Cluster Computing 8, 127–133, 2005

Computation of Minimal Uniform Transmission Range in Ad Hoc Wireless Networks

QING DAI and JIE WU *
Department of Computer Science and Engineering, Florida Atlantic University, Boca Raton, FL 33431

Abstract. Power conservation is a critical issue for ad hoc wireless networks. The main objective of the paper is to find the minimum uniform transmission range of an ad hoc wireless network, where each node uses the same transmission power, while maintaining network connectivity. Three different algorithms, Prim's Minimum Spanning Tree (MST), its extension with Fibonacci heap implementation, and an area-based binary search are developed to solve the problem. Their performance is compared by simulation study together with Kruskal's MST, a known solution proposed by Ramanathan and Rosales-Hain for topology control by transmission power adjustment, and an edge-based binary search used by the same study in order to find the per-node-minimality after Kruskal's algorithm is applied. Our results show that Prim's MST outperforms both Kruskal's MST and the two binary searches. The performance between Prim's MST implemented with binary heap and Fibonacci heap is fairly close, with the Fibonacci implementation slightly outperforming the other.[1]

Keywords: ad hoc wireless network, graph connectivity, minimum spanning tree, power control, transmission power

1. Introduction

An ad hoc wireless network is an infrastructureless network. The end nodes establish connections by themselves without a base station, and communicate with each other in a multi-hop manner. In this environment, each node, typically a mobile computing device, is powered by battery. Example applications include personal area networks, wireless sensor networks, military, law enforcement, disaster recovery, etc [4,6,13]. The limit of battery life places a constraint on the power consumption. It is desirable for routing algorithms to select route and transmission power that optimize energy efficiency whenever possible. Researchers have proposed different approaches to conserve power in routing [1,11,12,16]. Meanwhile, the ad hoc wireless network is highly dynamic and autonomous, and algorithms that can operate in a distributed fashion is therefore desirable. Some interesting works are done on distributed algorithms and protocols [5,7,10,14,15].

In the routing of the ad hoc wireless network, it is believed that the minimum transmission power required to sustain a link between two nodes is

$$P_{ij} = r^{\alpha},$$

where r is the distance between node i and node j, and α is between 2 and 4. The transmission power cannot be reduced without limit, since the transmission range of a node will be shortened along with the reduction of transmission power. When the transmission power is too low, the network could suffer from partition.

Ad hoc wireless networks are typically modelled with the transmission graph [8], a graph $G = (V, E)$ in Euclidean space, where there exists an edge from node u to v, if u can directly transmit to v. The topology of an ad hoc network depends on several uncontrollable factors such as host mobility and interference, and on several controllable ones such as antenna orientation and transmission power. This paper deals with adjusting the controllable parameters, for example, the transmission power, to create desired topology to meet certain criteria, specifically, the network connectivity.

Among the alternative metrics to optimize transmission power, this paper addresses optimizing power of a single node. We believe this is more practical, because if to obtain a lower overall power, certain nodes must transmit at an extra high power level, this situation might not be acceptable for all cases. Moreover, it is assumed that each node uses the identical transmission power and therefore reaches the same transmission range. In other words, each node does not have the ability to dynamically adjust its transmission power for different destinations. The unit disk graph $G(V, E)$ is typically used to model ad hoc wireless networks under this situation, where two nodes are connected when their distance is within this fixed transmission range. Throughout this paper, we consider ad hoc wireless networks in which the node locations are fixed, or are snapshots of the network at a particular time frame. The focus is to develop algorithms that find the minimum uniform transmission range that keeps the network connected, assuming global information of node location, link state information, and fixed network topology.

To find this minimum uniform transmission range, we explored five different algorithms, area-based binary search, Prim's minimum spanning tree (MST), its extension with Fibonacci heap implementation, and the two well-known solutions by Ramanathan and Rosales-Hain [9], one based on Kruskal's MST, the other an edge-based binary search (used to find the per node minimality). A network of n nodes, which are

* Corresponding author.
E-mail: jie@cse.fau.edu

randomly distributed over a region of size $\ell \times \ell$, is generated in performance study. Each node has multiple transceivers, and can thus support any multi-cast sessions within its transmission range. A graph is constructed according to the node location and the transmission range information; that is, an edge between nodes u and v is added to the graph if u and v are within the transmission range of each other. The execution time of different algorithms to find the *minRange* is compared. Our simulation results show that Prim's MST and its extension outperform Kruskal's MST and the two binary searches under the given condition.

