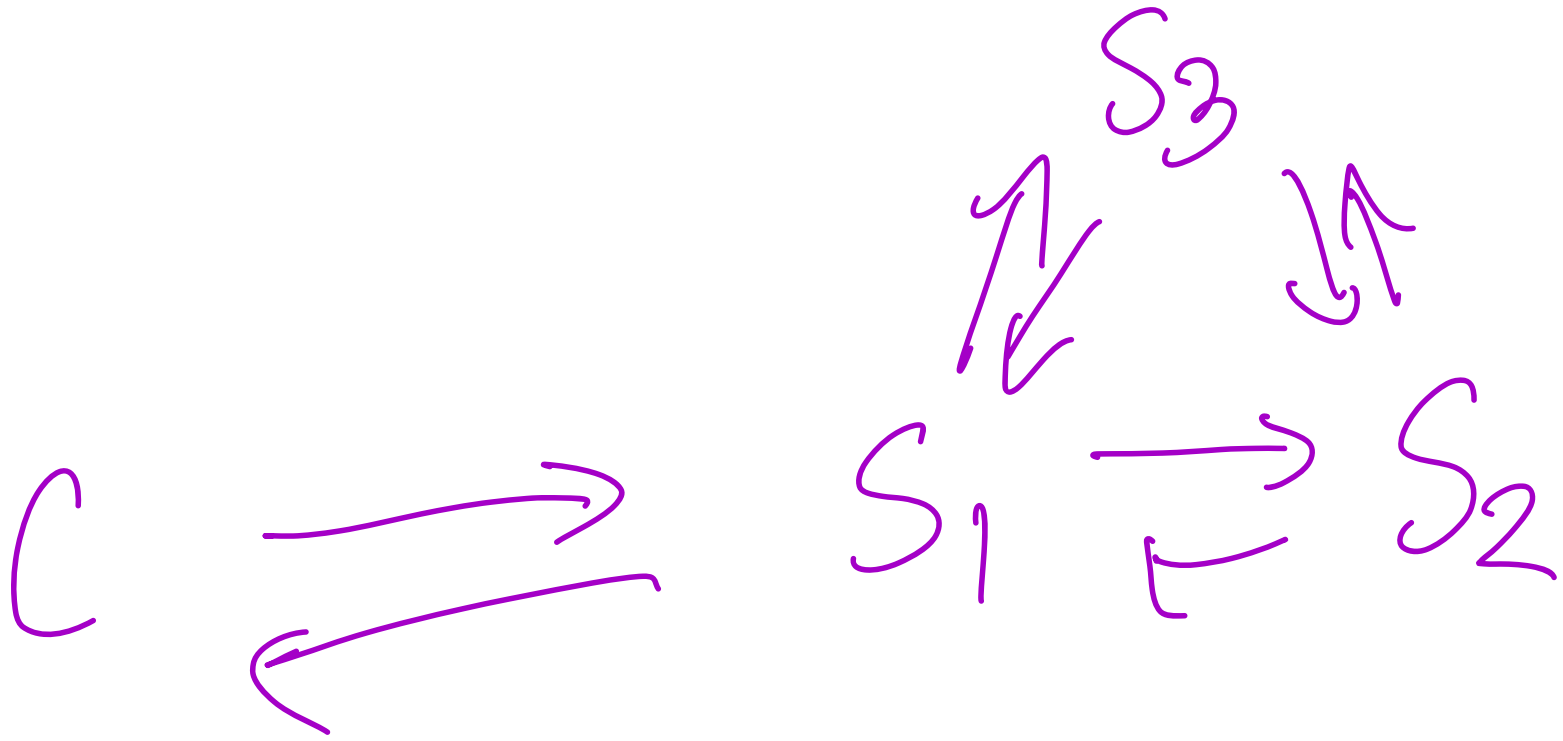
CSE P 552

Distributed Systems

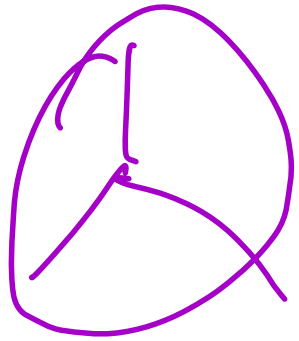
Week 6



✓ fault tolerant

✗ increase performance

Partitioning / Sharding

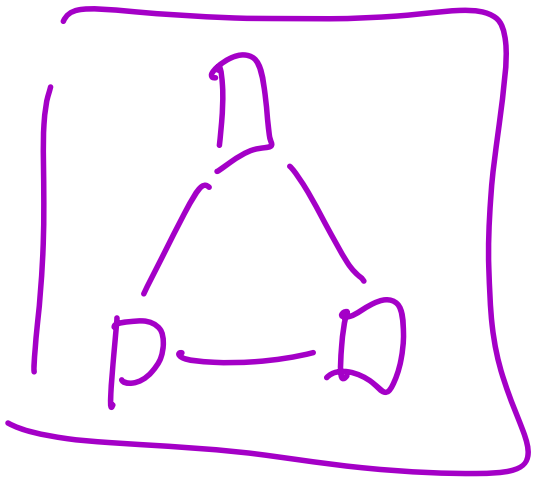


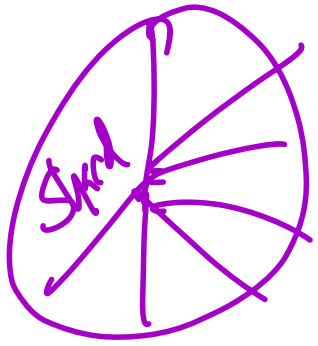
- split up the data
  - one partition has only some of the data
- what operations are available on the data?
  - if they can be served by single partition, then easy to horizontally scale
    - just buy more machines + repartition

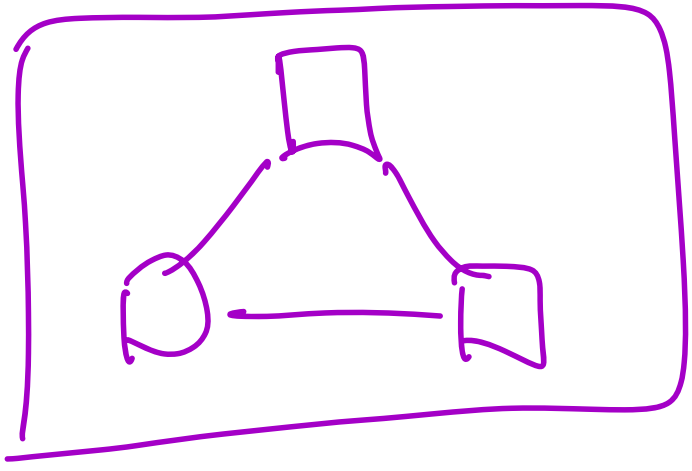
To dos:

