Brief Announcement: Faster Asynchronous MST and Low Diameter Tree Construction with Sublinear Communication

Ali Mashreghi
Department of Computer Science, University of Victoria, BC, Canada
ali.mashreghi87@gmail.com

Valerie King
Department of Computer Science, University of Victoria, BC, Canada
val@uvic.ca

Abstract

Building a spanning tree, minimum spanning tree (MST), and BFS tree in a distributed network are fundamental problems which are still not fully understood in terms of time and communication cost. The first work to succeed in computing a spanning tree with communication sublinear in the number of edges in an asynchronous CONGEST network appeared in DISC 2018. That algorithm which constructs an MST is sequential in the worst case; its running time is proportional to the total number of messages sent. Our paper matches its message complexity but brings the running time down to linear in n. Our techniques can also be used to provide an asynchronous algorithm with sublinear communication to construct a tree in which the distance from a source to each node is within an additive term of $\sqrt{n}$ of its actual distance.

2012 ACM Subject Classification Theory of computation → Distributed computing models

Keywords and phrases Distributed Computing, Minimum Spanning Tree, Broadcast Tree

Digital Object Identifier 10.4230/LIPIcs.DISC.2019.49

Related Version `https://arxiv.org/abs/1907.12152`

Funding *Ali Mashreghi*: Funded with an NSERC grant.
Valerie King: Funded with an NSERC grant.

1 Problem and Results

A distributed network of processes can be represented as an undirected graph $G = (V, E)$, where $|V| = n$ and $|E| = m$. Each node corresponds to a process and each edge corresponds to a communication link between the two processes. The nodes can communicate only by passing messages to each other. Computing the spanning tree and the minimum spanning tree (MST) are problems of fundamental importance in distributed computing. Efficient solutions for building these trees directly improve the solution to other distributed computing problems or at least provide valuable insights. Leader election, counting, and shortest path tree are examples of such problems. The breadth-first search tree (BFS) is also important; it can be used to simulate a synchronous algorithm in an asynchronous network.

The problem of constructing an MST in a distributed network has been studied for many years. In the earlier works, researchers focused on improving the time complexity since it was believed that any spanning tree algorithm in the CONGEST model would require $\Omega(m)$ messages (See [2]). After the algorithm of King et al. [5] which constructs the MST in the synchronous CONGEST model in $\tilde{O}(n)$ time and messages, there has been renewed interest in message complexity. Mashreghi and King [6] achieved the first algorithm to compute a spanning tree in an asynchronous CONGEST model with $o(m)$ communication complexity

33rd International Symposium on Distributed Computing (DISC 2019).
Editor: Jukka Suomela; Article No. 49; pp. 49:1–49:3
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

when m is sufficiently large. However, the time complexity of their algorithm matches its communication complexity, $\tilde{O}(n^{3/2})$; in the worst case, their algorithm essentially operates in a sequential manner.

In this work, we match the message complexity of [6] but bring the running time down to $O(n)$, a time which matches the time of the fastest known asynchronous MST algorithms that use $\Theta(m)$ communication [1]. Full version of our paper is available on Arxiv [7].

The classic Layered BFS algorithm for the asynchronous CONGEST network uses $O(D^2)$ time and $O(Dn + m)$ messages, where D is the diameter of the network. We show how to construct a "nearly BFS" tree in this model with sublinear number of messages (for small enough D, and sufficiently large m) such that for each node, the distance from the source node is within an additive term of $O(\sqrt{n})$ from its actual distance in the network. Such a tree can be used to simulate a synchronous algorithm in an asynchronous network with an overhead of $O(D + \sqrt{n})$ time per step. To the best of our knowledge, there is no previously known algorithm to construct a low diameter tree using a sublinear number of messages in an asynchronous network. Specifically, we show:

▶ **Theorem 1.** *There exists an asynchronous algorithm in the KT1 CONGEST model that, w.h.p. computes the MST in $O(n)$ time and with $O(\min\{m, n^{3/2}\log^2 n\})$ messages.*

This result achieves communication sublinear in m when m is sufficiently large, and is optimal for time when the diameter is $\Theta(n)$. We also prove the following more general theorem.

▶ **Theorem 2.** *Given an asynchronous MST algorithm with time $T(n, m)$ and message complexity of $M(n, m)$ in the KT1 CONGEST model, w.h.p., the MST in an asynchronous network can be constructed in $O(n^{1-2\epsilon} + T(n, n^{3/2+\epsilon}))$ time and $\tilde{O}(n^{3/2+\epsilon} + M(n, n^{3/2+\epsilon}))$ messages, for $\epsilon \in [0, 1/4]$.*

For the BFS problem we show:

▶ **Theorem 3.** *In an asynchronous KT1 CONGEST model, a network with diameter D can construct a spanning tree with $O(D + \sqrt{n})$ diameter using time $O(D^2 + n)$ and messages $\tilde{O}(n^{3/2} + nD)$.*

Model

We consider the asynchronous CONGEST model. Messages sent by nodes are delayed arbitrarily. Communication is event-driven, in that actions are taken upon receiving a message or waking up. We assume that all nodes wake up at the start. Time complexity in the asynchronous communication model is the worst-case execution time, if each message takes at most one time unit to traverse one edge. The time for computations within a node is not considered.

