On bounding node-to-sink latency in wireless sensor networks with multiple sinks

Donghyun Kim*

Department of Mathematics and Computer Science
North Carolina Central University,
Durham, NC 27707, USA
Email: donghyun.kim@nccu.edu
*Corresponding author

Wei Wang

Department of Mathematics,
Xi'an Jiaotong University,
Xi'an, 710049, China
Email: wang_weiw@163.com

Weili Wu

Department of Computer Science,
University of Texas at Dallas,
Richardson, TX 75083, USA
Email: weiliwu@utdallas.edu

Deying Li

School of Information,
Renmin University of China,
Beijing, 100872, China
Email: deyingli@ruc.edu.cn

Changcun Ma

Institute for Theoretical Computer Science,
Tsinghua University,
Beijing 100084, China
Email: mcc03@mails.tsinghua.edu.cn

Nassim Sohaee

Division of Liberal Arts & Life Sciences,
University of North Texas at Dallas,
Dallas, TX 75241, USA
Email: nassim.sohaee@unt.edu

Wonjun Lee

Department of Computer Science and Engineering,
Korea University,
Seoul, 136-713, South Korea
Email: wlee@korea.ac.kr

Yuexuan Wang

Institute for Theoretical Computer Science,
Tsinghua University,
Beijing, 100084, China
Email: wangyuexuan@tsinghua.edu.cn

Copyright © 2013 Inderscience Enterprises Ltd.

Ding-Zhu Du

Department of Computer Science,
University of Texas at Dallas,
Richardson, TX 75083, USA
and
WCU FNOT Research Center,
Korea University,
Seoul, 136-713, South Korea
Email: dzdu@utdallas.edu

Abstract: Bounding node-to-sink latency is an important issue of wireless sensor networks (WSNs) with a quality of service requirement. This paper proposes to deploy multiple sinks to control the worst case node-to-sink data latency in WSNs. The end-to-end latency in multihop wireless networks is known to be proportional to the hop length of the routing path that the message moves over. Therefore, we formulate the question of what is the minimum number of sinks and their locations to bound the latency as the minimum d-hop sink placement problem. We also consider its capacitated version. We show problems are NP-hard in unit disk graph (UDG) and unit ball graph, and propose constant factor approximations of the problems in both graph models. We further extend our algorithms so that they can work well in more realistic quasi UDG model. A simulation study is also conducted to see the average performance of our algorithms.

Keywords: WSNs; wireless sensor networks; multiple sink placement; relay node placement; mobile computing; approximation algorithm; graph theory; UDG; unit disk graph; UBG; unit ball graph; quasi unit disk graph.

Reference to this paper should be made as follows: Kim, D., Wang, W., Wu, W., Li, D., Ma, C., Sohaee, N., Lee, W., Wang, Y. and Du, D-Z. (2013) 'On bounding node-to-sink latency in wireless sensor networks with multiple sinks', *Int. J. Sensor Networks*, Vol. 13, No. 1, pp.13–29.

Biographical notes: Donghyun Kim is an Assistant Professor in the Department of Mathematics and Computer Science at the North Carolina Central University, Durham, USA. He received the BS degree in Electronic and Computer Engineering from the Hanyang University, Ansan, Korea (2003), and the MS degree in Computer Science and Engineering from Hanyang University, Korea (2005). He received the PhD degree in Computer Science from the University of Texas at Dallas, Richardson, USA (2010). His research interests include wireless networks, mobile computing, and approximation algorithm design and analysis.

Wei Wang received the BS degree from ZheJiang University, China (1991) and MS degree and PhD degree from Xi'an Jiaotong University, China (1994, 2006) respectively. He is currently an Assistant Professor in the Department of Mathematics at Xi'an Jiaotong University. His research interests include algebraic graph theory and approximation algorithm design and analysis.

Weili Wu received the PhD degree in Computer Science from the University of Minnesota (2002). Her main research interest is in database systems. 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. She is currently an Associate Professor in the Department of Computer Science at the University of Texas at Dallas. She is a member of the IEEE.

Deying Li received the MS degree in Mathematics from Huazhong Normal University (1988) and PhD degree in Computer Science from City University of Hong Kong (2004). She is currently a Professor in the Department of Computer Science, Renmin University of China. Her research includes wireless networks, mobile computing and algorithm design and analysis.

Changcun Ma is a PhD student at the Institute for Interdisciplinary Information Sciences at Tsinghua University, China and under the supervision of Prof. Andrew C. Yao. He is interested in algorithms design and coding theory.

Nassim Sohaee received her PhD in Computer Science from University of Texas at Dallas (2009), PhD in Mathematics from University of Texas at Arlington (2003), a MS in Mathematics form Sharif University of Technology, Tehran, Iran (2000), and a BS in Mathematics from Amir-Kabir University of Technology, Tehran, Iran (1998). She is currently a Lecturer at Division of Liberal Arts and Life Sciences at University of North Texas at Dallas. Her area of research interests include, design and optimisation of underwater sensor networks, protein complex prediction, and biological network alignment.

Wonjun Lee received BS and MS degrees in Computer Engineering from Seoul National University, Seoul, Korea (1989, 1991), respectively. He also received an MS in Computer Science from the University of Maryland, College Park, USA in 1996 and a PhD in Computer Science and Engineering from the University of Minnesota, Minneapolis, USA (1999). In 2002, he joined the faculty of Korea University, Seoul, Korea. He has held the faculty position at the University of Missouri – Kansas City, USA. His research interests include mobile wireless communication protocols and architectures, cognitive radio networking, and VANET.

Yuexuan Wang is Professor at Tsinghua University, Deputy Dean of Institute for Interdisciplinary Information Sciences (IIIS) and Director of "Tsinghua Xuetang Special Pilot CS Class". She received her PhD from Zhejiang University (2003). Her research interests include quantum network and wireless sensor network. She is a senior member of China Computer Federation and member of China Computer Federation Technical Committee on Sensor Network. She has already published over 50 papers on computer science in high-ranking journals and international conferences, and has been member of program committee for MobiHoc and several other international conferences.

Ding-Zhu Du received the MS degree from the Chinese Academy of Sciences (1982) and the PhD degree from the University of California at Santa Barbara under the supervision of Professor Ronald V. Book (1985). Before settling at the University of Texas at Dallas, he worked as a Professor in the Department of Computer Science and Engineering, University of Minnesota. He also worked at the Mathematical Sciences Research Institute, Berkeley, for one year, in the Department of Mathematics, Massachusetts Institute of Technology for one year, and in the Department of Computer Science, Princeton University for one and a half years.

1 Introduction

The recent advances in microelectronic technologies have enabled a whole new kind of network, namely Wireless Sensor Network (WSN). WSN has a broad range of important applications and thus attracted lots of attentions recently (Akyildiz et al., 2002). A mechanism for predictable node-to-sink latency is an important building block of WSNs with a Quality of Service (QoS) requirement. Owing to the reason, many efforts have been made to bound the node-to-sink data latency in WSNs (Sohrabi et al., 2000; Caccamo et al., 2002; Lu et al., 2002; Akkaya and Younis, 2003; He et al., 2003; Felemban et al., 2005). In most cases, it has been implicitly assumed that there exists a node-to-sink routing path satisfying an application's worst case node-to-sink latency requirement if certain conditions are met (e.g. lower congestion level, proper routing path), and such path was secured using various mechanisms such as congestion control, proper routing path selection and packet admission control.

It is well known that, in multi-hop routing, end-to-end latency increases proportional to the number of hops in the routing path due to queuing and processing delay at each intermediate node, which is significantly greater than the propagation delay in radio communication (Youssef and Younis, 2007; Misra et al., 2008). Therefore, if the hop distance from a node to a sink is very large, it will be very difficult to support a very tight node-to-sink data latency requirement of some WSNs. To deal with this issue, we propose to deploy multiple sinks (e.g. miniature helicopters, vehicles, robots) in proper locations which can relay the messages they collected from the nodes to the users directly. Intuitively, this approach will help to reduce the worst case data collection latency of a WSN. In addition, our approach can be harmoniously used with the other existing latency control mechanisms. Despite the fact that adopting multiple sinks may increase the cost of building a WSN, this could be still a promising approach for many node-to-sink delay sensitive applications such as intrusion detection and earthquake monitoring over a vast area in which we cannot design a WSN satisfying a rigorous node-to-sink delay requirement only using a single sink.

To the best of our knowledge, the idea of exploiting multiple sinks (or relay nodes) to bound the node-to-sink latency in WSNs is rarely discussed in the literatures (Kim et al., 2011). In fact, most existing researches about multiple sink (or relay node) deployment problems were focused on how to utilise the sinks (or relay nodes) to meet a certain connectivity requirement (Zhang et al., 2007; Misra et al., 2008) or to maximise network lifetime (Lloyd and Xue, 2007). Therefore, the existing researches on sink (or relay node) placement problems do not apply to our case, and new algorithms have to be investigated.

This paper considers a popular heterogeneous WSN model (Akyildiz and Kasimoglu, 2004; Rezgui and Eltoweissy, 2007; Cheng et al., 2008; Wang et al., 2009a; Wang et al., 2009b; Kim et al., 2010), which consists of (a) several sinks (or relay nodes) with a huge battery and a long range communication device (e.g. satellite transceiver) to forward the messages to users directly and (b) numerous cheap static sensor nodes. This model is a nice abstraction of several important WSNs such as battlefield surveillance where a set of cheap lightweight sensor nodes are deployed from aeroplane and a set of properly equipped vehicles or miniature flying machines (e.g. helicopters) act as sinks. Given that multiple sinks are available, it is apparent that the worst case node-to-sink latency in such system can be better controlled by carefully locating the available sinks. It is also important to notice that such sinks are generally expensive to purchase and operate. This observation arises the following interesting question: given a user's maximum tolerable node-to-sink latency bound, what is the minimum number of sinks needed and their locations? In the rest of this paper, we strive to answer this question. The major contributions of this paper are as follows.

1 It is known that in wireless networks, the worst case end-to-end data latency is proportional to the hop distance between them (Youssef and Younis, 2007; Misra et al., 2008). Inspired by this fact, we introduce a new approach to bound the worst case maximum data collection latency in WSNs using minimum number of sinks. We define the problem of deploying the minimum number of sinks to meet a given worst case node-to-sink latency requirement as the Minimum d-Hop Sink Placement (MdHSP) problem. Note that this multiple sink placement strategy can be used together with the other existing ones to efficiently bound the worst case data collection latency of a WSN.

2 In a WSN, a sink becomes a bottleneck of the network if there are too many nodes which desire to send messages to the sink. Apparently, as the WSN becomes busier, this situation will be aggravated and it will be very difficult to control the worst case data collection latency. In other word, if we limit the number of nodes forwarding messages to the same sink, the worst case node-to-sink latency can be controlled more effectively. Based on this observation, we introduce the capacitated version of M*d*HSP, namely the Capacitated Minimum d-Hop Sink Placement (CM*d*HSP) problem, is equivalent to M*d*HSP with one additional requirement that the number of sensor nodes serviced by each sink is restricted by some given constant.

