Hop-Constrained Relay Node Placement in Wireless Sensor Networks

Xingjian Ding[1], Guodong Sun[2], Deying Li[1], Yongcai Wang[1], and Wenping Chen[1(✉)]

[1] School of Information, Renmin University of China, Beijing 100872, China
{dxj,deyingli,ycw,chenwenping}@ruc.edu.cn
[2] School of Information Science and Technology, Beijing Forestry University, Beijing 100083, China
sungd@bjfu.edu.cn

Abstract. Placing relay nodes in wireless sensor networks is a widely-used approach to construct connected network topology. Previous works mainly focus on the relay node minimization while achieving network connectivity but pay less attention on the path performance guarantee. In this paper we first investigate the hop-constrained relay node placement optimization which aims at using as few relays as possible to construct sensor-to-sink paths meeting the hop constraint given by the end user. We present a heuristic-based algorithm to solve the above optimization problem and evaluate its performance by extensive simulation. The experimental results demonstrate that the efficiency of our designs in comparison with two baselines.

Keywords: Wireless sensor networks · Connectivity
Relay node placement · Hop constraint

1 Introduction

Recent years have witnessed a growing interest of using wireless sensor networks (WSNs) to monitor the physical world [1,2]. A wireless sensor network contains lots of wireless sensor nodes, which are deployed in unattended filed to monitor the physical events, such as the temperature, the light strength, vibration, etc [3]. With the built-in radio chip, these wireless sensor nodes can be organized or self-organized into a wireless multihop network that delivers the sensory data to a sink node at the user end [3].

In real-world wireless sensor network systems, the on-board radio chip of sensors is the most energy hungry component. To reduce the energy consumption and achieve a long-term monitoring, researchers and engineers entirely have

W. Chen—This work was supported, in part, by the National Natural Science Foundations of China with Grants No. 11671400 and No. 61672524, the Fundamental Research Funds for the Central University, and the Research Funds of Renmin University of China with Grant No. 18XNH109.

S. Chellappan et al. (Eds.): WASA 2018, LNCS 10874, pp. 78–89, 2018.
https://doi.org/10.1007/978-3-319-94268-1_7

moved to low-power wireless communication [4]. One typical approach is to reduce the transmit power level of sensor nodes. In practice, therefore, the sensor node provides a limited coverage of wireless communication. In the other hand, some wireless sensor network applications are deployed in harsh environments, such as unattended forests, deserts, and so on. The propagation of low-power wireless signals often suffer nonnegligible reflection, scattering, and diffraction in these fields [5]. Consequently, the network connectivity of the initialized deployment cannot be sufficiently guaranteed in some cases, which leads to uneasy data collection. Of course, placing a great amount of sensor nodes in the monitoring area would offer us a connected network, however, such a redundant deployment is sure to increase the system cost, especially when the targets to be detected locate sparsely with respect to the communication coverage of sensor nodes. To obtain or recover connected network, a promising approach is to place extra nodes in the monitoring area. These nodes are also called *relay nodes* (or *relays*), which usually participate into the data propagation and do not acquisition any sensory data.

It is very challenging to achieve efficient relay placement which has been proven to be NP-Hard [6–8]. Existing works have focused on achieving connected network topology by including least extra relay nodes [6,7,9–11]. Some works study how to achieve connectivity with least relays [6,7,9]. Some articles [10,11] extend the previous researches: besides achieving the connectivity, they also guarantee the fault tolerance. However, they do not pay direct attention on the performance of forwarding paths—usually, these paths traverse count-uncontrollable hops before reaching the sink. As we know, the more the hops along a path are, the longer the delivery time is. Additionally, excessive hops increase the risk of losing data packets in transmission because the low-power links of wireless sensor network are time-varying.

In this paper, we first investigate the Hop-constrAined Relay Placement problem (HARP, in short), to the best of our knowledge, and we present a heuristic-based algorithm for the HARP problem. The major contributions of our work are as follows. First, we present some insightful observations that are beneficial to dissect the HARP problem and help us design better heuristics. Second, to solve the HARP problem, we propose two algorithms which can collectively find efficient solutions. Third, we conduct extensive simulations to evaluate our designs, and results show that our approach always outperforms the baseline algorithm with approximation ratio of 3 [6].

