Energy-Efficient Distributed Minimum Spanning Tree Construction: Tight Bounds and Algorithms

Yongwook Choi * Maleq Khan † V.S. Anil Kumar ‡ Gopal Pandurangan *

Abstract

Traditionally, the performance of distributed algorithms has been measured in terms of running time and message complexity. However, in many settings, a more accurate and relevant measure of performance is required. In ad hoc wireless networks, energy is a very critical factor for measuring the efficiency of a distributed algorithm. Thus in addition to the traditional time and message complexity, it is also relevant to consider *energy complexity* that accounts for the total energy associated with the messages exchanged among the nodes in a distributed algorithm.

This paper focuses on the energy complexity of distributed algorithms for the Euclidean minimum spanning tree (MST) problem, one of the most important problems in distributed computing. We show tight upper and lower bounds on the energy complexity of distributed (Euclidean) MST algorithms. We also study distributed *approximation* algorithms for MST that have almost optimal energy complexity, but give a constant factor approximation to the MST. We present two sets of results, one where nodes are distributed randomly in a plane, and the other where nodes can be arbitrarily distributed. For random distribution of nodes, we show that $\Omega(\log n)$ is a lower bound on the energy complexity of any distributed MST algorithm. We then give a distributed algorithm that constructs an optimal MST with $O(\log n)$ energy complexity on average and $O(\log n \log\log n)$ energy complexity with high probability. This is an improvement over the previous best known bound on the average energy complexity of $\Omega(\log^2 n)$. All the above results assume that nodes do not know their geometric coordinates. If nodes know their own coordinates, then we give an algorithm with $O(1)$ energy complexity that gives an $O(1)$ approximation to the MST. For arbitrary distribution of nodes, we give a lower bound on the energy complexity for any distributed MST algorithm and present an algorithm matching this lower bound within a polylogarithmic factor. Our algorithm gives an $O(1)$ approximation to the MST. Our results show that message optimal algorithms (such as the classical algorithm due to Gallager et al.) are not energy-optimal, and hence it is necessary to design new distributed algorithms that are energy-optimal.

Keywords: Distributed Algorithm, Energy-Efficient, Euclidean Minimum Weight Spanning Tree, Distributed Approximation Algorithm.

*Department of Computer Science, Purdue University, West Lafayette, IN 47907, USA. E-mail: ywchoi@purdue.edu, gopal@cs.purdue.edu

†Network Dynamics and Simulation Science Laboratory, VBI, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061, USA. E-mail: maleq@vbi.vt.edu

‡Department of Computer Science, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061, USA. E-mail: akumar@vbi.vt.edu

1 Introduction

Emerging technologies such as ad hoc wireless networks and sensor networks operate under inherent resource constraints. Consider a sensor network, an ad hoc wireless network in a geographic area built up of a large number of inexpensive devices called as sensors. Sensors have energy constraints due to the limited battery power of the sensor nodes; this severely constrains the amount of computation the nodes can do and the distance to which they can communicate. A distributed algorithm which consumes a relatively large amount of power may not be suitable in such a resource-constrained network. The topology of these networks can change frequently due to mobility or node failures. Communication cost and running time is even more crucial in such a dynamic setting. A distributed algorithm that runs on such devices should have as little communication as possible and should run as fast as possible (i.e., requiring a small number of communication rounds) and use as small energy as possible. Hence it becomes critical to design energy-efficient distributed algorithms that operate on these networks.

Traditionally, the performance of distributed algorithms has been measured in terms of running time and message complexity. In fact, in standard distributed computing literature, these are the two most widely used measures [18, 21]. Message complexity concerns the total number of messages transmitted over all the edges during the course of the algorithm. However, in many settings, we require a more accurate and relevant measure of performance. A good example is radio and wireless networks, where energy is a very critical factor for measuring the efficiency of a distributed algorithm. Transmitting a message between two nodes has an associated cost (energy) and moreover this cost can depend on the two nodes (e.g., the distance between them among other things). Thus in addition to the traditional time and message complexity, it is also relevant to consider *energy complexity* that accounts for the total energy associated with the messages exchanged among the nodes in a distributed algorithm.

This paper addresses the minimum spanning tree (MST) problem, one of the most important problems in distributed computing. In particular, motivated by ad hoc wireless and sensor networks, we focus on the Euclidean MST problem. Our goal in this paper is to study distributed algorithms for the Euclidean MST problem that have low energy complexity. We show energy complexity lower bounds on the performance of any distributed MST. We also study distributed *approximation* algorithms for MST that give better energy complexity at the price of some additional information about the coordinates of the nodes and obtain a constant factor approximation to the MST.

1.1 Model and Problem

Network Model. We assume that the network is modeled as a weighted undirected graph $G = (V, E, w)$ where V is the set of the nodes (vertices) and E is the set of the (bi-directional) communication links between them and $w(e)$ is the weight of the edge $e \in E$. Without loss of generality, we assume that G is connected. The weight $w(u, v)$ represents the energy associated with transmitting a message between u and v. The graph G has the following underlying geometry. The nodes are set of $|V| = n$ points distributed in a unit square and two nodes are connected if they are within distance r of each other. Without loss of generality, we assume that r is chosen such that G is connected. We study two different node distributions: (1) random, where the points are distributed randomly (the graph induced is also known as a *random geometric graph* [20]); and (2) arbitrary, where we make no assumption on the distribution of the points. Thus G can be considered as a weighted unit disk graph. This is a standard geometric graph model that has been widely used in the literature for modeling communications networks in a plane, e.g., wireless and ad hoc (sensor) networks.

Formally, the energy complexity of the distributed algorithm is defined as $\sum_{i=1}^{m} w_i$ where w_i is the weight of the edge which connects the nodes exchanging the ith message, and m is the total number of messages exchanged by the algorithm. The weight of an edge is a function of the distance between the two points. Although our results are generalizable to a wide variety of weight functions, for concreteness, we assume that $w(u, v)$ is proportional to some fixed power (denoted as α) of the distance between u and v, i.e., $w(u, v) = a(d(u, v))^{\alpha}$ where $d(u, v)$ is the (Euclidean) distance between u and v, and a is some fixed constant. The motivation for this comes from energy requirements in a radio (wireless) communication paradigm: to transmit

a message over a distance r, the required *energy* is (proportional to) r^α, where typically α is 2 [6, 12]. In this paper, unless otherwise stated, we assume $\alpha = 2$. (Our results can be generalized to other values of α as well.)

Distributed Computing Model. Each node in G hosts a processor with limited initial knowledge. Specifically, we make the common assumption that each node has unique identity numbers (this is not really essential, but simplifies presentation) and at the beginning of computation, each vertex v accepts as input its own identity number. Thus, a node has only *local* knowledge limited to itself. Nodes do not even know the weights of its incident edges (or equivalently the distance to its neighbors). We assume that the communication is synchronous and occurs in discrete time steps. The energy associated with a bi-directional communication between neighbors u and v is $O(w(u, v))$, i.e., if u wants to send a message to v and v replies back to u then the cost associated with this bi-directional communication is $2w(u, v)$. (A message is of size $O(\log n)$ bits). In a one-directional communication, when a node u can send a message to a distance $d \leq r$, we assume that any node within distance d can receive the message. The cost associated with this message is (proportional to) d^2. This is called as *local broadcasting* and is a feature of radio and wireless networks. We assume that a node can receive messages from more than one neighbor in the same time step. In this model, we ignore "collisions" between messages; such issues do arise in real-world radio and wireless networks. However, the collisions can easily be resolved by a constant-factor increase in energy complexity of a distributed algorithm [15].

MST Problem. We are interested in studying energy-efficient distributed algorithms for the Euclidean MST problem in the above model. Formally, the Euclidean MST problem is, given a network $G = (V, E, w)$, to find a tree T spanning V such that $\sum_{(u,v)\in T} d(u, v)$ is minimized. We actually consider a generalized version of the above problem: Find a tree T spanning V such that $\sum_{(u,v)\in T} d^\alpha(u, v)$ is minimized where α is a (small) positive number. The motivation for this objective function comes from energy requirements in a wireless communication paradigm as mentioned earlier. It can easily be shown (e.g., using Kruskal's algorithmic construction [5]) 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. Note that α plays a role in determining the approximation ratio guarantee (for exact algorithms, it does not matter). Although our results can be generalized to any α, we focus on $\alpha = 1$ (the Euclidean MST) and $\alpha = 2$.