3 We prove both of M*d*HSP and CM*d*HSP are NP-hard in both Unit Disk Graph (UDG) and Unit Ball Graph (UBG). Then, we propose constant factor approximation algorithms for them. In detail, we show a simple *d*-hop colouring algorithm (Algorithm 1) is $O(d^2)$-approximation algorithm for M*d*HSP in UBG. As a corollary, we prove this colouring strategy is an $O(d)$-approximation for M*d*HSP in UDG. Furthermore, we introduce a simple $O(d^2)$-approximation algorithm (Algorithm 2) for CM*d*HSP in UBG. We also show this approach is an $O(d)$-approximation for CM*d*HSP in UDG.

4 Due to the salient features of wireless communication such as signal collision and interference, UDG and UBG sometimes do not abstract 2-D and 3-D wireless networks precise enough for some applications of WSNs. To overcome this limitation, we study the M*d*HSP and CM*d*HSP in δ-quasi Unit Disk Graph (δ-qUDG),[1] which is a more realistic abstraction of wireless network than UDG (Kuhn et al., 2003), and propose new heuristic algorithms for them. While we only consider δ-qUDG, this algorithm also works in 3-D counterpart, namely δ-qUBG. We also describe how CM*d*HSP can be solved in δ-qUDG and δ-qUBG.

The rest of this paper is organised as follows. Section 2 introduces several notations, definitions and assumptions. Our main results including the constant factor approximations for M*d*HSP and CM*d*HSP are presented in Section 3. The extensions of our algorithms for M*d*HSP and CM*d*HSP in δ-qUDG are presented in Section 4. The simulation results and their analysis are given in Section 5. Section 6 introduces related work. Finally, Section 7 concludes this paper and presents some future works. The NP-hardness proofs of M*d*HSP and CM*d*HSP are presented in Appendix.

2 Notations, definitions and assumptions

Notations and definitions: in this paper, $G = (V, E)$ is a graph with a vertex set $V = V(G)$ and an edge set $E = E(G)$. Depending on the context, G can be either UDG, UBG or δ-qUDG. $Hopdist(u,v)$ means the number of edges on the shortest path between two nodes u and v. $Eucdist(u,v)$ is the Euclidean distance between two nodes u and v. $N[v]$ is the set of nodes neighbouring to v. $N[v]$ is $\{v\} \cup N[v]$. $N_d[v] = \{u \mid u \in V$ such that $u \neq v, Hopdist(v,u) \leq d\}$. $N_d(v) = \{v\} \cup N_d[v]$.

Definition 1 (Unit disk graph – UDG): *G in the 2-D Euclidean space is called an UDG, if for any two different nodes* $u, v \in V$, $(u,v) \in E$ *if and only if the Euclidean distance between u and v is at most one.*

Definition 2 (Unit ball graph – UBG): *G in the 3-D Euclidean space is called an UBG, if for any two different nodes* $u, v \in V$, $(u,v) \in E$ *if and only if the Euclidean distance between u and v is at most one.*

Definition 3 (δ-quasi unit disk graph – δ-qUDG): *G in the 2-D Euclidean space is called a δ-qUDG for a given constant* $0 < \delta \leq 1$ *if for each pair* $u, v \in V$, $Eucdist(u,v) \leq \delta \rightarrow (u,v) \in E$ *and* $Eucdist(u,v) > 1 \rightarrow (u,v) \notin E$.

In δ-qUDG, there will always be a communication link between two nodes if they are close enough (i.e. at most δ apart from each other). On the other hand, there will be no link if they are far enough from each other (i.e. greater than 1). For moderate distance (i.e. $\delta < Eucdist(u,v) \leq 1$), the existence of a communication link will be dependent on the runtime environmental factors such as ambient noise and interference level. It is known that δ-qUDG models real-world wireless networks much better than UDG (Kuhn et al., 2003). In the rest of the paper, we will assume that there will be a link between two nodes u, v such that $\delta < Eucdist(u,v) \leq 1$ with some known probability $0 < p < 1$. Note that if $\delta = 1$, then a δ-qUDG is equivalent to an UDG.

Definition 4 (Dominating Set – DS): *$D \subseteq V$ is a DS of G if $\forall v \in V$, either $v \in D$ or $\exists u \in D$ such that $(v,u) \in E$.*

Definition 5 (Independent Set – IS): *$I \subset V$ is an IS of G if $\forall u, v \in I$, $(u,v) \not\subseteq E$.*

Definition 6 (Maximal Independent Set – MIS): *An IS I is a MIS if for any $v \in (V/I)$, $I \cup \{v\}$ is not an IS.*

Note that an MIS of *G* is a DS of *G*. Also, finding a minimum DS in UDG is a NP-hard problem (Johnson, 1974), and thus is NP-hard in UBG.

Definition 7 (d-hop dominating set – d-DS): *$D \subseteq V$ is a d-DS of G if $\forall v \in V$, either $v \in D$ or $\exists u \in D$ such that $v \in N_d(u)$.*

Definition 8 (d-hop independent set – (d-IS) *$D \subseteq V$ is a d-IS of G if $\forall v, u \in D$, $v \not\subseteq N_d[u]$.*

Definition 9 (Maximal d-hop independent set – MdIS): *A subset I_d is a MdIS of G if 1) I_d is a d-IS and 2) for any $v \in (V \setminus I_d)$, $I_d \cup \{v\}$ is not a d-IS anymore.*

Note that M*d*IS of *G* is *d*-DS of *G*. Now, we formally introduce the problems of our interest. The NP-hardness proofs of these problems are in Appendix.

Definition 10 (MdHSP): *Given a set V of nodes in the Euclidean space, the minimum d-hop sink placement (MdHSP) problem is to place a minimum number of sinks S in V such that each node in V is d-hop dominated by at least one sink in S.*

Definition 11 (CMdHSP): *Given a set V of nodes in the Euclidean space, the capacitated minimum d-hop sink placement (CMdHSP) problem is to find an optimal solution of MdHSP under an additional constraint that each sink can d-hop dominate at most k nodes.*

Assumptions: Now, we enumerate the major assumptions of this paper. First, this paper assumes that WSNs consist of a set of homogenous (i.e. the same hardware) sensor nodes with a set of powerful sinks. Second, we assume that the sinks can be located at any place. This can be possible if the sinks are flying machines such as miniature helicopters or the space on which the sensor nodes are deployed has no obstacles. Third, we assume the input graph is randomly generated and is connected.

If an input graph is disconnected, our algorithms will treat each of connected components as an independent graph. In fact, this assumption will not degrade the performance of our algorithms arbitrarily bad. That is, if an input graph consists of a set of connected components such that the distance between any two connected components is greater than two, then this assumption does not degrade the performance of the algorithms. If there are two components whose distance is no greater than two, the approximation factor of our algorithms under such assumption increases no greater than five times in UDG and 12 times in UBG. This is because in UDG, a node has at most five independent neighbourhoods (Lemma 7), and in UBG, a node has at most 12 independent neighbourhoods (Lemma 1).

3 Constant factor approximations of M*d*HSP and CM*d*HSP in UDG and UBG

In Section 3.1, we show a simple colouring algorithm (Algorithm 1) is a constant factor approximation algorithm for M*d*HSP in both UDG and UBG models. In Section 3.2, we use this result to propose a constant factor approximation algorithm (Algorithm 2) for CM*d*HSP in both UDG and UBG models.

Algorithm 1 *d*-hop Colouring ($G = (V,E)$)

1: Set $S \leftarrow \varnothing$
2: Colour all nodes in *V* white
3: **while** there is a white node $v \in V$ **do**
4: Colour *v* black and set $S \leftarrow S \cup \{v\}$
5: Colour every node in $N_d(v)$ grey
6: **end while**
7: Place a sink *s* very near to each node *u* in *S* so that all nodes in $N_d[u]$ is *d*-hop dominated by *s*.

3.1 MdHSPA: constant factor approximations of MdHSP

In this section, we prove a Maximal *d*-hop Independent Set (M*d*IS) computed by a simple *d*-hop colouring strategy (Algorithm 1) is an $O(d^2)$-approximation for M*d*HSP in UBG. We also show this is actually an $O(d)$-approximation for M*d*HSP in UDG.

Note that in both UBG and UDG, an M*d*IS is a *d*-hop Dominating Set (*d*-DS), which is a feasible solution of M*d*HSP. Therefore, we will focus on the worst case performance analysis of this strategy. In Section 3.1.1, we assume *G* is UBG and proceed our analysis. In Section 3.1.2, we modify this proof to analyse the performance ratio of Algorithm 1 for M*d*HSP in UDG.

3.1.1 Approximation Ratio of Algorithm 1 in UBG

In this section, we assume *G* is UBG. The outline of our performance analysis is as follow: let OPT_{MdHSP} and $OPT_{d\text{–}DS}$ be an optimal solution of M*d*HSP and an optimal *d*-DS, respectively. For each $v \in OPT_{d-DS}$, if we can bound the maximum cardinality (size) of $I_d(v)$, an M*d*IS of $N_d(v)$ by some constant α (Lemma 5) and if we can show that $|OPT_{d-DS}| \leq \beta |OPT_{MdHSP}|$ (Lemma 6), Algorithm 1 is an $\alpha \cdot \beta$-approximation of M*d*HSP in UBG.

For each $v \in OPT_{d-DS}$, let $K_v = \{x \mid x \in I_d(v)$ and $Hopdist(x,v) \leq \lceil d/2 \rceil\}$, and $K'_v = \{x \mid x \in I_d(v)$ and $x \notin K_v\}$. Clearly, α is the possible maximum of $|K_v| + |K'_v|$.

Lemma 1 (Butenko et al., 2010): *A unit ball of radius one can contain at most 12 independent points.*

Lemma 2: *Suppose I_d is an MdIS of G = (V,E). Then, for each $v \in V$, there are at most 12 nodes in I_d, which are $\lceil d/2 \rceil$-hop dominated by v. (which implies $|K_v| \le 12$.)*

Proof: Suppose $U = \{u_1, u_2, ..., u_k\}$ is the set of all nodes in I_d such that $Hopdist(v, u_i) \le \lceil d/2 \rceil$ for all *i*. Then, every possible situation is in one of following two distinct cases:

- *Case 1 – d is an even number*: if $k \le 1$, then this lemma is trivially true. Now, suppose $k > 1$ and there are two different $u_a, u_b \in U$. This implies $Hopdist(u_a, v) \le \lceil d/2 \rceil = d/2$, $Hopdist(v, u_b) \le \lceil d/2 \rceil = d/2$, and thus $Hopdist(u_a, u_b) \le 2 \cdot d/2$. However, this contradicts to our assumption that u_a and u_b are two different elements of *U*, and therefore more than *d*-hop far from each other. As a result, if d is even, $k \le 1$ must be true.
- *Case 2 – d is an odd number*: for each $u_i \in U$, consider the shortest path from u_i to *v*. Let w_i be the node in this path, adjacent to *v*. Then, all of $w_1, w_2, ..., w_k$ are in the ball with centre *v* and radius one.