This paper is organized as follows: Section 2 proposes our three algorithms, Algorithm II, III and IV. Kruskal's MST and edge-based binary search by Ramanthan and Rosales-Hain are also reviewed as Algorithm I and V. Section 3 shows simulation results. Section 4 concludes this paper and discusses some future work.

2. Algorithms

As discussed above, our objective is to minimize the uniform transmission range while maintaining the network connectivity at the same time. Three new algorithms are proposed below to solve the problem: Algorithms II and III are Prim's MST with either binary heap or Fibonacci heap implementation, Algorithm IV uses the area-based binary search technique. Ramanathan and Rosales-Hain's algorithms based on Kruskal's MST as well as their edge-based binary search used to achieve per-node-minimality are also reviewed, presented as Algorithm I and V.

The notations used in this section are as follows: in an area of size $\ell \times \ell$, the number of nodes is noted as n. D is the largest possible distance between any two nodes in this area, or Diameter, which is $\sqrt{2}\,\ell$. In the undirected graph representation of the area, V represents the number of vertices, which is equal to n, and E is the number of edges. An edge connecting nodes u and v is noted as (u, v).

2.1. Algorithm I: Kruskal's minimum spanning tree (MST)

Algorithm I is proposed by Ramanathan and Rosales-Hain [9]. An MST was constructed using Kruskal's algorithm [2]. Briefly, each node is initialized as a separate connected component. Edges are sorted first, and then traversed in a non-decreasing order. An edge is added to the MST whenever it connects two connected components, until all nodes are included in a single connected component. The last edge added to MST, which is also the largest edge in Kruskal's MST, will be the minimum uniform transmission range of the network (see figure 1). The proof of correctness for the algorithm was provided in the same paper [9].

The construction of MST takes $O(E \lg E)$, which will be $O(V^2 \lg V^2)$ for a complete graph. Therefore, the asymptotical complexity of Algorithm I is $O(V^2 \lg V)$.

```
MinRange-Kruskal
1  minRange ← 0
2  sort all edges in non-decreasing order
3  initialize n clusters, one per node
4  while numberOfClusters > 1
5    for each (u, v) in the sorted order
6      do if cluster(u) ≠ cluster(v)
7           merge cluster(u) with cluster(v)
8           minRange ← distance(u, v)
9  return minRange
```

Figure 1. Algorithm I: compute *minRange* by Kruskal's MST.

In this algorithm, all edges in the graph are sorted first, which costs $O(E \lg E)$. In reality, the transmission range can be determined without going through all the edges (in the sorted order). Therefore, some efforts in the sorting process could be wasted.

2.2. Algorithms II and III: Prim's MST

Algorithms II and III are modifications of Algorithm I, both involving MST construction. In both Algorithms II and III, a modified Prim's algorithm [2] is used in building the MST (see figure 2). Prim's MST starts with an arbitrary root and constructs a single tree, until it spans all the vertices. At each step, an edge of the smallest possible distance that reaches a non-tree node is added, and at any stage, all nodes in Prim's MST forms a single tree. To facilitate the MST construction process, a key for each node is maintained to represent its distance to the tree, and whenever a new node is included, the key value of its direct non-tree neighbors are updated. After the MST is constructed, the tree is traversed and the maximum edge is the minimum uniform transmission range. The correctness of the algorithm is proved in two different methods below.

