Distributed Algorithms for Constructing Approximate Minimum Spanning Trees in Wireless Networks

Maleq Khan, *Member, IEEE,* Gopal Pandurangan, *Member, IEEE,* and V.S. Anil Kumar

***Abstract*— While there are distributed algorithms for the Minimum Spanning Tree (MST) problem, these algorithms require relatively large number of messages and time, and are fairly involved, making them impractical for resource-constrained networks such as wireless sensor networks. In such networks, a sensor has very limited power, and any algorithm needs to be simple, local, and energy efficient. Motivated by these considerations, we design and analyze a class of simple and local distributed algorithms called Nearest Neighbor Tree (NNT) algorithms for energy-efficient construction of an approximate MST in wireless networks. Assuming that the nodes are uniformly distributed, we show provable bounds on both the *quality* of the spanning tree produced and the *energy needed* to construct them. We show that while NNT produces a close approximation to the MST, it consumes asymptotically less energy than the classical message-optimal distributed MST algorithm due to Gallager, Humblet, and Spira. Further, the NNTs can be maintained dynamically with polylogarithmic rearrangements under node insertions/deletions. We also perform extensive simulations, which show that the bounds are much better in practice. Our results, to the best of our knowledge, demonstrate the first tradeoff between the quality of approximation and the energy required for building spanning trees on wireless networks, and motivate similar considerations for other important problems.**

***Index Terms*— Distributed Algorithms, Randomized Approximation Algorithms, Energy-Efficient Algorithms, Minimum Spanning Tree, Wireless Networks, Sensor Network.**

I. Overview

A. Introduction and Motivation

THE Minimum Spanning Tree (MST) problem is an important and commonly occurring primitive in the design and operation of data and communication networks. For instance, in ad hoc sensor networks, MST is the optimal routing tree for data aggregation [2]. Traditionally, the efficiency of distributed algorithms is measured by the running time and the number of messages exchanged among the computing nodes, and a lot of research has gone into the design of algorithms that are optimal with respect to such criteria. The classical algorithm due to Gallager, Humblet, and Spira (henceforth referred to as the GHS algorithm) [1] uses $\Theta(n \ln n + |E|)$ messages, and is essentially optimal with respect to the message complexity. There are distributed algorithms that find the MST [3], [4] and are essentially optimal in terms of time complexity: they run in $O(Diam(G) + n^{1/2}\text{polylog}(n))$ time, and there are (almost) matching lower bounds. However, these time-optimal algorithms involve a lot of message transfers (much more than GHS). Even for a wireless network modeled by a unit disk graph or even a ring, any distributed algorithm to construct an MST needs $\Omega(n \ln n)$ messages [5], [6]. Despite their theoretical optimality, these algorithms are fairly involved, require synchronization and a lot of book keeping; such algorithms are impractical for ad hoc and sensor networks [5]. For example, consider sensor networks — an ad hoc network formed by large numbers of small, battery-powered, wireless sensors. In many applications, the sensors are typically "sprinkled" liberally in the region of interest and the network is formed in an ad hoc fashion by local self-configuration. Since each sensor usually knows only its neighbors, the network management and communication has to be done in a *local and distributed* fashion. Additionally, because of battery limitations, energy is a very crucial resource. A distributed algorithm which exchanges a large number of messages can consume a relatively large amount of energy (and also time) is not suitable in an energy-constrained sensor network. This is especially true in a *dynamic* setting – when the network needs to be reconfigured (e.g., due to mobility or failures) frequently and quickly. Reconfiguration is also necessary to evenly distribute the energy consumption among all nodes and thus, to increase the network lifetime [7].

Thus it is necessary to develop simple, local, distributed algorithms which are energy-efficient, and preferably also time-efficient, even at the cost of being *sub-optimal* (see e.g., [5], [8], [9] for such algorithms in the context of wireless sensor networks — discussed more below). This adds a new dimension to the design of distributed algorithms for such networks. Thus we can potentially *tradeoff* optimality of the solution to work done by the algorithm. In a sensor network, the total energy required (*"energy complexity"*) in a distributed algorithm typically depends on the time needed, the number of messages exchanged, and the radiation energy needed to transmit the messages over a certain distance [9], [10]. The radiation energy needed to transmit a message is typically proportional to some *work function* f (typically square or some small power) of the distance between the sender and the receiver [7], [11]. Thus it becomes important to measure efficiency of a distributed algorithm in terms of energy, besides the number of messages.

While there has been a lot of recent work on local algorithms for construction of *low-weight* connected subgraphs in wireless ad hoc networks (motivated by topology control and energy-efficient routing) [12], [13], to the best of our knowledge, there has been

Maleq Khan is with the Virginia Bioinformatics Institute, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061, USA. E-mail: maleq@vbi.vt.edu.

Gopal Pandurangan is with the Department of Computer Science, Purdue University, West Lafayette, IN 47907, USA. E-mail: gopal@cs.purdue.edu.

V.S. Anil Kumar is with the Department of Computer Science, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061, USA. E-mail: akumar@vbi.vt.edu

little work on localized construction of exact or approximate MSTs, especially in the context of wireless ad hoc networks. A structure is *low weight* if its total edge length is within a small factor of the total edge length of the MST, but the structure may have cycles. It is easy to show that MST cannot be constructed in a purely localized manner, i.e., each node cannot determine which edge is in the defined structure by using only the information of the nodes within some constant hops. For example, Li, Hou, and Shia [8] proposed a method to build what they call a *local minimum spanning tree (LMST)*, which is guaranteed to be connected and has bounded degree, but is *not* a low-weight structure. In fact, X. Li et al. [5] demonstrate the difficulty in constructing an MST and gives a localized algorithm to construct a low-weight connected subgraph (that can have cycles) for topology control in wireless ad hoc networks.

In this paper, we study a class of *simple, local, distributed, approximation* algorithms called the *Nearest Neighbor Tree (NNT) algorithms* that are provably good: they build slightly sub-optimal trees with low energy complexity and are easy to maintain dynamically. A fundamental step in all existing algorithms for the MST problem is *cycle detection*: given an edge, one needs to determine whether the edge would form a cycle with the edges already chosen. This deceptively simple operation leads to a big overhead: a significant amount of book keeping and message passing needs to be done in order to maintain the components, and answer such queries. The NNT algorithms bypass such a step completely by a very simple idea: each node chooses a unique *rank*, a quantity from a totally ordered set, and a node connects to the *nearest* node of higher rank. Observe that this immediately precludes cycles, and the only information that needs to be exchanged is the rank; also, this information does not need to be updated continuously over the course of the algorithm.

The NNT scheme is closely related to the approximation algorithm for the *traveling salesman problem* (coincidentally called Nearest Neighbor algorithm) analyzed in a classic paper by Rosenkrantz, Lewis, and Stearns [14]. Imase and Waxman [15] also used a scheme based on [14], which can also be considered a variant of the NNT scheme, to show that it can maintain an $O(\ln n)$-approximate Steiner tree dynamically assuming only node additions, but not deletions. These results can easily be used to show that NNT with *any* ranking of the nodes gives an $O(\ln n)$-approximation to MST on a metric graph, i.e., a complete graph with edge weights satisfying the triangle inequality. In contrast, in this paper, we consider a different graph model (more suitable to model wireless ad hoc networks): a random geometric graph where nodes are distributed uniformly at random in a unit square and show an expected approximation ratio of $O(1)$.

B. MST and Work Complexity

Formally, our focus is the following geometric weighted minimum spanning tree problem: given a set N of points (nodes)[1] in a plane, find a tree T spanning N such that $\sum_{(u,v)\in T} d^\alpha(u,v)$ is minimized where $d(u,v)$ is the length of edge (u,v), the Euclidean distance between u and v, and α is a small positive number. The motivation for this objective function comes from the energy requirements in a wireless communication paradigm: to transmit a signal over a distance r, the required *radiation energy* is proportional to r^α, where typically α is 2 and can range up to 4 in environments with multiple-path interferences or local noise [7], [11]. In this paper, we mainly focus on $\alpha = 2$. Thus, given a spanning tree T, the *cost (or quality)* of a spanning tree T is defined by $Q_\alpha(T) = \sum_{e\in T} |e|^\alpha$, where e denotes an edge of T, and our goal is to find a tree that minimizes the cost for a given α. Notice that when $\alpha = 1$, this problem becomes the traditional MST problem. It can easily be shown (e.g., using Kruskal's algorithmic construction [16]) that the MST which minimizes $\sum_{(u,v)\in T} d(u,v)$ also minimizes $\sum_{(u,v)\in T} d^\alpha(u,v)$ for any $\alpha > 0$. In the rest of the paper, we use the terms *cost* and *quality* interchangeably.

Two important applications of an MST in wireless networks are broadcasting and data aggregation. An MST can be used as broadcast tree to minimize energy consumption since it minimizes $\sum_{(u,v)\in T} d^\alpha(u,v)$. It was shown in [17]–[19] that broadcasting based on MST consumes energy within a constant factor of the optimum. In data aggregation, the idea is to combine the data coming from different sources enroute to eliminate redundancy and minimize the number of transmissions and thus saving energy. Some common aggregate functions are minimum, maximum, average, etc. [2]. One popular paradigm for computing aggregates is to construct a tree rooted at the sink where each node forwards its (locally) *aggregated* data collected from its subtree to its parent [20]. For such cases, MST is the optimal data aggregation tree.

Since energy is an important constraint in the setting of sensor networks, a lot of work has focused on constructing low energy subgraphs [6], [9]. However, it is counterproductive to use a lot of resources (e.g., time and energy) in order to compute a low cost subgraph, e.g., an MST; the energy *used* by the algorithm is also an important measure. Motivated by this consideration, in addition to the traditional time and message complexity of distributed algorithms, we consider a complexity term called *work complexity* defined as $w = \sum_{i=1}^{M} r_i^\alpha$ where r_i is the transmission distance for message i and M is the number of messages exchanged by the nodes to run the algorithm/protocol (this is implicit in many papers, see e.g., the survey of [6]). Thus total radiation energy is directly proportional to the work done by the algorithm.

C. Network Model

We consider a wireless network composed of n nodes distributed uniformly at random in a unit square (a popular probabilistic model for wireless ad hoc networks, e.g., see [21]). We assume that the nodes have distinct identifiers, each node has an omni-directional antenna, and a single transmission can be received by *any* node within the transmission radius (called *local broadcasting*). (We assume a directional antenna only for dynamic algorithm given in Section VII.) We utilize this broadcasting property to reduce the communications needed in our algorithm. Each node can adjust its transmission radius (power level) to any value up to a given maximum level. When the maximum transmission power of the nodes is large enough so that any two nodes can communicate directly with each other, we call it a *complete graph model*. Otherwise, we model it as a *unit disk graph (UDG)*, where two nodes u and v communicate directly if and only if $d(u,v) \le R$ for some given R; that is, there is an edge between u and v if and only if $d(u,v) \le R$. In this paper, we also refer the UDG model as *multihop setting*. This model is a popular graph model for multihop wireless networks [6]. When the nodes

[1] E.g., the nodes may represent sensors. We assume that these nodes have distinct identifiers.

