- 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, Election and Consensus

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.

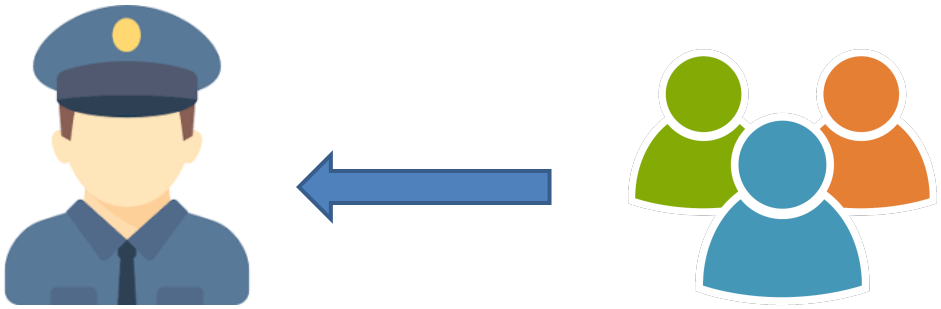
- This requires that only one program can access this resource at a time

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***

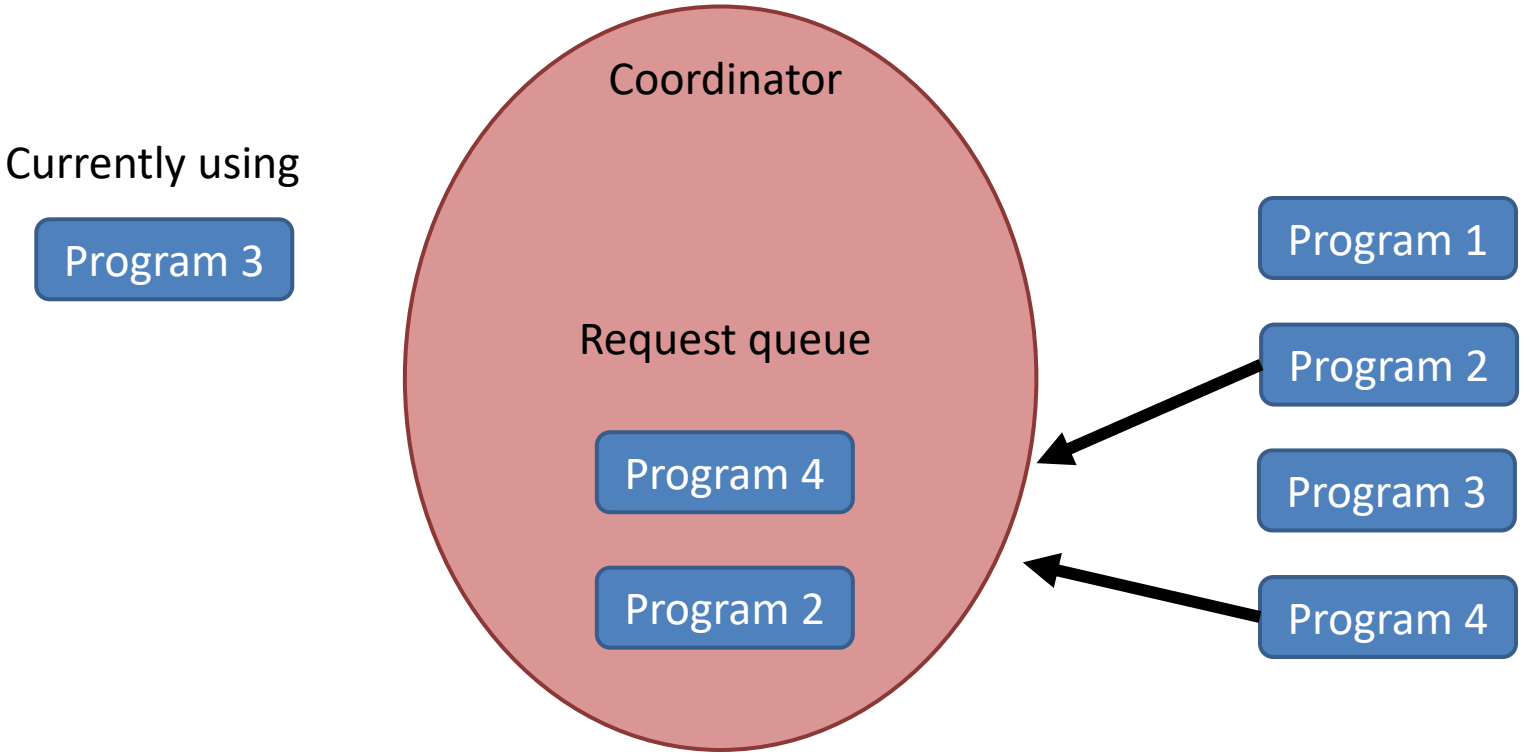
Method 1: Centralized algorithm



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



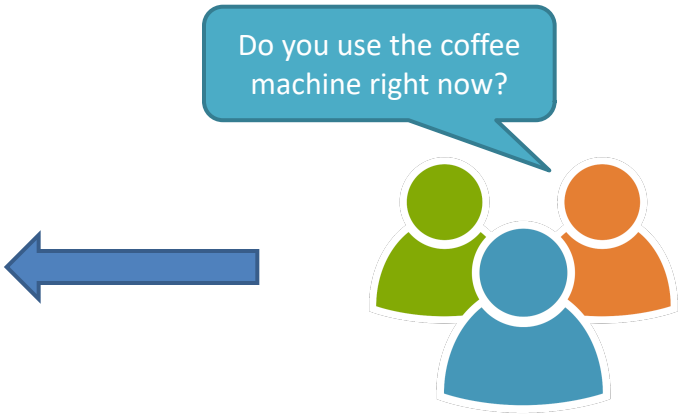
Method 1: Centralized algorithm

- Advantages:
  - Intuitive and simple
  - Less information interaction
  - Easy to implement
  - All programs need only communicate with the coordinator, no communication is required between programs

Method 1: Centralized algorithm

- Disadvantages:
  - The coordinator will become the performance bottleneck of the system
    - If there are 100 programs accessing critical resources, the coordinator has to process 100 * 3 = 300 messages. The number of messages processed by the coordinator increases linearly with the number of programs that need to access critical resources
  - It is easy to cause a single point failure. Poor reliability.
    - The failure of the coordinator will make all programs lose access to critical resources and the entire system unavailable.

Method 2: Distributed algorithm

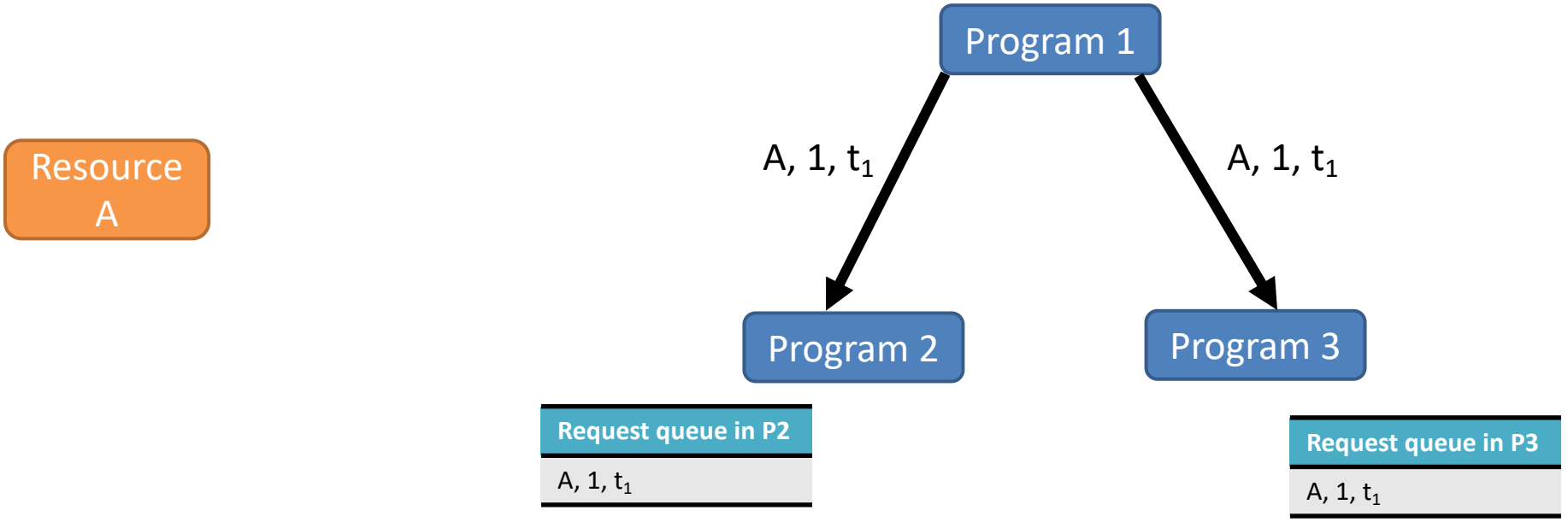


When you need to use a self-service coffee machine, you can ask other people first. When confirming that no other people are using, you can make your coffee.

