Chord : A Scalable Peer-to-peer Lookup Service for Internet Applications

128569G Daichi TOMA
128575B Yui SHIMURA
128567A Atsuhiro HORIKAWA

What is Chord?

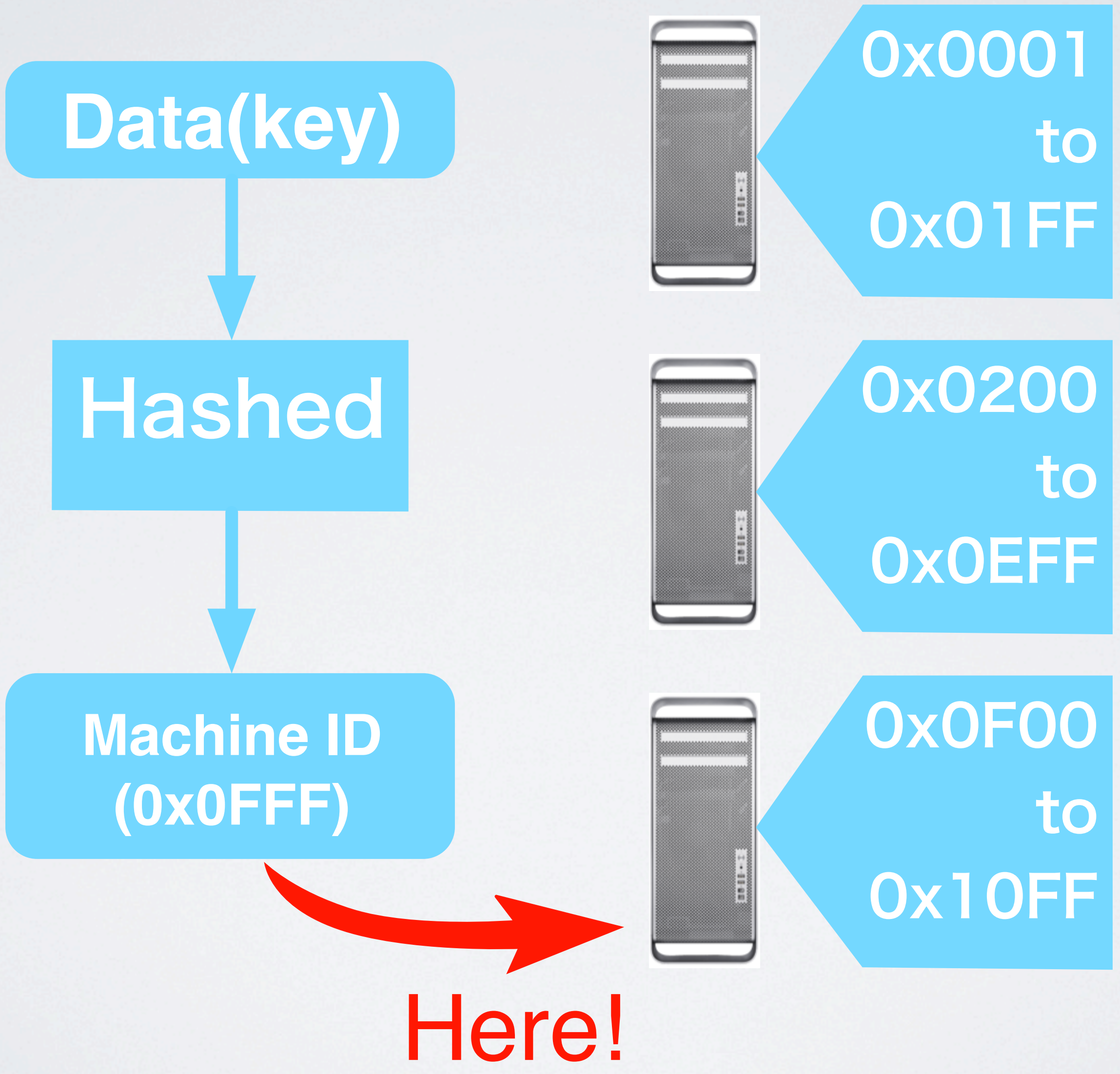
- Algorithm for distributed computing
- P2P network
- This dissertation is many quote on Computer Science dissertation
- Chord is a DHT algorithm

What is DHT?

- DHT = Distributed hash table
- DHT is generic name which provides a lookup service similar to hash table
- Data have “Key” and “Value”
- Node have unique “Machine ID” and “Routing table”
- DHT provides a store and get of Data to some of the nodes

What is DHT?

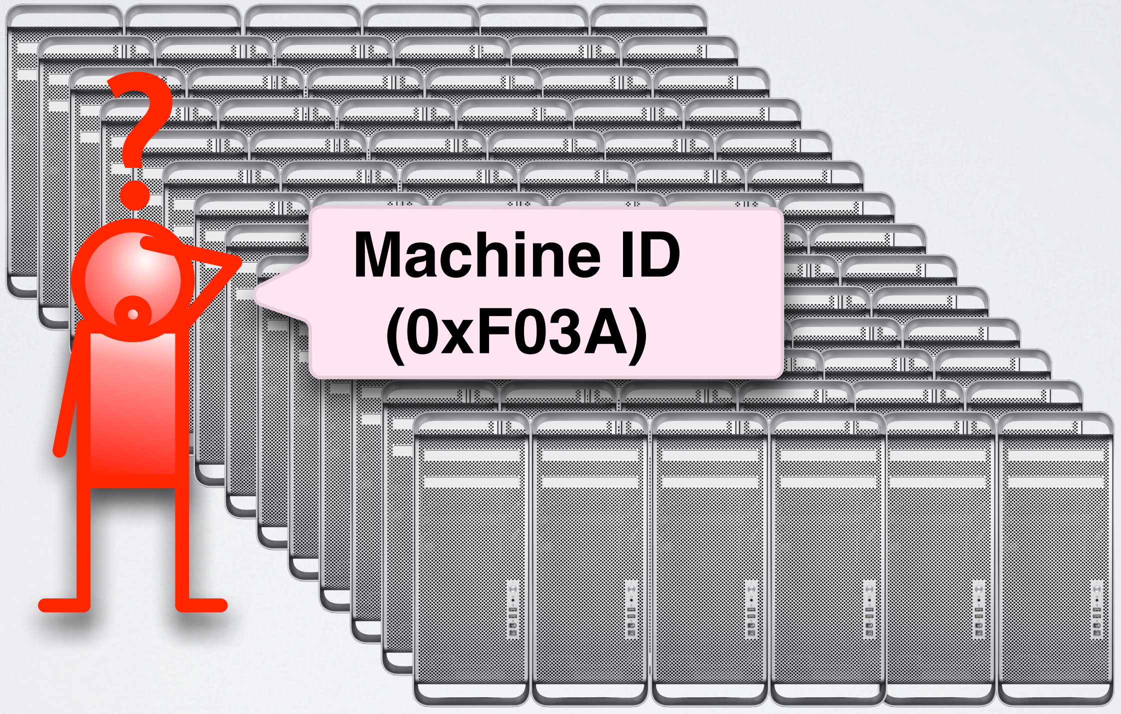
Example : Data registration



What is DHT?