For contradiction, suppose $k \ge 13$. By Lemma 1, a unit ball can contain at most 12 independent nodes. By the Pigeonhole principle, there have to be two different $u_a, u_b \in U$ such that $Hopdist(w_a, w_b) \le 1$ (either w_a or w_b are neighbouring with each other or $w_a = w_b$). By combining this with the facts that $Hopdist(u_a, w_a) \le \lceil d/2 \rceil - 1$ and $Hopdist(u_b, w_b) \le \lceil d/2 \rceil - 1$, we can conclude that $Hopdist(u_a, u_b) \le 2(\lceil d/2 \rceil - 1) + 1 = d$ since *d* is odd. However, this contradicts to the fact that u_a and u_b are two discrete members of *U*, and thus $k \le 12$ if *d* is odd.

By combining our analysis on Case 1 and Case 2 above, we can conclude that $k \le 12$ is true for any $d \ge 1$ and thus this lemma holds true.

Now, we find the upper bound of $|K'_v|$. Consider $u \in K'_v$. Let

- P_u be the set of nodes in the shortest path from *u* to *v* which includes *u* and *v*,
- $Q_u = \{x \mid x \in P_u \text{ and } Hopdist(x, u) \le \lceil d/2 \rceil - 1\}$,
- B_w be a unit ball whose centre is *w* and radius 1/2 for each $w \in Q_u$, and
- $A_u = \bigcup_{w \in Q_u} B_w$.

Now, we show some property of A_u.

Lemma 3: *All A_u for $u \in K'_v$ are disjoint in G.*

Proof: Suppose for some $u, u' \in K'_v$ with $u \ne u'$, $A_u \cap A_{u'} \ne \emptyset$. Then there exist $w \in Q_u$ and $w' \in Q_{u'}$ such that the distance between *w* and *w'* is at most one (i.e. $Hopdist(w, w') \le 1$). This implies that there is a path between *u* and *u'* whose length is at most *d*-hops since we also know $Hopdist(w, u) \le \lceil d/2 \rceil - 1$ and $Hopdist(w', u') \le \lceil d/2 \rceil - 1$, and thus $Hopdist(u, u') \le 2 \left(\lceil d/2 \rceil - 1\right) + 1 \le d$. This contradicts that *u* and *u'* are *d*-hop independent. Therefore, the lemma is true.

Lemma 4: *The volume of A_u is at least $(d-3)\pi/24$ in G.*

Proof: Let $Q_u = \{u, w_1, ..., w_k\}$, where $k = \lceil d/2 \rceil - 1$ and the elements in Q_u are in the ordering from *u* toward *v* in P_u. Clearly, B_u cannot intersect with $B_{w_2}, B_{w_3}, ..., B_{w_k}$, B_{w_2} cannot intersect with $B_{w_4}, B_{w_5}, ..., B_{w_k}$, and so on. It follows that $B_u, B_{w_2}, B_{w_4}, ..., B_k$ are disjoint if *k* is even (or $B_u, B_{w_2}, B_{w_4}, ..., B_{k-1}$ are disjoint if *k* is odd). Hence, the volume of A_u is at least

$$\left(\frac{\lceil d/2 \rceil - 2}{2} + 1\right) \cdot \frac{4\pi}{3 \cdot 2^3} > \frac{d-3}{4} \cdot \frac{\pi}{3 \cdot 2} = \frac{(d-3)\pi}{24}.$$

Lemma 5: *Suppose $OPT_{d\text{-}DS}$ is an optimal d-DS of G and S is an output of Algorithm 1 (i.e. an MdIS of G). Then $|S| \le O(d^2)|OPT_{d\text{-}DS}|$.*

Proof: For every $u \in K'_v$, A_u lies inside the ball with centre *v* and radius $d + 0.5$. From Lemma 3, each A_u is disjoint with $A_{u'}$ for any $u' \in K'_v$, $u' \ne u$. Therefore, $|K'_v|$ can be bounded using the volume of the big ball whose radius is $d + 0.5$ and the lower bound of the volume of each A_u, which is given in Lemma 4. In detail

$$|K'_v| \le \frac{(4/3)\pi(d+0.5)^3}{(d-3)\pi/24} = O(d^2).$$

By combining this with Lemma 2, we can conclude that

$$|S| \le (|K'_v| + |K_v|)|OPT_{d\text{-}DS}| = \left(O(d^2) + 12\right)|OPT_{d\text{-}DS}|.$$

Lemma 6: $|OPT_{d\text{-}DS}| \le 12|OPT_{MdHSP}|$.

Proof: Suppose $OPT_{MdHSP} = \{c_1, \cdots, c_s\}$ is an optimal solution of M*d*HSP in UBG, where $s = |OPT_{MdHSP}|$. By definition, OPT_{MdHSP} does *d*-hop dominate *G*. Denote by $MIS(N(c_i))$ an MIS of $N(c_i)$.

Now, we claim if a node $v \in V$ is *d*-hop dominated by c_i, it is also *d*-hop dominated by at least one node in $MIS(N(c_i))$. To show this claim is correct, let us define a shortest path from *v* to c_i as $P_v = \{v, n_1, n_2, \cdots, n_k, c_i\}$. Since *v* is *d*-hop dominated by c_i, $k \le d-1$ has to be true. Clearly,

n_k should be either in $MIS(N(c_i))$ or adjacent to a node w in $MIS(N(c_i))$. In the first case, v is $(d-1)$-hop dominated by n_k, a member of $MIS(N(c_i))$. In the second case, v is d-hop dominated by w, a member of $MIS(N(c_i))$. In either case, v is d-hop dominated by at least one member of $MIS(N(c_i))$, and thus this claim is true. From this claim, we can conclude that $Y = \bigcup_{i=1}^{s} MIS(N(c_i))$ is d-hop dominating G as $OPT_{MdHSP} = \{c_1, c_2, \cdots, c_s\}$ does.

From Lemma 1, each c_i can have at most 12 independent neighbours in UBG, and therefore, $|MIS(N(c_i))| \le 12$. Since it is possible $MIS(N(c_i)) \cap MIS(N(c_j)) \ne \varnothing$ for $i \ne j$ such that $1 \le i, j \le s$,

$$|Y| = |\bigcup_{i=1}^{s} MIS(N(c_i))| \le 12s = 12|OPT_{MdHSP}|. \quad (1)$$

Also, since Y is a d-DS of G, $|d - DS_{opt}| \le |Y|$. As a result,

$$|d - DS_{opt}| \le |Y| \le 12|OPT_{MdHSP}|,$$

and the lemma is true.

Theorem 1: *The approximation ratio of Algorithm 1 for MdHSP in UBG is $O(d^2)$.*

Proof: From Lemma 5, we have $|S| \le O(d^2)|d - DS_{opt}|$, where S is an output of Algorithm 1, which implies that an MIS of G is a constant factor approximation of the minimum d-hop dominating set problem in UBG. From Lemma 6, we have $|d - DS_{opt}| \le 12|OPT_{MdHSP}|$, which means that a minimum d-hop dominating set is a constant factor approximation of MdHSP in UBG. By combining these two lemma, we have $|S| \le O(12d^2)|OPT_{MdHSP}|$. Therefore, Algorithm 1 is an $O(d^2)$-approximation algorithm for MdHSP in UBG.

3.1.2 Approximation ratio of Algorithm 1 in UDG

In this section, we assume G is UDG. The outline of our performance analysis is similar to the one in the previous section. We reuse the definitions of OPT_{MdHSP}, $OPT_{d\text{-}DS}$, K_v, K'_v, P_u, Q_u, and A_u. However, we define B_w be a unit disk (instead of a unit ball) whose centre is w and radius 1/2.

Lemma 7 (Wan et al., 2002): *A unit disk of radius one can contain at most five independent points.*

Lemma 8: *Suppose I_d is an MdIS of $G = (V, E)$. Then, for each $v \in V$, there are at most five nodes in I_d, which are $\lceil d/2 \rceil$-hop dominated by v.*

Proof: The proof of this lemma is very similar to that of Lemma 2. Only difference is that since we assume G is UDG, we need to use Lemma 7 instead of Lemma 1. In detail, by applying Lemma 7 to the Case 2 analysis of Lemma 2, it is trivial to show that $|K_u| \le 5$ in UDG. (i.e. $|K_v| \le 5$).

Clearly, Lemma 3 is still true in UDG since its proof does not rely on the fact that G is UBG. However, Lemma 4 has to be modified properly.

Lemma 9: *The area of A_u is at least* $(d-3)\pi/16$ in G.

Proof: The basic structure of this proof is equivalent to the proof of Lemma 4. However, since each B_w is a unit disk whose radius is 0.5, and its centre is w, the area of A_u is at least

$$\left(\frac{\lceil d/2 \rceil - 2}{2} + 1\right) \cdot \frac{\pi}{2^2} > \frac{d-3}{4} \cdot \frac{\pi}{2^2} = \frac{(d-3)\pi}{16}.$$

Lemma 10: *Suppose $OPT_{d\text{-}DS}$ is an optimal d-DS of UDG G and S is an output of Algorithm 1 (i.e. an MdIS of G). Then* $|S| \le O(d)|OPT_{d-DS}|$.

Proof: For every $u \in K'_v$, A_u lies inside the disk with centre v and radius $d + 0.5$. From Lemma 3, each A_u is disjoint with $A_{u'}$ for any $u' \in K'_v$, $u' \ne u$. Therefore, $|K'_v|$ can be bounded using the area of the big disk whose radius is $d + 0.5$ and the lower bound of the area of each A_u, which is given in Lemma 9. In detail

$$|K'_v| \le \frac{\pi(d+0.5)^2}{(d-3)\pi/16} = O(d).$$

By combining this with Lemma 8, we can conclude that

$$|S| \le (|K'_v| + |K_v|)|OPT_{d-DS}| = (O(d) + 5)|OPT_{d-DS}|.$$

Lemma 11: $|OPT_{d-DS}| \le 5|OPT_{MdHSP}|$ *in UDG.*

Proof: In order to show this lemma is true, we can reuse most of our argument for the proof of Lemma 6. We just need to replace Lemma 1 with Lemma 7. In particular, equation (1) will be changed to

$$|Y| = \left|\bigcup_{i=1}^{s} MIS(N(c_i))\right| \le 5s = 5|OPT_{MdHSP}|,$$

and we can prove

$$|d - DS_{opt}| \le 5|OPT_{MdHSP}|.$$

Theorem 2: *The approximation ratio of Algorithm 1 for MdHSP in UDG is O(d).*

Proof: From Lemma 10, we have $|S| \le O(d)|d - DS_{opt}|$, where S is an output of Algorithm 1, which implies that an MIS of G is a constant factor approximation of the minimum d-hop dominating set problem in UDG. From Lemma 11, we have $|d - DS_{opt}| \le 5|OPT_{MdHSP}|$, which means that a minimum d-hop dominating set is a constant factor approximation of MdHSP in UDG. By combining these two lemmas, we have $|S| \le O(5d)|OPT_{MdHSP}|$. Therefore, Algorithm 1 is an $O(d)$-approximation algorithm for MdHSP in UDG.