It is interesting to note that the traditional minimum spanning tree algorithms can be applied to different problems. The original MST algorithms minimize the the total weight of the tree. In our problem, the objective is to minimize a single uniform transmission range that keeps the network connected. In another word, it is the maximum edge in the spanning tree that is minimized. The following theorem shows that the largest

```
MinRange-Prim
1   unreachedNodes ← V[G]
2   for each u ∈ unreachedNodes
3      do key[u] ← ∞
4   key[root] ← 0
5   while unreachedNodes ≠ ∅
6      do u ← extractMin(unreachedNodes)
7         for each v ∈ Adj[u]
8            do if v ∈ unreachedNodes and
                   distance(u, v) < key[v]
9               then key[v] ← distance(u, v)
10  traverse the MST, find the longest edge minRange
11  return minRange.
```

Figure 2. Algorithm II: Prim's minimum spanning tree.

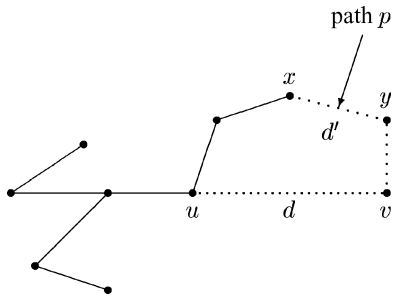

Figure 3. Proof of Theorem 1.

edge selected in Prim's MST algorithm is the minimum uniform transmission range.

Theorem 1. The maximum edge in Prim's MST is the minimum uniform transmission range.

Proof 1. Suppose the longest edge in Prim's MST is (u, v) with $d = \text{distance}(u, v)$ (see figure 3). And suppose to the contrary that d is not minimum, then there must exist a path p to connect u and v, and on this path, each of the edges is less than d. By the time (u, v) is being added to change the MST from T_i to $T_{i+1} = T_i \cup \{(u, v)\}$, assume vertex u is in the MST, and v is not, without loss of generality. If v is chosen to be included to the MST, then it must have the smallest key among all vertices currently not in the tree. Meanwhile, if path p exists, then there must exist an edge (x, y) with $d' = \text{distance}(x, y)$ on path p, where x is in the tree, and y is not. (In extreme cases, x could be u, or y could be v, but not both). d' is less than d, because on path p, each edge is less than d. Therefore, y should have a smaller key than v. According to Prim's algorithm, y should be added to the MST instead of v. This is contrary to the result of Prim's algorithm, where v is added. Therefore, this path p, in which every edge is smaller than d, does not exist, and d is the minimum. □

The same theorem can be proved in another way. For all the MSTs, if all the edges of a connected graph have different weights, it has been proved that the MST is unique [3]. In that case, the largest edge in Kruskal's and Prim's MST is exactly the same edge. Since the edge weights might not always be distinct, we define *edge weight set* of a tree to be a set that contains all the weights of tree edges, with each element in this set corresponding to the weight of an individual edge. We can prove that the edge weight set for any MST is the same, that is, the edge weights for all MSTs is a unique set. Note that there can be duplicates of edge weights within one MST, as well as the MST edge weight set. For example, an MST edge weight set A of MST_A is $\{3, 4, 4, 7, 9, 11, 11, 31\}$ and contains two edges of weight 4 and two of weight 11. Also, each element in an edge weight set is treated only as number, without any edge identity information or meaning. If another edge weight set B of an MST_B is $\{3, 9, 11, 4, 7, 31, 11, 4\}$, we say it is identical to set A, even though the edge with the weight 9 in MST_A might not be the same edge as the edge of weight 9 in MST_B.

First, we know that for a graph of n nodes, all MSTs have $n - 1$ edges. Therefore, all its MST edge weight sets have $n - 1$ elements. Now we prove below that this set of numbers is unique for a given graph.

Lemma 1. In any two different MSTs over the same graph, MST_0 and MST_1, for each edge e that is in MST_0 but not in MST_1, we can find an edge e' that is in MST_1 but not in MST_0 such that $w(e) = w(e')$, where $w(e)$ represents the weight of edge e.

