CPSC 5520—Distributed Systems

Token Ring

with Synchronizing Coordinator

Distributed System

Each node (student) needs to get a matching token (secret message). Every node has a token to start, but the wrong one. We will use a token ring algorithm to relocate the tokens to their correct nodes.

Topology

There is a single coordinator node (the professor) and *n* member nodes (the students). Each member node connects with a ***successor node***. Successors are in clockwise order within your table and then to the next table clockwise. In both cases, start with the chair/table closest to the professor's podium. In our case, $n = 31$ in section 1 or $n = 19$ in section 2.

Algorithm

The algorithm is "the dance" of the node's interactions.

Each node produces an arbitrary token when they get the signal from the coordinator (PREPARE message). Each node then activates the token. The following steps are repeated until every node has exited the ring:

1. Check if the token matches the node (i.e., is this "my" token?).
2. If so, then exit the ring.
3. If not, on next SYNC message broadcast from the coordinator pass the token to successor node (PASS message).

Protocol

*The **protocol** is the set of "dance steps" that describe the rules of each node's interactions.*

PREPARE Message

The ***payload*** is a random token (name card). The node "activates" the token like so:

1. Write a secret admirer message in pencil/pen on the back of the name card. The writer's identity will not be known by the token owner. A suitable message might be, "Best of luck in the course! I bet you'll get an A."
2. Stand up to indicate you have joined the ring. (Exit the ring when the time comes by sitting down.)

SYNC Message

Note your current successor node. Your previous successor may have exited the ring. Then send a PASS message to her with your current token. Make sure not to reveal the originator of your current token.

PASS Message

The payload is a token.
Receiver examines the token and if it matches (has her name on it), then she exits the ring (by sitting down).

Analysis

What is the efficiency class (where *n* is the number of nodes)? (Answer with big-oh notation.)

1. For number of connections?

2. For number of messages?

3. Elapsed time (you can use the number of SYNC messages as a surrogate for time)?

How "good" is this algorithm? We can evaluate on four criteria (see textbook p. 10 *cf*):

4. **Access to shared resources**. One way of evaluating this is to see if the total work is more distributed than without the system. In our case, the single-node solution is for the professor to hand out the tokens individually. Has this improved on that?

5. **Distribution transparency**. This can take forms (p. 12) of access transparency, location transparency, relocation transparency, migration transparency, replication transparency, concurrency transparency, and failure transparency. Let's x`just think about the last one, failure transparency. What would happen if a node crashed (i.e. the student fell asleep or left the room while holding someone else's token)?

6. **Scalability**. This follows from our efficiency classes above. But also consider network bottlenecks and risks as well as single points of failure. Explain if this is a scalable algorithm and what the risks are. Also, can we add new nodes dynamically?

7. **Openness**. Is our system and its components usable/adaptable for other systems.