CS 7172
Parallel and Distributed Computation

Decentralized Structure

Kun Suo

Computer Science, Kennesaw State University

https://kevinsuo.github.io/

Outline

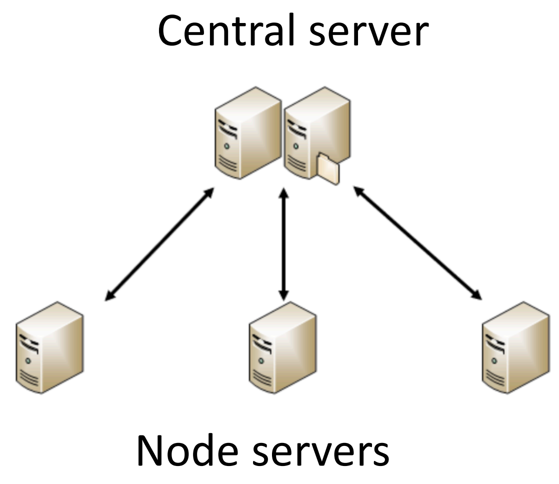
- Computer networks, primarily from an application perspective
- Protocol layering
- Client-server architecture
- End-to-end principle
- TCP
- Socket programming

Revisit Centralized Structure

- The central server is composed of one or more servers.

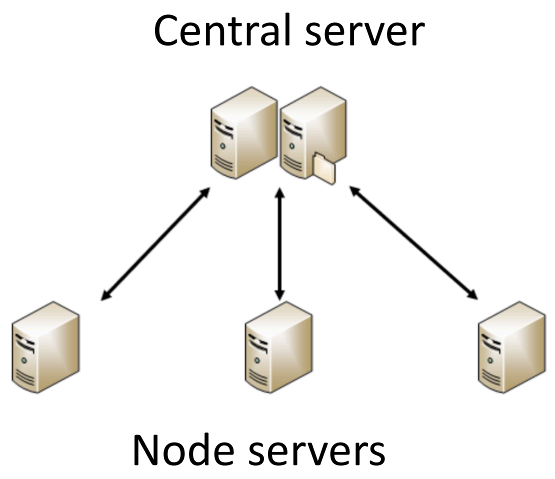
- All data in the system are stored in the central server, and all business in the system is processed by the central server first. The central server performs resource and task scheduling.

- Multiple node servers are connected to the central server and report their information to the central server. The central server sends tasks to the node server based on the information; the node server executes the tasks and sends the results & feedback to the central server.



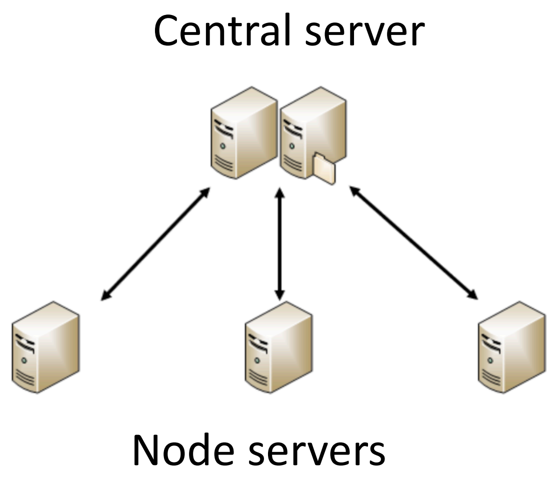
Revisit Centralized Structure

- Advantages:
  - Simple structure
  - Centralized manage and scheduling, the node servers do not need to communicate with each other and only need to communicate with the central node



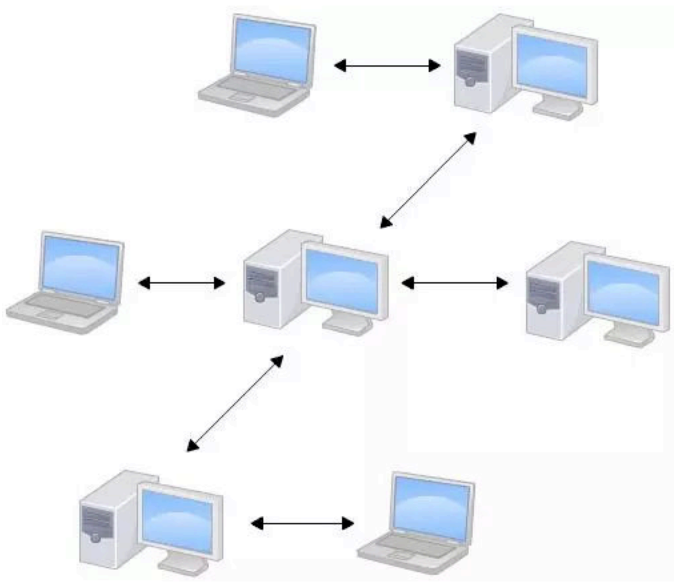
Revisit Centralized Structure

- Disadvantages:
  - This structure has high requirements on the performance of the central server
  - Single point of bottleneck
  - Single point of failure



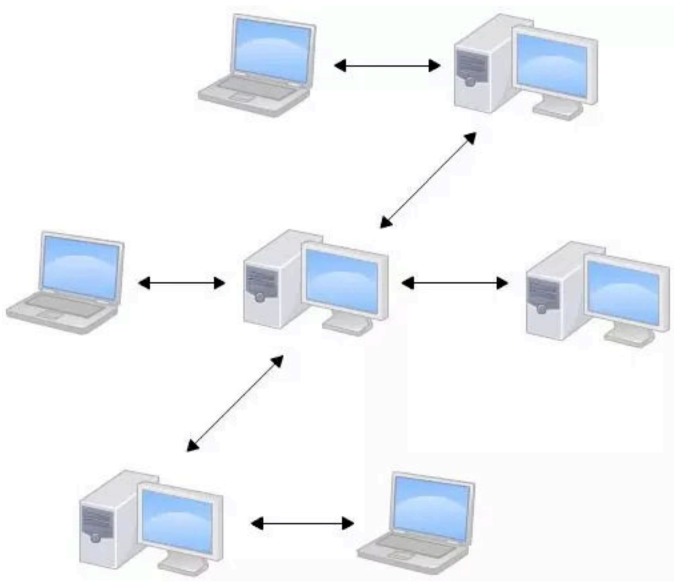
Decentralized Structure

- Decentralized Structure is also named as distributed structure
- Service execution and data storage are distributed to different server clusters
- Communication and coordination between server clusters are through message passing