Method 2: Distributed algorithm

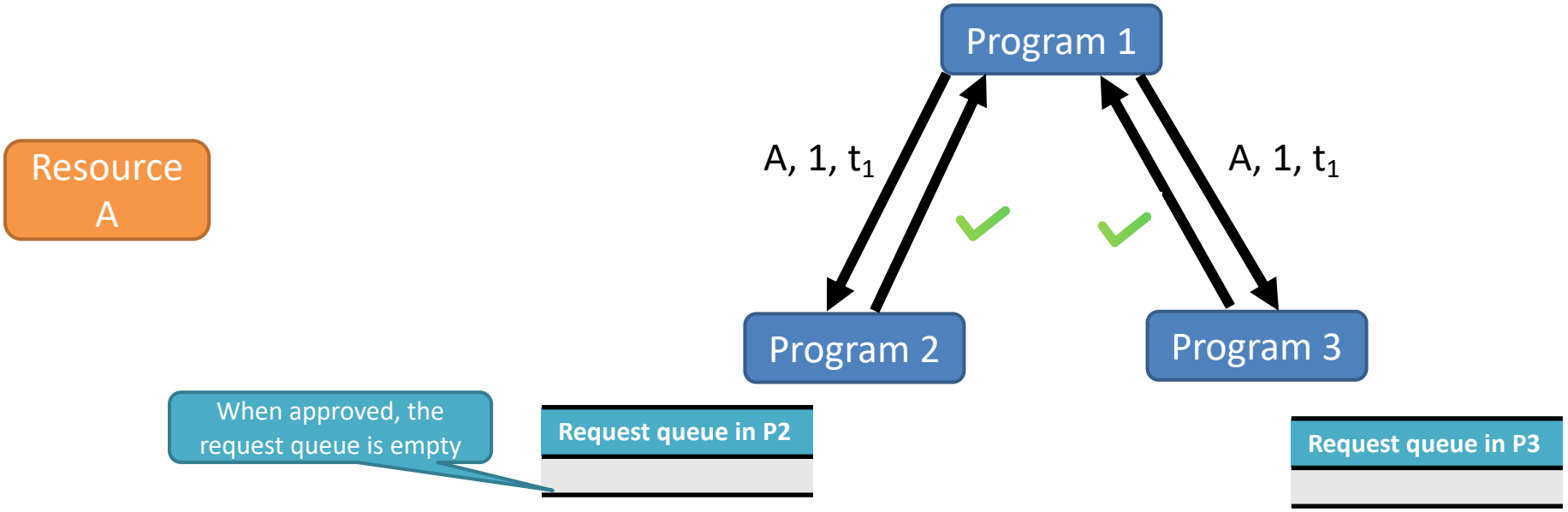
- How distributed algorithm works?
  - When a program wants to access a critical resource, it first sends a request message to other programs in the system.
  - After receiving the messages returned by all programs that no programs are using the resource, it can access the critical resource.
  - The request message includes ***the requested resource (which), the requester's ID (who), and the time the request was made (when).***

