Delay Constrained Optimal Relay Placement for Planned Wireless Sensor Networks

Abhijit Bhattacharya and Anurag Kumar

Dept. of Electrical Communication Engineering
Indian Institute of Science, Bangalore, 560012, India
email: abhijit@ece.iisc.ernet.in, anurag@ece.iisc.ernet.in

***Abstract*—In this paper, we study the problem of wireless sensor network design by deploying a minimum number of *additional* relay nodes (to minimize network design cost) at a subset of *given potential* relay locations in order to convey the data from already existing sensor nodes (hereafter called source nodes) to a Base Station within a certain *specified mean delay bound*. We formulate this problem in two different ways, and show that the problem is NP-Hard. For a problem in which the number of existing sensor nodes and potential relay locations is n, we propose an $O(n)$ approximation algorithm of polynomial time complexity. Results show that the algorithm performs efficiently (in over 90% of the tested scenarios, it gave solutions that were either optimal or exceeding optimal just by one relay) in various randomly generated network scenarios.**

I. Introduction

Large industrial establishments such as refineries, power plants, and electric power distribution stations, typically have a large number of sensors distributed over distances of 100s of meters from the control center. Individual wires carry the sensor readings to the control center. Recently there has been increasing interest in replacing these wireline networks with wireless packet networks ([1], [2]). A similar problem arises in an intrusion detection application using a fence of passive infrared (PIR) sensors [3], where the event sensed by several sensors has to be conveyed to a Base Station (BS) quickly and reliably.

The communication range of the sensing nodes is typically several tens of meters. Therefore, usually multi-hop communication is needed to transmit the sensed data to the BS. The problem then is *to design a multi-hop wireless mesh network with minimum deployment cost, i.e., minimum number of additional relays,* so as to communicate from each sensing (source) node to a central node, which we will call the BS (we shall use the terms BS and sink interchangebly), while meeting a certain performance objective such as a *delay bound.*

The relay placement problems can be broadly classified into two classes, namely, the *unconstrained* relay placement problem, where the relay locations can be anywhere in the 2-dim plane, and the *constrained* relay placement problem, where the relays can be placed only at certain pre-specified potential locations. In most practical applications, due to the presence of obstacles to radio propagation (e.g., a firewall, a large machine, or a building), or due to taboo regions (e.g., a pond or a ditch), we can not place relay nodes anywhere in the plane, but only at certain designated locations. This leads to the problem of *constrained relay placement*. In this paper, we shall study the problem of *constrained relay placement subject to an end-to-end delay bound.*

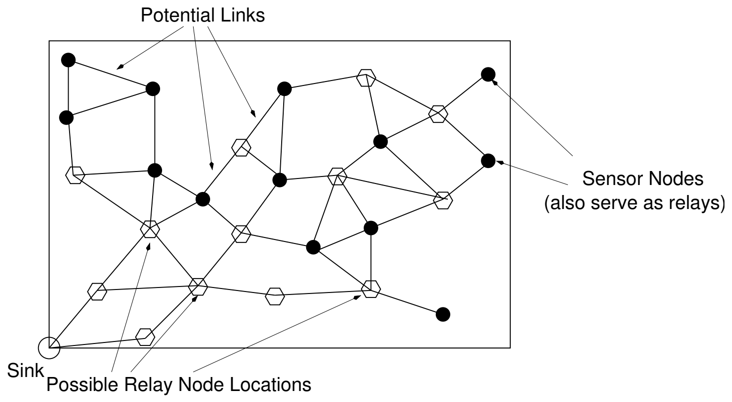
Potential Links
Sensor Nodes (also serve as relays)
Sink
Possible Relay Node Locations

Fig. 1. The constrained relay placement problem; circles indicate sources, hexagons indicate potential relay locations. The edges denote the useful links between the nodes.

Figure 1 depicts the constrained relay placement problem.

- The source locations and possible relay locations are specified.
- Only certain links are permitted. This is because some links could be too long, leading to high bit error rate and hence large packet delay. Other potential links may not exist due to an obstacle, e.g., a firewall, or a building.
- The problem is to obtain a subnetwork that connects the source nodes to the base station with the requirement that
    1) A minimum number of relay nodes is used.

2) The mean delay is bounded by a given value d_{max}.

In this paper, we address this problem for the case in which (a) the nodes use the CSMA/CA Medium Access Control (as standardized in IEEE 802.15.4), and (b) the traffic from the source nodes is such that at any point of time only one measurement packet flows from a source in the network to the base station. We call this the "lone packet traffic model", which is realistic for many applications where the time between successive measurements being taken is sufficiently long so that the measurements can be staggered so as not to occupy the medium at the same time. Note that even if the traffic is so infrequent, the end user may still like to constrain the delay between when a measurement packet is generated and when the packet is received. In applications, the measurements are currently conveyed to the BS via a wireline network. While replacing the wireline network (which is expensive to install and maintain) with a wireless mesh network, we would like to constrain the end-to-end performance achieved by the wireless network. Moreover, we expect the design for the lone packet model to serve for the case of continuous traffic from all sources, under very light load.

A. Related Literature

We see that the problem we have chosen to address belongs, broadly, to the class of Steiner Tree Problems (STP) on graphs ([4], [5], [6]).

The classical STP is stated as: *given an undirected graph $G = (V, E)$, with a non-negative weight associated with each edge, and a set of* required vertices $Q \subseteq V$*, find a minimum total edge cost subgraph of G that spans Q, and may include vertices from the set $S := V - Q$, called the* Steiner vertices.