Decentralized Structure

- Advantages:
  - No central server or node server, all servers are equal (peer-to-peer) and the structure is flat
  - No single point bottleneck and single point of failure
  - Improved system concurrency, more suitable for large-scale cluster management



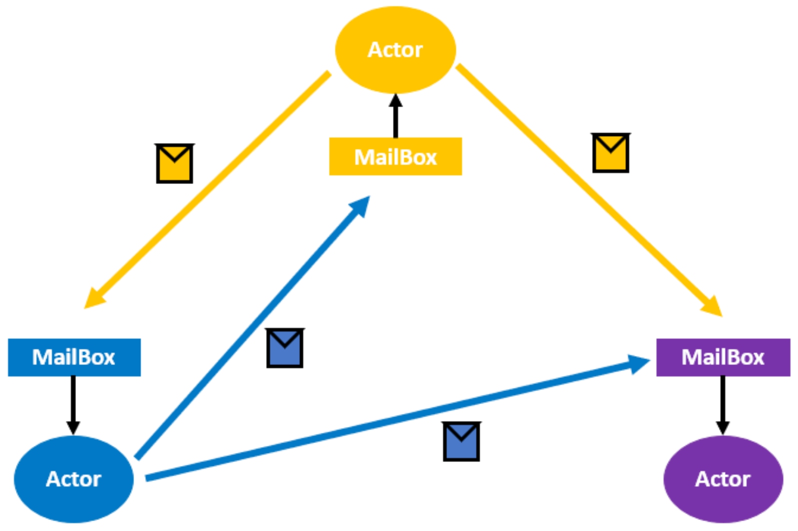
Example of Decentralized Structure: Akka cluster

- Actor model: An object that encapsulates state and behavior. It receives messages and performs computation based on the message

- Actors are isolated from each other and do not share memory

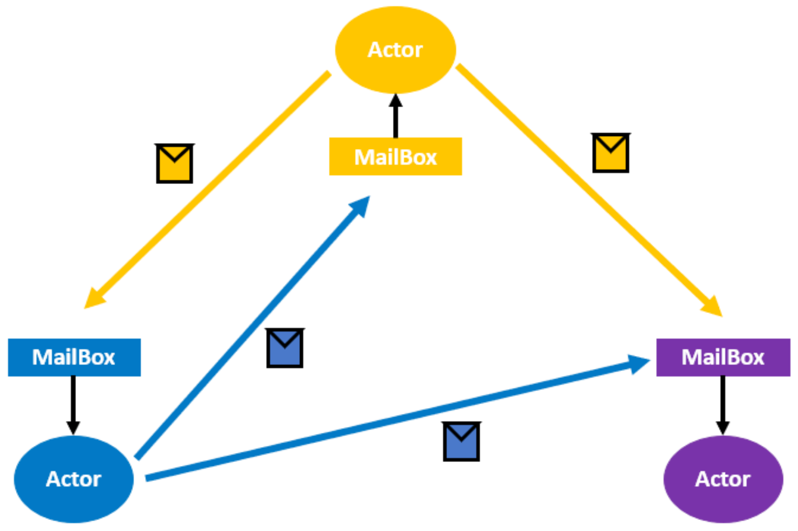
- Actors can communicate through mail exchange

- In a distributed system, a server or a node can be regarded as an Actor. Mail is used for communication between Actors.



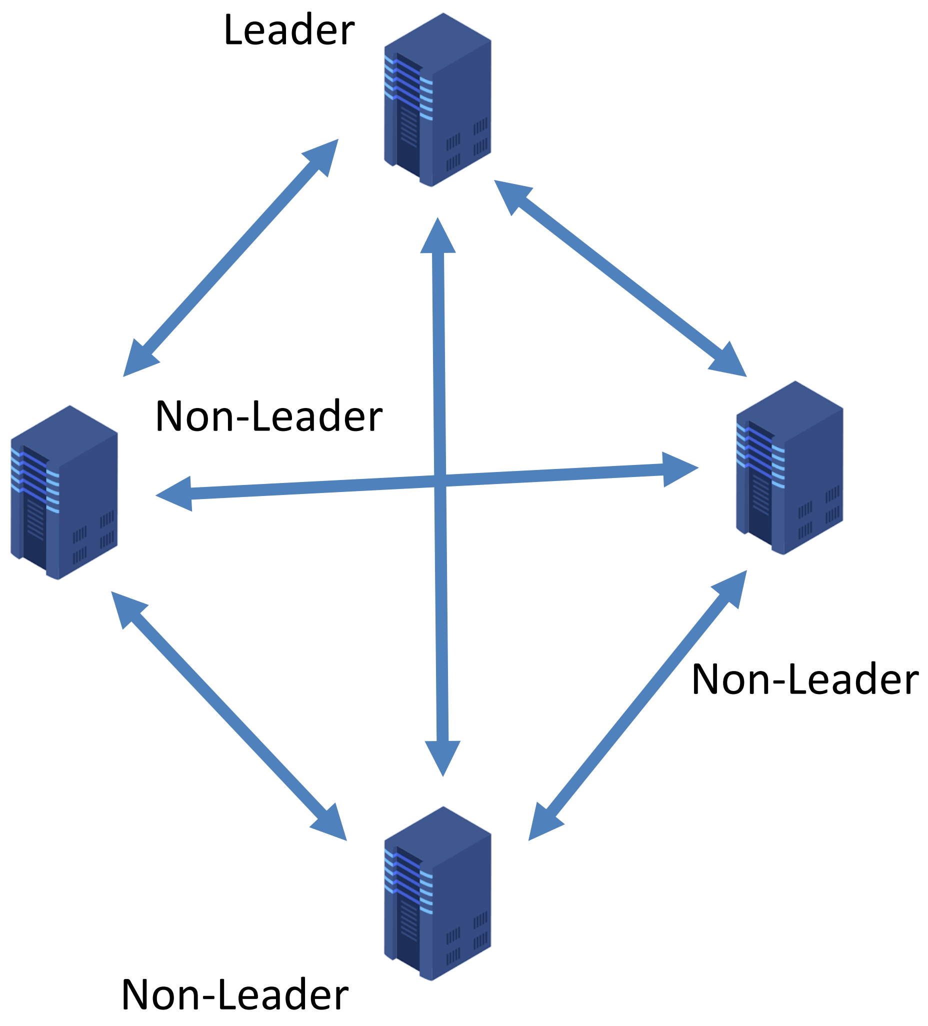
Example of Decentralized Structure: Akka cluster