Computing an MST by a distributed algorithm is a fundamental task, as the following distributed computation can be carried over the best backbone of the communication graph. Two important applications of MST are in broadcasting and data aggregation. In wireless networks, an MST can be used as a communication tree to minimize energy consumption since it minimizes $\sum_{(u,v)\in T} d^\alpha(u, v)$. 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. Common aggregate functions are minimum, maximum, average, etc [17]. One popular paradigm for computing such aggregates is to construct a (directed) tree rooted at the sink where each node forwards its (locally) *aggregated* data collected from its subtree to its parent [13]. For such cases, MST is the optimal data aggregation tree [15]. It was shown in [1, 4, 25] that broadcasting based on MST consumes energy within a constant factor of the optimum.

1.2 Our Results

We show tight upper and lower bounds on the energy complexity of distributed MST algorithms. We present two sets of results, one for random distribution, and the other for arbitrary distribution.

Random distribution[1]: We first show that $\Omega(\log n)$ is a lower bound on the energy complexity of any distributed MST algorithm. In fact, we show that this is the lower bound for constructing any spanning tree in the network. We then give a distributed algorithm that constructs an optimal MST with $O(\log n)$ energy complexity on average and $O(\log n \log \log n)$ energy complexity with high probability (whp)[2]. The previous best known bound on the average energy complexity for distributed MST in this model was $\Omega(\log^2 n)$ [15]. This bound was obtained in [15] for a natural implementation of the classical algorithm of Gallager, Humblet, and

[1] A summarized version of our results for the random distribution case appeared as a brief announcement in the proceedings of the 20th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA) 2008 [3].

[2] Throughout this paper, "whp" means with probability tending to 1 as $n \to \infty$, where n is the number of nodes in the network.

Spira (henceforth called as GHS algorithm) [9]. All the above results assume that nodes do not know their geometric coordinates. If nodes know their own coordinates, then we give an algorithm with $O(1)$ energy complexity that gives an $O(1)$ approximation to the MST. We note that $\Omega(1)$ is a lower bound on the energy complexity of any distributed MST algorithm (even with nodes knowing their coordinates) since any algorithm has to communicate at least once using the tree edges of an MST. For an instance specified by the set V of nodes, we denote this lower bound as $LMST(V) = \sum_{(u,v)\in MST(V)}(d(u,v))^2$, where $MST(V)$ denotes the minimum Euclidean spanning tree on V. If the nodes are distributed uniformly at random, it is well-known that $\sum_{(u,v)\in MST(V)}(d(u,v))^2 = \Omega(1)$ (e.g., see [15]).

Arbitrary distribution: We first give a lower bound on the energy complexity for any distributed MST algorithm and present an algorithm matching this lower bound within a polylogarithmic factor. Our algorithm gives an $O(1)$ approximation to the MST. Our lower bound says that any distributed algorithm has energy complexity that is at least $\Omega(\sqrt{n} \times LMST(V))$, where $LMST(V)$ is defined as above. We then give a distributed algorithm that has energy complexity $O(\sqrt{n}\log^3 n \times LMST(V))$. We also show that a natural implementation of the GHS algorithm will take $\Omega(n \times LMST(V))$.

1.3 Related Work

Although message complexity of a distributed algorithm directly influences the energy complexity, algorithms that have optimal message complexity are not necessarily energy optimal. The message-optimal GHS algorithm [9] uses $O(n\log n + |E|)$ messages. It was shown in [15] that this algorithm requires $\Omega(\log^2 n)$ energy on average under random distribution; in contrast we show that there is an algorithm that takes $O(\log n)$ energy on the average and this is asymptotically optimal. There are distributed algorithms that construct the MST optimally in terms of time complexity [8, 21]. But these algorithms require much more messages than GHS algorithm, and consequently require a lot more energy. The distributed algorithm of [14, 15] requires only $O(\log n)$ energy, but it gives an $O(\log n)$-approximation to the MST. The work of [15] raised the question of whether there exists a distributed algorithm of $O(\log n)$ energy complexity and this paper answers this in the affirmative.

Our model is related to the more general cost-sensitive communication model of Awerbuch et al. [2] which was also inspired by the need to go beyond the traditional notion of message complexity. In this model, the cost of sending a message across an edge is equal to the weight of the edge. Awerbuch et al. give a distributed MST algorithm with communication cost complexity of $O(\min(W(G) + LMST(V)\log n, n \cdot LMST(V))$, where $W(G)$ is the sum of the weights of all the edges in G (see also the algorithm of [22]). Our results cannot be directly compared with this since our weight functions are of a certain type (based on the underlying geometry) and ours uses local broadcasting instead of the point-to-point message passing model of [2]. Nevertheless, comparing the energy complexity of our algorithm on arbitrary distribution to the above shows that our bounds are significantly better at the cost of providing a $O(1)$ approximation to the MST.

2 Random Distribution of Nodes

2.1 Lower Bound

We show a non-trivial lower bound of $\Omega(\log n)$ on the energy required by any distributed algorithm to construct any spanning tree of the network ($\Omega(1)$ lower bound is trivial, as mentioned in Section 1.2). This bound holds under the following assumptions: (1) the model is synchronous (hence the lower bound applies to asynchronous model as well); (2) any non-empty set of processors may start the algorithm; a processor that is not started remains asleep until a message reaches it and can be awakened spontaneously; (3) no assumption is made on the size of the messages; this assumption only strengthens our bound; (4) nodes do not have any information on their geometric coordinates.

Theorem 2.1 *Any distributed algorithm needs $\Omega(\log n)$ energy WHP to construct a spanning tree.*

Proof: The proof makes use of a classical lower bound due Korach et al. [16] that shows that $\Omega(n\log n)$ messages is needed by any distributed algorithm for constructing a spanning tree (or equivalently, leader election)

in a complete network. More precisely, Korach et al. lower bound shows that $\Omega(n \log n)$ *different* edges need to be used by any algorithm. This bound can be shown to apply for Las Vegas type randomized algorithms also.

Our model can also be viewed as complete weighted network, where the weight between any two nodes u and v is $w(u, v) = (d(u, v))^2$. According to the Korach et al. bound, at least $an \log n$ different edges need to be used by any distributed MST algorithm for some fixed constant a. To obtain a lower bound on the energy complexity, we compute the minimum energy needed to send at least one message through $n \log n$ different edges. We need the following lemma.

Lemma 2.1 *For* every *node, WHP, at least k/bn energy is needed if the node wants to communicate with its closest k neighbors, for all $k > a_1 \log n$, where $a_1 < a$ is a fixed positive constant and b is a suitably large constant.*

Proof: Fix an arbitrary node v. Let X be the random variable that denotes the total number of nodes within distance $\sqrt{k/bn}$ of v. $E[X] = k/b$. Using a Chernoff bound [19], for suitably large b,

$$\Pr(X \geq k) = \Pr(X \geq (1 + b - 1)k/b) \leq (e/b)^k \leq (e/b)^{a_1 \log n} = o(1/n).$$

That is, WHP, the number of neighbors of v within distance $\sqrt{k/bn}$ is less than k. Hence, if a node wants to communicate with its closest k neighbors, it has to send a message to a distance of at least $\sqrt{k/bn}$ WHP. Thus, the energy needed for this is k/bn. By the union bound [19], this holds for every node WHP. □

We only focus on those nodes that communicate with more than $a_1 \log n$ of its closest neighbors. Let the set of such nodes be denoted by R (relevant set). We ignore the energy spent by the rest of the nodes and focus only on lower bounding the energy needed by the nodes in R. Since, the total number of edges used should be at least $an \log n$ and since $a_1 < a$, the nodes in R need to use at least $\Omega(n \log n)$ edges, i.e., communicate with at least $\Omega(n \log n)$ (closest) neighbors. Each node in R communicates with at least $k > a_1 \log n$ neighbors and by Lemma 2.1, it has to spend at least k/bn energy WHP. Thus, WHP, the total energy needed is at least: $\sum_{v \in R} k/bn = 1/bn \sum_{v \in R} k \geq \Omega(\log n)$. □

2.2 An Energy-Optimal Distributed MST Algorithm

In this section, we give an energy-optimal distributed MST algorithm of energy complexity $O(\log n)$, matching the lower bound shown in the previous section.

