- Distributed coordination and synchronization:
  - Distributed mutex, distributed election, distributed consensus, distributed transaction, distributed locks
- Distributed management and resources
  - Centralized structure, decentralized structure, scheduling
- Distributed computation
  - MapReduce, Spark
- Distributed communication
  - RPC, publish and subscribe, message queue
- Distributed storage
  - CAP, distributed storage, distributed cache

CS 7172
Parallel and Distributed Computation

Distributed Mutex and Election

Kun Suo

Computer Science, Kennesaw State University

https://kevinsuo.github.io/

What is Distributed Mutex?

- Suppose you are making coffee at Starbucks, and someone takes away your cup, some other takes away the coffee machine

- Ideal: you want to keep using the machine and cup without interference

What is Distributed Mutex?

- Like the coffee machine, in distributed system, for the same shared resource, one program does not want to be disturbed by other programs while it is being used.

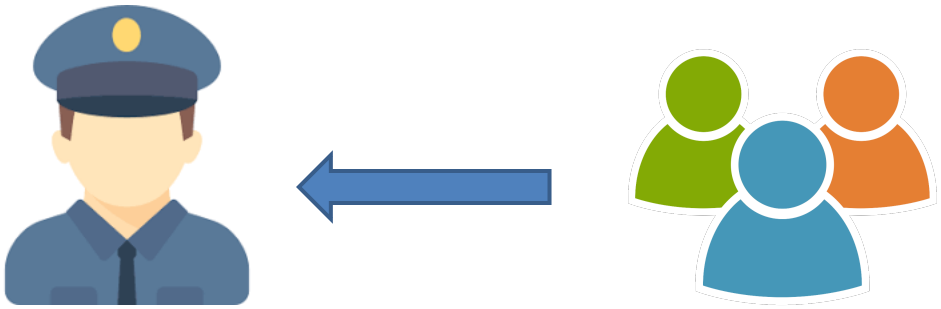
- This requires that only one program can access this resource at a time

What is Distributed Mutex?

- In a distributed system, the method to achieve access to exclusive resource is called ***Distributed Mutual Exclusion***

- The shared resource that is accessed by mutual exclusion is called ***Critical Resource***

