A Distributed Algorithm to Construct Multicast Trees in WSNs: An Approximate Steiner Tree Approach

Hongyu Gong
Dept. of Electronic Engineering
Shanghai Jiao Tong University
ann@sjtu.edu.cn

Lutian Zhao
Dept. of Mathematics
Shanghai Jiao Tong University
golbez@sjtu.edu.cn

Kainan Wang
Dept. of Computer Science
Shanghai Jiao Tong University
sunnywkn@sjtu.edu.cn

Weijie Wu
School of Information Security Engineering
Shanghai Jiao Tong University
weijiewu@sjtu.edu.cn

Xinbing Wang
Dept. of Electronic Engineering
Shanghai Jiao Tong University
xwang8@sjtu.edu.cn

ABSTRACT

Multicast tree is a key structure for data dissemination from one source to multiple receivers in wireless networks. Minimum length multicast tree can be modeled as the Steiner Tree Problem, and is proven to be NP-hard. In this paper, we explore how to efficiently generate minimum length multicast trees in wireless sensor networks (WSNs), where only limited knowledge of network topology is available at each node. We design and analyze a simple and distributed algorithm, which we call Toward Source Tree (TST), to build multicast trees in WSNs. We show three metrics of TST algorithm, i.e., running time, tree length and energy efficiency. We prove that its running time is $O(\sqrt{n \log n})$, the best among all existing solutions to our best knowledge. We prove that TST tree length is in the same order as Steiner tree, give a theoretical upper bound and use simulations to show the ratio between them is only 1.114 when nodes are uniformly distributed. We evaluate energy efficiency in terms of the number of forwarding nodes in multicast trees, and prove that it is order-optimal. We give an efficient way to construct multicast tree in support of transmission of voluminous data.

1. INTRODUCTION

Wireless Sensor Network (WSN) is a network of wireless sensor nodes into which sensing, computation and communication functions are integrated. Sensors are self-organizing and deployed over a geographical region [1]. Multicasting, i.e., one-to-many message transmission, is one of the most common data transmission patterns in WSNs. Tree is the topology for non-redundant data transmission. To enable efficient multicast, multicast tree has been proposed and widely used. It has not only been applied to multicast capacity analysis in wireless networks [2–4], but has also been used to support a wide range of applications like distance education, military command and intelligent systems [5].

Many researchers have been working on constructing efficient multicast trees [6–8]. They have proposed a number of algorithms so as to minimize the routing complexity as well as achieve the time and energy efficiency (for details, please refer to the next section), but most of them did not focus on an important performance measure: the tree length. This is a critical metric since larger tree length clearly results in longer delay. To enable the messages to be forwarded farther, sensors have to increase their transmission power, causing more energy consumption as well as more serious interference to neighboring nodes. Besides, as the transmission distance increases, the messages suffer from higher probability of transmission failure.

Recently, GEographic Multicast (GEM), inspired by Euclidean Steiner Tree, was proposed for routing in dense wireless networks [9]. Formally, given a network $G = (V, E)$, the weight of each edges, and a set of terminals $S \subseteq V$, the Steiner Tree Problem is to find a tree in G that spans S with the minimum total weight [10]. This problem has been proven to be NP-hard [11], and has not been visited for a long time. Former forms of its approximate implementation were not appropriate for constructing multicast trees in WSNs for various reasons (details will be discussed in the following section.) In GEM, the authors took the first step to utilize the Steiner tree for constructing multicast trees in WSNs, achieving routing scalability and efficiency. This approach can potentially reduce the tree length, but this very simple form of utilization only considers the hop count in an unweighted graph, but not the total length of the multicast tree in a weighted graph. Furthermore, as for the performance analysis, the statistical properties were all under the assumption that all nodes are uniformly distributed, making it difficult to tell its efficiency under a realistic network environment.

In this paper, inspired by taking the advantage of the Steiner tree property, we design a novel distributed algorithm to construct an approximate minimum-length multicast tree for wireless sensor networks, aiming at achiev-

MobiHoc'15, June 22 - 25, 2015, Hangzhou, China

http://dx.doi.org/10.1145/2746285.2746296.

Table 1: Comparison of Distributed Algorithm for Approximate Steiner Construction

Algorithm	Expected tree length	Expected time	Assumptions
SPH [13]	2-approximation Steiner tree, $O(\sqrt{m})$	$O(m\sqrt{\frac{n}{\log n}})$	Shortest paths need to be known; Applied in Point-to-point network.
$KSPG$ [13]		$O(m\sqrt{\frac{n}{\log n}})$	
ADH [14]		$O(m\sqrt{\frac{n}{\log n}})$	
DA [15]	$O(\sqrt{m})$	$O(n^2)$	No shortest path information required; Applied in Point-to-point network.
TST	$O(\sqrt{m})$	$O(\sqrt{n \log n})$	No shortest path information required; Applied in Wireless network.

ing energy efficiency, ease of implementation and low computational complexity, at an affordable cost on the suboptimality of tree length. In what follows, we call our design Toward Source Tree Algorithm, or TST for short. We quantitatively evaluate TST algorithm performance under general node distribution, and show that TST has the following satisfactory metrics:

- Its running time is $O(\sqrt{n \log n})$, the best among all existing solutions for large multicast groups.
- Its tree length is in the same order as Steiner tree, and simulation shows the constant ratio between them is only 1.114 with uniformly distributed nodes.
- The number of nodes that participate in forwarding is order-optimal, yielding high energy efficiency during transmission for sensor networks.

The rest of paper is organized as follows. Section 2 states related work. In Section 3, we introduce our network model. In Section 4, we present our Toward Source Tree Algorithm to construct a multicast tree. In Section 5, 6 and 7, we evaluate the performance of our algorithm mainly from three aspects: multicast tree length, running time and energy efficiency separately. In Section 8, we use extensive simulations to further evaluate the performance. Section 9 concludes this paper.

2. RELATED WORK

We review related works in two categories: the multicast tree construction and the approximate Steiner tree.

Multicase tree construction. Many studies focus on multicast routing in wireless networks, and useful techniques for routing have been proposed in WSN. Sanchez *et al.* proposed Geographic Multicast Routing (GMR), a heuristic neighborhood selection algorithm based on local geographic information [6]. Later Park *et al.* [12] combined distributed geographic multicasting with beaconless routing. In Localized Energy-Efficient Multicast Algorithm (LEMA), forwarding elements apply the MST algorithm locally for routing [7]. Dijkstra-based Localized Energy-Efficient Multicast Algorithm (DLEMA) finds energy shortest paths leading through nodes with maximal geographical advance towards desired destinations [8]. However, few works have considered minimizing the distance of multicast routing or providing comprehensive quantitative analysis theoretically on the performance of routing policies.