3.2 CMdHSPA: constant factor approximations of CMdHSP

Algorithm 2 Capacitated minimum d-hop sink placement algorithm ($G = (V,E)$, k)

1: Execute Algorithm 1 to place a set of sinks $S = \{s_1,\ldots,s_l\}$ in G. At this point, each node is a member of its nearest sink (and being serviced by the sink). A tie can be broken arbitrarily.
2: **for** $i = 1$ to l **do**
3: $M \leftarrow M(s_i)$/* the members of the sink s_i*/
4: **while** $|M| > k$ **do**
5: Construct a shortest path tree T of M rooted at s_i. For each node $w \in T$, let T_w be the subtree of T, rooted at w.
6: Let u be the root of T.
7: **while** u has a child v with $|T_v| > k$ **do**
8: $u \leftarrow v$
9: **end while**
10: Let $v_1, v_2,\ldots, v_h$ be all children of u in the ordering $\left|T_{v_1}\right| \le \left|T_{v_2}\right| \le \cdots \le \left|T_{v_h}\right| \le k$. Also, let $M(u) = \bigcup_{j=1}^{h} T_{v_j}$
11: **while** $|M(u)| > k$ **do**
12: Find an i, $2 \le i \le h$ such that $\sum_{j=i}^{h}\left|T_{v_j}\right| \le k$, but $\sum_{j=i-1}^{h}\left|T_{v_j}\right| > k$.
13: Place a new sink s' very near to u so that all the nodes in $\bigcup_{j=i}^{h} T_{v_j}$ can be d-hop dominated by s'. Also, all the nodes in $T_{v_i}, T_{v_{i+1}}, \cdots, V_{v_h}$ will be the member of s'.
14: $M \leftarrow M \Big/ \left(\bigcup_{j=i}^{h} T_{v_j}\right)$ and $h \leftarrow i-1$.
15: **end while**
16: **if** $|M(u)| = k$ **then**
17: Place a new sink s' at u, make all nodes in $M(u)$ to be the members of s', and set $M \leftarrow M / M(u)$. Note that u is still in M.
18: **end if**
19: **end while**
20: **end for**

In this section, we propose approximation algorithms for the capacitated minimum d-hop sink placement (CMdHSP) problem, namely the Capacitated Minimum d-hop Sink Placement Algorithm (CMdSPA). The overall strategy of CMdHSPA is as follow.

1 Executes Algorithm 1 to assign a set S of sinks to the given set V of sensor nodes without considering the additional constraint of the CMdHSP problem that each sink can serve at most k sensor nodes.

2 Suppose each node is a member of its nearest sink (a tie can be broken arbitrarily). For each sink $s_i \in S$ and the set $M(s_i)$ of the members of s_i, additional sinks are placed if necessary so that no sink would d-hop dominate more than k nodes in $M(s_i)$.

The first step is well discussed in Section 3.1. Therefore, we will focus our discussion on the second step. The outline of this step is as follow. For each $s_i \in S$, if $|M(s_i)| \le k$, we are done. Otherwise, a shortest path tree T of $M(s_i) \cup \{s_i\}$ rooted at s_i is constructed. For each node w of T, let T_w denote the subtree of T, rooted at w and $|T_w|$ the number of nodes in T_w. Now, we find a node u of T such that u must satisfy the property: $|T_u| > k$ and for every child v of u, $|T_v| \le k$. Clearly, such u has to exist since we assumed $\left|T_{s_i}\right| > k$. Let $v_1, v_2, \ldots, v_h$ be all children of u in the ordering $\left|T_{v_1}\right| \le \left|T_{v_2}\right| \le \cdots \le \left|T_{v_h}\right| \le k$. Find an i, $2 \le i \le h$ such that $\left|T_{v_i}\right| + \left|T_{v_{i+1}}\right| + \cdots + \left|T_{v_h}\right| \le k$ but $\left|T_{v_{i-1}}\right| + \left|T_{v_i}\right| + \cdots + \left|T_{v_h}\right| > k$. Clearly, such i should exist due to the property of T_u. Note that

$$\frac{k}{2} \le \left|T_{v_i}\right| + \cdots + \left|T_{v_h}\right| \le k \tag{2}$$

has to be true since otherwise $\left|T_{v_{i-1}}\right| > \frac{k}{2}$ has to be true so that $\left|T_{v_{i-1}}\right| + \left|T_{v_i}\right| + \cdots + \left|T_{v_h}\right| > k$ is true. However, this implies $\left|T_{v_{i-1}}\right| > \left|T_{v_i}\right|$, which contradicts to our initial assumption.

Now, at the position of u (or very near to the position of u in practice), we add a new sink for all the nodes in $T_{v_i} \cup \cdots \cup T_{v_h}$ and make the nodes as the members of the new sink, but keep u outside of this subset of the members. Delete $T_{v_i}, \ldots, T_{v_h}$ from the tree T_u. We repeat this until the remaining part of T_u is smaller than or equal to k. At last, if the size of remaining part is exactly k, we can employ one more sink at u and assign all the remaining nodes as the member of u.

Once one or more sinks are assigned, we remove the nodes which are members of the new sinks from $M(s_i)$ and repeat the procedure above (find another u and assign additional sinks). In this way, $M(s_i)$ will be broken into several (zero to many) smaller subsets of nodes, each of whose size is at least $k/2$ [see equation (2)]. Also there can be at most one exceptional subset of nodes whose size is less than $k/2$. Algorithm 2 is the formal description of CMdSPA.

Note that after CMdSPA is executed, we may want to reorganise each $M(s_i)$ such that the average distance between (s_i) and its member can be reduced. While such optimisation technique can reduce average delay further down, it does not improve the performance of CMdSPA algorithmically.

Theorem 3: *Algorithm 2 can be implemented in $O(n^2)$ time where n is the number of nodes in input graph.*

Proof: Clearly, Line 1 of Algorithm 2, which is a simple colouring algorithm, takes $O(n^2)$ time. In Line 5, the algorithm constructs a shortest path tree rooted at s_i and this takes $O(\tilde{n})$ time where $\tilde{n} = |M(s_i)|$. Note that at the same time, each node can remember the number of ancestors. This makes the running time of Lines 7–9 to be $O(n)$. For Line 10, we need to use a sorting algorithm and this takes totally $O(\tilde{n}\log\tilde{n})$ time since each node involves sorting at most once. For the last while-loop (Lines 11–15), the total running time is $O(\tilde{n}^2)$ because when S is updated, $M(v)$, the set of descendants of every ancestor v of u will be updated. Therefore, Lines 2–20 takes $O(\tilde{n}^2)$ time. As a result, the total running time of Algorithm 2 is $O(n^2)$.

3.2.1 *Approximation ratio of CMdHSP in UBG*

Now, we analyse the performance ratio of Algorithm 2 for CM*d*HSP in UBG. Let us define the CM*d*HSP-R problem whose goal is equivalent to that of CM*d*HSP except that we are allowed to put each sink only on an existing node in V. Clearly, Algorithm 2 produces a feasible solution of CM*d*HSP-R since the algorithm picks a subset D of the nodes V and place a sink very near to each of the nodes. Let $OPT_{CMdHSP-R}$ and OPT_{CMdHSP} be an optimal solution of CM*d*HSP-R and an optimal solution of CM*d*HSP, respectively. Then, we first show that $|D| \le O(d^2) |OPT_{CMdHSP-R}|$, and use result to show that $|D| \le O(d^2) |OPT_{CMdHSP}|$.

Lemma 12: *In UBG, the approximation ratio of Algorithm 2 for CMdHSP-R is* $O(d^2)$.

Proof: Let $S = \{s_1, \cdots, s_l\}$ be the set of the sinks deployed by Line 1 of Algorithm 2. For each sink $s_i \in S$, let $M(s_i)$ be a subset of the members of (s_i). Now, we will bound the maximum number of subsets generated from $M(s_i)$ after Lines 2–20 of Algorithm 2 is applied. For this purpose, we will count the number p of the subsets whose size are less than $k/2$ first and the number q of subsets whose size are at least $k/2$ later.

1 After Algorithm 2 is applied, we will obtain *at most* one subset with size less than $k/2$ from $M(s_i)$. That is, if we have more than two of those, they have to be merged into one by the algorithm.

2 After Algorithm 2 is applied, the number q of subsets with size at least $k/2$ from $M(s_i)$ cannot exceed $2n/k$ since otherwise, the total number of nodes in $M(s_i)$ must exceed n.

By Lemma 5, we have $|S| \le O(d^2) |OPT_{d-DS}|$. Also, it is clear that $|OPT_{d-DS}| \le |OPT_{CMdHSP-R}|$. Therefore, $p \le |S| \le O(d^2) |OPT_{CMdHSP-R}|$. From the Case (b), we have $q \le 2n/k$. Furthermore, $n/k \le |OPT_{CMdHSP-R}|$ is trivially true. Therefore, we have $q \le 2|OPT_{CMdHSP-R}|$.

Therefore, the total number of subsets is at most $(p+q)$ $|OPT_{CMdHSP-R}| = (O(n^2)+2)|OPT_{CMdHSP-R}|$. Since each subset includes one sink, this lemma is true.

Theorem 4: *Given a UBG G and a constant k, Algorithm 2 produces an approximation solution with at most* $O(12(d^2+2))|OPT_{CMdHSP}|$ *sinks.*

Proof: To prove this theorem, we will modify the proof of Lemma 6. Suppose $OPT_{CMdHSP} = \{c_1, \cdots, c_s\}$, where $s = |OPT_{CMdHSP}|$ and $M(c_i)$ is the members of a sink (c_i). Then, $M(c_i)$ is d-hop dominated by $MIS(N(c_i))$ as we showed in the proof of Lemma 6.

We would like to emphasise that it is always possible for each node in $M(c_i)$ to be a member of a node in $MIS(N(c_i))$ such that the number of members assigned to a particular node in $MIS(N(c_i))$ does not exceed k. This is because

1 Each node in $M(c_i)$ should be d-hop dominated by at least one node in $MIS(N(c_i))$ as we showed in Lemma 6, and

2 For all (c_i), $|M(c_i)| \le k$ since (c_i) is a member of OPT_{CMdHSP}.

Therefore, $Y = \bigcup_{i=1}^{s} MIS(N(c_i))$ is a feasible solution of CM*d*HSP-R after such membership assignment, which implies $|OPT_{CMdHSP-R}| \le |Y|$. From Lemma 6, we also know $|MIS(N(c_i))| \le 12$ for all c_i, and thus we have $|Y| \le 12s = 12|OPT_{CMdHSP}|$. In conclusion, we have

$$|OPT_{CMdHSP-R}| \le |Y| \le 12|OPT_{CMdHSP}|.$$

By combining this result with Lemma 12, the number of sinks introduced by Algorithm 2 is at most

$$O(d^2+2)|OPT_{CMdHSP-R}| \le O(12(d^2+2))|OPT_{CMdHSP}|,$$

and this theorem holds.

