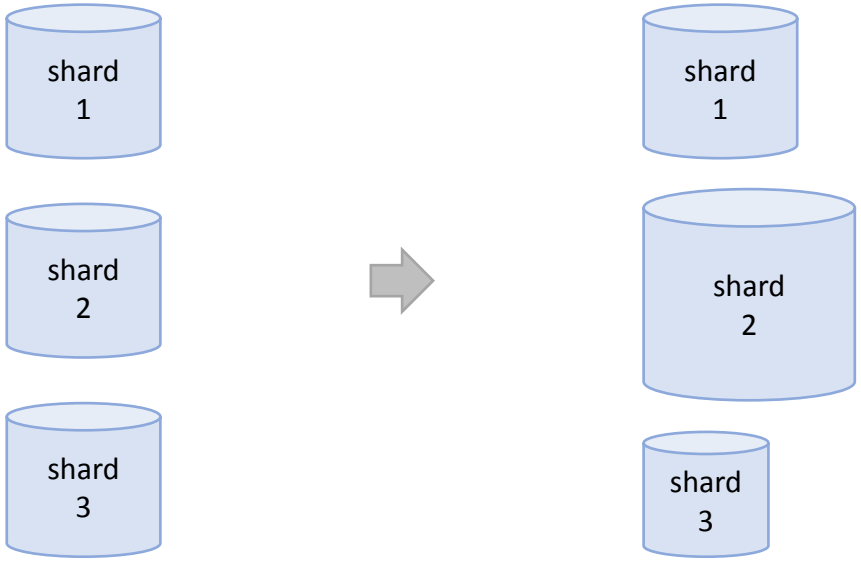
Rebalancing partitions

problem

uneven load (number of requests)
on shard servers



reasons

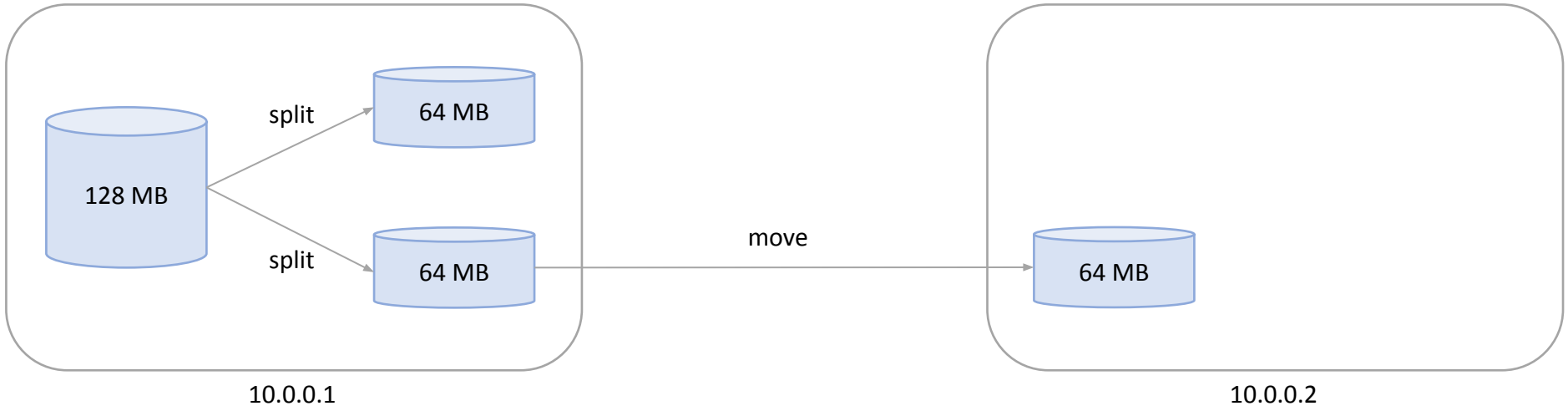
- uneven data distribution
- hot keys
- shard server failures

solution

rebalancing
(the process of moving the load from one shard server to another)

