Simulation of the Merge of Independent Chord Rings

Andrew Rosen Brendan Benshoof

Outline

File Lookup

- Most important function of a decentralized Peer-to-Peer system.
- Need to discover which node in the network has a certain file.
- Need to do it efficiently.

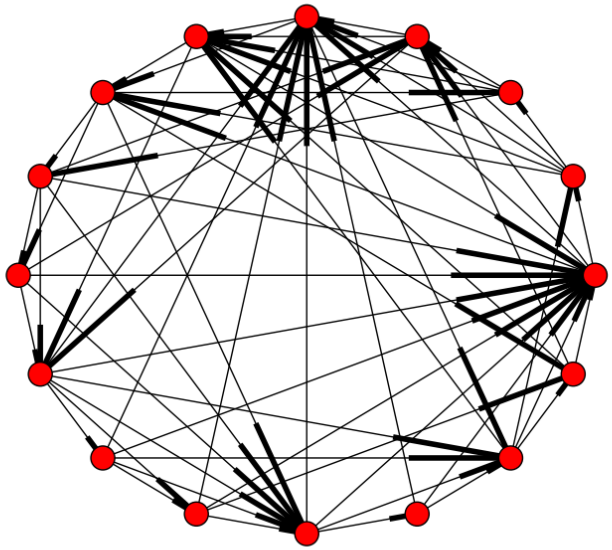
Hashing Protocols

- One of the best solutions for this task.
- Map nodes and files identifiers to keys.
- Nodes are responsible for files with keys that match some criteria.
- Imposes an structure on the network.

What is Chord

- Arranges a network into a ring.
- Maximum 2^m nodes in the network.
- Nodes and filenames are hashed to create an m-bit key.

How Are Files Stored



- Node responsible for for key κ is the *sucessor*(κ).
- *sucessor*(κ) is the node with key κ or first following key.

How Are Files Retrieved

- To find a file with key κ, we find *sucessor*(κ).
- We could just ask our successor, who would do the same.
- We use a *finger-table* make this quicker
  - *m* entries in the table.
  - Stores location of *successor*$(n + 2^{i-1})$ *mod* 2^m.
  - If κ is between us and an entry, skip to that entry.
- Once we know *sucessor*(κ), we can directly connect.
- This reduces the lookup time to $O(\log n)$

How Is Churn Handled

- Join a network by asking some node in it to find your successor.
- Periodically update the entries in the finger table.
- Periodically find your successor's predecessor.

Goals

We want to examine global behavior. As a result we ignore:

- Actual network topology outside of Chord.
- Real Network Latency.
- File storage.

We instead focus on:

- Overlay topology.
- What choices are made in a race condition.
- What happens when rings compete for nodes.
- Who ends up in which ring.

Agents

- Nodes for nodes, Links for links.
- Messages are represented by:
  - Seekers.
  - Updates.

Setup

- Generate nodes in random locations (ring visualization optional)
- Specified number of nodes for independent rings.
- Set the hash size.

Creating Rings

- Nodes not in a ring look within a radius for nodes in a ring
- Nodes then join the closest ring.