TABLE I
PERFORMANCE OF GHS AND NNT ALGORITHMS

	Expected quality $E[Q_\alpha]$, $\alpha = 1$	Expected quality $E[Q_\alpha]$, $\alpha = 2$	Expected work $E[W]$, $\alpha = 2$	Expected msgs, $E[M]$	Time T
Co-NNT	$O(\sqrt{n})$	$O(1)$	$O(1)$	$O(n)$	$O(\ln^3 n)$ WHP
Rnd-NNT	$O(\sqrt{n})$	$O(\ln n)$	$O(\ln n)$	$O(n)$	$O(\ln^3 n)$ WHP
UDG-NNT	$O(\sqrt{n} \cdot \sqrt{\ln n})$	$O(\ln n)$	$O(\ln n)$	$O(n)$	$O(\sqrt{n} \cdot \ln^{3/2} n)$ WHP
GHS	$\Theta(\sqrt{n})$	$\Theta(1)$	$\Omega(\ln^2 n)$	$\Omega(n \ln n)$	$O(n \ln n)$

are uniformly distributed in a unit square, to have a connected graph with high probability, it is necessary and sufficient that R be $\Theta\left(\sqrt{\frac{\ln n}{n}}\right)$ [21]. Thus, we assume that $R = \Theta\left(\sqrt{\frac{\ln n}{n}}\right)$.

D. Our Contributions and Results

Our main contribution is a detailed theoretical and experimental study of the NNT algorithms in the context of wireless ad hoc and sensor networks for the above network models. First we present NNT algorithms for the complete graph model where the maximum transmission range of the nodes are large enough so that any pair of nodes can communicate directly with each other (cf. Section III). Depending on how the ranks are chosen, we study two NNT algorithms: Random-NNT (ranks are chosen randomly) and Coordinate-NNT (Co-NNT in short; ranks are based on coordinates of the nodes)[2]. For multihop wireless networks modeled by a unit disk graph (UDG), we present another NNT algorithm, which we refer to as UDG-NNT. Given the simple and local nature of this construction, it is quite surprising to have trees of reasonable properties. We show that the NNTs have some properties that can make them attractive for the ad hoc networks. Our main results are: (i) The tree produced by such an algorithm, called the NNT, has low cost, (ii) The NNT paradigm can be used to design a *simple dynamic algorithm* for maintaining a low cost spanning tree, and (iii) The *time*, *message* and *work* complexities of the NNT algorithms are close to the optimal in several settings.

Our performance analysis is with respect to the following metrics: the *quality of the spanning tree produced* by the NNT algorithm, and the *message*, *time*, and *work* needed by the algorithm to construct the tree. The results are summarized in Table I. Quality, work, and the number of messages are expected (average) values for all of the algorithms (including GHS), and the time complexity is the worst-case bound for all algorithms. While the NNTs are a close approximation to MST, NNT algorithms consume much less energy compared to the message-optimal GHS algorithm. The radiation energy to transmit messages is directly proportional to the work complexity. Some energy is consumed to process the messages by the electronic devices at the nodes. Energy consumption in electronic devices also depends on running time – longer the running time, more the energy consumption. Thus, the number of messages, work, and time together determines the total energy (energy consumption in transceiver electronics plus radiation energy) consumed in running an algorithm/protocol. The NNT algorithms perform significantly better in all three: the number of messages, work, and time.

Although most of our analysis are generalized to any α, we mainly focus on $\alpha = 2$ for the purpose of discussion. For quality, the case $\alpha = 1$ is also interesting by the fact that in this case, the problem becomes the traditional MST problem. Thus, we emphasize quality for $\alpha = 1$ and 2, and the work complexity for $\alpha = 2$.

Quality bounds: We show that with respect to the expected quality (or cost) of the tree, Random-NNT gives an $O(1)$ and $O(\ln n)$ approximation to MST for the case of $\alpha = 1$ and $\alpha = 2$, respectively; and UDG-NNT gives an $O(\sqrt{\ln n})$ and $O(\ln n)$ approximation, respectively. In contrast, Co-NNT gives an $O(1)$ approximation for both $\alpha = 1$ and $\alpha = 2$. Thus, NNT algorithms give good bounds on the cost of the trees, with Co-NNT being better than Random-NNT — this shows that at a cost of increased information (i.e., about the coordinates), we can get better approximations.

Message, time, and work complexity: NNT algorithms have significantly lower message, time, and work complexity compared to the message-optimal GHS algorithm which computes the exact MST. We show that the average work complexities of Co-NNT, Random-NNT, and UDG-NNT are $O(1)$, $O(\ln n)$, and $O(\ln n)$, respectively, for $\alpha = 2$, whereas the work complexity of GHS algorithm is $\Omega(\ln^2 n)$. For all of the NNT algorithms, the expected message complexity is $O(n)$, which is essentially the best possible, while GHS takes expected $\Omega(n \ln n)$ messages. The time complexity of both Random-NNT and Co-NNT is $O(\ln^3 n)$ with high probability (WHP[3]), while the running time of GHS is $O(n \ln n)$. The running time of UDG-NNT is $O(\sqrt{n} \cdot \ln^{3/2} n)$ WHP, which is time-optimal up to a polylogarithmic factor (see Section IV).

Simulation results: We also performed extensive simulations of our algorithms. We tested our algorithms on both uniformly random distributions of points, and on realistic distributions of points in an urban setting obtained from TRANSIMS [22]. Experimental results show that the work and number of messages for NNT algorithms are significantly smaller than that for an optimal MST algorithm, while the quality of the NNT trees are very close to MST. For example, for the TRANSIMS data, we found that the cost of the trees found by the NNT algorithms are within a factor of 2 of the MST, but there is more than a ten-fold saving on the work and about a five-fold saving on the number of messages.

Maintaining a low cost tree dynamically: We show that the degree of a node in NNT is $O(\ln n)$ with high probability. This property of low node degree can be used to design a simple

[2]Both are well motivated: when nodes don't know their geometric coordinates, Random-NNT is natural (in contrast most previous work (e.g., [5], [6] assume that nodes know their coordinates or their relative locations) but if nodes know their coordinate location (say, using GPS), then Co-NNT is more suitable.

[3]i.e., with probability at least $1 - 1/n^{\Omega(1)}$.

dynamic algorithm for maintaining a Random-NNT. We show that the expected number of *rearrangements*, i.e., the number of nodes whose outgoing edge must change, as a result of a node insertion or deletion is $O(\ln n)$. This dynamic algorithm does not require any complicated data structures or severe constraints on the sensors. The dynamic aspect of the NNT scheme makes them very useful in a sensor network setting, where it is very common for nodes to fail, or become alive asynchronously.

E. Other Related Work

X. Li et al. [5] give a local algorithm to construct a low-weight subgraph that has many desirable properties: connectivity (but may have cycles), sparseness, spanner, bounded degree, and planarity. They assume that the nodes need to know coordinate or at least relative positions, whereas for Random-NNT, no coordinate information is needed. Their algorithm takes $O(n)$ messages, which is asymptotically optimal, in the worst case for arbitrary node distribution. However, their low-weight structure is not a tree and can not be used for applications where a tree is needed, e.g., data aggregation. Moreover, their structure is low-weight only if the weight of an edge is interpreted as the distance between two nodes (and not as αth power of the distance, for some $\alpha > 1$). Quality of this structure is $O(n^{\alpha-1})$ in the worst case. N. Li et al. [8] devised a localized algorithm to build similar structure called local minimum spanning tree (LMST). They use only one hop neighbor information to build LMST. However LMST is not a low-weight structure even for $\alpha = 1$ [5]. Both [5] and [8] assume arbitrary node distribution. Kempe et al. [23] presented an algorithm to construct an approximate Euclidean MST (assuming uniform node distribution) using spatial gossip mechanism where they use a ranking of the nodes, similar to our Random-NNT. This algorithm achieves an expected $O(\ln n)$ approximation to the MST for $\alpha = 1$, where our NNTs gives an expected $O(1)$ approximation. They did not show any approximation factor for $\alpha > 1$ in which case approximation ratio can be significantly larger than $O(\ln n)$. The expected message complexity is $O(nf(n)\ln n)$, where $f(n)$ is some poly-logarithmic function, which can even be larger than the number of messages in GHS algorithm.

II. A Local Distributed Algorithm for Construction of Approximate MST

In this section, we describe the NNT scheme to construct a low cost spanning tree, where each node chooses a rank, and connects to the closest node of higher rank. An abstract form of the scheme is given in Algorithm 1. For a node v, let $nnt(v)$ denote the node that v connects to, if it exists. If v has the highest rank, $nnt(v)$ is not defined. If $nnt(v)$ is defined, it must be the case that $rank(nnt(v)) > rank(v)$ and $rank(v) > rank(w)$, for each node w that is closer to v than $nnt(v)$. If we think of the edges $(v, nnt(v))$ as being directed from v to $nnt(v)$, it is clear that each edge is directed from a low rank node to a higher rank node — this immediately rules out cycles, and gives a spanning tree. Thus, the NNT algorithm is extremely simple, local, requires no complex synchronization among the nodes, and is naturally robust.

For a given choice of ranks, let $NE(v)$ denotes the size of the neighborhood that v needs to look for in order to find the connecting edge. $NE(v)$ is a measure of the locality, and has a bearing on the time and message complexity. For arbitrary choices of ranks, the average neighborhood size could be $\Omega(n)$; but it decreases significantly for Random-NNT.

Algorithm 1 Basic NNT Scheme

All nodes have distinct ids from a totally ordered set.
Output: A spanning tree.
Every node v executes the following steps independently:

1) Choose a unique rank $rank(v)$.
2) Connect to the nearest node w such that $rank(w) > rank(v)$, i.e., add the edge (v, w) to the NNT.

We consider the following two rankings of the nodes:
Random-NNT:

1) v chooses $p(v)$, a uniform random number $\in [0, 1]$.
2) $rank(w) > rank(v)$ if $p(w) > p(v)$ or if $p(w) = p(v)$ and $id(w) > id(v)$.

Coordinate-NNT:

1) Assume that V is a set of points in a plane. $rank(v) = (x(v), y(v))$, i.e., the coordinates of v.
2) For two nodes v and w, $rank(w) > rank(v)$ if $x(w) > x(v)$ or if $x(w) = x(v)$ and $y(w) > y(v)$.

Lemma 1: In the case of a random-NNT, for any node v, the probability that v connects to the ith nearest neighbor (NN) is $\frac{1}{i(i+1)}$ and $E[NE(v)] = \Theta(\ln n)$.

Proof: Let x_0 be the random number generated by v and x_i the random number generated by the ith NN of v. Then, the probability that v connects to the i^{th} NN is equal to the probability that x_i and x_0 are the largest and second largest, respectively, among $(i+1)$ random numbers: $x_0, x_1, \ldots, x_i$. This probability is $\frac{1}{i(i+1)}$. Now, $E[NE(v)] = \sum_{i=1}^{n-1} i \cdot \frac{1}{i(i+1)} = H_n - 1 = \Theta(\ln n)$. ■

III. Distributed Implementation of the NNT Scheme

In these section, we describe an algorithm to construct a spanning tree based on the NNT scheme. In this implementation, We assume that each node *can* communicate directly with all other nodes by suitably increasing its transmission radius. However, it turns out that most of the nodes need to communicate with only a small number of nearby neighbors, but some nodes may need to communicate with distant nodes. For the case where the maximum power level of the nodes is not large enough to reach another node at that distance, we provide an alternative implementation of the NNT scheme in Section IV.

A. The NNT Algorithm

The algorithm consists of exchanging three types of messages: *request*, *available*, and *connect* among the nodes. Each node begins with broadcasting a *request* for connection message. Each node broadcasts *request* messages successively *in phases* to the distances $\frac{2}{\sqrt{n}}, \frac{4}{\sqrt{n}}, \frac{8}{\sqrt{n}}, \ldots$, until it finds a node with higher rank. The highest ranked node among all the nodes, can never find a node with higher rank. This node stops transmitting *request* message when it reaches the maximum possible distance between any two nodes. Considering a unit square, the maximum distance between any two nodes is $\sqrt{2}$. A *Request* message carries rank information (coordinates or random number). The other nodes who can hear the message send back an *available* message if

their rank is higher. The sender of the *request* message selects the nearest node from the senders of *available* messages if more than one *available* message is received, and thus it finds the nearest higher ranked node.