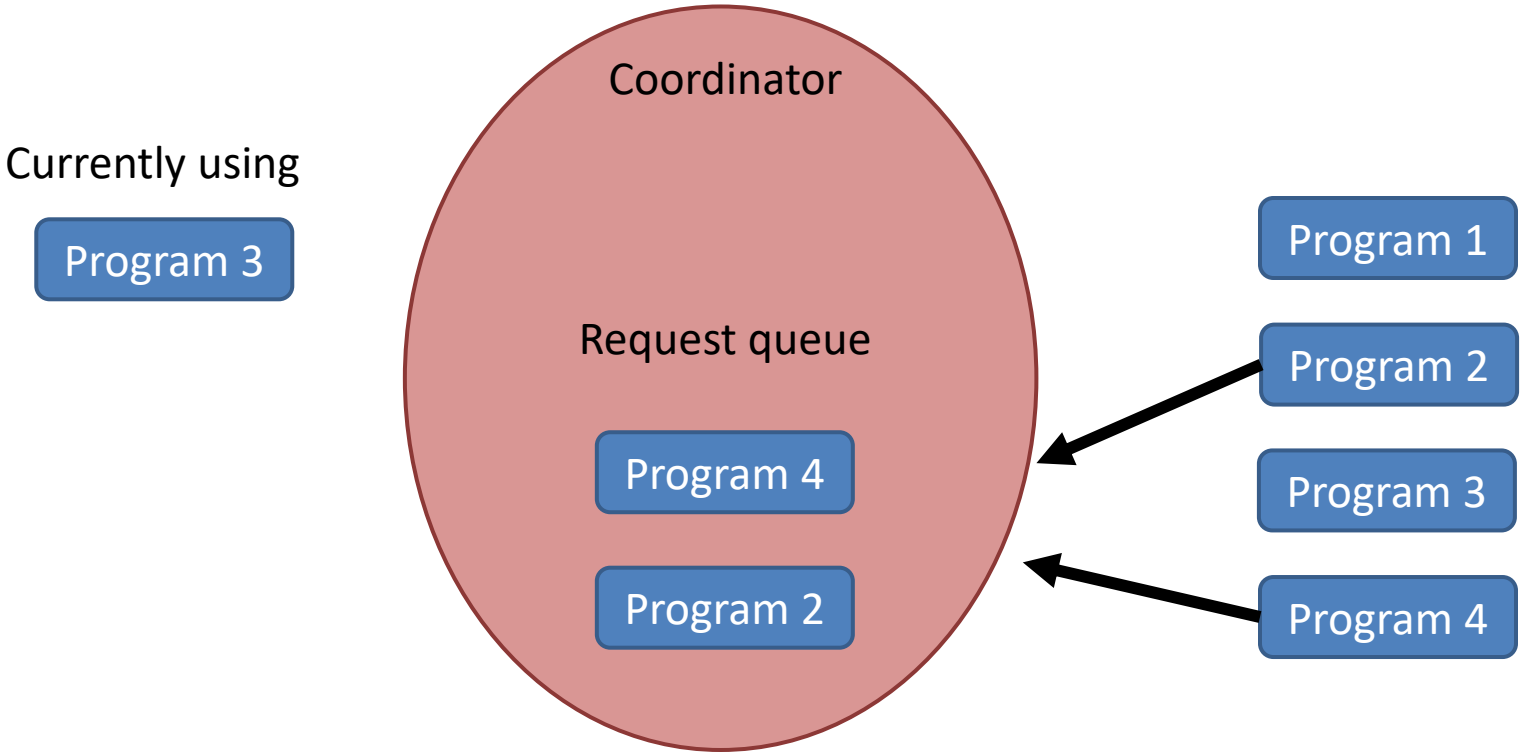
Method 1: Centralized algorithm



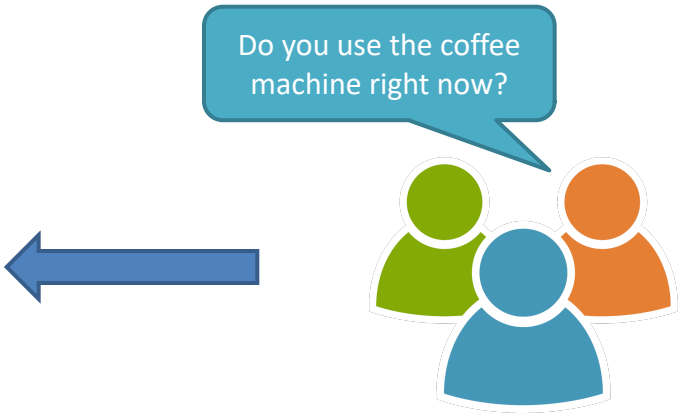
Add a "Coordinator" to restrict everyone to use self-service coffee machines in order to solve the problem of forcibly interrupting others

Method 1: Centralized algorithm

- Centralized algorithm is also named as Central Server algorithm in distributed system

