CSE 452

Distributed Systems

Partitioning
aka Sharding
aka Splitting the data

Why build a distributed system?

- fault tolerance — replication
  - redundant computation + data
- performance
  - "edge computing" - move things close to users
  - split up the work / parallelism
- "horizontal scaling"
  add more machines → get better performance

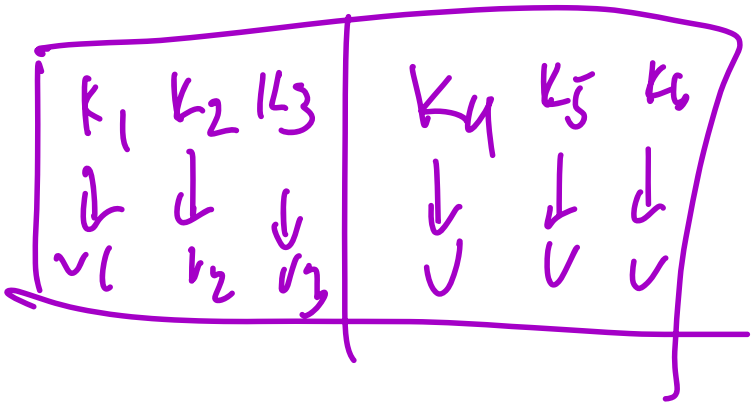
KV Store

- distributed
- fault tolerant
- linearizable
- partioning / sharding / splitting the data
- cross-partition transactions

Partitioning

- split the data
- some keys in one partition
  some keys in another

get(k)
put(k, v)
append(k, v)

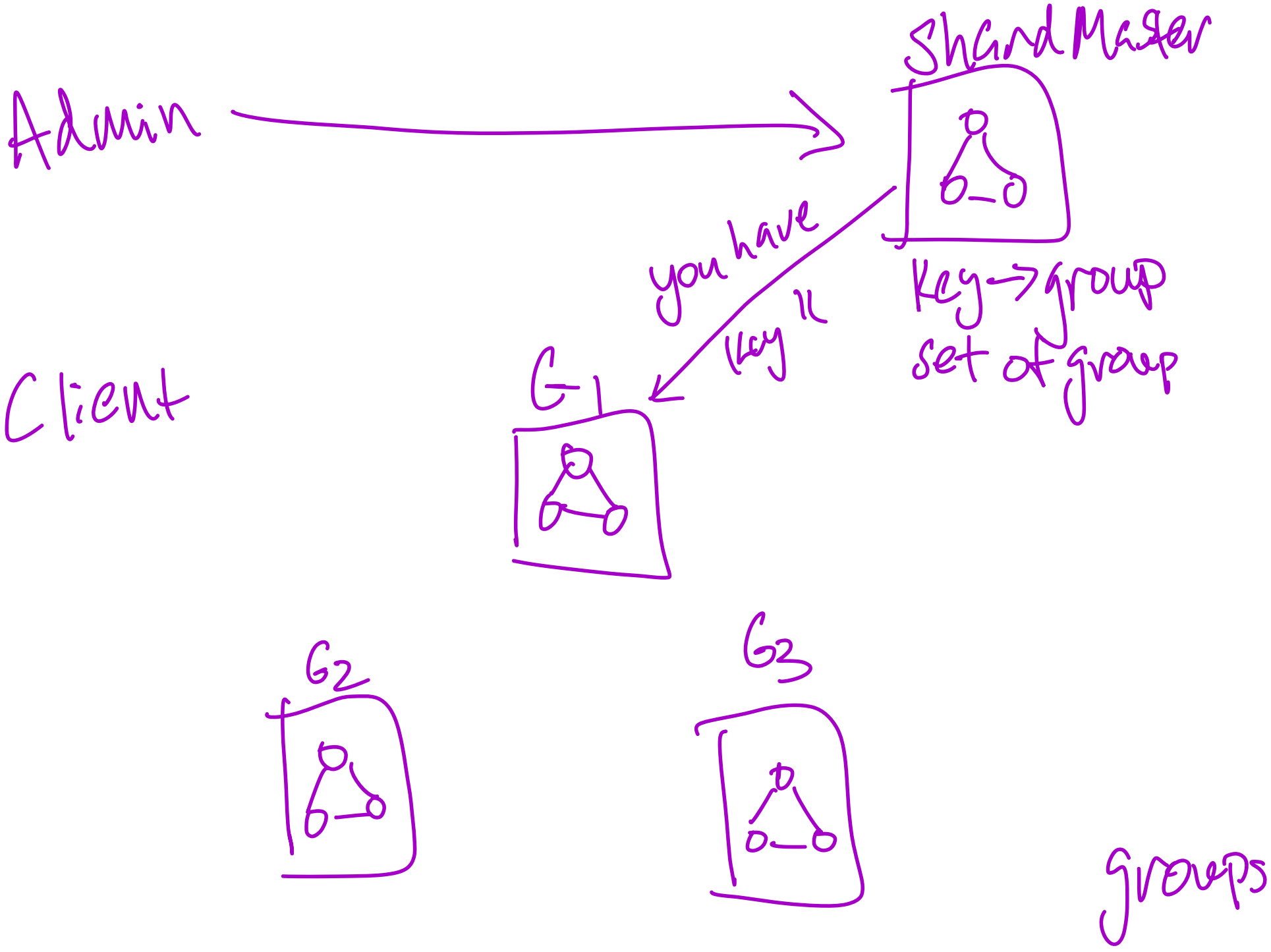


Static Partitioning

- ahead of time, decide which keys go to which partitions
- cons
  - stuck with choice of key → partition map
  - changing workloads
  - # of partitions

Dynamic Partitioning

- move keys between partitions
- supports load balancing
  - lab 4: manual load balancing
- keep track of current mapping of keys -> partitions
- add/remove entire partitions


Admin
Shard Master
Key → group
set of group
you have
key 11
Client
G1
G2
G3
groups

lab 4

part 1: Shard Master (not very distributed)

part 2: other groups that implement the sharded KV store
- moving keys

part 3: multi key operations
cross partition operations

Shard Master

- Application: Command, Result
- Query - reads current configuration
- Join — add new group*
- Leave — remove group*
- Move — move a shard to specific group

* = rebalance key space — move as few shards as possible

Configuration: metadata

- config num — monotonic
- keys → groups (shards → groups)
- set of groups

---

shard = small set of keys

group = one paxos cluster available for KVStore ops