The rest of the paper is organized as follows. We introduces works related to ours in Sect. 2. We describes the problem to be addressed in Sect. 3. We presents the details of algorithm designs in Sect. 4. We evaluates the performance of our work in Sect. 5. Finally, We concludes this paper in Sect. 6.

2 Related Work

In the past few years, placing relay nodes for wireless sensor networks has been studied in various contexts [7–12]. Cheng et al. [6] present two algorithms of

approximation ratios 3 and 2.5 to solve the general optimal relay node placement problem. Ma et al. [13] present a set-covering-based algorithm for delay constrained relay node placement problem in two-tiered wireless sensor networks. They analyzed the time complexity and the approximation ratio of the proposed algorithms, and demonstrate their performance through extensive simulations. Lee and Younis [7] propose an optimized relay placement algorithm under the minimum Steiner tree model with convex hull, and validate its performance through simulation experiments. In [14], the authors address the problem of placing least relays to establish multi-hop paths. They first derive an optimal solution for the case with three terminals and then present three heuristic algorithms to deal with the cases with more than three terminals; the simulation results demonstrate the performance of their algorithms. Ma et al. [9] present a local search approximation algorithm (LSAA) to solve the relay node single cover problem, aimed at improving fault tolerance; also, they extend LSAA to solve the relay node double cover problem.

To address the delay constrained relay node placement problem, [15] presents a set cover-based approximation algorithm (SCA), which deploys relays iteratively from the sink to the given sensor nodes; during the iteration, sensors are gradually connected to the sink with satisfying delay constraint. Authors of [16] propose a subtree and mergence based approach to addressing the delay constrained relay node placement problem. Different from our research, [15,16] assume that relay nodes can only be deployed at the positions specified in advance.

3 Problem

3.1 Network Model

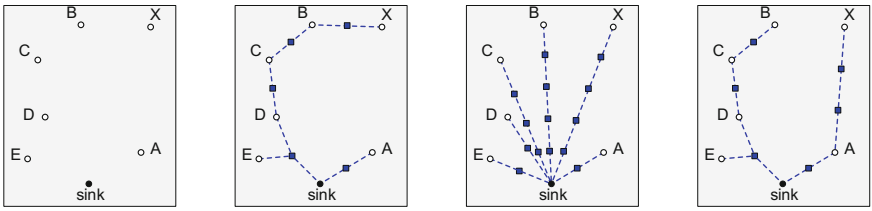
In this paper we assume that the wireless sensor network consists of n sensor nodes and a sink node that collects the data acquired by the sensor nodes. All the sensor nodes and the sink are deployed in a two-dimensional area; the locations of these sensors can be obtained once being deployed. We assume that the sensor nodes and the relay nodes have the same transmit radius ρ. We consider such a scenario: the initial deployment of sensor nodes are sufficiently sparse so that any two sensor nodes, u and v, cannot directly communicate with each other, i.e., $L(u, v) > \rho$ where $L(u, v)$ denotes the physical distance between u and v.

3.2 Problem Description

Relay nodes can help establish a network that connects all the sensor nodes to the sink: offering a forwarding path from each sensor node to the sink. Intuitively, if we deploy relay nodes densely, it is easy to obtain a connected topology. However, the possible redundancy of relay nodes will lead to not only serious co-channel conflicts but also higher deployment cost. This paper studies a novel optimal relay placement problem (HARP) which is to place minimum number of relay

nodes in the target area such that the network topology is connected and satisfies the hop constraint from every sensor to sink.

Figure 1 describes the motivation of the HARP problem, and Fig. 1(a) shows the original network topology that is unconnected. Figure 1(b) shows a case that uses only five relays—least relays—to connected all the segments. In Fig. 1(b), however, the path from X to the sink goes through eight hops. Figure 1(c) simply places the relays on the straight line connecting the segmented sensor and the sink. Figure 1(c) uses fourteen relays, more than those used in Fig. 1(b), and (c) constructs hop-minimum paths for sensors D, E, and X. Obviously, the trivial relay placement scheme shown in Fig. 1(c) results in high deployment cost, especially when the end user does not have a strict demand on the hop count. Figure 1(d) trades off the two relay placements shown in Fig. 1(b) and (c): it pursues not-too-long paths with as few relays as possible.