Approximate Steiner Tree. Shortest Path Heuristic (SPH) and Kruskal Shortest Path Heuristic (KSPH) add new nodes to existing subtrees through the shortest paths [13]. Average Distance Heuristic (ADH) joins subtrees that contain receivers by a path passing non-receivers with minimal average distance to existing subtrees [14]. Santos *et al.* push forward distributed dual ascent (DA) algorithm, achieving good performance in practice [15]. The comparison of these algorithms with our TST algorithm is shown in Table 1. These algorithms were proposed for point-to-point networks. In this paper, we consider the Steiner Tree Problem in wireless sensor networks that are broadcast in nature. In addition, each node has limited computation and storage capability. Devices are usually battery-powered, and therefore energy-efficiency is of great importance. Due to these specific features and requirements, existing algorithms for P2P are not suitable for WSN.

To sum up, there have been extensive existing works focusing on multicast tree construction or the approximate Steiner tree problems, but we have not found a perfect adoption of Steiner tree into constructing multicast trees.

3. NETWORK MODEL

Let us first use a mathematical model to capture a wireless sensor network. We assume the network consists of n nodes in total (or we call the *network size* is n), distributed independently and identically in a unit square. Each node is assigned with a unique identifier to be distinguished from others. Each time when a source needs to transmit messages, it chooses m receivers randomly. In other words, m is the number of nodes that participate in a multicast group, or we call it the *multicast group size*. For our statistical analysis, we focus on the dense network and large multicast group where m and n are both very large, and $m \leq n$. This is particularly suitable to describe a wireless sensor network.

The geographical distribution of nodes is described by a density function $f(\mathbf{x})$ where $\mathbf{x}$ is the position vector. Here we allow $\mathbf{x}$ to be of any dimension; in the rest of this paper we let it be a two-dimensional vector for ease of presentation, but it does not hurt any generality. We assume $f(\mathbf{x})$ is independent

of n and m. We also assume that $0 < \epsilon_1 \leq f(\mathbf{x}) \leq \epsilon_2$ where ϵ_1 and ϵ_2 are both constants, i.e., a node has a positive probability to be located at any point of this area.

To ensure the connectivity of the whole network, we set the transmission range $r = \Theta\left(\sqrt{\frac{\log n}{n}}\right)$ [16]. For all nodes, r is the same and fixed. We assume that two nodes u and v can communicate with each other directly if and only if the Euclidean distance between them, d_{uv}, is no larger than r. Every node can obtain its own geographical location, e.g., via the Global Position System (GPS). However, nodes do not know the exact location of other nodes until they receive messages containing that piece of information.

Table 2: System Parameter

n	number of all nodes in the network
m	number of multicast group members
r	transmission range
r_c	search coverage range
L_V	length of a temporary tree
L_M	length of a multicast tree

4. ALGORITHM

In this section, we describe our Toward Source Tree algorithm in detail. This algorithm consists of three phases. In the first phase, the source broadcasts a message and wakes up all receivers it chooses. In the second phase, every receiver chooses the closest neighboring receiver that has shorter Euclidean distances to the source node than the receiver node itself, and then a temporary tree can be established among all receivers. However, till the end of this stage cycles might exist. Hence we eliminate these cycles in the third phase. In what follows we describe the process in detail, and we will use an example to illustrate how to generate such a tree at the end of this section.

4.1 Phase 1: Identifying Receivers

Each node has a label indicating its role in the multicast tree: "S" stands for the source and "R" for receivers. In this phase, a message containing all receivers' identifiers is sent from the source so that all nodes in the network can be aware whether they are receivers. Upon receiving this message, receivers then wake up, label themselves with "R" and be ready to participate in the multicast routing.The source will also specify its own location in this message.

This step is necessary for multicast routing since no one except the source knows which nodes the messages are destinated for. In this phase, the broadcast information will notify the nodes who are selected into the multicast group, and all receivers will be awakened.

4.2 Phase 2: Connecting All Receivers

In this phase, we first build a "temporary tree" consisting of only the multicast group members, and then find the minimum-hop shortest path between each pair of members that are directly connected in the "temporary tree". All multicast group members will be connected with the newly added relays.

Step 1: Searching Receivers in the Neighorhood

In this step, each multicast member chooses an appropriate neighbor to connect to. The neighboring member selection criteria is: each member chooses the closest one from the set of members that have shorter Euclidean distances to the source node than this node itself. If no such neighboring member can be found, then this multicast member directly connects to the source.

When a member tries to contact its neighbor members, it is regarded as the sender that sends *request message*. Its form is: *<sender id, sender location, location of previous hop, coverage range r_c, node sequence, total hop H, path length p>*. *Sender id* is used to identify the multicast members sending the *request message*, and *path length* can be updated with the location of previous hop and current hop. The coverage range r_c sets the range within which the multicast member searches for its neighboring members. The Euclidean distance between the sender and current node can be calculated with sender location, and messages will be discarded if the distance is larger than r_c. *Node sequence* records in order the nodes through which this message has passed, which acts as a guide for response from neighboring receivers so that the response can be routed via the available path. The hop count H is the number of hops the message has passed through, and p is the path length the message have been through when it reaches the current node.

In each search session, the member broadcasts the *request message* within search coverage range. The sender sets an appropriate timeout interval. Once the sender receives replies from neighboring nodes, the search session terminates. Then it enters step 2. However, if time runs out and no reply is obtained, it means that no appropriate neighboring members are found. The sender then doubles its search range and initiates another search. In Algorithm 1, we show how a multicast group member connects to their neighboring members or the source.

Algorithm 1 Neighbor Request from Multicast Members

```
1: for all receiver R in a multicast group do
2:    the number of request session: k ← 0
3:    coverage range: r_c ← r
4:    time out interval: T_0 ← Θ(2^k log n)
5:    set the node sequence as {R}
6:    total hop: H ← 0
7:    path length: p ← 0
8:    forward the request message to its neighborhood
9:    while no response is received when time is out for the
      k^th request session do
10:      k ← k + 1
11:      r_c ← 2^k r
12:      T_{k+1} ← 2T_k
13:      set the node sequence as {R}
14:      H ← 0
15:      p ← 0
16:      forward the request message to its neighborhood
17:   end while
18: end for
```

A node may receive more than one *request message* from the same sender. If it is within *coverage range*, it will choose the one with the fewest hops among all the messages. If the

numbers of hops are the same, it picks out the message with the shortest path length. Then it modifies this message. It adds itself to the *node sequence*, increases the *hop count* by 1 and calculate new *path length* given the location of the previous hop. With these information updated, it forwards the message. Algorithm 2 describes how nodes deal with *request messages* in detail.

Algorithm 2 Request Forwarding

```
1: for all node u receiving request message do
2:    dist = || location of u - sender location ||
3:    if dist < r_c then
4:       add u to node sequence
5:       H ← H + 1
6:       newDist = ||location of u - location of previous hop||
7:       p ← p + newDist
8:       forward the request message to its neighborhood
9:       if u is in the multicast group then
10:         nodeSourceDist = || location of u - source location ||
11:         senderSourceDist = || sender location - source location ||
12:         if nodeSourceDist < senderSourceDist then
13:            send respond message back to the sender
14:         end if
15:      end if
16:   end if
17: end for
```

When a multicast member finds it closer to the source than the sender of the *request message*, it might be chosen as the neighbor by the sender. Therefore, this member will choose a path to the sender and respond with the *respond message*. The form of the *respond message* is: *<sender id, respondent id, node sequence, total hop H, path length p>*. The *respond message* can be routed with the path information provided by the node sequence.

