GridGain

Data Distribution in Apache Ignite

Andrey Gura

03.12.2019

About me

Andrey Gura

- Software Engineer at GridGain
- Apache Ignite committer and PMC
- Email: agura@apache.org

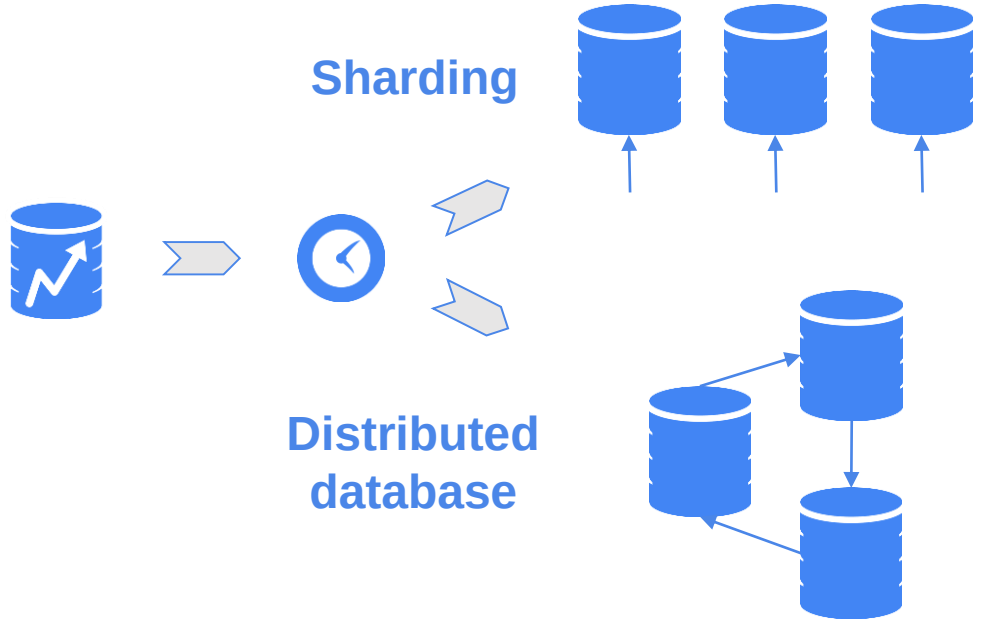
Contents

- Problem definition
- Possible solutions
- Apache Ignite

Problem Definition

Horizontal scalability

- High load (read/write scalability)
- Big data



Problem definition

Data distribution and requirements:

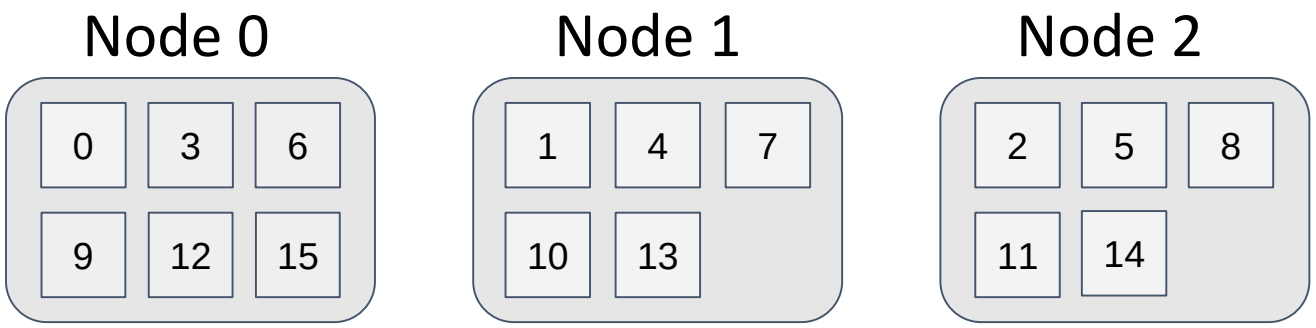
- **Distributed *k*-agreement**
  - Clients need to agree on which nodes objects are assigned to.
- **Load balancing**
  - Uniform distribution across the nodes.
- **Minimal disruption**
  - When a node fails only the objects mapped to that node need to be remapped.

Naive approach

h(K) mod N

where

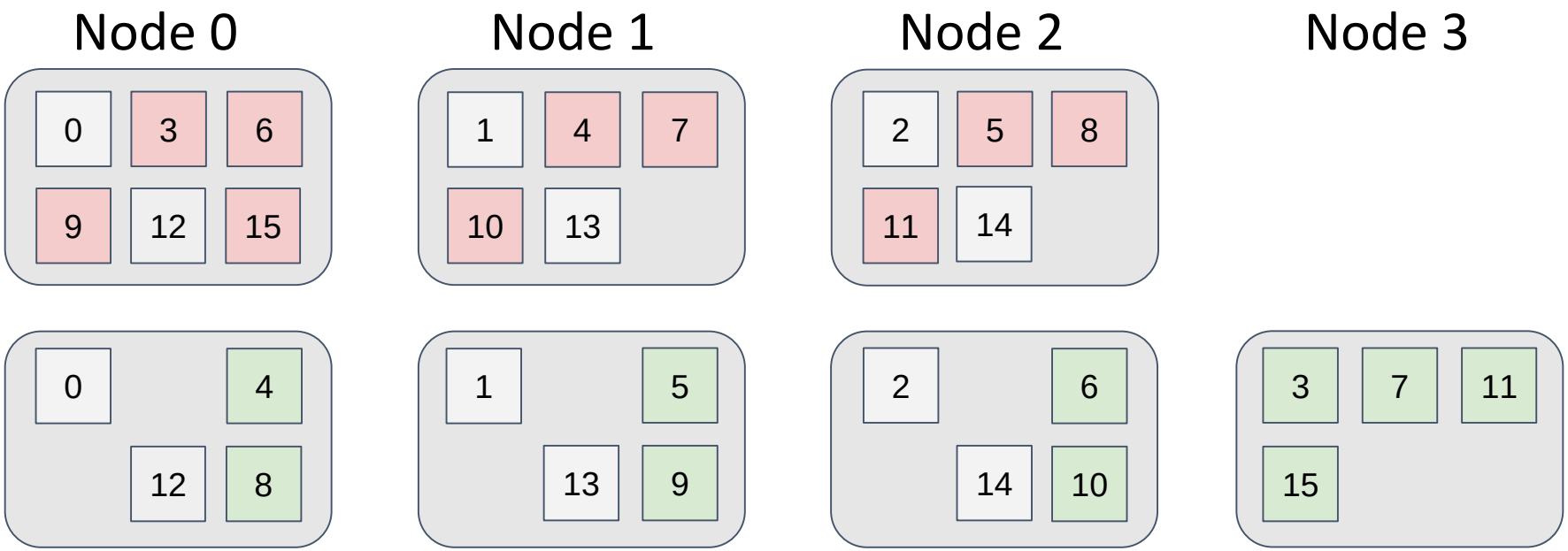
- *K* - object key
- *h* - hash function
- *N* - amount of nodes



Naive approach

h(K) mod N

Add new node and ...