We assume that these phases are synchronized; i.e., all nodes begin Phase i, for each i, simultaneously. The phases can easily be synchronized by making all nodes wait for T_i time in Phase i, where T_i is the time required to complete exchanging the messages of Phase i. In the proof of Theorem 6 and 10, for Random-NNT and Co-NNT, respectively, we show how T_i can be calculated. If there is a node of higher rank within the transmission radius of Phase i, the reply from that node is received by the end of the Phase i.

When coordinates are not available (e.g., for Random-NNT), senders include the transmission power levels in the *available* messages and the recipient determine the relative distances of the senders from these power levels and the signal-strengths of the received messages. Finally, the node sends a *connect* message to the nearest higher ranked node, creating an edge between these two nodes. The details are given in Algorithm 2.

Algorithm 2 Distributed NNT algorithm for wireless networks

/* The algorithm is executed by each node u independently and simultaneously. Messages are written in the format ⟨msg name, sender, [recipient], [other info]⟩. When a message is broadcasted, the recipient is not specified. ℓ is the maximum possible distance between any two nodes.*/
$i \leftarrow 1$
Repeat
  Set transmission radius (power level) $r_i \leftarrow \frac{2^i}{\sqrt{n}}$
  If $r_i > \ell$, set $r_i \leftarrow \ell$
  Broadcast $\langle \text{request}, u, rankinfo \rangle$ // rankinfo is the random
    // number $p(u)$ & ID for Random-NNT
    // and coordinates (x_u, y_u) for Co-NNT
  $i \leftarrow i + 1$
until (receipt of an *available* message) or ($r_i = \ell$)
For all v, upon receipt of $\langle \text{request}, v, rankinfo \rangle$ do
  if $rank(v) > rank(u)$,
    set transmission radius to $distance(u, v)$
    send $\langle \text{available}, u, v \rangle$ to v
Upon receipt of "available" message(s):
  Select the nearest node v from the senders
  Send $\langle \text{connect}, u, v \rangle$ to v

B. Analysis of the NNT Algorithms

We measure the quality of the tree produced by NNT, $Q_\alpha(T) = \sum_{(u,v)\in T} d^\alpha(u,v)$, the work complexity $w = \sum_{i=1}^{M} r_i^\alpha$, the number of messages M, and the time complexity of NNT algorithms. Although our analysis generalizes to any α, for clarity we consider $\alpha = 1$ and 2.

It is known that $E[Q_1(MST)]$ is asymptotically $\Theta(\sqrt{n})$ and $E[Q_2(MST)]$ is asymptotically $\Theta(1)$ [24], [25]. We show that for Co-NNT, $E[Q_1] = O(\sqrt{n})$ and $E[Q_2] = O(1)$ giving an approximation factor of $O(1)$ for both of them. For Random-NNT, $E[Q_1] = O(\sqrt{n})$ and $E[Q_2] = O(\ln n)$ giving approximation factors of $O(1)$ and $O(\ln n)$, respectively. The expected work complexities for Random-NNT and Co-NNT (for $\alpha = 2$) are $O(\ln n)$ and $O(1)$, respectively. For both NNT algorithms, the expected number of messages is $O(n)$ and the time complexity is $O(\ln^3 n)$ WHP. The following lemmas and theorems prove these claims. We prove the bounds on work and message complexity assuming that each message is transmitted successfully in one attempt. Then we provide a protocol for scheduling messages and resolving conflicts, and show that with this protocol, the bounds can only be increased by a constant factor.

1) Analysis of Random-NNT:

***Theorem** 2: $E[Q_\alpha(\text{Random-NNT})]$ is $O(\ln n)$ for $\alpha = 2$, $O(n^{1-\alpha/2})$ for $\alpha < 2$, and $O(1)$ for $\alpha > 2$.*

Proof: Consider an arbitrary node u, and concentric circles centered at u with radius $r_i = \frac{2^i}{\sqrt{n}}$ for $i = 1, 2, \ldots, m$. Considering a unit square, the maximum distance between any two nodes is $\sqrt{2}$. Thus, $r_{m-1} < \sqrt{2} \le r_m$, i.e., the maximum number of circles $m < \frac{1}{2}\lg n + \frac{3}{2}$. Let C_i be the set of nodes in the circle with radius r_i, $R_i = C_i - C_{i-1}$ for $i \ge 2$, and $R_i = C_i$ for $i = 1$. For a node $v \in R_i$, distance $d(u,v) \le r_i$.

Let A_i be the event that u connects to a node $v \in R_i$. By Lemma 1, the probability that u connects to any node between jth nearest neighbor (NN) and $(k-1)$st NN is $\sum_{i=j}^{k-1} \frac{1}{i(i+1)} = \frac{1}{j} - \frac{1}{k}$, where $j \le k$. For $i \ge 2$, $|C_{i-1}| \ge 1$ since C_{i-1} contains at least one node, which is u. Probability that a particular node, other than u, is in C_{i-1} is $p \ge \frac{1}{4}\pi r_{i-1}^2 = \frac{2^{2i}\pi}{16n}$ (for a node at the corner or next to the border, probability p can be as low as $\frac{1}{4}$ of the area of the circle with radius r_{i-1}). Thus for $i \ge 2$,

$$\begin{aligned}
\Pr\{A_i\} &= \sum_{j=1}^{n}\sum_{k=j}^{n}\left(\frac{1}{j}-\frac{1}{k}\right)\Pr\{|C_{i-1}| = j \wedge |C_i| = k\} \\
&\le \sum_{j=1}^{n}\frac{1}{j}\Pr\{|C_{i-1}| = j\} \\
&= \sum_{j=1}^{n}\frac{1}{j}\binom{n-1}{j-1}p^{j-1}(1-p)^{n-j} \\
&= \frac{1}{np}\left\{1-(1-p)^n\right\} \\
&\le \frac{1}{np} \le \frac{16}{2^{2i}\pi}.
\end{aligned}$$

$$\begin{aligned}
E[d^\alpha(u,v)] &\le \Pr\{A_1\}r_1^\alpha + \sum_{i=2}^{m}\Pr\{A_i\}r_i^\alpha \\
&\le r_1^\alpha + \sum_{i=2}^{m}\frac{16}{2^{2i}\pi}r_i^\alpha \\
&= n^{-\alpha/2}\left\{2^\alpha + \frac{16}{\pi}\sum_{i=2}^{m}2^{(\alpha-2)i}\right\}.
\end{aligned}$$

By linearity of expectation for n nodes,

$$E[Q_\alpha] = nE[d^\alpha(u,v)].$$

When $\alpha = 2$,

$$E[Q_\alpha] \le \frac{8}{\pi}\lg n + \frac{24}{\pi} + 4 = O(\lg n) = O(\ln n).$$

When $\alpha \ne 2$,

$$E[Q_\alpha] \le \left\{2^\alpha - \frac{2^{2+2\alpha}}{\pi(2^\alpha - 4)}\right\}n^{1-\alpha/2} + \frac{2^{1+5\alpha/2}}{\pi(2^\alpha-4)}.$$

For $\alpha < 2$, $E[Q_\alpha] = O(n^{1-\alpha/2})$; for $\alpha > 2$, $E[Q_\alpha] = O(1)$. ■

***Theorem** 3:* The expected work complexity of Random-NNT algorithm $E[W]$ is $O(\ln n)$ for $\alpha = 2$, $O(n^{1-\alpha/2})$ for $\alpha < 2$, and $O(1)$ for $\alpha > 2$.

Proof: Again consider an arbitrary node u. First transmission radius for *request* message is $r_1 = 2\sqrt{\frac{1}{n}}$ and for the ith transmission, $r_i = 2r_{i-1} = 2^i\sqrt{\frac{1}{n}}$. Then, the maximum number of transmissions, $m < \frac{1}{2}\lg n + \frac{3}{2}$. Let C_i be the set of nodes in the circle centered at u with radius r_i and $R_i = C_i - C_{i-1}$, the set of nodes in the ith ring. Let $A(v,u,i)$ be the event that v replies to u in phase i. For $i \geq 2$, the event $A(v,u,i)$ occurs iff $v \in R_i$ and $rank(v) > rank(u) > rank(s)$ for all $s \in C_{i-1}$. The probability that a particular node is in C_{i-1} is $p \geq \frac{\pi 2^{2^i}}{16n}$, and $\Pr\{v \in R_i\} \leq 3p$. Letting $|C_{i-1}| = k$, we have $\Pr\{\forall_{s\in C_{i-1}}\, [rank(v) > rank(u) > rank(s)]\, | v \in R_i\} = \frac{1}{k(k+1)}$. Then for $i \geq 2$,

$$\begin{aligned}\Pr\{A(v,u,i)\} &\leq 3p\sum_{k=1}^{n-1}\frac{1}{k(k+1)}\binom{n-2}{k-1}p^{k-1}(1-p)^{n-k-1}\\ &\leq \frac{3}{n(n-1)p} \leq \frac{48}{\pi(n-1)2^{2i}}.\end{aligned}$$

$$\begin{aligned}\Pr\{A(v,u,1)\} &= \Pr\{v \in C_1, rank(v) > rank(u)\}\\ &\leq \frac{4\pi}{n}\cdot\frac{1}{2} = \frac{2\pi}{n}.\end{aligned}$$

Potentially, there are $n-1$ nodes that can reply to u. Thus, by linearity of expectation, the expected work done by the all replies to u is less than or equal to

$$\begin{aligned}&(n-1)\left\{\frac{2\pi}{n}r_1^\alpha + \sum_{i=2}^{m}\frac{48}{\pi(n-1)2^{2i}}r_i^\alpha\right\}\\ &\leq n^{-\alpha/2}\left\{2\pi 2^\alpha + \frac{48}{\pi}\sum_{i=2}^{m}2^{i(\alpha-2)}\right\} \qquad (1)\end{aligned}$$

Now we calculate the work done by the *request* and *connect* messages. Let T_i denotes the event that u needs ith transmission. $\Pr\{T_1\} = 1$. For $i \geq 2$, u needs ith transmission if and only if rank of u is the largest among all nodes in C_{i-1}. Thus,

$$\Pr\{T_i\} = \sum_{k=1}^{n}\frac{1}{k}\binom{n-1}{k-1}p^{k-1}(1-p)^{n-k} \leq \frac{16}{2^{2i}\pi}$$

In each phase, there is 1 *request* message, and at most 1 *connect* message by u. Thus expected work done by u for *request* and *connect* messages is

$$\sum_{i=1}^{m}\Pr\{T_i\}2r_i^\alpha \leq n^{-\alpha/2}\left\{2\times 2^\alpha + \frac{32}{\pi}\sum_{i=2}^{m}2^{i(\alpha-2)}\right\} \qquad (2)$$

From Eq. 1 and 2, the expected total work for node u,

$$E[W_u] \leq n^{-\alpha/2}\left\{2(\pi+1)2^\alpha + \frac{80}{\pi}\sum_{i=2}^{m}2^{i(\alpha-2)}\right\}$$

Expected work by the algorithm, $E[W] = nE[W_u]$. Thus,

$$E[W] \leq n^{1-\alpha/2}\left\{2(\pi+1)2^\alpha + \frac{80}{\pi}\sum_{i=2}^{m}2^{i(\alpha-2)}\right\} \qquad (3)$$

This gives the desired result stated in the theorem. ■

***Corollary** 4:* For $i \geq 2$, the expected number of nodes that needs ith transmission is $n\Pr\{T_i\} \leq \frac{16n}{4^i\pi}$, and the expected number of required transmissions by a node to find a higher ranked node is $\sum_{i=1}^{m}\Pr\{T_i\} \leq 1 + \frac{4}{3\pi}(1-\frac{1}{2n}) \leq 1 + \frac{4}{3\pi} < 1.425$.

