Two Issues With Content distribution

* meta server overload (storing <key, value>)
* Origin server overload (Storing content)

Origin Server overload

Two Solutions

- Web proxy
  - not good enough for slashdot effect

Origin Server overload

Two Solutions

- Web proxy
  - not good enough for slashdot effect
- CDN
  - Content mirroved stored geographically
  - User request dynamically re-routed to geo-local mirror

Origin Server overload

Two Solutions

- Web proxy
  - not good enough for slashdot effect
- CDN
  - Content mirrored stored geographically
  - User request dynamically re-routed to geo-local mirror

  ⇒ pay CDN providers to avoid origin overload

Origin Server overload

Two Solutions

- Web proxy
  - not good enough for slashdot effect
- CDN
  - Content mirrored stored geographically
  - User request dynamically re-routed to geo-local mirror

  ⇒ pay CDN providers to avoid origin overload

How to Democratize content distribution?

Origin Server overload

Two Solutions

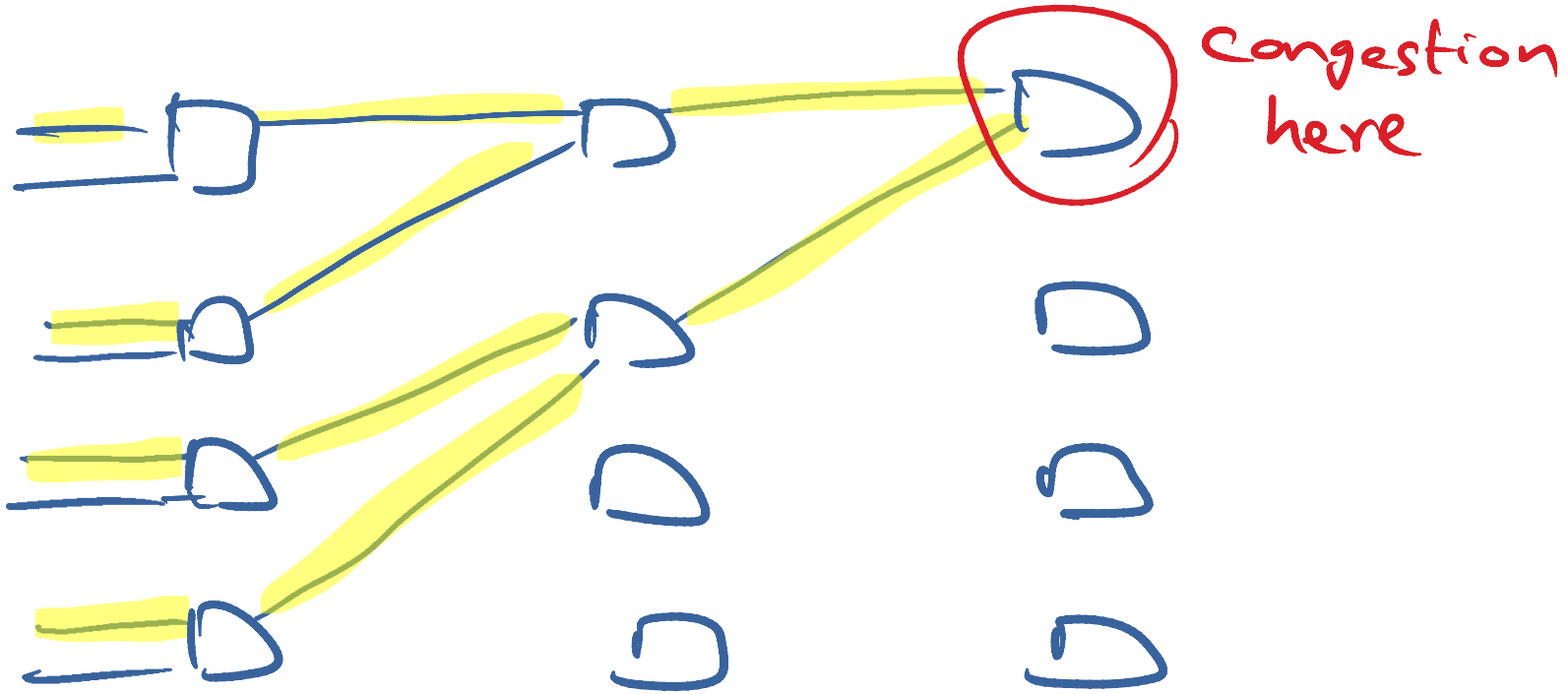
- Web proxy
  - not good enough for slashdot effect
- CDN
  - Content mirrored stored geographically
  - user request dynamically re-routed to geo-local mirror

  ⇒ pay CDN providers to avoid origin overload

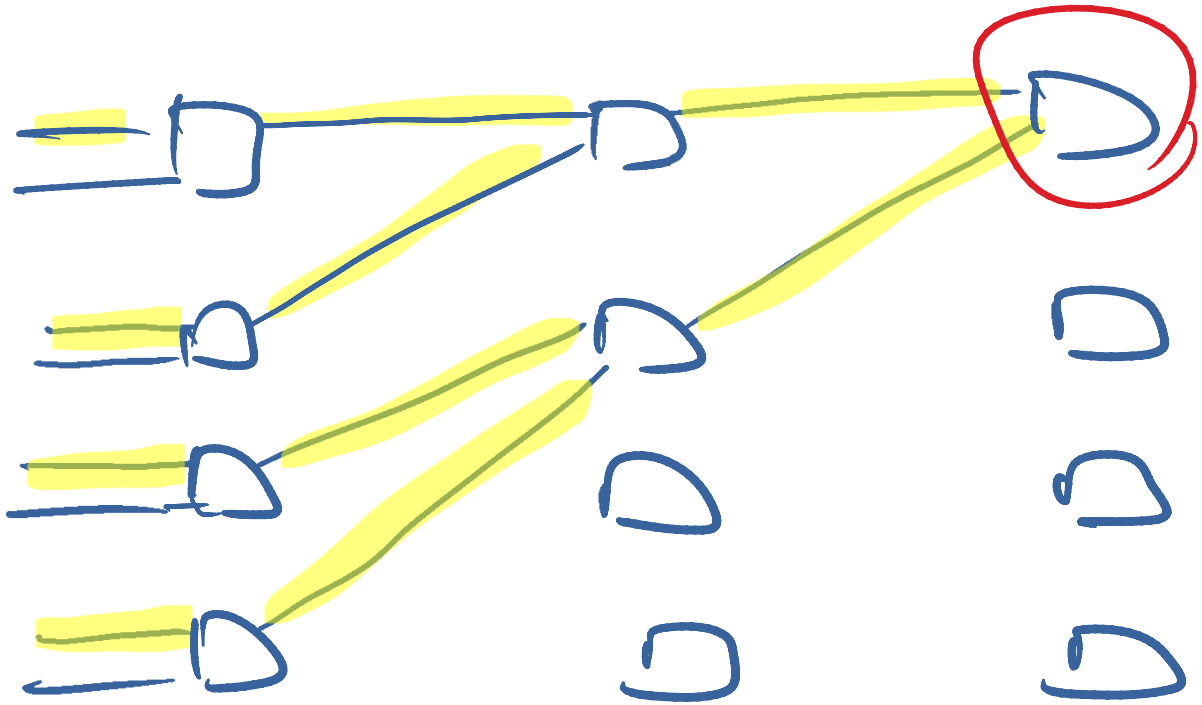
How to Democratize content distribution?