- What is a Akka cluster?
  - Akka cluster is based on Actor model
  - It is an asynchronous, non-blocking, high-performance event-driven programming model
  - It provides a non-centralized architecture cluster management module for building scalable and elastic distributed applications



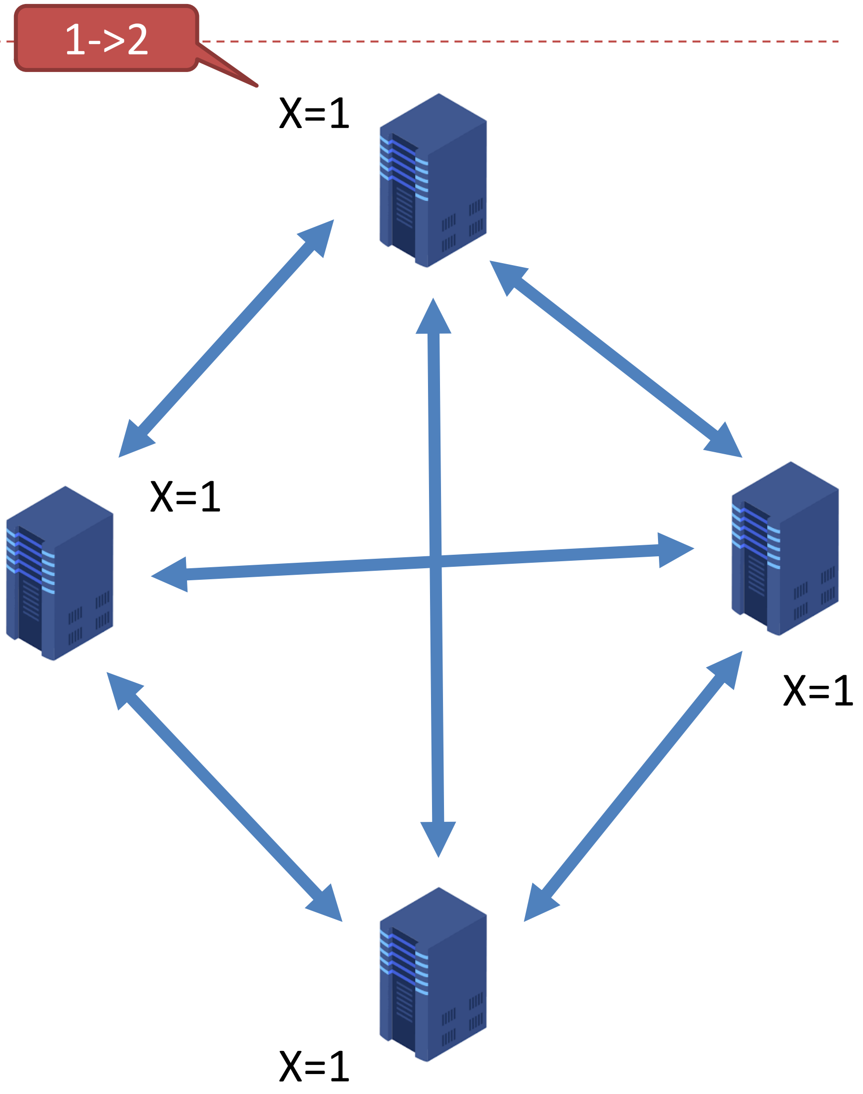
Example of Decentralized Structure: Akka cluster

- Who to manage the nodes in Akka cluster?
  - Nodes are divided into Leader nodes and non-Leader nodes.
  - Compared with non-Leader nodes, the Leader node only adds the function of adding and removing nodes responsible for the node.
  - All nodes are equal in the decentralized structure