***Theorem** 5:* The expected message complexity of Random-NNT algorithm is $O(n)$.

Proof: If we consider work needed for every message is 1, i.e., when $\alpha = 0$, the total work is simply the number of messages, M, exchanged in the algorithm. Thus, from Equation 3, putting $\alpha = 0$ in the right hand side, we get

$$E[M] \leq n\left\{2(\pi+1) + \frac{80}{\pi}\sum_{i=2}^{m}2^{-2i}\right\} = O(n).$$

■

Scheduling Messages and Resolving Collisions. Consider an arbitrary node u. Let k_i be the number of messages (including the messages to be transmitted by u) that can potentially collide with u's messages in Phase i of the algorithm. Note that u may have more than one among these k_i messages. In the proof of the next theorem, we show how u can determine an upper bound on k_i. Let F_i be a time frame containing at least k_i time slots. Each phase i of the algorithm contains $O(\ln n)$ such time frames F_i. In the first time frame, for each message, u chooses a slot uniformly at random. If a message is in collision, u again picks a random slot in the next frame for this message, and so on.

Now, assume $k_i \geq 2$. If $k_i \leq 1$, there is no collision. In an attempt, a particular message does not collide with probability at least $(1-1/k_i)^{k_i-1} \geq \frac{1}{e}$, using the known inequality $(1 + t/k)^k \geq (1 - t^2/k)e^t$ with $t = -1$. Thus, the expected number of retransmissions required for a message is at most e, meaning the bounds on the expected message and work complexity increase by a factor of at most e, a constant. Further, following Corollary 7 in [26], with high probability (WHP), we have all k_i messages successfully transmitted within $O(\ln n)$ time frames, i.e., WHP, the time to complete Phase i is given by,

$$T_i = O(k_i \ln n) \qquad (4)$$

***Theorem** 6:* The time complexity of Random-NNT algorithm is $O(\ln^3 n)$ with high probability.

Proof: The radius of the first transmission by each node is $r_1 = \frac{2}{\sqrt{n}}$. The expected number of nodes within this radius, $E[|C_1|] \leq \pi r_1^2 n = 4\pi$. Using the following standard Chernoff bound [27],

$$\Pr\{x \geq (1+\delta)\mu\} < \left(\frac{e^\delta}{(1+\delta)^{1+\delta}}\right)^\mu$$

with $x = |C_1|, \mu = E[|C_1|]$ and $\delta = \frac{c\ln n}{\mu} - 1$, we can show that with high probability, $|C_1| < c\ln n$ for sufficiently large constant c, and each node sends at most $c\ln n$ *available* messages. Thus, the total number of message by the nodes within the radius r_1 is $k_1 = (c\ln n)^2 = O(\ln^2 n)$ WHP. By Equation 4, time to complete the first phase is at most $T_1 = O(\ln^3 n)$ W.H.P.

Now consider an ith transmission phase to distance $r_i = \frac{2^i}{\sqrt{n}}$. After the $(i-1)$st phase, the distance between any two unconnected nodes is at least r_{i-1}; otherwise, one node has lower rank than the other and would connect to that in some previous phase. Thus, the maximum number of unconnected nodes in any circle with radius r_i is $O(1)$ (this is the maximum number of nodes that can be packed in a circle with radius $2d$, for any d, such that distance between any two nodes is at least d. Notice that there can be at most one node in any square with side $d/2$). Next we show that each such unconnected node receives at most $O(\ln n)$ *available* messages W.H.P.

Consider an arbitrary node u. Let $C_i = y$. Assume that $y \geq 60\ln n$. (If $y < 60\ln n$, then u receives $O(\ln n)$ *available*

messages with probability 1.) Let x denotes the number of nodes in C_{i-1}. For any node v, $\Pr\{v \in C_{i-1} | v \in C_i\} \geq \frac{1}{4}$ (inequality, instead of equality, comes from the fact that u can be close to the borders). Thus, $E[x] \geq y/4 \geq 15 \ln n$. Since the position of the nodes are independent and identically distributed, using the standard Chernoff bound with $\delta = \frac{1}{2}$ and $\mu = E[x]$, we have

$$\Pr\{x < y/8\} \leq \Pr\{x < (1-\delta)\mu\} < \left(\frac{e^{-\delta}}{(1-\delta)^{1-\delta}}\right)^{\mu} \leq \frac{1}{n^{2.3}}$$

Let $z = |R_i| = y - x$. Then $\Pr\{z \geq 7y/8\} = \Pr\{x < y/8\} < \frac{1}{n^{2.3}}$. Since u is at the ith transmission phase, it is known that u has the largest rank among the x nodes in C_{i-1}. Now, u receives exactly t *available* messages iff exactly t out of z nodes in R_i have higher ranks than u. The probability of such event is

$$\binom{z}{t} \frac{t!(y-t-1)!}{y!} \leq \left(\frac{z}{y}\right)^t$$

Let A be the event that u receives more than $20 \ln n$ *available* messages, and B be the event that $z < 7y/8$. Then,

$$\Pr\{A\} \leq \sum_{t=\lceil 20 \ln n \rceil}^{z} \left(\frac{z}{y}\right)^t \leq \frac{y}{y-z} \left(\frac{z}{y}\right)^{20 \ln n}$$

$$\Pr\{A|B\} < \frac{8}{n^{2.6}}$$

$$\Pr\{A\} \leq \Pr\{A|B\} + \Pr\{\bar{B}\} < \frac{1}{n^{2.3}} + \frac{8}{n^{2.6}}$$

Excluding the first phase, there are at most $\frac{1}{2}(\lg n + 1)$ phases. By the union bound (i.e., Boole's inequality [27]), the probability that some of the n nodes receives more than $20 \ln n$ replies in some phase is less than

$$\frac{1}{2}(\lg n + 1) n \left(\frac{1}{n^{2.3}} + \frac{8}{n^{2.6}}\right) = O\left(1/n^{1.2}\right)$$

Thus, to apply Equation 4, we have $k_i = O(\ln n)$ and $T_i = O(\ln^2 n)$ WHP. Therefore, total time taken by all $\frac{1}{2}(\lg n + 1)$ phases is $O(\ln^3 n)$. ■

We note that only a very few nodes may need to go far to find a node of higher rank. Most of the nodes are connected to the closer neighbors. From Corollary 4, we see that the number of nodes that need i^{th} transmission is decreasing exponentially with i. The average number of transmissions by a node is at most 1.425. Thus, almost all of the nodes get connected after the first few transmissions. The radii for the first few transmissions are $\frac{2}{\sqrt{n}}$, $\frac{4}{\sqrt{n}}$, etc., which are very small and decreasing with n. This shows that the proposed algorithm is highly scalable and local in nature.

2) Analysis of Coordinate NNT: Now, we show analogous theorems for Co-NNT.

***Theorem** 7:* The expected quality of Co-NNT for $\alpha = 1$ and 2 are $O(\sqrt{n})$ and $O(1)$, respectively.

Proof: Consider an arbitrary node u and a vertical line through u (the thick leftmost vertical line shown in Fig. 1). This vertical line divides the given unit square into two rectangles. The left rectangle is not shown in the figure. Node u connects to the nearest node in the right rectangle, which is shown in Fig. 1. For the purpose of analysis, let us subdivide this right rectangle into square cells, where each side of each cell is $b = \frac{1}{\sqrt{n}}$. If necessary, include empty space from the right hand side of the unit square to make the cells in the rightmost column squares with sides $b = \frac{1}{\sqrt{n}}$.

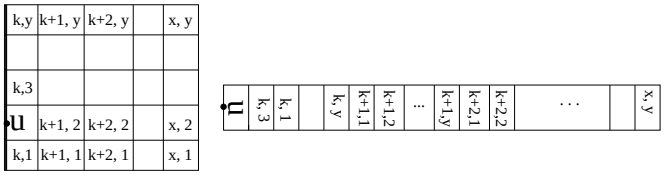

Fig. 1. The leftmost thick vertical line through node u divides the unit square into two rectangles. The right rectangle, which is shown in the figure, is subdivided into square cells with sides $b = \frac{1}{\sqrt{n}}$, by adding empty space to the rightmost column if necessary. Here $x \leq \sqrt{n}$ and $y = \sqrt{n}$. Then the cells are rearranged in a single row.

We further rearrange the cells in these columns, along with the nodes in it, in a single row as shown in Fig. 1. The cells in the column containing u, Column k, are arranged in a different way than the cells in the other columns. First, we put the cell containing u, then one cell from above u and one cell from below u, by interleaving them. Then we put the cells in Column $k+1$ in their original order beginning from the bottommost cell to the topmost cells, then the cells in Column $k+2$ in the same order, and so on. In this new arrangement, we are moving the nodes further away and increasing the distances among the nodes; and thus, increasing the length of the edges comparing to the original Co-NNT. As a result, the expected quality of the original Co-NNT is less than that of the Co-NNT in this new arrangement. Node u connects to a node in the ith next cell, if the next $i-1$ cells are empty and there is a node in the ith next cell. The probability that the next $i-1$ cells are empty is $\left(1 - \frac{i-1}{n}\right)^{n-1}$. Let P' be the probability that there is a node in the ith cell given that the first $i-1$ cells are empty, and P_i be the probability that u connects to a node v in the ith cell.

$$P_i = \left(1 - \tfrac{i-1}{n}\right)^{n-1} P' \leq \left(1 - \tfrac{i-1}{n}\right)^{n-1} \leq e^{-\frac{i-1}{n}(n-1)} \leq e^{-\frac{i-1}{2}}$$

$$E[d^\alpha(u,v)] \leq \sum_{i=1}^{n-1} (ib)^\alpha P_i \leq \sum_{i=1}^{n-1} \left(\tfrac{i}{\sqrt{n}}\right)^\alpha e^{-\frac{i-1}{2}}$$

$$E[Q_\alpha] = nE[d^\alpha(u,v)] \leq n^{1-\alpha/2} \sum_{i=1}^{n-1} i^\alpha \left(\tfrac{1}{\sqrt{e}}\right)^{i-1}$$

$$E[Q_1] \leq \sqrt{n}\sqrt{e} \sum_{i=1}^{n-1} i \left(\tfrac{1}{\sqrt{e}}\right)^i = O(\sqrt{n})$$

$$E[Q_2] \leq \sqrt{e} \sum_{i=1}^{n-1} i^2 \left(\tfrac{1}{\sqrt{e}}\right)^i = O(1)$$

■

***Theorem** 8:* The expected work complexity of Co-NNT algorithm, for $\alpha = 1$ and 2 are $O(\sqrt{n})$ and $O(1)$, respectively.