Enter Coral System

Greedy Approach leads to "tree saturation"



Greedy Approach leads to "tree saturation"

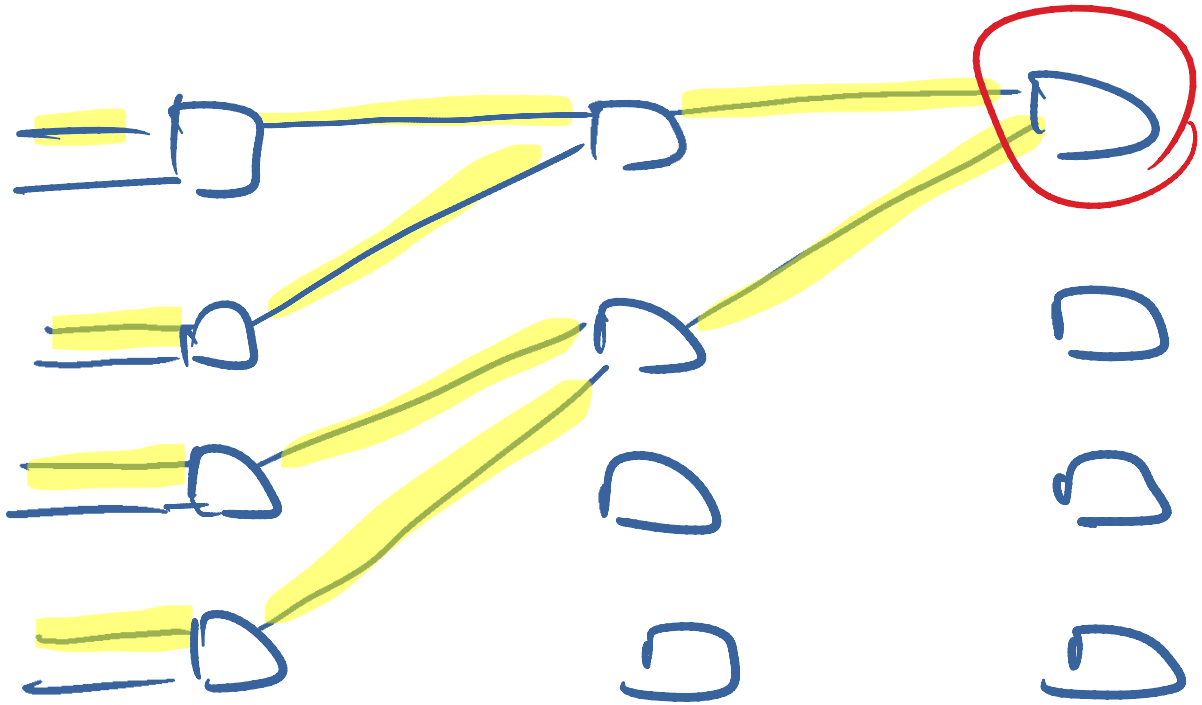


Congestion here

Coral approach

Don't be greedy!

Greedy Approach leads to "tree saturation"



Congestion here

Coral approach

Don't be greedy!

K ≪ N

⇒

hint not absolute

Coral DHT

- Get/put satisfied by nodes different from $<N \cong key>$ ⇒ Sloppy DHT

Rationale

- avoid tree saturation
- spread metadata overload

Coral DHT

- Get/put satisfied by nodes different from $\langle N \cong key \rangle$ ⇒ Sloppy DHT

Rationale

- avoid tree saturation
- spread metadata overload

How?

- Novel key-based routing
- basis
  - XOR distance between src & dst
    N_{src} XOR N_{dst} (e.g., 14 XOR 4 = 10)

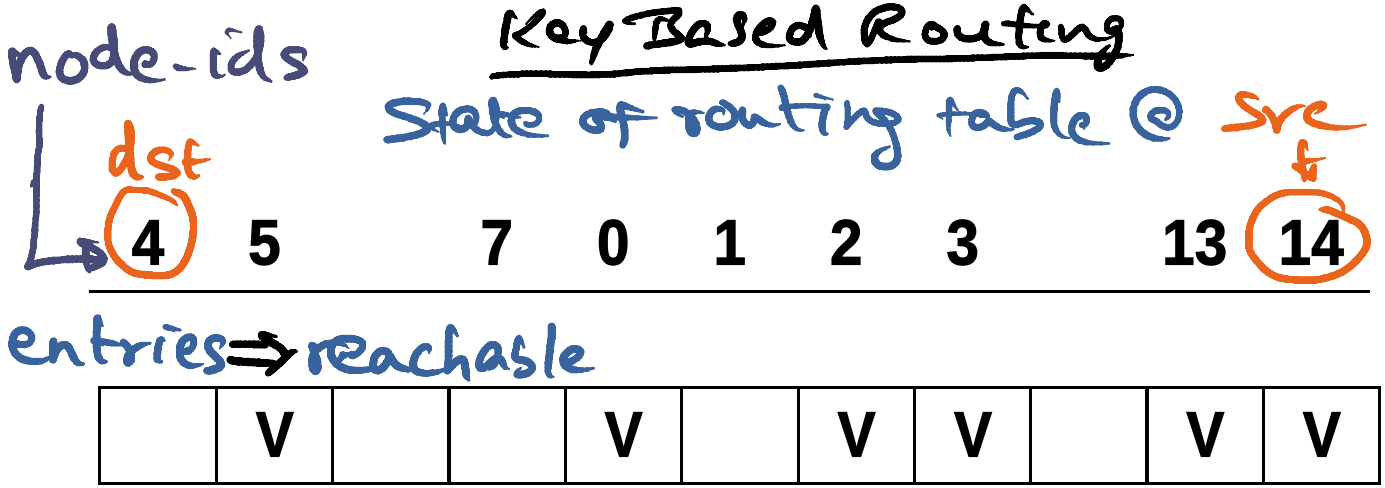
Key Based Routing

node-ids

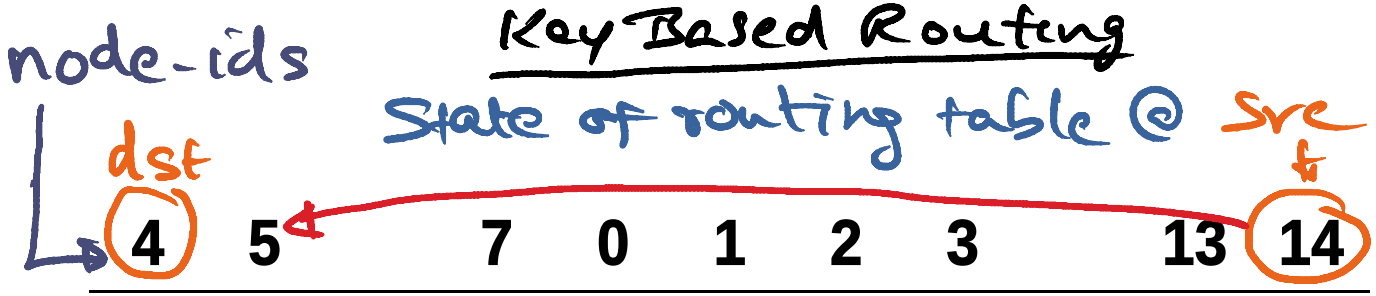
dst

4	5		7	0	1	2	3		13	14
	V			V		V	V		V	V

Key Based Routing



Key Based Routing

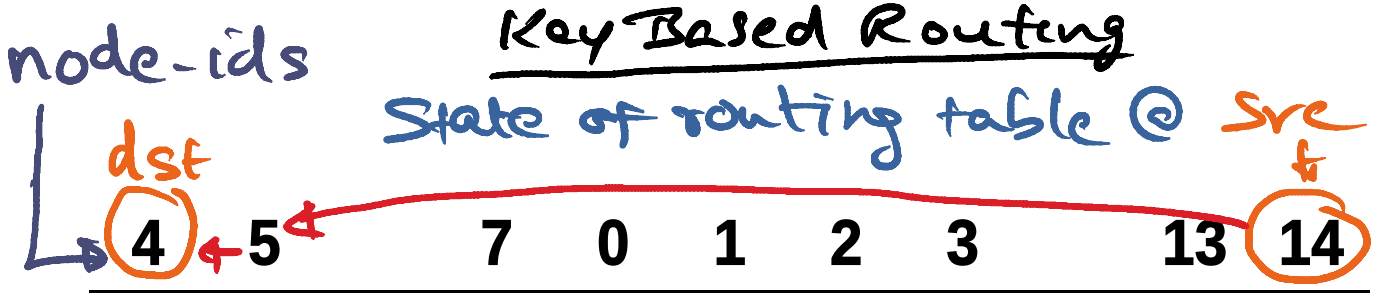


entries ⇒ reachable

	V			V		V	V		V	V

Intuition in greedy approach
- get as close to the desired dst in node-id namespace