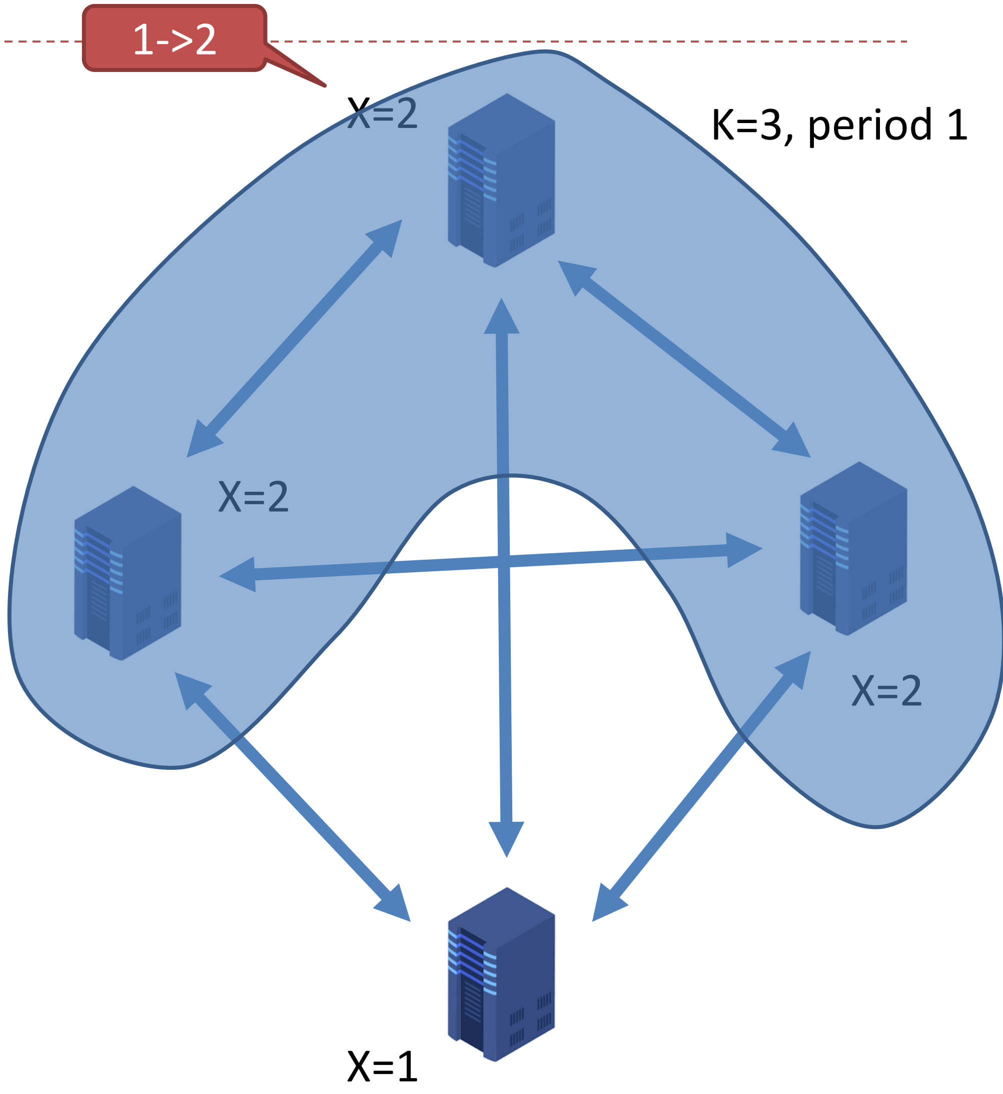
Example of Decentralized Structure: Akka cluster

- How to transmit data in Akka cluster?
  - Suppose node 1 tries to modify X from 1 to 2, how to guarantee the data consistency?
    - Review distributed consensus in previous courses



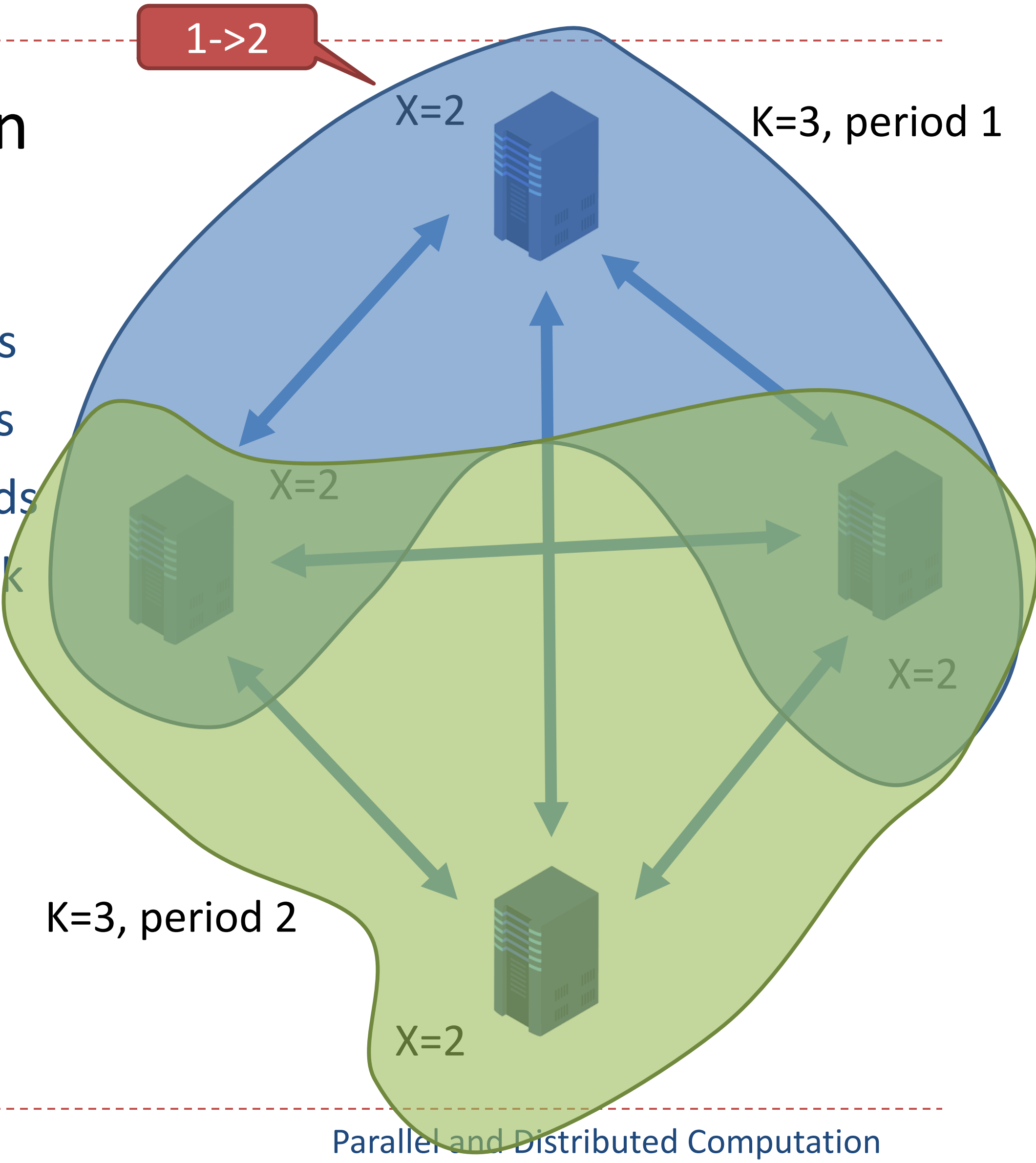
Example of Decentralized Structure: Akka cluster

