Consistent hashing

to implement hash partitioning we need to define…

- boundaries for each shard (a range of hashes each shard owns)
- where each shard lives (which physical server each shard lives on)
- how to rebalance shards (pick and implement a rebalancing strategy)

consistent hashing

Consistent hashing

hash functions

hash function (key) $\longrightarrow$ *integer* $[0, 2^{31})$

1. How to split this range into smaller hash value intervals (shards).
2. How to assign each interval (shard) to a server.

👎

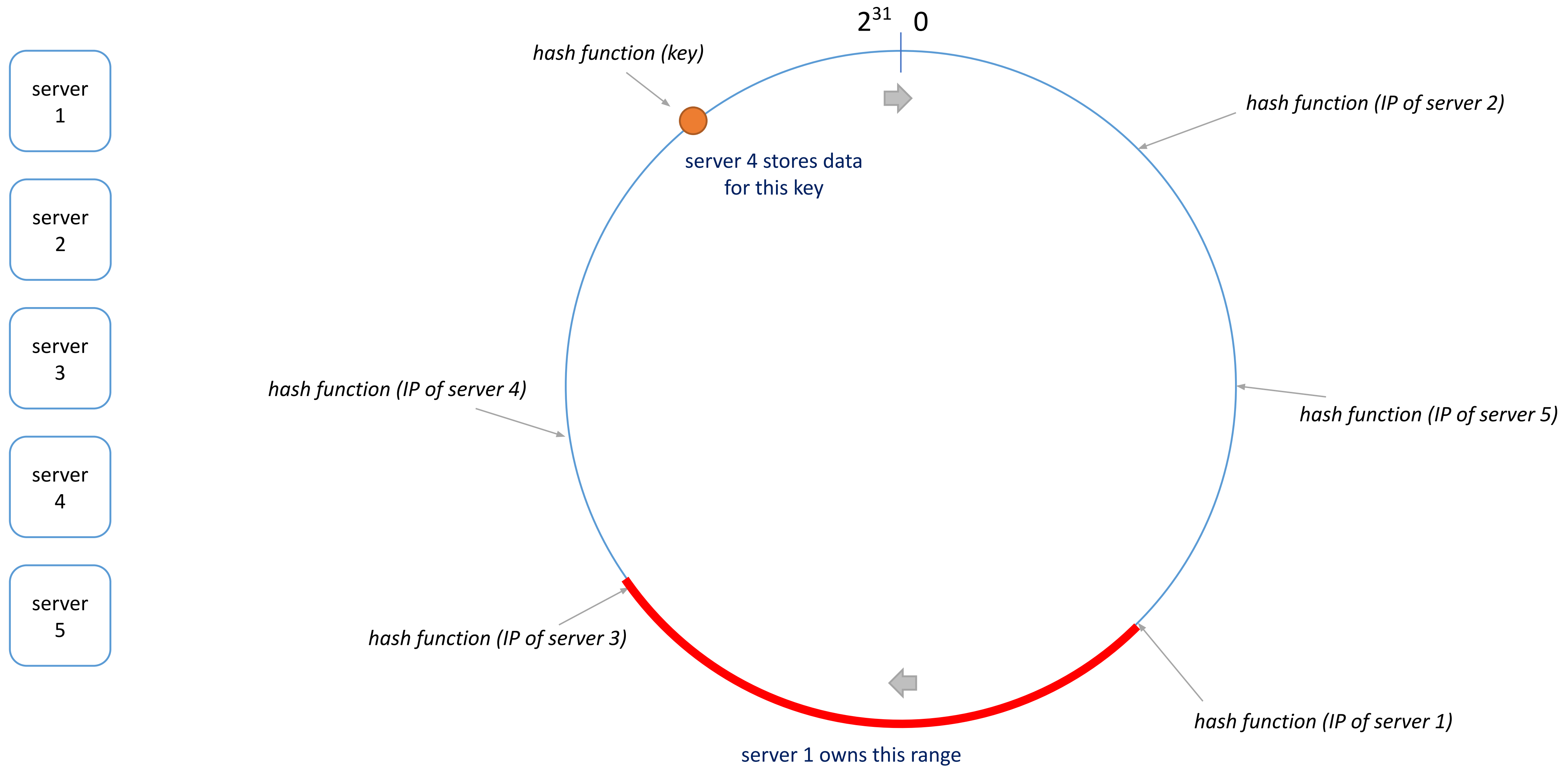
cryptographic hash functions
(e.g. SHA)