(a) original (b) least relays (c) least hops (d) constrained hops

Fig. 1. Illustration of three relay placement schemes. The blue square represents the relay node. (Color figure online)

To profile the path performance requirement, we set a hop constraint k for the sensor-to-sink path and $k > 1$. Specifically, the hop count of the path from sensor s_i to the sink s_0, denoted by $h(s_i, s_0)$, should not be greater than k times the hop count $\lceil L(s_i, s_0)/\rho \rceil$ which is the minimum hop count. Denote N_r as the number of relays used. Then, the HARP problem is formally modeled as Eq. (1). The requirement for guaranteeing the network connectivity is implicitly indicated by Eq. (2) and then is not given here for clearance.

$$\textbf{min}: \quad N_r \tag{1}$$

$$\textbf{s.t.}: \quad \frac{h(s_i, s_0)}{\lceil L(s_i, s_0)/\rho \rceil} \leq k, \qquad i \in \{1, 2 \ldots n\} \tag{2}$$

Theorem 1. *The HARP problem is NP-Hard.*

Proof. We consider a special case of HARP problem, in which the hop constraint k is large enough such that it will never pose a valid constraint to the minimization of relay nodes. That is the optimal relay node placement problem in [6]. It means that the HARP problem contains the optimal relay node placement problem as a particular case. Since the optimal relay node placement problem has been proven to be NP-Hard in [6], the HARP problem is also NP-Hard. □

4 Algorithm Designs

4.1 Benefical Observations on the HARP Problem

For the Euclidean Steiner tree problem [17], the goal is to connect given n nodes with minimum total length of all edges (wireless links). For the case of $n = 3$, if the maximum angle of the triangle formed by the given three nodes is less than 120°, the Steiner point locates at the Fermat point which is a point such that the total distance from the three vertices of the triangle to the point is the minimum possible; otherwise, the Steiner point locates at the vertex of the maximum angle. However, the HARP problem targets the reduction of relays with the hop constraint, other than the total length of all edges, and then, it is different from typical Euclidean Steiner tree problems. The following provable observations further profile the HARP problem and provide heuristics for our algorithm designs.

Definition 1. *Given three nodes in a two-dimensional area, for a given point, we evenly place relays between the point and the three vertex to achieve the connectivity. The point is called the best Steiner point if the number of relays used is minimum.*

Observation 1. *The Fermat point is not always the best Steiner point.*

Proof. For a Euclidean triangle Steiner tree, if the largest interior angle is smaller than 120°, then the Steiner point is just the Fermat point; otherwise, it is at the vertex of the angle that is 120° or larger. If each angle of the triangle is less than 120°, the sum of the distances between the Fermat point and the three vertices can then achieve the minimum. However, the HARP model does not target the Euclidean distance optimization. Therefore, Fermat point is not always the best Steiner point in the HARP model. □

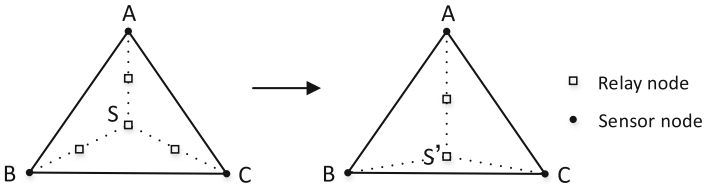


Fig. 2. Illustration of a triangle Steiner tree

As shown in Fig. 2, for example, $\triangle ABC$ is regular and its Fermat point is labeled with S, then we have $|SA| = |SB| = |SC|$. Suppose $|SA| = 1.1\rho$. To connect vertices A, B, and C, we will need four relays to build a triangle Steiner tree if the Fermat point is thought of as the best Steiner point. However, we will need only two relays if point S' is chosen. Obviously, the Fermat point is not optimal in this example.