Key Based Routing



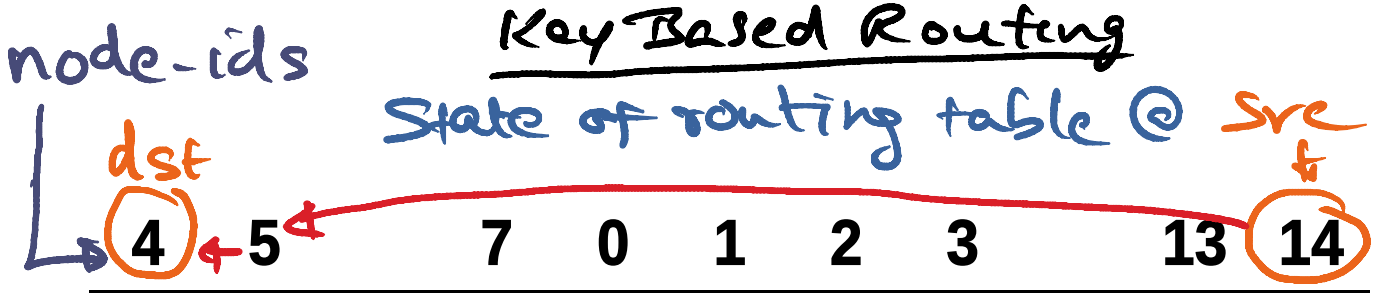
entries ⇒ reachable

	V			V		V	V		V	V

Intuition in greedy approach

- get as close to the desired dst in node-id namespace
- he may know route to get me to desired dst

Key Based Routing



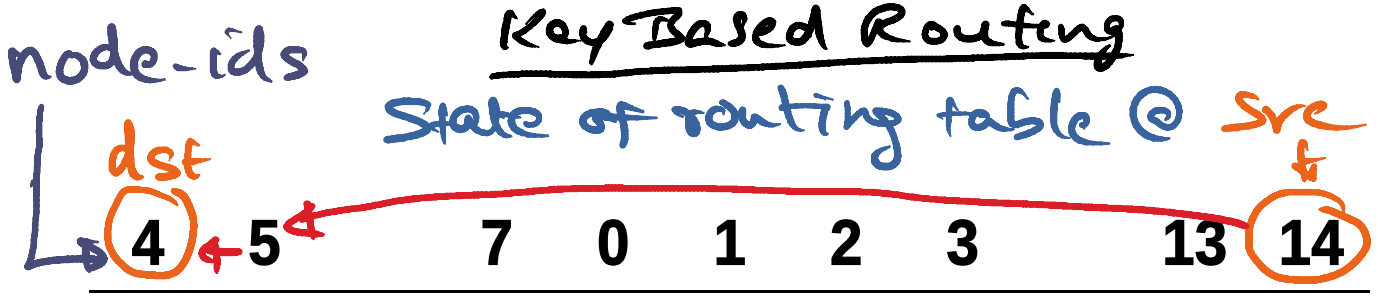
entries ⇒ reachable

	V			V		V	V		V	V

Intuition in greedy approach
- get as close to the desired dst in node-id namespace
- he may know route to get me to desired dst

Objective?
- reach dst with fewest number of hops

Key Based Routing



entries ⇒ reachable

	V			V		V	V		V	V

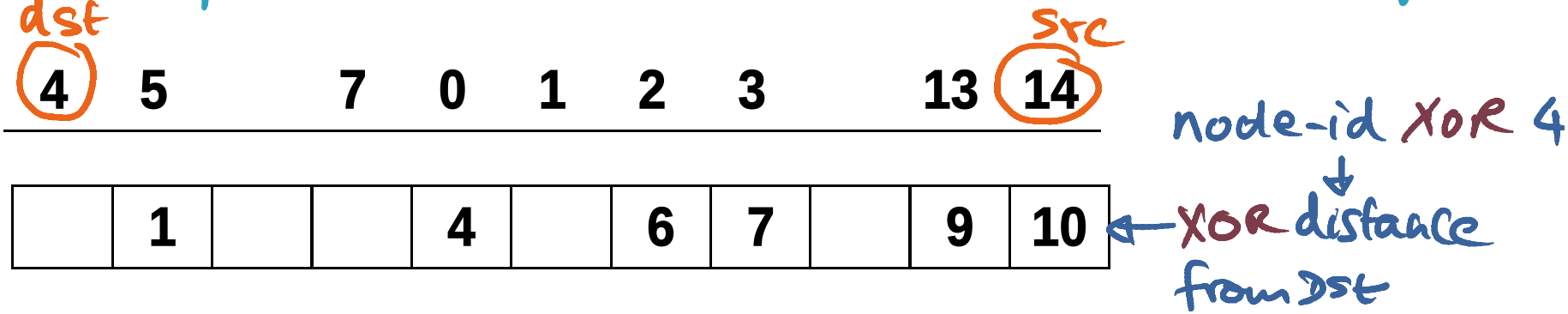
Intuition in greedy approach
- get as close to the desired dst in node-id namespace
- he may know route to get me to desired dst

Objective?
- reach dst with fewest number of hops

⇒ "me first" approach leads to congestion

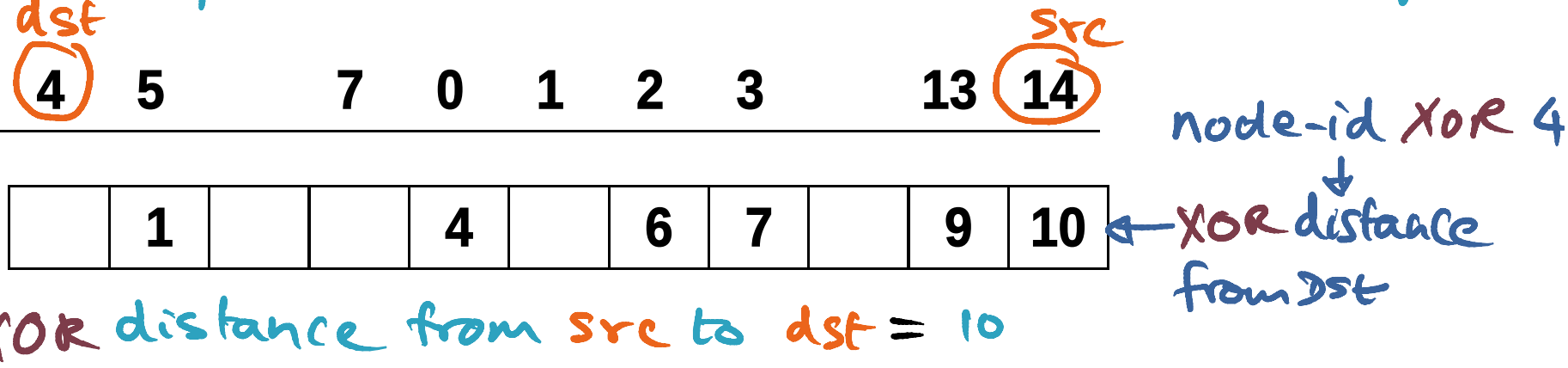
Coral Key-based routing

- Each hop go to some node that is half the distance to dst in node-id namespace



Coral Key-based routing

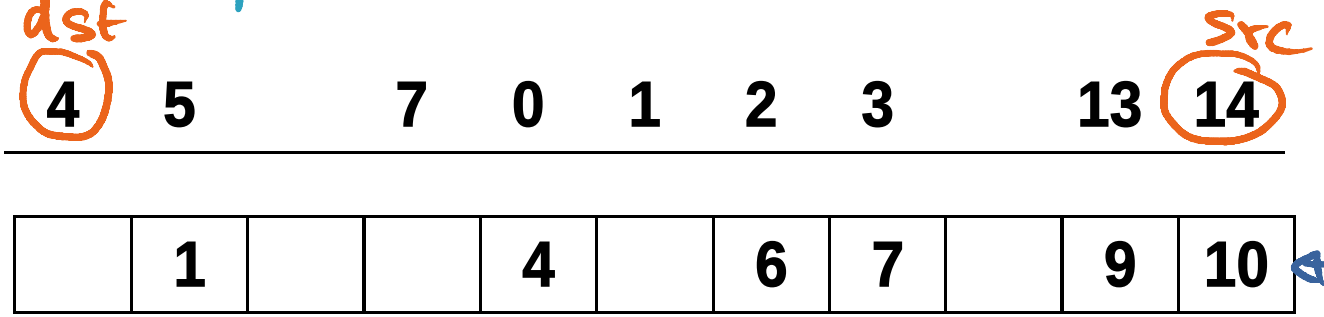
- Each hop go to some node that is half the distance to dst in node-id namespace