- fault tolerance
- repartitioning
- Multi partition operations



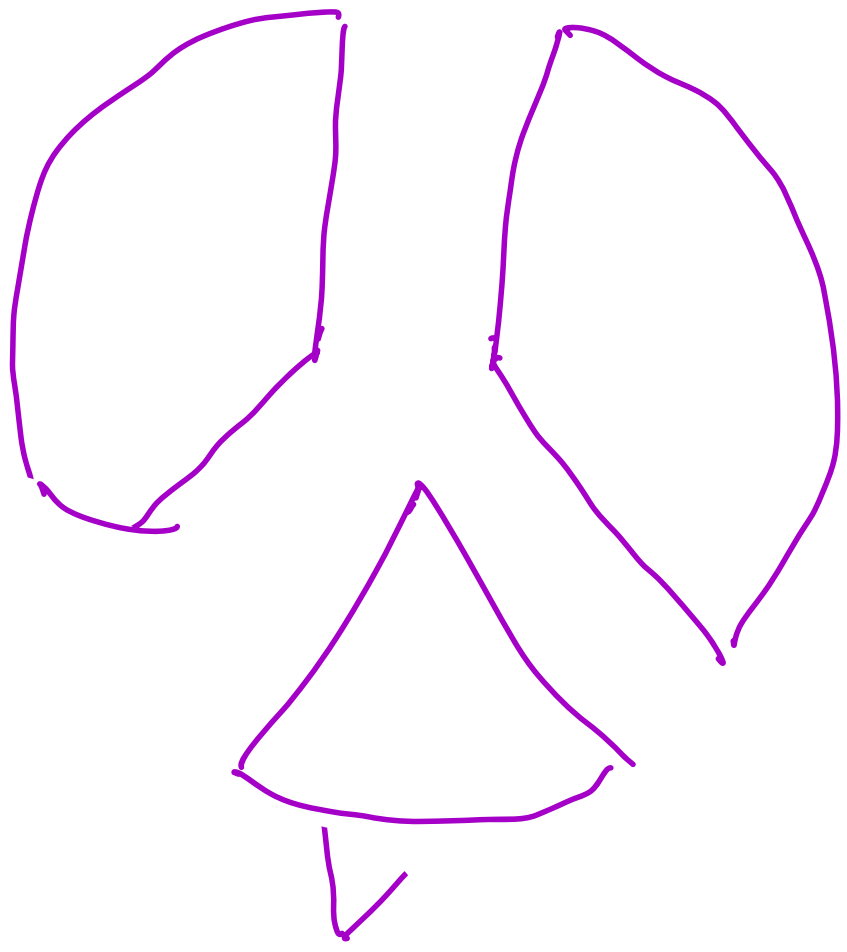








ShardMaster



Who owns which keys?

- static partitioning
- dynamic partitioning
  - indirection table (lab 4)
    - consistent hashing (not lab 4)

Indirection Table (lab 4 configuration)

Configuration: (static map: keys → shards)

map: shards → groups