Observation 2. *The best Steiner point definitely is in a triangle or on its edges.*

Proof. A rough proof of this theorem was given in [7]. Here we propose a strict proof. We need only to prove that every best Steiner point is not outside of the given triangle. Suppose P is a Steiner point of $\triangle ABC$, as shown in Fig. 3, and the three side lengths of $\triangle ABC$ are all greater than ρ, then, we can place relays on the three dashed lines starting from P and ending at A, B, and C, respectively. Now we choose a point Q on side BC such that $|BQ| = |BP|$. Since $|BP| + |CP| > |BQ| + |CQ|$, we have $|CP| > |CQ|$. Obviously, $|AP| > |AQ|$. Thus, we have

$$\left\lceil \frac{|AP|}{\rho} \right\rceil + \left\lceil \frac{|BP|}{\rho} \right\rceil + \left\lceil \frac{|CP|}{\rho} \right\rceil \geq \left\lceil \frac{|AQ|}{\rho} \right\rceil + \left\lceil \frac{|BQ|}{\rho} \right\rceil + \left\lceil \frac{|CQ|}{\rho} \right\rceil \tag{3}$$

By Eq. (3), we know that Q is better than P as a Steiner point; in other words, given a Steiner point outside the triangle, we can always find a better one that locates on some side of the triangle. □

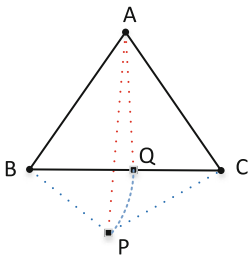


Fig. 3. Illustration for Observation 2

Observation 3. *Given a triangle whose side lengths are all greater than ρ, its circumcenter or midpoint of long edge is possibly the best Steiner point.*

Proof. Consider a special case that a triangle exactly needs one relay node to build a connected topology, then the choice of the best Steiner point can be quite tricky. As only one relay node is needed, we must confirm that the distances from the relay to the three vertices are not greater than ρ, then the triangle's circumcenter or midpoint of long edge that minimizes the maximum distance must be a suitable point. Thus this observation is true. □

4.2 Algorithm Description

Based on the three observations, we present a greedy algorithm for the HARP problem. The main idea of our algorithm (RPC) is heuristically selecting each sensor node and iteratively connecting them to the sink, at each iteration, RPC invokes a local optimization algorithm (FBS) to find the best Steiner point by searching grid cells, and deploy relays to achieve the connectivity.

4.2.1 Locally Optimal Determination of Best Steiner Point

In this subsection, we solve the local optimal relay deployment problem. Given a connected graph $G(V_G, E_G)$, and a separate sensor node u, and a hop constraint k, our goal is to construct a connected graph $G(V_G \cup \{u\}, E'_G)$ such that the total number of relay nodes used is minimized. We first mesh the network area, then greedily find the shortest edge(u, v_0) that meets the hop constraint where $v_0 \in V_G$, and then we form triangles with node u, v_0 and any v_j where v_j is a node incident to v_0 in G. After that, we search the grid cells to find the best Steiner point that satisfies the hop constraint.

The local optimization algorithm is shown in Algorithm 1. Figure 4 shows an walk-through example of Algorithm 1.

Algorithm 1. Finding Best Steiner point (FBS)

Input: $G(V_G, E_G)$, sensor node u, and hop constraint k
Output: a locally optimized graph $G(V_G \cup \{u\}, E'_G)$

1: For each node $v_i \in V_G$ that has a path to sink s_0, compute $h(v_i, s_0)$ and $L(u, v_i)$
2: Find node $v_0 \in V_G$ that is nearest to u, such that

$$h(v_0, s_0) + \lceil L(u, v_0)/\rho \rceil \leq k \times \lceil L(u, s_0)/\rho \rceil$$

3: Form a triangle $\triangle_j$ with nodes u, v_0, and any v_j, where v_j is a node incident to v_0 in G. For each $\triangle_j$, search the grid cells to find the best Steiner point that meets the hop constraint; and calculate n_j^r for each $\triangle_j$
4: **if** $\exists\ n_j^r > 0$ **then**
5: Determine triangle $\triangle^* = \arg\max_{\triangle_j}\{n_j^r\}$
6: Place a relay at the best Steiner point of $\triangle^*$ to build a triangle Steiner tree
7: **else**
8: Connect u with v_0 directly
9: **end if**
10: Place least relays on the edges whose length is greater than ρ
11: Update $h(v_i, s_0)$ for each node $v_i \in V_G$
12: Return $G(V_G \cup \{u\}, E'_G)$

