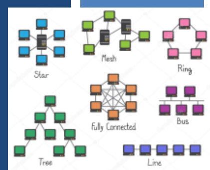
TCSS 558: APPLIED DISTRIBUTED COMPUTING

Chapter 6 – Coordination – IV
Final Exam Practice

Wes J. Lloyd
School of Engineering & Technology (SET)
University of Washington - Tacoma



1

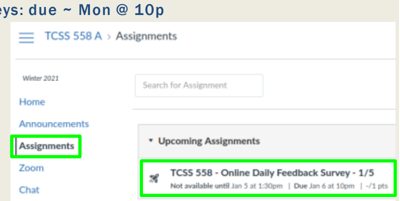
OBJECTIVES – 3/7

- Questions from 3/5
- Assignment 3: Replicated Key Value Store
- Class Activity 4 – Total Ordered Multicasting
- Class Activity 5 – Causality and Vector Clocks
- Chapter 6: Coordination
  - Chapter 6.3: Distributed Mutual Exclusion
  - Chapter 6.4: Election Algorithms

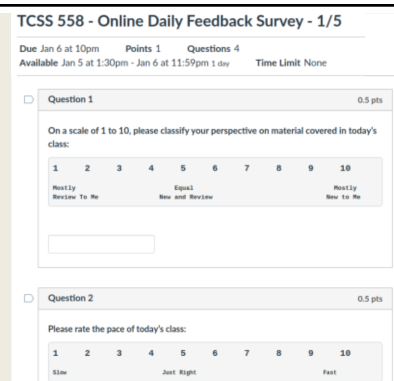
2

ONLINE DAILY FEEDBACK SURVEY

- Daily Feedback Quiz in Canvas – Available After Each Class
- Extra credit available for completing surveys ***ON TIME***
- Tuesday surveys: due by ~ Wed @ 10p
- Thursday surveys: due ~ Mon @ 10p



3



4

MATERIAL / PACE

- Please classify your perspective on material covered in today's class (22 respondents):
- 1-mostly review, 5-equal new/review, 10-mostly new
- **Average – 6.68** (↓ - *previous 6.87*)

- Please rate the pace of today's class:
- 1-slow, 5-just right, 10-fast
- **Average – 5.36** (↓ - *previous 5.83*)

5

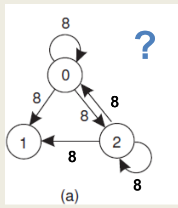
FEEDBACK FROM 3/5

- ***How does the distributed algorithm (slide 36) break a tie when two processes broadcast the same lamport clock value?***
- The distributed algorithm uses Lamport's logical clocks
- The distributed algorithm also requires a **total ordering of all events in the system**
  - This means we have to have a means to "order" all of the events so that it is unambiguous which actually happens first
- With total ordered multicasting, to break ties when logical clock values are the same for processes, we use the unique process IDs to break ties, and represent clock values using tuples instead of only logical clock values

6

BREAKING A LOGICAL CLOCK TIE WITH TUPLES

- Consider if p0 and p2 both want access to the shared resource
- But what if p0 and p2's logical clock values are both the same (=8)
- Using tuples, we express the clocks as:
  - <8,p0> and <8,p2>
- Ties are broken by ordering the lower process IDs event as first
  - p0 < p2
- The distributed algorithm example does not use tuples for logical clocks for simplicity
- But if there was a tie, this is how it can be broken



March 7, 2024 | TCSS558: Applied Distributed Computing [Winter 2024] School of Engineering and Technology, University of Washington - Tacoma | L18.7

7

FEEDBACK - 2

- ***Can't multiple nodes get a simple majority of the votes? If there are 7 nodes and 2 make requests for the resource, won't the other 5 all reply with "OK", meaning both nodes would have OK's from more than half the nodes?***
- Assume this question is for the distributed algorithm
- Distributed algorithm uses total ordered multicasting, and clock value ties are resolved by process ID
- When second node (p2) with higher logical clock (proc ID) requests access from the other node (p0) with lower logical clock (proc ID), the node with the higher logical clock (p2) will learn its clock is logically higher and yield to the node (p0) with the lower logical clock (proc ID).
  - p2 is blocked until p0 is done

March 7, 2024 | TCSS558: Applied Distributed Computing [Winter 2024] School of Engineering and Technology, University of Washington - Tacoma | L18.8