Algorithm 3 M*d*HSPA-qUDG ($G = (V,E)$, d, P, p)

1: Let $w = \lfloor log_p P \rfloor$. /* maximum number of unreliable links permitted in a routing path */
2: From the δ-qUDG G, we induce $\tilde{G}$ as follows: (a) copy G to $\tilde{G}$, (b) for each $u, v \in V(\tilde{G})$, add an *unreliable link* (u, v) to $E(\tilde{G})$ if $\delta < Eucdist(u, v) \le 1$.
3: Execute **SatisfiabilityTest** ($\tilde{G}$, w), and keep all $M_{i,j}^m$s returned by the function.
4: Set $S \leftarrow \varnothing$.
5: Colour all nodes in V white.
6: **while** there is a white node $u \in V(\tilde{G})$ **do**
7: Colour u black and set $S \leftarrow S \cup \{u\}$.
8: Colour each white node $v \in V$ grey if $M_{i,j}^m \le d$.
9: **end while**
10: Place a sink s on (or nearby) each node u in S.

3.2.2 *Approximation Ratio of CMdHSP in UDG*

Now, we analyse the performance ratio of Algorithm 2 for CM*d*HSP in UDG.

Theorem 5: *Given a UDG G and a constant k, Algorithm 2 produces an approximation solution with at most* $O(5(d+2))|OPT_{CMdHSP}|$ *clusters.*

Proof: Suppose D is the set of sinks placed by Algorithm 2. We will show $|D| \le (O(d)+2)|OPT_{CMdHSP-R}|$ first and later show $|OPT_{CMdHSP-R}| \le 5|OPT_{CMdHSP}|$. Then, the correctness of this theorem naturally follows.

Now, we prove $|D| \le (O(d)+2)|OPT_{CMdHSP-R}|$ is true by modifying the proof of Lemma 12. For this purpose, we simply need to employ Lemma 10 instead of Lemma 5. As a result, we have $p \le O(d)|OPT_{CMdHSP-R}|$. Since $q \le 2|OPT_{CMdHSP-R}|$ is still true in UDG, we have $|D| \le (p+q)|OPT_{CMdHSP-R}| \le (O(d)+2)|OPT_{CMdHSP-R}|$.

Next, we show $|OPT_{CMdHSP-R}| \le 5|OPT_{CMdHSP}|$. This can be directly obtained from the proof of Theorem 4 if we employ Lemma 11 instead of Lemma 6. As a result this theorem is true.

4 Extensions to δ-quasi unit disk graph

In the previous section, we studied M*d*HSP and CM*d*HSP in UDG and UBG. However, due to the salient features of wireless communication such as signal collision and interference, UDG and UBG may not be accurate enough for some applications. To overcome this limitation, we study the M*d*HSP and CM*d*HSP in δ-qUDG and δ-qUBG, which are known to be more precise and realistic than UDG and UBG, and propose new heuristic algorithms for them based on our results in the previous section.

4.1 A heuristic for MdHSP in δ-qUDG

Let us first introduce a heuristic algorithm for M*d*HSP in δ-qUDG, namely M*d*HSPA-qUDG (Algorithm 4). Roughly speaking, this algorithm consists of two phases. The first phase (Lines 1–3) consists of several pre-processing steps for the second phase (Lines 4–10). Algorithm 4 uses a popular variation of δ-qUDG as a network abstraction, in which a routing path from a node to a sink can include both *reliable links* (whose length is at most δ) and *unreliable links* (whose length is between δ and 1, and each of which will be available with a probability *p*). Clearly, if the routing path includes more number of unreliable links, it is less likely that a message can be successfully delivered over the path.

In such environment, we claim that a wireless sensor network can sufficiently support a user's worst case node-to-sink latency requirement if there is a path from each node to a sink whose hop length is no greater than *d* and the probability that the communication over the path is successful is at least some probability *P* given by the user. Note that under the second condition, the number of unreliable links in a routing path should not exceed $w = \lfloor \log_p P \rfloor$.

Now, we describe the detail of Algorithm 4. In Line 1, the algorithm computes *w*. In Line 2, a new graph $\tilde{G}$ is copied from given a δ-qUDG *G* such that $V(\tilde{G}) \leftarrow V(G)$ and $E(\tilde{G}) \leftarrow E(G)$. In a δ-qUDG *G*, there is no link between two nodes *u*, *v* whose Euclidean distance is $\delta < Eucdist(u,v) \le 1$. However, in $\tilde{G}$, we add an edge between every such pair of nodes. We will call $E(\tilde{G}) \bigcap E(G)$ as *reliable links* and $E(\tilde{G}) / E(G)$ as *unreliable links*. In Line 3, for each pair of nodes, we check the length of shortest path with at most *w* unreliable links between every pair of nodes. We call the path between two nodes by a *satisfying path* if (a) the hop length of the path is at most *d* and (b) the path does not include not more than *w* unreliable links. In the following subsection, we will discuss about the detail of our algorithm (Algorithm 4) to compute the length of a shortest satisfying path between each pair of nodes in $\tilde{G}$.

Lines 4–9 is a colouring algorithm similar to Algorithm 1. The major difference is that while in Algorithm 1, when a node *u* is coloured black, all white nodes *v*'s such that there is a shortest hop path from *u* to *v* with length at most *d* becomes grey, Algorithm 3 colour a white node *v* grey if there is a path from *u* to *v* whose length is at most *d* and which does not include no more than *w* unreliable links. Finally, in Line 10, by putting a sink on each black node, we have a feasible solution of M*d*HSP in δ-qUDG.

4.2 SatisfiabilityTest: testing the existence of satisfiable path between nodes

Algorithm 4 that we introduced in the previous section relies on an algorithm to compute the length of the shortest path between two nodes with at most *w* unreliable paths. In this section, we implement this algorithm as a dynamic programming (Algorithm 4).

Algorithm 4 SatisfiabilityTest (*G,w*)

1: Suppose $n = V(\tilde{G})$. Let $M^m_{i,j}$ be the length of the shortest path from *i*-th node to *j*-th node with no more than *m* unreliable edges. Initialise each $M^m_{i,j}$ with ∞.
2: **for** each $1 \le i, j \le n$ pair **do**
3: $M^0_{i,j} \leftarrow 1$ if $(i,j) \in E(\tilde{G})$ and is reliable, i.e. $(i,j) \in G$.
4: **end for**
5: Optimise $M^0_{i,j}$ s using an all-pair shortest path algorithm in $\tilde{G}$ excluding any unreliable link, i.e. in *G*.
6: **for** each $1 \le i, j \le n$ pair **do**
7: $M^1_{i,j} \leftarrow 1$ if $(i,j) \in E(\tilde{G})$.
8: **end for**
9: **for** each $1 \le m \le w$ **do**
10: **for** each $1 \le l \le n-1$ **do**
11: **for** each $1 \le i, j \le n$ pair **do**
12: $M^m_{i,j} = \min\{M^{m-1}_{i,j}, \min_{1\le k\le n} f(i,j,k,m)\}$, where

$$f(i,j,k,m) \leftarrow \begin{cases} M^{m-1}_{i,k}+1 & \text{if}(k,j) \in E(\tilde{G}) \text{ and is unreliable.} \\ M^{m}_{i,k}+1 & \text{if}(k,j) \in E(\tilde{G}) \text{ and is reliable.} \\ \text{Ignore} & \text{if}(k,j) \notin E(\tilde{G}). \end{cases}$$

13: **end for**
14: **end for**
15: **end for**
16: Return all $M^m_{i,j}$ s.

The detail of this algorithm is as follows. In this algorithm, $M_{i,j}^{m}$ is the length of the shortest path from *i*-th node to *j*-th node with no more than *m* unreliable edges. In Line 1, we initialise each $M_{i,j}^{m}$ with ∞. In Lines 2–5, the hop distance of every pair of nodes v_i, v_j in *G* (which does not include any unreliable link) is computed and stored in $M_{i,j}^{0}$. Lines 6–15 is a variation of an all-pair shortest path computation algorithm. The main difference is that we compute the length of the shortest hop path with at most *m* unreliable links between every pair of nodes such that *m* gradually increases from 1 to *w*. At Line 16, the algorithm returns all $M_{i,j}^{m}$ s. Note that if $M_{i,j}^{w} \le d$, then there is a satisfying path from v_i to v_j.

4.3 Extension of CMdHSPA for δ-qUDG

This can be done in a very straightforward way by changing two parts of Algorithm 2. First, Line 1 of Algorithm 2 should use Algorithm 4 instead of Algorithm 1 to have a feasible solution of M*d*HSP in δ-qUDG. Second, the shortest path tree *T* in Line 5 of Algorithm 2 should be constructed only using the subset of shortest path identified during the computation of $M_{i,j}^{m}$ s, where s_i is the sink and $v_j \in M(s_i)$ is the member of sensor nodes being serviced by s_i.

4.4 Remark on δ-qUBG extensions

The extensions of M*d*HSPA and CM*d*HSPA for δ-qUDG do not use any dimensional property. That is, M*d*HSPA is based on a greedy colouring strategy. CM*d*HSPA is an extension of M*d*HSPA with a tree partitioning strategy Therefore, those extensions still work in the 3-D counterpart of δ-qUDG, namely δ-qUBG.

5 Simulation results and discussions

In this section, we study the average behaviour characteristics of our strategy via simulation. In particular, we will perform this study using Algorithm 1 in the physical interference model. We use the parameters for the physical model from Shi et al.'s (2011) study. Note that for the ease of exposition, we also normalised all units for bandwidth, distance, rate and power with appropriate dimensions as was done in the work of Shi et al. (2011). We set the path-loss exponent, α, to 4 and the SINR threshold, β, to 3. We also ignore the ambient noise, which is very tiny value. We assume each node can transmit a signal at most 20 unit distances. Then, the maximum transmission power of each node is $P_{\max} = \alpha\ (20^4)$. We also assume that each node is using its maximum transmission power for communication.

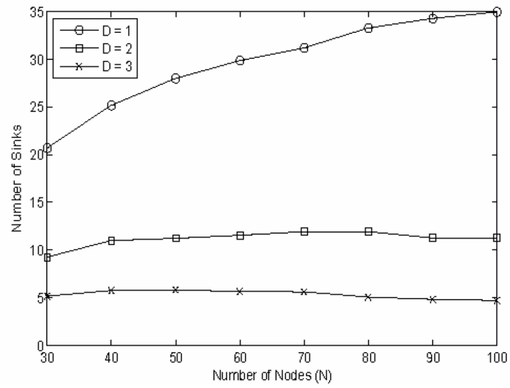
Under the parameter setting, we prepare a 60 × 60× 60 3-dimensional space and randomly generate *N* nodes, which varies to 30, 40, …, 100. Once *N* nodes are deployed, we check whether a graph induced by the nodes is connected with the maximum transmission power of each node if each communication link is not affected by any interference. If the graph is disconnected, we discard the graph and produce a new graph so that we can have a connected network. We set the maximum hop distance *D* from a sink to its members to 1, 2 and 3. Per each parameter setting, we produce 100 graphs and average the results.

In this simulation, we introduce the following strategy to optimise the performance of M*d*HSPA and use it. That is, once M*d*HSPA is applied and a graph is partitioned such that each partition has a single sink, we relocate the sink very near to another node in the partition such that the average hop distance from a node in the partition to the sink is minimised while the maximum hop distance from a node in the partition to the sink does not exceed *D*.