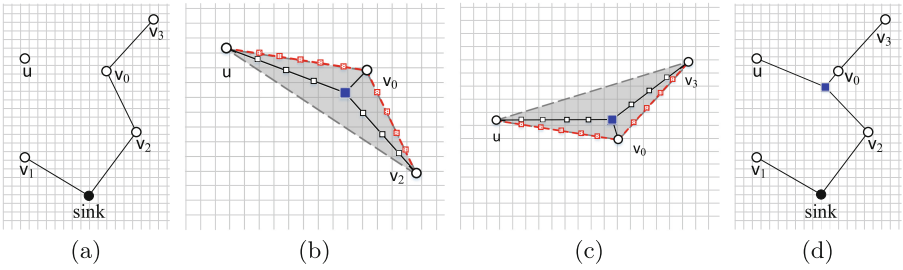


Fig. 4. An example of the FBS algorithm, squares in (b) and (c) represent relays.

Consider node u that will be added into G in Fig. 4. In Fig. 4(a), suppose node v_0 is physically nearest to u and it provides a path from u to sink s_0 that

satisfies the hop constraint. Then we obtain two candidate triangles $\triangle_1$ and $\triangle_2$ shown in Fig. 4(b) and (c), both of which are formed with nodes $\{u, v_0, v_2\}$ and nodes $\{u, v_0, v_3\}$, respectively. Algorithm 1 scans the grid cells that partially or fully locate within the two triangles, and then calculates n_1^r and n_2^r. Here we use n_j^r to denote the number of relays that we can save if the Steiner tree is built on triangle $\triangle_j$. Algorithm 1 chooses the triangle achieving maximum n^r. If ties occur, it chooses the triangle that minimizes $h(u, s_0)$ and then deploys a relay node at the Steiner point to build a triangle Steiner tree. Here, we suppose triangle $\triangle_1$ has the maximum n^r and then is chosen; the final topology is shown in Fig. 4(d), which includes a relay node to construct a connected network.

By Observation 3 we know that the circumcenter of triangle $\triangle$ and the midpoint of $\triangle$'s long edge are special points that can achieve the best Steiner point with considerable probability. So Algorithm 1 also examines the two special points in line 3, when it searches the grid cells. Noticeably, deploying relays at Steiner points affect the path formation of other nodes and then their hop count. In Fig. 4(d), for instance, the inclusion of a relay node increases the hop count of node v_3. Therefore, when searching the grid cells to find the best Steiner point in line 3, Algorithm 1 needs to guarantee that besides node u, every node in V_G satisfies the hop constraint.

4.2.2 Relay Placement with Hop Constraint

The relay placement with hop constraint algorithm first initializes graph $G(V_G, E_G)$ by setting $V_G = \{s_0\}$ and $E_G = \emptyset$, and then puts all sensor nodes into set S. RPC iteratively connects each sensor to the sink node. The choice of which node will be added into G is determined by a Prime's-like heuristic, meanwhile any edges examined satisfy the hop constraint. Once RPC figures out the node to be added into G, it will invoke algorithm FBS to complete the relay placement process. The RPC algorithm is shown as follows:

Algorithm 2. Relay Placement with hop Constraint (RPC)

Input: locations of all the sensors and sink s_0, hop constraint k
Output: a connected Graph $G(V_G, E_G)$ with k-constrained hops
1: Initialize $G(V_G, E_G)$ with $V_G = \{s_0\}$ and $E_G = \emptyset$
2: Put all the sensors into a set S
3: Compute $L(s_i, s_0)$ for each sensor $s_i \in S$
4: **while** $S \neq \emptyset$ **do**
5: Of all edges $\{(u, v) | u \in S, v \in V_G\}$, find a shortest edge (u_0, v_0) that satisfies the hop constraint: $h(v_0, s_0) + \lceil L(u_0, v_0)/\rho \rceil \leq k \times \lceil L(u_0, s_0)/\rho \rceil$.
6: Run FBS to add node u_0 into $G(V_G, E_G)$
7: Remove node u_0 from S
8: **end while**
9: **return** $G(V_G, E_G)$

Theorem 2. *The network topology generated by Algorithm 2 is feasible for the HARP problem.*