We assume that graph G (cf. Section 1.1) is connected by setting the transmission radius to the r value given below. Theorem 2.2 shows that this guarantees the connectivity of random geometric graphs.

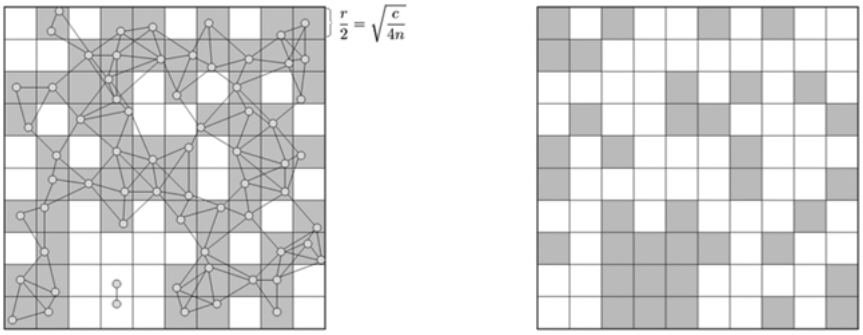
Theorem 2.2 *[11, 20] If $r = \sqrt{\frac{c_2 \log n}{n}}$, where c_2 is a constant larger than 4, then the graph is connected whp.*

Our algorithm crucially depends on the following Theorem 2.3. It essentially says that, if $r = \sqrt{\frac{c_1}{n}}$ (for some constant c_1), then there will be a unique giant component and other small components. Refer to Figure 1. In Figure 1(a), the giant component is shown. A maximal connected cluster of white cells in Figure 1(a) is called a *small region*. In Figure 1(b), these small regions are represented as gray cells. All small components are inside such small regions, and moreover there are not too many small components in any one small region.

Theorem 2.3 *There exists a positive constant c_1 such that, if $r = \sqrt{\frac{c_1}{n}}$, then there is a unique giant component containing $\Theta(n)$ nodes whp. Furthermore, whp, all remaining components of nodes are trapped inside small regions, each of which contains at most $\beta \log^2 n$ nodes, for some positive constant β.*

The theorem is essentially similar to Theorem 1 in [23], but the conditions are different. In our model two nodes are connected to each other if they are within $r = \sqrt{\frac{c_1}{n}}$ (for some constant c_1) of each other, whereas in [23] each node is connected to the K closest nodes where K is some fixed constant (independent of n). We refer to Appendix A for the proof of Theorem 2.3.

Our distributed MST algorithm consists of two steps, each of which uses the GHS algorithm with some modifications. For constants c_1, c_2, and β (as defined in Theorems 2.2 and 2.3), our algorithm works as follows. (The modified GHS algorithm is described in Section 2.2.1.)



(a) giant component (b) small regions

Figure 1: A giant component and small regions

Step 1:

1. Each node sets its radius to $\sqrt{\frac{c_1}{n}}$ (i.e., it communicates with nodes only within this distance).
2. Run the modified GHS algorithm.

Step 2:

1. Each component computes its size, the number of nodes it contains.
   If the size is greater than $\beta \log^2 n$, it considers itself as a giant component.
2. Each node increases its radius to $\sqrt{\frac{c_2 \log n}{n}}$.
3. Run the modified GHS algorithm on the remaining component.
   (The giant component does not participate but only accepts connection messages from small components.)

The main idea of our algorithm is based on the fact that, after the first step, with high probability, there will be one unique giant component and other small components, and that those small components will be trapped inside small regions, each of which contains at most $O(\log^2 n)$ nodes. In the second step, the small components in each small region are merged with each others in the same small region or with the giant component, and eventually all nodes will be connected with high probability. By controlling the transmission radius in each step, we bound the energy complexity as in the following theorem.

Theorem 2.4 *Our algorithm constructs an optimal MST using $O(\log n)$ energy on average and $O(\log n \log\log n)$ energy whp.*

The correctness of our algorithm immediately follows from Theorem 2.2 and the correctness of GHS algorithm. The proof of energy complexity is given in Section 2.2.2.

2.2.1 A Modified GHS Algorithm

In this section we describe the modified GHS algorithm and analyze its message complexity. In the modified GHS algorithm, most of steps are the same as those in the original GHS algorithm [9]. The difference is that each node additionally keeps a list of its neighbors that are in other fragments with their distance information. As in [9], a subtree of MST is called a *fragment*. In each phase, after two or more fragments are merged, each node sends a message to its neighbors to announce its new fragment id if the id has changed. Each node updates its list when it receives those announcements from its neighbors. This modification enables each node to find its minimum outgoing edge without any additional messages — just by looking up its list and picking up the one with the minimum distance.

Let us compute the message complexity of this modified GHS algorithm. For each node, the number of messages needed to announce its new fragment id is bounded by the total number of phases. The number of messages needed for broadcast and convergecast is the same as the original algorithm. Thus the total message complexity is $O(n\phi)$ where n is the number of nodes and ϕ is the number of phases.

In the modified GHS algorithm in Step 2, two simple techniques are used to reduce the expected energy complexity. Firstly, the giant fragment does not participate but only accepts connection messages from small fragments. Secondly, when small fragments are merged with the giant fragment, small fragments change their ids. That is, the giant fragment keeps its fragment id so that its nodes do not need to announce new ids.

2.2.2 Energy Complexity Analysis

In this section we analyze the energy complexity of our algorithm. We first give a high probability analysis.

In Step 1, the total number of phases in the modified GHS algorithm is $O(\log n)$, and consequently the total number of messages is $O(n \log n)$. Sending one message requires $O(1/n)$ energy since the transmission radius is $O(\sqrt{1/n})$. Therefore, the total energy required in Step 1 is $O(\log n)$. At the beginning of Step 2, each fragment needs to compute its size. This can be done with one broadcast and one convergecast, which need $O(n)$ messages and consequently $O(1)$ energy in total.

We now compute the energy required by the modified GHS algorithm in Step 2. It is shown that the number of nodes in a small region is at most $O(\log^2 n)$ whp. Thus, the number of fragments in a small region is at most $O(\log^2 n)$, and each small fragment just needs to connect only with other small fragments in the same small region or the giant fragment. Therefore, the total number of phases in the modified GHS algorithm is at most $O(\log \log n)$ whp. Thus the total number of messages needed in Step 2 is $O(n \log \log n)$ whp. The energy needed for each message is $O(\log n/n)$ since we increased the transmission radius to $O(\sqrt{\log n/n})$. Thus, the total energy required in Step 2 is $O(\log n \log \log n)$. Therefore, the overall energy complexity is $O(\log n \log \log n)$ whp.

Now we show that the expected energy complexity is $O(\log n)$. The expected energy required by the modified GHS algorithm in Step 1 and the computation of each fragment's size in Step 2 is clearly $O(\log n)$. Thus it suffices to show that the expected energy required by the modified GHS algorithm in Step 2 is $O(\log n)$. This can be shown from the following lemma. The proof of Lemma 2.2 can be found in Appendix B.

Lemma 2.2 *In the modified GHS algorithm used in Step 2, the expected number of messages needed by all nodes in any one small region is a constant.*

By Lemma 2.2, it follows that the expected energy required to connect all nodes in one small region is $O(\log n/n)$. Since there are at most $O(n)$ small regions, the required energy for all nodes in all small regions is $O(\log n)$. The energy needed by all nodes in the giant fragment is $O(\log n)$ since there are at most $O(n)$ messages for accepting connection requests from small fragments. Therefore, the total expected energy required is $O(\log n)$. This completes the proof of Theorem 2.4.

2.3 An $O(1)$ Approximation Algorithm with $O(1)$ Energy Complexity

We know that the lower bound on energy complexity for distributed construction of any spanning tree, hence also MST, is $\Omega(\log n)$. However, if some additional information such as coordinates of the nodes is given to the nodes, a more energy-efficient algorithm can be developed. In this section, we present a distributed algorithm to construct a spanning tree assuming that each node knows its own coordinates. This spanning tree gives a constant approximation to MST, and the energy complexity of the algorithm is also constant.