Step 2: Connecting to the Nearest Neighbor

With *respond messages*, every member selects the closest neighbor. Once a neighbor is chosen, the *connect message* is forwarded via the minimum-hop shortest path. The *connect message* is used to establish a connection between nodes in the multicast group. At the same time, all relay nodes on the minimum-hop shortest path record this pair of members, previous hop and the next hop on the path. When all receivers send the *connect message*, a "temporary tree" among all mutlicast group members including the source is constructed.

4.3 Phase 3: Eliminating Cycles

In Phase 2, we construct a "temporary tree" made up of multicast group members. However, when other nodes are added to it as relays, cycles might be formed. In particular, when paths connecting different pairs of multicast members share the same relay nodes, such node may receive redundant information, which indicates that cycles come into being. Therefore, we check the existence of cycles in this phase and eliminate them if any.

Suppose a node u acts as a relay for k ($k > 1$) pairs of nodes in the multicast group, which are directly connected in the temporary tree, denoted as (R_{11}, R_{12}), (R_{21}, R_{22}),..., (R_{k1}, R_{k2}). Let us assume that in each pair, R_{i1} is closer to source than R_{i2} ($1 \leq i \leq k$). A relay stores its previous and the next hop of the path from R_{i1} to R_{i2}, and they are denoted as PH_i and NH_i respectively. Then it chooses one pair randomly, say, (R_{j1}, R_{j2}) and keeps the information: (R_{j1}, R_{j2}, PH_j, NH_j). For other pairs (R_{i1}, R_{i2}) where $R_{i1} \neq R_{j1}$, the relay modifies their information as (R_{j1}, R_{i2}, PH_j, NH_i). Define a set Q, where $Q = \{q \mid q =< R_{i1}, R_{i2}, PH_i >, \forall\ R_{i1} \neq R_{j1}\}$. Last, it sends "*Eliminate* message Q" and its previous hops delete unnecessary edges accordingly. In Algorithm 3, we show how to wipe out the cycles.

Algorithm 3 Cycle Elimination

```
1: for all node u engaged in paths between k pairs of members do
2:    choose an integer j such that 1 ≤ j ≤ k
3:    for all i such that 1 ≤ i ≤ k and i ≠ j do
4:       forward Eliminate message Q_i =< R_i1, R_i2, PH_i >
5:    end for
6: end for
7: for all node w receiving "Eliminate message Q_i" do
8:    if w is exactly PH_i then
9:       if w is not in the multicast group then
10:         PH_i ← previous hop of w on path (R_i1, R_i2)
11:         forward the modified Q_i to the previous hop
12:         eliminate information: (R_i1, R_i2, PH_i, NH_i)
13:      end if
14:   end if
15: end for
```

4.4 Proof of Tree Topology

The previous subsections describe how we can connect mutlicast group members using our TST algorithm. Now let us prove that the topology constructed by TST algorithm is exactly a tree. We first show that temporary tree formed in the second phase has a tree topology in Lemma 4.1. But relays are added into the temporary tree to connect receivers, which might result in the existence of cycles. In Theorem 4.1 we show that cycle elimination can in fact guarantee the tree topology.

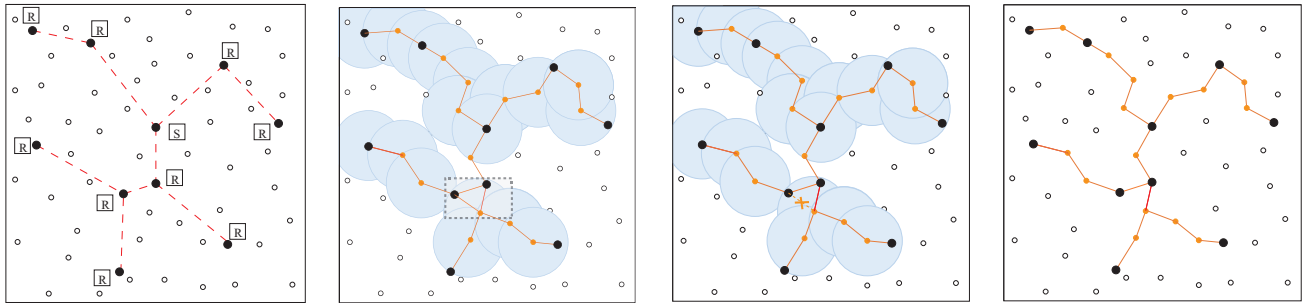
Lemma 4.1. *The temporary tree connecting m receivers has a tree topology.*

PROOF. We prove it by contradiction. Suppose that a cycle exists and k nodes are contained in this cycle, denoted by $n_1, n_2, ..., n_k$. All of them are multicast members. Without loss of generality, we assume n_i selects n_{i+1} as its neighboring member, $i = 1, \cdots, k-1$, and that n_k chooses n_1 as its neighbor. According to the criteria of neighbor selection, we know that n_{i+1} is closer to source than n_i for $i = 1, \cdots, k-1$. Therefore, n_k must be closer to source than n_1. Due to the fact that n_k's selection is n_1, we conclude that n_1 must be closer to source than n_k, which is a contradiction. This concludes the proof. □

Based on Lemma 4.1, we have the following theorem:

Theorem 4.1. *The topology connecting nodes generated by TST algorithm is a tree that spans all multicast group members.*

PROOF. The existence of cycles means that some nodes in the multicast tree may receive redundant messages, i.e., some nodes have more than one previous hop. For these



(a) Building a temporary tree spanning multicast group members

(b) Adding relay nodes

(c) Eliminating redundant edges and maintaining the topology of tree

(d) Constructing the multicast tree with relays added

Figure 1: Steps of the TST algorithm

nodes, they send "Eliminate messages" and ensure that they have only one previous hop. When all nodes in the multicast tree have only one previous hop, no cycle exists.

Multiple previous hops also indicate that multiple paths may exist between two nodes. Once some previous hops are unnecessary, the paths involving these hops can also be eliminated. Thus Algorithm 3 can eliminate these unnecessary paths, and this completes our proof. □

4.5 Illustration

We use an example to illustrate our TST algorithm. Nodes are distributed in the unit square as shown in Figure 1(a). Solid nodes represent source nodes labeled by "S", or multicast members labeled by "R". The hollow nodes can be chosen as relays. The first step is to build a temporary tree spanning all multicast members. The dashed lines denote virtual connections between two members. Then nodes on the minimal-hop shortest path are engaged as relays between two neighboring members. They form the topology as shown in Figure 1(b). Note that there exists a cycle marked with dotted rectangular box. The last step is to eliminate unnecessary edges as is done in Figure 1(c). Finally we obtain the multicast tree as is shown in Figure 1(d).