👍

non-cryptographic hash functions
(e.g. MurmurHash)

Consistent hashing

consistent hashing ring


server 1
server 2
server 3
server 4
server 5
2^{31}
0
hash function (key)
server 4 stores data for this key
hash function (IP of server 2)
hash function (IP of server 4)
hash function (IP of server 5)
hash function (IP of server 3)
hash function (IP of server 1)
server 1 owns this range

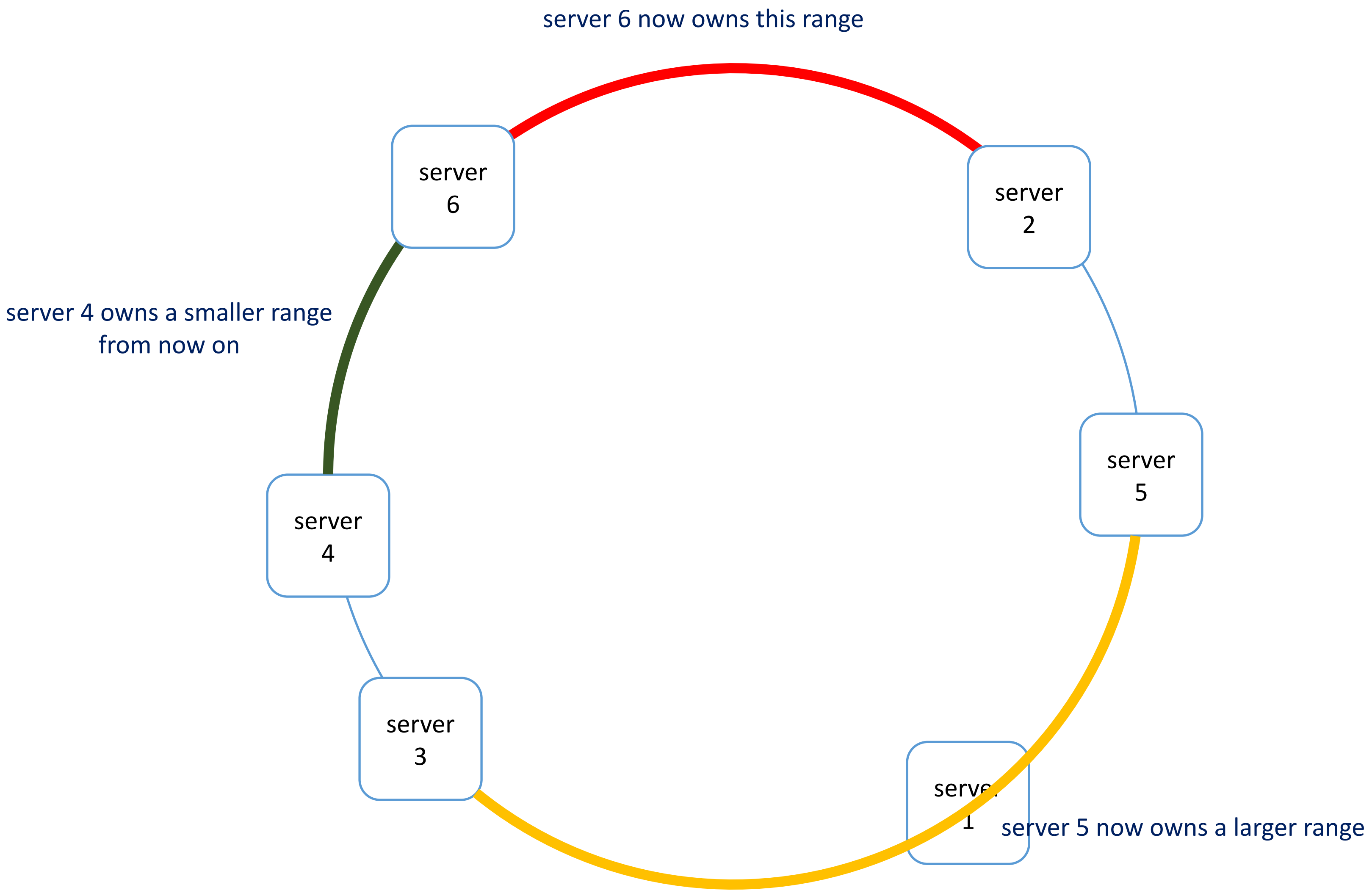
Consistent hashing

implementation details

$$\begin{tikzcd}
\text{list of server identifiers (e.g. IP, hostname)} \arrow[r] & \text{apply the hash