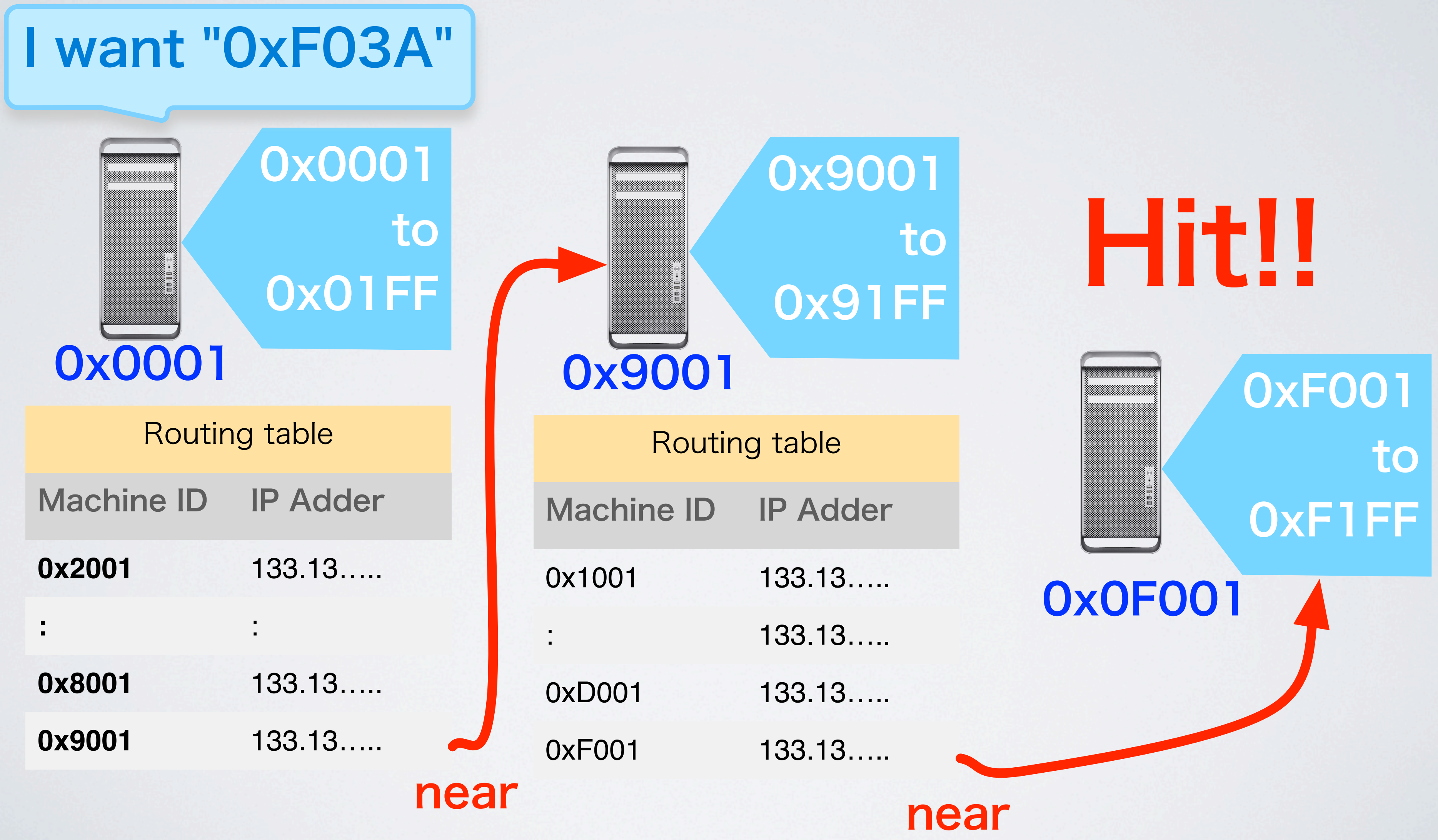
Found Data

If distributed field of huge...



What is DHT?

Example : Data retrieve



Routing table

Machine ID	IP Adder
0x2001	133.13…..
:	:
0x8001	133.13…..
0x9001	133.13…..

Routing table

Machine ID	IP Adder
0x1001	133.13…..
:	133.13…..
0xD001	133.13…..
0xF001	133.13…..

