**TECHNOLOGICAL INSTITUTE OF THE PHILIPPINES**
**College of Computer Studies**

**CS 007 – PARALLEL AND DISTRIBUTED COMPUTING**
**FINAL PERIOD**

| | |
|---|---|
| **Names:**<br>**Cadeliña, Jaspher F.** | **Date: July 07, 2025** |
| **Program/Section: BSCS/CS33S4** | **Instructor: Ms.  Elsa V.  Isip - Barcelos** |
| **Activity: _Discussion 4.1 Distributed System** | |

## Leader Election Problem

- **Definition:**
  From the word itself, it is all about selecting a single coordinator (the “leader”) among a processes group so that it can manage tasks or make decisions on behalf of all.

- **Possible Algorithms:**
  - **Bully Algorithm**
  - **Ring Algorithm (Chang–Roberts)**

- **Field & Example:**
  The first thing I thought in my mind is the master-slave database cluster, where the leader handles all write operations; if it fails, a new leader is chosen to keep the system consistent.

  Researching the leader election taught me how subtle timing and message delays can complicate agreement—choosing the highest-ID process seems simple, but ensuring everyone hears about the outcome reliably requires careful design and failure handling.

**Consensus Problem**

- **Definition:**
  Ensuring all non-faulty processes in a distributed system agree on the same value, even when some processes fail or messages are lost.

- **Possible Algorithms:**

  - **Paxos**
  - **Raft**

- **Field & Example:**
  In distributed key-value stores (e.g., etcd), consensus (via Raft) guarantees that configuration changes apply atomically across nodes.
  According to the articles I have read, most of them stated that working through Paxos and Raft shows the trade-offs between safety and liveness: Well as a cs student we all know that a few simple rules about message exchanges and quorums can achieve reliable agreement.

---

**Distributed Search**

- **Definition:**
  Locating data or services across a network of peers without a centralized index, by routing queries through multiple nodes.

- **Possible Algorithms:**

  - **Flooding/Breadth-First Search**
  - **Distributed Hash Tables (e.g., Chord, Kademlia)**

- **Field & Example:**
  Peer-to-peer file sharing (BitTorrent DHT) uses Kademlia to find which peers hold a given torrent ID.

I am a user of BitTorrent but this is the first time I have learned that it is using Parallel Algorithm rather than Sequential ones. For me, xploring DHTs highlighted how hashing and clever routing tables let us scale lookups to millions of nodes in logarithmic time.

**Spanning Tree Generation**

- **Definition:**
  Building a loop-free subset of links that hooks every node into one connected network without any cycles. I personally want to think this as pruning extra branches so the packets don't wander in endless circles.

- **Possible Algorithms:**
  - **Gallager–Humblet–Spira (GHS) for Minimum Spanning Tree**
  - **Breadth‑First Spanning Tree (BFS‑based)**

- **Field & Example:**
  Ethernet's Spanning Tree Protocol (IEEE 802.1D) uses these ideas under the hood: switches exchange BPDUs, shut down redundant ports, and keep a clean, loop-free topology to dodge broadcast storms. Reading a GHS simulator taught me how each node, armed only with neighbor info, picks its smallest outgoing edge and gradually merges fragments—local chats yielding a perfectly minimal global tree.

**Mutual Exclusion Problem**

- **Definition:**
  Guaranteeing that at most one process at a time enters its critical section so shared resources aren't accessed concurrently and conflicts are avoided.

- **Possible Algorithms:**
  - **Ricart–Agrawala**
  - **Token‑Ring**

- **Field & Example:**
  One example I found is the Distributed file systems like Google File System grant write access via leases—tokens that ensure only one writer holds permission at any moment.
  I may be still confused to be honest BUT I think Ricart–Agrawala showed me how timestamped requests plus reply counting replace a single lock object, yet still guarantee fairness and avoid deadlock in parallel computing algorithms.

**Resource Allocation Problem**

- **Definition:**
  Assigning limited resources (CPU, memory, I/O) to competing processes in a way that prevents deadlock while maximizing overall efficiency.

- **Possible Algorithms:**

  ○ **Distributed Banker's Algorithm**
  ○ **Wait‑for Graph Deadlock Detection**

- **Field & Example:**
  As someone who uses docker and Kubernetes, I immediately realized that those two might be the best example for here then researched this to AI. Then chatgpt says "Kubernetes schedules containers onto nodes based on declared resource requests and real-time usage, preventing overcommitment and ensuring reliability". For me, a distributed banker's algorithm helped me see how safe‑state checks can be decentralized, though at the cost of more communication overhead.

**Replication**

- **Definition:**
  Keeping multiple copies of data or services across nodes to improve availability, fault tolerance, and performance.

- **Possible Algorithms:**

  ○ **Primary‑Backup**

    - **Quorum-Based Replication**

- **Field & Example:**
  Cassandra offers tunable consistency: clients pick read/write quorum sizes to balance latency against correctness guarantees. Reading and leaning basic quorum settings taught me that stronger consistency invariably raises latency; finding the sweet spot always depends on specific application needs.

---

**References**

1. Coulouris, G., Dollimore, J., Kindberg, T., & Blair, G. (2011). *Distributed Systems: Concepts and Design* (5th ed.). Pearson.
2. https://ftp.utcluj.ro/pub/users/civan/CPD/3.RESURSE/6.Book_2012_Distributed%20systems%20_Couloris.pdf
3. Ongaro, D., & Ousterhout, J. (2014, June). *In search of an understandable consensus algorithm* [Paper presentation]. *2014 USENIX Annual Technical Conference*. https://web.stanford.edu/~ouster/cgi-bin/papers/raft-atc14.pdf
4. Tanenbaum, A. S., & Van Steen, M. (2007). *Distributed Systems: Principles and Paradigms* (2nd ed.). Prentice Hall.
5. Kurose, J. F., & Ross, K. W. (2016). *Computer Networking: A Top-Down Approach* (6th ed.). Pearson.
6. MIT OpenCourseWare. (2021, *3.9 years ago*). *Asynchronous Distributed Algorithms: Shortest-Paths Spanning Trees*. [YouTube]. https://www.youtube.com/watch?v=EQjwWn-WrdI
7. YouTube. (2024, *8 months ago*). *How do computers elect leaders? – Consensus and Leader Election*. https://www.youtube.com/watch?v=7qRVMKXo4M4
8.

“I affirm that I have not given or received any unauthorized help on this assignment, and that this work is my own.”