GridGain

Solutions

Most popular

- Consistent hashing - 1997
  - https://en.wikipedia.org/wiki/Consistent_hashing
  - https://www.akamai.com/us/en/multimedia/documents/technical-publication/consistent-hashing-and-random-trees-distributed-caching-protocols-for-relieving-hot-spots-on-the-world-wide-web-technical-publication.pdf
  - http://theory.stanford.edu/~tim/s16/l/l1.pdf

- Rendezvous hashing (highest random weight) - 1996
  - https://en.wikipedia.org/wiki/Rendezvous_hashing
  - http://www.eecs.umich.edu/techreports/cse/96/CSE-TR-316-96.pdf

Consistent hashing

Map nodes and keys to the same keyspace

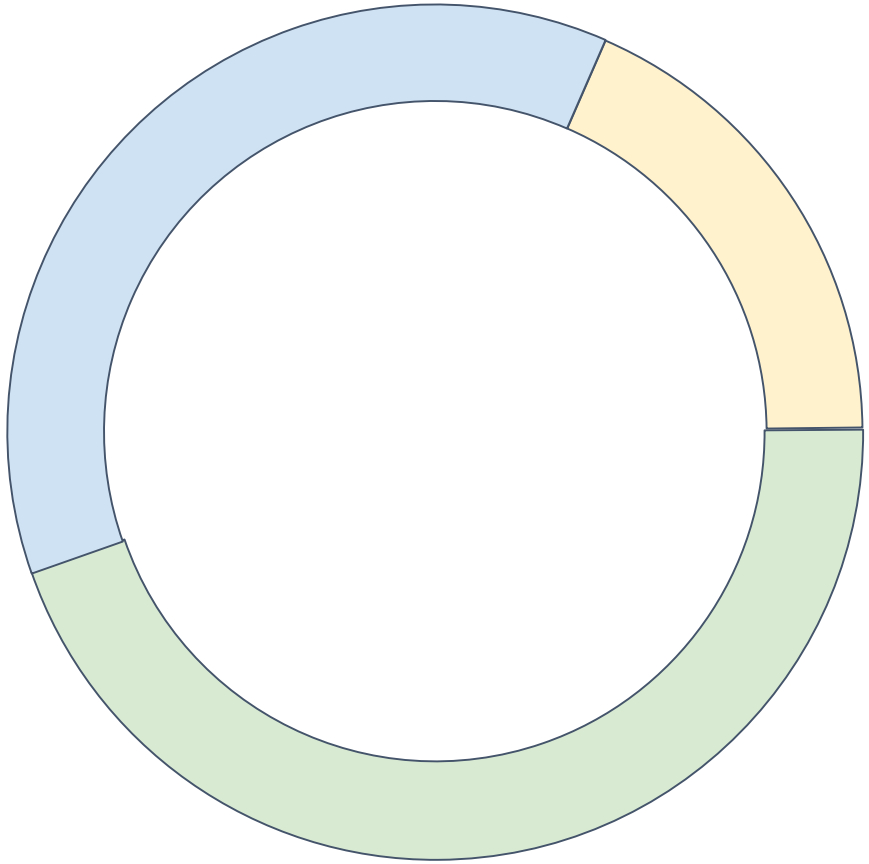
- For each node determine a random point on the keyspace - token
- For each key determine a random point on the keyspace
- Lookup the node for the key as the closest (direction is fixed) token on the keyspace

How to map nodes and keys to the keyspace?