5. ANALYSIS ON TREE LENGTH

The previous section describes our Toward Source Tree algorithm. In this and the following three sections, we will discuss its performance in terms of tree length, time complexity, and energy efficiency. In this section, we discuss the length of TST. We first obtain the length of temporary tree, assuming uniform distribution of nodes and then extending to a general setting. Next we explore the length of minimal-hop shortest path that connects two receivers. Combining the length of temporary tree and the path, we can derive the upper bound for the multicast tree length.

5.1 Temporary Tree in Uniform Distribution

Assuming that all nodes are uniformly distributed, we start by discussing the tree length of the temporary tree.

Lemma 5.1. *Assume nodes are uniformly distributed in a unit square. The expected length of the temporary tree is upper bounded by* $c\sqrt{m}$*, where* $c = 5.622$.

PROOF. We establish a two-dimensional coordinate system shown in Figure 2. Let the source be the origin, and

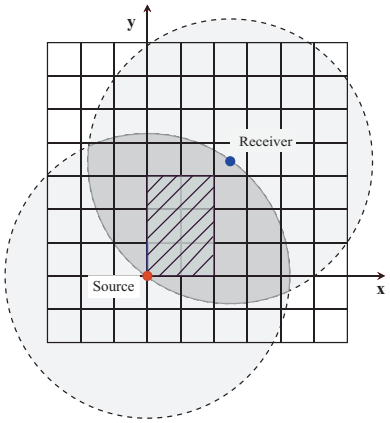


Figure 2: Two-dimensional coordinate system, actual and approximate neighbor regions

X-axis as well as Y-axis parallel to the square edge. The whole network is divided into m square cells, and the edge length of each cell is $\frac{1}{\sqrt{m}}$. In the coordinate system, the edge length is normalized to be 1, so intersections in Figure 2 have integer coordinates. In Figure 2, the red node is the source node S and the blue node is the receiver R whose coordinate is (x, y). Without loss of generality, suppose that x and y are both positive. Set the radius to be the Euclidean distance between S and R, and draw two circles with the center in S and in R respectively. The intersected region of the unit square and two circles' overlapping area is the shaded region. According to criteria for neighbor receiver selection, the shaded region is where the possible neighbor receivers of R are located. If no receiver lies in this shaded area, R connects to S directly.

We shrink the possible neighbor region and consider an approximate region marked with parallel lines in Figure 2. The shrinking region is denoted by N'_R, where

$$N'_R = \{(a, b) | 0 \le a \le \lfloor x \rfloor, 0 \le b \le \lfloor y \rfloor\}. \quad (1)$$

We also rearrange the nodes in each cell. Suppose that the coordinate of a node is (a, b), then we move it to the point with coordinate $(\lfloor a \rfloor, \lfloor b \rfloor)$. All nodes are now further from receiver R after the rearrangement. By shrinking the possi-

ble neighbor region and rearranging the nodes, we can derive the upper bound of the temporary tree length.

Let M_R denote the set of all receivers. We use the coordinate of the point farthest from origin in a cell as the coordinate of this cell, and let $p_{i,j}$ be the probability that the node in the cell (i,j) is chosen by the receiver R as a neighbor to connect to. Let p_S be the probability that the receiver directly connects to the source. Assume that the nodes in cell (i,j) is chosen if all cells (i',j') are empty, where $i \le i' \le \lfloor x \rfloor$ and $j \le j' \le \lfloor y \rfloor$. This assumption makes the edge length estimation larger than the actual edge length. We have

$$p_{i,j} \le \left(1 - \frac{(\lfloor x \rfloor + 1 - i)(\lfloor y \rfloor + 1 - j) - 1}{m}\right)^{m-1} - \left(1 - \frac{(\lfloor x \rfloor + 1 - i)(\lfloor y \rfloor + 1 - j)}{m}\right)^{m-1}, \quad (2)$$

$$p_S \le (1 - \frac{\lfloor x \rfloor \lfloor y \rfloor}{m})^{m-1}. \quad (3)$$

The expected length of temporary tree is:

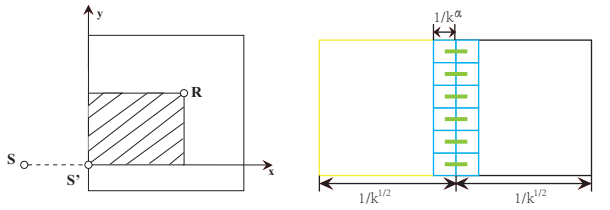
$$E(L_V) \le \frac{1}{\sqrt{m}} \sum_{R \in M_R} E\,(\sum_{i=1}^{\lfloor x \rfloor} \sum_{j=1}^{\lfloor y \rfloor} p_{i,j} \sqrt{(x-i+1)^2 + (y-j+1)^2} + p_S \sqrt{x^2+y^2}\,) . \quad (4)$$

Combining (2), (3) and (4), we can derive the upper bound of tree length with some integration and summation methods, and finally obtain $E(L_V) \le 5.622\sqrt{m}$. □

5.2 Temporary Tree in General Distribution

We further study the case that nodes are non-uniformly distributed. We partition the unit square into k small squares, where $m = k^{1+\gamma}$ and $0 < \gamma < 1$. Lemma 5.2 can be applied to estimate the intra-square connection, and we study the inter-square connection in Lemma 5.3. With both inter- and intra- square edge estimation, we derive upper bound for temporary tree length in general distribution.

Lemma 5.2. *Let m nodes be independently distributed in a unit square with a density function $f(\mathbf{x})$. The source S is located outside the square. Let each node connect to the closest neighbor that has shorter Euclidean distance to S than the node itself. If no such receiver exists, it does not connect to other nodes. A tree can be constructed among m nodes, and the expected length of such a tree is upper bounded by $c\sqrt{m}$, where $c = 5.622$.*



(a) Approximate neighbor region when the source is located outside the square

(b) Inter-square edges between nodes in adjacent square cells

Figure 3: Intra- and inter-square edges

PROOF. We denote the tree generated in the way described in this lemma as T. There is a difference of this lemma from Lemma 5.1, since the source S is located outside the square as is shown in Figure 3(a). Point S' is the closest point to S among all nodes in the network. With S' as the source, a temporary tree as mentioned in TST algorithm can be established spanning all nodes in the network. We denote the temporary tree as T'. In the following, we demonstrate that the tree length of T' can be used to estimate the upper bound of tree length of T.

For a node R, we use N_R to denote the regions where nodes might be selected by R as a its neighbor. We use a rectangular region as approximate neighbor region N'_R, and $N'_R \subseteq N_R$. The approximate neighbor region is the region marked with parallel lines in Figure 3(a). We use the method adopted in the proof of Lemma 5.1 to obtain the length of the temporary tree T'.