Proof 2. Let e be any edge that is in MST_0 but not in MST_1. The edge set $\{e\} \cup MST_1$ must contain a cycle, and at least one edge e' of this cycle is not in MST_0, since MST_0 can not contain any cycle. $MST_1 - \{e'\} \cup \{e\}$ is a spanning tree, because it is a tree of $n - 1$ edges. Since MST_1 is an MST, we have $w(e) \geq w(e')$. Similarly, $MST_0 - \{e\} \cup \{e'\}$ is also a spanning tree. With MST_0 being a MST, we have $w(e') \geq w(e)$. Therefore, we have $w(e') = w(e)$. □

Theorem 2. The edge weight set of any MST for a certain graph is unique.

Proof 3. Because for any edge e that is in MST_0 but not in MST_1, we can always find a corresponding equal-weight edge e' in MST_1 but not in MST_0, we replace e' with e in MST_1 to form MST_2, which is still an MST. The number of different edges between MST_2 and MST_0 is one less than that of between MST_1 and MST_0. Repeat the above process recursively. For each replacement that changes MST_i to MST_{i+1}, the number of different edges will decrease by 1. During the replacement, the edge weight set never changes. When there is no different edge between MST_0 and MST_n, the replacement procedure terminates. The two MSTs are now identical trees. The edge weight set of all these MSTs including $MST_0, MST_1, \ldots, MST_i, \ldots, MST_n$, are the same. Since MST_0 and MST_1 are any two different MSTs, we can conclude that all MSTs have the same edge weight set. □

Corallary 1 The maximum-weight edges in Kruskal's MST and Prim's MST have the same weight.

Corallary 1 is obvious from Theorem 2. Because the edge weight set for MSTs of a certain graph is unique, as one of the edges in MST, the weight of the maximum edge is unique, i.e., the weight of the maximum edge for any MST of the same graph is equal.

We can now prove the correctness of Theorem 2 easily.

Proof 2 of Theorem 1. It has been proved that the maximum edge in Kruskal's MST is the optimum maximum edge that keeps a network connected [9]. Moreover, from Corallary 1, the weight of the maximum edge in Prim's MST equal to that of the maximum edge in Kruskal's MST. Therefore, the maximum edge in Prim's MST is the minimum uniform

transmission range. This completes the second proof for Theorem 1. □

Between the two proofs of Theorem 1 above, Proof 2 combining Corallary 1 and the previous proof by Ramanathon and Rosales-Hain is more general for the problem, because it is independent of the MST algorithm used. As a result, for a given graph, the maximum edge of any MST, including MST obtained by distributed algorithms, has the same weight. For an ad hoc wireless network instance, the distance of this maximum edge is the minimum uniform transmission range of the network.

Algorithm II uses a binary-heap implementation, with a complexity of $O(E \lg V)$, which will be $O(V^2 \lg V)$ in the worst case, and is asymptotically the same as that of Algorithm I. Because Prim's algorithm has a Fibonacci heap implementation, which can speed up to run in $O(E + V \lg V)$, or $O(V^2)$ for a complete graph [2], Algorithm III takes advantage of this property. It is the same algorithm as Algorithm II, but uses the Fibonacci heap implementation. Fibonacci heap is a data structure based on binomial heaps. It is more relaxed so that work to maintain the structure can be delayed until it is convenient to perform, therefore allowing for improved asymptotic bounds [2]. Both Algorithms II and III are implemented and their performance is compared through simulation study.

2.3. Algorithm IV: Area-based binary search

Algorithm IV adopts a simple brute force approach. Ranging from 0 to the longest possible transmission range, the diameter D of the area, we use binary search to find the lowest transmission range that keeps the network connected. For each *curRange* tested, a unit disk graph is generated, where an edge (u, v) is added to the graph if the distance between u and v is less than *curRange*. The value of *curRange* is then either decreased or increased depending on whether the resultant unit disk graph is connected or not (see figure 4). We name it the area-based binary search, or *area-binary* in brief.