XOR distance from src to dst = 10

Coral Key-based routing

- Each hop go to some node that is half the distance to dst in node-id namespace

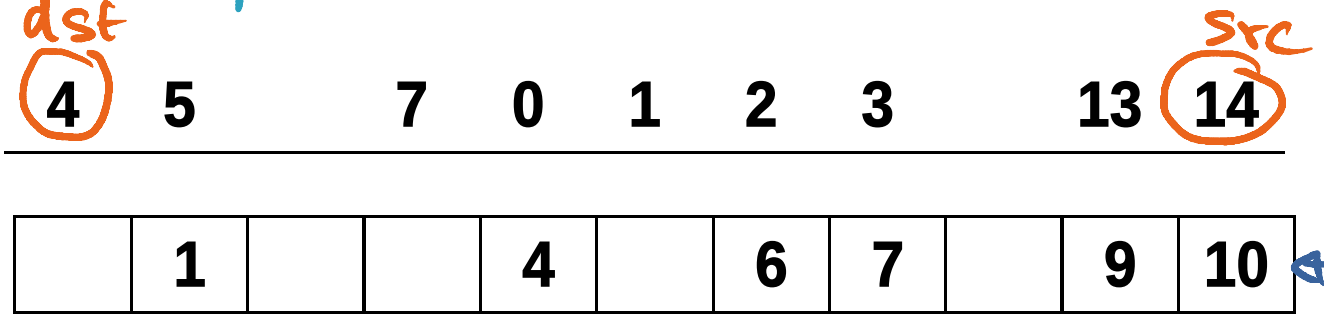


XOR distance from src to dst = 10

1st hop go to $\frac{10}{2} \Rightarrow 5$ distant

Coral Key-based routing

- Each hop go to some node that is half the distance to dst in node-id namespace



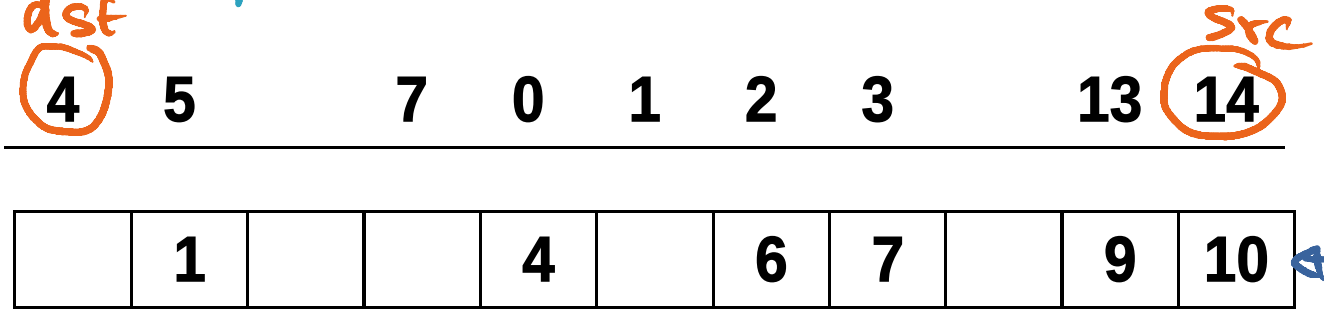
XOR distance from src to dst = 10

1st hop go to $\frac{10}{2}$ => 5 distant

2nd hop go to $\frac{5}{2}$ => 2 distant

Coral Key-based routing

- Each hop go to some node that is half the distance to dst in node-id namespace



XOR distance from src to dst = 10

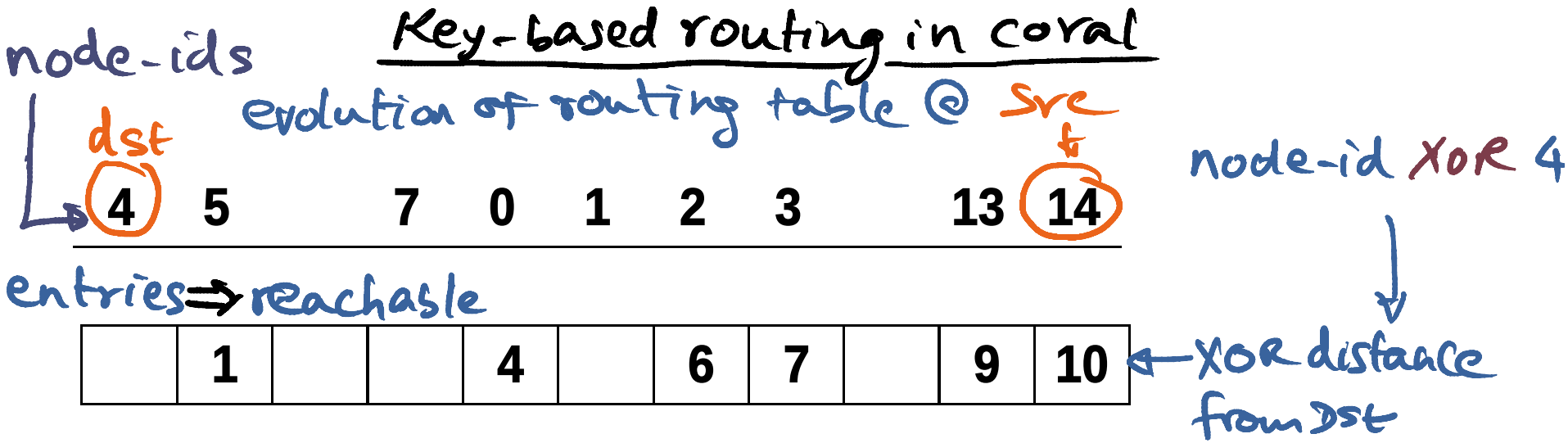
1st hop go to $\frac{10}{2}$ => 5 distant

2nd hop go to $\frac{5}{2}$ => 2 distant

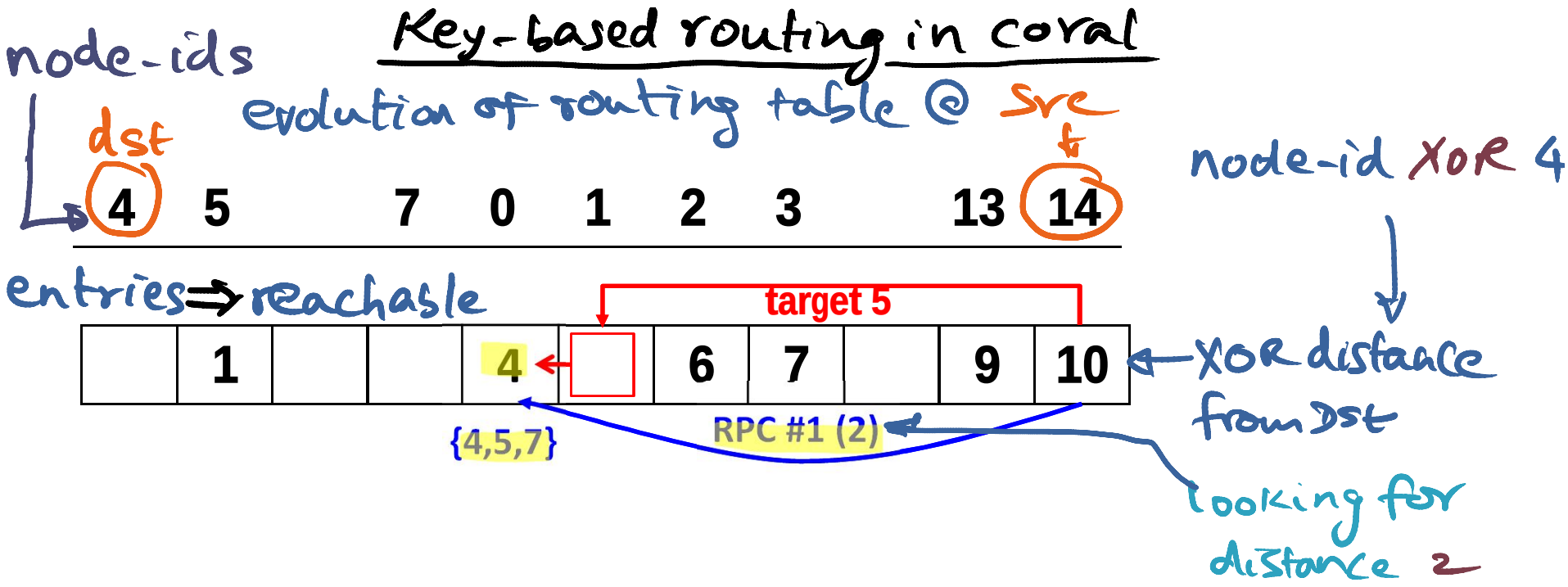
3rd hop go to $\frac{2}{1}$ => 1 distant

.... finally home!!