We can observe that the approximate neighbor regions are the same in both cases that we take S as the source and that we take S' as the source. There are some details that need to be clarified. Firstly, when we only consider the nodes in approximate neighbor region, the estimated tree length is larger than actual length, because we ignore the nodes that are closer to node R in accurate neighbor region. Secondly, if neighbors exist in the approximate region, the estimations of edge length are the same for both T and T'. Thirdly, if no neighbor is found in approximate region for a node, we assume that it doesn't connect to others in T but it connects to the source in T' in our calculation. From the analysis above, we can conclude that length of T is upper bounded by the length of T'.

Also recall that in our proof of Lemma 5.1, and $5.622\sqrt{m}$ is the upper bound for the length of temporary tree wherever S' is. In summary, we can directly use estimated tree length in Lemma 5.1 as the upper bound of the tree length of T. This completes our proof. □

Lemma 5.3. *Let m nodes be independently distributed in a unit square with density function $f(\mathbf{x})$. The unit square $[0,1] \times [0,1]$ can be partitioned into k square cells with edge length of $\frac{1}{\sqrt{k}}$, where $m = k^{1+\gamma}$ and $0 < \gamma < 1$. The expected minimal distance between two nodes in adjacent cells is in the order of $o\left(\frac{1}{\sqrt{k}}\right)$.*

PROOF. We partition the cells with edge length of $\frac{1}{\sqrt{k}}$ into smaller grids with edge length of $\frac{1}{k^\alpha}$, where $\alpha > \frac{1}{2}$. Then we look for a pair of nodes located in adjacent grids of the adjacent cells, as is shown by green lines in Figure 3(b). Denote P as the probability that one pair of such nodes exists, and we have

$$P = 1 - \left(1 - \left(1 - \left(1 - \frac{1}{k^{2\alpha-1}}\right)^{\frac{m\epsilon_1}{k}}\right)^2\right)^{k^{\alpha-1/2}}. \quad (5)$$

when $\alpha < \gamma + \frac{1}{2}$, we have

$$P \sim 1 - \exp(-2\epsilon_1 k^{1/2-\alpha+\gamma} - \frac{\epsilon_1}{\log 2} k^{1/2-\alpha}). \quad (6)$$

Furthermore, we can show that

$$P^k \to 1. \quad (7)$$

Hence the probability that k pairs of nodes exist at the same time in adjacent grids of k adjacent cells converges to 1. We

conclude that the expectation of minimal distance between two nodes in adjacent cells is in an order of $o\left(\frac{1}{\sqrt{k}}\right)$. The minimal total length of inter-square edges connecting k cells is $o(\sqrt{k})$, and is therefore $o(\sqrt{m})$. □

With both inter- and intra-square edges, we derive upper bound for the length of temporary tree in general distributions in the following lemma.

Lemma 5.4. *Let m nodes be independently distributed with a density function $f(\mathbf{x})$. The expectation for the total length of temporary tree is $E[L_V] \leq c\sqrt{m}\int_{\mathbf{x}\in[0,1]^2}\sqrt{f(\mathbf{x})}d\mathbf{x}$, where $c \approx 5.622$.*

Proof. We partition the unit square into k small squares with edge length of $\frac{1}{\sqrt{k}}$. We estimate the temporary tree length in the case of non-uniform distribution by considering two types of edges between nodes. One is intra-square edge, i.e., a node chooses to connect to another node located in the same square. The other is inter-square edge, i.e., a node connects to another one located in a different square.

We first consider the intra-square edges. We use a modified neighbor selection criteria to build a tree among nodes in the same small squares: a node connects to the nearest one that is closer to source and in the same square. Considering that the source is located outside the square, Lemma 5.2 can be used to derive the upper bound of the length of intra-square edges: $\frac{c}{\sqrt{k}}\sqrt{m\int_{Sq_i}f(x)dx}$, where Sq_i is a small square region in the set of k regions $\{Sq_1, \cdots, Sq_k\}$.

When modified neighbor selection criteria is applied, more constraints are added to neighbor selection. Therefore, the distance between a node and its selected neighbor may increase. Temporary tree length should be no larger than the sum of intra-square edge length and inter-square edge length. The length of intra-square edges within k squares is

$$\sum_{i=1}^{k} c\frac{1}{\sqrt{k}}\sqrt{m\int_{Sq_i}f(\mathbf{x})d\mathbf{x}}. \tag{8}$$

By Riemann-Stiejies integration, we have

$$\sum_{i=1}^{k} c\frac{1}{\sqrt{k}}\sqrt{m\int_{Sq_i}f(\mathbf{x})d\mathbf{x}} \leq c\sqrt{m}\int_{\mathbf{x}\in[0,1]^2}\sqrt{f(\mathbf{x})}d\mathbf{x}. \tag{9}$$

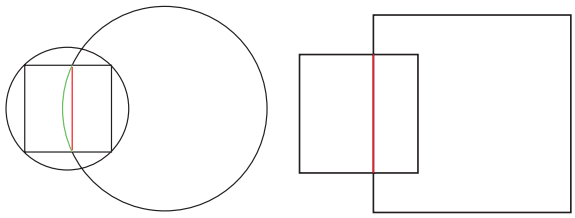
The length of inter-square edges is $o(\sqrt{m})$ according to Lemma 5.3, and it can be ignored compared with intra-square edge length. Therefore, the temporary tree length is upper bounded by $c\sqrt{m}\int_{\mathbf{x}\in[0,1]^2}\sqrt{f(\mathbf{x})}d\mathbf{x}$. □

5.3 Path with Minimal Hops

Members are connected by the minimal-hop shortest path. In this part, we study the relationship between the path length and the Euclidean distance between two nodes.

Lemma 5.5. *Let n nodes be independently and identically distributed over $[0,1]\times[0,1]$ with a distribution function $f(\mathbf{x})$. Suppose that the Euclidean distance between two nodes u and v is x. The following properties hold:*

1. *The expected number of fewest relays that are needed to connect u and v converges to $\frac{x}{r}$ as n approaches ∞;*
2. *The expected length of the path connecting uv and involving the fewest relays converges to x.*



(a) 2-norm (b) infinite-norm

Figure 4: We estimate the probability of points on the red segment instead of green arc

Proof. We set the transmission range as $r = \sqrt{\frac{2}{\epsilon_1}}\sqrt{\frac{\log n}{n}}$. The left circle in Figure 4(a) shows the transmission range of u. The circumscribed square is embedded properly in this left circle. The right circle with center in v has a radius of x. The intersection of the square and the right circle is marked as the green arc. For simplicity, we consider the red segment rather than the green arc in the following analysis to estimate number of relays, and this will overestimate the hop count. Then we can use ∞-norm in Figure 4(b) for estimation instead of the 2−norm.

