Notes on Distributive Computing and MST Construction

Sarah Clapoff

December 12, 2022

1 Introduction

Previously, the solutions to all of our problems had been in regards to a single computer or processor. However, we can look at analogous problems when we consider a distributive network, with each processor being a node and the edges being communication links. In this way, the nodes can talk to each other by sending messages along the edges.

Here, we will consider the important problem of finding and constructing a minimum spanning tree (MST) for a communications network. One example of this type of network (or a way to visualize the problem) is trivially multiple houses with landlines, each broadcasting messages to their neighbours along the telephone lines. We want to find a way to represent this network that is efficient in both broadcasting messages and maintaining the network. This is a heavily researched problem, with applications used across distributed computing.

2 CONGEST Model

The CONGEST model is one way of many to model a distributed communications network. The synchronous model, consisting of n nodes, labeled with a unique $ID \in 1, 2, ..., n$, and m undirected edges. Each node also contains the following information: it's own ID, the ID of the other endpoint for all incident edges, and the upper bound of the size of the whole network. Each node is also able to communicate with its neighbours. The CONGEST model specifies that messages are $O(\log n)$ bits long. [2] Furthermore, the model is structured in rounds, where in a single round each node can send a message to all of its neighbours. The messages do not have to be the same across all neighbours.

3 MST Construction in a Distributed Network

3.1 General Idea

The problem we are looking at for our network is as follows: Suppose we have two nodes in a network that want to send the same message. This is redundant. We can fix this redundancy with a MST.

General Idea: We will build a MST to represent the network. Since each node is aware of its incident edges in a spanning tree, if a node wants to send a message it will first send it to its root node. Then, that root will send it to everyone else. That way, if two nodes are sending the same message, it will be broadcast only once by the root.

3.2 Building the MST

We will refer back to the Dynamic Monte Carlo algorithm in order to build the MST. In particular, we will modify the `FindAny` Algorithm for over a distributive network. [2], [3] For every node x, execute `BuildST`

`BuildST(`x`):`

1: $time \leftarrow 0$
2: **for** $i \leftarrow 1$ to $C\log n$ **do**
3: Elect a Leader in T_x *//Typically the node with the smallest ID.*
4: The leader initiates `FindAny`
5: **if** edge a, b is returned by `FindAny` **then**
6: a sends the message to b across a, b
7: a, b is added to the MST.
8: **end if**
9: **while** $time < i \cdot maxTimeST(n)$ **do**
10: wait.
11: If any message is received to add an edge, mark that edge and add it to the tree.
12: **end while**
13: Determine if there is a cycle and break it.
14: **end for**

Note that the leader is determined by starting at the leaves, comparing IDs with neighbours, and sending up to the root. How do we detect a cycle though?

3.3 Cycles: A Problem

When electing the leader, if a node doesn't hear from all of its neighbours (except for the two neighbouring on the cycle), then that node is in a cycle. We can fix the problem of being stuck in a cycle by randomly deleting edges, as follows.

After the maximum time, $i \cdot maxTimeST(n)$, has passed in choosing a leader, these nodes will be aware that they are in a cycle, since they know their neighbours as well as the neighbours

they have heard from in choosing the leader. Each node in the cycle will then randomly pick an edge in the cycle (there are only two options for cycle edges incident to a node) and send a message along it. If both endpoints pick the same edge, then it is unmarked (deleted from the tree). A leader is once again elected; if a cycle remains then all edges are unmarked.

The deletion of one edge does result in the breaking of the other edges in that cycle. This further results in an overall loss as leaders have to be re-elected; unfortunately there is no good way to do this asynchronously.

4 Analysis

In this distributive network, we are concerned with the efficiency regarding the number of rounds (time), as well as the number of messages sent (communication). As the algorithm used for constructing MSTs over our distributive network is very similar to the one used for the Dynamic Monte Carlo Algorithm, we refer the analysis to that of [3].

5 References

References

[1] King, V., In-class Lecture, CSC425, November 28, 2022
[2] Wang, W. and King, V., "Spanning Tree Construction in Distributed Network", CSC425 Scribe notes, Feb. 25, 2015
[3] Kapron, B.M., King, V., Mountjoy, B., "Dynamic graph connectivity in polylogarithmic worst case time" Proceedings of the twenty-fourth annual ACM-SIAM symposium on Discrete algo- rithms. Society for Industrial and Applied Mathematics, 2013.