The classical STP dates back to Gauss and it has been proven to be NP-Hard. Lin and Xue [7] proposed the Steiner Tree Problem with Minimum Number of Steiner Points and Bounded Edge Length (STP-MSPBEL). The STP-MSPBEL was stated as: given a set of n terminal points Q in 2-dimensional Euclidean plane, find a tree spanning Q, and some additional Steiner points such that each edge has length no more than R, and the number of Steiner points is minimized. The problem was shown to be NP-complete and a polynomial time 5-approximation algorithm was presented. This problem was the first well-studied problem on optimal relay placement (relay locations *unconstrained*).

Cheng et al. [8] studied the same problem as Lin and Xue, and proposed a 3-approximation algorithm and a 2.5-approximation algorithm.

Lloyd and Xue [9] studied a generalization of STP-MSPBEL problem where each sensor node has range r and each relay node has range $R \geq r$. They provided a 7-approximation polynomial time algorithm. They also studied the problem of minimum number of relay placement such that there exists a path consisting solely of relay nodes between each pair of sensors. For this problem, they provided a $(5 + \epsilon)$-approximation algorithm. The problems studied by Lloyd and Xue, as well as Cheng et al. fall in the category of *unconstrained* relay placement problem.

Voss [10] studied the Steiner Tree Problem with Hop Constraints (STPH). This problem is stated as: *given a directed connected graph $G = (V, E)$, with non-negative weight associated with each edge, consider a subset of V, namely, $Q = \{0, 1, 2, \ldots, n\}$ with 0 being the root vertex, and a positive integer H. The problem is to find a minimum total edge cost subgraph T of G such that there exists a path in T from 0 to each vertex in $Q \backslash \{0\}$ not exceeding H arcs (possibly including vertices from $S := V - Q$).* We can call this problem the *Rooted Steiner Tree-Minimum Weight-Hop Constraint* problem (RST-MW-HC). This problem was shown to be NP-Hard, and a Minimal Spanning Tree based heuristic algorithm was proposed to obtain a good quality feasible solution, followed by an improvement procedure using a variation of *Local Search* method called the *Tabu search* heuristic. No performance guarantee or complexity analysis of the heuristic was provided.

Costa et al. [11] studied the Steiner Tree Problem with revenue, budget, and hop constraints. Given a graph $G = (V, E)$, with a cost associated with each edge, and a revenue associated with each vertex, the problem is to determine a revenue maximizing tree subject to a total edge cost constraint, and a hop constraint between the root vertex and every other vertex in the tree. They propose a greedy algorithm for initial solution followed by destroy-and-repair or tabu search to improve the initial solution.

Kim et al. [12] studied the Delay and Delay Variation Constrained multicastng Steiner Tree Problem. The problem is similar to the one studied by Voss, with a delay constraint instead of the hop constraint, and a constraint on delay variation between two sources. With the delay variation constraint relaxed, Kim's problem becomes the *Rooted Steiner Tree-Minimum Weight-Delay Constraint* problem. They proposed a polynomial time heuristic algorithm to obtain reasonably good feasible solutions, but they did not provide any performance guarantee for their algorithm.

Bredin et al. [13] studied the problem of optimal relay placement *(unconstrained)* for $k-$connectivity. They proposed an $O(1)$ approximation algorithm for the problem with any *fixed* $k \geq 1$.

Misra et al. [14] studied the *constrained* relay placement problem for connectivity and survivability. They provided $O(1)$ approximation algorithms for both the problems. We can call their first problem the *Rooted Steiner Tree-Minimum Relays* problem, and their second problem, the *Rooted Steiner Tree-Minimum Relays-Survivability* problem. Although their formulation takes into account an edge length bound, namely edge length$\leq r_c$, which can model the link quality, the formulation does not involve a path constraint such as the hop count along the path; hence, there is *no constraint on the end-to-end delay.*

We shall provide a brief comparison of our work with some of the closely related works discussed here after we have described our problem formulation in Section II.

The rest of the paper is organized as follows: in Section II, we describe the assumptions made and the two different problem formulations, and show that the problems are NP-Hard. In Section III, we describe the proposed algorithm, and provide a complete analysis of the algorithm. In Section IV, we provide numerical results for the algorithm applied to a set of random scenarios. Finally, we conclude the paper in Section V.

II. Assumptions and Problem Formulation

A. Assumptions

In several industrial telemetry applications, the rate at which measurements are obtained from the sensors is low, for example, as little as one reading per hour from each sensor. We also assume that the alarm traffic is so infrequent that it does not interfere with any regular data transmission. Then, if the data transmission from the sensors is staggered over the hour, it can be assumed that each measurement packet flows over the network with no interference from any other packet flow. Our work in this paper is concerned with this "lone packet" traffic model. We also assume that IEEE 802.15.4 standard [15, p. 30-179, p. 640-643] is used for PHY and MAC layers.

Under the above assumptions, we can obtain an expression for the received signal to noise ratio (SNR) as a function of the hop distance, given the transmit power level, using the path loss model given in the standard, and accounting for a shadow fade margin of 20 dB. Hence, the bit error rate (ϵ) on a link can be obtained as a function of hop distance, again by using a formula given in the standard. Then, for a PHY packet data unit length of L bytes, the packet error rate (PER) on a link can be obtained as $1-(1-\epsilon)^L$. Given the PER on a link as a function of hop distance, we can obtain an expression for the mean delay on that link as a function of hop distance, using the backoff behavior and parameters of IEEE 802.15.4 CSMA/CA MAC. We use this analysis for Formulation 1 in Section II-B below. Also, given the end-to-end distance from a source to the sink, and assuming all hops to be of equal length from the source to the sink, we can obtain the end-to-end mean delay as a function of the single hop distance. Figure 2 shows a plot of mean single hop delay as a function of hop distance, and also a plot of mean end-to-end delay as a function of hop distance for an end-to-end distance of 500 meters and five different power levels, assuming a PHY packet data unit of 90 bytes (corresponding to a network layer packet length of 40 bytes), and a shadow fade margin of 20 dB. For a small hop distance, the mean single hop delay is just the time taken to send a packet once on the link; this time includes the mean initial back-off time in IEEE 802.15.4 CSMA, and also any physical layer and MAC layer overheads. The end-to-end delay, is large for small hop length since there is a large number of hops, although each hop has a small PER, and is again large for a large hop length since the PER on each hop is large, although the number of hops is small.

