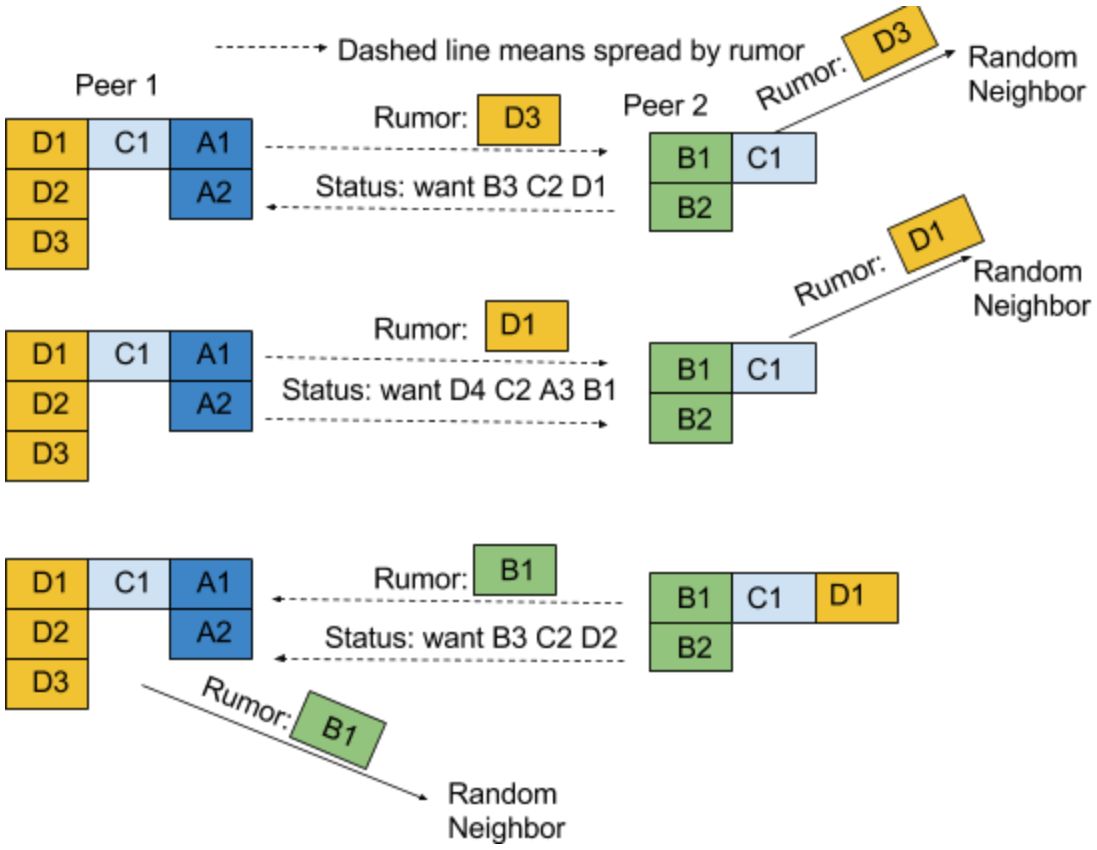
CS5450 Networks & Distributed Computing
Project 3 : Peer to Peer networking
Chumeng Xu, cx87@cornell.edu
Saran Kumar Krishnasamy, sk2998@cornell.edu

Gossip Protocol

In this lab we implemented a rumormongering protocol with simple anti-entropy mechanism. Messages are propagated through UDP connections, there is no guarantee of reliable communication.

Rumormongering

- When peer 2 gets a rumor, it sends a copy to a random neighbor
- It then sends out a status message, indicating desired sequence number for each message.
- Peer 1 then compares the status msg with its own vectors. SInce it has new messages peer 2 does not have, it starts rumormongering D1. It also discovers that peer 2 has something it does not have (B), so it sends out a status message requesting B1.
- Peer 2 starts rumormongering B1.
- Each message (ABCD) is kept in a separate vector to make sure each message is assembled in the right order.
- Rumormongering stops when everyone has the same set of messages.



Anti-entropy

To ensure all peers eventually gets all messages, each peer sends out a status message to a random neighbor every 10 seconds. If it sees a different set of messages, the neighbor starts rumormongering.

Tricky Parts of Implementation

Implementing rumormongering correctly was tricky. We ran into issues at the beginning of rumormongering, when each peer's status map has not been updated yet. We also ran into problems when testing our clients with other people's implementations (QMap vs QVarientMap messages), we solved the issue by changing the format of our messages.

Test Cases

We tested our implementation in the following two cases and it works correctly.

- Multiple clients (up to 4)
- Clients across different implementations