Let A be the event that the next relay exists with distance s from u, and let B be the event that there is a node within its transmission region. We have

$$P(A|B) = \frac{e^{-2n\epsilon_1 r^2}}{1-e^{-2n\epsilon_1 r^2}}(e^{2n\epsilon_1 sr}-1). \tag{10}$$

Denote $E_r(x)$ as the expected number of relay nodes on the path with the minimal hops between u and v. Here we consider that the square with side length r in the ∞-norm is contained in a circle with diameter $\sqrt{2}r$ in the 2−norm. We have the following functional equation when plugging in $r = \epsilon_1^{-1/2}\sqrt{\frac{\log n}{n}}$:

$$\begin{aligned} E_r(x) &= \int_0^r (1+E_r(x-s))\frac{2n\epsilon_1 re^{-2n\epsilon_1 r^2}}{1-e^{-2n\epsilon_1 r^2}}e^{2n\epsilon_1 sr}ds \\ &= 1+\frac{2\log n}{n^2-1}\int_0^1 E_r(x-\alpha r)n^{2\alpha}d\alpha. \end{aligned} \tag{11}$$

Clearly, the lower bound of $E_r(x)$ is x/r. As for the upper bound of hop count, we can prove that for a fixed $d > 1$, $E_r(x) \leq d(x/r)+1$ by induction. $E_r(x) = 1$ when $x \in (0, r)$. Suppose that $E_r(x-\alpha r) \leq d\frac{x-\alpha r}{r}+1$, then we can prove

$$\begin{aligned} E_r(x) &\leq 2+\frac{2\log n}{n^2-1}\int_0^1 d\frac{x-\alpha r}{r}n^{2\alpha}d\alpha \\ &= 2+d\frac{x}{r}-d\frac{n^2}{n^2-1}+\frac{d}{2\log n} \leq d(x/r)+1. \end{aligned} \tag{12}$$

The last inequality in (12) holds for

$$d \geq \frac{2\log n(n^2-1)}{n^2(2\log n-1)+1}. \tag{13}$$

From (12) and (13), $E_r(x)$ converges to x/r as n approaches ∞, so the first property holds. Since the transmission range is r, path length approaches x. Hence the second property also holds. □

5.4 Length of Multicast Tree

Based on the lengths of temporary tree and minimum-hop shortest path, now we derive the length of multicast tree constructed by TST algorithm in the following theorem.

Theorem 5.1. *Let n nodes be independently and identically distributed in a unit square with density function $f(\mathbf{x})$. The expected length of Toward Source Tree spanning m receivers randomly chosen from n nodes is upper bounded by $c\sqrt{m}\int_{\mathbf{x}\in[0,1]^2}\sqrt{f(\mathbf{x})}d\mathbf{x}$, where $c = 5.622$.*

PROOF. Denote $e_{i,j}$ as the edge connecting receivers i and j in the temporary tree T_V, and denote $l_{i,j}$ as the length of the minimal-hop shortest path between these two receivers. Since redundant links are eliminated, it holds that

$$E(L_M) \leq \sum_{e_{i,j}\in T_V} E(l_{i,j}). \tag{14}$$

The path length converges to the Euclidean distance as network size approaches ∞ according to Lemma 5.5. Therefore we have:

$$E(L_M) \leq c\sqrt{m}\int_{\mathbf{x}\in[0,1]^2}\sqrt{f(\mathbf{x})}d\mathbf{x}. \tag{15}$$

This completes our proof. □

Remark: We derive an upper bound for the Toward Source Tree, but it is not a tight bound. In Section 8, we will show that TST algorithm has even better empirical performance than our theoretical bound. Let l_{MST} and l_{ST} denote the lengths of the minimum spanning tree and the Steiner tree connecting m receivers respectively. Steele proved that when m nodes are independently distributed in a unit square with density function $f(\mathbf{x})$, then $l_{MST} = c_1\sqrt{m}\int_{[0,1]^2}\sqrt{f(\mathbf{x})}d\mathbf{x}$ where c_1 is a constant [17]. Roberts estimates that $c_1 = 0.656$ [18]. Du *et al.* proved that $l_{ST} \geq \frac{\sqrt{3}}{2}l_{MST}$ [19]. Therefore the lower bound of Steiner tree length is

$$L_{ST} \geq \frac{\sqrt{3}}{2}c_1\int_{[0,1]^2}\sqrt{f(\mathbf{x})}d\mathbf{x}. \tag{16}$$

The length of Toward Source Tree with m receivers is in the same order as that of Steiner tree, and thus the difference between them is only a constant ratio less than 10.

6. ANALYSIS ON TIME COMPLEXITY

Wireless sensor networks have dynamic topology since some sensors run out of energy while new ones are added to the network. To enable information to arrive at destinations in a short time, we expect the algorithm for multicast tree construction to be at small time costs. In this section, we derive the time complexity of TST.

Theorem 6.1. *Let n nodes be independently and identically distributed in a unit square. The running time of TST algorithm has an upper bound of $O(\sqrt{n\log n})$ and a lower bound of $O\left(\sqrt{\frac{n}{\log n}}\right)$.*

PROOF. There are three serial phases in TST algorithm, so we discuss the time cost of each phase one by one. In Phase 1, the messages containing location information of the source are broadcast in the network. The furthest distance between the source and another node is $O(1)$, so $O\left(\frac{1}{r}\right)$ relays are needed for a message to reach one node. In expectation, there are $\pi r^2\epsilon_2 n = O\left(nr^2\right)$ nodes within transmission range of a node and hence a node has to wait for $O\left(nr^2\right)$ time slots to transmit a message at most. The time cost in Phase 1 is

$$O\left(\frac{1}{r}\right) \leq E(t_1) \leq O\left(nr\right). \tag{17}$$

In Phase 2, the dominant time cost comes from searching for neighboring receivers. In the k^{th} search session, the coverage range is $2^k r$. We need $O\left(2^k\right)$ relays to forward *request messages* from one receiver to any other nodes within its search coverage range. Since the coverage range does not exceed $\sqrt{2}$, the number of search sessions cannot be more than $\left\lceil\log_2\frac{\sqrt{2}}{r}\right\rceil$.

$$E(t_2) \leq O\left(\sum_{k=0}^{\left\lceil\log_2\frac{\sqrt{2}}{r}\right\rceil} 2^k nr^2\right) \leq O(nr). \tag{18}$$

In Phase 3, the worst case is that relays on the path whose length is $O(1)$ form cycles. Time for cycle elimination is

$$E(t_3) = O\left(\frac{1}{r}nr^2\right) = O(nr). \tag{19}$$

Given that $E(t) = \sum_{i=1}^{i=3} E(t_i)$ and $r = O\left(\sqrt{\frac{\log n}{n}}\right)$, the total running time is

$$O\left(\sqrt{\frac{n}{\log n}}\right) \leq E(t) \leq O(\sqrt{n\log n}). \tag{20}$$

This completes our proof. □

Remark: For any algorithm to construct a multicast tree among a group of nodes, broadcast in Phase 1 is necessary. Since no node has a knowledge of the multicast group except the source, such information has to be forwarded to every node in the network so that they can know whether they should participate in multicasting. The lower bound of time for multicast tree construction is $O\left(\sqrt{\frac{n}{\log n}}\right)$. Since TST achieves the time complexity upper bounded by $O(\sqrt{n\log n})$, the minimal time cost to construct a multicast tree is also upper bounded by $O(\sqrt{n\log n})$. Hence the time complexity of TST algorithm shares the same upper and lower bounds as the minimal time cost, and the ratio between these two bounds is only $O(\log n)$.