Nodes are distributed uniformly at random in a unit square with lower-left corner at $(0, 0)$ and upper-right corner at $(1, 1)$ (see Figure 2a). Each node v knows its coordinates (x_v, y_v). We define the *ranks* of the nodes as follows: for any two nodes u and v,

$$\text{rank}(u) < \text{rank}(v) \text{ iff } (x_u + y_u < x_v + y_v) \text{ or } (x_u + y_u = x_v + y_v \text{ and } y_u < y_v).$$

Assuming that no two nodes have the same coordinates, for any pair of nodes u and v, either $\text{rank}(u) < \text{rank}(v)$ or $\text{rank}(v) < \text{rank}(u)$. To build the spanning tree, each node, except the node with the highest rank, is connected to the nearest node of higher rank. It is easy to see that in such a construction, the resulting graph is a single connected component with no cycle, i.e., a tree. This tree is called *nearest neighbor tree* (NNT) (cf. [15]). In [15], an NNT is constructed using a different ranking: $\text{rank}(u) < \text{rank}(v)$ iff $(x_u < x_v)$ or $(x_u = x_v$ and $y_u < y_v)$, which also gives us constant approximation and constant energy complexity. However, in that

ranking, there are few nodes that need to go far away to find the nearest node of higher rank. As a result, it is not suitable for the unit disk graph model with $r = \Theta(\sqrt{\frac{\log n}{n}})$ that we are using in this paper. With our modified ranking scheme, we show that every node finds the nearest node of higher rank within distance $r = \Theta(\sqrt{\frac{\log n}{n}})$ with high probability (see Lemma 2.5 below). To achieve our goal with this modified ranking of the nodes requires an entirely different technique to prove the bounds as given below.

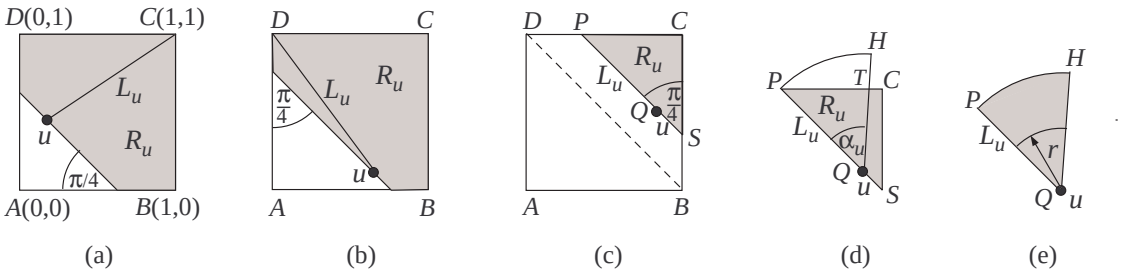


Figure 2: (a), (b), (c): The potential region R_u marked by dark color and the potential distance L_u for an arbitrary node u. (d): A pie slice with area equal to the area of the potential region shown in (c); α_u is the potential angle for u.

Consider an arbitrary node u as shown in Figure 2. The straight line $x+y = x_u+y_u$, which passes through u making equal angles with both axes, divides the unit square into two regions. The region in the half plane $x + y > x_u + y_u$, the dark region in the figure, is called the *potential region* for u, denoted by R_u. Any node in R_u has higher rank than u, and u is connected to the nearest node in R_u. The area of R_u is called the *potential area* for u, denoted by A_u. The distance to the farthest point in R_u from u is called the *potential distance* for u, denoted by L_u. Now, as shown in Figure 2(d), consider a pie slice with angle α_u (in radian) of the circle with center u and radius L_u such that the area of the pie slice equals the potential area for u, i.e., $\frac{1}{2}\alpha_u L_u^2 = A_u$. that is, $\alpha_u = \frac{2A_u}{L_u^2}$. Angle α_u is called the *potential angle* for u.

Lemma 2.3 *For any u, the potential angle $\alpha_u \geq \frac{1}{2}$ radian.*

Proof: For a node u with $x_u + y_u \geq 1$, i.e., u is in triangle BCD at some point Q as shown in Figure 2(c): $A_u = \triangle PSC$ and $L_u \leq PS$. Thus, $\alpha_u = \frac{2A_u}{L_u^2} \geq \frac{1}{2}$. If $x_u + y_u < 1$, i.e., u is in triangle ABD, $A_u \geq \triangle BCD$, $L_u \leq BD$, and thus $\alpha_u \geq \frac{1}{2}$. □

Lemma 2.4 *If d_u denotes the distance from u to the nearest node in the potential region R_u, $E[d_u^2] \leq \frac{2}{n\alpha_u}$.*

Proof: Consider Figure 2(d). By construction, the area of the pie slice PQH with angle α_u and radius L_u is equal to area of the potential region R_u, which is region PSC. Thus, the areas of regions PTH and $TQSC$ are equal. Now remove the nodes from region $TQSC$ and place them in region PTH uniformly at random. In this process, we are only moving some nodes away from u. Thus if d'_u denotes the distance from u to the nearest node in the pie slice PQH, we have $d_u \leq d'_u$. Now we compute $E[d'^2_u]$.

Consider the region in the pie slice PQH within distance r from u as shown in Figure 2(e). By uniform distribution, the probability that a particular node resides in a particular region is equal to the area of that region since the area of the unit square $ABCD$ is 1. Thus, the probability that there is at least one node, other than u, within distance r from u in the pie slice, is given by

$$F(r) = 1 - \left(1 - \tfrac{1}{2}\alpha_u r^2\right)^{n-1}$$

$$\text{Density function, } f(r) = \tfrac{d}{dr}F(r) = (n-1)\alpha_u r \left(1 - \tfrac{1}{2}\alpha_u r^2\right)^{n-2}$$

$$E[d_u^2] \leq E[d'^2_u] = \int_0^{L_u} r^2 f(r)\, dr = \tfrac{2}{n\alpha_u}\left\{1 - nx^{n-1} + (n-1)x^n\right\} \leq \tfrac{2}{n\alpha_u}$$

where $x = 1 - \frac{1}{2}\alpha_u L_u^2$. The last inequality follows from the fact that $0 \leq x \leq 1$. □

Using the above lemmas, in the following theorem, we show that expected sum of the squared edge lengths of the NNT is constant. It is well-known that the expected sum of the squared edges of MST is $\Theta(1)$ [24] when the nodes are distributed in a unit square uniformly at random. Thus NNT gives us a constant approximation.

Theorem 2.5 *The expected sum of the squared edge lengths of the NNT,* $E\big[\sum_{e\in NNT}|e|^2\big] = O(1)$.

Proof: $E\big[\sum_{e\in\text{NNT}}|e|^2\big] = E\big[\sum_{u\in V} d_u^2\big] = \sum_{u\in V} E\big[d_u^2\big] = nE\big[d_u^2\big] \le 4$, by Lemma 2.3 and 2.4. □

Using a slightly different technique, we can show that the expected sum of the edge lengths (i.e., the case of Euclidean MST, in contrast to the sum of the *squared* edge lengths) of the NNT, $E\big[\sum_{e\in\text{NNT}}|e|\big] = O(\sqrt{n})$. For MST, $E\big[\sum_{e\in\text{MST}}|e|\big] = \Theta(\sqrt{n})$ [24]. Thus, in this case, we also have constant approximation to MST.

In the following lemma, we show that all nodes find the nearest node in their potential regions within distance $\Theta(\sqrt{\frac{\log n}{n}})$ with high probability. That is, with high probability, the NNT can be constructed in unit disk graph, where the transmission radius for each node is $\Theta(\sqrt{\frac{\log n}{n}})$.

Lemma 2.5 *Simultaneously for all* $u \in V,\ d_u \le c\sqrt{\frac{\log n}{n}}$ *with probability at least* $1 - \frac{1}{n^{c^2/8-1}}$.