function} \arrow[r] & \text{get hash ranges each server (shard) owns} \arrow[r] & \text{sort the list of hash ranges} \arrow[d] \\
& \text{client forwards the request to the identified server} & \text{find the range (server) that the key belongs to using the binary search algorithm} \arrow[l] & \text{share this ordered list with clients} \arrow[l]
\end{tikzcd}$$

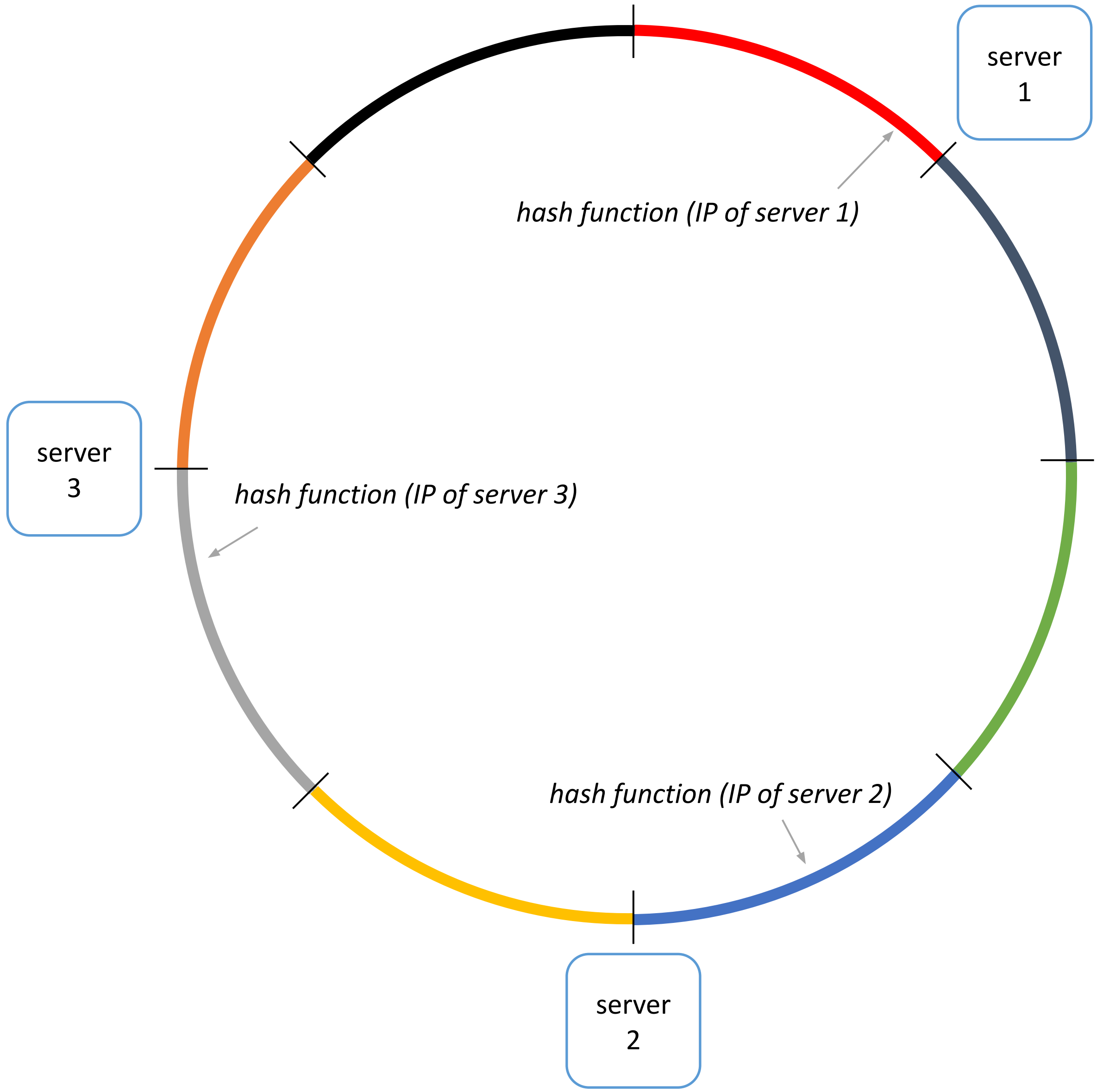
Consistent hashing

rebalancing


server 6 now owns this range
server 6
server 2
server 4 owns a smaller range from now on
server 5
server 4
server 3
server 1
server 5 now owns a larger range