Proof: Again we subdivide the area into cells and consider the rearrangement of the cells in a single row as described in the proof of Theorem 7. The transmission radius for ith phase is $\frac{2^i}{\sqrt{n}}$. Length of each cell is $b = \frac{1}{\sqrt{n}}$. Thus, a node u need ith transmission if the next 2^{i-1} cells are empty. Let T_i be the event that u needs ith transmission. $\Pr\{T_1\} = 1$ and for $i \geq 2$,

$$\Pr\{T_i\} = \left(1 - \frac{2^{i-1}}{n}\right)^{n-1} \leq e^{-2^{i-1}/2}$$

The number of *available* messages u receives in phase i is the number of nodes in $2^i - 2^{i-1} = 2^{i-1}$ cells that are covered by the transmission i but not by transmission $i-1$. For $i \geq 2$, the expected number of such nodes in these 2^{i-1} cells, given that the first 2^{i-1} cells are empty, is

$$\frac{2^{i-1}}{n - 2^{i-1}}(n-1) \leq 2^{i-1} \frac{n}{n - 2^{i-1}}$$

$$= 2^{i-1}\left(1+\frac{2^{i-1}}{n-2^{i-1}}\right)$$
$$\leq 2^{i-1}\left(1+2^{i-1}\right)$$

The expected number of replies in the first phase is $\frac{2}{n}(n-1) \leq 2$. In addition, in each phase, there are at most one *request* message and one *connect* message by u. Thus, the expected work by u,

$$E[W_u] \leq (2+2)(2b)^\alpha \Pr\{T_1\} + \sum_{i=2}^{\lceil \lg n \rceil} \left\{2+2^{i-1}\left(1+2^{i-1}\right)\right\}(2^i b)^\alpha \Pr\{T_i\}$$
$$\leq 4\frac{2^\alpha}{n^{\alpha/2}} + \frac{2^\alpha}{n^{\alpha/2}}\sum_{i=2}^{\infty}(2+i+i^2)i^\alpha\left(\frac{1}{\sqrt{e}}\right)^i$$

The total work by n nodes, $E[W] = nE[W_u]$. Thus,

$$E[W] \leq 2^\alpha n^{1-\alpha/2}\left\{4+\sum_{i=2}^{\infty}(2+i+i^2)i^\alpha\left(\frac{1}{\sqrt{e}}\right)^i\right\} \quad (5)$$

Putting $\alpha = 1$ and 2, we have the desired result. ■

***Theorem** 9:* The expected message complexity of Co-NNT algorithm is $O(n)$.

Proof: When $\alpha = 0$, the total work is equal to the number of messages M. Thus, from Equation 5, using $\alpha = 0$, we have $E[M] = O(n)$. ■

***Theorem** 10:* The time complexity of distributed Co-NNT algorithm is $O(\ln^3 n)$ with high probability.

Proof: A part of the proof of this theorem is similar to the proof of the Theorem 6. Using the same argument as in Theorem 6, (1) the running time for the first phase of Co-NNT algorithm is $T_1 = O(\ln^3 n)$ W.H.P., (2) after $(i-1)$st phase, the maximum number of unconnected nodes in any circle of radius r_i is constant, $O(1)$. Next we show that in phase i, each unconnected node receives $O(\ln n)$ *available* messages W.H.P. The number of unconnected nodes in C_i and the number of *available* messages received by an unconnected node jointly determine the running time of ith phase.

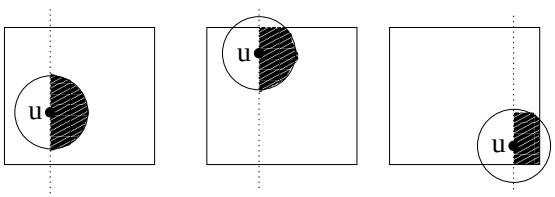

Fig. 2. Three cases for a node u in the unit square. In phase i, the radius of the circle centered at u is $r_i = 2^i/\sqrt{n}$; only the nodes in the shaded region reply back to u.

Assume a vertical line through node u (the dotted line in Fig. 2), which divides the plane that contains the unit square into two half-planes. Let B_i denotes the common region (the shaded region in Fig. 2) among the right half-plane, the disk with radius r_i centered at u, and the unit square. Let a_i be the area of the region B_i. Using simple geometry, it can easily be shown that $2a_{i-1} \leq a_i \leq 4a_{i-1}$ for any position of u in the unit square. Let n_i be the number of nodes in B_i excluding u. Now, we consider the following two cases.

Case $a_i \leq \frac{12\ln n}{n-1}$: Then, $E[n_i] \leq 12\ln n$. By Chernoff bound [27] with $\mu = E[n_i]$ and $\delta = \frac{36\ln n}{\mu} - 1$, we have

$$\Pr\{n_i \geq 36\ln n\} = \Pr\{n_i \geq (1+\delta)\mu\} < (e^\delta(1+\delta)^{-(1+\delta)})^\mu \leq (e/(1+\delta))^{(1+\delta)\mu} \leq 1/n^3.$$

The number of replies u receives in phase i cannot be more than n_i. Thus, the probability that u receives more than $36\ln n$ replies is at most $1/n^3$.

Case $a_i > \frac{12\ln n}{n-1}$: Then, $a_{i-1} \geq \frac{a_i}{4} > \frac{3\ln n}{n-1}$. In this case, the probability that u needs ith transmission, i.e., the probability that B_{i-1} is empty, is

$$(1-a_{i-1})^{n-1} \leq e^{-(n-1)a_{i-1}} < 1/n^3.$$

Thus, with probability at least $1-\frac{1}{n^3}$, either u does not need phase i or the number of replies in phase i is $O(\ln n)$. This statement holds simultaneously for all of $O(\ln n)$ phases for all n nodes with probability at least $1-\frac{1}{n}$ (by the union bound). Thus, by Equation 4, the running time of each phase $i \geq 2$ is $T_i = O(k_i \ln n) = O(\ln^2 n)$ and the total time is $O(\ln^3 n)$ WHP. ■

IV. NNT Algorithm for Multihop Wireless Networks

If the maximum power level of a node is not large enough to communicate directly with the other nodes, i.e., for the unit disk graph (UDG) model, we propose the following algorithm to construct an NNT, called UDG-NNT. Two nodes u and v can communicate directly (i.e., there is an edge between them), if and only if $d(u,v) \leq R$. We assume $R = \Theta(\sqrt{\frac{\ln n}{n}})$ (see Section I-C).

Typically, in a sensor network, a special node called *sink* gathers data from the sensors and is the root of the tree. Thus we assume that a special node s, the sink, is designated to be the root of NNT[4].

A. The UDG-NNT Algorithm

The algorithm is executed in two phases. In the first phase, the nodes choose their ranks randomly as follows. This phase is initiated by the sink s. The sink picks an arbitrary large number $p(s)$ and sends this number along with its ID to its neighbors in one transmission. As soon as any other node u receives the first message from a neighbor v, it generates a random number $p(u) \in [p(v)-1, p(v))$, and transmits $p(u)$ and $ID(u)$ to its neighbors. If u receives another message later from another neighbor v', u simply stores $p(v')$ and $ID(v')$, and does nothing else. Notice that at some point, every node in the graph, except s, receives a message from at least one of its neighbors if the given unit disk graph is connected. Identifier $ID(u)$ and random number $p(u)$ constitute the rank of u. It is easy to see that at the end of the first phase, i) each node knows the ranks of all of its neighbors, ii) each node u, except the sink s, has at least one neighbor v such that $rank(u) < rank(v)$, and iii) the sink s has the highest rank.

In the second phase, each node u, except s, selects the nearest node w among its neighbors such that $rank(u) < rank(w)$ and sends a *connect* message to w to inform that (u,w) is an edge in the NNT.

[4]If no such node is designated, a leader election algorithm can be executed before running the distributed NNT algorithm.

B. Analysis

a) Quality of UDG-NNT: Since each node has at least one neighbor with higher rank, the length of any edge of an NNT is at most R. Hence $Q_\alpha(NNT) \leq (n-1)R^\alpha = O(n^{1-\alpha/2}\ln^{\alpha/2} n)$; that is, $E[Q_1(NNT)] = \Theta(\sqrt{n \ln n})$ and $E[Q_2(NNT)] = O(\ln n)$. Since $E[Q_1(MST)] = \Theta(\sqrt{n})$ and $E[Q_2(MST)] = \Theta(1)$, we have approximation ratio of $O(\sqrt{\ln n})$ and $O(\ln n)$ for $\alpha = 1$ and 2, respectively.

b) Message Complexity: In each phase, each node transmits exactly one message. Thus, the total number of messages is at most $2n = O(n)$.

c) Work Complexity: Each node transmits a message to a distance of at most R. Hence, he total work for transmitting at most $2n$ messages is $W \leq 2nR^\alpha = O(n^{1-\alpha/2}\ln^{\alpha/2} n)$. Using $\alpha = 2$, we have $W = O(\ln n)$.

d) Time Complexity: Let D be the diameter of the given unit disk graph. Using Chernoff bound, it can be shown that the number of nodes within distance R from any point is $O(\ln n)$ WHP. Since in each phase each node transmit at most one message, using Equation 4, a node needs to wait at most $O(\ln^2 n)$ time to transmit its message. Further, the rank propagation messages from the sink s to any other node in the graph traverse at most D links, leading to the running time of $O(D\ln^2 n)$ for the first phase and $O(\ln^2 n)$ for the second phase WHP. Thus, the total time is $O(D\ln^2 n)$ WHP, which is optimal up to polylogarithmic factor; because to build any spanning tree requires $\Omega(D)$ time. By a simple analysis with the help of Chernoff bound, it can easily be shown that for the unit disk graph model under consideration, $D = \Theta(1/R) = \Theta(\sqrt{n/\ln n})$ WHP, which implies a running time of $O(\sqrt{n}\cdot\ln^{3/2} n)$ WHP.

All of the above bounds hold even if a node u picks an arbitrary number (instead of a random number) $p(u) < p(v)$ after receiving the first message from a neighbor v. However, as indicated by the simulation results, the random choice may improve the bound on quality to $O(1)$ and $O(\ln\ln n)$ for $\alpha = 1$ and 2, respectively. Due to the dependencies among these random numbers, analyzing quality using this randomness seems to be difficult, which we leave for future work.

V. Lower Bounds on Expected Work and Message Complexity of GHS Algorithm

We compare our NNT algorithm with a well-studied distributed MST algorithm, the GHS algorithm [1], which is message-optimal. The upper bounds on the message and time complexity of this algorithm have been shown in [1]. Here we give lower bounds on the expected work and message complexity of implementing the GHS algorithm in our network model. We do not provide the details of the implementation of GHS algorithm here; instead, we refer to [1] for a detailed description. Here we focus on what is essential to show lower bounds on the expected work and message complexity.

We consider the radius of the neighborhood of each node to be $\Theta(\sqrt{\frac{\ln n}{n}})$. Since each node sends at least one message to each of its neighbor (*test* message — to check if the neighbor is in the same fragment), work and the number of messages of GHS algorithm increases as the number of neighbors of the nodes increases. Here we note that the way GHS algorithm works, it needs to test the neighbors sequentially. Thus, it cannot take the advantage of local wireless broadcasting. Even if we consider that a node tests all of its neighbors by broadcasting a single message to all of its neighbors, each neighbor must reply to this test message individually. Thus, the number of messages related to testing the neighbors is still no less than the number of neighbors. As a result, the best performance of the GHS algorithm can be achieved by keeping radius of neighborhood as small as possible; we choose $\Theta(\sqrt{\frac{\ln n}{n}})$, the minimum required for connectivity [21].

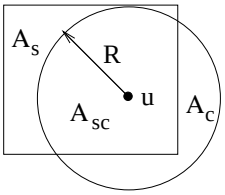

Fig. 3. Illustration for Lemma 11.

The expected number of neighbors of a node, i.e., the expected number of nodes within distance $\Theta(\sqrt{\frac{\ln n}{n}})$ is $\Theta(\ln n)$ (in fact it is true with high probability). Thus, each node exchanges at least $\Omega(\ln n)$ messages; that is, the total expected number of messages is $\Omega(n\ln n)$.