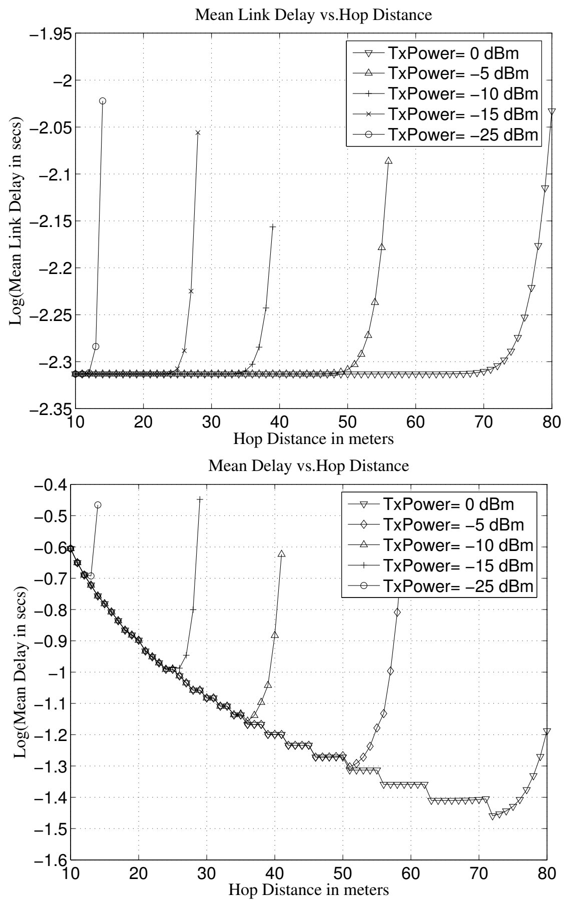

Fig. 2. Upper panel: Mean single hop delay vs. hop distance. Lower panel: Mean end-to-end delay vs. hop distance for an end-to-end distance of 500m. All hops are assumed to be of equal length. On each hop, unslotted CSMA/CA as defined in IEEE 802.15.4 is used. A transmitted packet fails only due to channel errors, which are modeled using standard formulas for the IEEE 802.15.4 PHY.

B. Problem Formulation

With the delay model obtained as above, we can proceed to formulate our relay placement problem as follows:

Formulation 1: Given the set of source nodes or required vertices Q (including the BS) and the set of potential relay locations R, also called Steiner vertices, consider the complete graph $G = (V, E)$, where $V = Q \cup R$ and E consists of all feasible edges. For each edge $e \in E$, assign to that edge an edge weight $d(e)$ which is the mean single hop delay on that edge, obtained from the above delay model (see Figure 2, top panel). Then, given the mean delay requirement d_{max}, the problem is to form a spanning tree on Q, rooted at the sink, using a minimum number of relays such that the mean delay from each source node to the BS is bounded by d_{max}. Let us call this problem the *Rooted Steiner Tree-Minimum Relays-Delay Constraint* (RST-MR-DC) problem.

Formulation 2: Given the maximum distance from a source to the BS, l_{max}, and the transmit power used, we can obtain the plot of the maximum end-to-end delay as a function of hop distance, as shown in Figure 2. From this plot, we can obtain the optimal hop distance r_c, and hence the hop count h_{max}, sufficient to maintain the specified delay bound of d_{max} for the farthest source. Then, we can construct a graph $G = (V, E)$ on $V = Q \cup R$ with E consisting of edges of length $\leq r_c$. Then the problem is again to extract from this graph, a spanning tree on Q, rooted at the BS, using minimum number of relays such that the hop count from each source to the BS is $\leq h_{max}$. Let us call this problem the *Rooted Steiner Tree-Minimum Relays-Hop Constraint* (RST-MR-HC) problem.

Remark: Evidently, the RST-MR-HC problem is a special case of RST-MR-DC problem. To see this, consider the complete graph on $V = Q \cup R$ and assign an edge weight of one to each edge of length $\leq r_c$ and an edge weight of infinity to all other edges. Then, on this weighted graph, the RST-MR-DC problem with $d_{max} = h_{max}$ is precisely the RST-MR-HC problem. Hence we shall design our algorithm for the general RST-MR-DC problem as it will apply without modification to the other case.

Limitation: Note that in Formulation 2, the edge length and hop count constraints provide a sufficient, *but not necessary*, condition to meet the delay requirement for the farthest source. Hence, for a given network scenario and for a given delay requirement, it may turn out that the RST-MR-HC problem is delay-infeasible, while the RST-MR-DC problem is feasible, i.e., there exist paths from each source to the BS satisfying the delay bound, but they do not satisfy the edge length and/or hop count bounds.