In this algorithm, it is important to decide when the binary search is completed, because theoretically, the *curRange* adjustment could continue forever. We decide to use integer distance, and terminate the searching process whenever

```
MinRange-AreaBinary
1  min ← 0, max ← √2 ℓ
2  while min < max − 1
3     curRange ← ⌊(min + max)/2⌋
4     G ← generateUnitDiskGraph(curRange)
5     if G is connected
6        max ← curRange
7     else
8        min ← curRange
9  return minRange ← max
```

Figure 4. Algorithm IV: compute *minRange* by area-based binary search.

```
MinRange-EdgeBinary
1  numNodes ← n, numEdges ← n ∗ (n − 1)
2  Sort all edges in edges[numEdges]
3  minIndex ← 0, maxIndex ← numEdges − 1
4  while minIndex < maxIndex − 1
5     curIndex ← ⌊(minIndex + maxIndex)/2⌋
6     G ← generateUnitDiskGraph(edges[curIndex])
7     if G is connected
8        maxIndex ← curIndex
9     else
10       minIndex ← curIndex
11 G ← generateUnitDiskGraph(edges[minIndex])
12 if G is connected
13    maxIndex ← minIndex
14 return edges[maxIndex]
```

Figure 5. Algorithm V: compute *minRange* by edge-based binary search.

the upper and lower boundary differs by less than 1 (max – min ≤ 1). This generally gives good performance in practice. Though the *minRange* obtained should be fairly close, it might not be the 'ultimate' minimum transmission range value, since the computation depends on the granularity of D. Note that the precise *minRange* can be obtained with a slight modification. We can sort all edges by distance first, then perform binary search on the array of edge weights. This modification achieves precision at the cost of higher complexity, since the cost of sorting is $O(E \lg E)$. It was used in [9] to obtain per-node-minimality. We review it later as Algorithm V (see figure 5).

Some optimizations could be done to improve the performance of the area-binary algorithm. For example, we can check whether there is more than one edge between the min and max (min end exclusive, max end inclusive) for potential earlier termination. If there is only one edge in between, we can determine that the weight of this single edge will be the minimum transmission power, since we know that the ultimate minimum uniform transmission range must equal to one of the edges, i.e., the maximum edge of the MST. Another possible optimization is, we can traverse the edges to identify the *minEdge* and the *maxEdge* first, and use them as the initial values of min and max, so as to decrease the range of the binary search.

The complexity of the Area-based Binary approach can be calculated as follows: each connectivity checking costs $O(V + E)$, with either depth-first search or breadth-first search. The overall complexity of the algorithm is $O((V + E) \lg D)$, or $O(V^2 \lg D)$ at the most. $\lg D$ indicates how many connectivity checks are needed in the worst case, which depends on the area size. For an area of fixed size, $\lg D$ can be treated as a constant.

2.4. Algorithm V: Edge-based binary search

We review the binary search based algorithm used in [9] to compute the per-node-minimality here as Algorithm V. In this algorithm, all edges are sorted first, and binary search is performed upon all edges of the graph. We name this algorithm

edge-binary, to distinguish it from the previous area-binary. Comparing to the area-binary, it might perform better when area size is large and node number is very small, i.e., when node distribution is sparse, because the number of connectivity check required until the *minRange* is determined could be reduced.

One possible optimization for this algorithm is, after all the edges are sorted, we could remove the duplicates, then perform the binary search only on the remaining unique edge weights, so that we could avoid repeating connectivity checks with the same test range. This optimization is not currently implemented in our simulation study.

The complexity of this approach can be calculated as follows: it costs $O(V^2 \lg V)$ to sort all the edges. Each connectivity check costs $O(V + E)$, the same as in area-binary. The number of connectivity checks is $O(\lg E)$. Therefore, the overall complexity is $O((V + E) \lg E)$, or $O(V^2 \lg V)$ for the complete graph. When the node number grows fast, and the area size is relatively stable, the complexity of edge-binary will be higher than the area-binary.

3. Simulation study

A network of n nodes, which are randomly distributed over an area of size $\ell \times \ell$, is generated. We compute the minimum transmission range using all five different algorithms described in the previous section, and record their execution time. The first set of simulation uses the set of node numbers from $n = 20$ to 800, but in different sizes of area, $l = 1000$, 4000 and 16000, respectively.