8

OBJECTIVES – 3/7

- Questions from 3/5
- **Assignment 3: Replicated Key Value Store**
- Class Activity 4 – Total Ordered Multicasting
- Class Activity 5 – Causality and Vector Clocks
- Chapter 6: Coordination
  - Chapter 6.3: Distributed Mutual Exclusion
  - Chapter 6.4: Election Algorithms
- Practice Final Exam Questions

March 7, 2024 | TCSS558: Applied Distributed Computing [Winter 2024] School of Engineering and Technology, University of Washington - Tacoma | L18.9

9

SHORT-HAND-CODES FOR MEMBERSHIP TRACKING APPROACHES

- Include readme.txt or doc file with instructions in submission
- Must document membership tracking method

>> please indicate which types to test <<

ID	Description
F	Static file membership tracking – file is not reread
FD	Static file membership tracking DYNAMIC - file is periodically reread to refresh membership list
T	TCP membership tracking – servers are configured to refer to central membership server
U	UDP membership tracking - automatically discovers nodes with no configuration

March 7, 2024 | TCSS558: Applied Distributed Computing [Winter 2024] School of Engineering and Technology, University of Washington - Tacoma | L18.10

10

ASSIGNMENT 3

- **Sunday March 10th**
- Goal: Replicated Key Value Store
- Team signup to be posted on Canvas under 'People'
- Build off of Assignment 2 GenericNode
- Focus on TCP client/server w/ replication
- How to track membership for data replication?
  - Can implement multiple types of membership tracking for extra credit
- REQUIREMENT: 'store' command needs to output 1 key-value pair per line using ASCII text (no binary)

March 7, 2024 | TCSS558: Applied Distributed Computing [Winter 2024] School of Engineering and Technology, University of Washington - Tacoma | L18.11

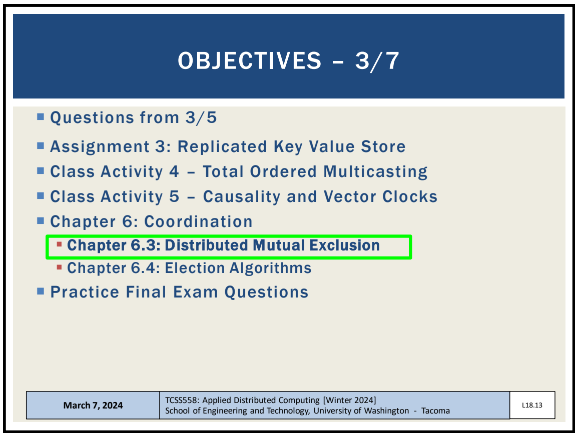
11

OBJECTIVES – 3/7

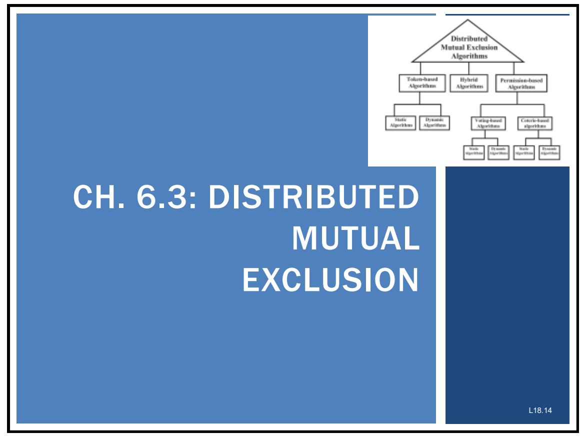
- Questions from 3/5
- Assignment 3: Replicated Key Value Store
- **Class Activity 4 – Total Ordered Multicasting**
- **Class Activity 5 – Causality and Vector Clocks**
- Chapter 6: Coordination
  - Chapter 6.3: Distributed Mutual Exclusion
  - Chapter 6.4: Election Algorithms
- Practice Final Exam Questions

March 7, 2024 | TCSS558: Applied Distributed Computing [Winter 2024] School of Engineering and Technology, University of Washington - Tacoma | L18.12