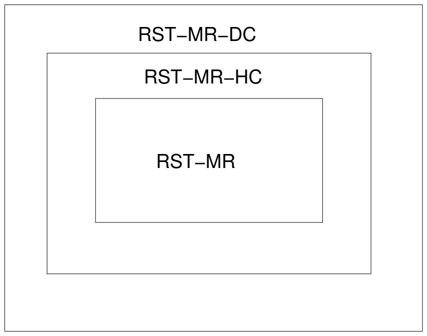

Fig. 3. Venn Diagram showing the relationship among RST-MR-DC, RST-MR-HC, and RST-MR problems

C. Complexity of the Problem

Proposition 1. *The RST-MR-DC and RST-MR-HC problems are NP-Hard.*

Proof: Consider the RST-MR-HC problem. The subset of RST-MR-HC problems where the hop count bound is trivially satisfied is precisely the class of RST-MR [14] problems (consider, for example, all RST-MR-HC problems where $|Q|+|R| = n$, n being some positive integer, and the hop count bound is $h_{max} = n - 1$. Clearly, the hop count bound is trivially satisfied in these problems). Thus, the RST-MR problem is a subclass of the RST-MR-HC problem. This (and an earlier remark in the previous subsection) leads to the situation shown in Figure 3. But, the RST-MR problem is NP-Hard (see [14]). Hence, the RST-MR-HC problem, being a superclass of the RST-MR problem, is also NP-Hard [16, p. 63, Section 3.2.1]. Since the RST-MR-HC problem is a special case of the RST-MR-DC problem, the RST-MR-DC problem is also NP-Hard. ■

D. A Comparison with Closely Related Works

In Table I, we present a brief comparison of the problem under study with some of the closely related problems studied in the literature.

TABLE I
A Comparison with Closely Related Literature; the "starred" problems are the ones we address in this paper

Problem	End-to-End Performance Objective	Complexity	Approximation Guarantee of Proposed Algorithm
RST-MR [14]	×	NP-Hard	6.2
RST-MW-HC [10]	✓	NP-Hard	×
RST-MW-DC [12]	✓	NP-Hard	×
RST-MR-DC∗	✓	NP-Hard	polynomial factor
RST-MR-HC∗	✓	NP-Hard	polynomial factor

III. Proposed Algorithm and its Analysis

We describe the proposed approximation algorithm for the RST-MR-DC problem, where we are given the sets

Q, R, and the edge weighted (edge weight = hop delay on that edge), undirected, complete graph $G = (V, E)$. Also given is the required mean delay bound d_{max}. Since the RST-MR-HC problem is a special case of the RST-MR-DC problem, the algorithm applies without modification to the RST-MR-HC problem.

A. Shortest Path Tree (SPT) based Iterative Relay Pruning Algorithm

1) **The Zero Relay Case:** Find the SPT on Q alone, rooted at the sink. If the delay $\leq d_{max}$ for each path, we are done; no relays are required in an optimal solution. Else, go to the next step.
2) Find the Shortest Path Tree T on G, rooted at the sink.
3) **Checking Feasibility:** If for any path in the SPT, the path weight exceeds d_{max}, declare the problem infeasible. (Clearly, if the shortest path from a node to the sink does not meet the delay bound, no other path from the node to the sink will meet the delay bound). Else go to the next step.

Pruning the SPT:

4) Discard all nodes in $R \backslash T$. Note that this step may lead to suboptimality as some nodes in $R \backslash T$ could be part of an optimal solution. See Figure 9 for example.
5) For paths with path weight= d_{max} (*binding paths*), declare the set of relays on those paths as the *locked set (L)*, i.e., none of those relays will be removed in further iterations.
6) Now, for the remaining relay nodes in R, define the weight of a relay node as the number of paths in the SPT that use the node.
7) Arrange the paths in SPT, except the binding paths, in increasing order of cost.
8) Choose the least cost path that contains relay nodes. Arrange the relay nodes on this path in increasing order of their weights as defined in (7).
9) Remove the least weight relay node and consider the restriction of G to the remaining nodes in T. Find an SPT on this graph. If in this SPT, path cost exceeds d_{max} for any path, then discard this SPT, replace the removed relay node, and repeat this step with the next least weight relay node. If all the relays in the least cost path have been tried without success, move on to the next least cost path, and repeat steps 8 and 9 for the relays in this path that have not yet been tried.
10) If in the above step, the SPT obtained satisfies the delay constraint for all the paths, then delete the removed relay node permanently from R and repeat steps 5 through 10. Note that as the algorithm progresses, the locked set L may grow (whenever Step 5 is repeated), as some more paths may become binding.
11) Stop when all the relays excepting those in the set L have been tried.

Discussion:

Step 1 of the above algorithm ensures that if the optimal design does not use any relay node, then the same goes true for our algorithm. That way we can make sure that the algorithm does not do infinitely worse in the sense that $\frac{Relay_{algo}}{Relay_{opt}}$ is finite.

The idea behind Steps 8, 9 and 10 is that choosing to remove a relay from the path with most slack in cost (i.e., delay or hop constraint), we stand a better chance of still meeting the delay requirement with the remaining relays. Also, removing a relay of less weight would mean affecting the cost of less number of paths. So by pruning relays in the manner as described in Steps 8, 9 and 10, we aim for a better exploration of the search space.

B. Analysis of the Algorithm

1) Complexity: The complexity of determining the shortest path tree on N nodes is $N \log N$ [17]. Let us denote this function by $g_{SPT}(.)$. In Iteration 1 of the algorithm, the complexity is $g_{SPT}(|Q|)$ and in iteration 2, it is $g_{SPT}(|Q| + |R|)$. In subsequent iterations, we remove 1 relay node at a time and find the SPT on the resultant complete graph; if no improvement is found, we replace that node and continue. Thus, for the k^{th} iteration, the worst case complexity will be $(|R| - k + 3)g_{SPT}(|Q| + |R| - k + 2)$, where in the worst case, $k = 3, 4, \ldots, |R| + 1$. Let $g(.)$ denote the overall complexity. Thus, the overall complexity will be

$$\begin{aligned} g(|Q| + |R|) &= g_{SPT}(|Q| + |R|) + \\ &\sum_{j=1}^{|R|} (g_{SPT}(|Q| + |R| - j))(|R| - j + 1) \\ &\leq (1 + |R|^2)(g_{SPT}(|Q| + |R|)) \end{aligned}$$

which is polynomial time.