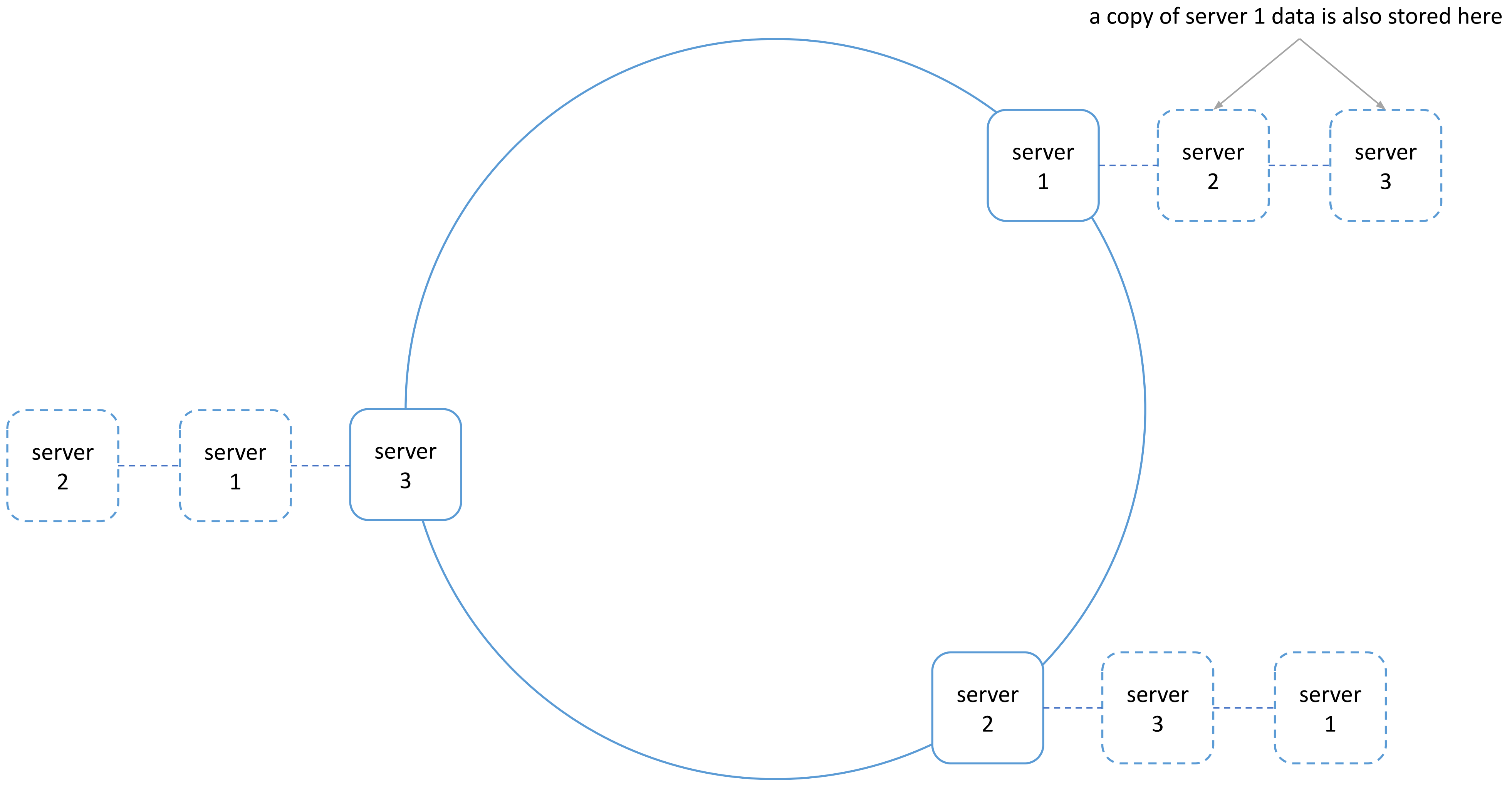
Consistent hashing

fixed number of shards


server 1
hash function (IP of server 1)
server 3
hash function (IP of server 3)
hash function (IP of server 2)
server 2