Proof. In Algorithm 2, sensor nodes are added into G one by one, until all of them are connected with the sink. So Algorithm 2 is sure to yield a connected topology. The connectivity establishment for each sensor is executed by Algorithm 1, which ensures that the hop constraint of every node can be met when the grid cells are searched. Thus the theorem holds. □

Theorem 3. *The time complexity of Algorithm 2 is* $\mathcal{O}\left(n^3 + \frac{\mathcal{A}}{d^2} \times n\right)$*, where* $\mathcal{A}$ *denotes the area of the network region, and* d*, the side length of the grid cell.*

Proof. Line 5 of Algorithm 2 takes time of $\mathcal{O}(n^2)$; and the time cost of Algorithm 1 is mainly determined by the number of grid cells to be searched. The area of a grid cell of side length d is equal to d^2, and then the number of grid cells is bounded by $\frac{\mathcal{A}}{d^2}$. Therefore, the total time complexity of Algorithm 2 is $\mathcal{O}\left(n \times \left(n^2 + \frac{\mathcal{A}}{d^2}\right)\right)$. This theorem holds. □

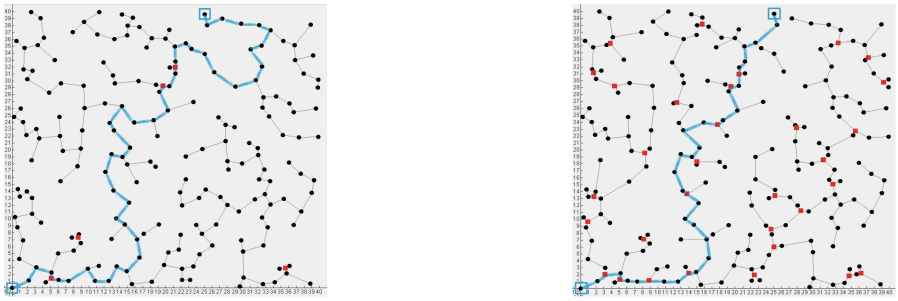
5 Simulation

5.1 Experimental Settings and Metrics

In our simulations, we consider two cases: fixed network area with different network scales, and fixed sensor density with random deployments. In the first case, the network area is a 40×40 square, within which sensor nodes were randomly deployed. In the second case, we set the sensor density with 0.1 and 0.2, and also, the sensor nodes were randomly deployed. In all experiments ρ was set to 1 and the sink node located at the left bottom corner of the network area. We set the hop constraint k to 2. In Algorithm 1, the side length of each grid cell was set to 0.1. Each data point was the average of 40 repeated experiments with random seeds. We use three metrics to evaluate the performance of our work by comparing it with the Direct algorithm and STP-MSP [6] that is a 3-approximation algorithm. As a baseline, the Direct algorithm places relay nodes only on the straight line that connects the isolated sensor node with the sink, as Fig. 1(c) does. The three metrics are (1) the number of relays used, (2) the average hop count of the resulted paths, and (3) the maximum hop count of the resulted paths.

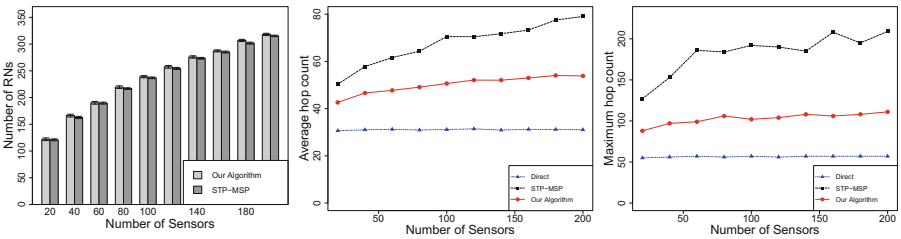
5.2 Results and Analyses

Figure 5 shows the resulted topologies of the STP-MSP and our designs when 200 sensor nodes were randomly deployed in a 40×40 square area. Note here that (1) STP-MSP runs without considering any hop constraint, and (2) in Fig. 5, we only show the relay nodes locating at the Steiner points and omit other relays that are evenly deployed on each edge. From Fig. 5 we can see that the topology returned by STP-MSP has much longer paths than that by the proposed RPC. Later figures will plot detailed comparisons.