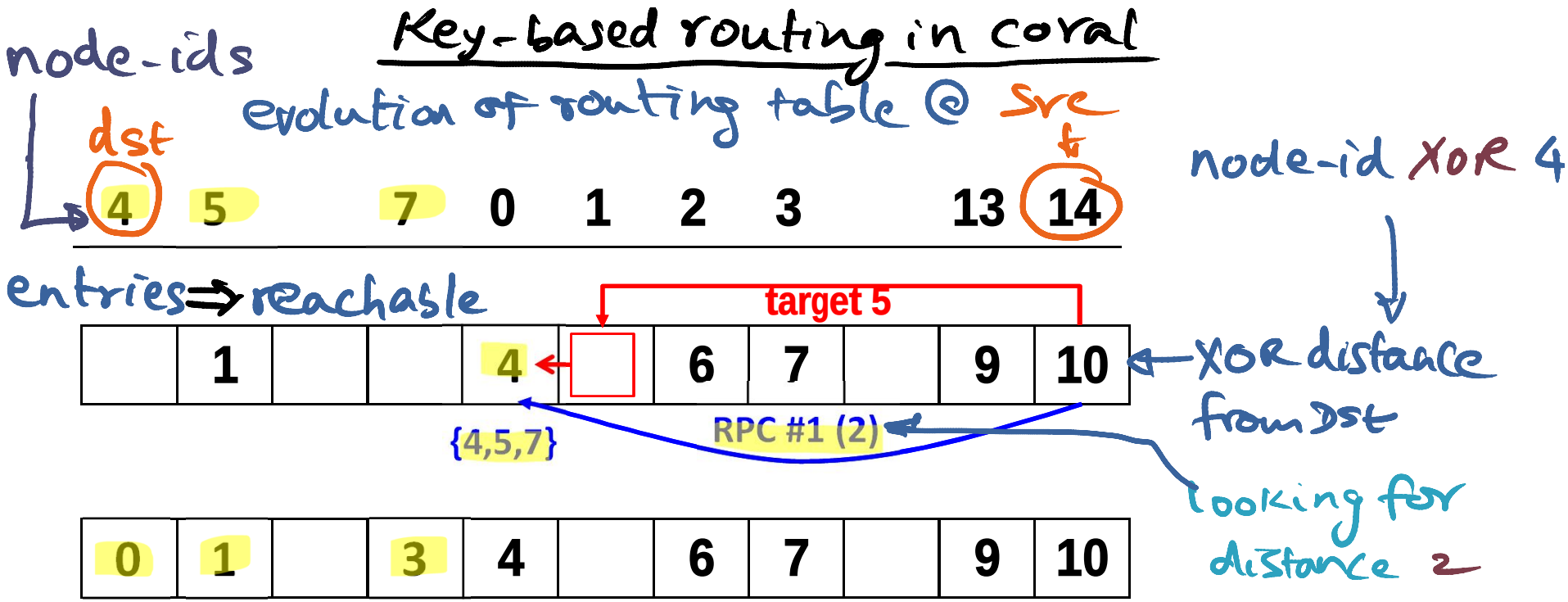
Key-based routing in coral



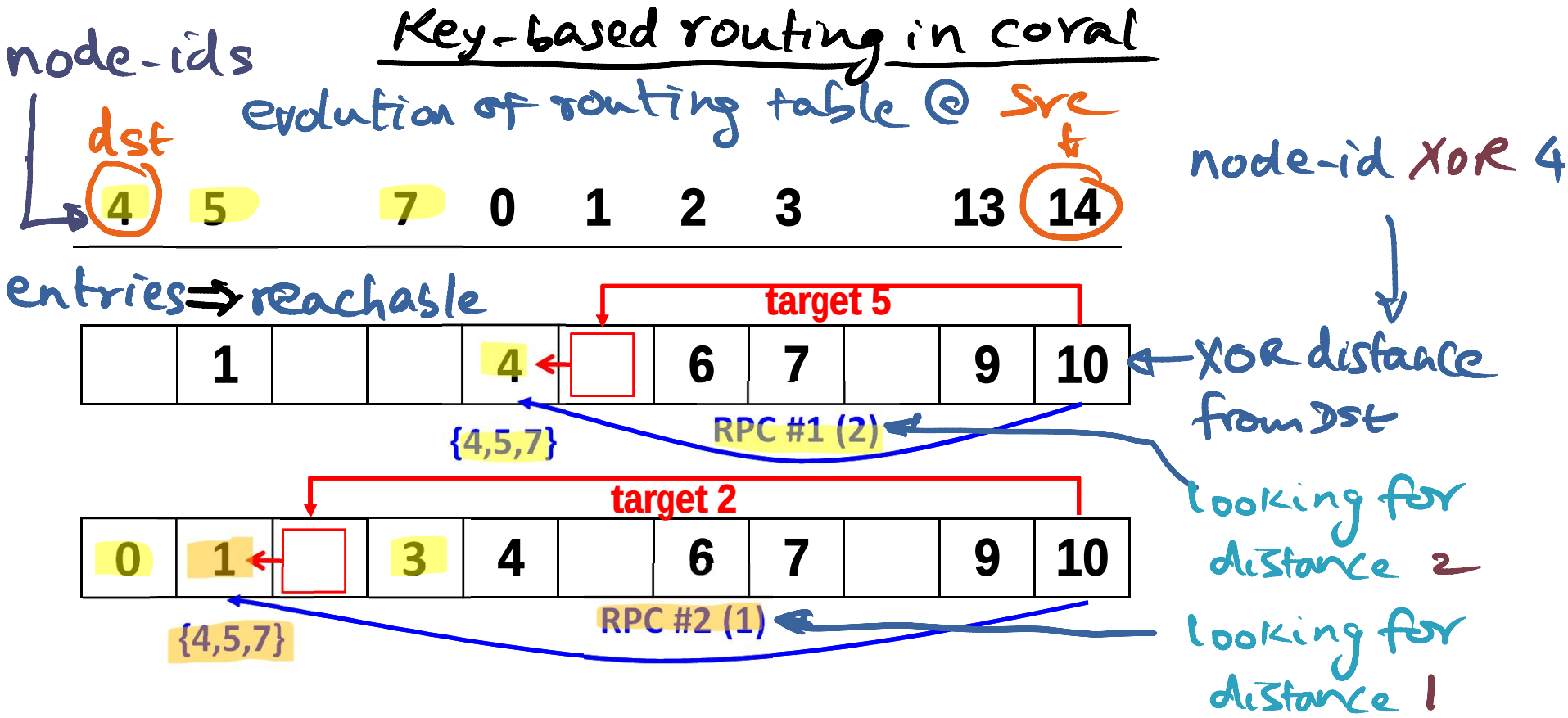
Key-based routing in coral



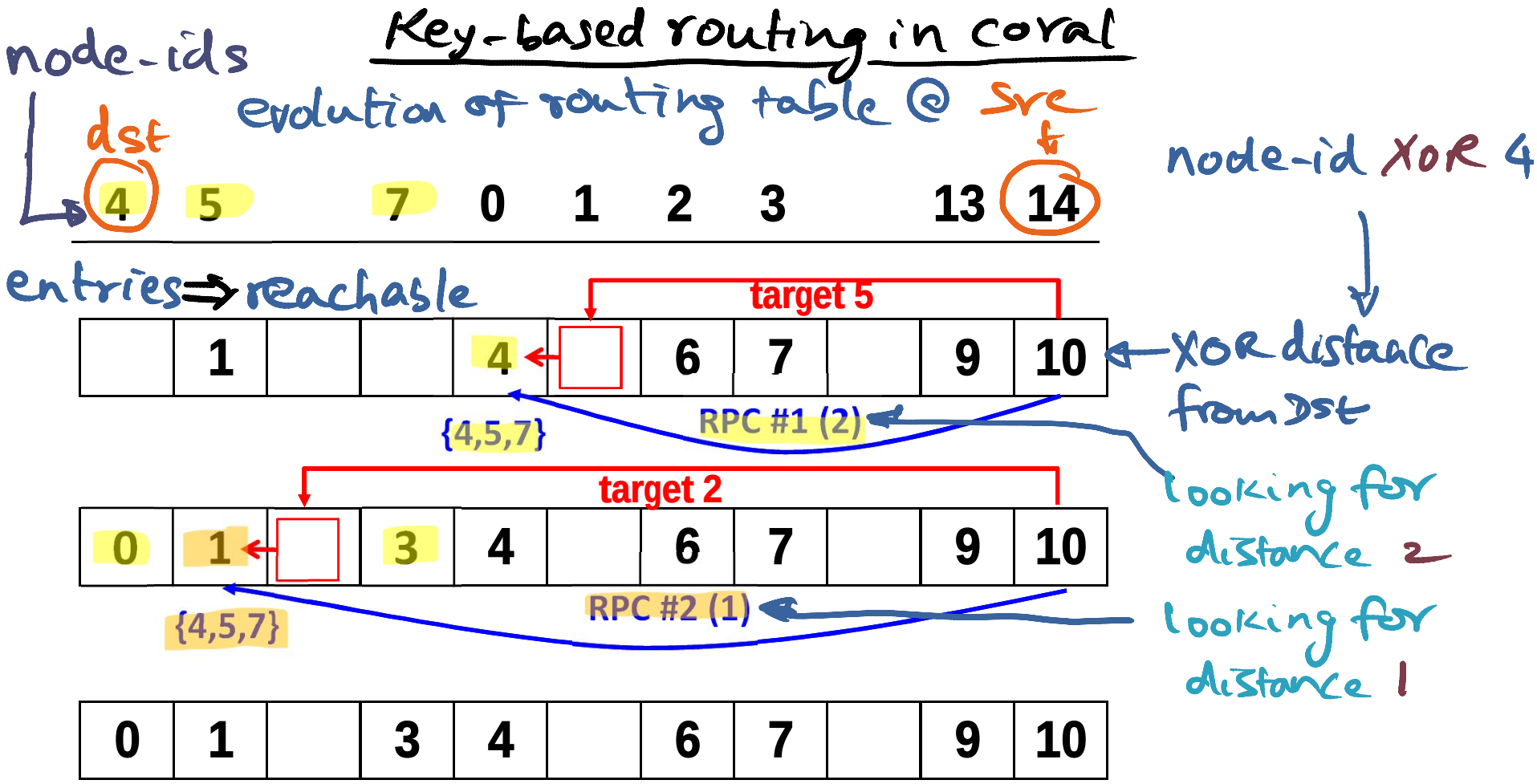
Key-based routing in coral



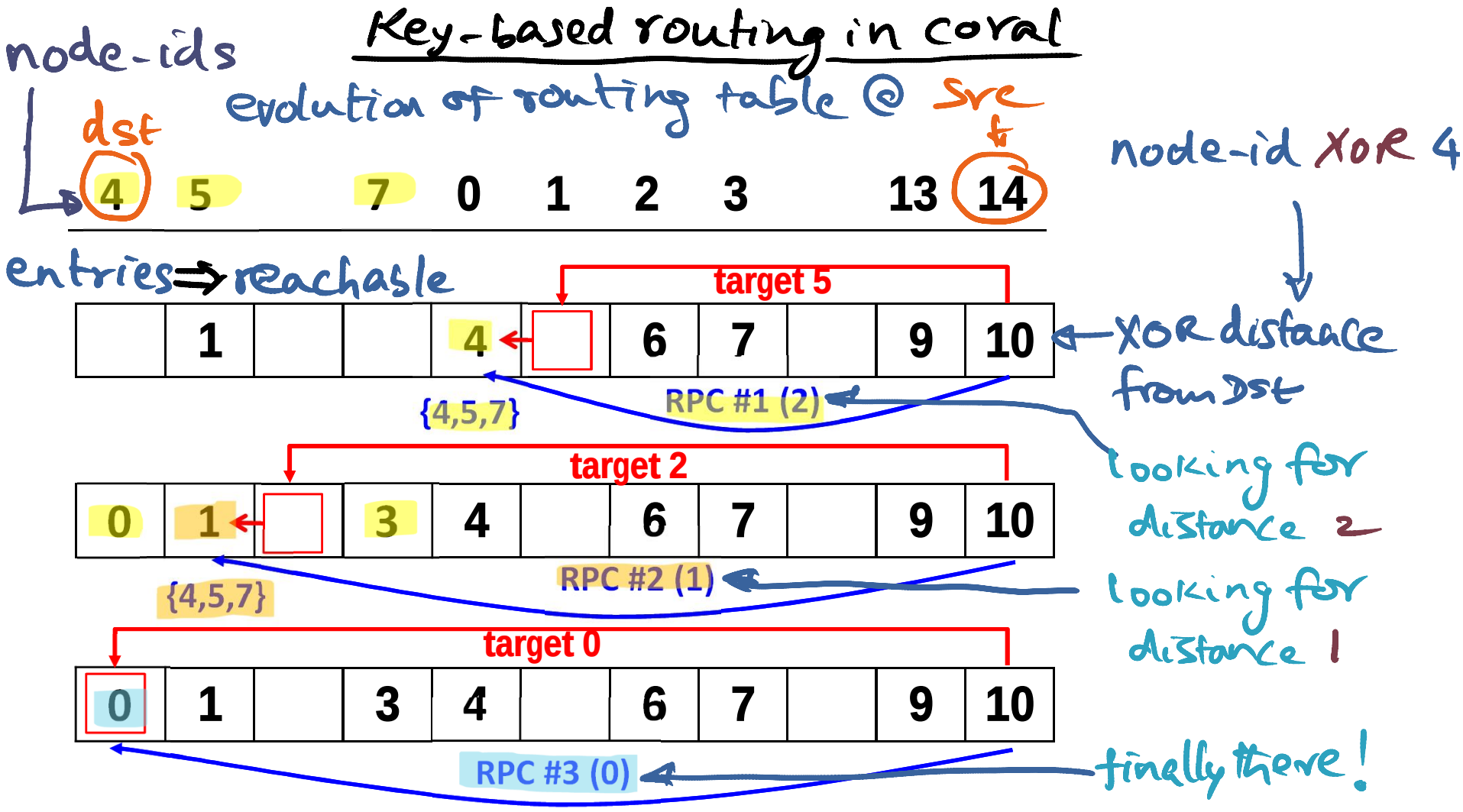
Key-based routing in coral



Key-based routing in coral



Key-based routing in coral



Coral Sloppy DHT

Coral primitives

- put (key, value)

  value: node-id of proxy with content for key

  * announcing willingness to serve as proxy

Coral Sloppy DHT

Coral primitives

- put (key, value)

  value: node-id of proxy with content for key

  * announcing willingness to serve as proxy
  * put <key, value> in an "appropriate" node

Coral Sloppy DHT

Coral primitives

- put (key, value)

  value: node-id of proxy with content for key

  * announcing willingness to serve as proxy
  * put <key, value> in an "appropriate" node