The following lemma is used to find the expected work complexity.

***Lemma** 11:* Let n nodes are distributed uniformly at random in a unit square and r_i be the distance of the ith nearest neighbor for any arbitrary node. Then, $E[r_i^2] = \frac{ci}{n} = \Theta\left(\frac{i}{n}\right)$ for some constant c where $\frac{1}{\pi} \leq c \leq 2$.

Proof: To get the lower bound, consider any arbitrary node u in the unit square. Now consider a circle centered at u with unit area, i.e., $\pi R^2 = 1$ where R is the radius of the circle. Let A_{sc} be the region that is common to both the unit square and the circle (see Fig. 3), A_s the region in the square but not in the circle, and A_c the region in the circle but not in the square. Since both the circle and the square have equal area (unit area), the area of A_s is equal to the area of A_c. Now consider a rearrangement (repositioning) of the nodes: keep the nodes in A_{sc} as they are and move all nodes in A_s to A_c; place the moved nodes in A_c randomly following the uniform distribution. Now it is easy to see that the distance to the ith nearest neighbor of u in the original arrangement of the nodes (i.e., unit square) is greater than or equal to that in the new arrangement (i.e., the ith nearest neighbor in the circle centered at u).

Now, the probability that a particular node (other than u) is within distance r from u (in the new arrangement) is $\frac{\pi r^2}{\pi R^2} = \frac{r^2}{R^2}$. Then, the probability that there are at least i nodes within distance r,

$$C_i(r) = 1 - \sum_{k=0}^{i-1} \binom{n-1}{k} \left(\frac{r^2}{R^2}\right)^k \left(1 - \frac{r^2}{R^2}\right)^{n-k-1}$$

The probability density function $P_i(r) = \frac{d}{dr}C_i(r)$, that is,

$$P_i(r) = -\sum_{k=0}^{i-1} \binom{n-1}{k} k\frac{2r}{R^2} \left(\frac{r^2}{R^2}\right)^{k-1} \left(1 - \frac{r^2}{R^2}\right)^{n-k-1} + \sum_{k=0}^{i-1} \binom{n-1}{k} (n-k-1)\frac{2r}{R^2} \left(\frac{r^2}{R^2}\right)^{k} \left(1 - \frac{r^2}{R^2}\right)^{n-k-2}$$

Let T_k be the first term inside the above sum, which is

$\binom{n-1}{k} k \frac{2r}{R^2} \left(\frac{r^2}{R^2}\right)^{k-1} \left(1-\frac{r^2}{R^2}\right)^{n-k-1}$. Then,

$$\begin{aligned} T_{k+1} &= \binom{n-1}{k+1}(k+1)\frac{2r}{R^2}\left(\frac{r^2}{R^2}\right)^{k}\left(1-\frac{r^2}{R^2}\right)^{n-k-2} \\ &= \binom{n-1}{k}(n-k-1)\frac{2r}{R^2}\left(\frac{r^2}{R^2}\right)^{k}\left(1-\frac{r^2}{R^2}\right)^{n-k-2} \end{aligned}$$

Now $T_0 = 0$, thus $P_i(r) = -\sum_{k=0}^{i-1}(T_k - T_{k+1}) = T_i$. Then,

$$\begin{aligned} E[r_i^2] &\geq \int_0^R r^2 P_i(r) dr \\ &= iR^2\binom{n-1}{i}\int_0^R \frac{2r}{R^2}\left(\frac{r^2}{R^2}\right)^{i}\left(1-\frac{r^2}{R^2}\right)^{n-i-1} dr \\ &= iR^2\binom{n-1}{i}\sum_{k=0}^{i}\binom{i}{k}(-1)^k\frac{1}{k+n-i} \end{aligned}$$

Since $n-i>0$, using the identity $\sum_{k=0}^{n}\binom{n}{k}\frac{(-1)^k}{k+x} = x^{-1}\binom{x+n}{n}^{-1}$ (page 188 in [28]),

$$E[r_i^2] \geq iR^2\binom{n-1}{i}\frac{1}{(n-i)\binom{n}{i}} = \frac{iR^2}{n} = \frac{i}{\pi n}.$$

To get the upper bound, we consider a node u in a corner of the unit square and a circle centered at u and with radius $R' = \sqrt{2}$, the length of a diagonal of the square. If we redistribute the nodes in this circle uniformly, the average distance to the ith nearest neighbor can only increase. Thus, $E[r_i^2] \leq \frac{iR'^2}{n} = \frac{2i}{n}$. ■

***Theorem** 12:* The expected work complexity of GHS algorithm is $\Omega(\ln^2 n)$.

Proof: We analyze the work complexity for *test*, *accept*, and *reject* messages only [1]. By the end of execution of the algorithm, each node tests all of its adjacent edges by using test/accept/reject messages through these edges one by one. The expected number of neighbors of each node is $c\ln n$, for some constant c. Thus, each node sends test messages to or receives reply messages from these $c\ln n$ neighbors. Using Lemma 11, the expected work by a node is at least $\sum_{i=1}^{c\ln n}\frac{i}{n\pi} = \Omega(\frac{\ln^2 n}{n})$. For n nodes, by linearity of expectation, the total work $W = n \times \Omega(\frac{\ln^2 n}{n}) = \Omega(\ln^2 n)$. ■

VI. Simulation Results

We performed extensive simulations of our algorithms to understand their empirical performance. Our experimental setup is the following:

Number of Nodes: Varying from 50 to 5000.

Node distributions: Uniform random distributions in the unit square and several realistic distributions of points in an urban setting obtained from TRANSIMS [22].

Number of Runs: 50

Measures: We compare the NNTs and the MST, with respect to the quality $Q_\alpha(T) = \sum_{(u,v)\in T} d^\alpha(u,v)$ for $\alpha = 1$ and 2 and the performance of the algorithms with respect to the following measures: (i) Number of messages, and (ii) Work, $W = \sum_{i=1}^{M} r_i^\alpha$ for $\alpha = 2$.

To be fair to GHS which does not exploit geometry per se (to compare with Co-NNT, which uses coordinate information of the nodes) we run the GHS algorithm on the Yao graph [29]. Yao graph is constructed as follows: the 2π angle around each node u is divided into six wedges of equal size (see Fig. 4), and the edge between u and the nearest node (if any) in each wedge is included in the Yao graph. A Yao graph is sparse (each node has degree at most 6) and contains the MST [29]. Running GHS on Yao graph reduces its message complexity to $O(n\ln n)$. Note that GHS-Yao should be compared with Co-NNT only as both of them use coordinate information. On the other hand, Random-NNT and UGD-NNT, which do not use coordinate information, must be compared with GHS without Yao graph.

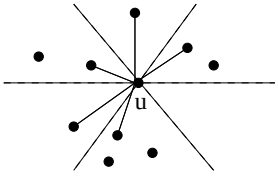

Fig. 4. Construction of Yao graph. Each wedges is 60°. Node u have an edge with the nearest node in each wedge.

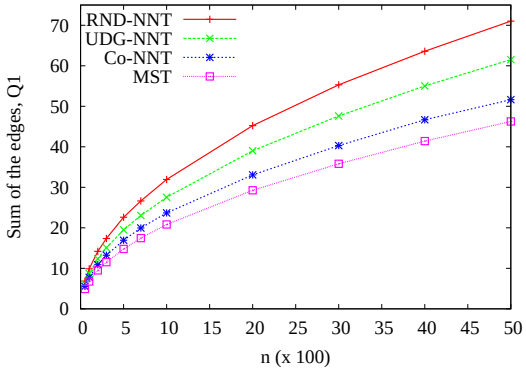

Fig. 5. Sum of the lengths of the edges, $Q_1(T)$, for MST, Random-NNT, UDG-NNT, and Co-NNT.

In our simulations, we ignore the effects of the MAC layer. Our main results are summarized below, which validate our theoretical results in earlier sections: (1) The Co-NNT algorithm always outperforms the Random-NNT algorithm, with respect to the quality, number of messages, and work. (2) For $\alpha = 1$, all NNTs give a very good approximation to the MST; in particular, the cost of Co-NNT is always within about 10% of that of the MST. (3) For $\alpha = 2$, Random-NNT does not give a very good approximation, but UDG-NNT and Co-NNT remains within a factor of 2. (4) The number of messages and the work done by both NNT algorithms are significantly smaller than that by GHS algorithm.

A. Quality of the Spanning Trees

We present the simulation results of quality $Q_\alpha(T)$ for α = 1 and 2. As Fig. 5 shows, all NNTs compare very well with the MST. As shown earlier, the MST cost is $\Theta(\sqrt{n})$ for $\alpha = 1$, and the NNTs seems to be within a small constant factor of this value; Fig. 6 demonstrates this by showing the values as a fraction of $\sqrt{n}$, and the plot for the NNTs are straight lines.

Fig. 7 shows $Q_2(T)$, the sum of squares of the edge lengths, for the NNTs and MST. Quality Q_2 for both the MST and Co-NNT are constant, and $Q_2(\textit{Co-NNT})$ is within a factor of 2 of $Q_2(MST)$. However, $Q_2(\textit{Random-NNT})$ increases with n as the asymptotic bound is $O(\ln n)$— this becomes clear from Fig. 8. In Section IV, we showed that $Q_2(\textit{UDG-NNT}) = O(\ln n)$. Fig. 7 shows that Q_2 for UDG-NNT is almost constant. In fact, by

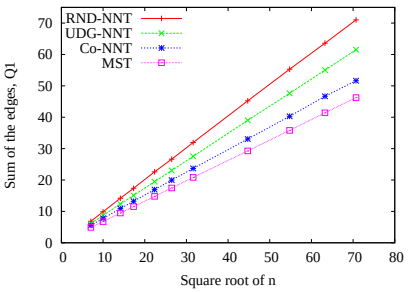

Fig. 6. Sum of the lengths of the edges, $Q_1(T)$, plotted with $\sqrt{n}$.

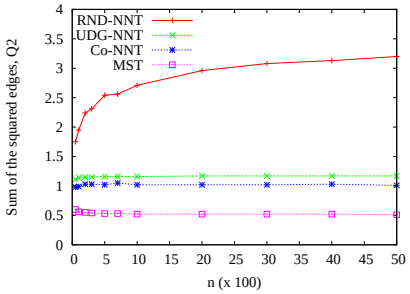

Fig. 7. Sum of the squares of the edge lengths, $Q_2(T)$.

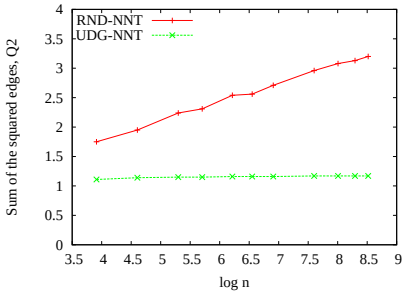

Fig. 8. $Q_2(T)$ for Random-NNT and UDG-NNT with respect to $\ln n$.

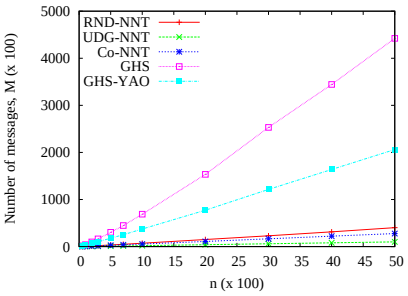

Fig. 9. Number of messages needed to construct the spanning trees.