(a) by STP-MSP (b) by RPC

Fig. 5. Comparison of two resulted topologies with $n = 200$. Black dots and red squares represent the sensor nodes and the relay nodes placed at the Steiner points, respectively. (Color figure online)

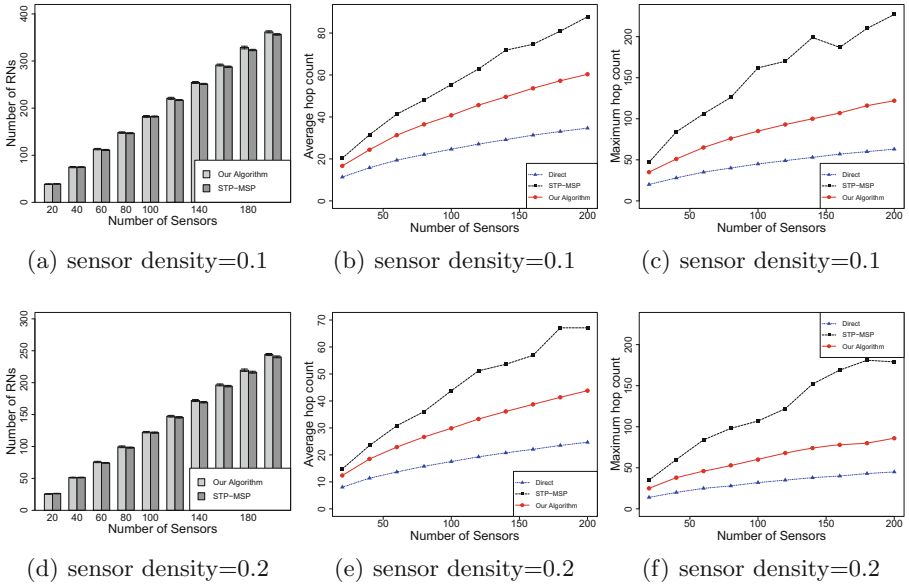


(a) (b) (c)

Fig. 6. Performance comparison under different network scales

Figure 6(a) compares the two algorithms (STP-MSP and RCP) with respect to the number of relays. The relay nodes needed by the Direct algorithm are far more than those of STP-MSP or RCP; so, we do not plot the number of relay nodes under Direct in Fig. 6(a), in order to make a clear comparison between STP-MSP and RCP. We can see that the relay nodes (RNs) used by our algorithm are slightly more (0.3–2.2%) than those of STP-MSP. Especially, as the network scale increases, the extra relay nodes of our algorithm over STP-MSP keeps relatively stable in amount. And then we think the proposed RPC can produce acceptable performance in terms of the number of relay nodes. Figure 6(b) and (c) shows the comparison of two algorithms in terms of average hop count and maximum hop count. In Fig. 6(b) and (c), we can see that the proposed RPC achieves shorter paths in comparison with STP-MSP, and the average and the maximum hop counts of RPC grow slowly with the number of sensor nodes increasing. Figure 6(b) and (c) collectively indicate that generally, the network created by RPC is more reliable than that by STP-MSP because of shorter delivery paths. In addition, compared to the Direct algorithm, the number of hops for each sensor node under our algorithm always satisfies the hop constraint ($k = 2$).

Figure 7 compares the performance of the three algorithms under two different densities of sensor deployment. It can be seen in Fig. 7(a) and (d) that



(a) sensor density=0.1 (b) sensor density=0.1 (c) sensor density=0.1

(d) sensor density=0.2 (e) sensor density=0.2 (f) sensor density=0.2

Fig. 7. Performance comparison with different densities of sensor deployment

for both algorithms, the number of relays needed increases in an approximately-linear manner, as the network scale increases, and that the relays needed by the proposed RPC and STP-MSP are almost identical in number. In the case with sparse sensor deployment (e.g., $n = 20$), specially, RPC needs relays 1.3% and 2.4% less than STP-MSP under the deployment densities of 0.1 and 0.2, respectively.

Figure 7(b) and (c) show the path performances of the three algorithms under the sensor deployment of density 0.1. We find that the average and the maximum hops both remain rising under the three algorithms, with the number of sensors increasing. But, our algorithm always keeps the hop constraint satisfied, while achieving better path performance than STP-MSP with almost the same amount of relays. Another advantage of RCP over STP-MSP is that RCP can achieve lower growth rates of the average and the maximum hops than STP-MSP. Combined with Fig. 6, we can draw a conclusion that the connected network topologies returned by our algorithm can achieve more acceptable network performances than the ratio-3 STP-MSP, regardless of the sensor deployment density and the network scale.