- Use hash function
- Hash function must be the same for keys and nodes

Consistent hashing

Distribution example for 3 nodes on keyspace

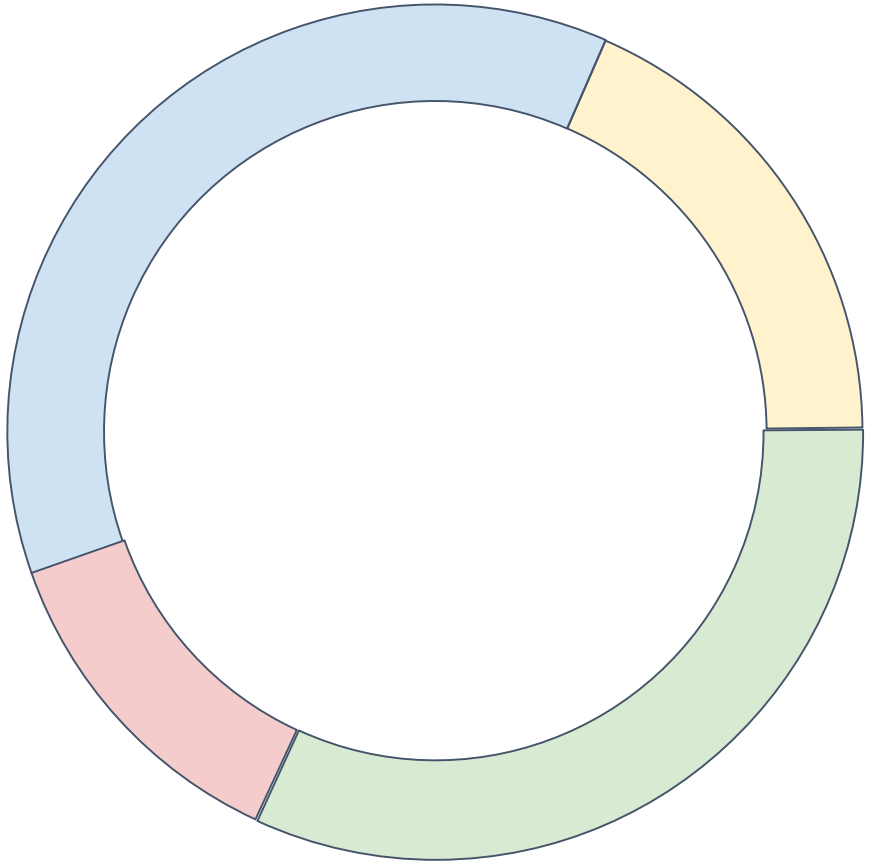


- Node 0
- Node 1
- Node 2

Big variance -
too uneven distribution

Consistent hashing

Distribution example for 4 nodes on keyspace

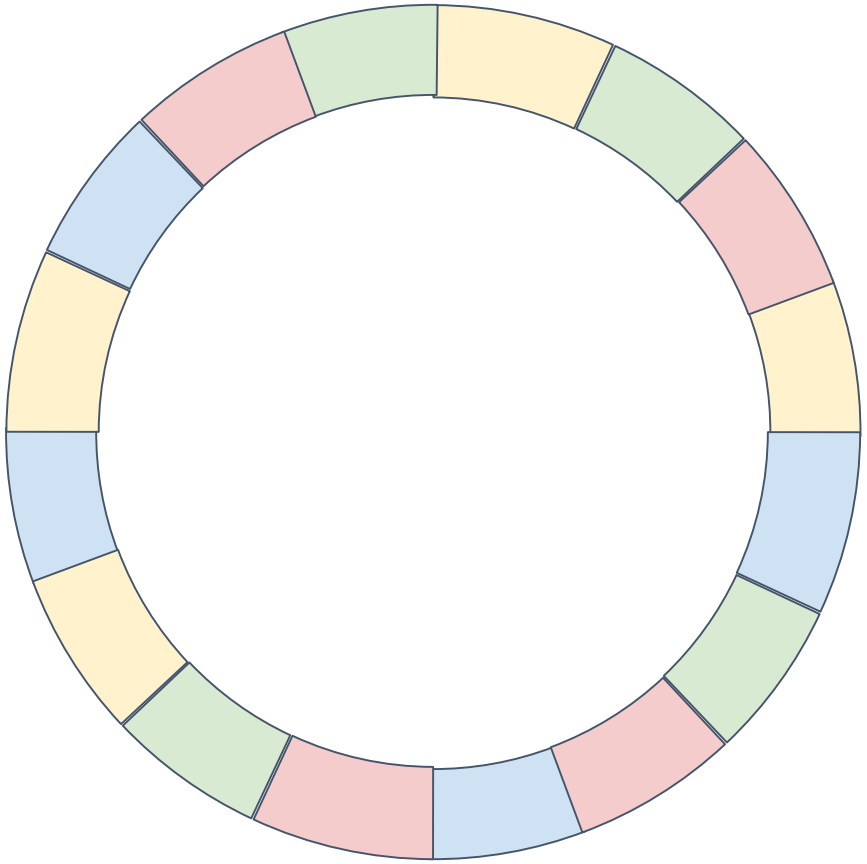


- Node 0
- Node 1
- Node 2
- Node 3

Big variance -
too uneven distribution

Consistent hashing

Distribution example for 4 nodes on keyspace with 4 tokens per node



- Node 0
- Node 1
- Node 2
- Node 3

Reduced variance -
almost even distribution

Rendezvous hashing (HRW)

Define relation order between nodes for each key

- Assign each node a weight for each key
- Assign the key to the node with highest weight

How to define relation order between nodes for the key?

- Use two place hash function for weight calculation - *h(key, node)*

Rendezvous hashing (HRW)

Distribution example: 4 keys, 3 nodes initially

K0	Node 2	Node 0	Node 1
K1	Node 1	Node 2	Node 0
K2	Node 0	Node 2	Node 1
K3	Node 2	Node 1	Node 0

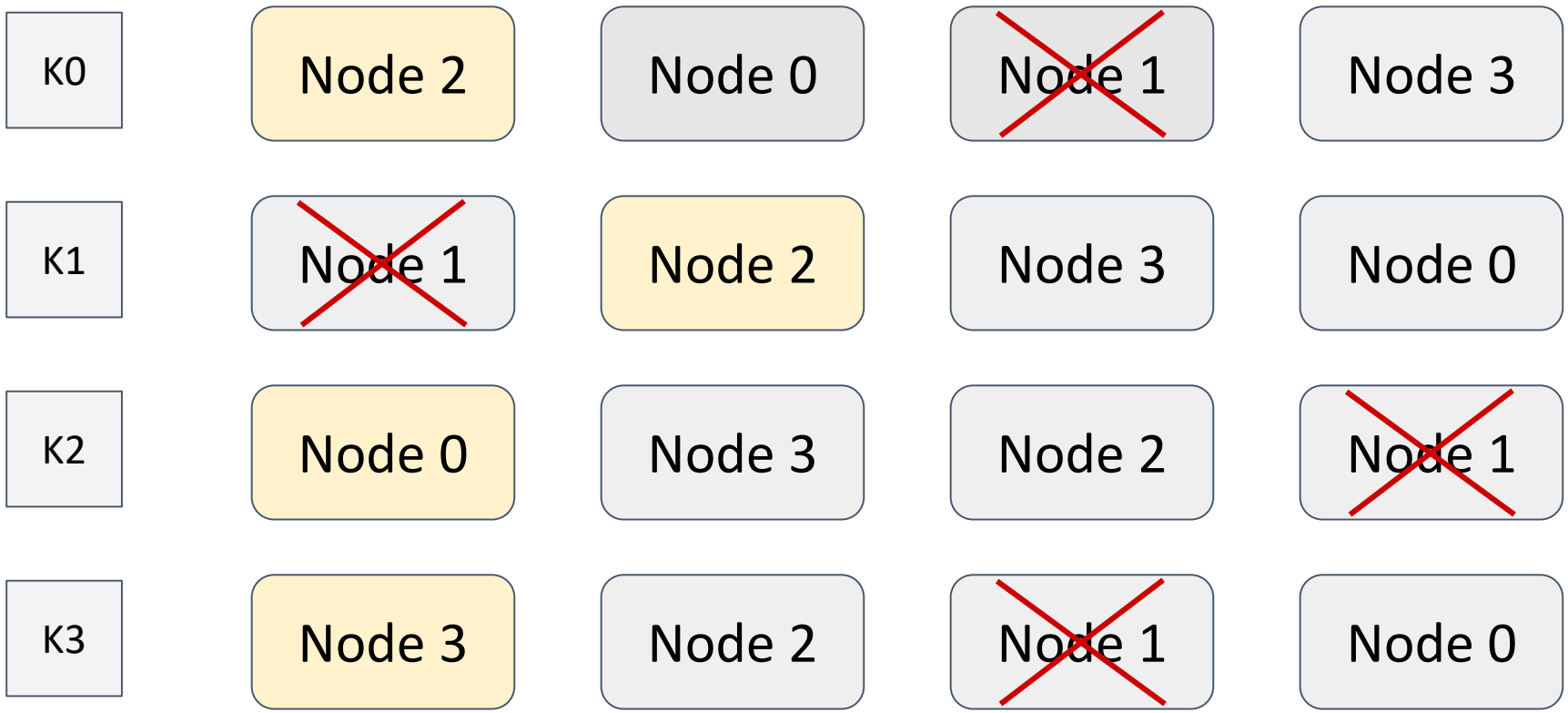
Rendezvous hashing (HRW)