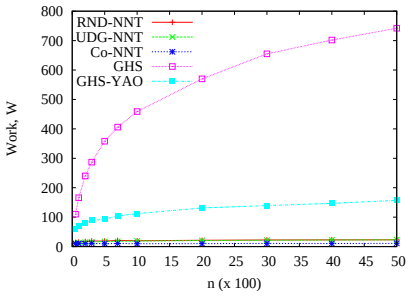

Fig. 10. Work done by the algorithms. In the figure, the lines for Random-NNT and UDG-NNT almost merged together.

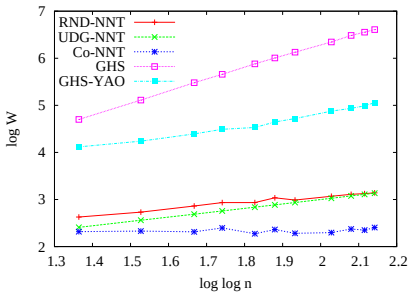

Fig. 11. Slope of the lines indicate the powers of log in work complexity.

looking at the numerical data, we found that it is increasing but at a very slow rate (which is not visible in the figure). This data indicates that a tighter bound for $Q_2(\textit{UDG-NNT})$ may exist.

B. Work and Message Complexities to Construct the Spanning Trees

In this section, we compare work W for $\alpha = 2$ and the number of messages needed by the algorithms. The input to GHS algorithm must be a connected graph to obtain a MST. We consider the radius of the neighborhood to be $1.6\sqrt{\frac{\ln n}{n}}$, the minimum required for connectivity. To determine the neighbors, each node can broadcast a message to distance $1.6\sqrt{\frac{\ln n}{n}}$ and consider another node as a neighbor if the node can hear the message from the other node. However, we did not incur any cost on GHS algorithm to find the neighbors (thus favoring GHS) — for GHS algorithm, we assume that each node knows its neighbors and their distances. In addition, we also simulate GHS on the Yao graph. Each node finds its Yao neighbors first, then executes GHS algorithm.

Fig. 9 depicts the number of messages needed to construct the trees. We see that the number of messages for NNT algorithms is significantly smaller than that for GHS algorithm. Moreover, the number of messages for NNT algorithms increases linearly. On the other hand, the number of messages for GHS increases at a slightly higher rate. In fact, the message complexity for GHS is $\Omega(n \ln n)$.

The required work for NNT algorithms is also significantly less than that of GHS algorithm (Fig. 10). In addition, with the number of nodes, work for NNT algorithms increases in a lower rate than that of GHS algorithm. In terms of both the number of messages and work, GHS with Yao graph is more efficient than GHS without Yao graph, as expected. However, it is still much less efficient than the Co-NNT algorithm.

Analytically, we know that for $\alpha = 2$, the work complexities for Co-NNT, Random-NNT, UDG-NNT, and GHS algorithms are $O(1)$, $O(\ln n)$, $O(\ln n)$, and $\Omega(\ln^2 n)$, respectively. We can also observe these results from experimental data. Let work $w = c \ln^a n$. Then, $\ln w = \ln c + a \ln \ln n$. Thus, if we plot $\ln w$ vs. $\ln \ln n$, the graph is an straight line and the slope of the line is a, the power of $\log$. In Fig. 11, the slope for GHS is greater than 2 and for both Random-NNT and UDG-NNT, it is about 1. For Co-NNT, the slope is 0, which indicates the work is $O(1)$, whereas for GHS-Yao, the slope is more than 1.

C. Experiments on Real Data

We consider a distribution of points in a section of downtown Portland, OR, measuring $2.9KM \times 2.95KM$ approximately 9 square KM. The distribution of points, corresponding to cars on the roadway, was obtained from the TRANSIMS simulation [22], which does a very detailed modeling of urban traffic, combining a variety of data sources ranging from census data to activity surveys to land use data. We use three snapshots, at one minute intervals. The distribution of nodes at one of the snapshots is shown in Fig. 12. The experimental results on these three snapshots are given in Table II. The work is computed for $\alpha = 2$.

We see that the number of messages and work are significantly larger for GHS algorithm. The work is about 10 times larger and the number of messages is about 5 times larger than those of NNT algorithms. On the other hand, both Q_1 and Q_2 for Co-NNT is within 2-approximation. Although approximation for Q_2 in Random-NNT is large, for Q_1, Random-NNT also provides a close approximation. In these experiments, we only considered the Yao graph assuming that the nodes know their coordinates. If the

TABLE II
EXPERIMENT RESULTS FOR SNAPSHOT 1, 2, AND 3

	Snapshot 1				Snapshot 2				Snapshot 3			
	Q_1	Q_2	Work	Msg	Q_1	Q_2	Work	Msg	Q_1	Q_2	Work	Msg
Co-NNT	38.72	6.77	90.54	4832	39.39	8.18	92.28	4647	38.32	6.25	83.42	4668
Rnd-NNT	50.75	14.13	131.42	5241	52.97	20.12	137.91	5250	52.57	18.47	148.88	5229
GHS-Yao	33.16	3.73	1271.11	20592	33.52	3.82	1083.99	20417	33.27	3.78	1083.99	20417

Fig. 12. The distribution of nodes in one of the snapshots.

coordinates are not available, for GHS algorithm, the input need to be a complete graph (each node is a neighbor of the others) to make sure connectivity since the points does not follow any particular (say, uniform) distribution. Thus, GHS algorithm would incur much larger work and messages. In that case, Random-NNT can still be a good choice over GHS, by sacrificing quality.

VII. DYNAMIC ALGORITHM FOR NNT

The local nature of the NNT algorithms naturally allows for simple dynamic versions, where the goal is to maintain a tree of good quality, as nodes are added or deleted. As long as we maintain an NNT, the cost remains within the bounds proven in the previous theorems. The measure we focus on, in the dynamic setting, is the expected number of rearrangements, when a node is added or deleted. We define the term *number of rearrangements* to be the number of the edges to be deleted from the tree and added to the tree, to maintain NNT, due to addition or deletion of a node.

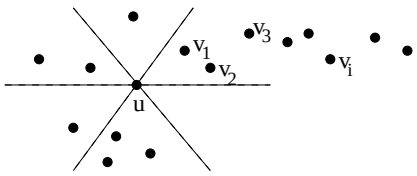

Fig. 13. Each wedge around node u is $60°$. $v_1, v_2, v_3 \ldots$ are the nodes in one wedge in increasing order of distance from u.

For the dynamic Random-NNT algorithm, each node v maintains two lists $Q(v)$ and $L(v)$, where $Q(v)$ is the set of nodes in closed ball $B(v, d(v, nnt(v)))$ and $L(v) = \{u | v \in Q(u)\}$. After a node v is added to the network, there can be some nodes u such that $rank(v) > rank(u)$ and $d(u, v) < d(u, nnt(u))$. In that case, u must change its connection from previous $nnt(u)$ to v, the new $nnt(u)$, and we say u is "affected" by v. The data structures $Q(v)$ and $L(v)$ facilitate an efficient way to find the affected nodes. Let node v just joined the network, and consider a partitioning of the 2π angle around v into 6 equal wedges as shown in Fig. 13. Let $w_1, w_2, \ldots, w_6$ be the closest nodes in these wedges; these nodes can be determined, for instance, by using directional antennas (such an assumption is made in several papers, e.g., [10]). Using the triangle inequality, it is easy to see that only the nodes in $L(w_i)$ can be affected. The details are given in Algorithm 3.

Algorithm 3 Dynamic Random-NNT

- If a node v is added:
  1) Node v chooses a random rank, and creates two lists of nodes, $Q(v)$ and $L(v)$, which are initially empty.
  2) Node v checks its neighbors $v_1, v_2, \ldots$ in non-decreasing order of $d(v, v_i)$, till it finds the closest neighbor v_j of higher rank. Then v adds each such v_i, $i \leq j$, in $Q(v)$ and sends a message to v_i to add it in $L(v_i)$.
  3) Node v finds the closest node in each of the 6 wedges. For each of these closest nodes w, v sends an UPDATE message to each $u \in L(w)$. Node u, upon receipt of this message, does the following:
     a) Let $u_1, u_2, \ldots$ be the neighbors of u in non-decreasing order of $d(u, u_i)$, and u_k be $nnt(u)$.
     b) If $d(u, v) \leq d(u, u_k)$ and $rank(v) > rank(u)$, u removes u_ℓ from $Q(u)$ and itself from $L(u_\ell)$, $\forall j < \ell \leq k$, where $v = u_j$. Further u connects to v, instead of u_k, and adds v to $Q(u)$ and itself to $L(v)$.
     c) If $d(u, v) \leq d(u, u_k)$ and $rank(v) < rank(u)$, u adds v to $Q(u)$ and itself to $L(v)$.
- If a node v is deleted:
  1) v sends a message to each $u \in Q(v)$. Node u, after receiving this message, removes v from $L(u)$.
  2) v also sends a message to each $u \in L(v)$. Node u, upon receipt of this message, removes v from $Q(u)$. If $nnt(u) = v$, u checks the neighbors v_i beginning from distance $d(u, v)$ onward until it finds the new nearest node of higher rank, adds each such v_i in $Q(v)$, and sends a message to v_i to add u in $L(v_i)$.

It is also easy to see that the lists $L(.)$ and $Q(.)$ of the nodes are correctly updated after addition and deletion of a node. Since $u \in L(v)$ iff $v \in Q(u)$, and the algorithm always updates $L(v)$ and $Q(u)$ or $L(u)$ and $Q(v)$ in pairs, it is sufficient to show that $L(v)$ and $Q(v)$ are updated correctly when node v is added or deleted. When v is added, $Q(v)$ is created in Step 2 following its definition straight-forwardly. For $L(v)$, notice that if $u \in L(v)$, then $u \in L(w_i)$ for some w_i, by the triangle inequality. Thus, Step 3b-c correctly updates $L(v)$. When v is deleted, again, the consistency of $L(v)$ and $Q(v)$ follows directly from their definitions. Thus, Algorithm Dynamic Random-NNT works correctly.

Next we analyze the number of rearrangements needed for each insertion and deletion. The complexity of a rearrangement depends on the model of the network and communication. For the purpose of analysis, For any node v, we say that a charge of 1 is placed on every node u in closed ball $B(v, d(v, nnt(v)))$. First, we show the following lemma, which bounds the charge placed on any node.

***Lemma** 13:* For any sequence of n node insertions and deletions, the total charge on any node u is $O(\ln n)$ WHP.

Proof: Consider any point u, and partition the 2π angle around u into 6 cones, each of angle $\pi/3$. Consider one such cone. We prove that the total charge from points in this cone on u is $O(\ln n)$, with high probability. Order the points in the cone as $v_1, v_2, \ldots$, based on increasing distance from u (Fig. 13). Node v_i places a charge on u only if $rank(v_i) > rank(v_j)$, for all $1 \leq j < i$. The probability of occurring this event is at most $1/i$ (the probability that a particular number is the largest among i identical random numbers is $1/i$). Thus, the total expected charge on u from these points is at most $\sum_{i=1}^{n-1}(1/i) \leq \ln n$. In order to bound the maximum charge on any node, we use a variant of Chernoff bound (Lemma 14) that holds in the presence of dependencies among the variables.

***Lemma** 14:* ([30]) Let $X_1, X_2, \ldots, X_l \in \{0, 1\}$ be random variables such that for all i, and for any $S \subseteq \{X_1, \ldots, X_i\}$, $\Pr[X_{i+1} = 1 | \bigwedge_{j \in S} X_j = 1] \leq \Pr[X_{i+1} = 1]$. Then, for any $\delta > 0$, $\Pr[\sum_i X_i \geq \mu(1+\delta)] \leq (\frac{e^\delta}{(1+\delta)^{1+\delta}})^\mu$, where $\mu = \sum_i E[X_i]$.