6 Conclusions

In this paper, we have presented the design (algorithm RPC) of approximately solving the HARP problem in wireless sensor networks. Based on greedy heuristics, RPC iteratively chooses the closest node that satisfies the hop constraint and adds it to the connected graph. RPC employs a locally-optimal algorithm

(FBS) which builds triangle Steiner trees in each RPC iteration and finds out the best Steiner points for the relay placement. We have also validated the performance of the proposed design and compared it with two baselines. The simulation results show the efficacy of our work. In the future, we will extend our designs to support fault tolerance and pursue approximation algorithms with proven competitiveness.

References

1. Younis, M., Akkaya, K.: Strategies and techniques for node placement in wireless sensor networks: a survey. Ad Hoc Netw. **6**(4), 621–655 (2008)
2. Manshahia, M.S.: Wireless sensor networks: a survey. Int. J. Sci. Eng. Res. **7**(4), 710–716 (2016)
3. Akyildiz, I.F., Su, W., Sankarasubramaniam, Y., Cayirci, E.: A survey on sensor networks. IEEE Commun. Mag. **40**(8), 102–114 (2002)
4. Potdar, V., Sharif, A., Chang, E.: Wireless sensor networks: a survey. International Conference on Advanced Information NETWORKING and Applications Workshops, pp. 636–641 (2009)
5. Ding, X., Sun, G., Yang, G., Shang, X.: Link investigation of IEEE 802.15.4 wireless sensor networks in forests. Sensors **16**(7), 987 (2016)
6. Cheng, X., Du, D.Z., Wang, L., Xu, B.: Relay sensor placement in wireless sensor networks. Wireless Netw. **14**(3), 347–355 (2008)
7. Lee, S., Younis, M.: Optimized relay node placement for connecting disjoint wireless sensor networks. Comput. Netw. **56**(12), 2788–2804 (2012)
8. Xu, X., Liang, W.: Placing optimal number of sinks in sensor networks for network lifetime maximization. In: IEEE International Conference on Communications, pp. 1–6 (2011)
9. Ma, C., Liang, W., Zheng, M., Sharif, H.: A connectivity-aware approximation algorithm for relay node placement in wireless sensor networks. IEEE Sens. J. **16**(2), 515–528 (2015)
10. Bhuiyan, M.Z.A., Cao, J., Wang, G.: Deploying wireless sensor networks with fault tolerance for structural health monitoring. IEEE Trans. Comput. **64**(2), 382–395 (2015)
11. Lee, S., Younis, M., Lee, M.: Connectivity restoration in a partitioned wireless sensor network with assured fault tolerance. Ad Hoc Netw. **24**(PA), 1–19 (2015)
12. Abbasi, A.A., Younis, M.F., Baroudi, U.A.: Recovering from a node failure in wireless sensor-actor networks with minimal topology changes. IEEE Trans. Veh. Technol. **62**(1), 256–271 (2013)
13. Ma, C., Wei, L., Meng, Z.: Delay constrained relay node placement in two-tiered wireless sensor networks: a set-covering-based algorithm. J. Netw. Comput. Appl. **93**, 76–90 (2017)
14. Senel, F., Younis, M.: Novel relay node placement algorithms for establishing connected topologies. J. Netw. Comput. Appl. **70**, 114–130 (2016)
15. Ma, C., Liang, W., Zheng, M.: Set covering-based approximation algorithm for delay constrained relay node placement in wireless sensor networks. Comput. Sci. **36**(8), 1–5 (2015)
16. Ma, C., Wei, L., Meng, Z.: Delay constrained relay node placement in wireless sensor networks: a subtree-and-mergence-based approach. Mob. Netw. Appl. 1–13 (2017). https://link.springer.com/journal/11036/onlineFirst/page/8
17. Lin, G.H., Xue, G.: Steiner tree problem with minimum number of steiner points and bounded edge-length. Inf. Process. Lett. **69**(2), 53–57 (1999)