Distribution example: 4 keys, new node added

K0	Node 2	Node 0	Node 1	Node 3
K1	Node 1	Node 2	Node 3	Node 0
K2	Node 0	Node 3	Node 2	Node 1
K3	Node 3	Node 2	Node 1	Node 0

Rendezvous hashing (HRW)

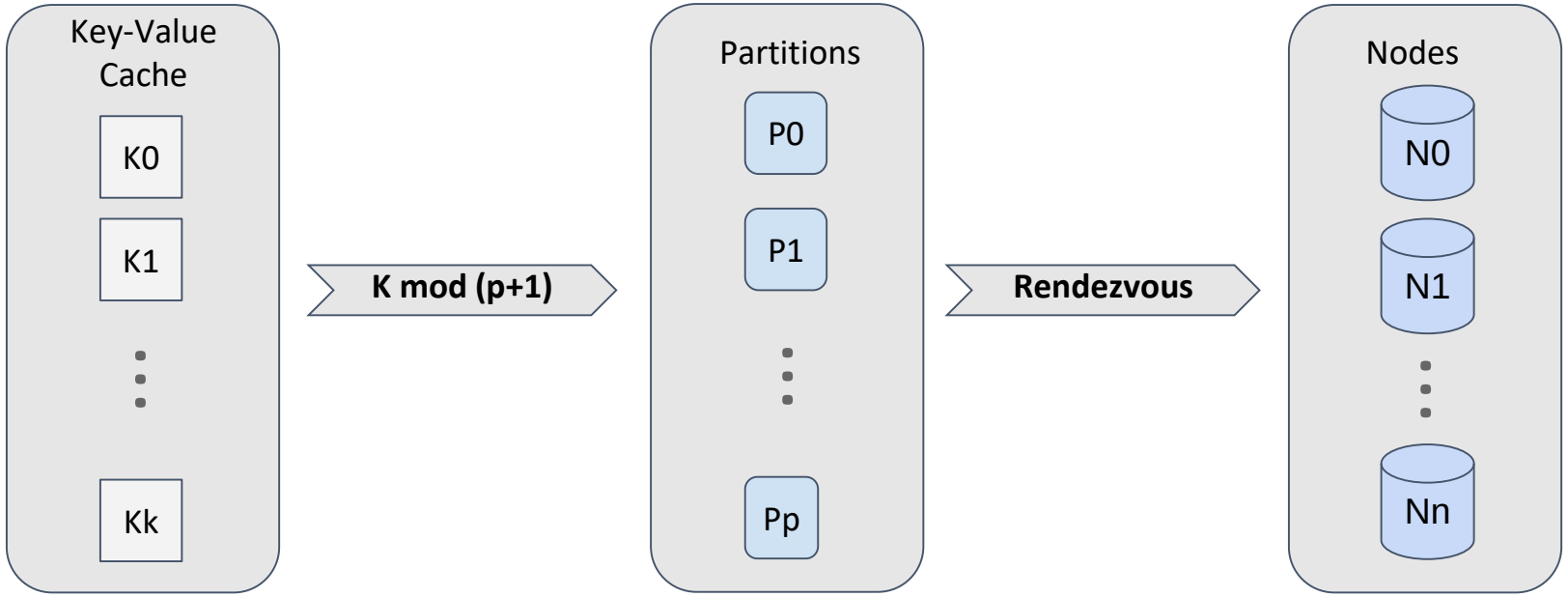
Distribution example: 4 keys, node 1 is removed



Apache Ignite

Data distribution and partitioning

- Partitioning is controlled by the affinity function
- The affinity function maps keys to partitions and partitions to nodes
- Partition (primary) can have several copies - backups

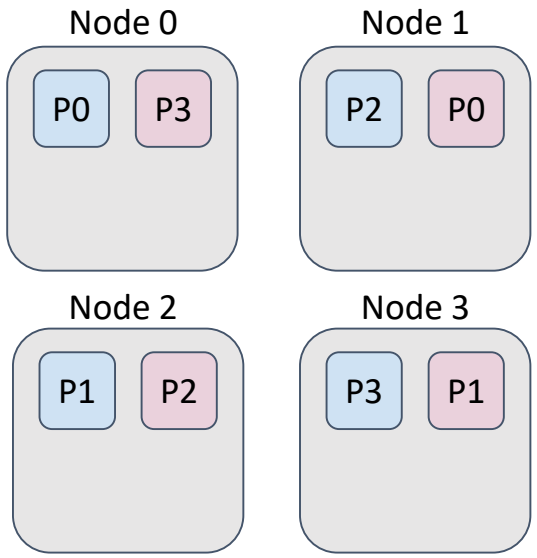


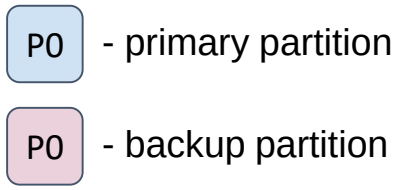
k > p > n

Apache Ignite

Partitioned cache

- Partitions (primary and backups) are evenly distributed between nodes
- Maximum performance