Consistent hashing

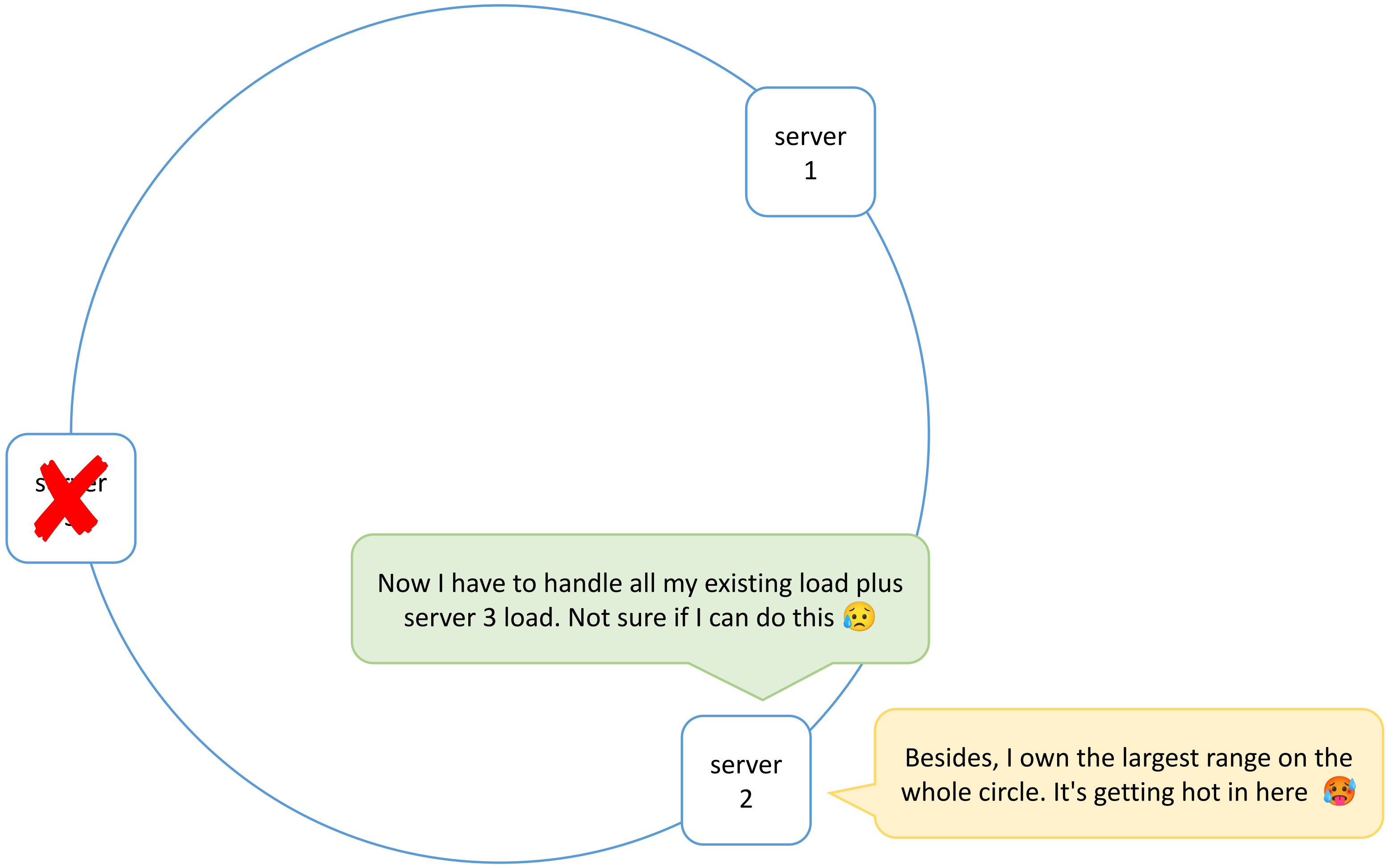
shard replicas



Consistent hashing

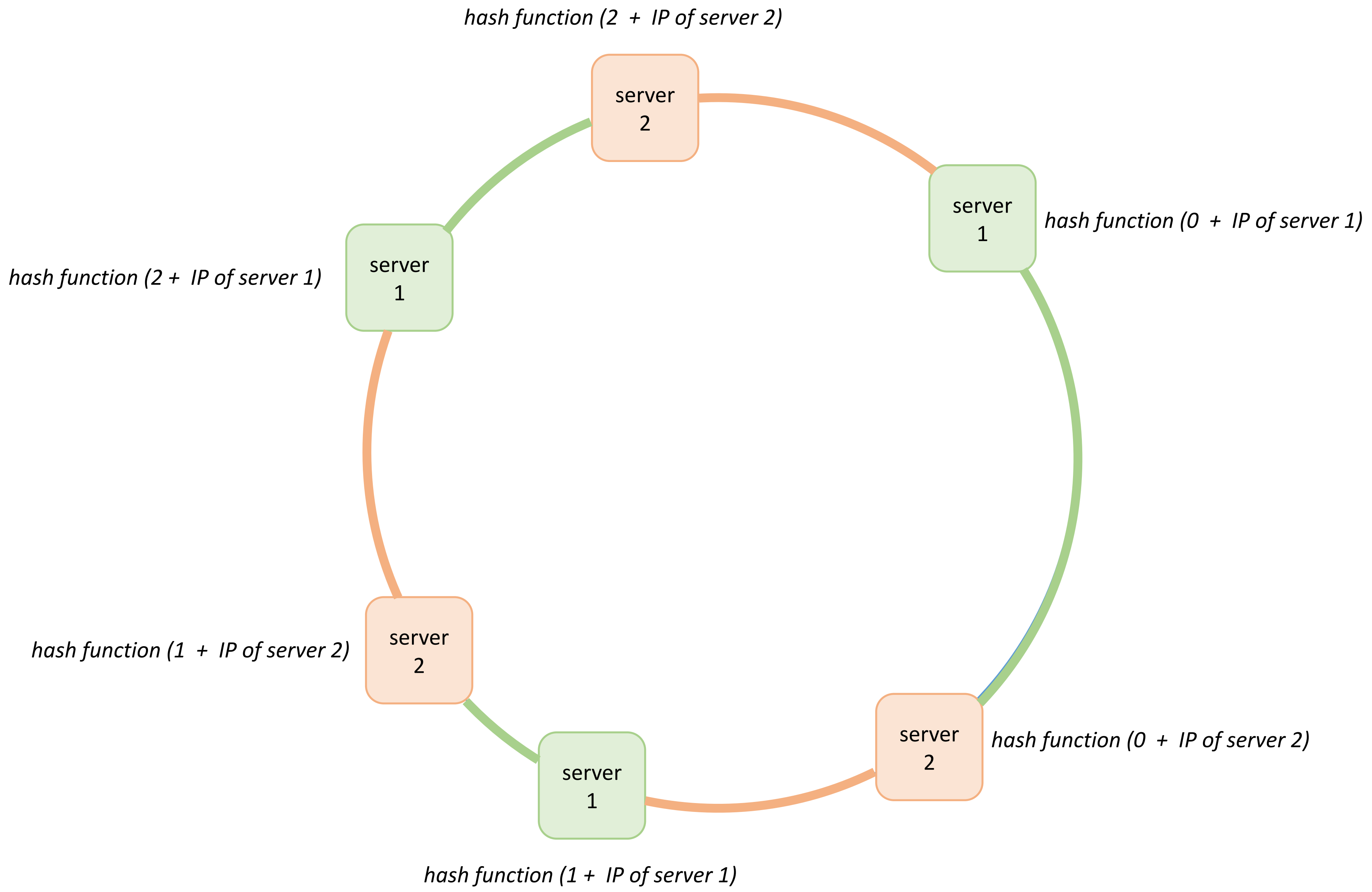
disadvantages

- domino effect
- servers do not split the circle evenly


server 1
server
server 2
Now I have to handle all my existing load plus server 3 load. Not sure if I can do this 😥
Besides, I own the largest range on the whole circle. It's getting hot in here 🥵