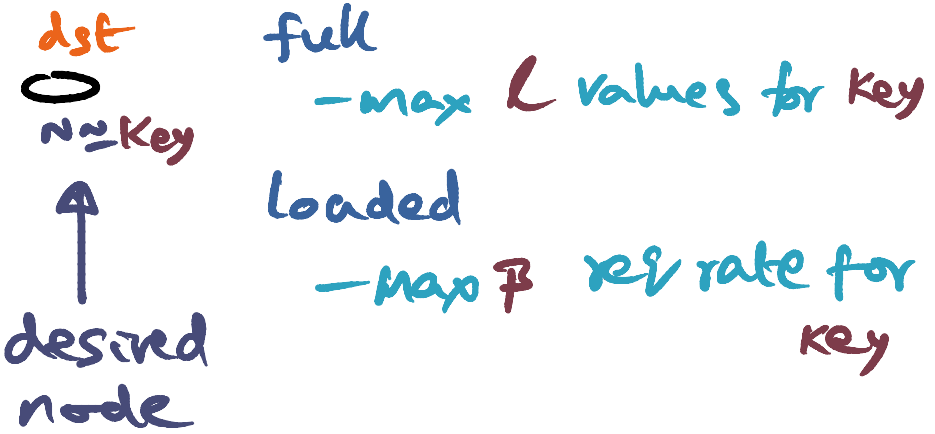
Coral Sloppy DHT

Coral primitives

- put (key, value)

  value: node-id of proxy with content for key

  * announcing willingness to serve as proxy
  * put <key, value> in an "appropriate" node



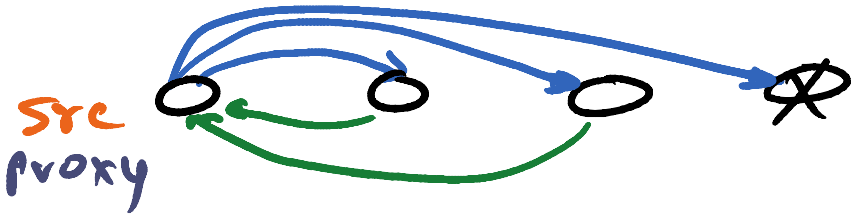
Coral Sloppy DHT

Coral primitives

- put (key, value)

node-id of proxy with content for key

* announcing willingness to serve as proxy
* put <key, value> in an "appropriate" node



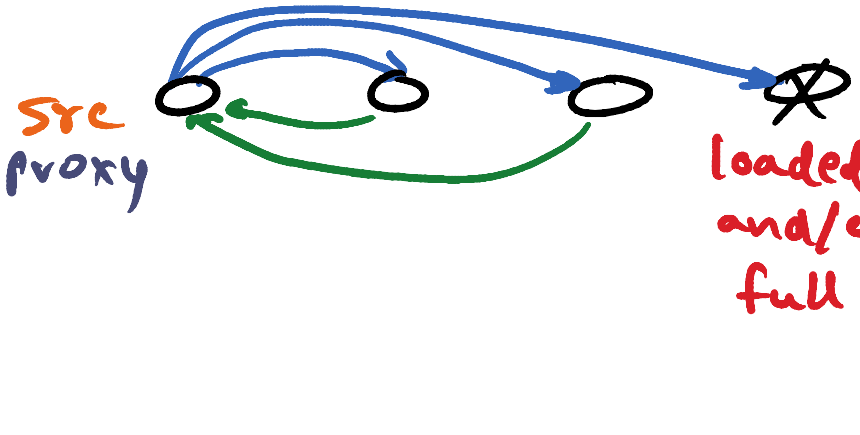
full
- max l values for key

loaded
- max β req rate for key

Coral Sloppy DHT

Coral primitives

- put (key, value)
  - node-id of proxy with content for key
  - * announcing willingness to serve as proxy
  - * put <key, value> in an "appropriate" node



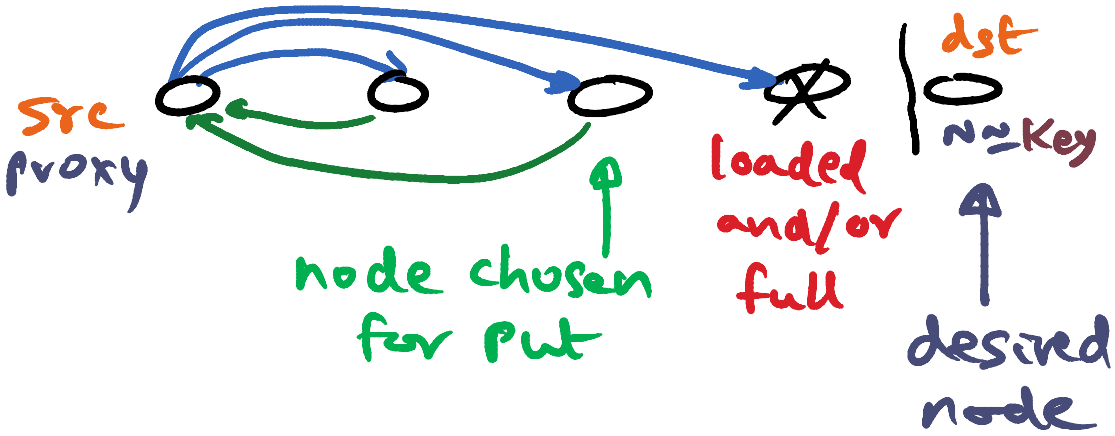
full
- max L values for key

loaded
- max β req rate for key

Coral Sloppy DHT

Coral primitives

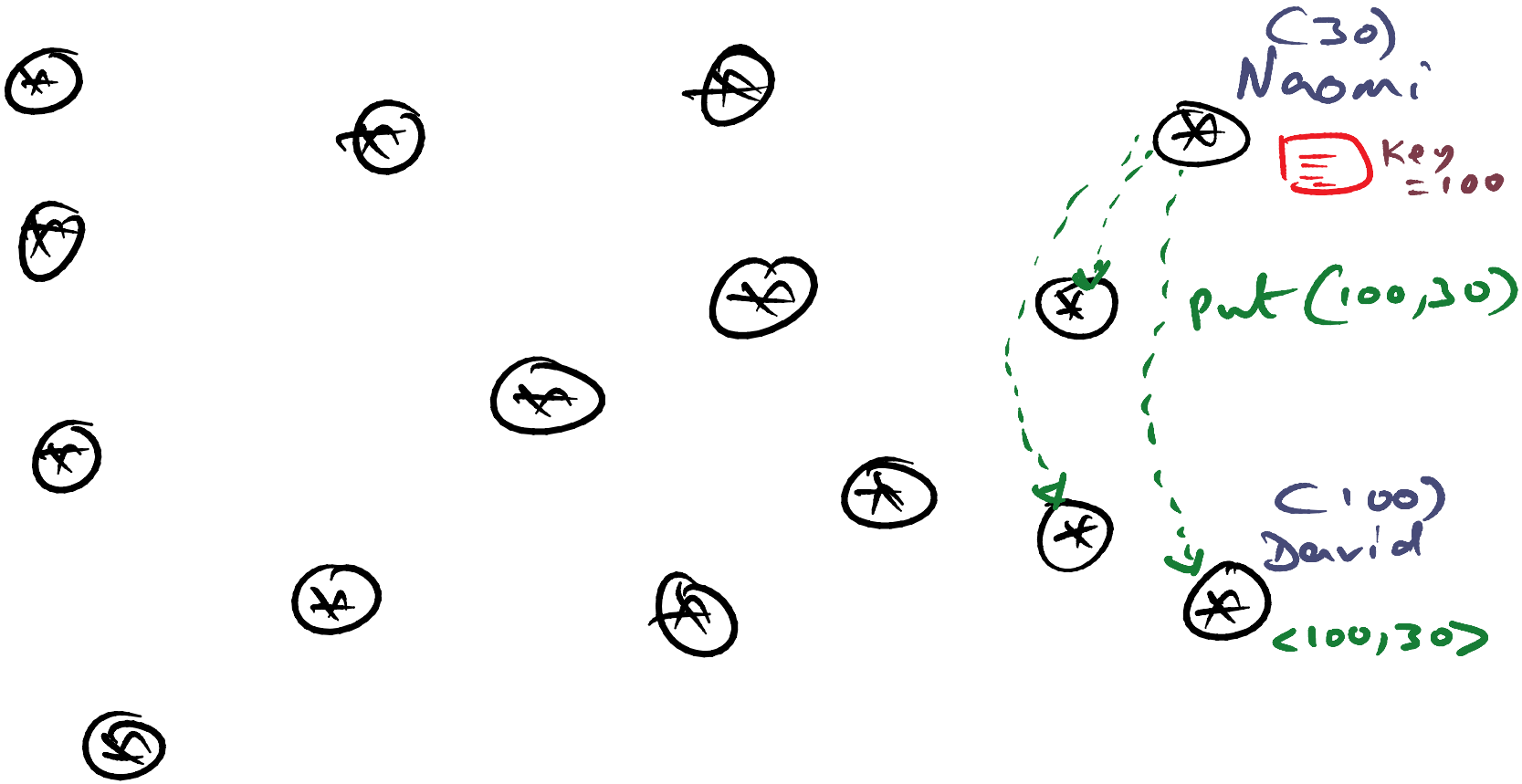
- put (key, value)
  - node-id of proxy with content for key
  - * announcing willingness to serve as proxy
  - * put <key, value> in an "appropriate" node