Method 2: Distributed algorithm

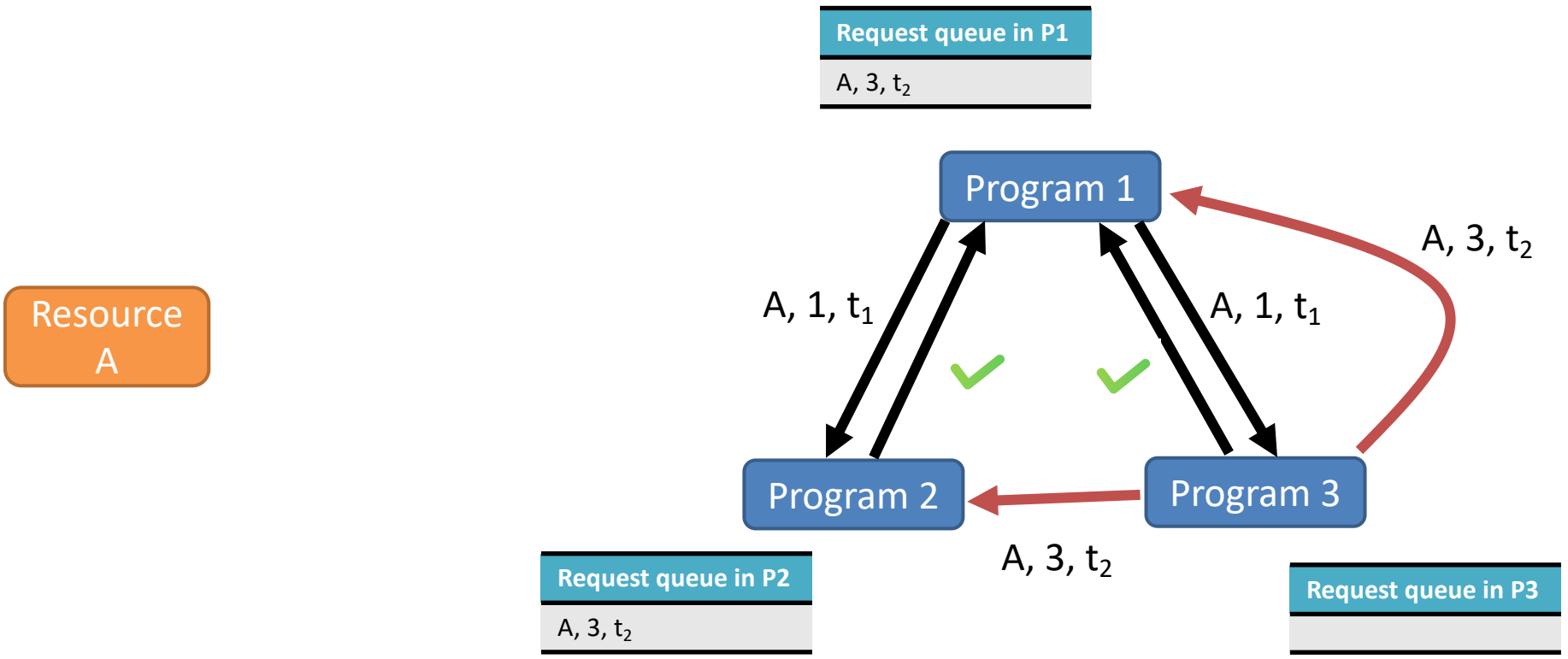


Which, Who and When

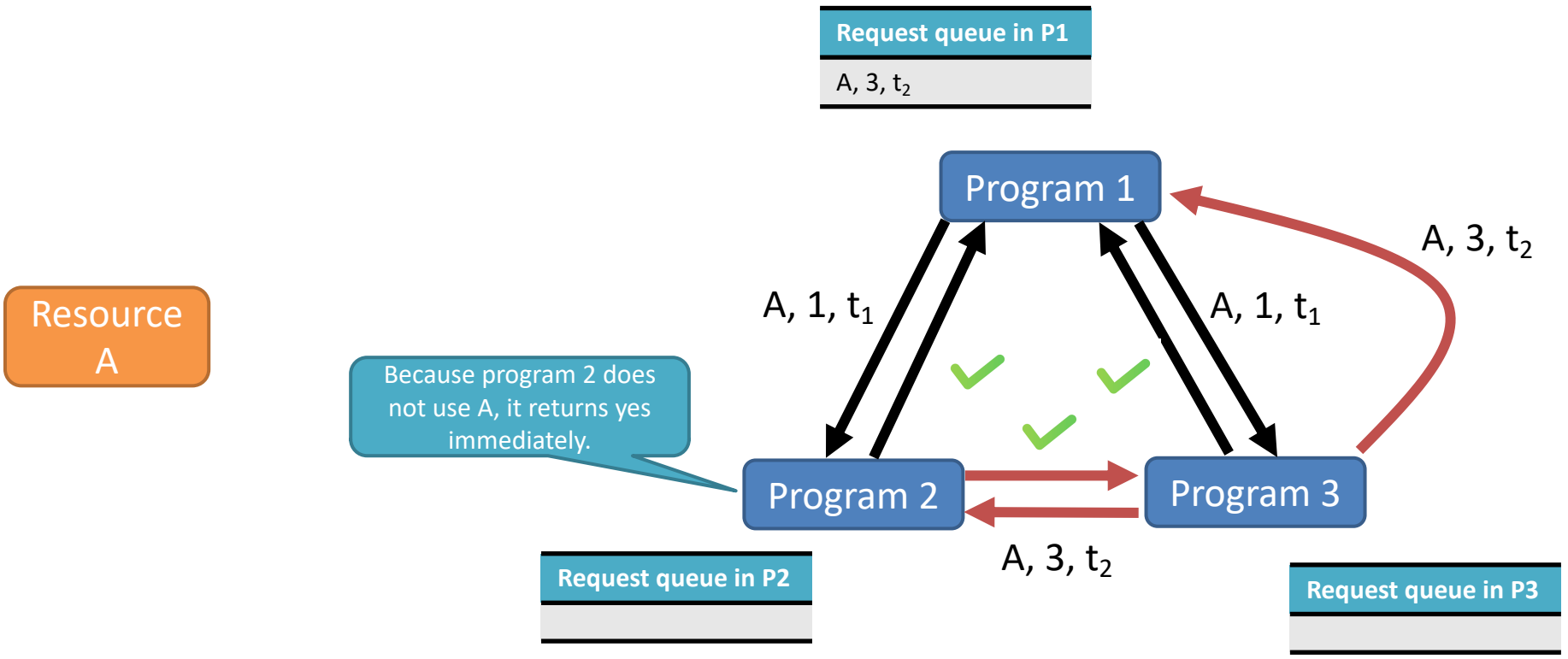
Method 2: Distributed algorithm



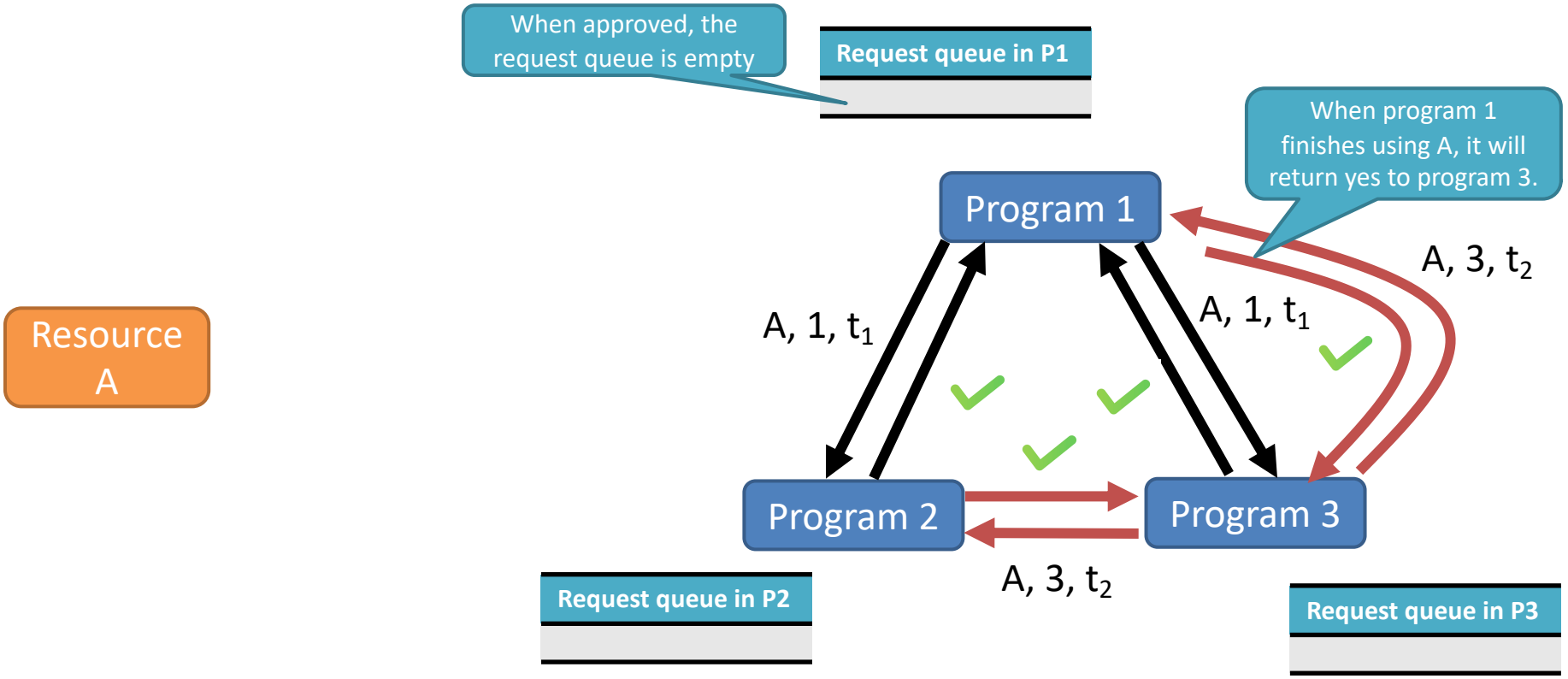
Method 2: Distributed algorithm



Method 2: Distributed algorithm



Method 2: Distributed algorithm



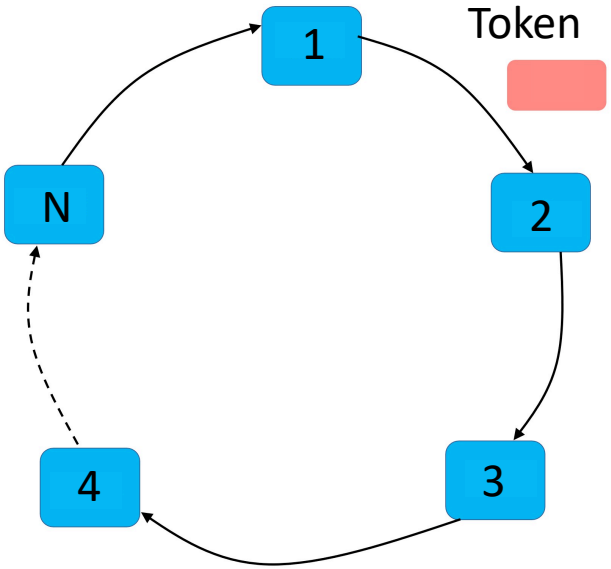
Method 2: Distributed algorithm

- Advantages:
  - Simple
  - Easy to implement

Method 2: Distributed algorithm

- Disadvantages:
  - The number of messages will increase **exponentially** with the number of programs that need to access critical resources, leading to high "communication costs"
    - n programs accessing to critical resources will produce 2n(n-1) messages
  - Once a program fails and the confirmation message cannot be sent, other programs are in a state of waiting for a reply, making the entire system unusable

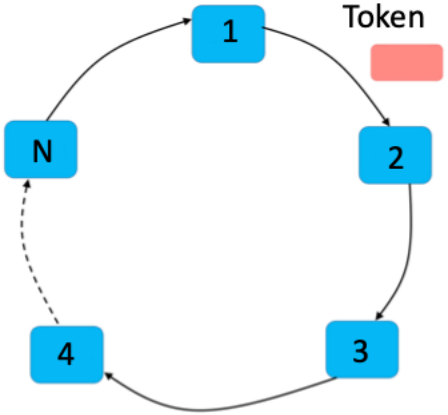
Method 3: Token Ring Algorithm



- 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

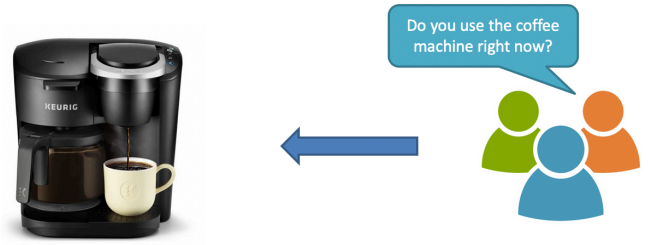
- Walkie-talkie:
  - Can send or receive messages
  - Every time only one walkie-talkie can send
  - The walkie-talkie that holds the token can send and others just receive