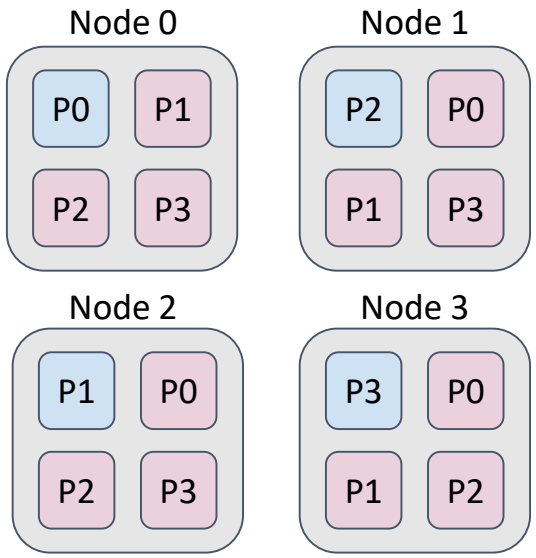


1 backup
1 node can be lost

Apache Ignite

Replicated cache

- Every partition is replicated to every node
- Maximum availability



P0 - primary partition

P0 - backup partition

N-1 backups

N-1 nodes can be lost

GridGain

Apache Ignite

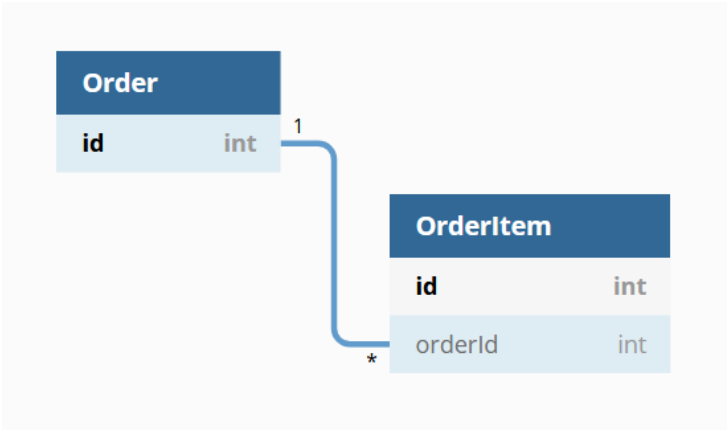
Affinity colocation

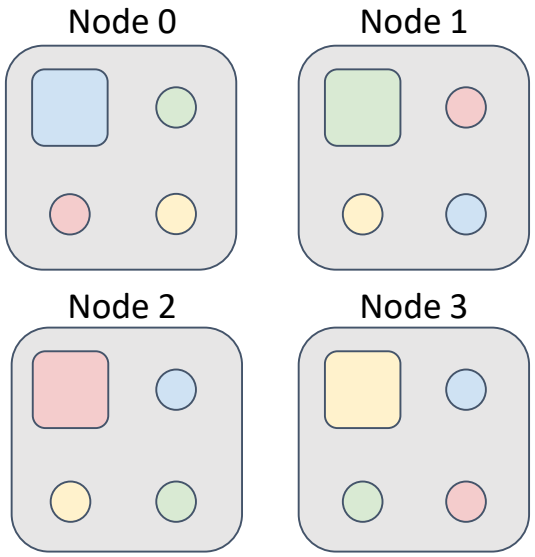
- Objects are assigned to partitions by the affinity function
- The objects that have the same affinity keys will be mapped to the same partitions
- By default primary key is affinity key
- But other object field could be used as affinity key (e.g. foreign key)
- Minimize data transfer between nodes
- Significantly reduce the computational task or query execution time