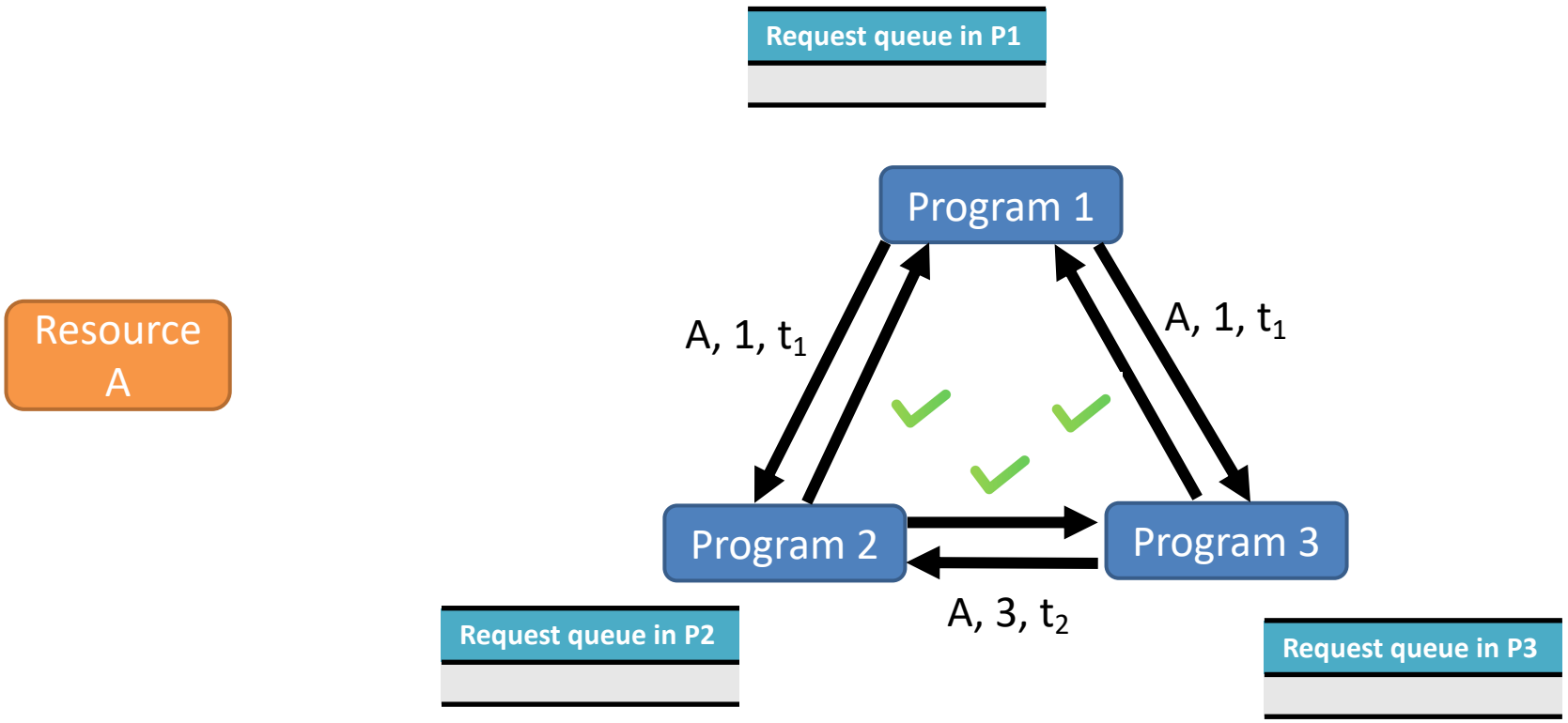


Method 2: Distributed algorithm

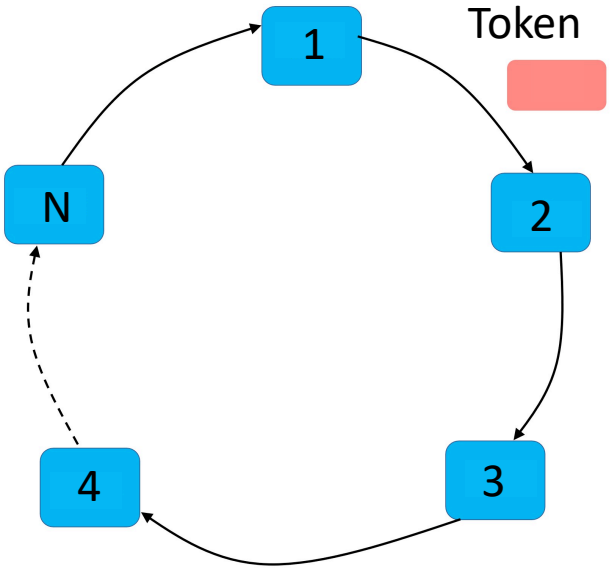


When you need to use a self-service coffee machine, you can ask other people first. When confirming that no other people are using, you can make your coffee.