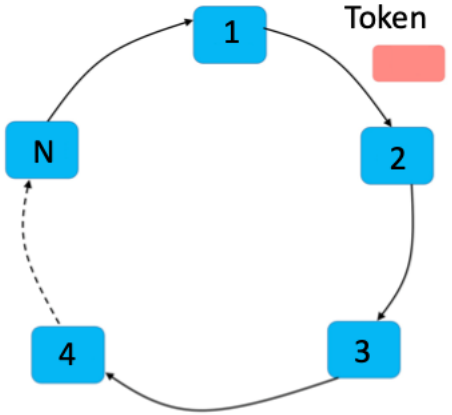
Distributed Mutex

- Centralized algorithm

- Distributed algorithm

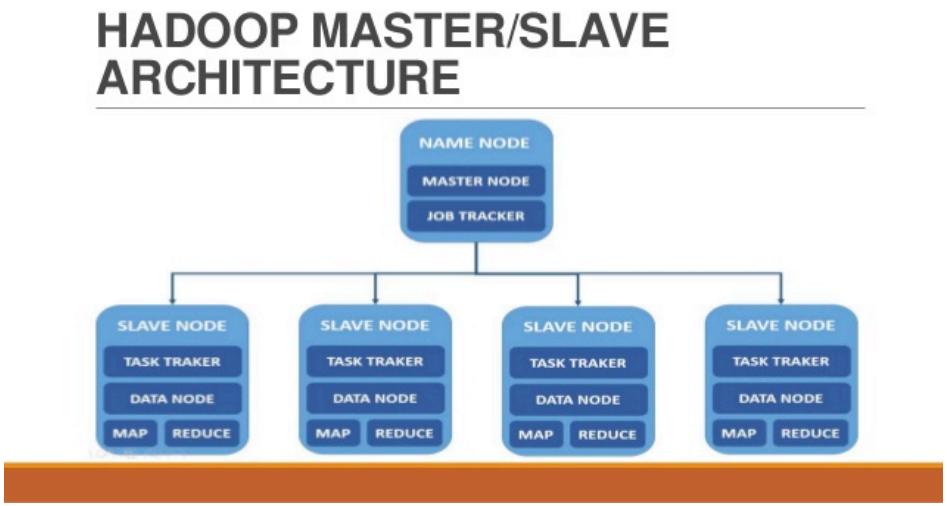


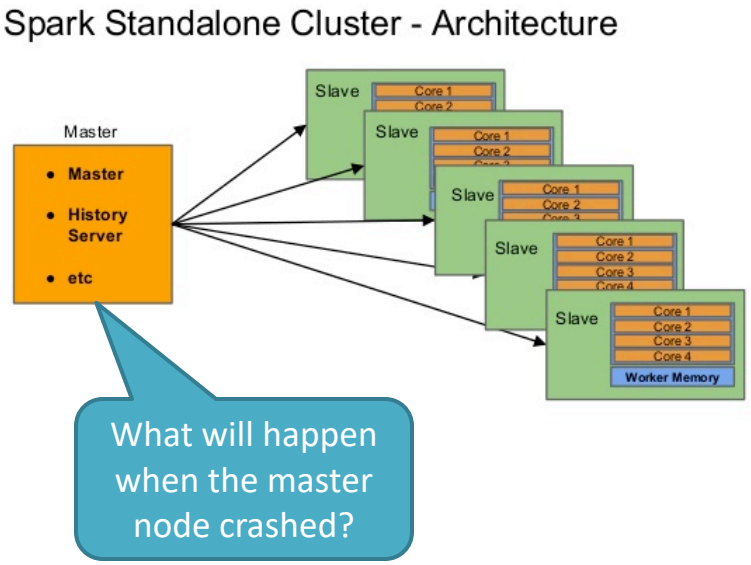
- Token Ring Algorithm



Why we need Distributed Election?

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





1. Bully algorithm

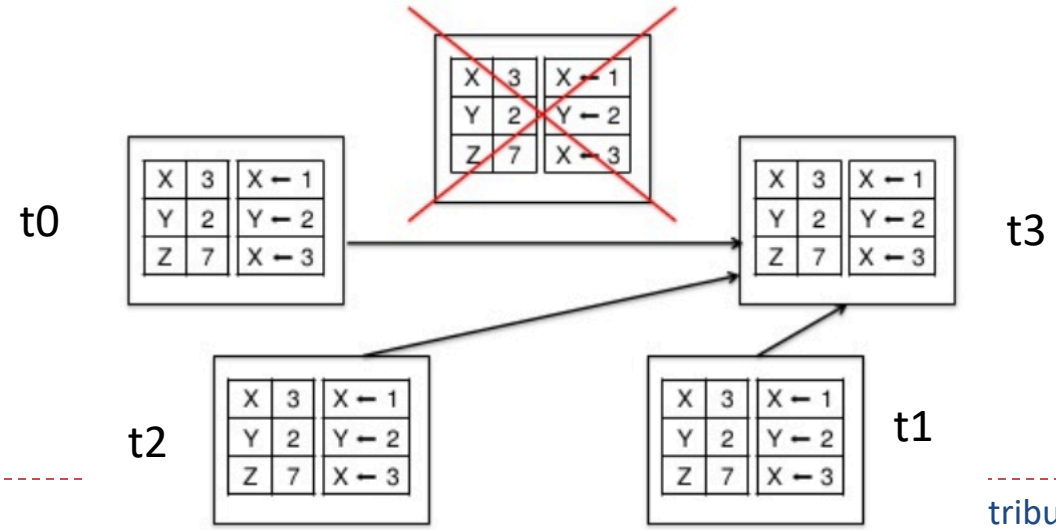
- Nodes have two types: normal nodes and master nodes.

- 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 → too much data
  - New election is required when any new node that is larger than the current master node ID → too many elections
  - Frequent switch over could happen when some nodes frequently join and exit the cluster → not stable

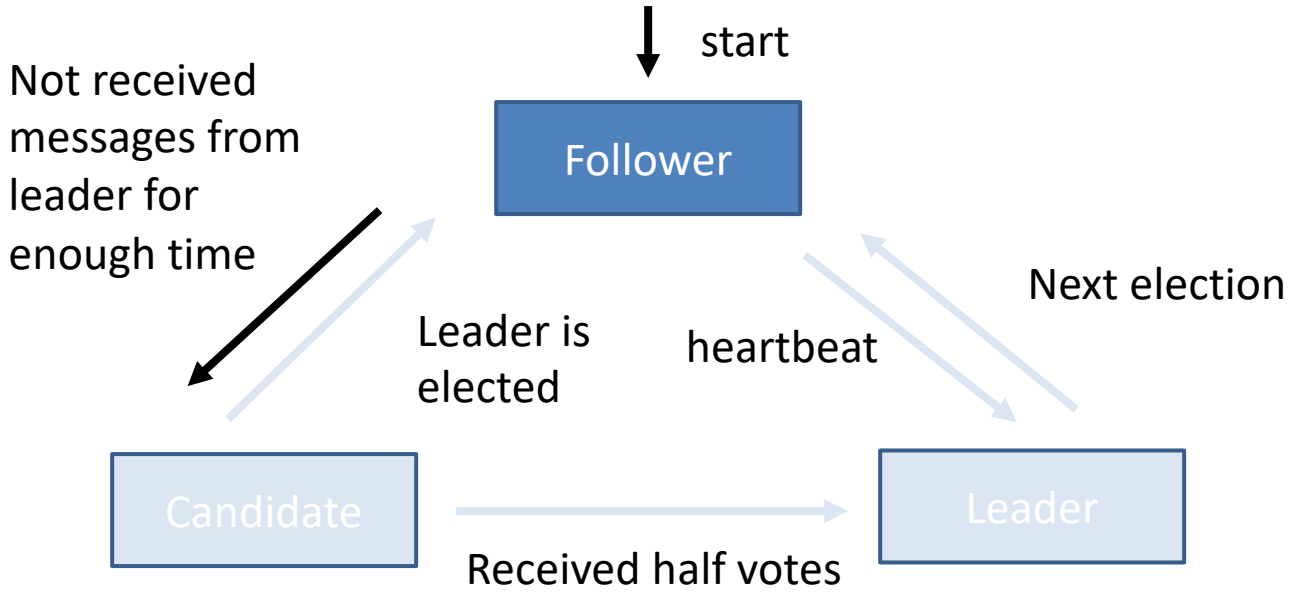
2. Raft algorithm

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

2. Raft algorithm

- How Raft algorithm works:

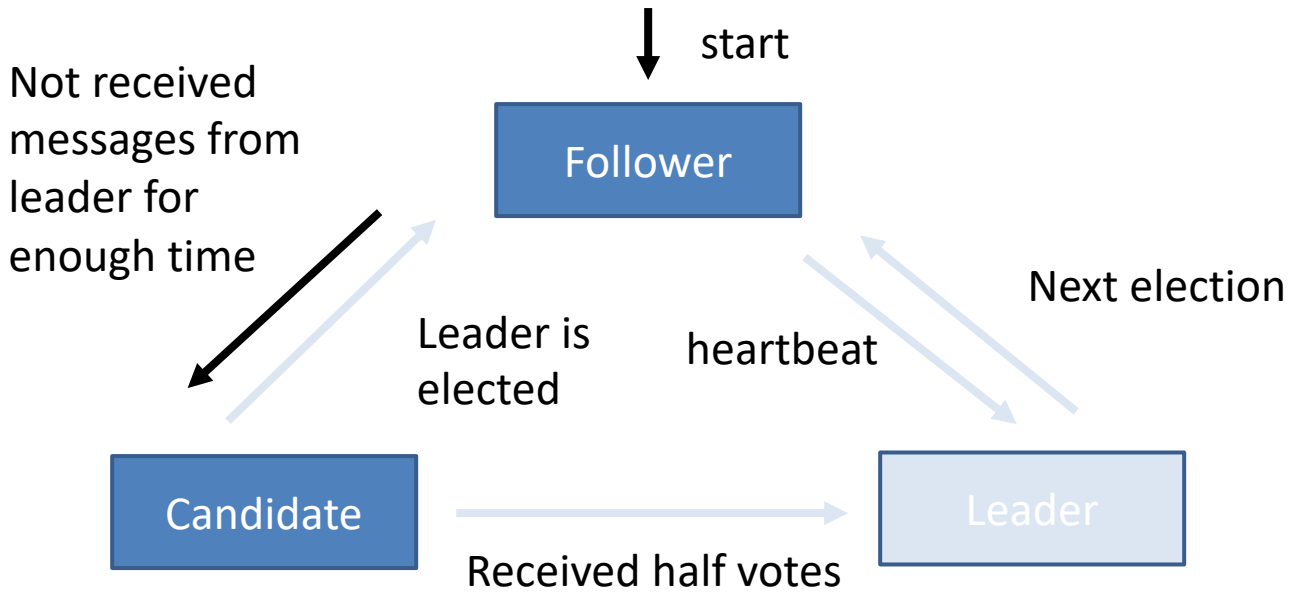
1. Initialization, all nodes are in the Follower state.



2. Raft algorithm

- How Raft algorithm works:

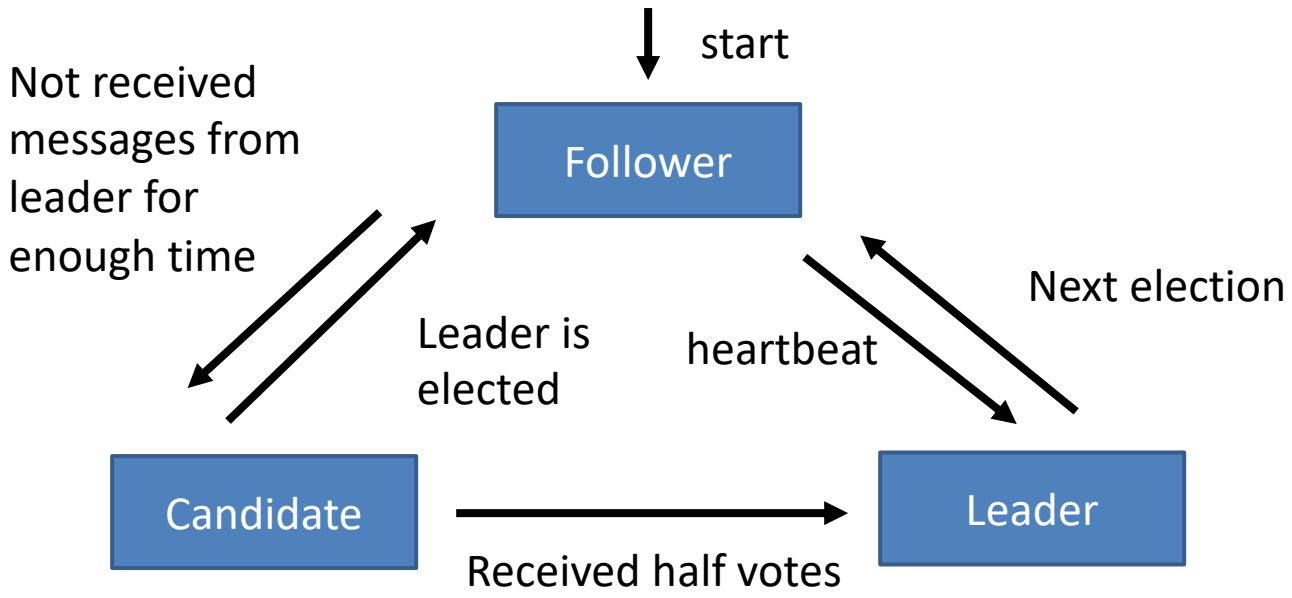
  2. At election, all nodes status change from Follower to Candidate, and send election requests to other nodes



2. Raft algorithm

- How Raft algorithm works:

  3. When nodes receive election requests, the voting starts. Every election, one node can only vote once.

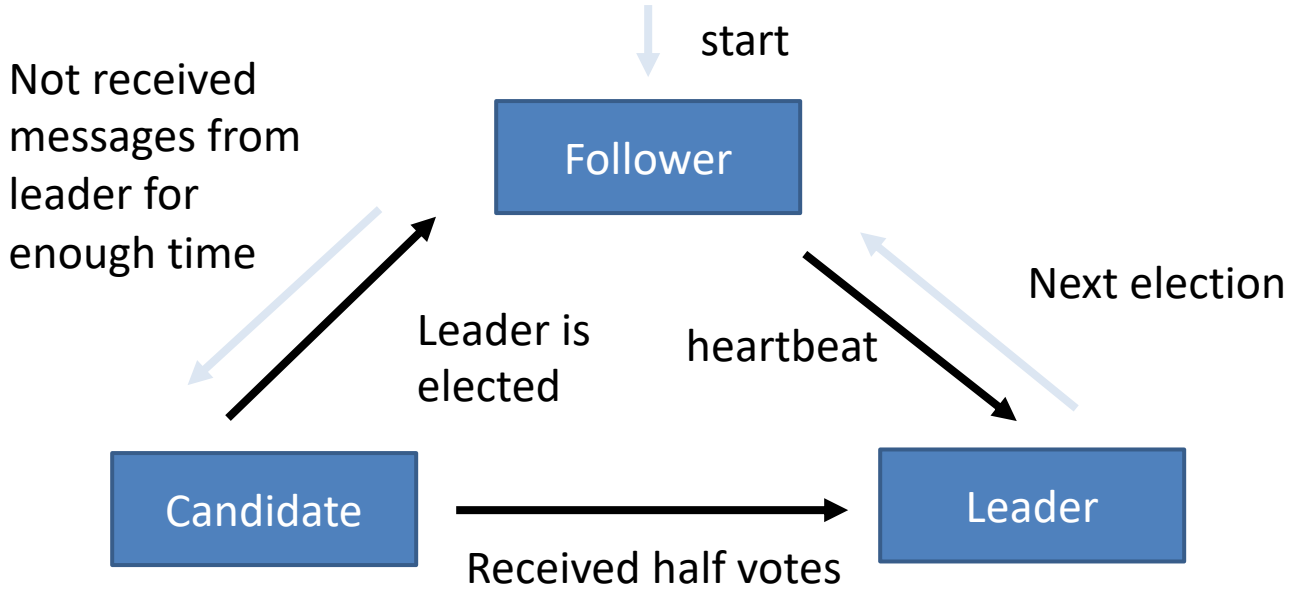


2. Raft algorithm

- How Raft algorithm works:

4. If the node that initiates the election request receives more than half of the votes, it will become the master node, and its status will be changed to Leader, and the status of other nodes will be reduced from Candidate to Follower.