2) Worst Case Approximation Factor: The worst case occurs when the SPT obtained before we enter Step (4) does not contain any relay node(s) that correspond to some optimal design. If no relays are used in any optimal design, then the algorithm will yield an optimal design (Step (1)). Hence, the worst possibility is that the optimal design uses just 1 relay node, whereas the SPT obtained in Step (2) consists of all the remaining $(|R| - 1)$ relays, and moreover, pruning any of these $(|R| - 1)$ relays will cause one or more paths in the resulting SPT to violate the delay constraint. Thus, in the worst case, the algorithm leads to a design with $(|R| - 1)$ relays instead of the optimal design with one relay. Hence, we have a polynomial factor worst case approximation guarantee of $(|R| - 1)$.

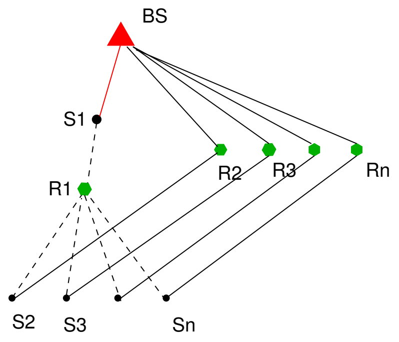

Fig. 4. A Sequence of Problems where the Worst Case Approximation Guarantee is Strict

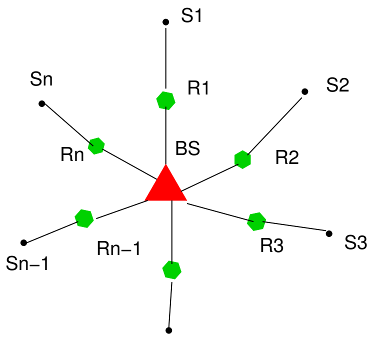

Fig. 5. A Sequence of Problems where the Algorithm gives Optimal Solution

3) Sharp Examples: Let us now present *a sequence of problems of increasing complexity for which the approximation guarantee is strict*, i.e., for these problems, the algorithm ends up using $|R|-1$ relays, while the optimum design uses one relay. Such examples are worthwhile to explore as they help to show the correctness of the performance analysis of a proposed algorithm. Consider the situation shown in Figure 4. The green hexagons denote the relay node locations and the black circles represent the source node locations. Only the edges shown (coloured or black) are permitted. Consider the RST-MR-HC problem on this graph with $h_{max} = 3$. Clearly the optimal solution will use only one relay, $R1$, to reach from each source to the BS within the specified hop count bound. The black dotted links correspond to the optimal solution. The red link will belong to both the optimal solution and the outcome of our algorithm as it is a direct link between source $S1$ and the BS. Our SPT based algorithm will calculate the shortest paths and thus end up using relays $R2, R3, \ldots, Rn$, leaving out $R1$. The black solid links correspond to the solution given by our algorithm. Clearly, in such problems, we end up using $|R|-1$ relays instead of just one.

Another sequence of problems of increasing complexity for which the algorithm gives the optimal design can be constructed as shown in Figure 5. Such examples help to show that a proposed algorithm does perform optimally at least in some scenarios.

As before, the green hexagons represent relay locations and the black dots represent source nodes. Suppose $h_{max} = 2$. Then clearly, the optimal solution is as shown in the figure. The algorithm, after calculating the SPT, will end up with the same solution.

Note that since RST-MR-HC is a special case of RST-MR-DC problem, the above sharp examples also hold for the algorithm applied to the RST-MR-DC problem.

IV. Numerical Results

To test the algorithm, we generated 620 random networks as follows:

A $200m \times 200m$ area is partitioned into square cells of side $10m$. Consider the lattice created by the corner points of the cells. 10 source nodes are placed at random over these lattice points. Then the potential relay locations are obtained by selecting 60 points uniformly randomly over the $200m \times 200m$ region.

Given the outcome of the SPT based algorithm, the optimal solution can be obtained as follows:

Suppose the SPT based algorithm uses n relays. Then perform an exhaustive search over all possible combinations of $(n-1)$ and fewer relays to check if the performance constraints can still be met.

In 109 of the 620 scenarios tested, the delay constraint turned out to be infeasible.The results for the remaining 511 scenarios are summarized in Table II.

TABLE II
Efficiency of the SPT based Algorithm in obtaining the Optimal Design

Scenarios	Optimal Design	Off by one	Off by 2 to 4	Off by 5 or more
511	320	141	50	0

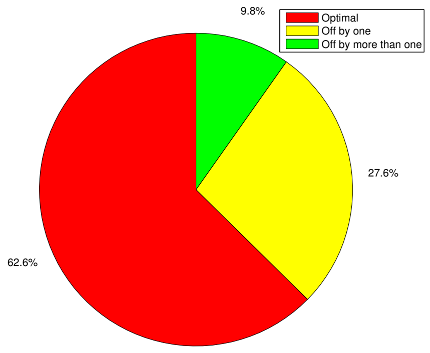

Fig. 6. Efficiency of the Algorithm as suggested by Test Results

The efficiency of the algorithm can be easily visualized from the pie chart in Figure 6.