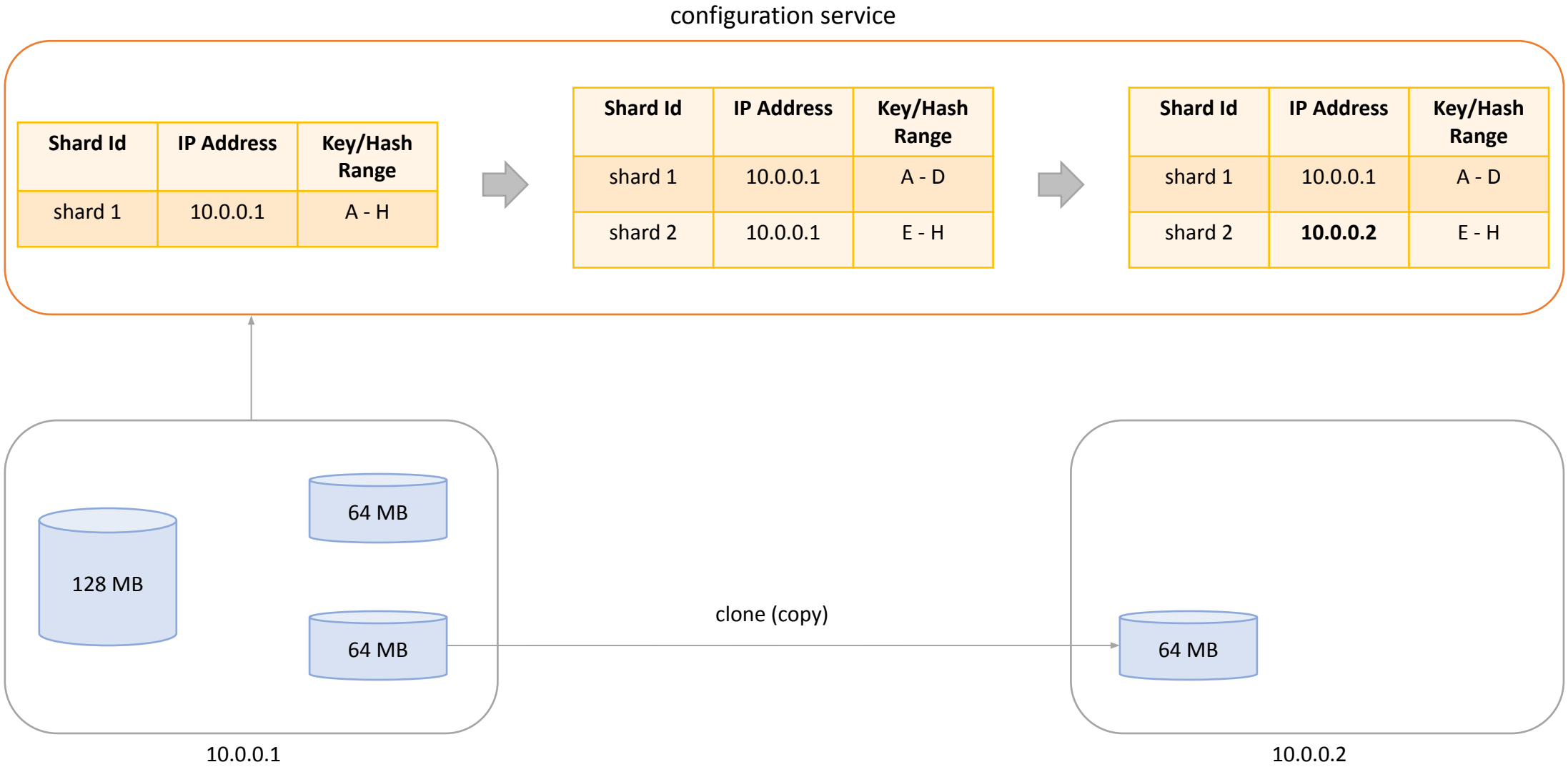
Rebalancing partitions

option 1
split a shard when it grows beyond the specified size



Rebalancing partitions

option 1
split a shard when it grows beyond the specified size



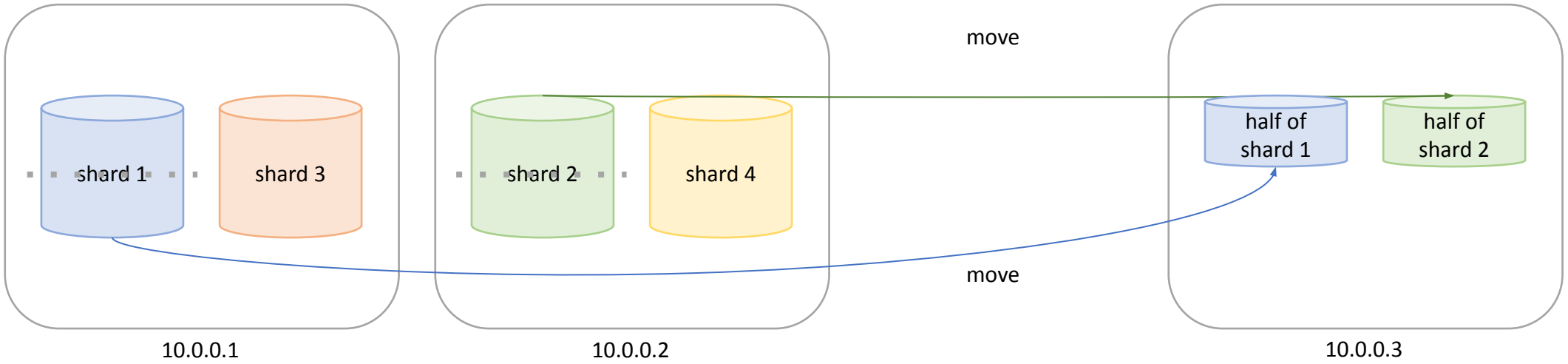
Rebalancing partitions

option 1
split a shard when it grows beyond the specified size

- splitting the range of keys that a shard owns is just a metadata change
- whether or not we need to redistribute the shards between servers
  is decided by another process, the balancer
- this is a background process that monitors the number of shards on each server
- adding a new server to a cluster triggers the balancer to start
  the shard migration process