Proof: Let $r = c\sqrt{\frac{\log n}{n}}$. Consider an arbitrary node u. If $L_u < r$, then $\Pr\{d_u \le r\} = 1$. Assume that $L_u \ge r$. Now, $\Pr\{d_u \le r\}$ is larger or equal to the probability that there is at least one node in R_u within distance r from u. Thus, we have

$$\Pr\{d_u \le r\} \ge 1 - \big(1 - \tfrac{1}{2}\alpha_u r^2\big)^{n-1} \ge 1 - e^{\frac{1}{2}\alpha_u r^2(n-1)} \ge 1 - \frac{1}{n^{c^2/8}}.$$

Using the union bound, $d_u \le r$ holds simultaneously for all $u \in V$ with probability at least $1 - \frac{1}{n^{c^2/8-1}}$. □

In the following theorem, we show that we can devise a distributed algorithm to construct NNT with constant energy complexity.

Theorem 2.6 *There is a distributed algorithm to construct NNT with expected energy complexity* $O(1)$ *and message complexity* $O(n)$.

Proof: Consider the following algorithm. Assume that each node u knows its potential distance L_u and the number of nodes n — node u can locally compute exact L_u from its coordinates and a rough approximation for n will work; the bounds on energy and messages hold as long as approximate value for n is $\Theta(n)$. To find the nearest node in the potential region R_u, each node u transmits a *request* message containing its coordinates (x_u, y_u) to distance $r_i = \sqrt{\frac{2^i}{n}}$ in rounds $i = 1, 2, \ldots, m = \lceil \lg nL_u^2 \rceil$. Any node v within distance r_i can hear the message and replies back to u if $\text{rank}(u) < \text{rank}(v)$, i.e., v is in R_u. If u gets back replies from one or more nodes, it selects the nearest node among them and sends a *connection* message to it, and stops exploration; otherwise, u continues to the next phase, $(i+1)$st phase. If u does not find any node in R_u within distance L_u (it happens only to the highest ranked node), it terminates anyway.

Node u needs the first transmission with probability $p_i = 1$. For $i \ge 2$, u needs the ith transmission only if there is no node within distance r_{i-1} in R_u. That is, the probability that u needs ith transmission is

$$p_i \le \big(1 - \tfrac{1}{2}\alpha_u r_{i-1}^2\big)^{n-1} \le \big(1 - 2^{i-3}/n\big)^{n-1} \le e^{-2^{i-4}}.$$

The expected number of nodes in R_u within distance r_i is at most $\frac{1}{2}\pi r_i^2 n = 2^{i-1}\pi$. Thus, the expected number of replies u receives is at most $2^{i-1}\pi$. Additionally, u sends at most one *request* message and one *connection* message in each phase. That is, the expected number of messages in phase i, $E[M_i] \le 2 + 2^{i-1}\pi$. Therefore, the expected number of messages for all n nodes is at most

$$n\sum_{i=1}^{m} p_i E[M_i] = n(2+\pi) + n\sum_{i=2}^{m}(2 + 2^{i-1}\pi)e^{-2^{i-4}} = O(n).$$

The expected energy is at most $n\sum_{i=1}^{m} p_i E[M_i] r_i^2 = 2(2+\pi) + \sum_{i=2}^{m}(2 + 2^{i-1}\pi)2^i e^{-2^{i-4}} = O(1)$. □

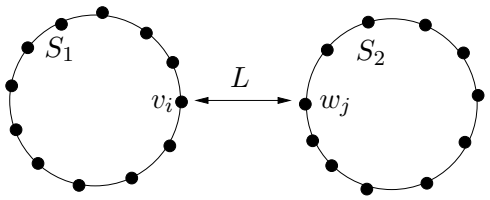


Figure 3: The circle S_2 can be moved around S_1, while keeping the distance from S_1 fixed, so that the closest pair of nodes $v_i \in S_1$ and $w_j \in S_2$ can be varied.

3 Arbitrary Distribution of Nodes

In this section, we produce a lower bound on energy complexity for any distributed algorithm where the nodes placed arbitrarily in a two dimensional plane and present an algorithm matching this lower bound within a polylogarithmic factor.

3.1 A Lower Bound on Energy Complexity

Let $r(v)$ be the largest radius that node v sends a message to, during the entire course of the algorithm. For a given set S of nodes on the plane, and a radius vector $\vec{r}$ that specifies a radius $r(v)$ for each $v \in S$, let $B(S, \vec{r}) = \{p = (x, y) \in \mathbb{R}^2 : \exists\ v \in S,\ d(p, v) \leq r(v)\}$, where $d(p, v)$ denotes the Euclidean distance between the point p on the plane and the node $v \in S$. If $r(v) = \ell$ for all v, we denote $B(S, \vec{r})$ simply by $B(S, \ell)$. Let $V = V_1 \cup V_2$ be a set of nodes on the plane, partitioned into V_1 and V_2, each containing $n/2$ nodes. The sets V_1 and V_2 form disjoint circles S_1 and S_2, respectively, where the nodes on each circle are spaced unit distance apart as shown in Figure 3. Let L be the distance between the circles S_1 and S_2. We consider a energy-optimum algorithm and the radius vector $\vec{r}$ chosen by it - so $r(v)$ denotes the radius explored by node v in this algorithm.

Lemma 3.1 *Let $\vec{r}$ denote the radius vector chosen by an optimum algorithm. Then, $B(S_1, L) \subseteq B(S_1, \vec{r})$, where L is the minimum distance between sets S_1 and S_2 in Figure 3.*

Proof: For simplicity, we assume below that the energy optimum algorithm only explores from nodes in S_1, i.e., it has $r(w) = 0$ for all $w \in S_2$; the argument can be completed to consider the general setting as well. Suppose $B(S_1, L) \not\subseteq B(S_1, \vec{r})$. Then, there exists a point $p = (x, y) \in \mathbb{R}^2$ and $v_i \in S_1$ such that $d(p, v_i) \leq L$ and $p \notin B(S_1, \vec{r})$. In this case, the adversary can position circle S_2 so that there is a node $w_j \in S_2$ at the location p, and the nodes v_i and w_j are the closest pair of nodes between these circles. That is, the radius vector $\vec{r}$ is not adequate to find the minimum outgoing edge from S_1. □

The above lemma implies that if the minimum outgoing edge for the set S has length ℓ, then any optimum algorithm must explore enough region so as to cover $B(S_1, \ell)$. We recall that $LMST(V)$ (defined in Section 1.2) is a trivial lower bound on the energy complexity.

Corollary 3.1 *There is an instance of Euclidean MST on a set V of n nodes for which the energy complexity of any algorithm that computes the MST is at least $\Omega(LMST(V)\sqrt{n})$.*

Proof: We consider the above instance, with $L = \sqrt{n}$. Let $\vec{r}$ denote the optimum radius vector chosen by the optimum algorithm. Then the MST T contains $\frac{n}{2} - 1$ edges among the nodes on each circle, and the edge between the closest pair of nodes $v_i \in S_1$ and $w_j \in S_2$. Thus, $LMST(V) = \sum_{e \in T} \ell(e)^2 = 2\left(\frac{n}{2} - 1\right) + L^2 = \Theta(n)$. Let R be the radius of circle S_1. Then, the circumference $2\pi R \geq \frac{n}{2}$. By Lemma 3.1, we must have $B(S_1, L) \subseteq B(S_1, \vec{r})$. The area of $B(S_1, L)$ is at least $\pi(R + L)^2 - \pi(R - L)^2 = 4\pi RL \geq n\sqrt{n}$. Then, the area of $B(S_1, \vec{r})$ is also at least $n\sqrt{n}$. Thus, we have $\sum_{v \in S_1} \pi r(v)^2 \geq n\sqrt{n}$. Therefore, $\sum_{v \in V} r(v)^2 \geq \sum_{v \in S_1} r(v)^2 \geq \frac{1}{\pi} n\sqrt{n} = \Omega(LMST(V)\sqrt{n})$. □

Notice that the above lower bound holds even if each node knows its own coordinates. Using the same instance given above, we can show the following lower bound on the performance of the GHS algorithm.