Apache Ignite

Affinity colocation

- Related entries are not colocated





- order

- order item

Order.id - primary and affinity key

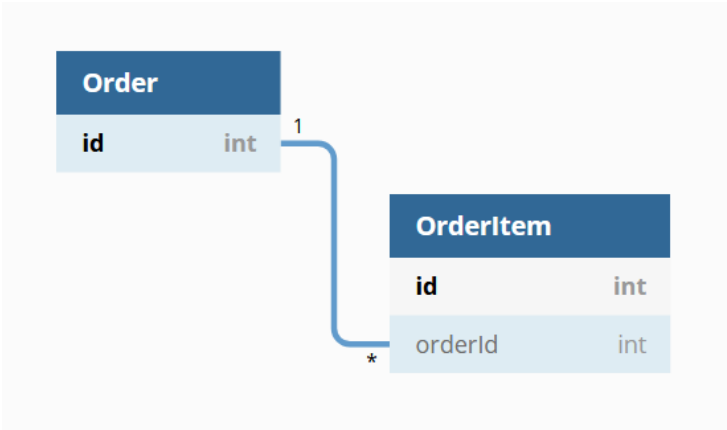
OrderItem.id - primary and affinity key

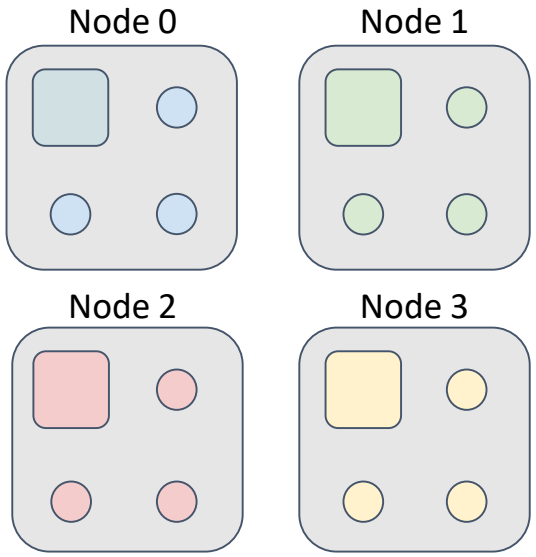
GridGain

Apache Ignite

Affinity colocation

- Related entries are colocated





- order

- order item

Order.Id - primary and affinity key

OrderItem.Id - primary key

OrderItem.OrderId - affinity key

GridGain

Rendezvous affinity function

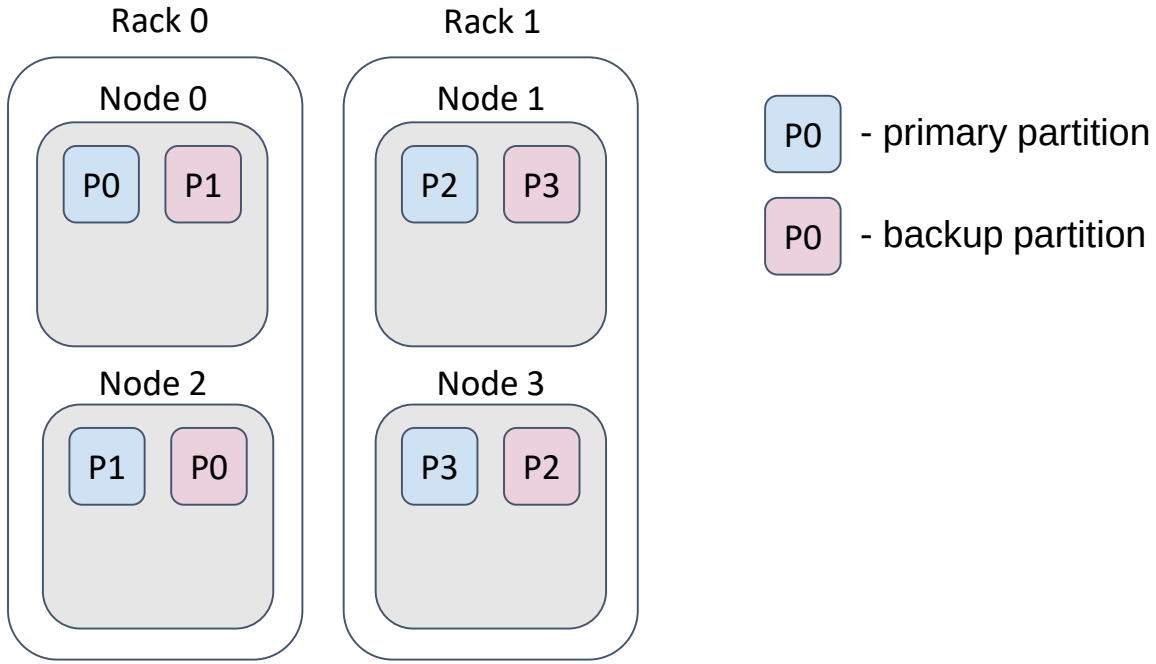
Configuration and input parameters

- Affinity function instance is cache configuration parameter
- Configuration (constant at runtime):
  - Number of partitions
  - Number of backups (from cache configuration)
  - Backup filter
  - Exclude neighbors flag
- Input:
  - Topology (list of nodes)

Rendezvous affinity function

Backup filter

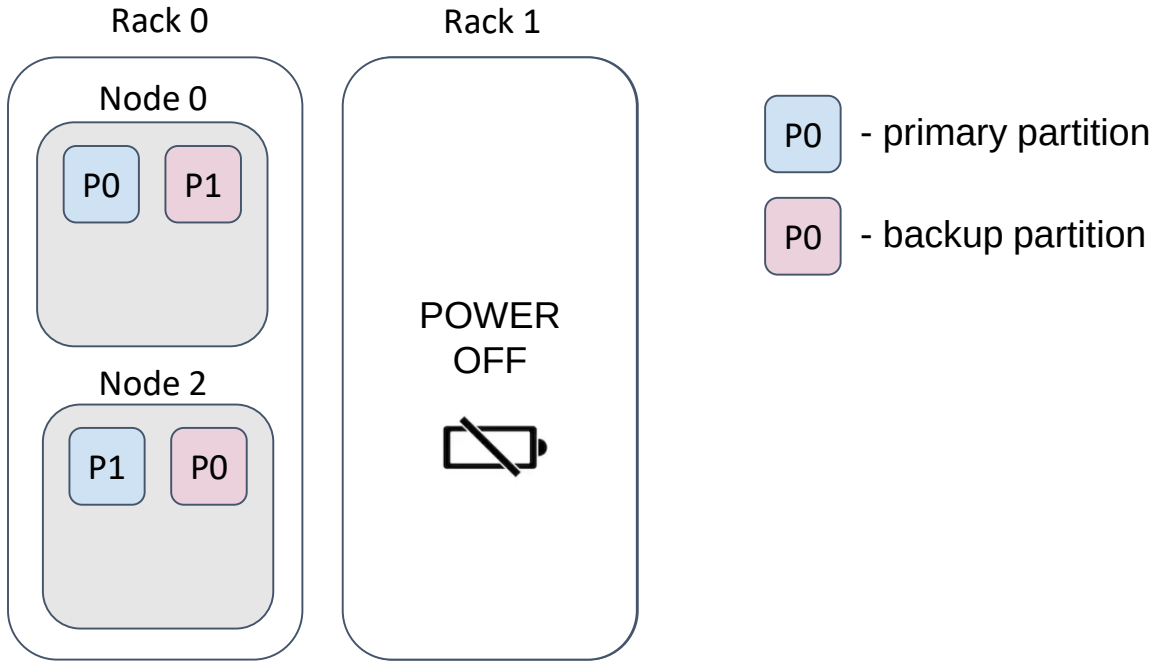
Let’s say the nodes are located in several racks.



Rendezvous affinity function

Backup filter

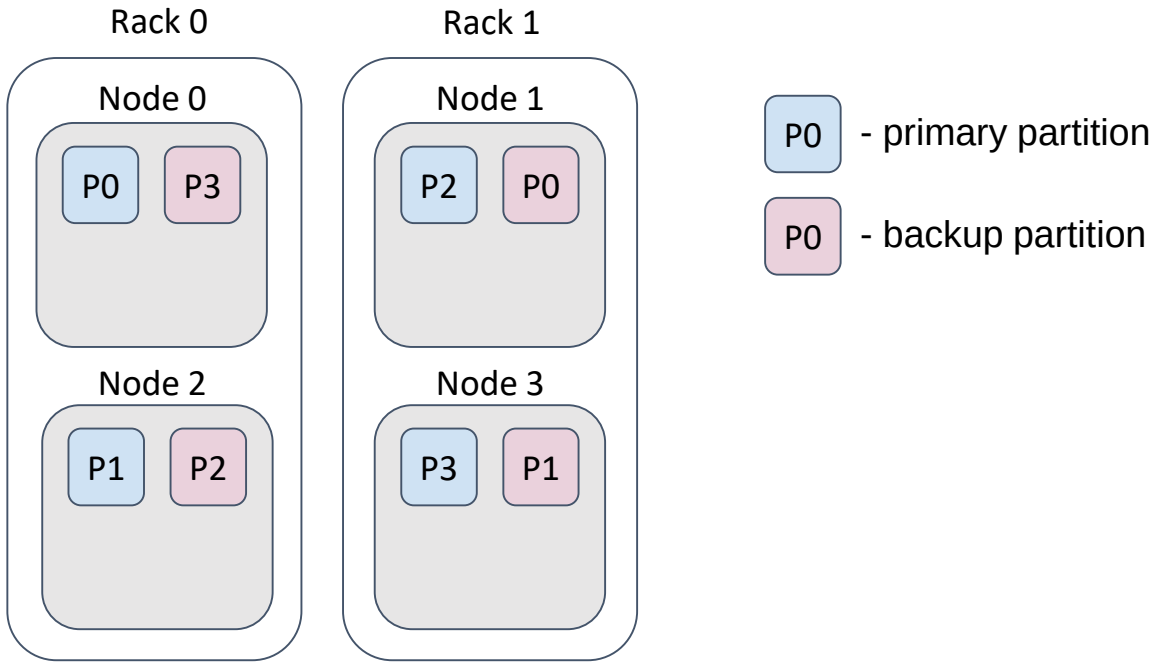
What if one of the racks will be turned off?
Data is lost or unavailable.