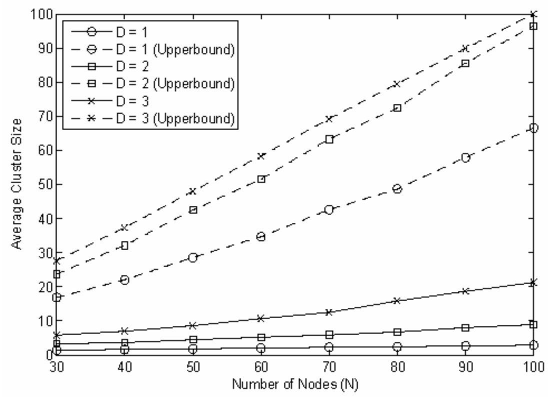
Figure 1 is the result of our simulation. Note that in this figure, *N* represents the number of nodes. The methodology of this simulation is as follows. Given a graph instance, we apply M*d*HSPA and install a set of sinks. Next, each sink is relocated as described above for optimisation. Then, we set the probability of each node to generate a node to 0.5%. For each unit time slot, which last for 100 unit time, following series of events will happen. First, each node generates a message using the probability (0.5%), and the node forwards it to the next hop in the shortest hop path to its sink. If the node also has a message to forward since it received the message from another node, the messages can be merged and transmitted altogether. Apparently, there can be more than one node which tries to transmit within one time slot, and therefore there can be some interfere among the concurrent transmissions. In such case, we simply drop all of those interfered messages. In reality, such problem can be solved by using appropriate scheduling algorithm, but will cause some extra delay. However, for our purpose, it is sufficient to count the number of such lost messages to see the severity of signal interference in our problem model since the actual worse case delay will increase proportional to drop ratio.

Before the discussion of our result, let us explain how dense the networks that we are working on by using following simple argument. In this simulation, we prepare a 60 × 60× 60 and deploy *N* wireless sensor nodes whose maximum communication radius is 20. Therefore, theoretically, we can place at most $60^3 \Big/ \frac{4}{3} \times \pi \times 20^3 = 6.445$ sensor nodes in the 3-D space such that the communication ranges of any two nodes do not overlap with each other. Therefore, when *N* = 30, the network is already quite dense in this space.

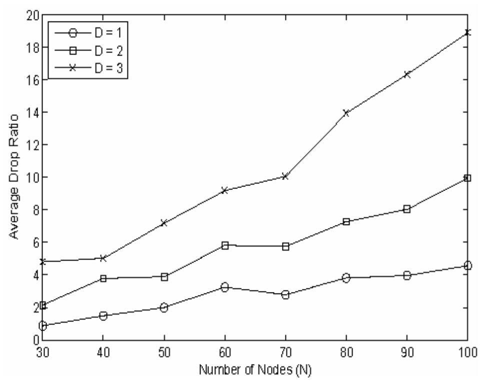
Figure 1 Performance of Algorithm 1 with $D = 1$, 2, and 3



(a) The average number of sinks deployed.



(b) The average number of nodes serviced by a sink.



(c) The average packet drop rate.

From Figure 1a, we can learn that (a) as the number of nodes, N, increases, the algorithm places more sinks, and (b) as the maximum allowable hop distance from a node to its nearest sink, represented by D, increases, the algorithm places less sinks. Figure1b shows the average number of nodes dominated by a sink by our algorithm against the upper bound, which is the average of the largest number of nodes which d-hop dominates a single node in the graphs tested per each parameter setting. As we can see from Figure1b, the size of each cluster increases as either N increases (since as network density increases, one sink can dominate more number of nodes) and D increases (since with a larger D, a sink can dominate more nodes). Beside from the fact that our bound is not tight, the simulation results show there are still a room to obtain a better result by making an extra effort, which is reserved as our future work. In general, the drop ratio increases as the size of each cluster increases. For example, the average cluster size with $D = 2$, $N = 60$ is 5.203 while that with $D = 3$, $N = 30$ is 5.825. In each case, the drop ratios are 5.8124% and 4.8070%, respectively. This means that D has a significant impact on the delay if the interference is considered. This also implies that CM*d*HSPA will work more efficiently than M*d*HSPA in busy network since it can suppress the drop ratio by limiting the number of nodes which can potentially send a message to the sink. Overall, our simulation result shows that hop distance is a good metric to represent the worst case latency of WSN if the network density (which is proportional to N) is low and the probability that each node produces a message in each unit time slot is low. However, if we want to deploy our algorithm in a large scale dense WSN with large D value, we have to further investigate a good scheduling strategy for our problem. We believe this is quite difficult to be done and thus reserve it as a part of our future work.

6 Related work

6.1 Related work in network community

Recently, several efforts are made to bound the node-to-sink data latency in WSNs (Sohrabi et al., 2000; Caccamo et al., 2002; Lu et al., 2002; Akkaya and Younis, 2003; He et al., 2003; Felemban et al., 2005). One of the earliest efforts to provide a bounded node-to-sink data latency guarantee for WSN was made by Sohrabi et al. (2000) who proposed the Sequential Assignment Routing (SAR) algorithm. This algorithm assigns the same priority to all the packets in the same flow. Each intermediate router forwards packets based on their priority. To find a routing path which can support a given worst case node-to-sink data latency requirement for a specific flow, multiple paths from the originator of the flow to its receiver are calculated and the one satisfying the requirement is used. Akkaya and Younis (2003) proposed a QoS protocol in which low priority is given to best-effort packets and high priority is given to real-time packets. Each intermediate router uses a scheduler and spends most of its resources for the real-time traffic, but also makes sure of assigning some resources for the best-effort traffic. As a result, packets with lower priority are not starved.

He et al. (2003) proposed the Stateless Protocol for Real-Time Communication (SPEED) algorithm for WSN. This algorithm also uses a per-flow based priority assignment strategy like SAR. However, instead of looking for a path satisfying a given worst case node-to-sink data latency requirement among multiple paths between a sender and a receiver, it uses a geographic forwarding strategy, which does not require to find a routing path before sending packets. One interesting feature of this algorithm is that the algorithm does not admit a packet whose required maximum tolerable latency is smaller than what the system can possibly support. Felemban et al. (2005) introduced the multi-path multi-SPEED (MMSPEED) routing protocol, which considers not only the maximum node-to-sink delay guarantee issue, but also the reliable transfer in WSN. The first feature for guaranteed maximum node-to-sink delay is similar to that of SPEED, in the sense that it is using a geographic forwarding. However, in MMSPEED, each packet may have a different priority, which can be changed at any intermediate router based on its deadline and remaining geographical distance between the router and the receiver of the packet like the real-time communication architecture for large-scale wireless sensor networks (RAP) by Lu et al. (2002). For the second feature for reliable transfer, multiple nodes and edge disjoint paths which can satisfy a worst case node-to-sink data latency are selected and the multiple copies of the packet are forwarded through the multiple disjoint paths.

The implicit Earliest Deadline First (EDF) packet scheduling algorithm by Caccamo et al. (2002) is a decentralised variation of the traditional EDF scheduling algorithm. If most traffic is periodic and all periods are known in advance, the implicit EDF can find the schedule which meets the real-time constraint of each packet as long as it does exist.

6.2 Related work in algorithm community

To the best of our knowledge, there are only few works which used multiple sinks to provide a worst case node-to-sink data latency in WSN. In the work of Kim et al. (2011), we studied the *k*-Sink Placement Problem (*k*-SPP), whose goal is to place *k*-sinks such that the maximum hop distance between a node and its nearest sink is minimum. That is, the goal of *k*-SPP is to best-utilise limited number of sinks so that the worst case node-to-sink delay is minimised. In opposition, in M*d*HSP studied in this paper, we have a maximum tolerable node-to-sink delay requirement, which cannot be negotiated unlike *k*-SPP, and we would like to minimise the number of sinks to meet the requirement so that our solution is the most cost-efficient. Algorithmically, they are completely different problems, since *k*-SPP is proven to be APX-complete and does not allow any approximation better than 2 (Kim et al., 2011), and M*d*HSP exhibits a structure to admit a PTAS and allow $(1 + \varepsilon)$-approximation for a small constant ε (Wang et al., 2009a).

M*d*HSP resembles the minimum d-hop dominating set (d-DS) problem in a sense that both of them are looking for a subset S of nodes which d-hop dominate a given graph $G = (V, E)$. However, there is a big difference between them: in the minimum d-DS problem, the $S \subseteq V$, but in M*d*HSP, the nodes in S can be located in any locations. In the work of Kim et al. (2011), we showed that we can compute a set of positions S' with size $O\left(|V|^2\right)$ which includes an optimal solution of M*d*HSP. Even with this technique, those problems are still algorithmically different.

As we will in this paper, an approximation for the d-hop dominating set problem is a feasible solution for M*d*HSP since (a) in UDG, this approach will require at least five times larger S than an optimal solution by Lemma 7 and in UBG, it is 12 times larger than an optimal solution by Lemma 1, and (b) the worst case performance analysis of this approach is not trivial.

7 Conclusion and future works

In this paper, we introduced two new problems, M*d*HSP and CM*d*HSP and proved their NP-hardness. We showed a simple colouring algorithm to compute a d-hop dominating set is in fact an $O(d)$-approximation for M*d*HSP in UDG, and an $O(d^2)$-approximation in UBG. Based on this result, we proposed another algorithm for CM*d*HSP and proved its performance ratio is $O(d)$ in UDG and $O(d^2)$ in UBG. Moreover, we extended those algorithms so that they can work in δ-qUDG. We believe that our approach can help to reduce the worst case data collection latency of a WSN in conjunction with the other existing latency control mechanisms harmoniously. As a future work, we plan to study a variation of our M*d*HSP and CM*d*HSP in which sinks cannot be located on arbitrary place, but on the subset of designated places. We are also interested in studying the problems in more realistic abstraction of WSN such as physical interference model (i.e. Signal-to-Inference and Noise Ratio (SINR) model).

Acknowledgements

This work was supported in part by NSF CREST No. HRD-0833184. This work was supported in part by US ARO No. W911NF-0810510. The research was supported in part by NSFC-11071191. This work was supported in part by NSF CNS-0831579, CNS-1016320 and CCF-0829993. This work was jointly supported in part by NSFC-61070191 and 91124001. This work was supported by the National Basic Research Program of China Grant 2011CBA00300, 2011CBA00302, NSFC-61033001, 61061130540 and 61073174. This research was jointly sponsored by MEST, Korea under WCU (R33-2008-000-10044-0), MEST, Korea under Basic Science Research Program (2011-0012216), MKE/KEIT, Korea under the IT R&D program [KI001810041244, SmartTV 2.0 Software Platform], and MKE, Korea under ITRC NIPA-2012-(H0301-12-3001).

References

Akkaya, K. and Younis, M. (2003) ‘An energy-aware QoS routing protocol for wireless sensor network’, *Proceedings of Workshops in the 23rd International Conference on Distributed Computing Systems*, pp.710–715.

Akyildiz, I.F. and Kasimoglu, I.H. (2004) ‘Wireless sensor and actor networks: research challenges’, *Ad Hoc Networks*, Vol. 2, No. 4, pp.351–367.

Akyildiz, I.F. and Su, W., Sankarasubramaniam, Y. and Cayirci, E. (2002) ‘A survey on sensor networks’, *IEEE Communications Magazine*, Vol. 40, pp.102–114.