Chord Algorithm

- ID space and Responsible Domain is based on <u>Consistent Hashing</u>

- Chord have one of the three pattern routing tables

  1 : Successor Only

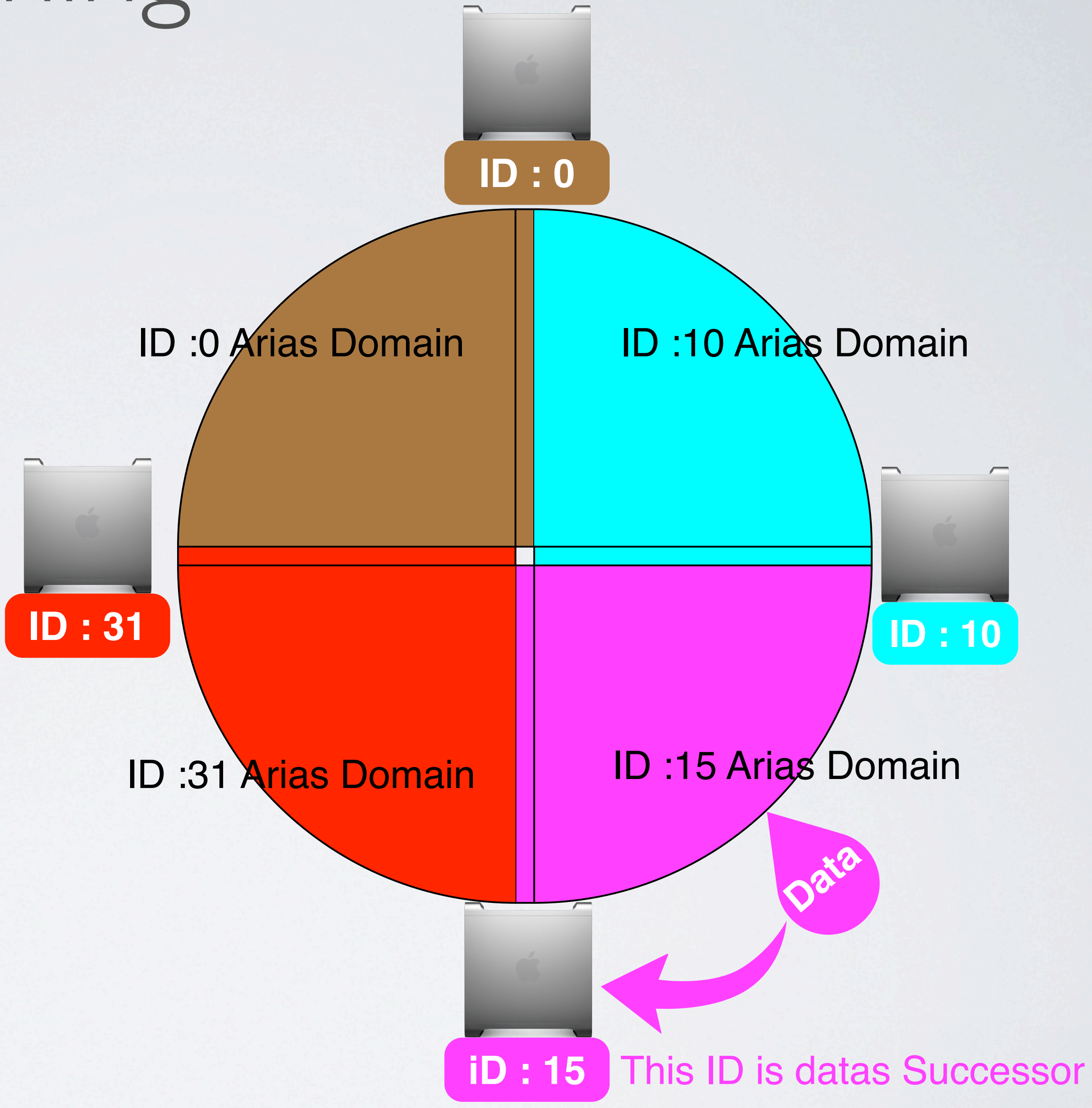
  2 : Successor + FingerTable

  3 : SuccessorList + FingerTble

- Responsible Domain can edited by Join Operation and Stabilize Operation

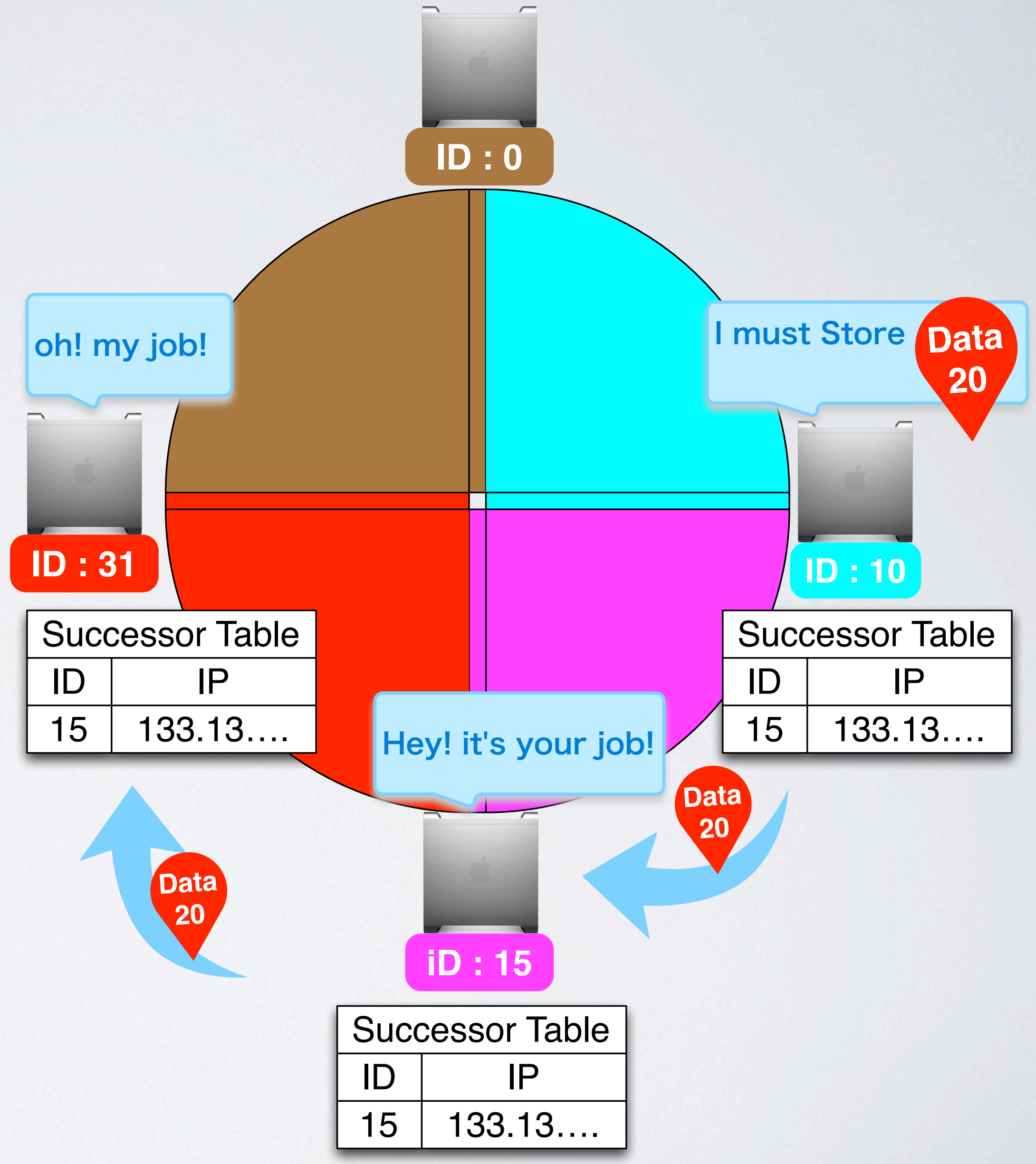
Consistent Hashing

- Machine ID is created by hashed machines IP address
- ID space is a ring



Successor

- Successor guarantees reachability
- if the system had many node, when took some time to process