Creating Rings

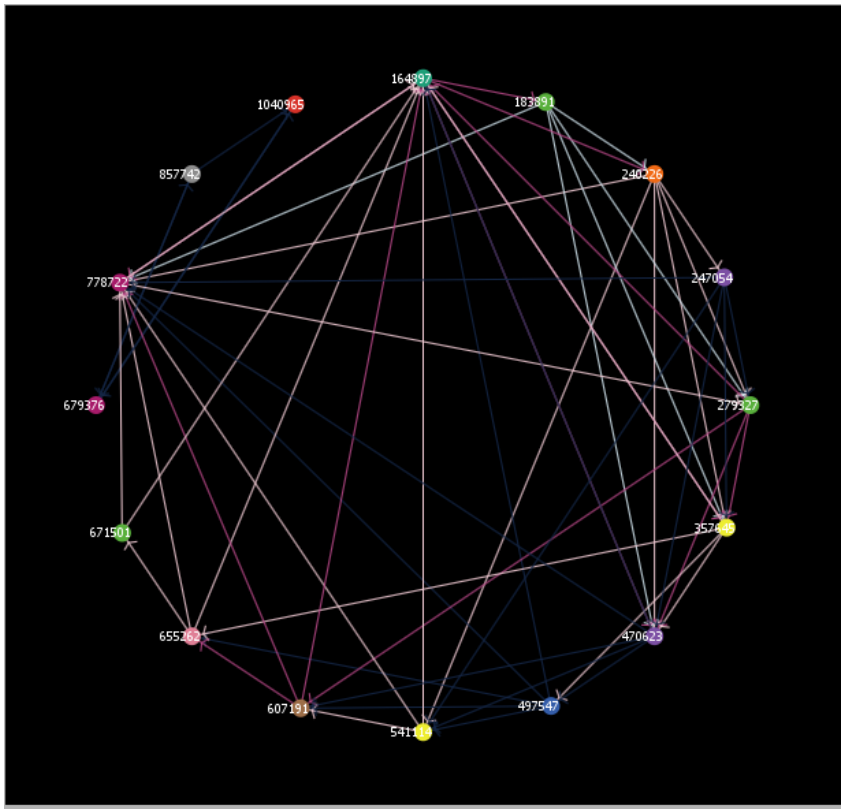


Figure: A chord network with multiple rings. Rings are differentiated by color.

Joining new Rings

- Nodes periodically invite a new node to the ring.
- Nodes accept the invitation if the new ring is "better."
- Accepting the invitation means joining like any new node.

Setup

- 200 simulations
  - 100 without any joining mechanism
  - 100 with our joining mechanism.
- Recorded the number of links over time.
- Analyzed for the mean and variance.