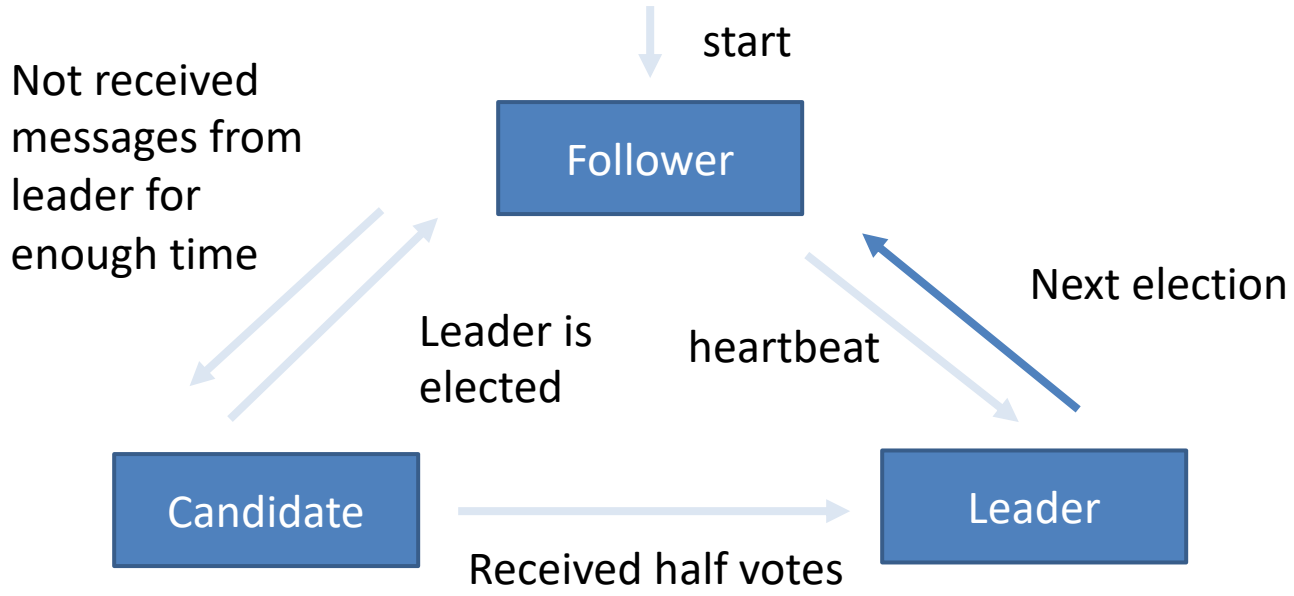
   Leader nodes and follower nodes send heartbeat packets periodically to detect whether the master node is alive.



2. Raft algorithm

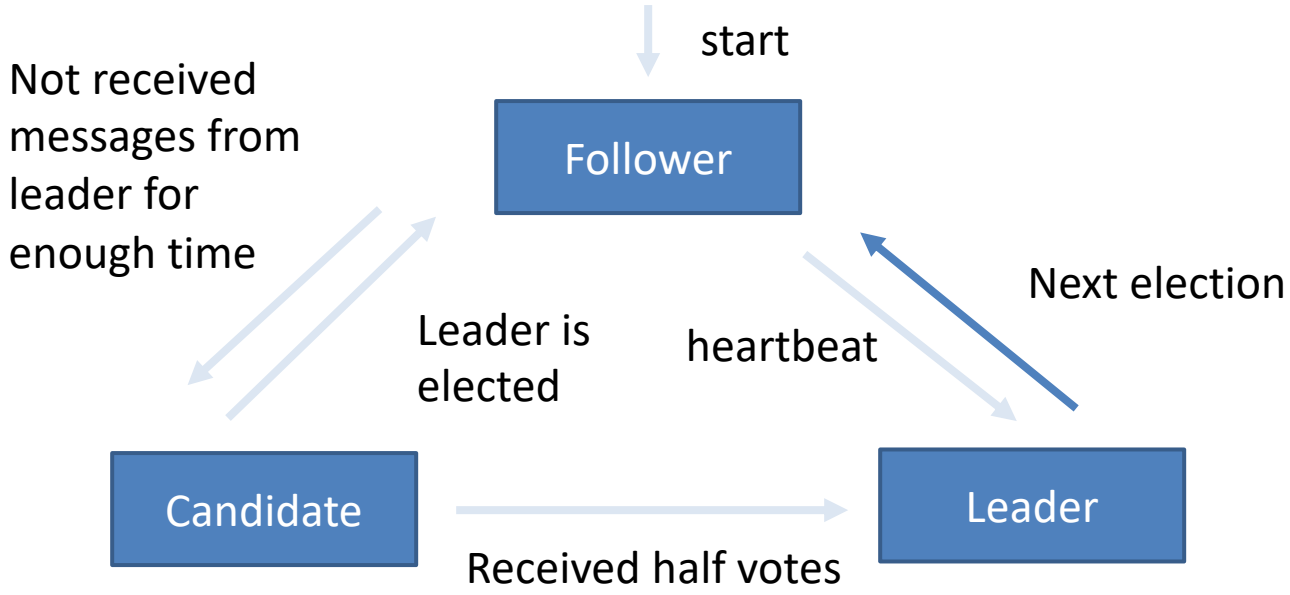
- How Raft algorithm works:

  5. When the term of the Leader node is reached, the status of the Leader node is changed from Leader to Follower, and a new round of election starts.



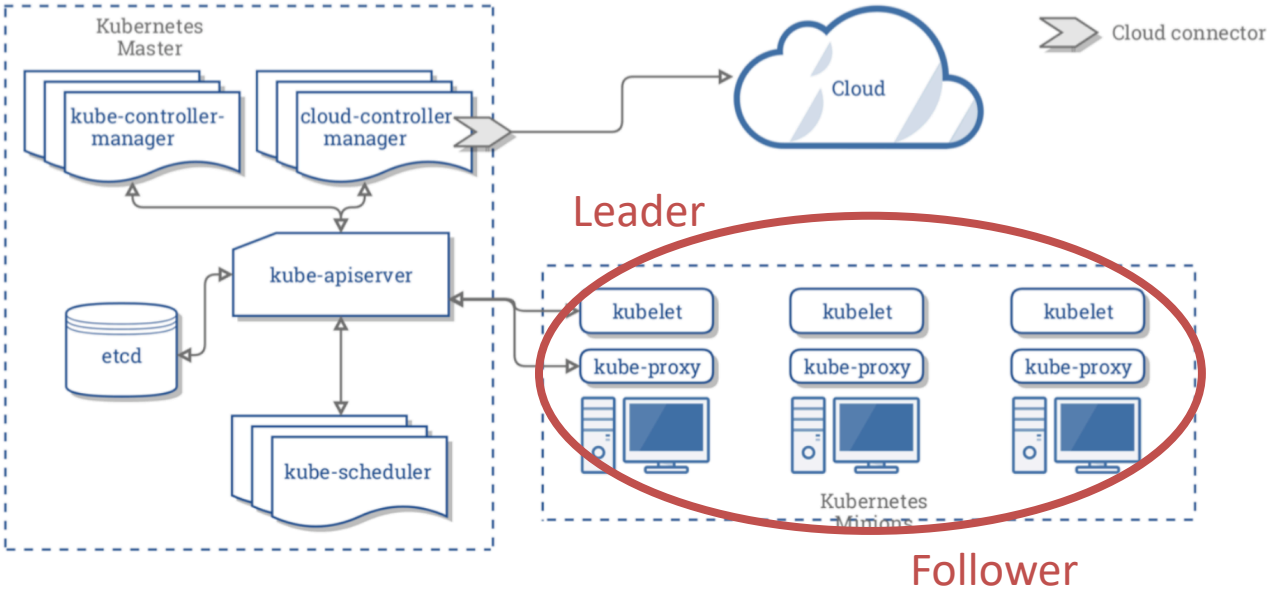
2. Raft algorithm

- When the new election starts in Raft algorithm:
  - When the leader term reaches
  - When the leader node fails or crashes



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 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