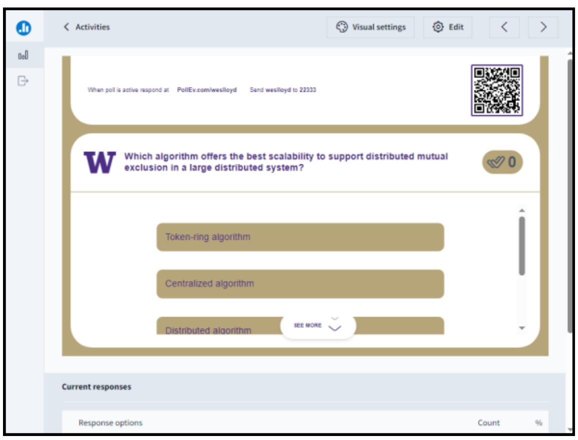
12



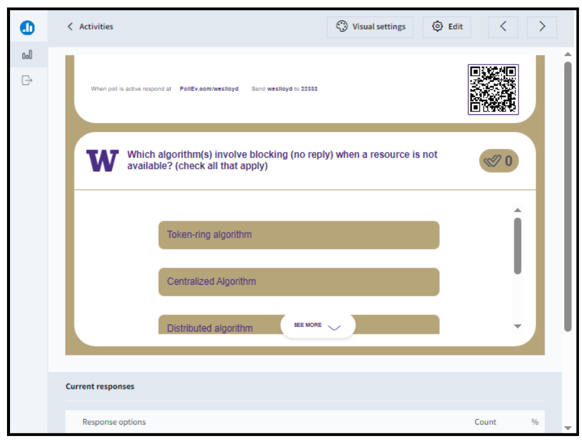
13



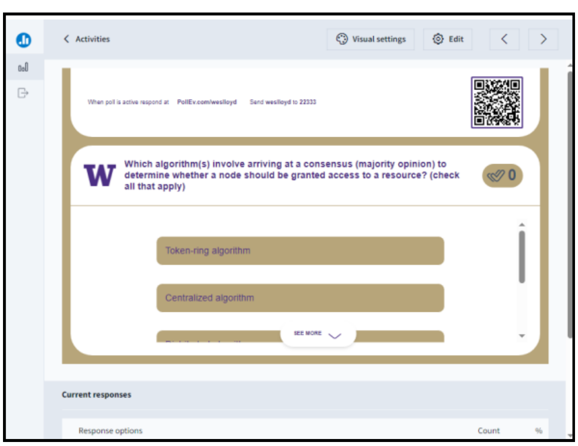
14



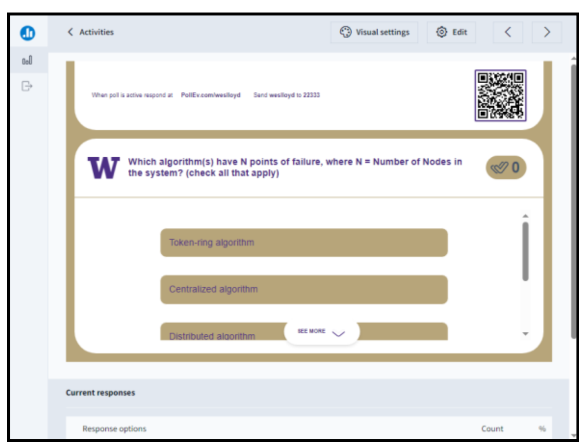
15



16



17



18

DISTRIBUTED MUTUAL EXCLUSION ALGORITHMS REVIEW

- Which algorithm offers the best scalability to support distributed mutual exclusion in a large distributed system?
- (A) Token-ring algorithm
- (B) Centralized algorithm
- (C) Distributed algorithm
- (D) Decentralized voting algorithm

19

DISTRIBUTED MUTUAL EXCLUSION ALGORITHMS REVIEW - 2

- Which algorithm(s) involve blocking (no reply) when a resource is not available?
- (A) Token-ring algorithm
- (B) Centralized algorithm
- (C) Distributed algorithm
- (D) Decentralized voting algorithm

20

DISTRIBUTED MUTUAL EXCLUSION ALGORITHMS REVIEW - 3

- Which algorithm(s) involve arriving at a consensus (majority opinion) to determine whether a node should be granted access to a resource?
- (A) Token-ring algorithm
- (B) Centralized algorithm
- (C) Distributed algorithm
- (D) Decentralized voting algorithm

21

DISTRIBUTED MUTUAL EXCLUSION ALGORITHMS REVIEW - 4

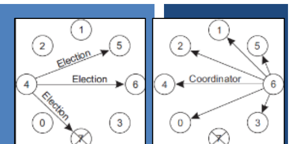
- Which algorithm(s) have N points of failure, where N = Number of Nodes in the system?
- (A) Token-ring algorithm
- (B) Centralized algorithm
- (C) Distributed algorithm
- (D) Decentralized voting algorithm