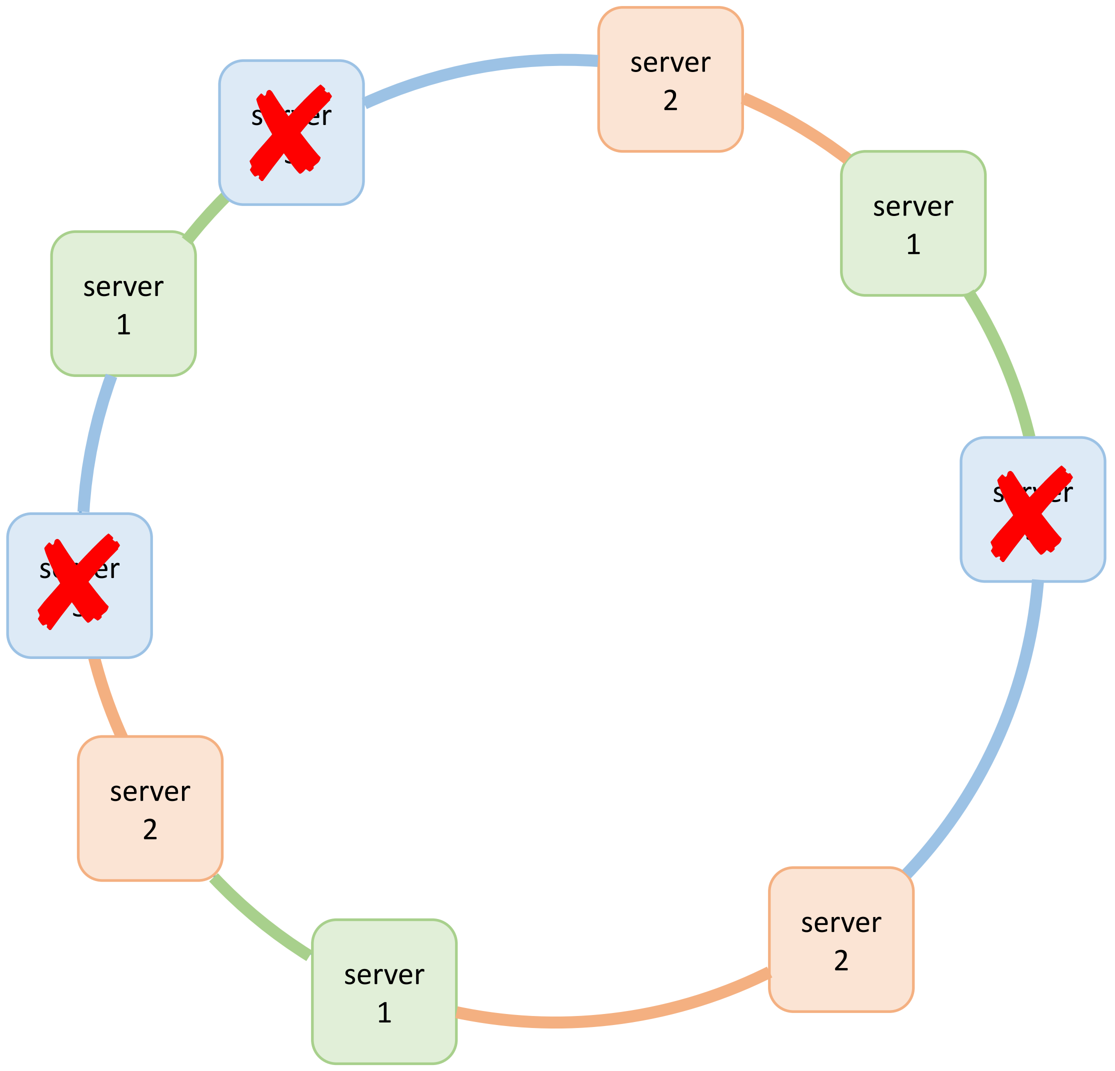
Consistent hashing

virtual nodes


hash function (2 + IP of server 2)
server 2
server 1
hash function (0 + IP of server 1)
hash function (2 + IP of server 1)
server 1
server 2
hash function (1 + IP of server 2)
server 2
hash function (0 + IP of server 2)
server 1
hash function (1 + IP of server 1)

Consistent hashing

virtual nodes



Consistent hashing

examples

- databases (Cassandra, Couchbase, Riak, Voldemort)
- distributed caches (client libraries, e.g. Ketama)
- content delivery networks (Akamai)
- network load balancers (Maglev)
- chat applications (Discord)
- messaging systems (RabbitMQ)