Our results show that both Kruskal's and Prim's MST outperform the two binary search algorithms (see figure 6). Prim's MST has a better performance than Kruskal's MST. Between the two Prim's implementations, the one with Fibonacci heap slightly outperforms the binary-heap. Between the two binary search techniques, the area-binary performs better than the edge-binary.

Similar results are observed with the same number of nodes in area of different size, 1000×1000, 4000×4000 and 16000×16000.

To further study the effect of node number on the performance, simulation with larger numbers of nodes, up to 2400, is performed (see figure 7). Due to the large number of nodes, bigger areas of 8000×8000 and 16000×16000 are used. In this simulation, Prim's MST still outperforms both Kruskal's MST and the two binary searches.

In the future, it would be interesting to see whether binary search could outperform Kruskal's MST (Algorithm I), or even Prim's MST with binary heap implementation (Algorithm II). Theoretically, the complexity of area-binary is $O(V + E) \lg D$, which is close to $O(V^2)$ if D is considered constant, in contrast to the complexity of edge-binary and Kruskal's MST, which are both $O(V^2 \lg V)$ in the worst case. When V is small and D is large, the effect of $\lg D$ is greater than $\lg V$. But when V is extremely large and D is relatively small, area-binary should eventually outperform Kruskal's MST. Although it is possible that at this point, the V might be too large to have any practical value.

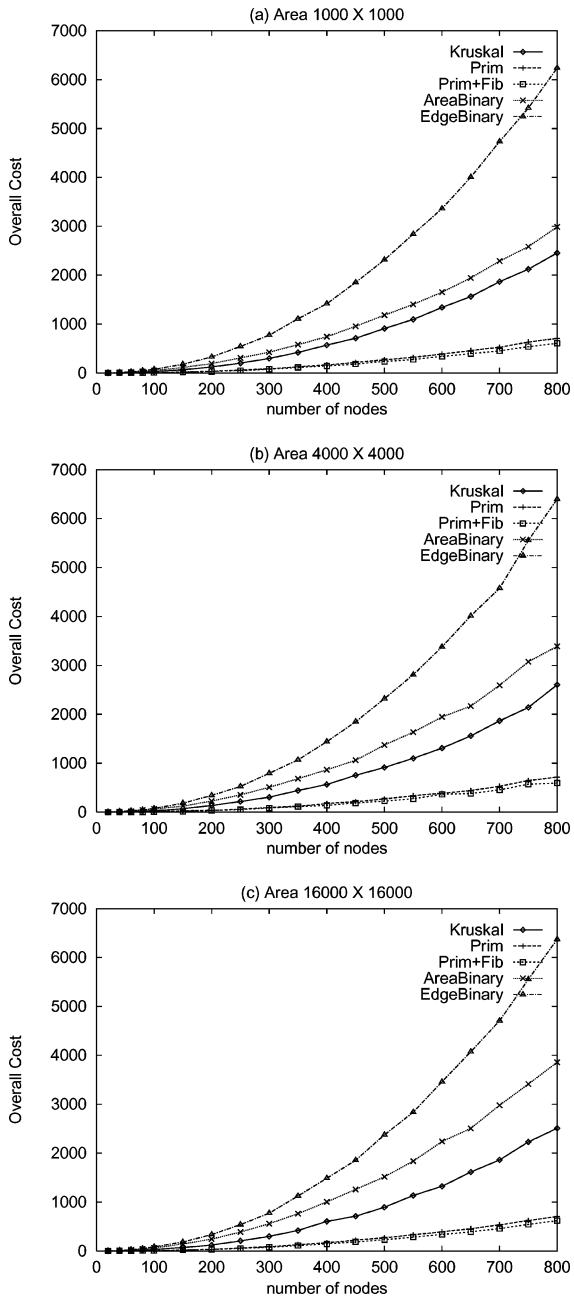

Figure 6. Simulation results. Number of nodes from 20 to 800, in an area of 1000×1000 (a), 4000×4000 (b), and 16000×16000 (c).

The above simulation studies provide us with the basic understanding of the relative performance among different algorithms to find the minimum uniform transmission range in ad hoc wireless networks. Their practical performance versus theoretical complexity also helps us in making decisions such as which algorithm to choose under a specific network environment with certain node number and area size.