Method 2: Distributed algorithm



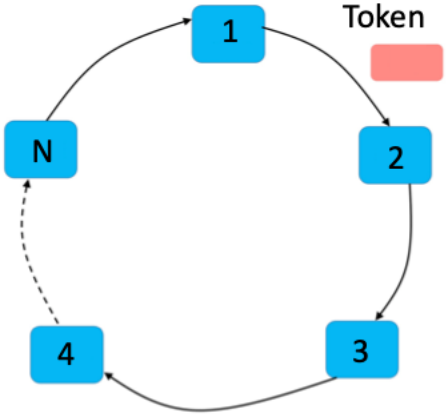
Method 3: Token Ring Algorithm



- How token ring algorithm works?
  - All programs form a ring structure. Tokens are passed between programs in a clockwise (or counterclockwise) direction.
  - The program that receives the token has the right to access critical resources. After the access is completed, the token is transferred to the next program.
  - If the program does not need to access critical resources, just passes the token to the next program

Scenario using Token Ring Algorithm

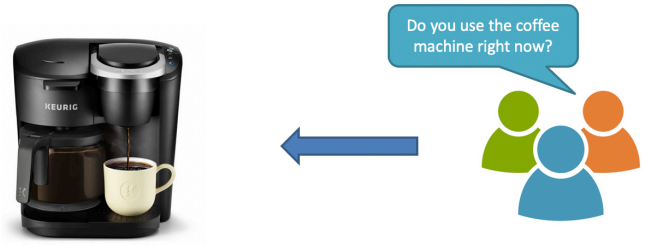
- Walkie-talkie:
  - Can send or receive messages
  - Every time only one walkie-talkie can send
  - The walkie-talkie that holds the token can send and others just receive