Results

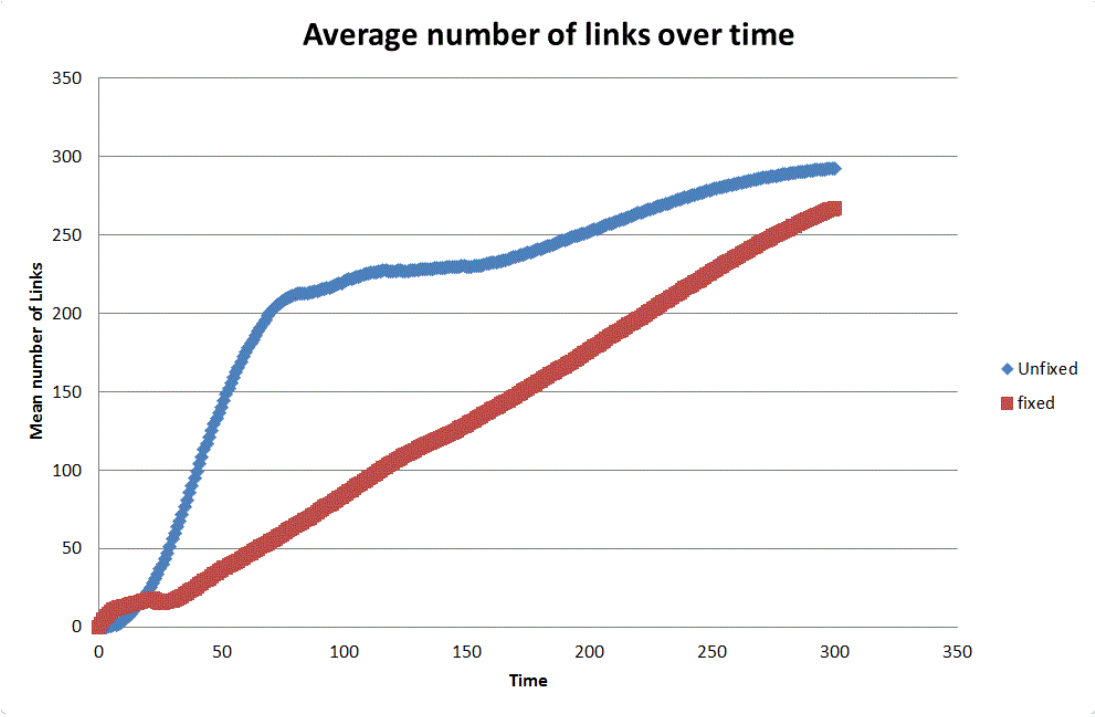


Figure: The unfixed simulation causes nodes to form links more rapidly. This is because nodes are not joining other rings, which would cut the number of links even as more nodes were discovered.

Results

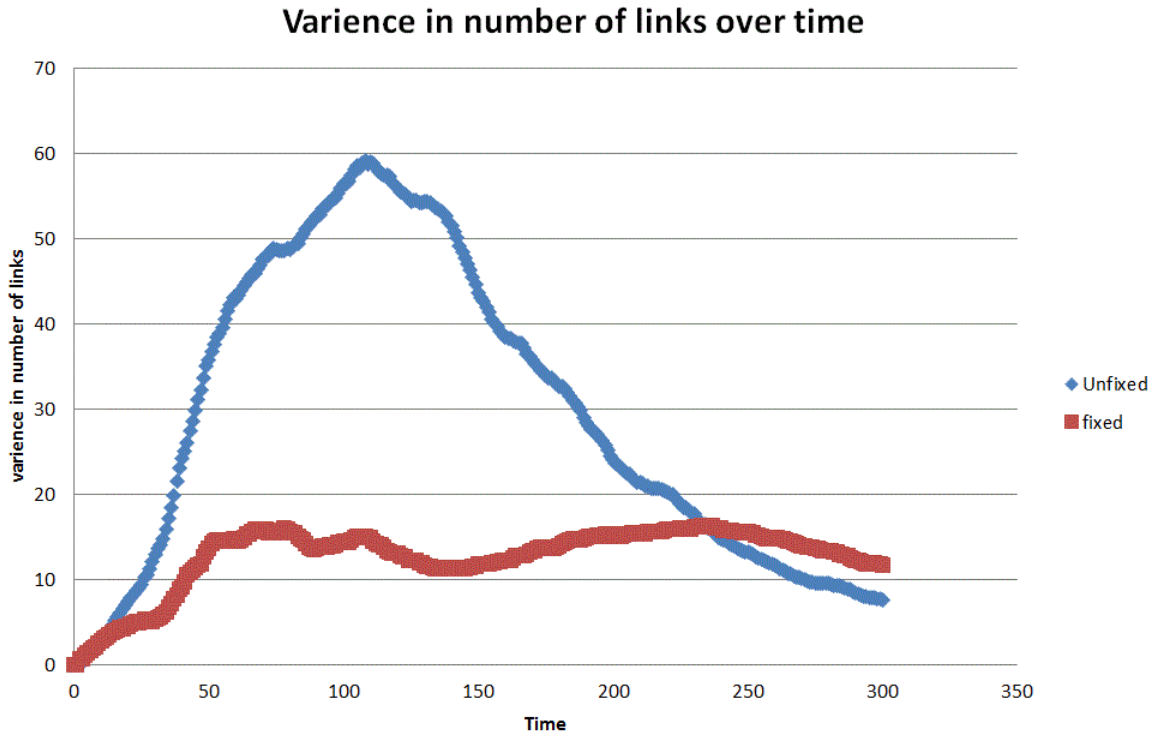


Figure: The difference in the number of links in the rings varies wildly when nodes don't join new rings. This is expected from the extremely fast creation of links. In the fixed network, joins keep the amount of links down to a steady number.

Conclusion

- We have a scheme to merge multiple rings.
- This was a necessary step for implementation.