Config_num

- each shard assigned to exactly one group
- but each group may be assigned 0, 1, many shards

Reading 2: small groups, 15 minutes

Groups of 3 or 4. Mix papers.

- Introduce yourselves.
- Say what paper you read.
- Give a brief overview of each paper to the group.

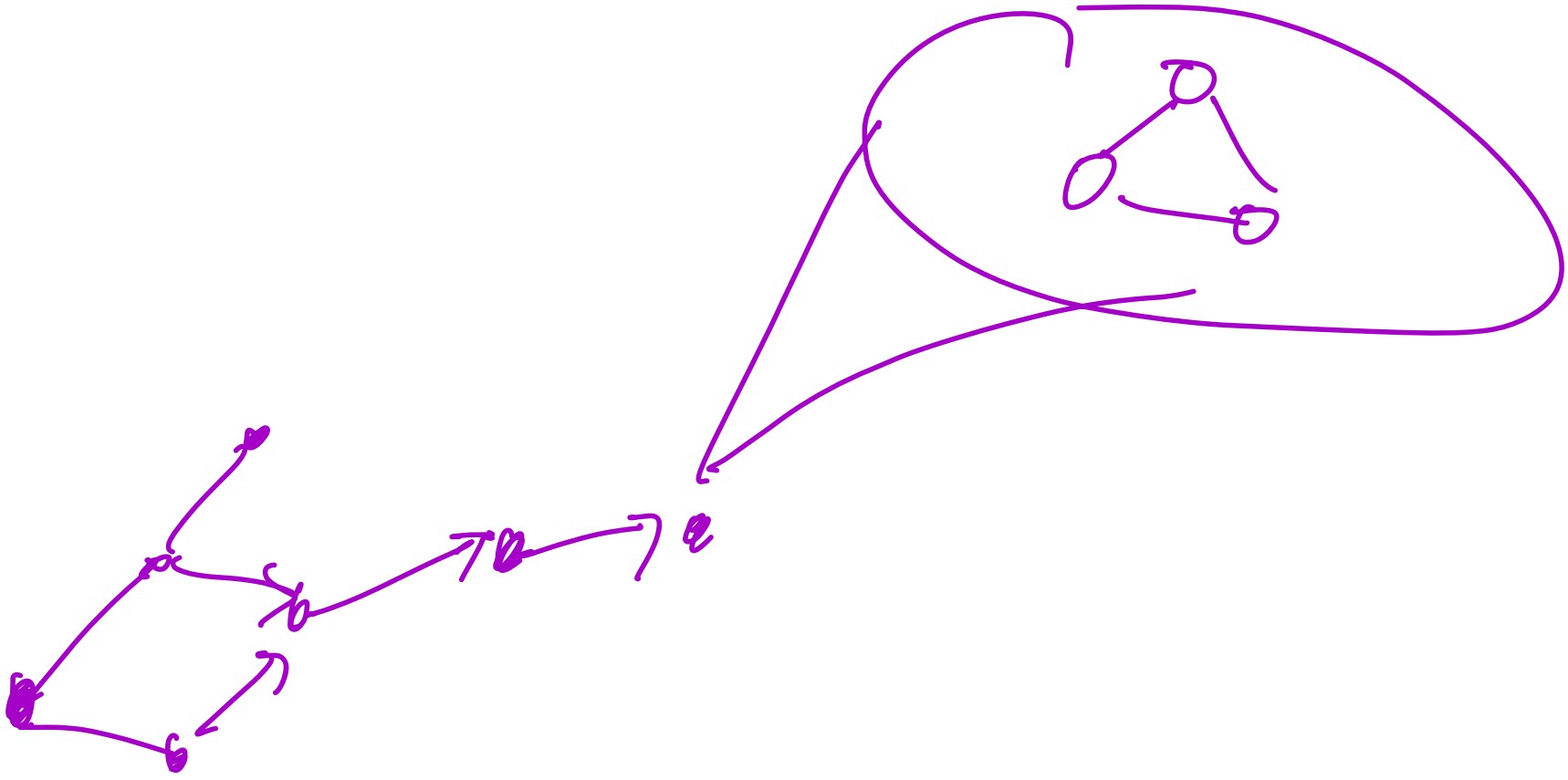
Some suggestions for things to discuss:

1. What does a “safety margin” mean in distributed systems?
2. What does a timeout tell you? Are they good or bad?
3. Anything else you think is interesting.

It may be interesting to also discuss how these relate to Lab 3.

We’ll debrief as a class.

- fault model?
- social/organizational influence on fault tolerance
- moving the problem around
- deviation from published protocol
- don't depend on ~~other~~ software
- hard to package & share distributed sys. solutions



- deterministic simulation testing

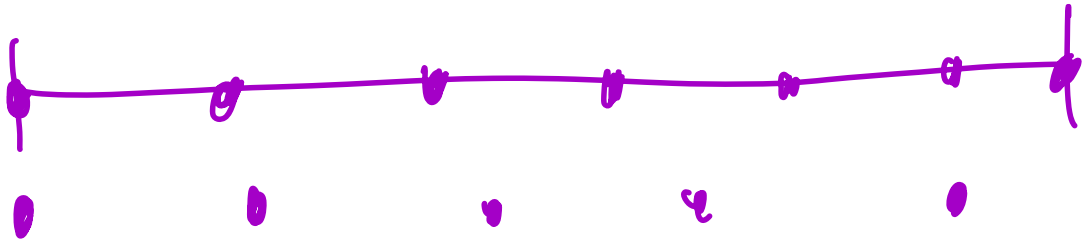
Consistent Hashing

- n = #groups

  $K \mapsto K \bmod n$

$\simeq \frac{n-1}{n}$ of data needs to move

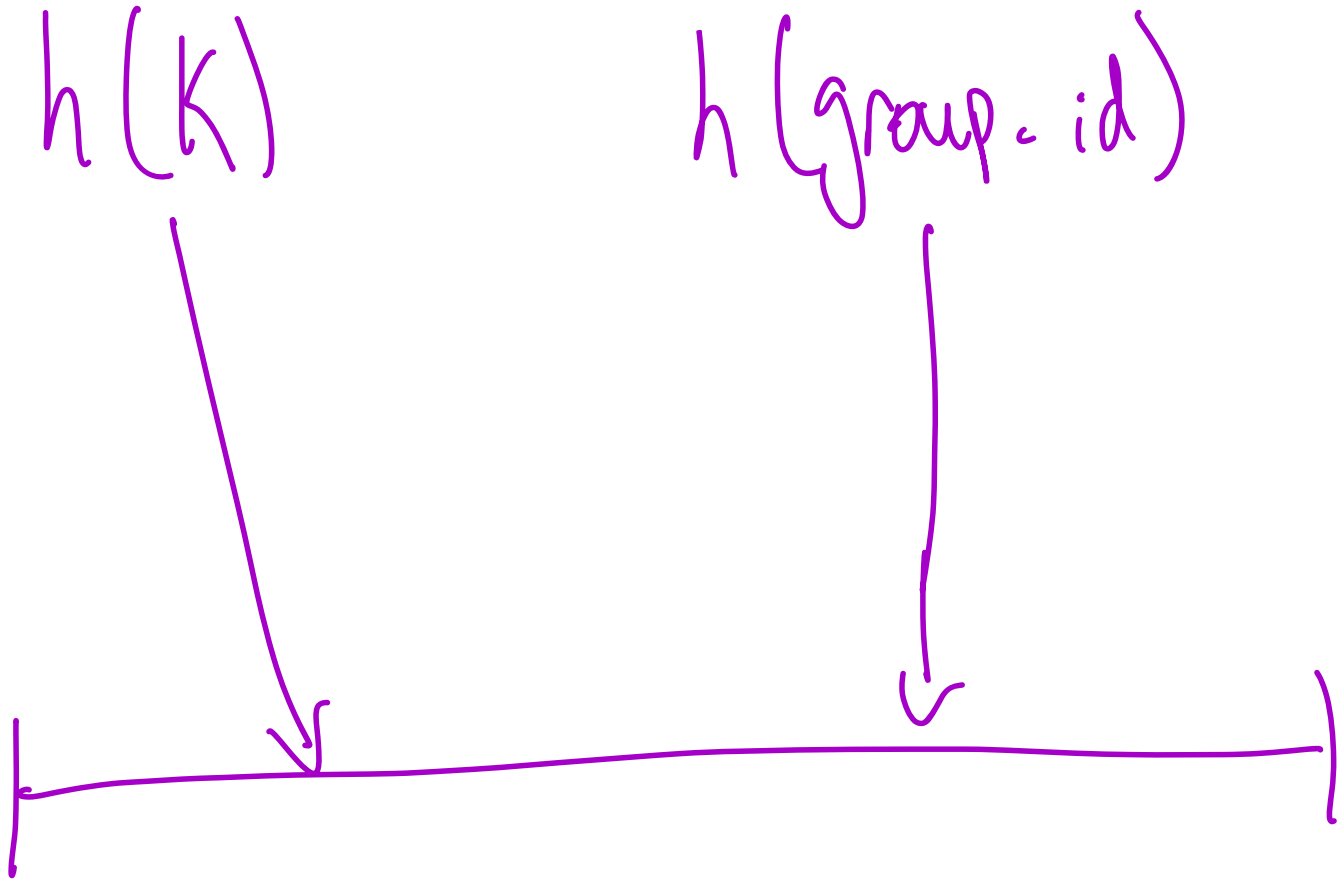
↳ add a group



h(k) mod n

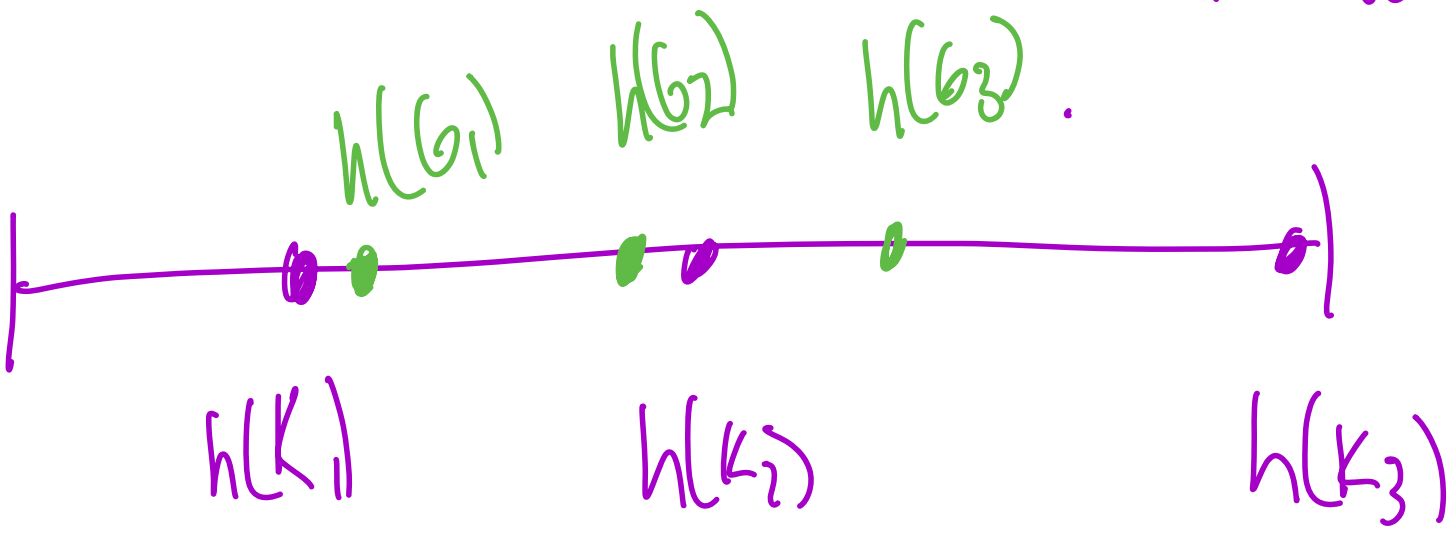
- balance workload
- still move most data