Let $\mathcal{E}(v)$ be the event that node v places a charge on u. In order to use the Chernoff bound, we need to show that, for any i, and any subset $S \subset \{v_1, \ldots, v_i\}$, $\Pr[\mathcal{E}(v_{i+1}) | \bigwedge_{w \in S} \mathcal{E}(w)] \leq \Pr[\mathcal{E}(v_{i+1})]$. First, suppose $d(w, v_{i+1}) \geq d(w, u)$ for each $w \in S$. Then, the events $\bigwedge_{w \in S} \mathcal{E}(w)$ do not place any constraint on $rank(v_{i+1})$, relative to $rank(v_j), j \leq i$, and therefore, $\Pr[\mathcal{E}(v_{i+1}) | \bigwedge_{w \in S} \mathcal{E}(w)] = \Pr[\mathcal{E}(v_{i+1})]$. Next, suppose $d(w, v_{i+1}) < d(w, u)$ for some $w \in S$. Then, occurrence of the event $\mathcal{E}(w)$ implies that $rank(w) > rank(v_{i+1})$. Further, $d(v_{i+1}, w) < d(w, u) \leq d(v_{i+1}, u)$. Therefore, $\Pr[\mathcal{E}(v_{i+1}) | \bigwedge_{w \in S} \mathcal{E}(w)] = 0 \leq \Pr[\mathcal{E}(v_{i+1})]$.

Next, we apply the Chernoff bound with $\delta = \frac{5 \ln n}{\mu} - 1$, where μ is the expected charge on u. Since $\mu \leq \ln n$, we have $\delta > 0$. Let X be the total charge on u. Then,

$$\begin{aligned} \Pr\{X \geq 5 \ln n\} &= \Pr\{X \geq (1+\delta)\mu\} < (e^\delta (1+\delta)^{-(1+\delta)})^\mu \\ &\leq (e/(1+\delta))^{(1+\delta)\mu} \leq 1/n^3. \end{aligned}$$

Thus, with probability at least $1 - \frac{1}{n^3}$, charge on u is $O(\ln n)$. Using the union bound, it holds simultaneously for all n steps of addition and deletion of nodes with probability at least $1 - \frac{1}{n}$. ■

***Corollary** 15:* The degree of any node v in the NNT is $O(\ln n)$ W.H.P.

Proof: The degree of v is at most the charge on v. Thus, the corollary follows from Lemma 13. ■

***Theorem** 16:* For any sequence of n node insertions and deletions, the number of rearrangements per insertion or deletion is $O(\ln n)$ WHP.

Proof: When a node v is deleted, only the nodes u such that $nnt(u) = v$ needs to find a new parent to connect to. Let $D(v)$ be the degree of v. Deletion of v results in deletion of $D(v)$ edges and addition of $D(v) - 1$ new edges. Thus the number of rearrangement is $2D(v) - 1 = O(\ln n)$ W.H.P. (Corollary 15). When a node v is added, a node $u \in L(v)$ may need to change its connection. For any other node $w \notin L(v)$, $d(w, nnt(w)) < d(w, v)$ and such node w does not need to change its connecting edge. Thus, due to addition of v, the number of rearrangements is at most $2|L(v)| = O(\ln n)$ W.H.P. by Lemma 13. ■

VIII. Concluding Remarks and Further Work

The NNT paradigm is a simple and local scheme for constructing and maintaining low cost trees in an ad hoc network setting. It does not require any complex synchronization, and is naturally robust. We study various properties, such as quality, degree and dynamic complexity for different NNTs where the nodes are uniformly distributed in two-dimensional plane, and it shows very promising results. Among the questions for further work, the most interesting ones are: (i) Analyze the NNT algorithms for arbitrary point distributions, (ii) Quantify the tradeoff between the amount of local information needed and the quality of the tree produced, (iii) Obtain a tighter bound for quality of UDG-NNT, and (iv) Determine whether it is possible to design a distributed exact MST algorithm with better work complexity than GHS, and obtain a lower bound on work complexity.

Acknowledgment

We are grateful to the referees for their careful reading of the paper and detailed comments which helped greatly in improving the paper.

References

[1] R. Gallager, P. Humblet, and P. Spira, "A distributed algorithm for minimum-weight spanning trees," *ACM Transactions on Programming Languages and Systems*, vol. 5, no. 1, pp. 66–77, January 1983.

[2] B. Krishnamachari, D. Estrin, and S. Wicker, "The impact of data aggregation in wireless sensor networks," in *Proceedings of the Int. Workshop on Distributed Event-Based Systems*, July 2002.

[3] M. Elkin, "Unconditional lower bounds on the time-approximation tradeoffs for the distributed minimum spanning tree problem," in *Proceedings of Symposium on Theory of Computing (STOC)*, June 2004.

[4] D. Peleg, *Distributed Computing: A Locality-Sensitive Approach*, SIAM, 2000.

[5] X. Li, Y. Wang, P. Wan, W. Song, and O. Frieder, "Localized low-weight graph and its applications in wireless ad hoc networks," in *Proceedings of INFOCOM, IEEE International Conference*, 2004.

[6] X. Li, "Algorithmic, geometric and graphs issues in wireless networks," *Journal of Wireless Communications and Mobile Computing*, vol. 3, no. 2, pp. 119–140, 2003.

[7] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Transactions on Wireless Communications*, vol. 1, no. 4, pp. 660–670, October 2002.

[8] N. Li, J. Hou, and L. Sha, "Design and analysis of an mst-based topology control algorithm," in *Proceedings of INFOCOM, IEEE International Conference*, 2003.

[9] M. Cheng, M. Cardei, X. Cheng, L. Wang, Y. Xu, and D. Du, "Topology control of ad hoc wireless networks for energy efficiency," *IEEE Transaction on Computers*, vol. 53, no. 12, pp. 1629–1635, December 2004.

[10] R. Wattenhofer, L. Li, P. Bahl, and Y. Wang, "Distributed topology control for power efficient operation in multihop wireless ad hoc networks," in *Proceedings of INFOCOM*, April 2001.

[11] L. Kirousis, E. Kranakis, D. Krizanc, and A. Pelc, "Power consumption in packet radio networks," *Theoretical Computer Science*, vol. 243, no. 1-2, pp. 289–305, July 2000.

[12] X. Li, G. Calinescu, and P. Wan, "Distributed construction of planar spanner and routing for ad hoc wireless networks," in *Proceedings of INFOCOM*, 2002.

[13] X. Li, P. Wan, Y. Wang, and O. Frieder, "Sparse power efficient topology for wireless networks," in *Proceedings of the 35th Annual Hawaii International Conference on System Sciences (HICSS)*, 2002.

[14] D. Rosenkrantz, R. Stearns, and P. Lewis, "An analysis of several heuristics for the traveling salesman problem," *SIAM J. Comput.*, vol. 6(3), pp. 563–581, 1977.
[15] M. Imase and B.M. Waxman, "Dynamic steiner tree problem," *Siam J. Discrete Math*, vol. 4(3), pp. 369–384, 1991.
[16] T. Cormen, C. Leiserson, and R. Rivest, *Introduction to Algorithms*, The MIT Press, 1990.
[17] C. Ambuhl, "An optimal bound for the mst algorithm to compute energy efficient broadcast trees in wireless networks.," in *Proceedings of the 32th Int. Coll. on Automata, Languages and Programming (ICALP)*, November 2005, pp. 1139–1150.
[18] P. Wan, G. Calinescu, X. Li, and O. Frieder, "Minimum-energy broadcasting in static ad hoc wireless networks," *Wireless Networks*, vol. 8, no. 6, pp. 607–617, November 2002.
[19] A. Clementi, P. Crescenzi, P. Penna, G. Rossi, and P. Vocca, "On the complexity of computing minimum energy consumption broadcast subgraph," in *Proceedings of the 18th Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, June 2001, pp. 121–131.
[20] C. Intanagonwiwat, D. Estrin, R. Govindan, and J. Heidemann, "Impact of network density on data aggregation in wireless sensor networks," in *Proceedings of International Conference on Distributed Computing Systems (ICDCS)*, July 2002.
[21] P. Gupta and P. Kumar, "Critical power for asymptotic connectivity in wireless networks," *Stochastic Analysis, Control, Optimization and Applications: A Volume in Honor of W. H. Fleming*, pp. 547–566, March 1998.
[22] "Transportation Analysis Simulation System (TRANSIMS)," *http://transims.tsasa.lanl.gov*.
[23] D. Kempe and J. Kleinberg, "Protocols and impossibility results for gossip-based communication mechanisms," in *Proceedings of The 43rd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, November 2002.
[24] J. Steele, "Asymtotics for euclidian minimal spanning trees on random points," *Probability Theory and Related Fields*, vol. 92, pp. 247–258, 1992.
[25] F. Avram and D. Bertsimas, "The minimum spanning tree constant in geometrical probability and under the independent model: a unified approach," *The Annals of Applied Probability*, vol. 2, no. 1, pp. 113–130, 1992.
[26] O. Johansson, "Simple distributed $(\Delta + 1)$-coloring of graphs," *Information Processing Letters*, vol. 70, no. 5, pp. 229232, 1999.
[27] R. Motwani and P. Raghavan, *Randomized Algorithms*, Cambridge University Press, 1995.
[28] R. Graham, D. Knuth, and O. Patashnik, *Concrete Mathematics: A Foundation for Computer Science, Second Edition*, Addison-Wesley Publishing Company, Inc., 1989.
[29] A. Yao, "On constructing minimum spanning trees in k-dimensional spaces and related problems," *SIAM Journal on Computing*, vol. 11, no. 4, pp. 721–736, 1982.
[30] A. Panconesi and A. Srinivasan, "Randomized distributed edge coloring via an extension of the Chernoff-Hoeffding bounds," *SIAM Journal on Computing*, vol. 26, pp. 350–368, 1997.

Maleq Khan is currently at the Network Dynamics and Simulation Science Laboratory at Virginia Bioinformatics Institute, Virginia Tech. He received a B.S. in Computer Science and Engineering from Bangladesh University of Engineering and Technology, an M.S. in Computer Science from North Dakota State University, and a Ph.D. in Computer Science from Purdue University. His research interests are in distributed algorithms, randomized algorithms, wireless ad hoc networks, and theoretical computer science in general. Maleq Khan, together with Gopal Pandurangan, received the Best Student Paper Award at the 20th International Symposium on Distributed Computing (DISC 2006). He is a member of IEEE.

Gopal Pandurangan is an assistant professor of Computer Science at Purdue University. He has a B.Tech. in Computer Science from the Indian Institute of Technology at Madras (1994), a M.S. in Computer Science from the State University of New York at Albany (1997), and a Ph.D. in Computer Science from Brown University (2002). His research interests are in design and analysis of algorithms, distributed computing and communication networks (especially sensor networks and peer-to-peer networks), Web/Internet algorithmics, and computational biology. On these subjects he has authored or co-authored over 50 publications in refereed journals and conferences. In the 2006 International Symposium on Distributed Computing, he and Maleq Khan won the Best Student Paper Award for their paper titled "A Fast Distributed Approximation Algorithm for Minimum Spanning Trees". Pandurangan is a member of IEEE.

V.S. Anil Kumar is currently at the Dept. of Computer Science and the Virginia Bioinformatics Institute at Virginia Tech. His interests are in the broad areas of algorithms, combinatorial optimization, probabilistic techniques and distributed computing, and their applications to wireless networks, epidemiology, and the modeling, simulation and analysis of social and infrastructure networks.