Successor Table

ID	IP
15	133.13.…

Successor Table

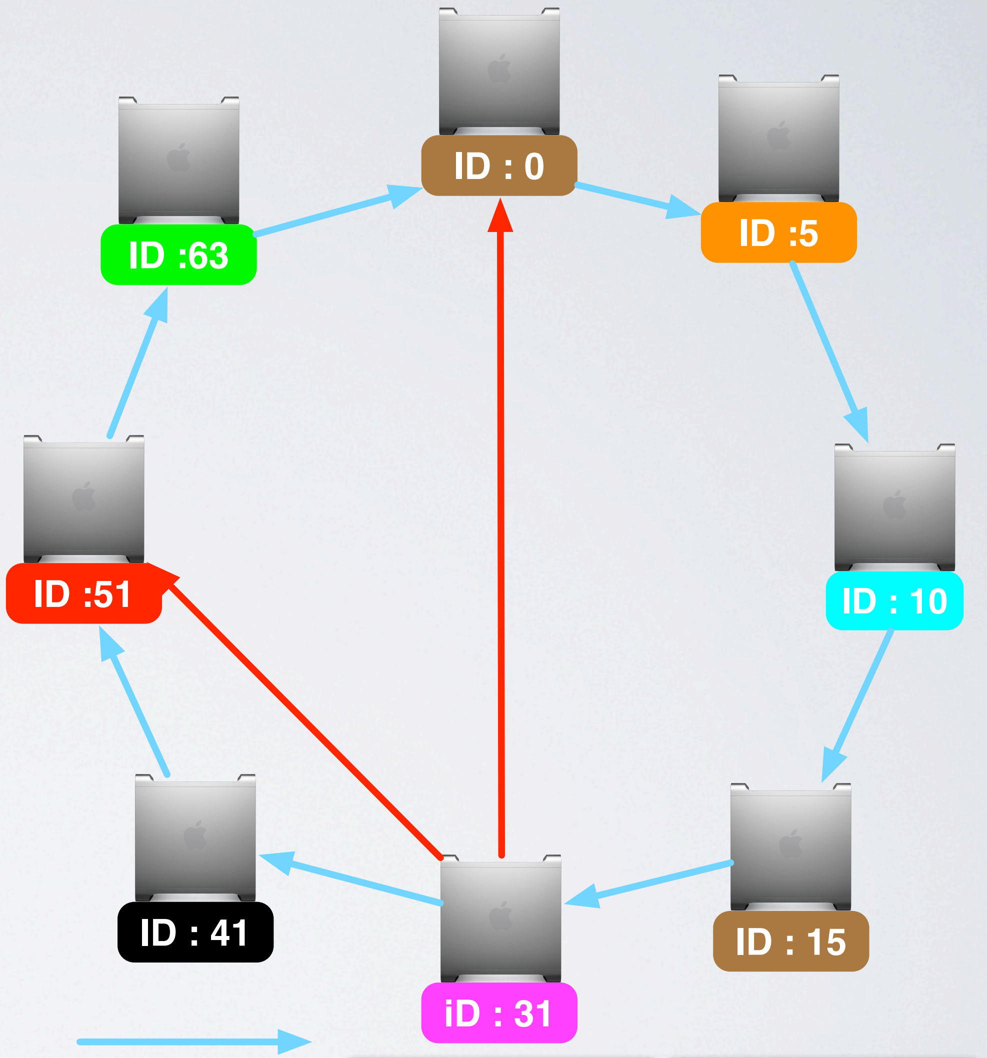
ID	IP
15	133.13.…

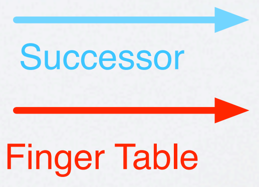
Successor Table

ID	IP
15	133.13.…

Successor + Finger Table

- Successor guarantees reachability
- Finger Table provide a quick routing
- Routing order is O(log N)