- How to transmit data in Akka cluster?
  - Each node periodically selects k nodes from the list of nodes maintained by itself, and sends its data information to these k nodes to reach a consensus



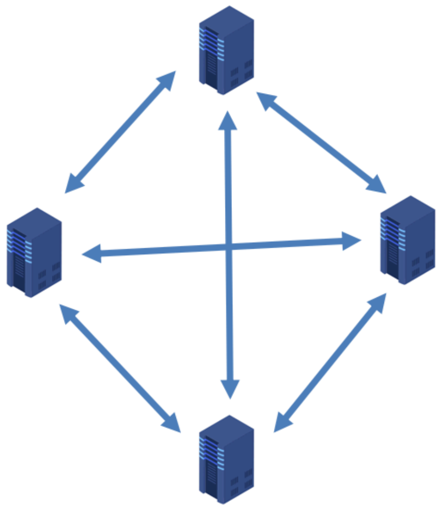
Example of Decentralized Structure: Akka cluster

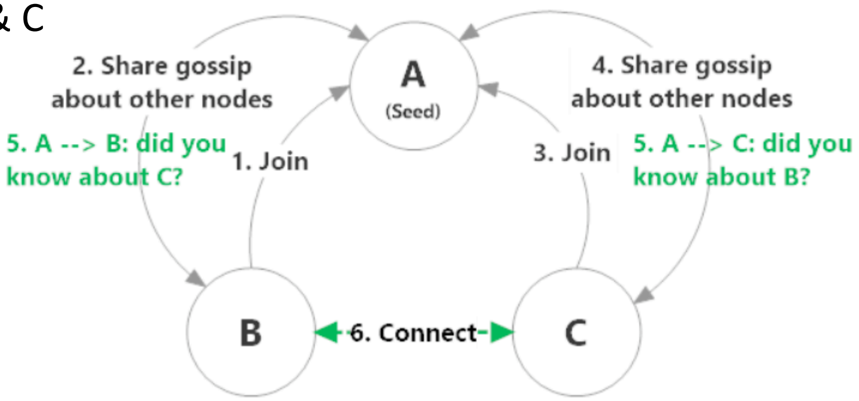
- How to transmit data in Akka cluster?
  - Each node periodically selects k nodes from the list of nodes maintained by itself, and sends its data information to these k nodes to reach a consensus



Example of Decentralized Structure: Akka cluster

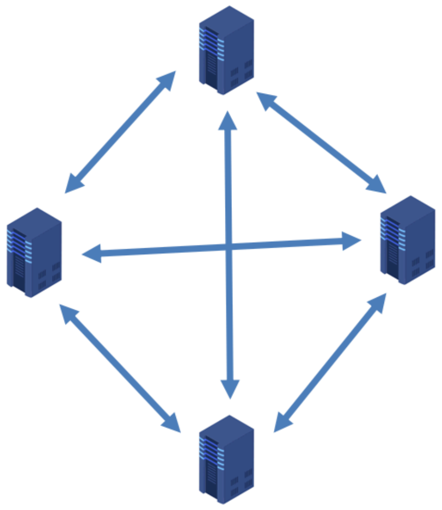
- How to create Akka cluster?
  - When creating Akka cluster, all nodes are divided into 3 categories:
    - Seed node: we can have multiple seeds and the seed nodes are defined by static configuration file or in runtime
    - First seed node: the first seed among all seed nodes. E.g., Node A
    - Normal node: they can send "Join" message to the seed nodes and apply to join the cluster. E.g., Node B & C

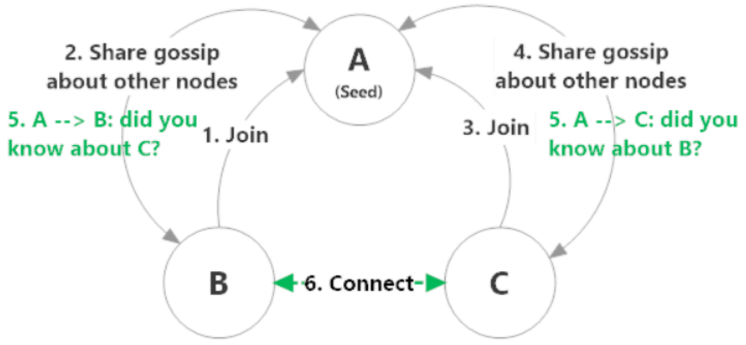



A
(Seed)
2. Share gossip about other nodes
4. Share gossip about other nodes
5. A --> B: did you know about C?
5. A --> C: did you know about B?
1. Join
3. Join
B
C
6. Connect

Example of Decentralized Structure: Akka cluster

- How to create Akka cluster?
  - Create Akka cluster:
    - If this node is the first seed node, add itself to the cluster list, build a cluster centered on itself;
    - if this node is a seed node, sends requests to the first seed node to join the cluster. If the first seed node approves, join the cluster;
    - if this node is a normal node, sends requests to any seed node (including the first seed node) to join the cluster. If seed nodes approve, join the cluster.