22

OBJECTIVES – 3/7

- Questions from 3/5
- Assignment 3: Replicated Key Value Store
- Class Activity 4 – Total Ordered Multicasting
- Class Activity 5 – Causality and Vector Clocks
- Chapter 6: Coordination
  - Chapter 6.3: Distributed Mutual Exclusion
  - **Chapter 6.4: Election Algorithms**
- Practice Final Exam Questions

23



CH. 6.4: ELECTION ALGORITHMS

24

ELECTION ALGORITHMS

- Many distributed systems require one process to act as a coordinator, initiator, or provide some special role
- Generally any node (or process) can take on the role
  - In some situations there are special requirements
  - Resource requirements: compute power, network capacity
  - Data: access to certain data/information
- Assumption:
  - Every node has access to a "node directory"
  - Process/node ID, IP address, port, etc.
  - Node directory may not know "current" node availability
- Goal of election: at conclusion all nodes agree on a coordinator or "leader"

25

ELECTION ALGORITHMS - 2

- Consider a distributed system with N processes (*or nodes*)
- Every process has an identifier id(P)
- Election algorithms attempt to locate the highest numbered process to designate as coordinator

- **Algorithms:**
- Bully algorithm
- Ring algorithm
- Elections in wireless environments
- Elections in large-scale systems

26

BULLY ALGORITHM

- When ***any*** process notices the coordinator is no longer responding to requests, it initiates an election
- Process P_k initiates an election as follows:
  1. P_k sends an ELECTION message to all processes with higher process IDs (P_{k+1}, P_{k+2}, … P_{N-1})
  2. If no one responds, P_k wins the election and becomes coordinator
  3. If a "higher-up" process answers (P_{k+n}), it will take over and run the election. P_k will quit sending ELECTION messages.
- When the higher numbered process receives an ELECTION message from a lower-numbered colleague, it responds with "OK", indicating it's alive, and it takes over the election.

27

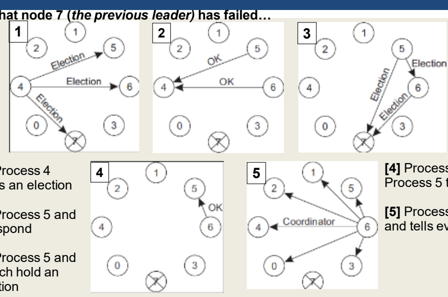
BULLY ALGORITHM - 2

- The higher numbered process then holds an election with ***only*** higher numbered processes (nodes).
- Eventually ***all*** processes give up except one, and the remaining process becomes the new coordinator.
- The coordinator announces victory by sending all processes a message stating it is starting as the coordinator.
- If a higher numbered node that was previously down comes back up, it holds an election, and ultimately takes over the coordinator role.
- The process with the *"biggest"* ID in town always wins.
- Hence the name, **bully algorithm**

28

BULLY ALGORITHM - 3

Note that node 7 (*the previous leader*) has failed…



[1] Process 4 starts an election

[2] Process 5 and 6 respond

[3] Process 5 and 6 each hold an election

[4] Process 6 tells Process 5 to stop

[5] Process 6 wins and tells everyone

29

BULLY ALGORITHM - 4

- **Requirement:** Every node knows who is participating in the distributed system
  - Each node has a group membership directory
- First process to notice the leader is offline launches a new election
- GOAL: Find the highest number node that is running
  - Loop over the nodes until the highest numbered node is found
  - May require multiple election rounds
- Highest numbered node is always the ***"BULLY"***

30

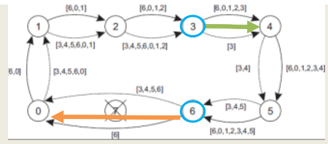
RING ALGORITHM

- Election algorithm based on a network of nodes in logical ring
- *Does not use a token*
- Any process (P_k) starts the election by noticing the coordinator is not functioning

1. P_k builds an **election message**, and sends to its successor in the ring
   - If successor is down, successor is skipped
   - Skips continue until a running process is found
2. When the **election message** is passed around, each node adds its ID to a *separate* **active node list**
3. When **election message** returns to P_k, P_k recognizes its own identifier in the **active node list**. Message is changed to COORDINATOR and "**elected(P_k)**" message is circulated.
   - Second message announces P_k is the NEW coordinator