- removing a server is basically the process of adding a server in reverse
- when a shard becomes too small, it can be merged with an adjacent shard

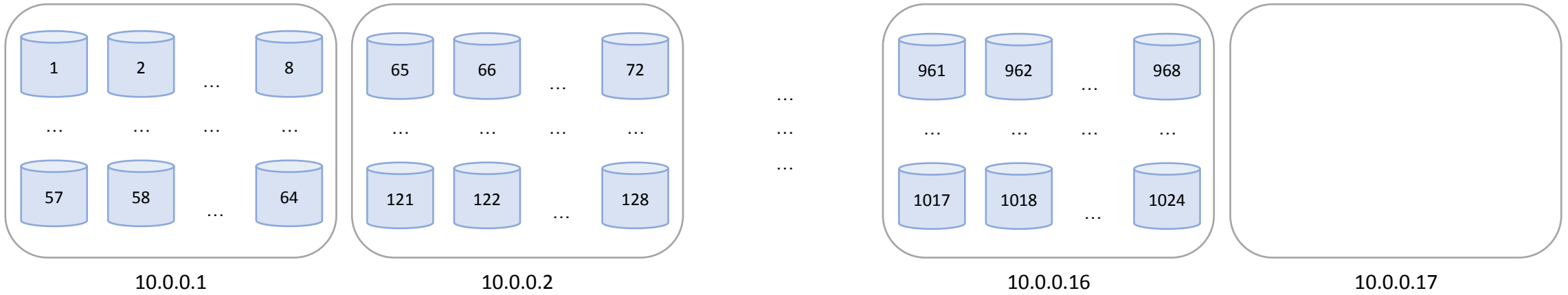
Rebalancing partitions

option 2
split shards when adding a new server



Rebalancing partitions

option 3
fixed number of shards



- easier (than strategies that require splits) to implement and maintain

- can be hard to choose the initial number of shards
- there is a risk that some shards will grow really large

Rebalancing partitions

Rebalancing Strategy	Partitioning Strategy	Real-world Examples
split a shard when it grows beyond the specified size	range, hash	MongoDB HBase
split shards when adding a new server	hash	Cassandra
fixed number of shards	hash, lookup	Couchbase Elasticsearch