Corollary 3.2 *There is an instance of Euclidean MST on a set V of n nodes for which the energy complexity of the GHS algorithm is at least $\Omega(n \cdot LMST(V))$.*

3.2 An Almost Optimal Algorithm

We present a distributed algorithm with energy complexity $O(\sqrt{n}\log^3 n \cdot LMST(V))$, which is within a factor of $O(\log^3 n)$ of the optimal energy complexity. The algorithm constructs a tree with cost within a constant factor of the cost of MST. The algorithm proceeds in stages, as in the GHS algorithm. In the GHS algorithm, initially at the first stage, each single node forms one singleton fragment. In each subsequent stage, each fragment finds its minimum outgoing edge (MOE) and merges with another fragment using the MOE. The algorithm continues until the whole network is connected in one fragment. The details of the algorithm can be found in [9]. Although GHS algorithm is message optimal, its energy complexity can be as much as $n \cdot LMST(V)$ in arbitrary distribution (Corollary 3.2). To improve energy complexity, we modify only how a fragment finds its minimum outgoing edge; the rest of the algorithm works exactly as GHS algorithm, where each fragment uses the following FINDMOE algorithm as a subroutine to find its MOE. The algorithm FINDMOE may not find the exact minimum out going edge at each stage; however, it finds an edge with length at most twice the length of the minimum outgoing edge, leading to a constant factor approximation to the MST overall.

Algorithm FINDMOE: Each fragment F performs the following steps to compute an approximate minimum outgoing edge (AMOE).

1. Perform a pre-order traversal $\mathcal{T}(F)$ of the tree F. This can be easily done in a distributed manner using energy at most $2 \cdot LMST(F)$. Let the nodes and edges in this tour be $v_1, \ldots, v_m$ and $e_1, \ldots, e_{m-1}$, respectively. Note that some edges might be duplicated in this order, and some nodes could appear multiple times, but $m \leq 2|F|$. Let $L = \sum_{i=1}^{m-1} \ell(e_i)$.
2. The algorithm runs in rounds. In the ith round, a fragment executes the following steps.
   (a) A subset $A_i = \{v_{i(1)}, \ldots, v_{i(k(i))}\}$ of $k(i)$ nodes is chosen such that the following conditions hold. Let $S_j(A_i)$ denote the set of nodes of $\mathcal{T}(F)$ that lie between the jth and $j+1$st nodes of A_i, i.e., $S_j(A_i) = \{v_{i(j)+1}, v_{i(j)+2}, \ldots, v_{i(j+1)}\}$. Let $E_j(A_i)$ denote the edges in the segment formed by $S_j(A_i)$, i.e., $E_j = \{(v_{i(j)+1}, v_{i(j)+2}), \ldots, (v_{i(j+1)-1}, v_{i(j+1)})\}$. We need the set A_i to ensure that for each j, we have (i) $k(i) \leq c' \cdot L/2^i$ for a constant c', and either (ii) $|E_j| = 1$ with the only edge in E_j having length at least 2^i, or (iii) $\sum_{e \in E_j} \ell(e) \leq c \cdot 2^i$, for some constant c. Such an A_i can be easily constructed in a distributed manner using energy $2 \cdot LMST(F)$, by maintaining prefix sums of the sequence $\mathcal{T}(F)$.
   (b) Each node $v_{i(j)} \in A_i$ broadcasts its fragment id using a radius of $c \cdot 2^i$. Any node w that lies in a different fragment that hears this message sends back an acknowledgement. If there are multiple such nodes w that can respond, they break ties by communicating to their fragment leader in time proportional to the energy of that fragment. If node $v_{i(j)}$ gets back such a response from some node w in a different fragment, it communicates to all other nodes in A_i and this fragment stops this phase. This communication can be done on the tree F using work $LMST(F)$.
3. The edge $(v_{i(j)}, w)$ is used as the AMOE for fragment F.

Lemma 3.2 *The cost of the AMOE chosen by any fragment F in any phase is within a factor of 2 of its minimum outgoing edge (MOE).*

Proof: In the above algorithm, if phases $1, \ldots, i-1$ have been unsuccessful, it means that the MOE for this fragment has length at least 2^i. Suppose phase i is successful, and some node $v_{i(j)} \in A_i$ gets a response from some node w in a different fragment. By construction, $\ell(v_{i(j)}, w) \leq 2^{i+1}$, and hence the lemma follows. □

Lemma 3.3 *In any phase, the energy done by fragment F in computing an AMOE is at most $\sqrt{n}\log^2 n(LMST(F) + \ell(MOE(F))^2)$, where $MOE(F)$ denotes the minimum outgoing edge for this fragment F.*

Proof: The step of constructing the traversal $\mathcal{T}(F)$ only requires a total energy of $O(LMST(F))$. Next, we consider the steps in any round i of the algorithm. Constructing the set A_i, and communicating with all nodes in A_i if an AMOE is found in step 2(b) takes energy $O(LMST(F))$, by doing all the communication using

only edges of F. Recall the definition of the quantity L. Let $L_2 = \sum_{e \in P} \ell(e)^2$; note that $L_2 \leq c \cdot LMST(F)$ for some constant c. Each node in A_i explores within the range 2^i, and so the total energy incurred in this phase is at most $k(i)2^{2i} + L_2$, where the first term of $k(i)2^{2i}$ comes from sets $S_j(A_i)$ which have more than one edge, and the second comes from considering sets $S_j(A_i)$ which have a single edge. Since $k(i) \leq c'L/2^i$, the energy done in this round is at most $c'L2^i + L_2$. There can be at most $c'' \log n$ rounds, so the cost of second term over all rounds is at most $O(L_2 \log n)$. We now bound the cost of the first term above, over all rounds. Let $d = 2^{i'}$ denote the length of the MOE. Then, the above process continues for $\Theta(i')$ rounds. Below, we derive a bound for $\sum_{i \leq i'} c'L2^i$. We have two cases:

Case 1: $d \leq L/\sqrt{n}$. In this case, we have $\sum_{i \leq i'} c'L2^i \leq c_1 L 2^{i'} \leq c_1 L^2/\sqrt{n}$, where the last inequality follows from the fact that $d \leq L/\sqrt{n}$. Let m_t be the edges in $\mathcal{T}(F)$ whose lengths are in the range $[2^t, 2^{t+1}]$, so that we have $L = \sum_t m_t 2^t$. Let s be an index such that $m_s 2^s \geq m_t 2^t$ for any $t \neq s$. Then, we have $m_t 2^t \leq L \leq m_t 2^t \log n$ and $L_2 \geq m_t (2^t)^2$. This implies,

$$L^2/\sqrt{n} \leq \frac{(m_t 2^t \log n)^2}{\sqrt{n}} \leq \frac{m_t L_2 \log^2 n}{\sqrt{n}} \leq L_2 \sqrt{n} \log^2 n,$$

where the last inequality follows because $m_t \leq n$. Therefore, the energy complexity in this case is at most $L_2 \sqrt{n} \log^2 n$.

Case 2: $d > L/\sqrt{n}$. In this case, we separate the sum $\sum_{i \leq i'} k(i)2^{2i} = \sum_{i \leq i''} k(i)2^{2i} + \sum_{i=i''}^{i'} k(i)2^{2i}$, where i'' is an index such that $2^{i''} = L/\sqrt{n}$. From the bound on $k(i)$ and Case 1, we have $\sum_{i \leq i''} k(i)2^{2i} \leq \sum_{i \leq i''} c'L2^i \leq c_1 L_2 \sqrt{n} \log^2 n$. For $i > i''$, we have $k(i) \leq \sqrt{n}$. Therefore, we have $\sum_{i=i''}^{i'} k(i)2^{2i} \leq c_2 \sqrt{n} 2^{2i'} = c_2 \sqrt{n} d^2$ for a constant c_2. Putting everything together, the total energy in this case is at most $c_1 L_2 \sqrt{n} \log^2 n + c_2 \sqrt{n} d^2$.

Therefore, the total energy for finding the AMOE in any phase is at most $c_1 L_2 \sqrt{n} \log^2 n + c_2 \sqrt{n} d^2 = O(\sqrt{n} \log^2 n (LMST(F) + d^2)$. □

Lemma 3.4 *The final tree constructed above is a constant factor approximation to the MST.*

Proof: Consider the ith phase of the above algorithm. In this phase, the algorithm chooses an outgoing edge of length at most $c \cdot 2^i$ between two fragments V' and V'', if the shortest edge between these two sets is in the range $(2^i, 2^{i+1}]$, for a constant c.