31

RING: MULTIPLE ELECTION EXAMPLE



- **PROBLEM:** Two nodes start election at the same time: P_3 and P_6
- P_3 sends **ELECT(P_3)** message, P_6 sends **ELECT(P_6)** message
  - P_3 and P_6 both circulate ELECTION messages at the same time
- Also circulated with ELECT message is an **active node list**
- Each node adds itself to the **active node list**
- Each node votes for the highest numbered candidate
- P_6 wins the election because it's the candidate with the **highest ID**

32

RING ALGORITHM - DIFFERENCES

- Assumes nodes are organized in a ring, where each node has a known successor node
- Any node in the ring, *not necessarily the one with the highest ID*, can become the leader
- The membership list (**active node list**) is generated when circulating the ELECT message around the ring
  - Nodes do not have to maintain the membership list
  - ELECT message is simply circulated to the next node in the ring
- When multiple nodes conduct an election at the same time, the node with the higher ID wins

33

ELECTIONS WITH WIRELESS NETWORKS

- Assumptions made by traditional election algorithms not realistic for wireless environments:
  - >>> Message passing is reliable
  - >>> Topology of the network does not change
- A few protocols have been developed for elections in ad hoc wireless networks
- Vasudevan et al. [2004] solution handles failing nodes and partitioning networks.
  - Best leader can be elected, rather than just a random one

34

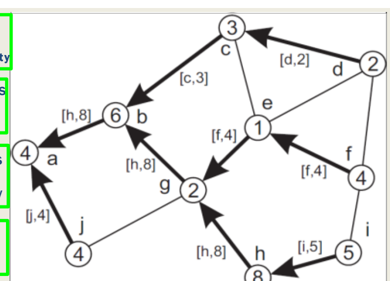
VASUDEVAN ET AL. WIRELESS ELECTION

1. Any node (***source***) (P) starts the **election** by sending an ELECTION message to immediate neighbors (any nodes in range)
2. Receiving node (Q) designates sender (P) as parent
3. (Q) Spreads election message to neighbors, ***but not to parent***
4. Node (R), receives message, designates (Q) as parent, and spreads ELECTION message to neighbors, ***but not to parent***
5. Neighbors that have already selected a parent immediately respond to R.
   - If ***all*** neighbors already have a parent, R is a leaf-node and will report back to Q quickly.
   - When reporting back to Q, R includes metadata regarding battery life and resource capacity
6. Q eventually acknowledges the ELECTION message sent by P, and also indicates the most eligible node (based on battery & resource capacity)

35

WIRELESS ELECTION - 2
SOURCE NODE: [A]

- Node [A] initiates election: **find the highest capacity**
- Election messages propagated to all nodes
- Each node reports to its parent node with best capacity
- Node A then facilitates Node H becoming leader



36

WIRELESS ELECTION - 3

- When multiple elections are initiated, nodes only join one
- Source node tags its ELECTION message with unique identifier, to uniquely identify the election.
- With minor adjustments protocol can operate when the network partitions, and when nodes join and leave

37

ELECTIONS FOR LARGE-SCALE SYSTEMS

- Large systems often require several nodes to serve as coordinators/leaders
- These nodes are considered ***"super peers"***
- ***Super peers*** must meet operational requirements:

1. Network latency from normal nodes to ***super peers*** must be low
2. ***Super peers*** should be evenly distributed across the overlay network (ensures proper load balancing, availability)
3. Must maintain set ratio of ***super peers*** to normal nodes
4. ***Super peers*** must not serve **too many** *normal nodes*

38

ELECTIONS FOR DHT BASED SYSTEMS

- DHT-based systems use a bit-string to identify nodes
- **Basic Idea**: Reserve fraction of ID space for super peers
- Reserve first k = $\log_2(N)$ bits for super-peer IDs
- Assume m=8 bit ID to identify nodes, with N=256 possible nodes
- m=number of bits to identify every node (m=8)
- Reserve left-most k-bits of ID to identify super peers (k=3)
- **Example:** For a system with m=8 bit identifier (256 nodes), and k=3 keys per node
- Required number of super peers is $2^{(k-m)} * N$, where N is the number of nodes, with N=256:
  - **8 total super peers required for 256 nodes**
  - ID (8-bits): 000|00000
    - left most bits identify super peers
    - right most bits identify local nodes