Butenko, S., Kahruman-Anderoglu, S. and Ursulenko, O. (2010) ‘On minimum connected dominating set problem in unit ball graphs’, *Optimization Letters*, Vol. 5, No. 2, pp.195–205.

Caccamo, M., Zhang, L., Sha, L. and Buttazzo, G. (2002) ‘An implicit prioritized access protocol for wireless sensor networks’, *Proceedings of IEEE Real-Time Systems Symposium (RTSS 02)*, pp.39–48.

Cheng, X., Du, D-Z., Wang, L. and Xu, B. (2008) 'Relay sensor placement in wireless sensor networks', *Wireless Networks*, Vol. 14, No. 3, pp.347–355.

Felemban, E., Lee, C-G., Ekici, E., Boder, R. and Vural, S. (2005) 'Probabilistic QoS guarantee in reliability and timeliness domains in wireless sensor networks', *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pp.2646–2657.

He, T., Stankovic, J., Lu, C. and Abdelzaher, T. (2003) 'SPEED: a stateless protocol for real-time communication in sensor networks', *Proceedings of the 23rd International Conference on Distributed Computing Systems*, pp.46–55.

Johnson, D.S. (1974) 'Approximation algorithms for combinatorial problems', *Journal of Computer System Science*, Vol. 9, pp.256–278.

Kim, D., Wang, W., Ding, L., Lim, J., Oh, H. and Wu, W. (2010) 'Minimum average routing path clustering problem in multi-hop 2-d underwater sensor networks', *Optimization Letters (OPTL)*, Vol. 4, No. 3, pp.383–392.

Kim, D., Wang, W., Sohaee, N., Ma, C., Wu, W., Lee, W. and Du, D-Z. (2011) 'Minimum data latency bound *k*-sinks placement problem in wireless sensor networks', *IEEE/ACM Transactions on Networking (TON)*, Vol. 19, No. 5, pp.1344–1353.

Kuhn, F., Wattenhofer, R. and Zollinger, A. (2003) 'Ad-hoc networks beyond unit disk graphs', *Proceedings of the Joint Workshop on Foundations of Mobile Computing (DIALM-POMC)*, pp.69–78.

Lichtenstein, D. (1982) 'Planar formulae and their uses', *SIAM Journal on Computing*, Vol. 11, pp.329–343.

Lloyd, E.L. and Xue,G. (2007) 'Relay node placement in wireless sensor networks', *IEEE Transactions on Computers*, Vol. 56, No. 1, pp.134–138.

Lu, C., Blum, B.M., Abdelzaher, T.F., Stankovic, J.A. and He, T. (2002) 'RAP: a real-time communication architecture for large-scale wireless sensor networks', *Proceedings of IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS 2002)*, September, pp.55–66.

Misra, S., Hong, S.D., Xue, G. and Tang, J. (2008) 'Constrained relay node placement in wireless sensor networks to meet connectivity and survivability requirements', *Proceedings of the 27st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pp.879–887.

Misra, S., Reisslein, M. and Xue, G. (2008) 'A survey of multimedia streaming in wireless sensor networks', *IEEE Communications Survey & Tutorials*, Vol. 10, No. 4, pp.898–903.

Nguyen, T.N. and Huynh, D.T. (2006) 'Connected d-hop dominating sets in mobile ad hoc networks', *Proceedings of the 4th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, 3–6 April.

Rezgui, A. and Eltoweissy, M. (2007) 'Service-oriented sensor-actuator networks: promises, challenges, and the road ahead', *Computer Communications*, Vol. 30, pp.2627–2648.

Shi, Y., Hou, Y.T., Kompella, S. and Sherali, H.D. (2011) 'Maximizing capacity in multihop cognitive radio networks under the SINR model', *IEEE Transactions on Mobile Computing (TMC)*, Vol. 10, No. 7, pp.954–967.

Sohrabi, K., Gao, J., Allawadhi, V. and Pottie, G. (2000) 'Protocols for self-organization of a wireless sensor network', *IEEE Personal Communications*, Vol. 7, No. 5, pp.16–27.

Valiant, L. (1981) 'Universality considerations in VLSI circuits', *IEEE Transactions on Computers*, Vol. C-30, pp.135–140.

Wan, P-J., Alzoubi, K.M. and Frieder, O. (2002) 'Distributed construction of connected dominating set in wireless ad hoc networks', *Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, New York, USA.

Wang, W., Kim, D., Sohaee, N., Ma, C. and Wu, W. (2009a) 'A PTAS for minimum *d*-Hop underwater sink placement problem in 2-d underwater sensor networks', *Discrete Mathematics, Algorithms and Applications (DMAA)*, Vol. 1, No. 2, pp.283–289.

Wang, W., Kim, D., Willson, J., Thuraisingham, B. and Wu, W. (2009b) 'A better approximation for minimum average routing path clustering problem in 2d-underwater sensor networks', *Discrete Mathematics, Algorithms and Applications (DMAA)*, Vol. 1, No. 2, pp.175–191.

Youssef, W. and Younis, M. (2007) 'Intelligent gateways placement for reduced data latency in wireless sensor networks', *Proceedings of IEEE International Conference on Communications (ICC)*, Glasgow, Scotland.

Zhang, W., Xue, G. and Misra, S. (2007) 'Fault-tolerant relay node placement in wireless sensor networks: problems and algorithms', *Proceedings of the 26th IEEE International Conference on Computer Communications (INFOCOM 2007)*, Anchorage, AK, pp.1649–1657.

Note

1 This is generally known as *d*-qUDG. In this paper, we will use δ instead of d for our notational convincible.

Appendix

Theorem A1: *MdHSP and CMdHSP are NP-hard in UDG.*

Proof: Throughout the rest of this section, we prove the decision version of M*d*HSP is NP-complete in UDG, and therefore M*d*HSP is NP-hard in UDG. Then, *CMdHSP* is also NP-hard in UDG since a subclass of CM*d*HSP, where $d \geq |G(V)|$ is equal to M*d*HSP in UDG.

Corollary A1: *MdHSP and CMdHSP are NP-hard in δ-qUDG.*

Proof: It follows immediately from the fact that UDG is a special case of δ-qUDG.

The decision version of M*d*HSP with $d = 1$ is the same as UDCP, which is proved to be NP-complete (Johnson, 1974). However, this does not necessarily mean the decision version of M*d*HSP is NP-complete. For instance, it was necessary to show the minimum connected *d*-hop dominating set problem is NP-hard even though the minimum (1-hop) dominating set problem is a well-known NP-hard problem (Nguyen and Huynh, 2006). In the rest of section, M*d*HSP means the decision version of M*d*HSP. It is easy to see that M*d*HSP is in the class-NP. In this section, we assume $d \geq 2$ and prove that M*d*HSP is still NP-complete. Formally, given a Boolean expression $C = C_1 \wedge C_2 \wedge \cdots \wedge C_m$, where $C_i = x_i \vee y_i \vee z_i, (\{x_i, y_i, z_i\} \subseteq V \bigcup \overline{V}$, $V = \{v_1, \ v_2, \cdots, v_n\}$, $V = \{\overline{v}_1, \overline{v}_2, \cdots, \overline{v}_n\}$), the 3-SATisfiability (3-SAT) problem is to determine if there is a set $S \subseteq \{v_1, \overline{v}_1, v_2, \overline{v}_2, \cdots, v_n, \overline{v}_n\}$ such that $S \bigcap \{x_i, y_i, z_i\} \neq \varnothing$, $forall 1 \leq i \leq m$ and $|S \bigcap \{v_j, \overline{v}_j\}| = 1$, $forall 1 \leq j \leq n$.

To show the NP-completeness of our problem, we reduce from the PLANAR 3-SAT problem, which is strongly NP-complete (Lichtenstein, 1982). In an instance of PLANAR 3-SAT, we are given a planar bipartite graph whose nodes on one class of the bipartition represent the variables u_1, …, u_n, and whose nodes on the other class represent the clauses, C_1, C_2,…,C_m, and edges connect each clauses to the three variables it contains. Moreover, there are edges like $(u_1, u_2), (u_2, u_3), \ldots, (u_{n-1}, u_n), (u_n, u_1)$. Later we will show that from any PLANAR 3-SAT instance, we can make an equivalent formula such that each variable appears at most three times.

Here is the general idea of our reduction. For a PLANAR 3-CNF $C = C_1 \wedge C_2 \wedge \cdots \wedge C_m$, we construct an UDG G_c on a grid space such that a satisfiable assignment for C implies an optimal solution T of M*d*HSP on G_c and vice versa. In M*d*HSP, each sink can be placed at any point on the plane. We will construct G_c carefully so that there is an optimal solution T such that $T \subseteq V(G_C)$. Precisely, in our construction, the maximum degree of G_c is at most four. By Valiant (1981), a Planar graph with maximum degree four always can be embedded in a grid. Therefore, cases as shown in Figure A2(a)–A2(c) will never happen. Figure A3(d)–A3(g) illustrate that a sink can move to the location of one of the nodes of the graph while it is *d*-hop dominating the same set of nodes of G_c. In our construction of G_c, a sink normally 1-hop dominates at most three nodes. Only exception is the case (*g*) in Figure A3, where a sink is placed on one of existing sensor node in G_c. In this way, we can show the NP-completeness of our problem by showing that finding an optimal *d*-DS on G_c is NP-complete. In the following sections, we give some details of our constructions. We always assume that all edges in the UDG have unit length, which is one.

Figure A1 This planar graph is induced from a 3-SAT instance, $C = \left(v_2 \vee \overline{v_3} \vee v_4\right) \wedge \left(\overline{v_1} \vee v_2 \vee \overline{v_4}\right) \wedge \left(v_1 \vee v_2 \vee \overline{v_3}\right)$. Therefore, C is a PLANAR 3-SAT instance

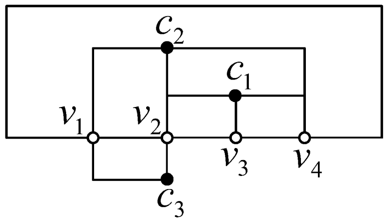
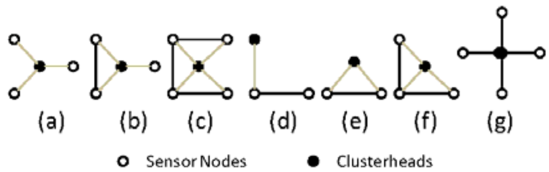
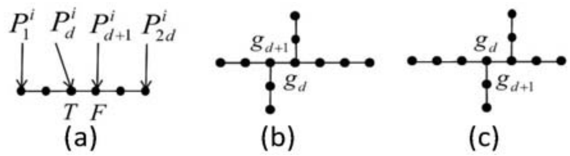


Figure A2 (a),(b), and (c) are prohibited in our construction. When a sink is located and dominates its neighbours as like (d), (e), and (f), we can move the sink on one of existing nodes 1-hop dominated by the sink such that the same nodes are still 1-hop dominated by the sink. (g) is only case that a sink 1-hop dominates four nodes. Other than this case, a sink 1-hop dominates at most three nodes



A1. Constructing a graph representing variables: In the reduction, both variables v_i and $\overline{v_i}$ are represented by the same $P^i = (P_1^i, P_2^i \cdots P_{2d}^i)$ of length $2d - 1$ [Figure A3(a)] for $i = 1, 2, \cdots, n$. Clearly, we can *d*-hop dominate the vertices in path P^i using only one sink. There are exactly two ways to place the sink. That is, we can put it at either P_d^i or P_{d+1}^i, and call them a true point and false point, respectively. In our truth assignment, if we assign a true value to v_i, the sink can be placed on the true point in G_c. Otherwise, on false point.

