Distributed Algorithms

Lecture 6

Lecture by Dr. Gadi Taubenfeld
Typeset by Steven Karas

2017-05-10
Last edited 18:18:31 2017-05-10

Disclaimer These lecture notes are based on the lecture for the course Distributed Algorithms, taught by Dr. Gadi Taubenfeld at IDC Herzliyah in the spring semester of 2017. Sections may be based on the lecture slides and accompanying book written by Dr. Gadi Taubenfeld.

Agenda

- Distributed MST

1 MST

Definition: Spanning Tree A spanning tree is a subgraph of a graph that is a tree that covers every vertex in the graph.

Definition: Fragment A subtree of a spanning tree. Conversely, a fragment is a tree subgraph that can be expanded into a spanning tree. The name of a fragment is its root.

We mark the edge leading from the fragment to the rest of the graph with the minimal weight as the "minimal outgoing edge".

Definition: Minimum Spanning Tree A spanning tree whose weights are minimal.

Property Let F be a fragment of an MST. Let e be a minimum outgoing edge of F. Thus, joining e and its adjascent vertex v to F gives us another fragment of an MST.

Proof There exists an MST that includes F, but does not include e. There exists another path whose first edge is e' in the graph from F to v. However, $w(e') \geq w(e)$, which means that we can remove e' from the MST and add e.

Property If all the edges of a connected graph have different weights, then the MST is unique. Let T, U be MSTs of such a graph. Let $e \in T$ and $e \notin U$. Note that e must form a cycle if it was added to U. Note that at least one of the edges in this cycle $e' \in U$ must not be in T. Note that $w(e) < w(e')$, and therefore $\{e\} \cup U - e'$ is an MST that is not U. Then it must not be an MST.

1.1 Prim-Dijkstra

Start from a vertex and add the minimal edge that keeps it as a tree. Repeat until all vertices added.

1.2 Kruskal

Sort edges and add from smallest to largest if it doesn't create a cycle. Takes $O(|E| \log |V|)$

2 Distributed MST

Published by R. G. Gallager et al in 1983[1].

Runs with $5n \log n + 2E$ message complexity, where each message contains $3 + 0 \ldots \log n$ bits. The time complexity is $O(n^2)$, unless all the processes start simultaneously, in which case it's $O(n \log n)$.

Model

- Asynchronous Network
- Symmetric code
- Undirected network with n processes and E links
- Unique weights for each link[2]
- Each process knows the weight of its links
- FIFO links
- No failures (of processes, links, messages)

2.1 Levels

A level 0 MST fragment is a fragment with 1 process.

Given two fragments with level i, the composed fragment formed by joining them has level $i + 1$.

Given two fragments L and M where $L < M$ such that L is connecting to M, the resulting fragment has level M.

Given two fragments L and M where $L > M$ such that L is connecting to M, then this should **never** happen, so it isn't interesting.

A more formal definition is that the level is the number of times a fragment was joined to another fragment of the same size.

2.2 Algorithm

This was described visually by Gadi, who did a much better job than I can in words.

Process states

- Sleeping - initial state
- Find - searching for the minimal outgoing edge in the fragment
- Found - all other times

Edge states

- Basic - initial state
- Branch - part of the MST
- Rejected - not part of the final MST

Fragments Each fragment tracks its level, and its "core"

Types of messages Note that a process does not reply to messages from a fragment with higher level.

1. initiate(L,F,S) - start search after merger.
2. test(L,F) - external edge?
3. reject - negative reply that the edge is an internal edge.
4. accept - affirmative reply that the edge can participate in the MST.
5. report(w) - report of edge with minimal weight.
6. change-root - send to the node with the minimal edge. Flips parent edges and when it arrives, initiates a connect.
7. connect(L) - request for merger.

Where L is the level, F is the fragment name, and S is either **find** or **found**.

The state is the current state of the process when sending the initiate message.

[1] A copy of the paper can be found alongside the lecture slides. The paper is a good example of a well-written paper

[2] We can relax this and will show how to at the end of the lecture

2.3 Message Complexity

Lemma The number of processes in a fragment of level L is greater or equal to 2^L. Proof by simple induction on the level. As such, the maximum level of an MST is $\log n$.

Proof An edge not in the MST is rejected once. This requires 2 messages (2 tests, or 1 reject and 1 test). This gives us $2E$ messages.

In each level except 0, the last level each node can receive: 1 initiate and 1 accept, and can send 1 successful test, 1 report, and 1 connect or 1 change-root. A node can be at $\log n - 1$ levels (except 0 and the last level). This gives us $5n(\log n - 1)$ messages.

3 Next week