39

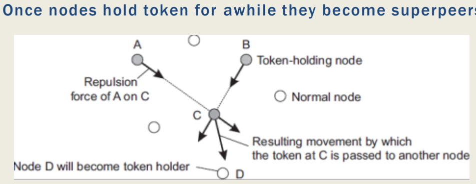
SUPER PEERS IN AN M-DIMENSIONAL SPACE

- Given an overlay network, the idea is to position superpeers throughout the network so they are evenly disbursed
- **Use tokens:**
- Give N tokens to N randomly chosen nodes
- No node can hold more than (1) token
- Tokens are "repelling force". Other tokens move away
- All tokens exert the same repelling force
- This automates token distribution across an overlay network

40

OVERLAY TOKEN DISTRIBUTION

- Gossping protocol is used to disseminate token location and force information across the network
- If forces acting on a node with a token exceed a **threshold**, token is moved away (sent to nodes farther away)
- Once nodes hold token for awhile they become superpeers



41

WE WILL RETURN AT 2:40 PM

42

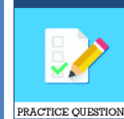
OBJECTIVES – 3/7

- Questions from 3/5
- Assignment 3: Replicated Key Value Store
- Class Activity 4 – Total Ordered Multicasting
- Class Activity 5 – Causality and Vector Clocks
- Chapter 6: Coordination
  - Chapter 6.3: Distributed Mutual Exclusion
  - Chapter 6.4: Election Algorithms
- **Practice Final Exam Questions**

March 7, 2024 | TCSS558: Applied Distributed Computing [Winter 2024] School of Engineering and Technology, University of Washington - Tacoma | L18.43

43

PRACTICE QUESTIONS



44

QUESTION 1: MULTI-TIERED ARCHITECTURE

- Consider a multi-tiered software architecture consisting of three distinct components: a web application server, a relational database server, and a log server. Describe the differences between a vertical distribution and a horizontal distribution of these components (Lecture 6)?
  - *Address the implications of these distributions for* ***scalability***
  - *Web application server:*
  - *Relational database server:*
  - *Log server:*

March 7, 2024 | TCSS558: Applied Distributed Computing [Winter 2024] School of Engineering and Technology, University of Washington - Tacoma | L18.45

45

QUESTION 2: CENTRALIZED SERVER ARCHITECTURE

- Consider a traditional centralized server architecture where many client nodes communicate with a single server node.
- Consider the four design goals of distributed systems from Ch. 1: Resource sharing, Distribution Transparency, Openness (*interoperability, portability, extensibility*), and Scalability.
- Describe challenges with ensuring these design goals when adopting a centralized server architecture.
- >> Consider citing an example if helpful.

Resource sharing:

Distribution transparency:

Openness:

Scalability:

March 7, 2024 | TCSS558: Applied Distributed Computing [Winter 2024] School of Engineering and Technology, University of Washington - Tacoma | L18.46

46

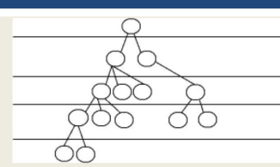
QUESTION 3: ARCHITECTURE DIFFERENCES

- Describe two communication differences between a traditional connection-oriented client/server architecture, and a publish/subscribe architecture where clients and servers communicate by exchanging tuples in a shared data space.

March 7, 2024 | TCSS558: Applied Distributed Computing [Winter 2024] School of Engineering and Technology, University of Washington - Tacoma | L18.47

47

QUESTION 4: UNSTRUCTURED PEER-TO-PEER NETWORK



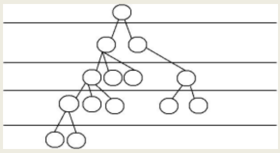
- Fourteen nodes communicate using an unstructured peer-to-peer network using random walks. The head node pictured at the top of the graph for this network receives a client request to retrieve a data element. The data element is only stored at one node. The precise node is unknown. Starting at the head node using message flooding without a specified time-to-live (TTL), how many messages are sent to locate the data item?

assume that messages are not resent to originating nodes

March 7, 2024 | TCSS558: Applied Distributed Computing [Winter 2024] School of Engineering and Technology, University of Washington - Tacoma | L18.48