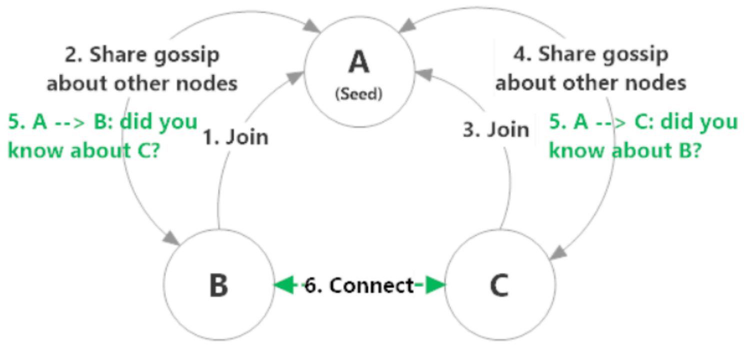


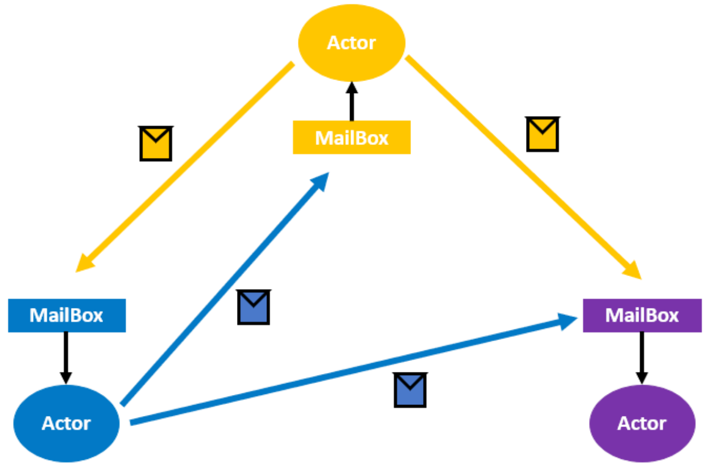

A
(Seed)
2. Share gossip about other nodes
4. Share gossip about other nodes
5. A --> B: did you know about C?
5. A --> C: did you know about B?
1. Join
3. Join
B
C
6. Connect

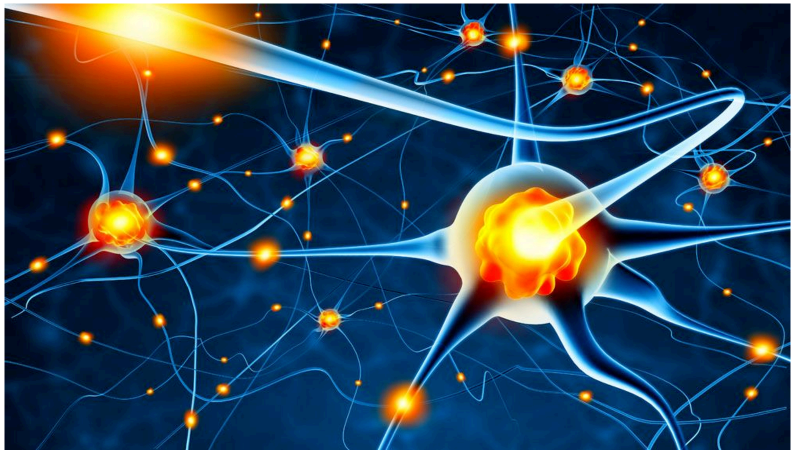
Example of Decentralized Structure: Akka cluster

- Akka cluster is a completely decentralized cluster management system. After the cluster is created, each node can execute the Actor application, so it supports concurrent operations

- Akka cluster works like nervous tissues

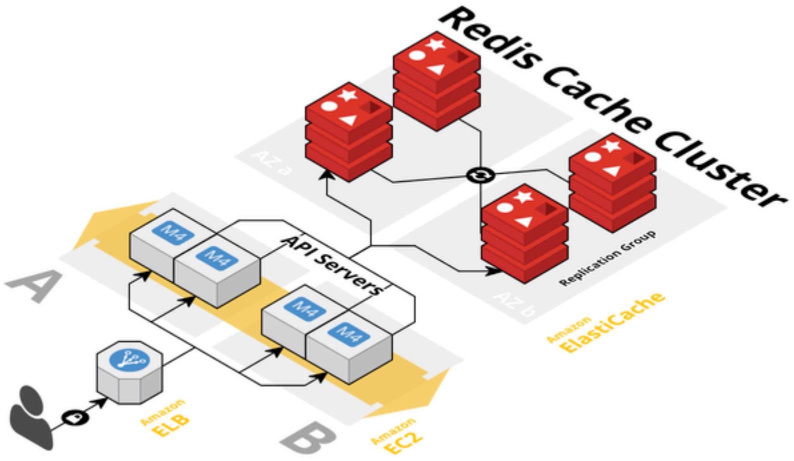

A
(Seed)
2. Share gossip about other nodes
4. Share gossip about other nodes
1. Join
3. Join
5. A --> B: did you know about C?
5. A --> C: did you know about B?
B
C
6. Connect