Figure A3 Figure (a), (b), and (c) are a variable v_i, Type *A*, and Type *B* gadgets, respectively, when $d = 3$



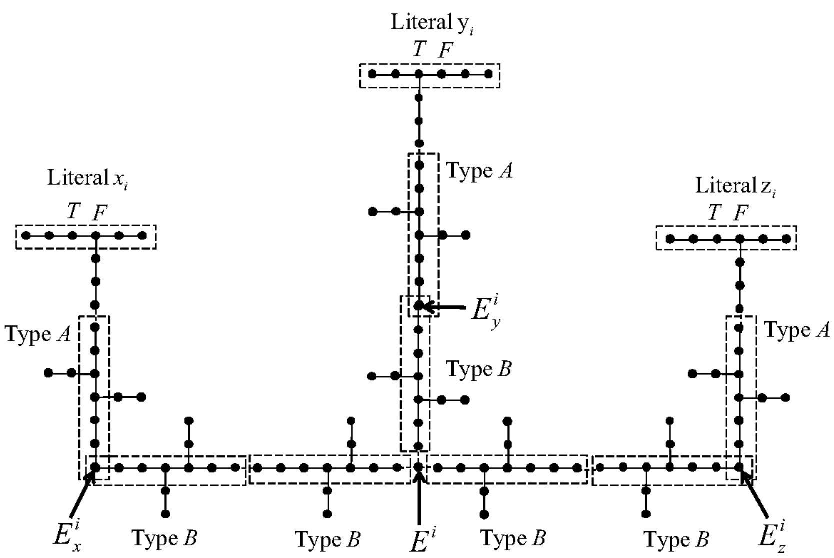
A2. Structure of two gadgets for G_c: Gadgets are important building blocks in our construction. They can be obtained from a path of length 2*d* by adding two pendent paths of length *d*–1 at the *d*-th and the (*d*+1)-th vertices, respectively. Both of these pendent paths are perpendicular to the original path and on the opposite sides. We will call *i*-th vertex in the path g_i. Based on definition, we will have two types of gadgets, Type *A* and Type *B*, as it illustrated in Figure A3(b) and A3(c). Basically they are the same UDGs, but are used in different ways. There are two special points in a gadget, namely, g_d and g_{d+1}, which play a key role in our arguments. We want to emphasise that *a gadget can be d-hop dominated by exactly one point on it – g_d. However, it cannot be d-hop dominated by two any other sinks placed on the nodes in other gadgets, if we connect more gadgets together to form a chain (as we will do later).* This means that to *d*-hop dominate a gadget in a graph, we need to place at least one sink in each gadget. Furthermore, because of its special structure, if we place the sink on any other vertex except g_d and g_{d+1} in a given gadget, one of the endpoints of pendent paths may not be *d*-hop dominated simultaneously. Note that both endpoints of the two pendent paths cannot be dominated by sinks lying outside the given gadget. Therefore, if we do not want to put a sink on g_d or g_{d+1}, we need at least two sink on one gadget to *d*-hop dominate. Thus, as we shall see later, when we try to minimise the total number of sinks placed in our constructed graph G_c or G_{C_i}, *we have to put sinks at either g_d or g_{d+1} in each gadget.*

A3. Constructing a graph representing a clause: Let $C_i = x_i \vee y_i \vee z_i$ be one clause in a given Boolean expression of PLANAR 3-SAT *C*. C_i is represented as an UDG G_{C_i}, as shown in Figure A4. There are four special vertices in G_{C_i}, denoted by E_x^i, E_y^i, E_z^i and E^i, corresponding to three literals x_i, y_i, z_i and clause C_i, respectively. E^i is connected to literal x_i through a chain of gadgets and a path of length $d - 1$. The endpoint of this path is connected (by adding an edge) to true or false point (depending on whether $x_i = v^j$ or $\overline{v}_j$) of the path P^i representing x_i, while the other endpoint of the path is connected to an endpoint of a gadget. Similarly, we can connect E^i and y_i, E^i and z_i. The number of gadgets does not affect the structure of the graph. So, we use enough gadgets to make the graph planar on grid. E^i and x_i (resp. y_i and z_i) are connected with a chain of Type *A* gadgets and Type *B* gadgets and there is an overlap point E_x^i (resp. E_y^i and E_z^i) on this path. (This is a key point to our construction). E^i does not belong to any gadget. Denote by N_{C_i} the number of gadgets in G_{C_i}. By $x_i = True$ (resp. $x_i = False$) we mean that the sink is placed at the truth point (resp. the false point) of x_i. Now we introduce an important property of the above construction.

Lemma A1: *G_{C_i} has an optimal solution of size $N_{C_i} + 3$ for MdHSP if and only if C_i is true.*

Proof: First we show if C_i is true, then $N_{C_i} + 3$ sinks are enough to *d*-hop dominate G_{C_i}, with the special point E^i being *d*-hop dominated by some sink on a gadget adjacent to E^i. In fact, if E_i is true, one of x_i, y_i and z_i has to be true. Without loss of generality, we call this (the one with true assignment) x_i. Now, put one sink as follows: true point of literal x_i, true (or false) point of y_i, true (or false) point of z_i; g_d points of each gadget of Type *A* (resp. *B*) connecting E_x^i and x_i (resp. connecting E^i and E_y^i, connecting E^i and E_z^i); g_{d+1} points of each gadget of type *B* (resp. *A*) connecting E_x^i and E^i (resp. connecting E_y^i and y_i, connecting E_z^i and z_i). Then it can easily be verified that E^i is *d*-hop dominated by the sink on the g_{d+1} point of the gadget adjacent to E^i, and G_{C_i} is *d*-hop dominated by exactly $N_{C_i} + 3$ sinks. This number is optimal, since to *d*-hop dominate each variable or each gadget, we need at least one sink. On the contrary, if C_i is false, to dominate G_{C_i} in an optimal way, we have to choose g_{d+1} point for each gadget of Type *A* and choose the g_d point for each gadget of Type *B*. This results that the special point E^i cannot be *d*-hop dominated invariably. Therefore, we need at least $N_{C_i} + 4$ sinks to *d*-hop dominate G_{C_i}.

Figure A4 A structure of a graph G_{C_i}, which represents a Boolean expression $C_i = x_i \vee y_i \vee z_i$, where $\{x_i, y_i, z_i\} \subseteq V \bigcup \overline{V}$



A4. Construction of the UDG G_c on the Grid Space from G(C): In this section, we will show how we can reduce PLANAR 3-SAT to M*d*HSP problem. Given a PLANAR 3-SAT Boolean expression *C*, we can construct a planar UDG G_c such that a truth assignment for *C*E implies a *d*-hop clustering of size $N_c + n$ and vice versa, where N_c is number of gadgets in G_c and *n* is number of variables in *C*. As we mentioned 3-PLANAR 3-SAT remains NP-complete, where any variable appears at most three times in any instance of 3-PLANAR 3-SAT. The idea of proof can be summarised as follow. We take a planar embedding of *G*(*E*) for some instance of PLANAR 3-SAT. Let $(u,c_1),(u,c_2),\cdots,(u,c_k)$ be the edges adjacent to variable-vertex *u* in the graph *G*(*C*), which are arranged in clockwise order according to the planar embedding. Now introducing new variables $w_1, w_2, \cdots, w_k$ and clause $\{w_1 \vee \overline{w}_2\}, \{w_2 \vee \overline{w}_3\}, \cdots,$ $\{w_{k-1} \vee \overline{w}_k\}, \{w_k \vee \overline{w}_1\}$. Replace literals $v, \overline{v}$ in C_i by $w_i, \overline{w}_i$, respectively, for $i = 1, 2, \cdots, k$. It is easy to see that the modified formula *C* is satisfiable if and only if *C* is satisfiable. Moreover *G*(*C′*) is a planar graph. In this way, the occurrences of variables in any clauses can be reduced to at most three. Now let *C* be an instance of PLANAR 3-SAT with each variable appearing in at most three clauses. We show how to construct graph G_c from a planar embedding of *G*(*C*): (a) remove the edges $(u_1,u_2),\cdots,(u_{n-1},u_n),(u_n,u_1)$ in *G*(*C*). (b) Replaced each variable-vertex *u* in *G*(*C*) with a path of length 2*d* as in Figure A3. (c) If v_i or $\overline{v}_i$ occurs in a clause C_i, replace the edge connecting c_i and v_i (resp. $\overline{v}_i$) by an edge connecting c_i and the truth (resp. false) point of variable v_i in the path. (d) Note that since each variable-vertex has degree at most three, we can always replace the edges incident to the it by introducing a new vertex of degree at most four [Figure A5(a) and A5(b)]. Embed the planar graph obtained (whose maximal degree is at most four) into a grid space. (e) Regarding clause-vertex c_i in G_c as a special point E^i, we can construct an UDG G_{C_i} by replacing each edge adjacent to c_i in *G*(*C*) with some chains of gadgets constructed previously (Figure A4). Note that each variable-vertex has degree at most three, so we can always replace the edges incident to that by some gadgets as it is shown in Figure A5(c). (f) After the above operations, we can obtain the UDG G_c.

A5. Correctness of our reduction: Let N_c be the number of gadgets used in G_c, and $|T| = N_C + n$.

Lemma A1: *G_c has optimal solution of size $|T| = N_C + n$ for MdHSP if and only if C is satisfiable.*

Proof: First assume that *C* is satisfied by a truth assignment. Note C_i is true, as we have shown before, G_{C_j} can be *d*-hop dominated by $N_{C_i} + 3$ sinks – each literal and each gadget are dominated by exactly one sink. It follows that G_c can be dominated by $N_c + n$ sinks. Note each gadget and each literal in G_c need a relay to *d*-hop dominate them. Thus, *T* is the minimum number of sinks that can *d*-hop dominate G_c. Suppose that G_c has an optimal solution *T*, we show that there exists a truth assignment for each variable such that *C* is satisfiable. We prove by contradiction. Assume that for each assignment there exists a clause C_j which is false. Then by Lemma A1, to *d*-hop dominate the subgraph G_{C_j} of G_c, we need at least $N_{C_j} + 4$ sinks. Thus, for all assignments of variables we need at least $N_c + 4$ sinks to *d*-hop dominate G_c. That means to *d*-hop dominate G_c, we need at least $N_c + 4$ nodes, which contradicts the assumption. It is easy to verify that the reduction is polynomial. This completes the proof.

Finally, we mention that when we are concerned with connected UDGs for our problems, it is always possible to make G_c to be connected by adding some 'blocks', and our proof can be easily adapted to this situation; we omit the details.

Figure A5 We assume that one variable can be used at most three times. We will use this structure to make the maximum degree of G_c at most four when a variable is used in three clauses