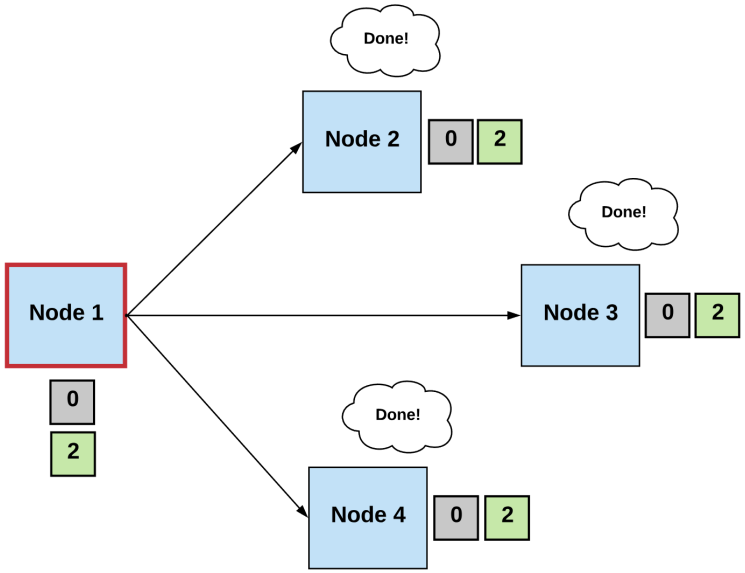
- Disadvantages:
  - It requires that each node in the system can communicate with each other (vote), and requires the node which has more than half of the votes to be the master, thus the ***communication traffic is large***

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	

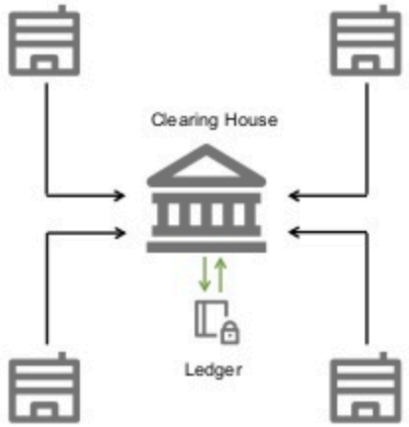
What is Distributed Consensus?

- Distributed consensus is the process of making all nodes agree on a certain state when multiple nodes can operate or record independently.



An Example of Distributed Consensus: distributed accounting

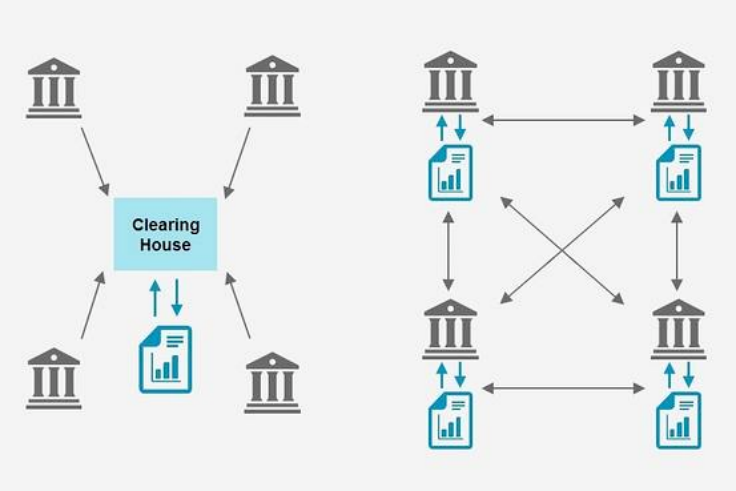
- Centralized accounting:
  - E.g., suppose only one bank exists in U.S.



- Potential problems:
  - The master node is very easy for accounting fraud
  - The master node could be performance bottlenecks
  - Low reliability, e.g., master node fails

An Example of Distributed Consensus: distributed accounting

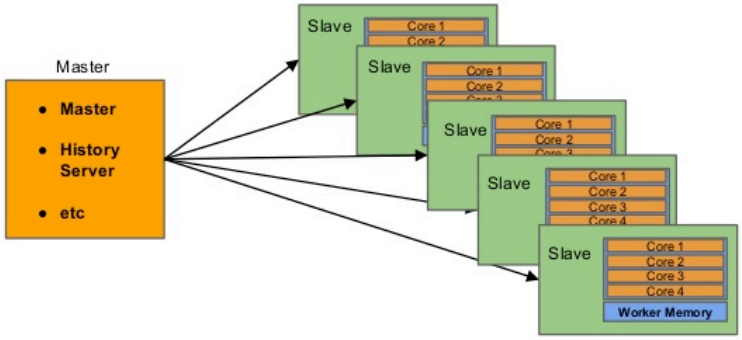
- Distributed accounting:
  - No centralized node
  - Any server that sees one transaction can record this transaction



- Potential benefits:
  - Results are trusted and accurate
  - System is strong and has high reliability

1. PoW (Proof-of-Work)

- In distributed election, only one node in the same round of elections could be the master node



- In distributed accounting, for one transaction, only one node can obtain the ***accounting right***, and then reaches ***an agreement*** with other nodes for the accounting results

- Two key points in distributed accounting :
  - Accounting right
  - Agreement

1. PoW (Proof-of-Work)

- PoW algorithm is a mechanism that competes for accounting rights based on the computing power of each node.

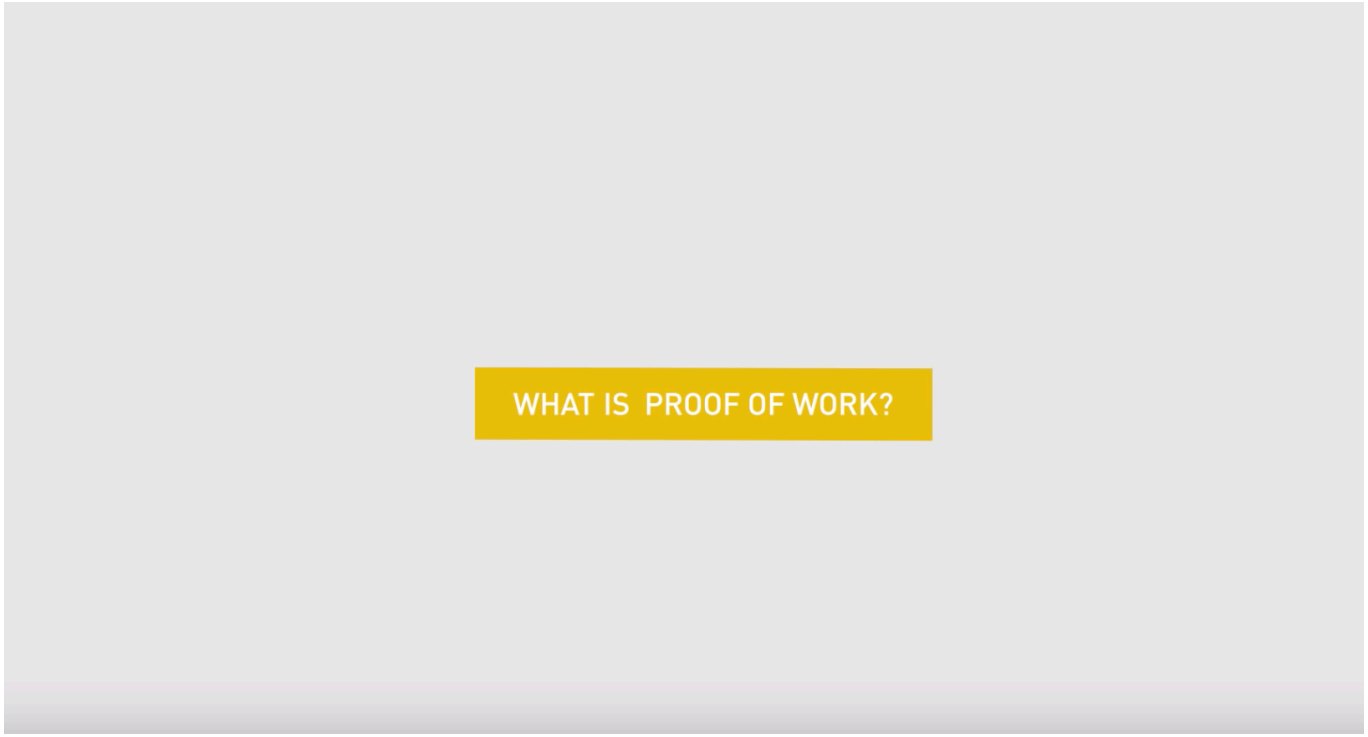
- Whoever has stronger computing power will be more likely to obtain accounting rights.