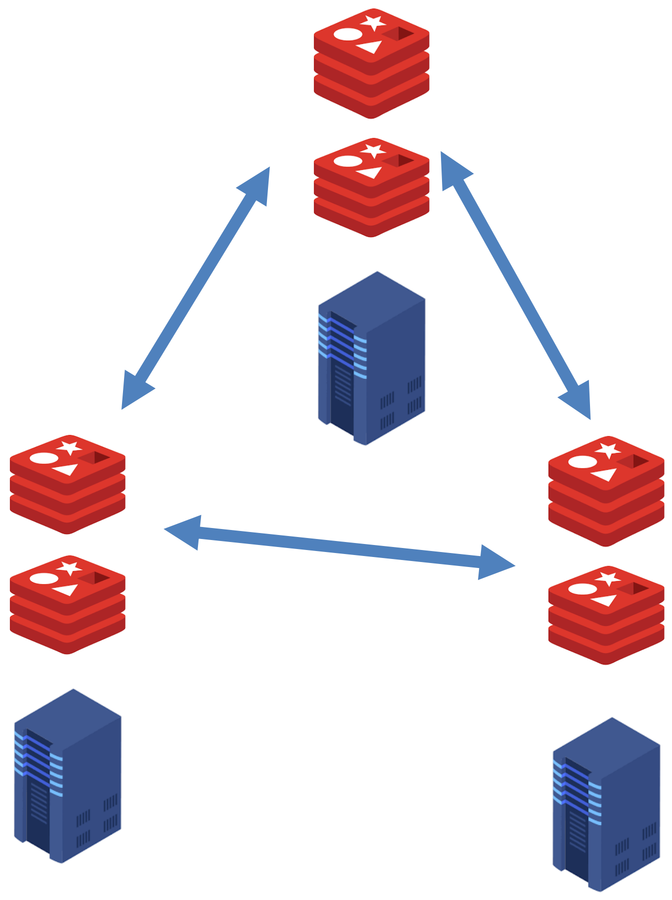
Example of Decentralized Structure: Redis cluster

- Redis is an open source, high-performance distributed key-value database and data can be stored on different Redis nodes
  - Support data persistence, which can save the data from memory to disk, and can be reloaded and used again when system reboots;
  - Support multiple data structures, including key-value, list, set, and hash
  - Support data backup



Example of Decentralized Structure: Redis cluster

- How to achieve data reliability on a distributed Redis cluster?
  - Each node in the cluster has one active and one standby, which means that two Redis services are running on each server

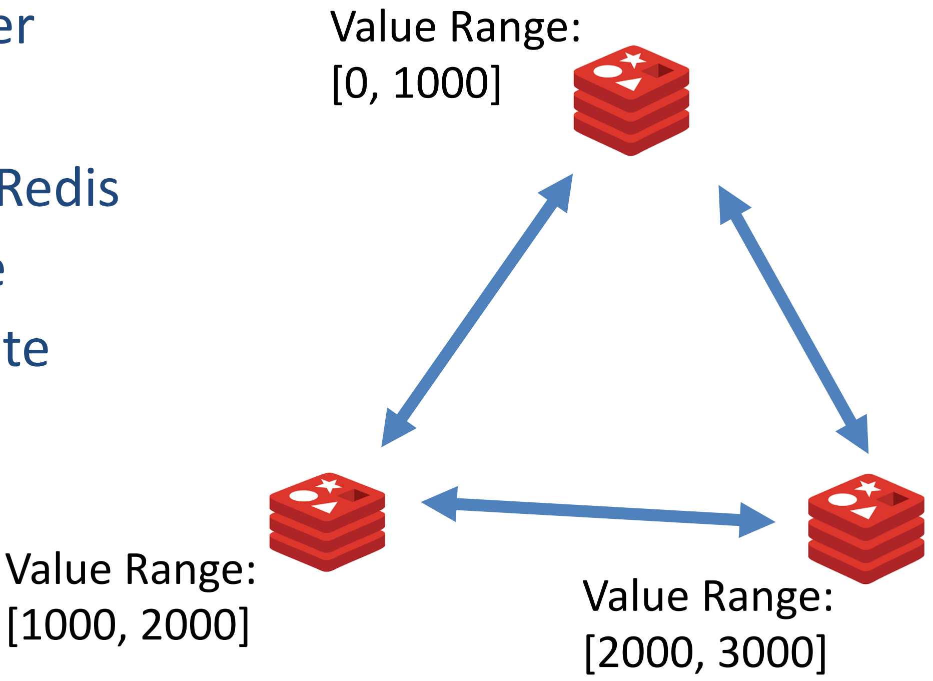


Example of Decentralized Structure: Redis cluster

- How to store data on a distributed Redis cluster?

$$\begin{tikzcd}\text{Input} \arrow[r] & \text{Key} \arrow[r] & \text{Value}\end{tikzcd}$$

  - The Redis cluster uses cluster sharding to implement data sharding, which distributes Redis write operations to multiple nodes and improves the write concurrency capability.



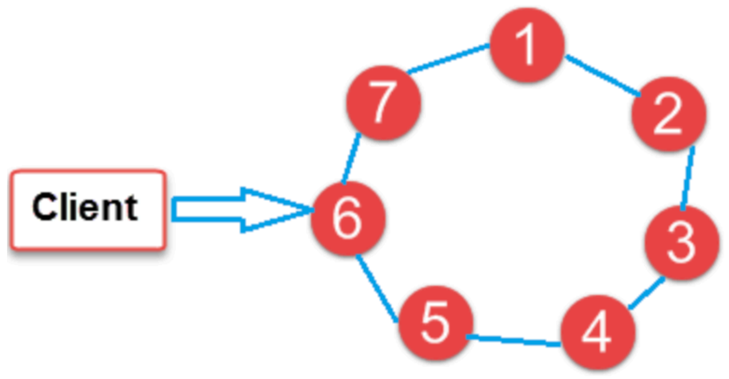
Example of Decentralized Structure: Redis cluster

- Redis distributed cluster
  - Non-centralized cluster management system, without a central node, will not cause a performance bottleneck due to a single node
  - Each node supports data storage and uses data sharding storage to improve the concurrent writing capacity.
  - Each node adopts the active and standby design, which improves the reliability

- Redis is widely used by Twitter, Uber, GitHub, Instagram, etc.