The length of multicast trees have a great influence on communication quality in terms of transmission delay and wireless interference. Construction of minimum-length trees is an NP-hard problem, and takes exponential time. Approximate algorithms achieve larger tree length with lower time complexity. Now we explore the relationship between tree length and time complexity in Figure 5. Since the lower bound of time needed for multicast tree construction is $O\left(\sqrt{\frac{n}{\log n}}\right)$, the region with time complexity smaller than $O\left(\sqrt{\frac{n}{\log n}}\right)$ is infeasible. Accurate solution to Steiner tree problem achieves the approximation ratio of 1 at the cost of exponential time, and our algorithm achieves the ratio of 10. The approximation ratio of other algorithms like those in Table 1 approaches 1 but they have larger time costs.

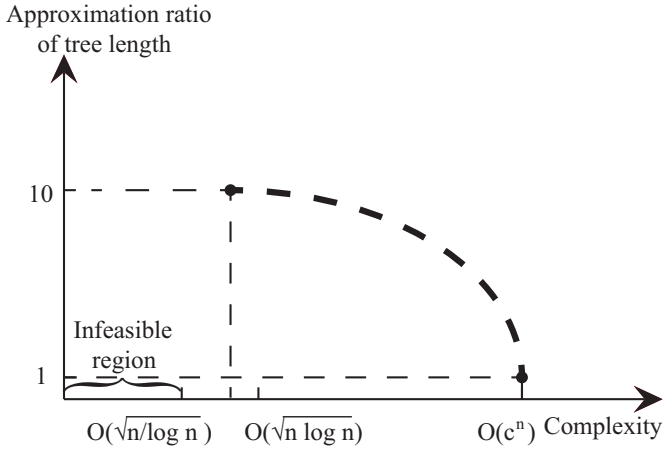


Figure 5: Relationship between tree length and time complexity

7. ANALYSIS ON ENERGY EFFICIENCY

Energy is a primary consideration in wireless sensor networks, since sensors are battery-powered and their energy is limited. There are two main types of energy consumption: 1) the energy consumed to construct such multicast trees; and 2) the energy needed to send messages along the tree constructed by this algorithm. When a large amount of information is transmitted, the latter type is the dominant part in energy consumption. Therefore in this section, we focus on the energy consumed in message forwarding along the constructed tree.

Since the transmission range is fixed, the number of forwarding nodes in the tree determines the energy consumption for information propagation. We evaluate the number of forwarding nodes in Theorem 7.1.

Theorem 7.1. *Let n nodes be independently and identically distributed in the unit square. The number of forwarding nodes in the multicast tree is*

$$N_{TST} = \begin{cases} \Theta\left(\sqrt{\frac{mn}{\log n}}\right), & m = O\left(\frac{n}{\log n}\right); \\ \Theta(m), & m = \omega\left(\frac{n}{\log n}\right). \end{cases} \quad (21)$$

When $m = O(n/\log n)$, the number of forwarding nodes in TST algorithm is order-optimal.

PROOF. Let T_V be the temporary tree, and $e_{i,j}$ be an edge in the temporary tree connecting two receivers i and j, and $d_{i,j}$ be the Euclidean distance between them. When m is small, relay nodes are the dominant part of the forwarding nodes in our multicast tree. The total number of transmitting nodes is $N_{TST} = \Theta\left(\sum_{e_{i,j}\in T_V} \frac{d_{ij}}{r}\right) = \Theta\left(\frac{\sqrt{m}}{r}\right)$. As m grows larger, receivers are closer to each other and thus fewer relay nodes are added. Therefore, receivers are the dominant part of all nodes in the multicast tree, so $N_{TST} = \Theta(m)$. The critical value of m to distinguish the two cases satisfies $\Theta\left(\frac{\sqrt{m_c}}{r}\right) = \Theta(m_c)$, so $m_c = \Theta\left(\frac{n}{\log n}\right)$.

Denote N_{min} as the minimal number of nodes that are engaged in message propagation. The upper bound of N_{min} is given assuming that nodes are uniformly distributed [2]. We use its technique and obtain N_{min} in general distribution,

$$N_{min} = \begin{cases} \Theta\left(\sqrt{\frac{mn}{\log n}}\right), & m = O\left(\frac{n}{\log n}\right); \\ \Omega\left(\frac{n}{\log n}\right), & m = \omega\left(\frac{n}{\log n}\right). \end{cases} \quad (22)$$

As we can see, when $m = O\left(\frac{n}{\log n}\right)$, the number of forwarding nodes in TST algorithm is optimal in the order sense. □

Remark: When $m = \omega\left(\frac{n}{\log n}\right)$, energy consumption of TST algorithm may not be order-optimal. However, to achieve the minimal energy consumption, we have to find the minimum connected dominating set of a given graph, which is proven to be NP-complete [20] and requires global information of network topology. Therefore, we regard it as an acceptable cost of energy to achieve the feasibility and time-efficiency in practice.

8. PERFORMANCE EVALUATION

We perform extensive simulations to evaluate the empirical performance of Toward Source Tree algorithm in terms of the multicast tree length and energy consumption. We consider two common distribution patterns in this part: uniform distribution and normal distribution.

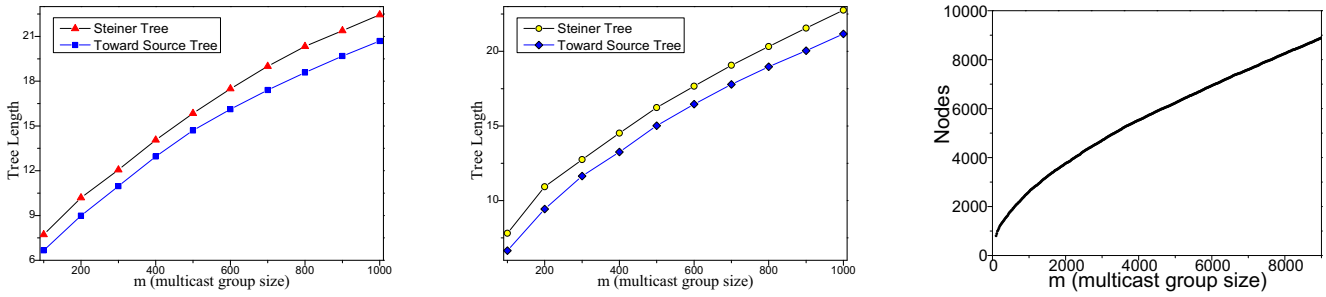
8.1 Uniform Distribution

We first consider nodes are uniformly distributed in a unit square and transmission range is set to be $r = \sqrt{\frac{\log n}{n}}$. We explore the effect of multicast group size m on the tree length. Assuming that the network size is fixed as 1000, we obtain the lengths of the Stenier tree and TST when the value of m varies. The length of the Steiner tree can be obtained via NewBossa in [21]. Two curves in Figure 6(a) describe the relationship between multicast group size and the lengths of the Toward Source Tree as well as the Steiner Tree. It is shown that the length of TST is larger than that of the Steiner Tree but quite close to it. We change the network size in the range $[200, 2200]$ and change the multicast group size accordingly, and run extensive simulations. According to the statistics, the ratio of the tree length achieved by the two algorithms is 1.114 on average. When nodes are uniformly distributed, Toward Source Tree is a good approximation of the Steiner Tree.