48

QUESTION 4 (2): UNSTRUCTURED PEER-TO-PEER NETWORK

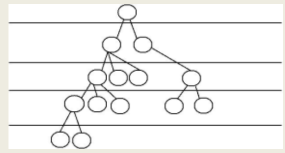


2-parts

- Using a random walk beginning at the head node at the top of the graph where only one walk per level is performed without a specified time-to-live (TTL), how many nodes will be visited? ***assume that messages are not resent to originating nodes***
- Given this number of node visitations, and considering that the data element is not replicated in the network as it exists at only one node, what is the probability (in %) that the data element will be found? **Specify a min to max range.**

49

QUESTION 4 (3): UNSTRUCTURED PEER-TO-PEER NETWORK

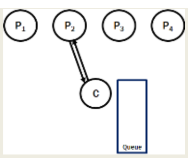


2-parts

- If we perform two parallel walks per level without a TTL, what is the worst-case probability (%) of finding the data element? ***at each node, two paths are followed ---***
- For this scenario, what is the best-case probability (%) of finding the data element? ***at each node, two paths are followed ---***

50

QUESTION 5: DISTRIBUTED MUTUAL EXCLUSION



- List one advantage, and one disadvantage for centralized distributed mutual exclusion:
- Advantage: Disadvantage:

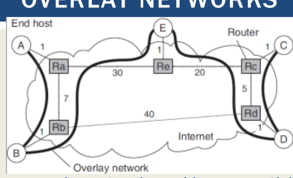
51

QUESTION 6: TIME MANIA

- Approaches to synchronizing time across all of the nodes of a distributed system focus on ensuring either one or both of the following: **accuracy** and/or **precision**
- For each time tracking approach below, identify whether it provides accuracy, precision, or both for coordinating time across the nodes in a distributed system.
- NTP:
- Berkeley:

52

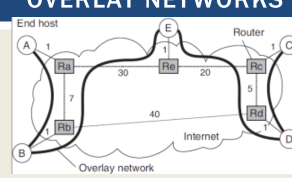
QUESTION 7: OVERLAY NETWORKS



- In the figure, an overlay network provides connectivity among the nodes: A, B, C, D, and E.
- The overlay network is implemented using "underlying" networks. In this case, the underlying network consists of a series of routers: Ra, Rb, Rc, Rd, and Re. Network "Weights" are assigned to each of the links between the routers indicating approximate communication delay. For example, the communication delay between Ra and Rb is 7 units, whereas the communication delay between node A and Ra is just 1 unit.

53

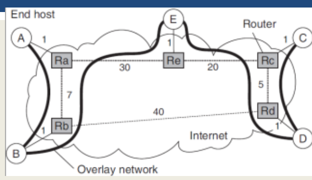
QUESTION 7: OVERLAY NETWORKS



- When nodes communicate using the overlay network, they must route messages via (by way of) the "overlay" links. In the diagram above, there are overlay links between: A → B, B → E, E → D, and D → C.

54

QUESTION 7: OVERLAY NETWORKS

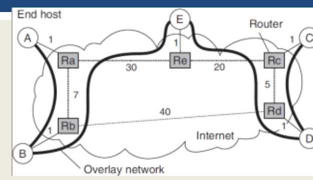


2-parts

- (A) What is the network delay when routing a message using the overlay network from node D to B? _______________units
- (B) What is the network delay when sending this same message from node D to B via the most efficient path using the underlying network? _______________ units

55

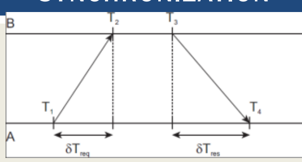
QUESTION 7: OVERLAY NETWORKS



- (C) Network "Stretch" is the ratio of the overlay network delay to the underlying network delay. For this example, what is the network stretch? _______________units

56

QUESTION 8: SYNCHRONIZATION



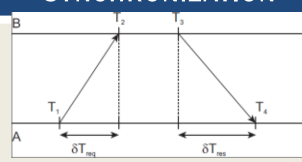
- In the Network Time Protocol, node A is a client that communicates with node B, which is an NTP server. The communication propagation delay is estimated with the formula:

$$\theta = \frac{|T_2 - T_1| + |T_3 - T_4|}{2}$$

- (a) What key assumption is made about the propagation delay between A and B?

57

QUESTION 8: SYNCHRONIZATION



- (b) When NTP is used to synchronize clocks of client computers, when client clocks are ahead of the NTP server due to clock skew, why do clients never set their local clock(s) backwards to match the time of the NTP server?

58

QUESTIONS

59