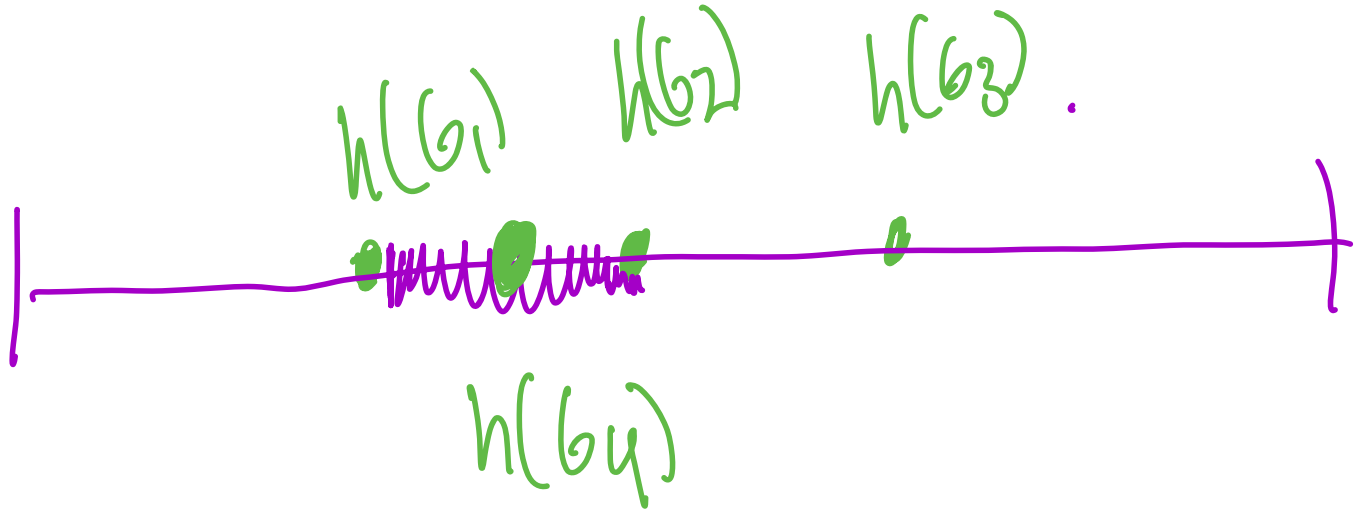
Consistent Hashing



Consistent Hashing

+ most data doesn't move
- all movement in ~2 groups

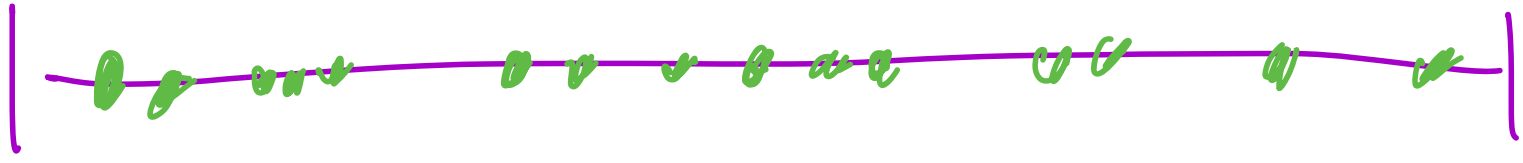




Virtual Groups

+ 1/n data is moved
+ work of moving is distributed

- 3 real groups G_1 G_2 G_3
- each w/ 100 virtual groups $G_{1,1}$ $G_{1,2}$



How to adjust indirection table?

What commands does Shard Master Accept?

writes
- Move(shard, to-group)
- Join(group) — "rebalance" key space while minimizing # of keys moved
- Leave(group)

read
- Query(config_num) that config
  or if num = -1, then most recent