Let $\mathcal{C}_i = \{C_1^i, \ldots, C_{n_i}^i\}$ be the set of components formed among the set V, if all edges of length greater than 2^i are deleted. Let $n_i = |\mathcal{C}_i|$ be the number of components in $\mathcal{C}_i$. Let T_{opt} be the minimum spanning tree on the set V of nodes. Let $\mathcal{D}_i$ denote the set of components formed by the above algorithm at the end of phase i. It is easy to prove by induction that $\sum_i n_i 2^{i\alpha} = \Theta(\sum_{e \in T_{opt}} \ell(e)^\alpha)$. It can be seen that the set $\mathcal{D}_i$ of components constructed by the algorithm at the end of phase i is a coarsening of $\mathcal{C}_{i-c'}$, for a constant c', i.e., if two nodes u and v are in the same component in $\mathcal{C}_{i-c'}$, they are in the same component in $\mathcal{D}_i$. Therefore, the cost of the tree constructed by the algorithm is within a constant factor of the optimum, for any fixed $\alpha \geq 1$. □

Theorem 3.1 *For any instance V, the total energy needed by the above algorithm is $O(LMST(V)\sqrt{n}\log^3 n)$.*

Proof: For any fragment F, the total energy done is $O(\sqrt{n}\log^2 n(LMST(F) + \ell(MOE(F))^2)$, where $MOE(F)$ denotes the minimum outgoing edge for this fragment. Therefore, the total energy done in any phase is $O(\sqrt{n}\log^2 n \cdot LMST(V))$, since the fragments are all disjoint in any phase. Since there are $O(\log n)$ phases, this leads to a total energy of $O(\sqrt{n}\log^3 n \cdot LMST(V))$. □

4 Concluding Remarks

We showed that, without coordinate information, the lower bound on energy complexity to construct any spanning tree, hence also MST, is $\Omega(\log n)$, under random distribution of nodes. With coordinate information, the best known lower bound is $\Omega(1)$ for random distributions. For both random and arbitrary distributions, we showed an (almost) energy-optimal algorithm that gives a constant approximation to the MST. An important question is to ask whether there is an energy-optimal algorithm to construct an (exact) MST when the coordinates are given to the nodes.

References

[1] C. Ambuhl. An optimal bound for the mst algorithm to compute energy efficient broadcast trees in wireless networks. In *32nd ICALP*, pages 1139–1150, November 2005.

[2] B. Awerbuch, A. Baratz, and D. Peleg. Cost-sensitive analysis of communication protocols. In *9th ACM PODC*, pages 177–187, 1990.

[3] Y. Choi, M. Khan, V.S.A. Kumar, and G. Pandurangan. Energy-optimal distributed algorithms for minimum spanning trees. In *20th ACM SPAA*, June 2008.

[4] A. Clementi, P. Crescenzi, P. Penna, G. Rossi, and P. Vocca. On the complexity of computing minimum energy consumption broadcast subgraph. In *18th STACS*, pages 121–131, June 2001.

[5] T. Cormen, C. Leiserson, and R. Rivest. *Introduction to Algorithms*. The MIT Press, 1990.

[6] K. Delin and S. Jackson. Sensor web for in situ exploration of gaseous biosignatures. In *Proceedings of IEEE Aerospace Conference*, March 2000.

[7] J. Deuschel and A. Pisztora. Surface order large deviations for high-density percolation. *Probability Theory and Related Fields*, 104(4):467–482, 1996.

[8] M. Elkin. A faster distributed protocol for constructing minimum spanning tree. In *SODA*, pages 352–361, 2004.

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

[10] G. Grimmet. *Percolation*. Springer-Verlag, 1999.

[11] P. Gupta and P. R. Kumar. Critical power for asymptotic connectivity. In *Proceedings of the Conference on Decision and Control*, 1998.

[12] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan. An application-specific protocol architecture for wireless microsensor networks. *IEEE Transactions on Wireless Communications*, 1(4):660–670, October 2002.

[13] C. Intanagonwiwat, D. Estrin, R. Govindan, and J. Heidemann. Impact of network density on data aggregation in wireless sensor networks. In *ICDCS*, July 2002.

[14] M. Khan and G. Pandurangan. A fast distributed approximation algorithm for minimum spanning trees. *Distributed Computing*, 20(6):391–402, April 2008.

[15] M. Khan, G. Pandurangan, and V.S.A. Kumar. Distributed algorithms for constructing approximate minimum spanning trees in wireless networks. *IEEE Transactions on Parallel and Distributed Systems*, 2008 (in press). available at http://staff.vbi.vt.edu/maleq/papers/TPDS.pdf.

[16] E. Korach, S. Moran, and S. Zaks. Optimal lower bounds for some distributed algorithms for a complete network of processors. *Theoretical Computer Science*, 64:125–132, 1989.

[17] B. Krishnamachari, D. Estrin, and S. Wicker. The impact of data aggregation in wireless sensor networks. In *International Workshop on Distributed Event-Based Systems*, July 2002.

[18] N. Lynch. *Distributed Algorithms*. Morgan Kaufmann Publishers, San Mateo, CA, 1996.

[19] M. Mitzenmacher and E. Upfal. *Probability and Computing*. Cambridge University Press, 2005.

[20] S. Muthukrishnan and G. Pandurangan. The bin-covering technique for thresholding random geometric graph properties. In *SODA*, 2005.

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

[22] T. Przytycka and L. Higham. Optimal cost-sensitive distributed minimum spanning tree algorithm. In *SWAT*, pages 246–258, 1996.

[23] E. Santis, F. Grandoni, and A. Panconesi. Fast low degree connectivity of ad-hoc networks via percolation. In *ESA*, pages 206–217, 2007.

[24] M. Steele. Asymtotics for euclidian minimal spanning trees on random points. *Probability Theory and Related Fields*, 92:247–258, 1992.

[25] P. Wan, G. Calinescu, X. Li, and O. Frieder. Minimum-energy broadcasting in static ad hoc wireless networks. *Wireless Networks*, 8(6):607–617, November 2002.

Appendix

A Proof of Theorem 2.3

A.1 The Giant Component

We prove the first part of Theorem 2.3, which says about the giant component. The overall proof of Theorem 2.3 is similar to that in [23]. The basic idea is to reduce our problem to site percolation in a finite box. In the original site percolation problem, we consider an infinite grid of cells, where each site is occupied with probability p, and we ask the probability at which an infinite cluster of sites emerges. It is well known that there is a critical probability p (denoted by p_o), below which the probability that an infinite cluster exists is asymptotically 0 and above which the probability is asymptotically 1. It is also known that in the supercritical phase ($p > p_o$), with high probability, there is a unique giant cluster in the box and that its complement consists of small regions, each containing $O(\log^2 n)$ sites [10].

To do this reduction, we first replace the uniform distribution of nodes with a Poisson distribution to exploit the strong independence property of the latter. That is, a distribution of nodes in one region does not affect the distribution of nodes in any other disjoint region. There is an easy way to connect these two settings (cf. [23]), and we can safely assume that we have n nodes that are generated by Poisson processes in a unit square. Here we repeat the same arguments and lemma as [23] because we need them later.

We consider two Poisson processes P_0 and P_t. Process P_0 has parameter $\mu_0 := n - \epsilon n$, where ϵ is a small positive constant. Process P_t is built on top of P_0 by adding to it a new independent Poisson process ΔP with parameter $2\epsilon n$. It is well known that P_t is a Poisson process with parameter $\mu_t := \mu_0 + 2\varepsilon n = n + \varepsilon n$. We then define a sequence of point processes $\{Q_i\}$ sandwiched between P_0 and P_t. Starting from $Q_0 := P_0$, Q_{i+1} is given by Q_i by adding one point chosen uniformly at random in $P_t - Q_i$. Our reduction to site percolation will apply simultaneously to all Q_i's, showing the existence of a unique giant component for each Q_i with high probability. Each Q_i generates points uniformly in the box (conditioned on the given number of points). The next lemma shows that, with high probability, one of the Q_i will generate exactly n points. As a consequence, if something holds for all Q_i's simultaneously, it also holds for the original n-nodes problem.