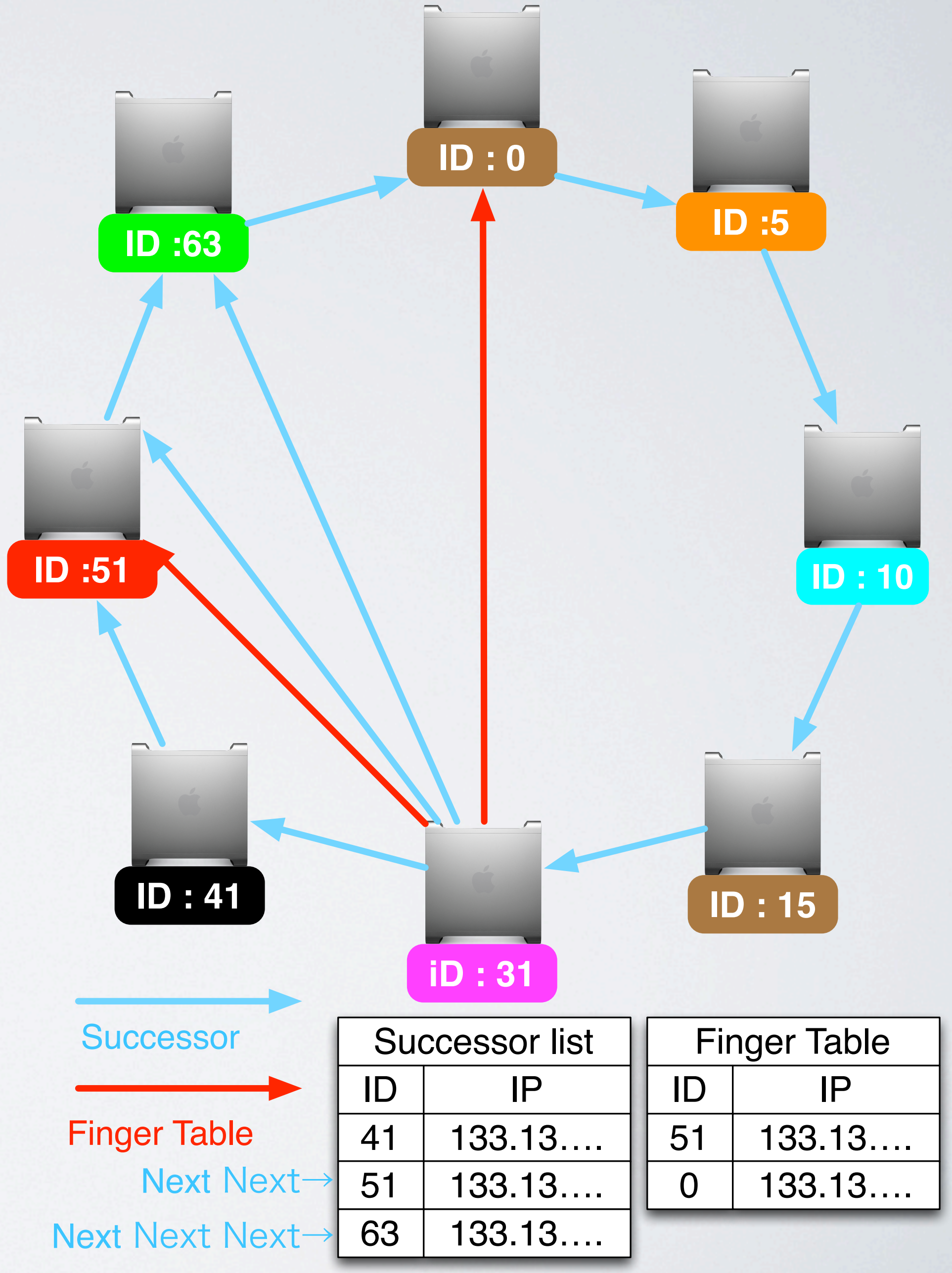


Successor Table	
ID	IP
41	133.13.…

Finger Table	
ID	IP
51	133.13.…
0	133.13.…

SuccessorList + FingerTable

- SuccessorList provide a failure resistant and reachability
- FingerTable provide a quick routing

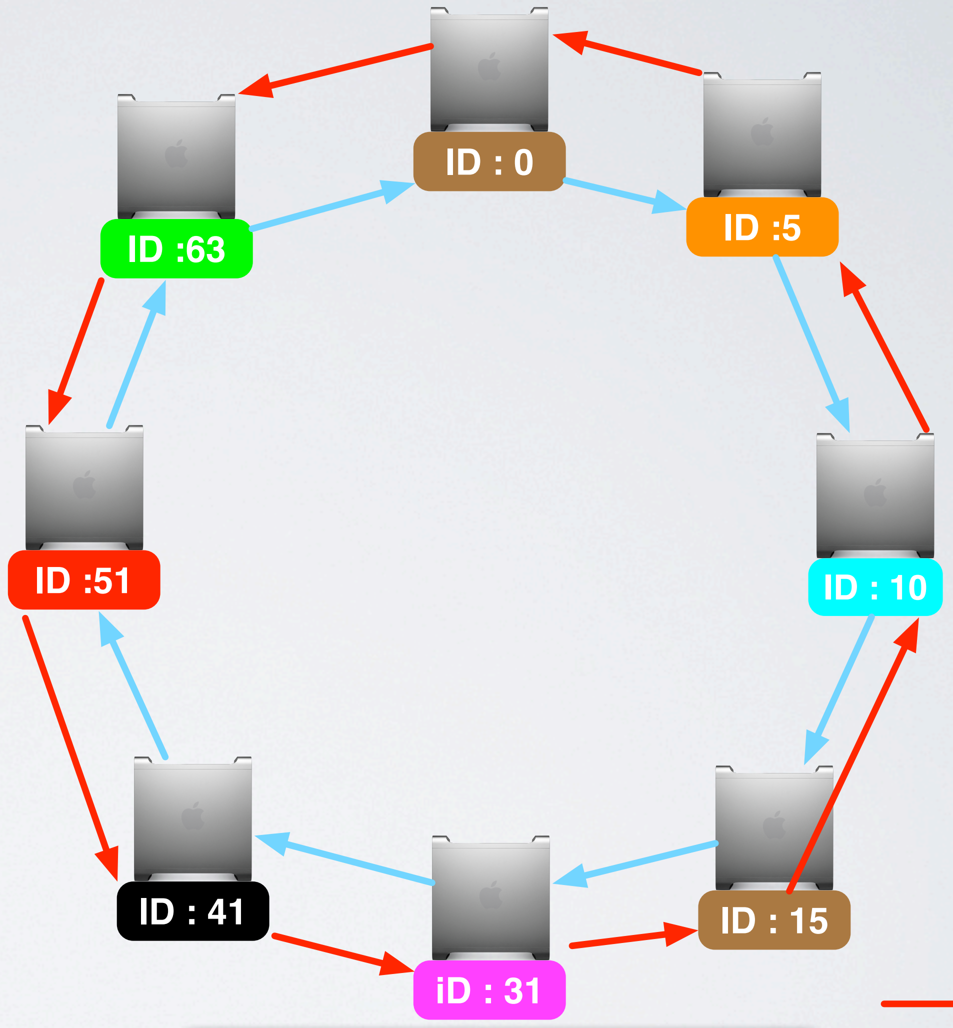


Successor list	
ID	IP
41	133.13….
51	133.13….
63	133.13….

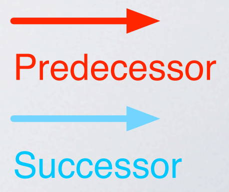
Finger Table	
ID	IP
51	133.13….
0	133.13….

Create Network : Predecessor

- Predecessor is a first machine in an anti clockwise direction
- Predecessor used Stabilize operation