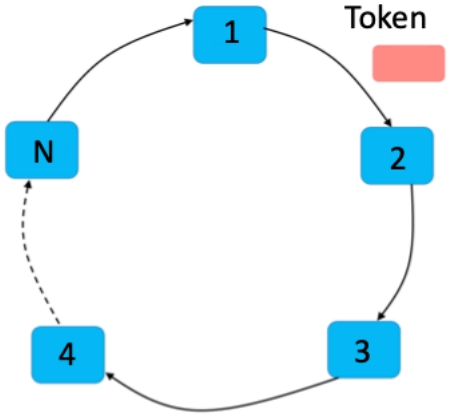
Distributed Mutex

- Centralized algorithm

- Distributed algorithm

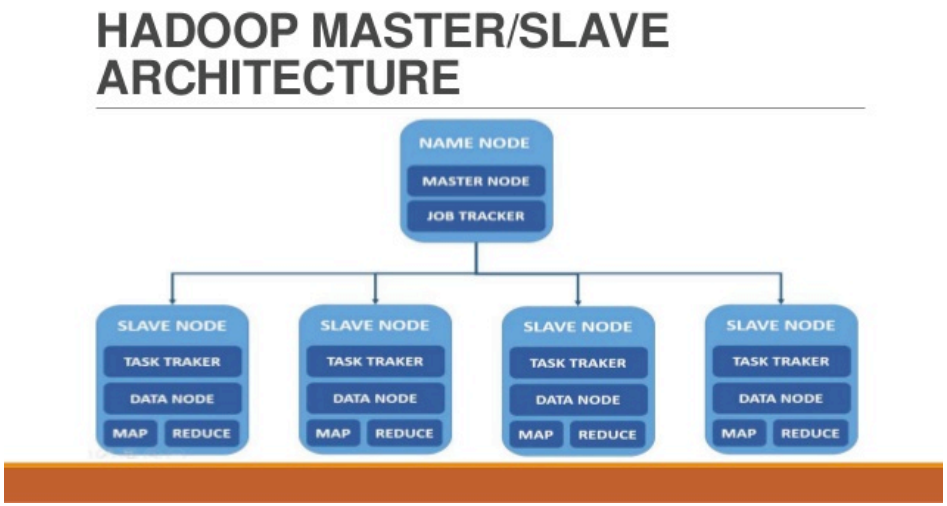


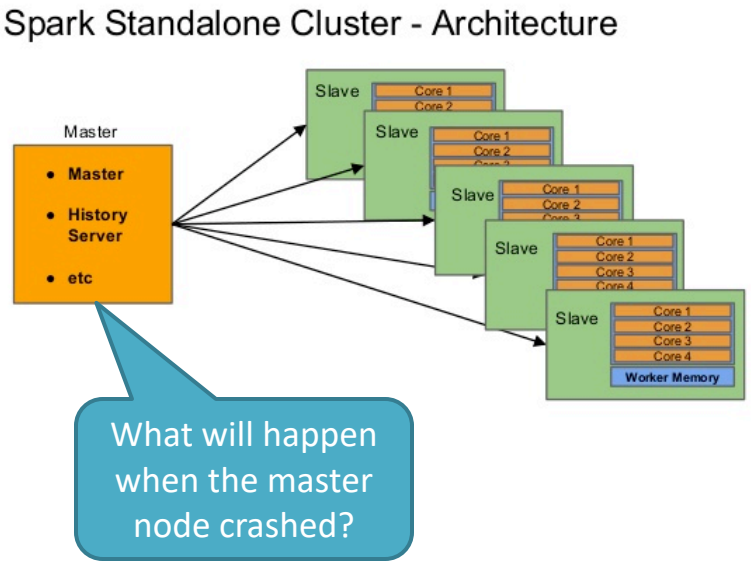
- Token Ring Algorithm



Why we need Distributed Election?

- Master node is so important in distributed system
  - Scheduling and managing other nodes





1. Bully algorithm

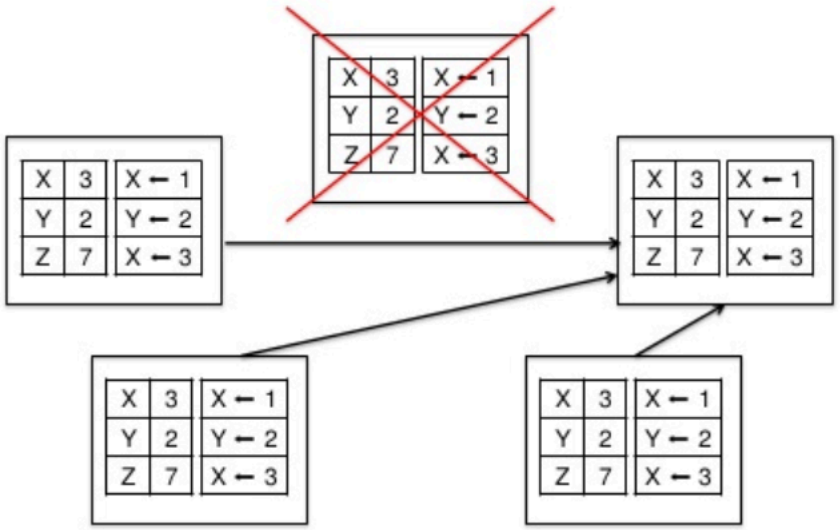
- Nodes have two types: normal nodes and master nodes.

- During initialization, all nodes are normal nodes, and have the right to become masters. However, after the election, only one node becomes the master node, and all other nodes are normal nodes. The node with the *highest ID number* from amongst the non-failed nodes is selected as the coordinator.

- The master will be reelected if and only if the master node fails or loses connect with other nodes.