Observations

1) In over 90% of the tested scenarios, the algorithm ends up giving optimal or near-optimal (exceeding optimum just by one relay) solutions.
2) In the remaining cases, where it is off by more than one relay, the maximum difference was found to be 4 relays.

TABLE III
COMPARISON OF SPT BASED ALGORITHM WITH OPTIMUM DESIGN FOR RST-MR-DC PROBLEM ON 12 RANDOM SCENARIOS

Scenario	d_{max} in $msec$	Relay Count Algorithm	Optimum	Execution Time in sec Algorithm	Optimum
1	22	4	4	0.655	1371.1
2	22	2	1	0.593	0.639
3	22	1	1	0.593	0.078
4	22	2	2	0.577	2.449
5	22	2	2	0.375	2.481
6	22	4	4	0.686	1379.5
7	22	3	3	0.78	70.902
8	22	4	3	0.796	369.22
9	22	2	2	0.889	2.481
10	22	3	3	0.687	70.419
11	22	2	2	0.718	2.496
12	22	3	3	0.905	70.715

In Table III, we take a closer look at the outcome of the SPT based algorithm applied to the RST-MR-DC problem on 12 random networks. As can be seen from the table, in all those 12 cases, the algorithm gives optimal or near optimal solutions within hundreds of milliseconds. Note from the table that even after knowing the outcome (say n) of the algorithm, computation of the optimum by exhaustive search over $(n-1)$ or fewer relays took as long as 23 mins in the worst case (scenarios 1 and 6), as the search was over all possible combinations of $(4-1) = 3$ relays out of 60, i.e., a total of 34220 combinations.

TABLE IV
COMPARISON OF SPT BASED ALGORITHM WITH OPTIMUM DESIGN FOR RST-MR-HC PROBLEM ON 12 RANDOM NETWORKS

Scenario	h_{max}	r_c in meters	Relay Count Algorithm	Optimum
1	6	64	5	5
2	6	64	1	1
3	6	64	3	2
4	6	64	3	3
5	6	70	2	2
6	6	64	3	3
7	6	64	3	3
8	6	64	3	3
9	8	70	2	2
10	6	64	3	3
11	6	64	4	3
12	8	70	2	2

In Table IV, we provide a snapshot of the application of the algorithm to the RST-MR-HC problem on 12 random networks. Here also, we find that in all the 12 cases, the algorithm gives optimal or near optimal solutions.

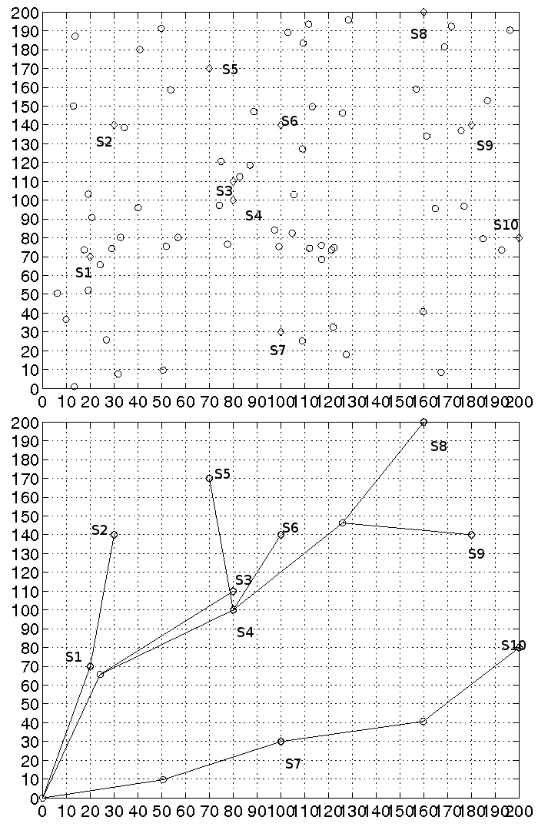

Fig. 7. RST-MR-DC problem on a random scenario where our algorithm gives optimal design; Top panel: Source node locations and potential relay locations. Bottom panel: The relay placements and paths obtained by the algorithm; circles indicate relay locations

We show in Figure 7 the node placement and the paths obtained by the algorithm for the RST-MR-DC problem on a random scenario for which the algorithm gives optimal design. In Figure 8, we show the node placement and the paths obtained by the algorithm for RST-MR-DC problem on a random scenario for which the algorithm gives near optimal (off by one) design. In these figures, the diamonds indicate source locations and the circles indicate relay locations. Note that if we are simply given the positions of the sources and potential relays, it is not at all clear by inspection how to select the relays to get an optimal design. A naive way to determine an optimal design is to perform an exhaustive search, which would require consideration of 2^{60} combinations. If each case took even 1 msec, it would take 2^{60} msec $= 3.2 \times 10^{11}$ hours $= 1.33 \times 10^{10}$ days $= 36558901.08$ years. This underlines the usefulness of our algorithm in giving optimal or near optimal solutions within an extremely reasonable time.