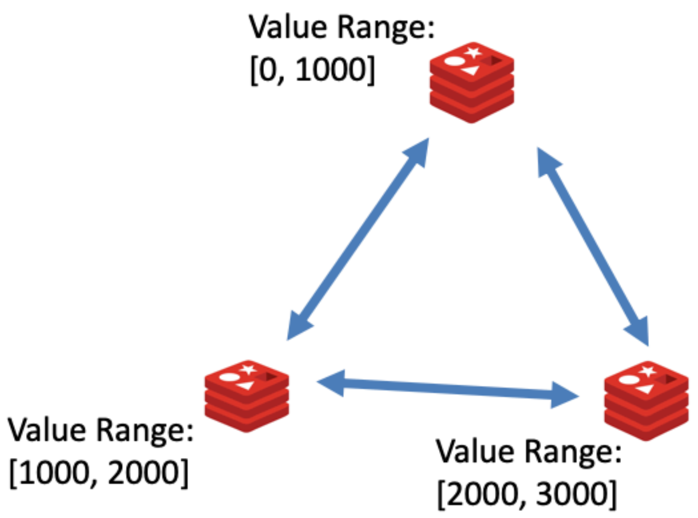
Example of Decentralized Structure: Cassandra cluster

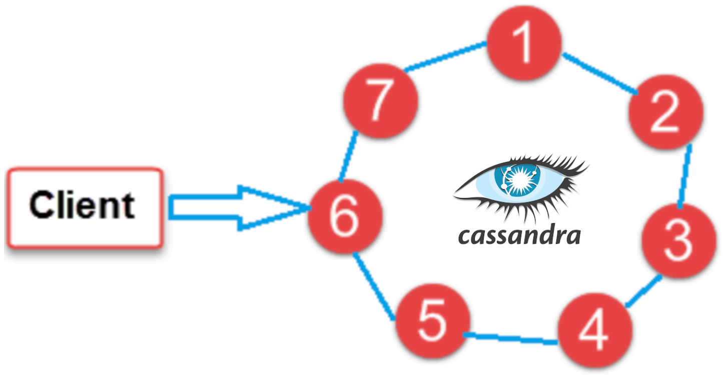
- The Cassandra architecture is a complete P2P structure based on consistent hashing
- No master node, all nodes have the same role
- The synchronization between Cassandra nodes is through P2P
- Each node in the cluster can store data and receive requests from clients



Example of Decentralized Structure: Cassandra cluster

- Difference between Redis cluster and Cassandra cluster:
  - Each node in Redis cluster contains a range of hash value
  - Each node in Cassandra cluster contains one single hash value





Example of Decentralized Structure: Cassandra cluster

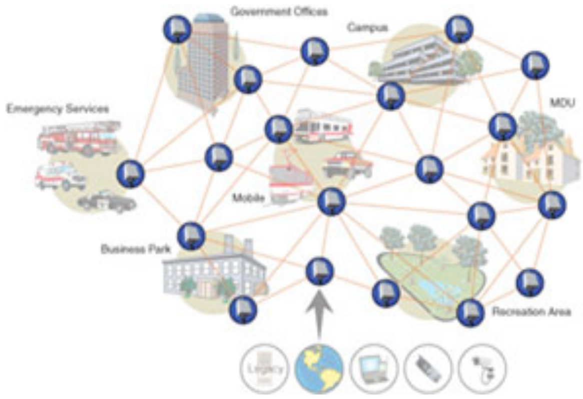
- Cassandra distributed cluster
  - Non-centralized cluster management system, without a central node, will not cause a performance bottleneck due to a single node
  - Each node stores a single hash value for distributed storage
  - The nodes use P2P for communication
- Redis is widely used by Apple, Comcast, eBay, Netflix etc.

Comparison

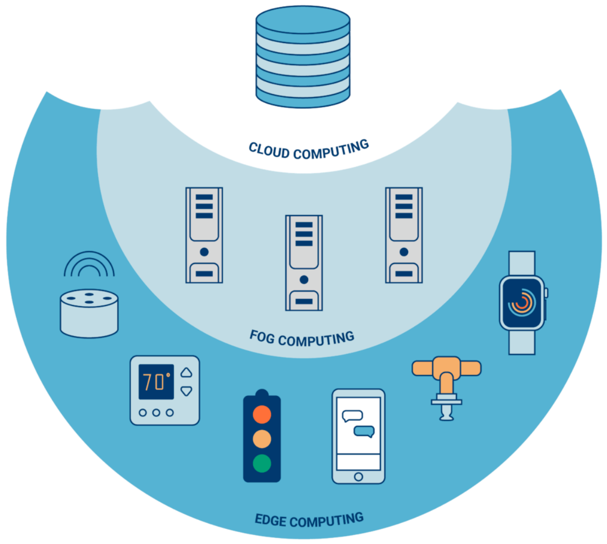
	Akka cluster	Redis cluster	Cassandra cluster
Support model	Actor model	Key-value	Key-value
System architecture	P2P structure	Network topology	P2P structure
Communication	Gossip protocol on network	Gossip protocol on network	Gossip protocol in P2P
Data storage	Store data on some configured nodes	Data sharding on different nodes	Store different hash value on various nodes
Whether the cluster needs a leader node?	Yes	Yes	No

Scenarios for Decentralized Structure

- Although the centralized structure is easy to build, however, it introduces single-point bottleneck and single-point failure. Non-centralized structures are the preferred options for large-scale distributed systems



- Edge computing is to provide application developers and service providers with cloud services at the edge of the network; the goal is to provide computing, storage, and network bandwidth near data inputs or users. Also, Edge computing involves massive devices and has high requirements for reliability and speed. It is better to use decentralized structure for edge computing.


CLOUD COMPUTING
FOG COMPUTING
EDGE COMPUTING
70°