Rendezvous affinity function

Backup filter

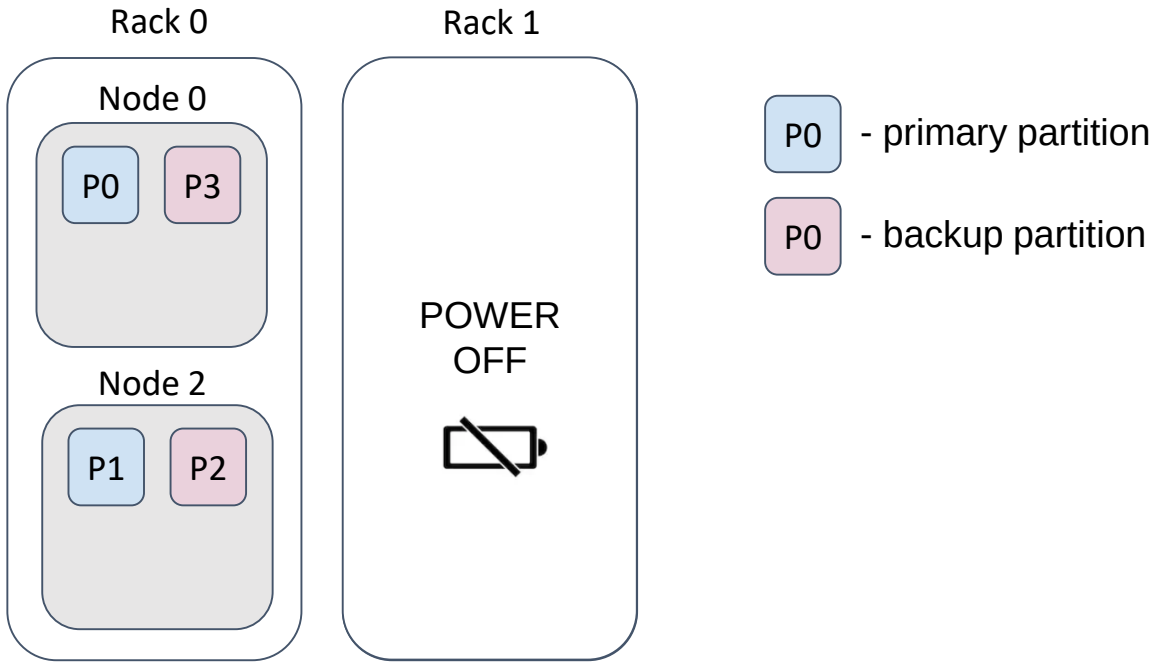
We can inform affinity function about nodes location using backup filter
Backup partitions should not be assigned to the node that is filtered out



Rendezvous affinity function

Backup filter

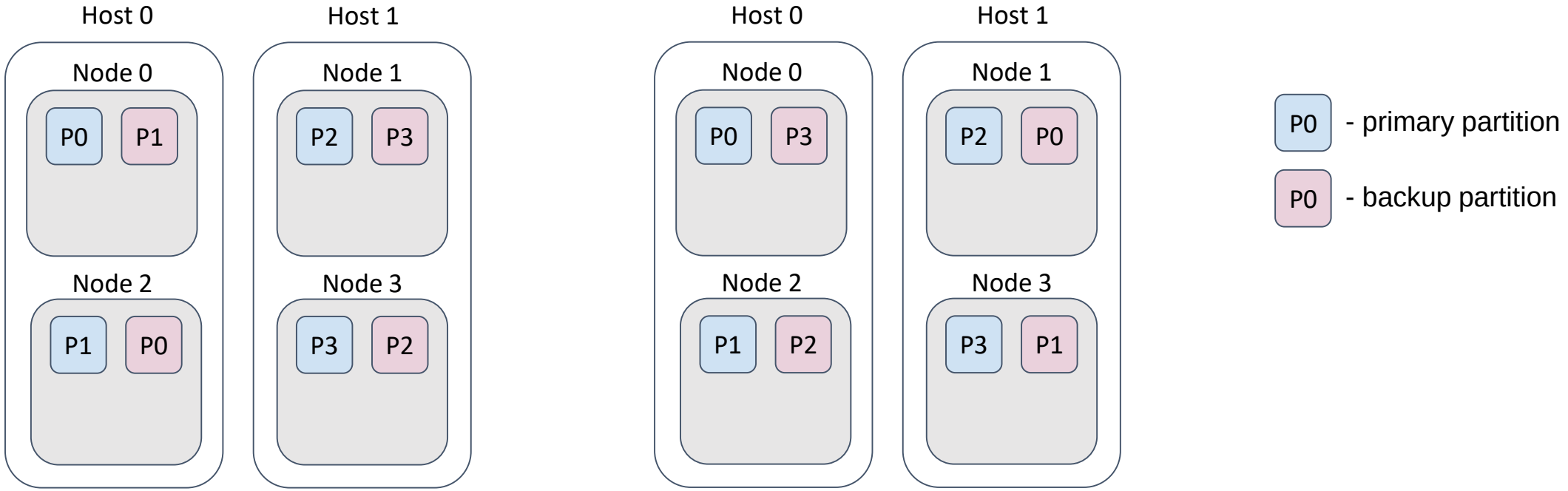
Rack 1 is turned off again, but ...
All partitions are available and no data loss.



Rendezvous affinity function

Exclude neighbors flag

Very similar to backup filter.
Backup partition should not be assigned to the node on the same host.