Bully algorithm example in MongoDB

- How MongoDB deals with failure:
  - The node's last operation timestamp is used to represent the ID
  - The node with the latest timestamp has the largest ID, thus the live node with the latest timestamp is the master node



1. Bully algorithm

- Advantages:
  - Fast election speed
  - low algorithm complexity
  - simple and easy to implement (who lives and who has the largest ID is the master node)

1. Bully algorithm

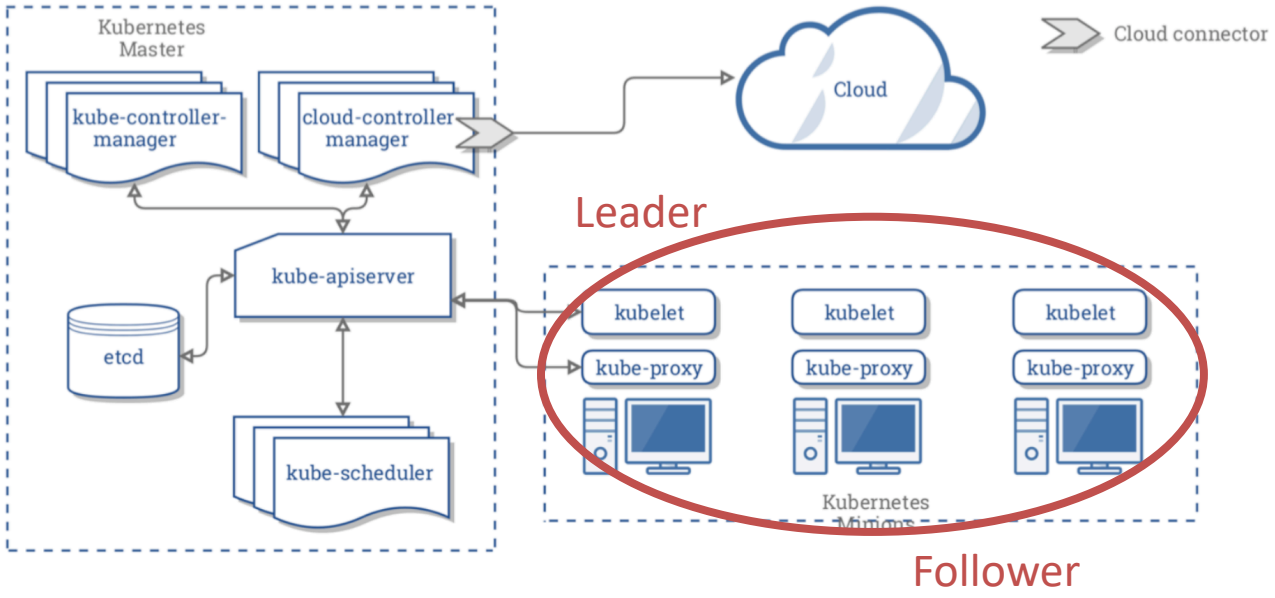
- Disadvantages:
  - Each node needs to have global node information (all node IDs), so additional information needs to be stored.
  - New election is required when any new node that is larger than the current master node ID
  - Frequent switch over could happen when some nodes frequently join and exit the cluster

2. Raft algorithm

- Similar as the democratic voting
- The core idea is "the minority obeys the majority"
- The node with the most votes becomes the master node

Raft algorithm example in Kubernetes

- How Kubernetes deals with data failure:
  - To ensure reliability, N nodes are usually deployed for data backup. One of the three nodes will be selected as the master, and the other nodes will be used as backups.



Kubernetes Architectural Overview (Retrieved from kubernetes.io)

2. Raft algorithm

- Advantages:
  - Fast election speed
  - low algorithm complexity
  - simple and easy to implement (who lives and who has the half votes is the master node)

2. Raft algorithm

- Disadvantages:
  - It requires that each node in the system can communicate with each other (vote), and requires the node which has more than half of the votes to be the master, thus the communication traffic is large

Comparison

	Bully	Raft
How to elect leader	Largest ID	Get half of the votes
How the algorithm works	When nodes find no responses from leader or leader fails, raise new election	Every node can be Candidate and selected as Leader. Every follower can only vote once in every election.
Election time	Short	Long
Performance	Bully < Raft	