All nodes have knowledge of n, the size of the network, within a constant factor. In the CONGEST model, each message has $O(\log n)$ bits. ID's are unique and are taken from the range of $[1, poly(n)]$. We assume that messages to a receiver are numbered by a sender in the order in which they are sent to it, so that a node receiving the message can wait for the message from a sender with the sender's next number before acting. W.l.o.g., we assume that the edge weights are unique and therefore, the MST is unique.

Nodes initially know their own ID and the ID of their neighbors. This is known as the KT1 model and is considered by some to be the standard model of distributed computing [8]. In weighted graphs, initially, nodes only know the weight of their incident edges in the input graph G. We assume that all nodes wake up at the same time. For constructing a subgraph like MST, the objective is that, upon termination of the protocol, all nodes must know which of their incident edges belong to the subgraph.

2 Outline of algorithms

We achieve some parallelism by starting with *initial* fragments of height 1, formed by high degree nodes and star nodes in parallel. A node is *high-degree* if its degree is at least $\sqrt{n}\log^2 n$. Otherwise, it is a *low-degree* node. A node selects itself to be a *star node* with probability of $\frac{c}{\sqrt{n}\log n}$ where c is a constant dependent on c' so that each high-degree node is adjacent to a star node with probability $1 - 1/n^{c'}$. Let G' be the subgraph on G induced by all high-degree and star nodes. We can construct a spanning forest F on G', from the initial fragments by adding $O(\sqrt{n}/\log n)$ edges such that the sum of the diameter of all trees in this forest is $O(\sqrt{n}/\log n)$. Thus we can afford to add these edges sequentially in the worst case, spending a time per edge proportional to $\log^2 n *$ (diameter of a maximum tree), for a total of $O(n)$ time. To find minimum outgoing edges, one approach is to have nodes test all of their incident edges, in the order of weight, to see whether an edge is outgoing. This results in $\Omega(m)$ messages. King et al. [5] provided an *asynchronous* subroutine, called FindAny, that with constant probability, finds an edge leaving a fragment T and uses only $\tilde{O}(|T|)$ messages, where $|T|$ is the number of nodes in T. The algorithm has three parts as follows:

1. Initial fragments are formed in parallel consisting of star nodes and their high degree neighbors. Our new algorithm MAXIMALTREE incorporates the asynchronous waiting technique of [6] to find the edges in F and enable each high-degree node to learn its low-degree neighbors while building a spanning forest F on G' from these initial fragments. To construct the nearly BFS tree, the nodes run Gallager's Layered BFS algorithm [3] on G_{sparse}, the subgraph of edges in F and those incident to at least one low-degree node.
2. We compute the minimum weight spanning forest F_{min} on G'. We use an idea of [4], along with the fact that the obtained trees in part (1) all have diameter of $O(\sqrt{n})$. This is done using the low-diameter trees in F, to simulate the MST algorithm of [4] in the asynchronous model on each connected component of G'.
3. We define S_{min} to be the edges in F_{min} and the edges incident to at least one low-degree node. We run an asynchronous MST algorithm with $O(n)$ time and $O(m)$ message complexity (e.g. [1]) on S_{min}. The result is the MST of G.

References

1 Baruch Awerbuch. Optimal distributed algorithms for minimum weight spanning tree, counting, leader election, and related problems. In *STOC*, pages 230–240, 1987.

2 Baruch Awerbuch, Oded Goldreich, Ronen Vainish, and David Peleg. A trade-off between information and communication in broadcast protocols. *Journal of the ACM*, 37(2):238–256, 1990.

3 R.G. Gallager. Distributed minimum hop algorithms. Technical Report LIDS-P-1175, MIT Laboratory for Information and Decision Systems, 1982.

4 Mohsen Ghaffari and Fabian Kuhn. Distributed MST and Broadcast with Fewer Messages, and Faster Gossiping. In *DISC*, pages 30:1–30:12, 2018.

5 Valerie King, Shay Kutten, and Mikkel Thorup. Construction and impromptu repair of an MST in a distributed network with o(m) communication. In *PODC*, pages 71–80, 2015.

6 Ali Mashreghi and Valerie King. Broadcast and Minimum Spanning Tree with $o(m)$ Messages in the Asynchronous CONGEST Model. In *DISC*, volume 121, pages 37:1–37:17, 2018.

7 Ali Mashreghi and Valerie King. Faster asynchronous MST and low diameter tree construction with sublinear communication. *arXiv e-prints*, July 2019. `arXiv:1907.12152`.

8 David Peleg. Distributed computing. *SIAM Monographs on discrete mathematics and applications*, 5, 2000.