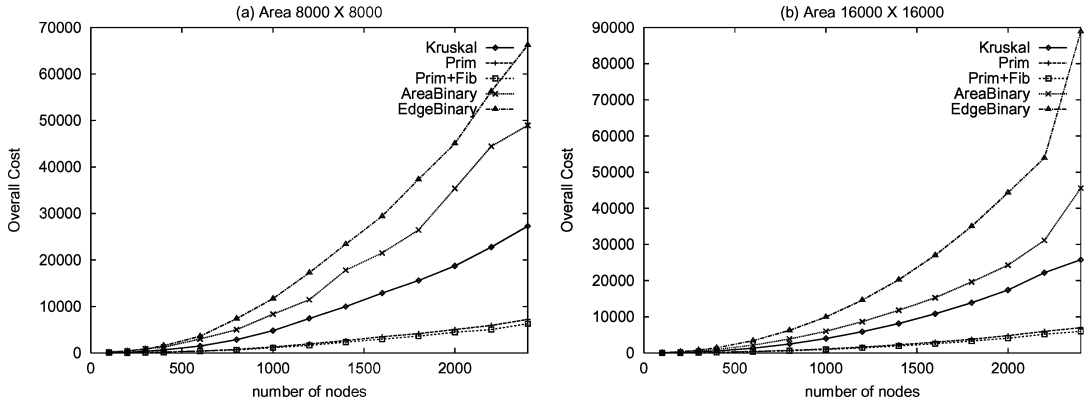

Figure 7. Simulation results. Number of nodes from 100 to 2400, distributed in 8000 × 8000 (a) and 16000 × 16000 area (b).

4. Discussion and conclusion

In this paper, minimum spanning tree algorithms from classic graph theory have been applied to solve the energy efficient routing problem in ad hoc wireless networks of finding the minimum uniform transmission range. We proposed three new algorithms, Prim's MST, Prim's MST with Fibonacci heap implementation, and the area-binary. We proved the correctness of using Prim's MST to determine the minimum uniform transmission range. The performances of the algorithms are compared with the two previously proposed ones, Kruskal's MST and the edge-binary.

From our simulation result, it seems that for node number up to 2400, Prim's MST has better performance than both Kruskal's MST and the two binary searches. The performance between the two Prim's implementations is fairly close, with the Fibonacci heap implementation slightly outperforming the binary-heap implementation. Between the two binary search based algorithms, area-binary shows better performance than the edge-binary.

A challenge would be to find distributed algorithms to solve the same problem. LINT and LILT introduced in [9] are helpful examples of finding the optimum transmission range in a distributed manner using local information and some available global topology information. In a previous research, a distributed algorithm has been proposed to construct a minimum spanning tree [3], by joining of MST nodes and fragments through the minimum-weight outgoing edge of different fragments. The same algorithm could be modified and applied to our problem of finding the minimum uniform transmission range, if each fragment saves the maximum weight of all its branches. Comparison of *maxBranchWeight* between joining fragments takes place at the time of their connection. The updated *maxBranchWeight* information is then broadcast again to all nodes of the new fragment at the start of a new cycle. When no node has outgoing edges, the fragment is now the MST, and the *maxBranchWeight* is the minimum uniform transmission range. The correctness of the algorithm to solve our problem can be proved similarly by applying the second proof for Theorem 1, since the maximum edges of any MST for a graph all have the same weight, and is consequently the minimum uniform transmission range for the network instance.

In our future study, first we plan to further increase the number of nodes, to examine the performance of different algorithms. Meanwhile, mobility and directional antenna will be taken into consideration in the future model. We would also like to further study and implement the distributed algorithm to compute the minimum uniform transmission range.

Note

1. The work was supported in part by a grant from Motorola Inc., and NSF grants CCR 9900646, ANI 10073737, EIA 0130806, and CCR 0329741.

References

[1] J.-H. Chang and L. Tassiulas, Energy conserving routing in wireless ad-hoc networks, in: *Proceedings of IEEE INFOCOM'00* Tel Aviv, Israel (Mar. 2000) pp. 22–31.