In Figure 9, we show the node placements and the paths obtained by the algorithm for the RST-MR-DC

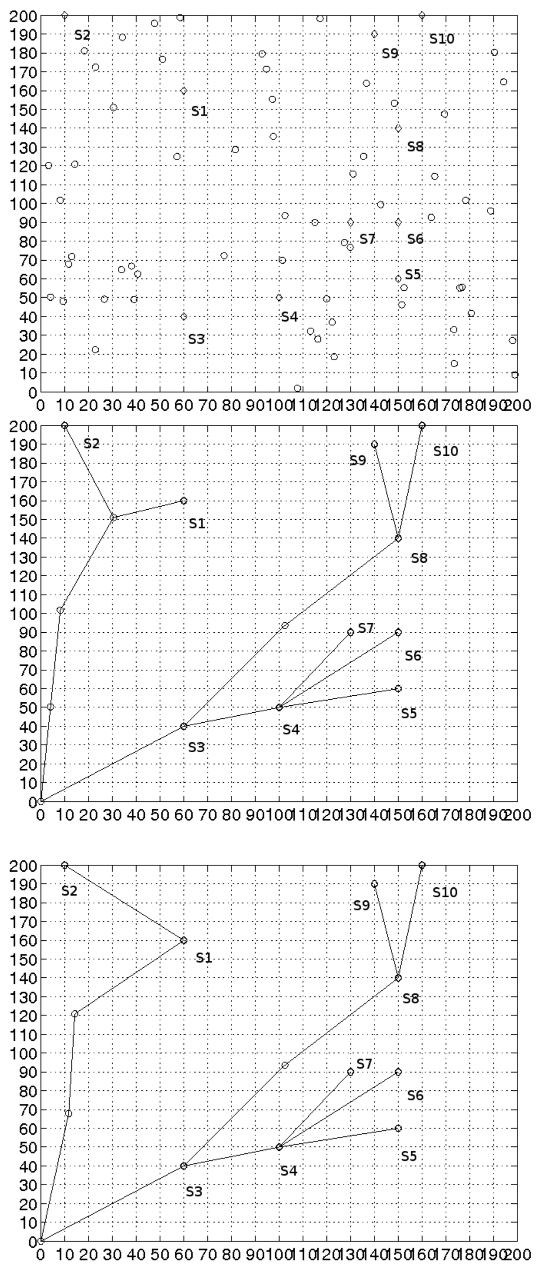

Fig. 8. RST-MR-DC problem on a random scenario where algorithm gives near optimal design; Top panel: Source node locations and potential relay locations. Middle panel: The relay placements and paths obtained by the algorithm; circles indicate relay locations. Bottom panel: An Optimal design

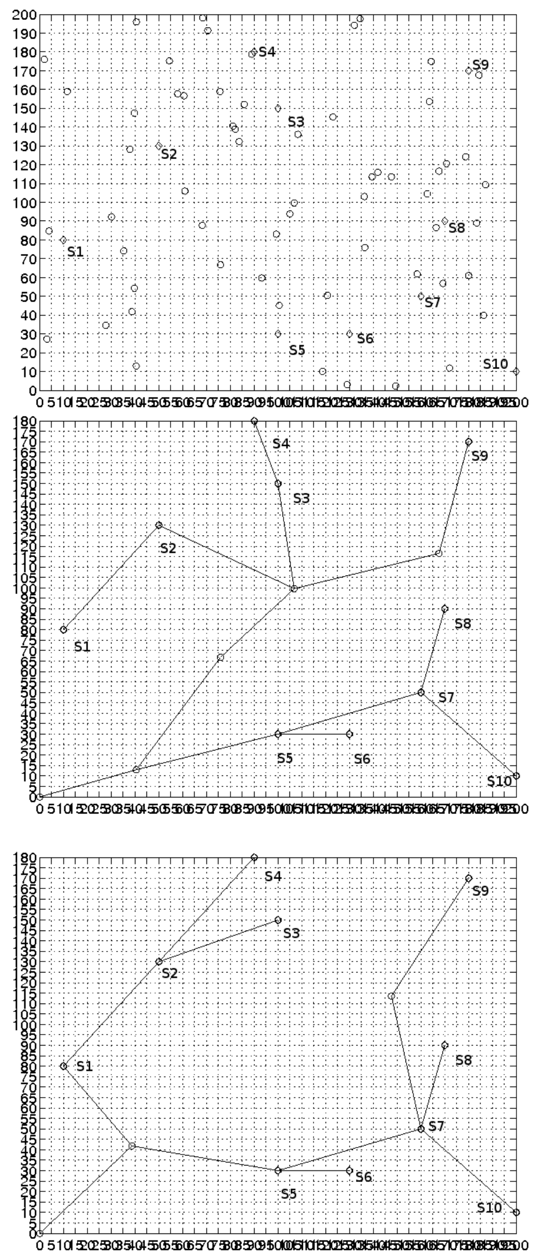

Fig. 9. RST-MR-DC problem on a random scenario where algorithm is off by more than one (2) from optimal design; Top panel: Source node locations and potential relay locations. Middle panel: The relay placements and paths obtained by the algorithm; circles indicate relay locations (using 4 relays). Bottom Panel: An Optimal design (using 2 relays)

problem on a random scenario where the algorithm is off from the optimal by 2.

Finally, in Figure 10, we present the relay placements and paths obtained by the algorithm for the RST-MR-HC problem on a random scenario where the algorithm achieves optimal design.