Next we evaluate energy consumption by considering the number of forwarding nodes engaged in multicasting. To derive the statistical properties of TST, we set the network size as $100,000$. In Figure 6(c), when the multicast group is small, the first part of the curve indicates that the number of forwarding nodes is $O(\sqrt{m})$. As there are more multicast group members, the number of forwarding nodes grows linearly with the group size.

8.2 Non-uniform Distribution

For the case of non-uniform distribution, we first choose the source location $\mathbf{x_s}$ randomly in the unit square. Then we consider that other nodes satisfy the normal distribution: $f(\mathbf{x}) = \frac{1}{\sqrt{2\pi}} e^{-\frac{\|\mathbf{x}-\mathbf{x_s}\|^2}{2}}$, where $\|\mathbf{x}-\mathbf{x_s}\|$ is the Euclidean distance between the node and the source. It is possible that the nodes are scattered outside unit square during simulations. If this happens, we relocate these nodes until they are within this unit square.



(a) Tree length in uniform distribution (b) Tree length in normal distribution (c) Number of forwarding nodes

Figure 6: Algorithm evaluation when nodes are uniformly distributed

In normal distribution, the length of Toward Source Tree is only a little larger than the optimal length as is shown in Figure 6(b), when the network size is 1000. When the network size and multicast group size change, a large amount of statistics show that the ratio between them is 1.110 on average. Similar to the case of uniform distribution, the number of forwarding nodes in the multicast tree is also linear with $\sqrt{m}$, and becomes linear with m as m increases and the network size keeps unchanged.

9. CONCLUSION

In this paper, we propose a novel algorithm, which we call Toward Source Tree, to generate approximate Steiner Trees in wireless sensor networks. The TST algorithm is a simple and distributed scheme for constructing low-cost and energy-efficient multicast trees in the wireless sensor network setting. We prove its performance measures in terms of tree length, time complexity and energy efficiency. We show that the tree length is in the same order as, and is in practice very close to, the Steiner tree. We prove its running time is the shortest among all existing solutions. We prove that the number of nodes that participate in forwarding is order-optimal, yielding high energy efficiency for applications.

10. ACKNOWLEDGEMENT

This work is supported by NSF China (No.61325012, 61271219, 61221001, 61428205); China Ministry of Education Doctor Program(No.20130073110025); Shanghai Basic Research Key Project (12JC1405200, 11JC1405100); Shanghai International Cooperation Project (No. 13510711300).

11. REFERENCES

[1] J. Crowcroft, M. Segal, and L. Levin, "Improved structures for data collection in wireless sensor networks." *Proc. IEEE INFOCOM*, 2014.

[2] Z. Wang, H. R. Sadjadpour, and J. J. Garcia-Luna-Aceves, "A unifying perspective on the capacity of wireless ad hoc networks," in *Proc. IEEE INFOCOM*, 2008.

[3] X. Y. Li, "Multicast capacity of wireless ad hoc networks," *IEEE/ACM Trans. Netw.*, vol.17, no.3, pp.950-961, 2009.

[4] S. Shakkottai, X. Liu, and R. Srikant, "The multicast capacity of large multihop wireless networks," *IEEE/ACM Trans. Netw.*, vol.18, no.6, pp.1691-1700, 2010.

[5] U. Varshney, "Multicast over wireless networks," *Communications of ACM*, vol.45, no.12, pp.31-37, 2002.

[6] J. A. Sanchez, P. M. Ruiz, and I. Stojmenovic, "GMR: Geographic multicast routing for wireless sensor networks," in *Proc. IEEE SECON*, 2006.

[7] J. A. Sanchez and P. M. Ruiz, "LEMA: Localized Energy-Efficient Multicast Algorithm based on Geographic Routing," in *Proc. of 31st IEEE Conference on Local Computer Networks*, pp.3-12, 2006.

[8] B. Musznicki, M. Bartosz, and P. Zwierzykowski, "Dijkstra-based localized multicast routing in wireless sensor networks," in *Proc. IEEE CSNDSP*, 2012.

[9] L. Galluccio, G. Morabito and S. Palazzo, "GEographic multicast (GEM) for dense wireless networks: protocol design and performance analysis," in *IEEE/ACM Trans. Netw.*, vol.21, no.4, pp.1332-1346, 2013.

[10] J. Byrka, F. Grandoni, T. Rothvoss and L. Sanità, "Steiner tree approximation via iterative randomized rounding." *Journal of the ACM,* vol.60, no.1, 2013.

[11] R. M. Karp, *Reducibility among Combinatorial Problems.* Springer, 1972.

[12] Park, Hosung, et al., "Distributed multicast protocol based on beaconless routing for wireless sensor networks," *Proc. IEEE ICCE*, 2013.

[13] F. Bauer, and A. Varma, "Distributed algorithms for multicast path setup in data networks," *IEEE/ACM Trans. Netw.*, vol.4, no.2, pp.181-191, 1996.

[14] L. Gatani, G. Lo. Re, and S. Gaglio, "An efficient distributed algorithm for generating multicast distribution trees," in *Proc. IEEE ICPP Workshops*, 2005.

[15] M. Santos, L. M. A. Drummond, and E. Uchoa, "Distributed dual ascent algorithm for steiner problems in networks," *Anais do Simpósio Brasileiro de Redes de Computadores*, pp.381-396, 2007.

[16] M. D. Penrose, "A strong law for the longest edge of the minimal spanning tree," *The Annals of Probability*, vol.27, no.1, pp.246-269, 1999.

[17] J. M. Steele, "Growth rates of euclidean minimal spanning trees with power weighted edges," *The Annals of Probability*, vol.16, no.4, pp.1767-1787, 1988.

[18] F. D. K. Roberts, "Random minimal trees," *Biometrika*, vol.55, no.1, pp.255-258, 1968.

[19] D. Z. Du, and F. K. Hwang, "A proof of the Gilbert-Pollak conjecture on the Steiner ratio," *Algorithmica* vol.7, no.1-6, pp.121-135, 1992.

[20] R. G. Michael, and S. J. David, *Computers and Intractability: A Guide to the Theory of NP-completeness*, W. H. Freeman, 1978.

[21] "Bossa", Dept. of CS at Princeton, [Online]. Available at: `http://www.cs.princeton.edu/~rwerneck/bossa/`. [Accessed: July 17, 2014].