Lemma A.1 *[23] Let N_0 and N_t be the Poisson variables relative to P_0 and P_t, respectively. There is a positive constant γ such that*

$$Pr\left(\overline{\{N_0 \leq n \leq N_t\}}\right) \leq e^{-\gamma n}.$$

We now introduce site percolation problem by subdividing the unit square into a grid of non-overlapping square cells as shown in Figure 1(a). Let $r = \sqrt{\frac{c}{n}}$ be the transmission radius where c is a constant, which will be fixed. Setting the transmission radius $r = \sqrt{\frac{c}{n}}$, we get an infinite grid of cells as n grows. To simplify our analysis, we define the distance between two nodes $u = (x_1, y_1)$ and $v = (x_2, y_2)$ as $\max(|x_1 - x_2|, |y_1 - y_2|)$ instead of using Euclidean distance. This simplification affects our energy complexity bounds only up to a constant factor. We set the length of a side of each cell to $\frac{r}{2}$ so that any two nodes in the neighboring cells are connected. Thus, all nodes in a cluster of occupied cells will be a connected component. The expected number of nodes in each cell is $\frac{c}{4}$. Let us define a cell to be *good* if the number of nodes inside the cell is greater than or equal to $\frac{c}{8}$.

Lemma A.2 *Let p_c be the probability that a cell is good. Then* $\lim_{c\to\infty} p_c \to 1$.

Proof: It follows from the large deviation principle of the Poisson random variables. □

Now we establish the theorem about the giant component by showing that the largest cluster of good cells form a giant component of nodes. Clearly, the giant component also includes the nodes in any occupied cells that are connected to the largest cluster of good cells.

Lemma A.3 *Let $\mathcal{G}$ be the largest component of nodes when the transmission radius $r = \sqrt{\frac{c}{n}}$. For any constant $\alpha \in (\frac{1}{4}, \frac{1}{2})$, there is a choice of c such that for a positive constant γ,*

$$Pr(|\mathcal{G}| \leq \alpha n) \leq e^{-\gamma\sqrt{n}}$$

Proof: Let $m \simeq \frac{4n}{c}$ be the number of cells and C be the largest cluster of good cells. By Theorem 1.1 in [7], for any given constant $\delta \in (0, \frac{1}{2})$, there is a value of p_c such that

$$Pr(|C| \leq (1-\delta)m) \leq e^{-\gamma_1\sqrt{m}}.$$

By definition, a good cell contains at least $\frac{c}{8}$ nodes. Thus, if C contains at least $(1-\delta)m$ good cells, then its corresponding component contains at least $(1-\delta)cm/8$ nodes.

$$\begin{aligned} Pr(|\mathcal{G}| \leq \alpha n) &\leq Pr(|C| \leq 8\alpha n/c) + Pr(\overline{\{N_0 \leq n \leq N_t\}}) \\ &= Pr(|C| \leq 2\alpha \cdot 4n/c) + Pr(\overline{\{N_0 \leq n \leq N_t\}}) \\ &\leq e^{-\gamma_1\sqrt{4n/c}} + e^{-\gamma_2 n} \leq e^{-\gamma\sqrt{n}} \end{aligned}$$

for some positive constant γ and large n. The first inequality follows from the fact that we do the reduction only if the condition $Pr(\{N_0 \leq n \leq N_t\})$ holds. □

A.2 The Small Regions

We prove the second part of Theorem 2.3 and complete the whole proof. We assume that we have chosen the constant c_1 so that there exists a giant component with high probability. Let us consider the complement of the largest cluster of good cells. We now show that the maximal connected clusters of cells in the complement of C are small clusters. We call this maximal connected cluster a *small region*. In Figure 1(b), gray area represents these small regions. Definitely small components of nodes will be inside this small region. The lemma below bounds the number of cells in a small region.

Lemma A.4 *Let $|S|$ be the number of cells in region S. For any small region S and some positive constant γ,*

$$Pr(|S| = k) \leq e^{-\gamma\sqrt{k}}$$

Proof: It follows from the result in the supercritical phase for site percolation [10]. □

The lemma below bounds the number of nodes in a small region.

Lemma A.5 *Let Z_i be the random variable that represents the number of nodes in cell i, and let S be a small region. For large n, there is a positive constant γ such that*

$$Pr(\sum_{i \in S} Z_i > h) \leq e^{-\gamma\sqrt{h}}.$$

Proof:

$$\begin{aligned} Pr\left(\sum_{i\in S} Z_i > h\right) &= \sum_k Pr\left(\sum_{i\in S} Z_i > h \mid |S| = k\right) Pr(|S| = k) = \sum_k Pr\left(\sum_{i\leq k} Z_i > h\right) Pr(|S| = k) \\ &= \sum_{k \leq \frac{2h}{c}} Pr\left(\sum_{i\leq k} Z_i > h\right) Pr(|S| = k) + \sum_{k > \frac{2h}{c}} Pr\left(\sum_{i\leq k} Z_i > h\right) Pr(|S| = k) \\ &\leq \sum_{k \leq \frac{2h}{c}} Pr\left(\sum_{i\leq k} Z_i > h\right) + \sum_{k > \frac{2h}{c}} Pr(|S| = k) \leq \sum_{k \leq \frac{2h}{c}} Pr\left(\sum_{i\leq \frac{2h}{c}} Z_i > h\right) + \sum_{k > \frac{2h}{c}} Pr(|S| = k) \\ &= \frac{2h}{c} Pr\left(\sum_{i\leq \frac{2h}{c}} Z_i > h\right) + \sum_{k > \frac{2h}{c}} Pr(|S| = k) \leq \frac{2h}{c} e^{-\gamma_1 h} + \sum_{k > \frac{2h}{c}} e^{-\gamma_2\sqrt{k}} \leq e^{-\gamma\sqrt{h}}. \end{aligned}$$

The second last inequality follows from the large deviation principle and Lemma A.4. □

The following lemma completes the proof of Theorem 2.3. Let us consider the event $\mathcal{E}$: with the transmission radius $r = \sqrt{\frac{c}{n}}$, there is a unique giant component containing at least αn nodes and all remaining components of nodes are trapped inside small regions, each of which contains at most $\beta \log^2 n$ nodes.

Lemma A.6 *When n is large, for every $\alpha \in (\frac{1}{4}, \frac{1}{2})$ and appropriate constants c and β,*

$$Pr(\overline{\mathcal{E}}) \leq n^{-d}$$

where d is a positive constant.

Proof: By Lemma A.3, the probability that there is no component with at least αn nodes is at most $e^{-\gamma_1\sqrt{n}}$. By Lemma A.5 and union bound, the probability that there exists a small region with more than $\beta \log^2 n$ nodes is at most $ne^{-\gamma_2\sqrt{\beta \log^2 n}} = n^{1-\gamma_2\sqrt{\beta}}$. Thus,

$$Pr(\overline{\mathcal{E}}) \leq e^{-\gamma_1\sqrt{n}} + n^{1-\gamma_2\sqrt{\beta}} \leq n^{-d}$$

for some positive constant d and large n by choosing β appropriately. □

B Proof of Lemma 2.2

Let S be a small region containing a node v at the end of Step 1. Let also N_v be the total number of messages needed by v during the modified GHS algorithm in Step 2. Then, N_v is bounded by $c \log F_S$ where c is some constant and F_S is the number of fragments in S at the end of Step 1. Let N_S be the number of messages needed by all nodes in S, that is, $N_S = \sum_{v \in S} N_v$. As in Lemma A.5, Z_i represents the number of nodes in cell i. Thus F_S is bounded by $\sum_{i \in S} Z_i$, which is the total number of nodes in S. The following inequalities complete the proof by showing that $\mathbf{E}[N_S]$ is bounded by some constant.

$$\begin{aligned}
\mathbf{E}[N_S] &\leq \mathbf{E}\left[\left(\sum_{i \in S} Z_i\right) c \log F_S\right] \\
&\leq c \cdot \mathbf{E}\left[\left(\sum_{i \in S} Z_i\right) \log\left(\sum_{i \in S} Z_i\right)\right] \\
&\leq c \cdot \sum_h h \log h \cdot Pr\left(\sum_{i \in S} Z_i \geq h\right) \\
&\leq c \cdot \sum_h h \log h \cdot e^{-\gamma\sqrt{h}} < \infty.
\end{aligned}$$

The bound on $Pr\left(\sum_{i \in S} Z_i \geq h\right)$ follows from Lemma A.5.