V. Conclusion

In this paper, we have studied the problem of determining an optimal relay node placement strategy such that a certain performance objective (in this case, mean delay) is met. We showed that the problem is NP-Hard, and proposed a polynomial time approximation

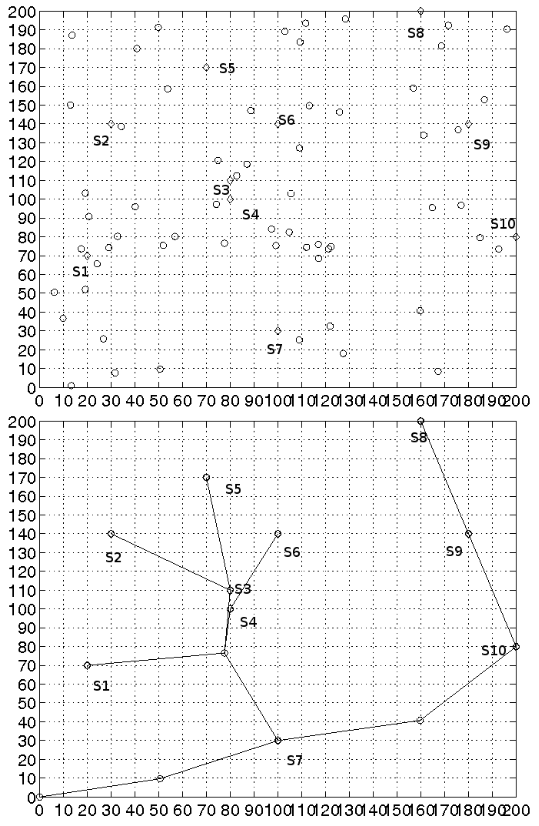

Fig. 10. RST-MR-HC problem on a random scenario where algorithm achieves optimal design; Top panel: Source node locations and potential relay locations. Bottom panel: The relay placements and paths obtained by the algorithm; circles indicate relay locations

algorithm, which, as can be concluded from numerical experiments, gives solutions of reasonably good quality, using extremely reasonable computation time. In the worst case, the algorithm can do as bad as using all but one potential relay locations instead of just one optimum relay. The algorithm can not do infinitely worse, in the sense that if the optimum design happens to use no relay node, the algorithm does not use any relay node either.

The algorithm proposed here is basically a local search algorithm where the starting solution is an SPT, and then we search neighbourhoods of that SPT until a local optimum is obtained. One might ask why this local search algorithm works so well in the tested random scenarios. The answer to this question is not immediately obvious, but, for the RST-MR-HC problem, a formal analysis of the properties of the underlying random geometric graph might provide some useful insights into the performance of this local search algorithm. We wish to address this issue in our future work.

In this paper, we worked with a mean delay objective. In the future, we wish to extend this to cover a probabilistic worst case delay guarantee, i.e., $P(delay \geq T) \leq \epsilon$. Also, so far we have only sought a tree with certain properties. But for fault-tolerance purposes, it is essential to seek an optimal relay placement to ensure k-connectivity, while still meeting the delay requirement. Further, we are working on extending the design to traffic models more complex than the lone packet traffic model considered here. This requires the analysis of packet delays in a mesh network with more complex traffic flows and the nodes accessing the medium using CSMA/CA as defined in IEEE 802.15.4 [18].

Acknowledgement

This work was supported by the Automation Systems Technology (ASTeC) Center, a program of the Department of Information Technology at CDAC, Trivandrum, Kerala, India. We thank Dr. Navin Kashyap for pointing out an error in an earlier version of this paper.

References

[1] "www.honeywell.com/ps/wireless."
[2] "www.isa.org/isa100."
[3] R. A. Sajana, R. Subramanian, P. V. Kumar, S. Krishnan, B. Amrutur, J. Sebastian, M. Hegde, and S. Anand, "A low-complexity algorithm for intrusion detection using a pir-based wireless sensor network," in *International Conference Series on Intelligent Sensors, Sensor Networks and Information Processing*, (Sydney), 2009.
[4] "en.wikipedia.org/wiki/steiner tree problem."
[5] F. K. Hwang, D. S. Richards, and P. Winter, *The Steiner Tree Problem*, vol. 53 of *Annals of Discrete Mathematics*. Elsevier Science Publishers B.V, 1992.
[6] V. V. Vazirani, *Approximation Algorithms*. Springer, 1st ed., 2001.
[7] G.-H. Lin and G. Xue, "Steiner tree problem with minimum number of Steiner points and bounded edge length," *Information Processing Letters*, vol. 69, pp. 53–57, 1999.
[8] X. Cheng, D.-Z. Du, L. Wang, and B. Xu, "Relay sensor placement in wireless sensor networks," *Wireless Netw*, vol. 14, pp. 347–355, 2008.
[9] E. L. Lloyd and G. Xue, "Relay node placement in wireless sensor networks," *IEEE Transactions on Computers*, vol. 56, January 2007.
[10] S. Voss, "The Steiner tree problem with hop constraints," *Annals of Operations Research*, vol. 86, pp. 321–345, 1999.
[11] A. M. Costa, J.-F. Cordeau, and G. Laporte, "Fast heuristics for the steiner tree problem with revenues, budget and hop constraints," *European Journal of Operational Research*, vol. 190, pp. 68–78, 2008.
[12] M. Kim, Y.-C. Bang, and H. Choo, "On multicasting Steiner trees for delay and delay variation constraints," 2006.
[13] J. L. Bredin, E. D. Demaine, M. T. Hajiaghayi, and D. Rus, "Deploying Sensor Networks with Guaranteed Capacity and Fault Tolerance," in *MobiHoc'05*, ACM, 2005.
[14] S. Misra, S. D. Hong, G. Xue, and J. Tang, "Constrained Relay Node Placement in Wireless Sensor Networks to Meet Connectivity and Survivability Requirements," in *IEEE INFOCOM*, 2008.
[15] I. C. Society, *IEEE Standards Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs)*. New York, October 2003.
[16] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Bell Telephone Laboratories, Inc., 1979.
[17] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*. The MIT Press and McGraw-Hill, 2 ed., 2001.
[18] C. K. Singh, A. Kumar, and P. M. Ameer, "Performance Evaluation of an IEEE 802.15.4 Sensor Network with a Star Topology," *Wireless Network*, vol. 14, pp. 543–568, 2008.