperative localization in wireless sensor networks," *IEEE Signal Process. Mag.*, vol. 22, no. 4, pp. 54–69, Jul. 2005.

[16] D. S. Hochbaum and D. B. Shmoys, "A unified approach to approximation algorithms for bottleneck problems," *J. ACM*, vol. 33, no. 3, pp. 533–550, 1986.

[17] M. E. Dyer and A. M. Frieze, "A simple heuristic for the p-center problem," *Oper. Res. Lett.*, vol. 3, no. 6, pp. 285–288, Feb. 1985.

[18] T. F. Gonzalez, "Clustering to minimize the maximum intercluster distance," *Theoret. Comput. Sci.*, vol. 38, pp. 293–306, 1985.

[19] D. S. Hochbaum and D. B. Shmoys, "A best possible heuristic for the k-center problem," *Math. Oper. Res.*, vol. 10, no. 2, pp. 180–184, 1985.

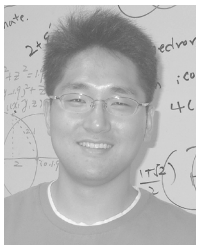
[20] W. Wu, H. Du, X. Jia, Y. Li, and S. Huang, "Minimum connected dominating sets and maximal independent sets in unit disk graphs," *Theoret. Comput. Sci.*, vol. 352, pp. 1–7, 2006.

[21] L. Valiant, "Universality considerations in VLSI circuits," *IEEE Trans. Comput.*, vol. C-30, no. 2, pp. 135–140, Feb. 1981.

[22] B. N. Clark, C. J. Colbourn, and D. S. Johnson, "Unit disk graphs," *Discr. Math.*, vol. 86, pp. 165–177, Dec. 1990.

[23] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York: Freeman, 1979.

[24] J. Tang, B. Hao, and A. Sen, "Relay node placement in large scale wireless sensor networks," *Comput. Commun.*, vol. 29, pp. 490–501, 2006.



Donghyun Kim (M'10) received the B.S. degree in electronic and computer engineering and M.S. degree in computer science and engineering from Hanyang University, Ansan, Korea, in 2003 and 2005, respectively, and the Ph.D. degree in computer science from the University of Texas at Dallas, Richardson, in 2010.

He is currently an Assistant Professor with the Department of Mathematics and Computer Science, North Carolina Central University, Durham. His research interests include wireless networks, mobile computing, distributed computing, and approximation algorithm design and analysis.

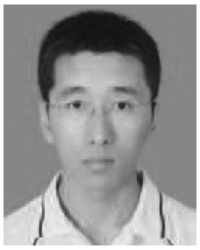
Dr. Kim is a member of the Association for Computing Machinery (ACM).

Wei Wang received the B.S. degree in applied mathematics from ZheJiang University, Hangzhou, China, in 1991, and the M.S. degree in computational mathematics and Ph.D. degree in mathematics from Xi'an Jiaotong University, Xi'an, China, in 1994 and 2006, respectively.

He is currently an Assistant Professor with the Department of Mathematics, Xi'an Jiaotong University. His research interests include algebraic graph theory and approximation algorithm design and analysis.

Nassim Sohaee received the B.S. degree in mathematics from Amir-Kabir University of Technology, Tehran, Iran, in 1998, the M.S. degree in mathematics from Sharif University of Technology, Tehran, Iran, in 2000, the Ph.D. degree in mathematics from the University of Texas at Arlington in 2003, and the Ph.D. degree in computer science from the University of Texas at Dallas in 2009.

She is currently a Post-Doctoral Fellow with the University of Texas Southwestern Medical Center, Dallas. Her area of research interests include design and optimization of underwater sensor networks, protein complex prediction, and biological network alignment.



Changcun Ma is currently a Ph.D. student with the Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing, China.

He is interested in algorithms design and coding theory.

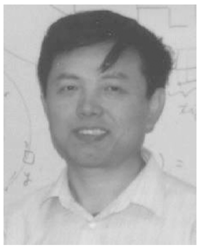
Weili Wu (M'09) received the Ph.D. degree in computer science from the University of Minnesota, Minneapolis, in 2002.

She is currently an associate professor in the Department of Computer Science at the University of Texas at Dallas. She has produced a number of research papers in spatial data mining, distributed database systems, and algorithm design. She was also a co-editor of a book on clustering and information retrieval. Her main research interest is in database systems.

Wonjun Lee (SM'06) received the B.S. and M.S. degrees in computer engineering from Seoul National University, Seoul, Korea in 1989 and 1991, respectively, the M.S. degree in computer science from the University of Maryland, College Park, in 1996, and the Ph.D. degree in computer science and engineering from the University of Minnesota, Minneapolis, in 1999.

In 2002, he joined the faculty of Korea University, Seoul, Korea, where he is currently a Professor with the Department of Computer Science and Engineering, Director of the World Class University Future Network Optimization Technology Center (WCU-FNOT), and Director of the Future Network Center (FNC). He has also held a faculty position with the University of Missouri–Kansas City. He has authored or coauthored over 115 papers in refereed international journals and conferences. His research interests include mobile wireless communication protocols and architectures, cognitive radio networking, and VANET.

Prof. Lee served as a Technical Program Committee member for IEEE INFOCOM 2008–2011, ACM MobiHoc 2008 and 2009, IEEE ICCCN 2000–2008, and over 110 international conferences.



Ding-Zhu Du (M'09) received the M.S. degree from the Chinese Academy of Sciences, Beijing, China, in 1982, and the Ph.D. degree from the University of California, Santa Barbara, in 1985.

Before settling at the University of Texas at Dallas, he was a Professor with the Department of Computer Science and Engineering, University of Minnesota, Minneapolis. He also was with the Mathematical Sciences Research Institute, Berkeley, CA, from 1985 to 1986; with the Department of Mathematics, Massachusetts Institute of Technology, Cambridge, from 1986 to 1987; and with the Department of Computer Science, Princeton University, Princeton, NJ, from 1990 to 1991. More than 40 Ph.D. students have graduated under his supervision.

Dr. Du is the Editor-in-Chief of the *Journal of Combinatorial Optimization* and is also on the editorial boards of several other journals.