[2] T.H. Cormen, C.E. Leiserson and R.L. Rivest, *Introduction to Algorithms* (McGraw-Hill, 1990).

[3] R.G. Gallager, P.A. Humblet and P.M. Spira, A distributed algorithm for minimum-weight spanning trees, ACM Transactions on Programming Languages and Systems 5(1) (Jan 1983) 66–77.

[4] J. Gomez, A.T. Campbell, M. Naghshineh and C. Bisdikian, Power-aware routing in wireless packet networks, *IEEE INFOCOM'99* (1999) pp. 380–383.

[5] L. Li, J.Y. Halpern, P.B., Y.-M. Wang and R. Wattenhofer, A cone-based distributed topology-control algorithm for wireless multi-hop networks, in: *Proceedings of ACM Principles of Distributed Computing (PODC)* (August 2001) pp. 264–273.

[6] Q. Li, J. Aslam and D. Rus. Distributed energy-conserving routing protocols for sensor networks. (June 2002).

[7] T.H. Meng and V. Rodoplu, Distributed network protocols for wireless communication, in: *Proceedings of IEEE ISCAS'98*, Monterey, CA (Jun. 1998) vol. 4, pp. 600–603.

[8] R. Rajaraman, Topology control and routing in ad hoc networks: A survey. SIGACT News (33) (June 2002) 60–73.

[9] R. Ramanathan and R. Hain. Topology control of multihop wireless networks using transmit power adjustmant., in: *Proceedings of IEEE INFOCOM'00*, Tel Aviv, Israel (Mar. 2000), pp. 404–413.

[10] V. Rodoplu and T. Meng. Minimum energy mobile wireless networks, IEEE Journal on Selected Areas in Communications. 17(8) (1999) 1344–1444.

[11] S. Singh and C.S. Raghavendra, PAMAS: Power aware multi-access protocol with signalling for ad hoc networks. ACM Computer Communication Review 28(3) (1998) 5–26.
[12] S. Slijepcevic and M. Potkonjak, Power efficient organization of wireless sensor networks, in: *Proceedings of IEEE ICC2001*, Helsinki, Finland (Jun. 2001) pp. 472–476.
[13] C.K. Toh, *Ad Hoc Mobile Wireless Networks: Protocols and Systems* (Prentice Hall PTR, 2002).
[14] Y. Wang and X.-Y. Li, Distributed spanner with bounded degree for wireless ad hoc networks, in: *Parallel and Distributed Computing Issues in Wireless networks and Mobile Computing* (April 2002).
[15] R. Wattenhofer, L. Li, P. Bahl and Y.-M. Wang, Distributed topology control for power efficient operation in multihop wireless ad hoc networks, in: *Proceedings of IEEE INFOCOM'01* (2001).
[16] W. Ye, J. Heidemann and D. Estrin, An energy-efficient MAC protocols for wireless sensor networks, *INFOCOM'02*, New York, NY (Jun. 2002).

Qing Dai received her M.S. degree in Computer Science from Florida Atlantic University on August 2003, and M.S. degree in Microbiology from Upstate University on July 2000. She is currently a software engineer at Motorola, Plantation, FL.
E-mail: qdai@cse.fau.edu

Jie Wu is a Professor at Department of Computer Science and Engineering, Florida Atlantic University. He has published over 200 papers in various journals and conference proceedings. His research interests are in the areas of wireless networks and mobile computing, routing protocols, fault-tolerant computing, and interconnection networks. He served on many conference organization committees. Dr. Wu is on the editorial board of IEEE Transactions on Parallel and Distributed Systems and was a co-guest-editor of IEEE Computer and Journal of Parallel and Distributed Computing. He is the author of the text "Distributed System Design" published by the CRC press. He was also the recipient of the 1996–97 and 2001–2002 Researcher of the Year Award at Florida Atlantic University. Dr. Wu has served as an IEEE Computer Society Distinguished Visitor. He is a Member of ACM and a Senior Member of IEEE.
E-mail: jie@cse.faau.edu