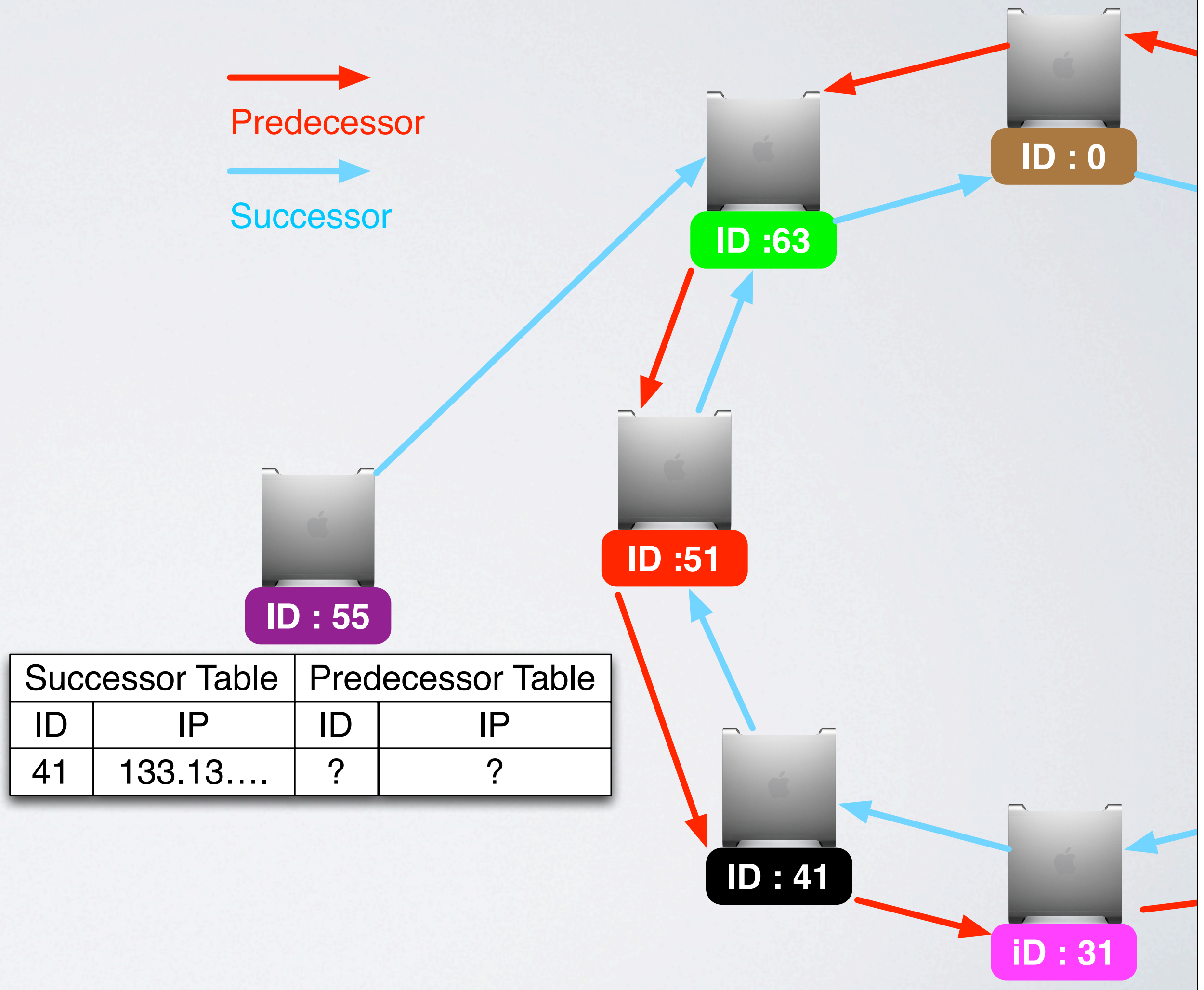


Successor Table		Predecessor Table	
ID	IP	ID	IP
41	133.13....	15	133.13....



Create Network : Join

- Join is operation which add node for Network
- New node send self hashed ID to any Network node
- Network returns "responsible Domain"
  - Update of the new nodes Successor Table
  - But, This is not yet available



Successor Table		Predecessor Table	
ID	IP	ID	IP
41	133.13....	?	?

Create Network : Stabilize

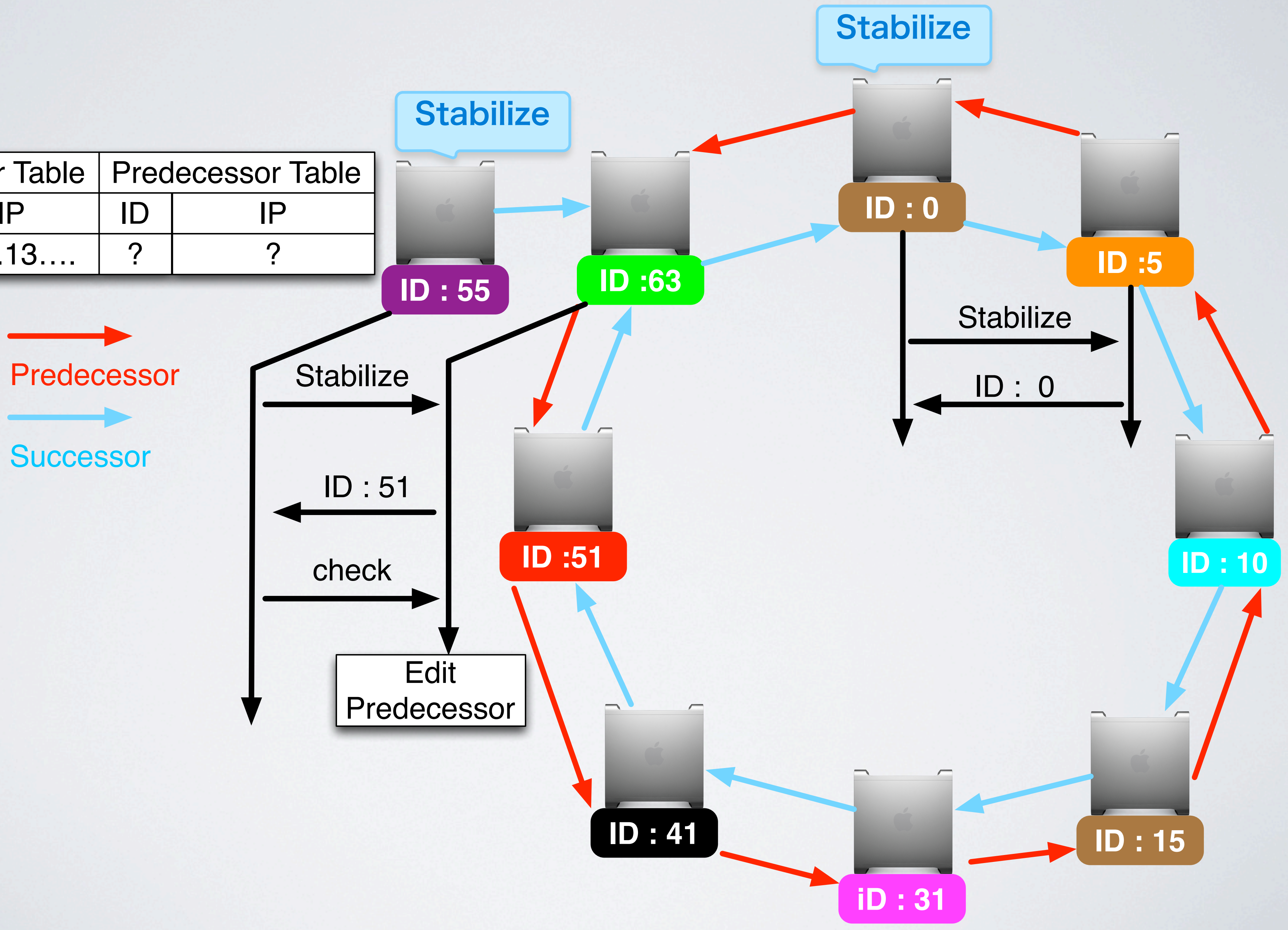
- Stabilize Operation is performed periodically
- stabilize is repeated many times
- Stabilize Operation consists of "For successor"and "For finger table"
- Stabilize Operation is guarantee of Availability when system is in a continuous state of change