VS

VS


1. PoW (Proof-of-Work)

- https://youtu.be/3EUAcxhuoU4



2. PoS (Proof-of-Stake)

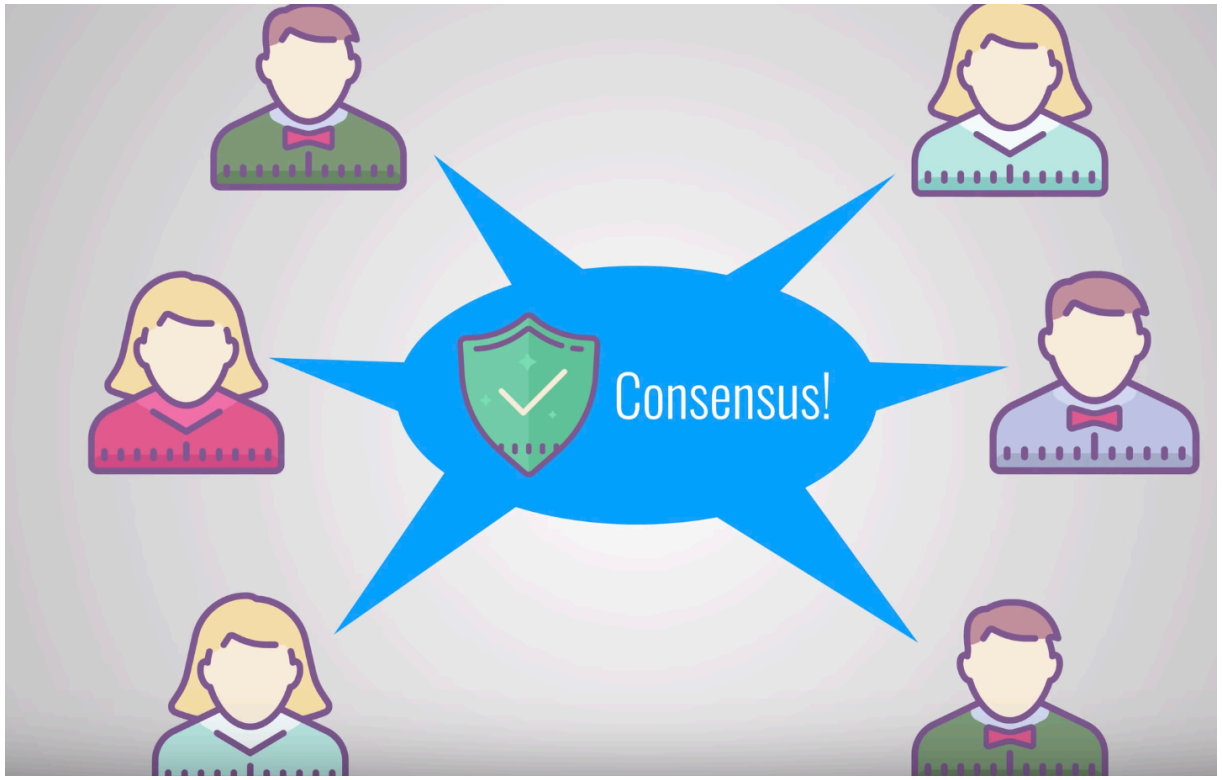
- Key idea: using ***equity (wealth or age)*** instead of computing power to determine the accounting right

- Equity (wealth or age) example:
  - If you hold 100 coins for a total of 50 days, then your coin age is 5000

- Whoever has more equities will be more likely to obtain accounting rights.

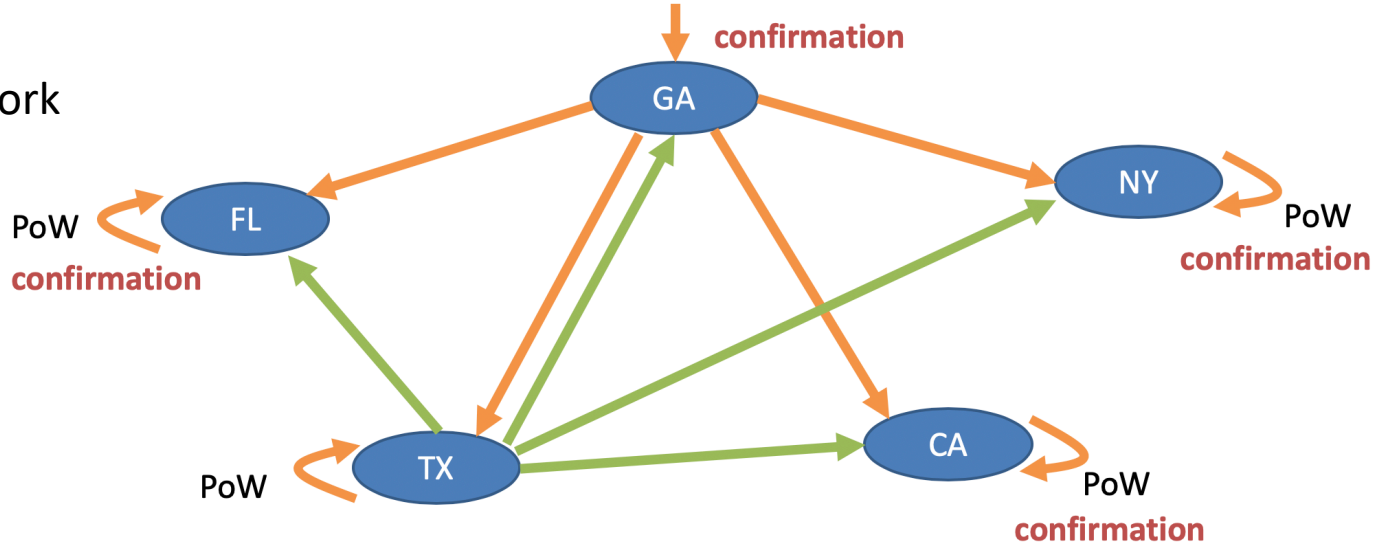
2. PoS (Proof-of-Stake)

- https://youtu.be/M3EFi_POhps

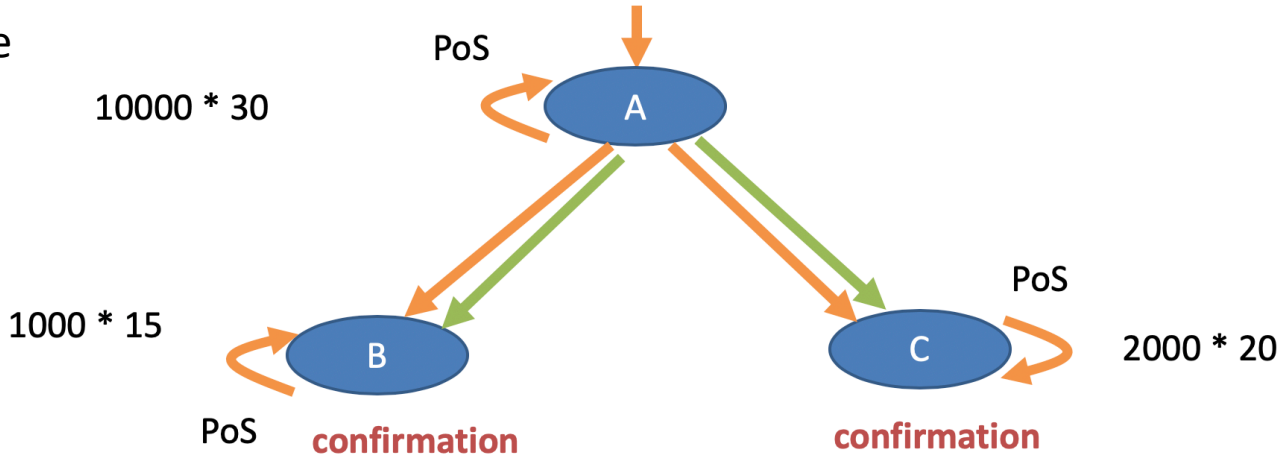


Example of Distributed Consensus: distributed accounting → compete for accounting right, and reaches an agreement

Proof-of-Work



Proof-of-Stake



Comparison

	PoW	PoS
Computing consumption	High	Medium
Structure	Distributed	Distributed
Performance (throughput)	PoW < PoS	
Transaction cost	High	Low
Application platform	Bitcoin	Ethereum