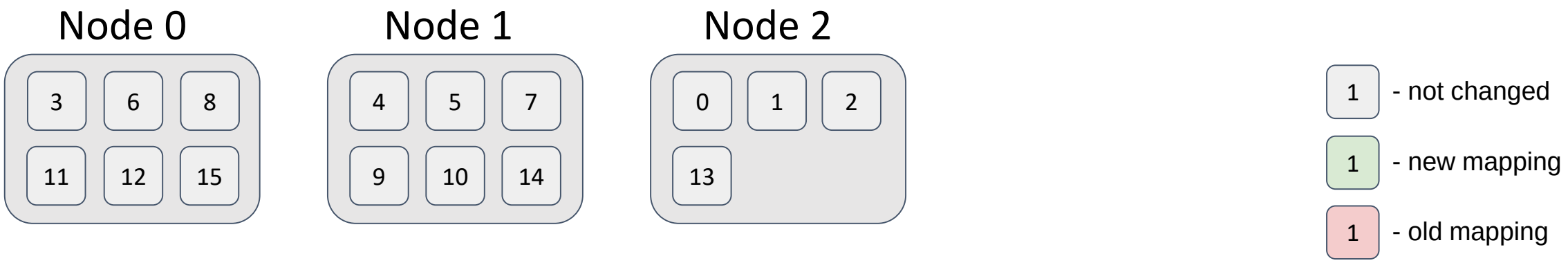
Rendezvous affinity example

Parameters

- Partitions - 16
- Backups - 0
- Nodes - 3-4

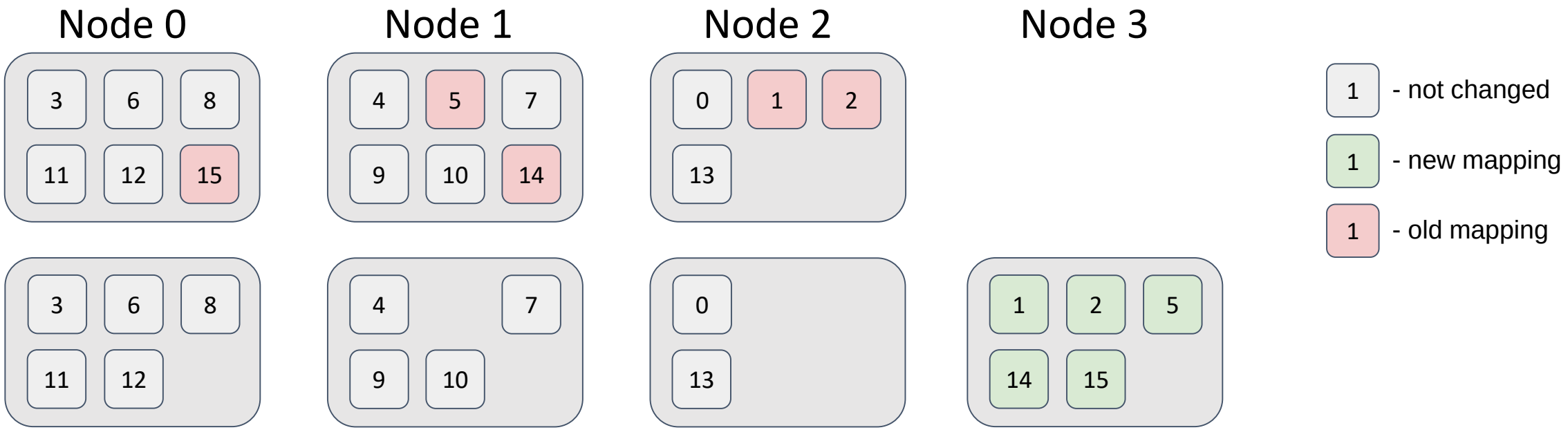
Rendezvous affinity example

Distribution for initial topology of 3 nodes



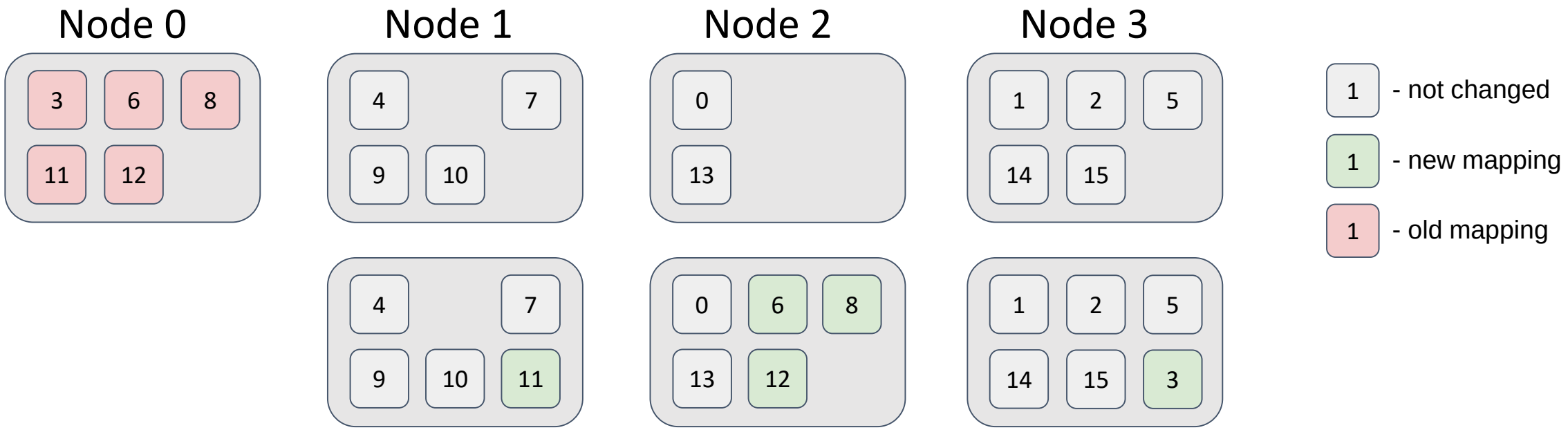
Rendezvous affinity example

New “Node 3” joined



Rendezvous affinity example

“Node 0” failed



GridGain

Apache Ignite community

Welcome to contribute!

- Project page
  - http://ignite.apache.org
- Users mailing list and forum
  - user@ignite.apache.org
  - http://apache-ignite-users.70518.x6.nabble.com
- Developers mailing list and forum
  - dev@ignite.apache.org
  - http://apache-ignite-developers.2346864.n4.nabble.com
- GitHub
  - https://github.com/apache/ignite