Create Network : Stabilize

Stabilize operation for successor is check own successors predecessor

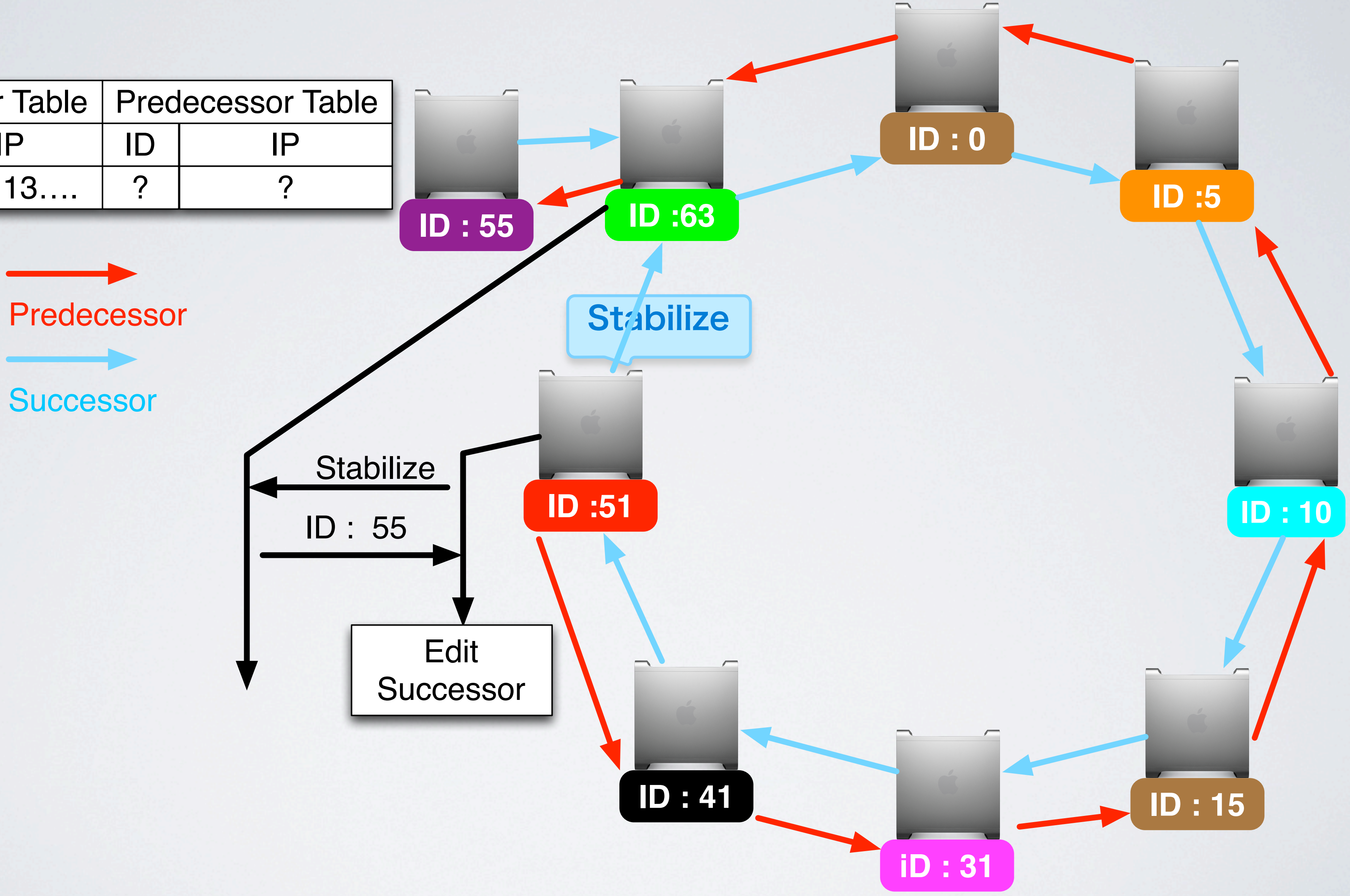
Create Network : Stabilize

Successor Table		Predecessor Table	
ID	IP	ID	IP
41	133.13….	?	?



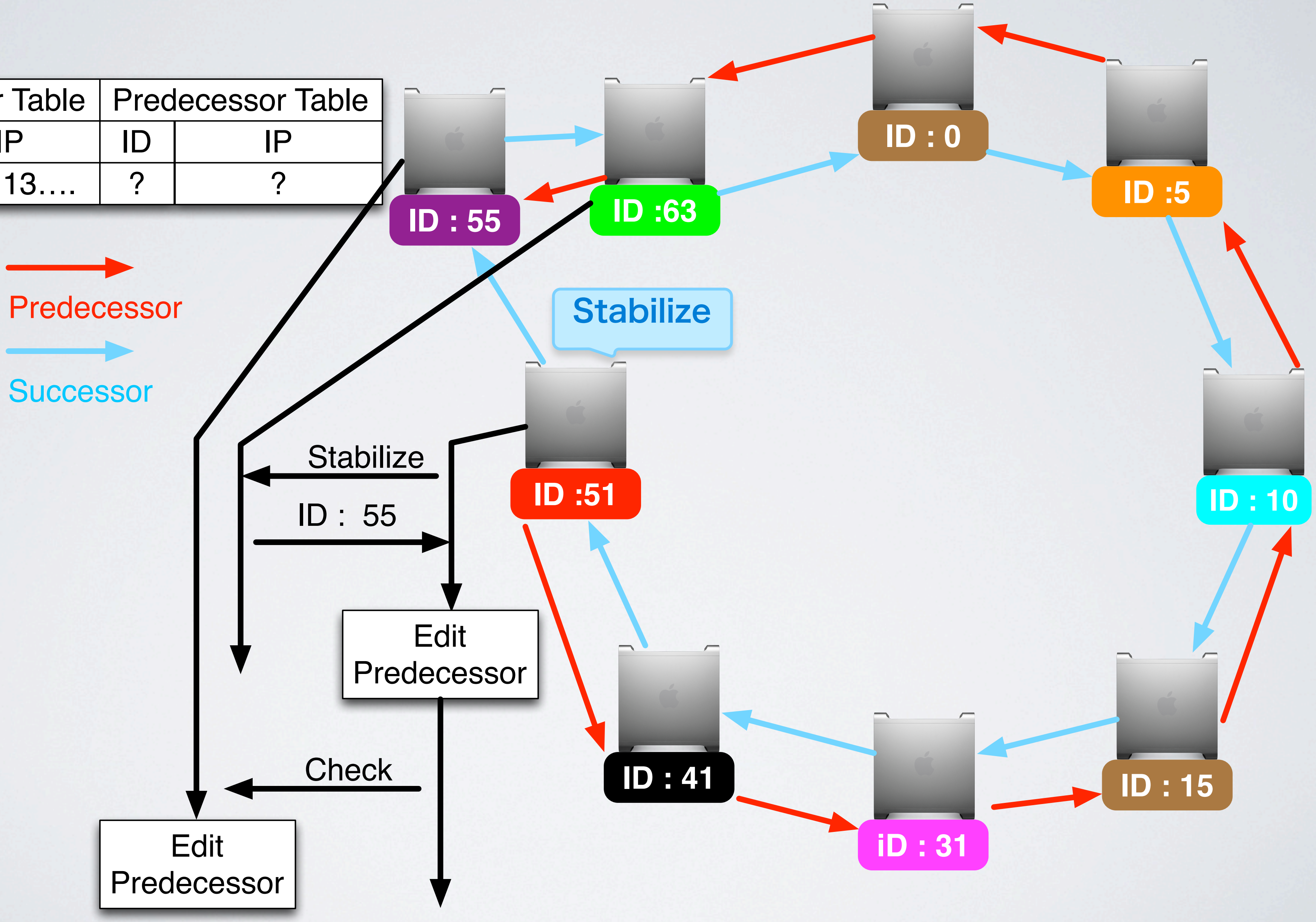
Create Network : Stabilize

Successor Table		Predecessor Table	
ID	IP	ID	IP
41	133.13….	?	?



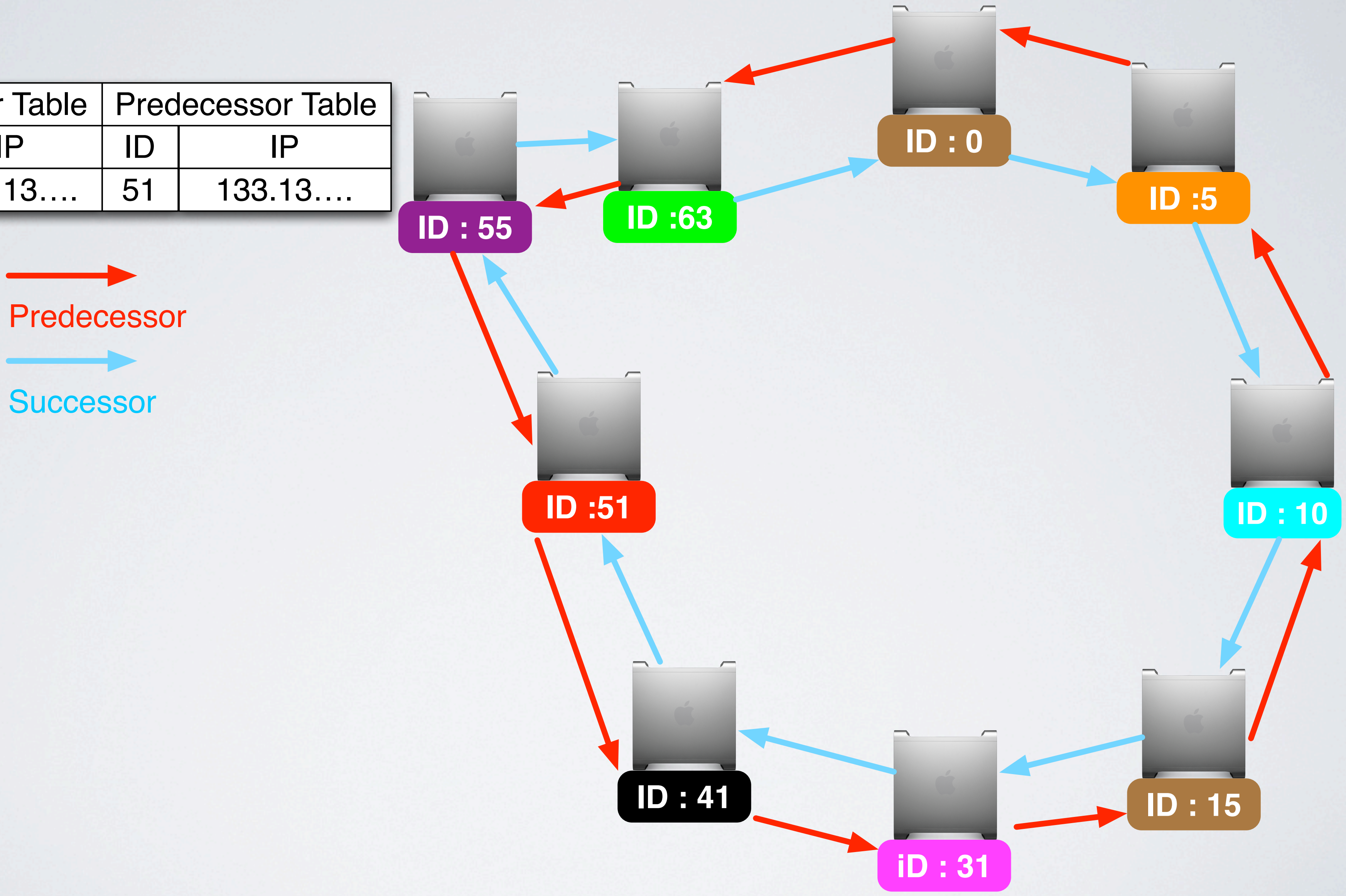
Create Network : Stabilize

Successor Table		Predecessor Table	
ID	IP	ID	IP
41	133.13….	?	?



Create Network : Stabilize

Successor Table		Predecessor Table	
ID	IP	ID	IP
41	133.13….	51	133.13….



Create Network : Stabilize

Stabilize operation for Finger table check randomly Finger table

When "i" is given a random

Finger Table	
Machine ID	**IP Adder**
Finger[1]	133.13.....
Finger[2]	:
:	:
FInger[2^i]	133.13.....
:	:
Finger[m]	133.13......

Check_Finger_Table_Number = 2^i -1

Conclusion

- hash provide Load balance and Flexble naming

- Code Decentralization

- SuccessorList + Finger Table provide Scalability O(log N)

- "Join and Stabilize" provides Availability,even if system is in a continuous state of change