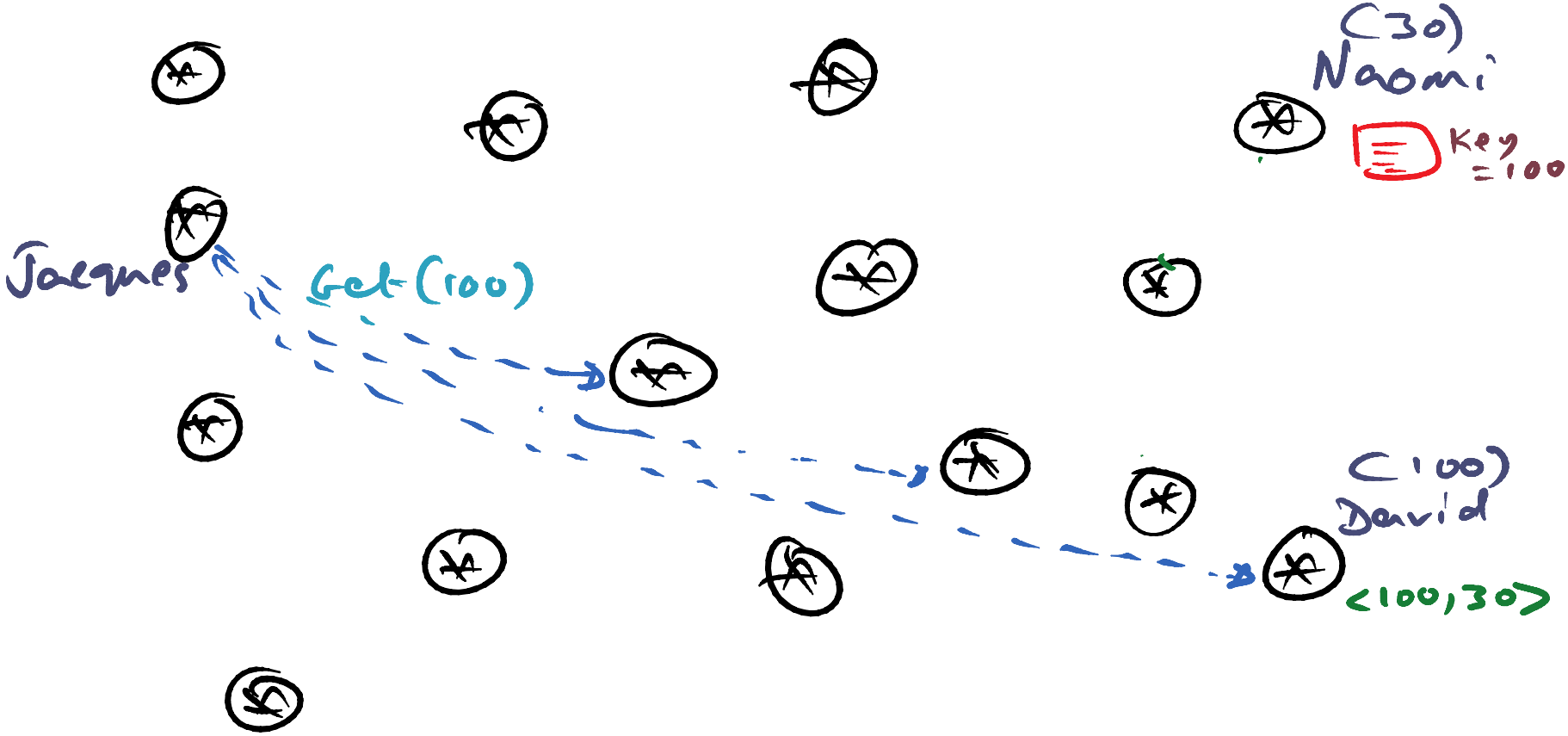
full
- max L values for key

loaded
- max β req rate for key

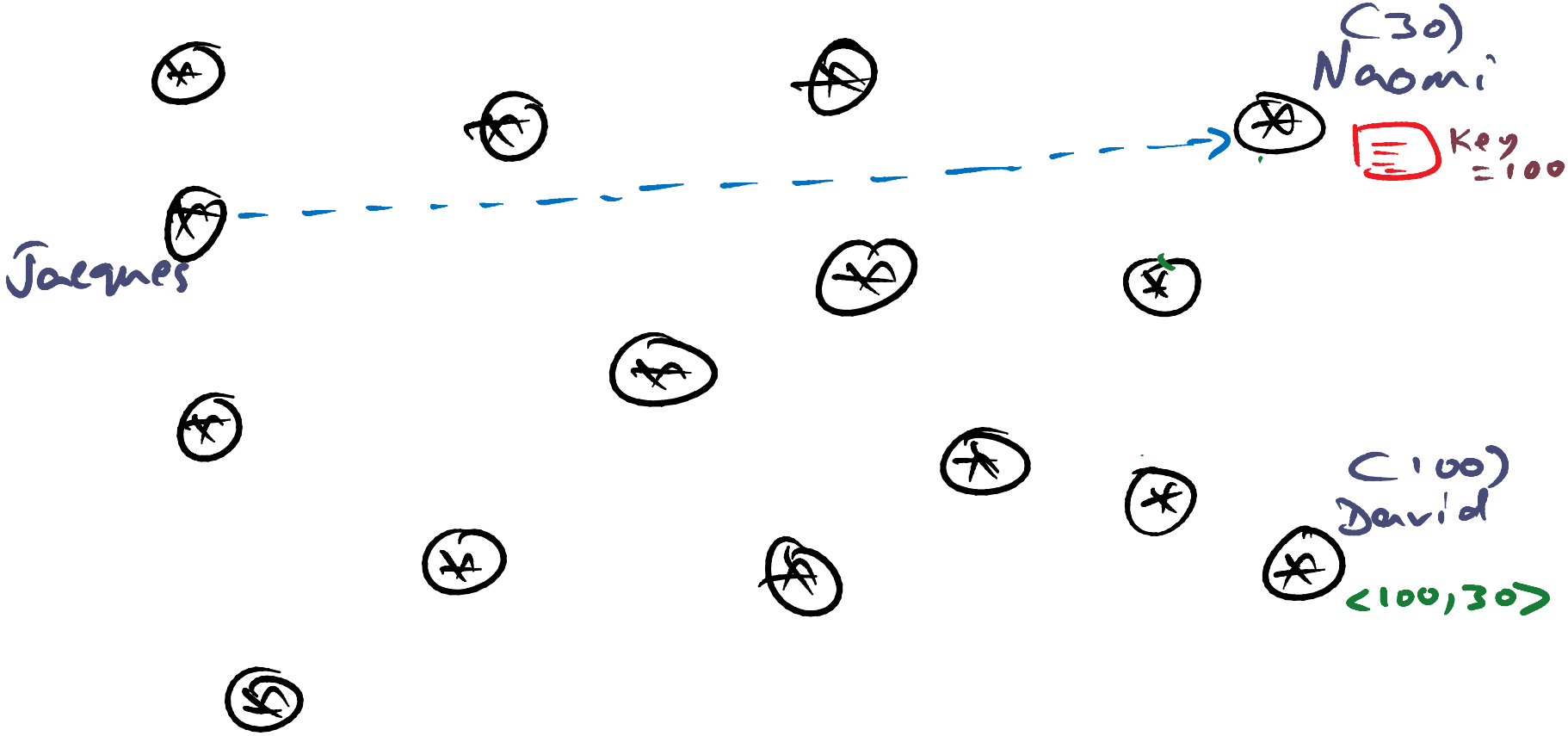
Coral in action



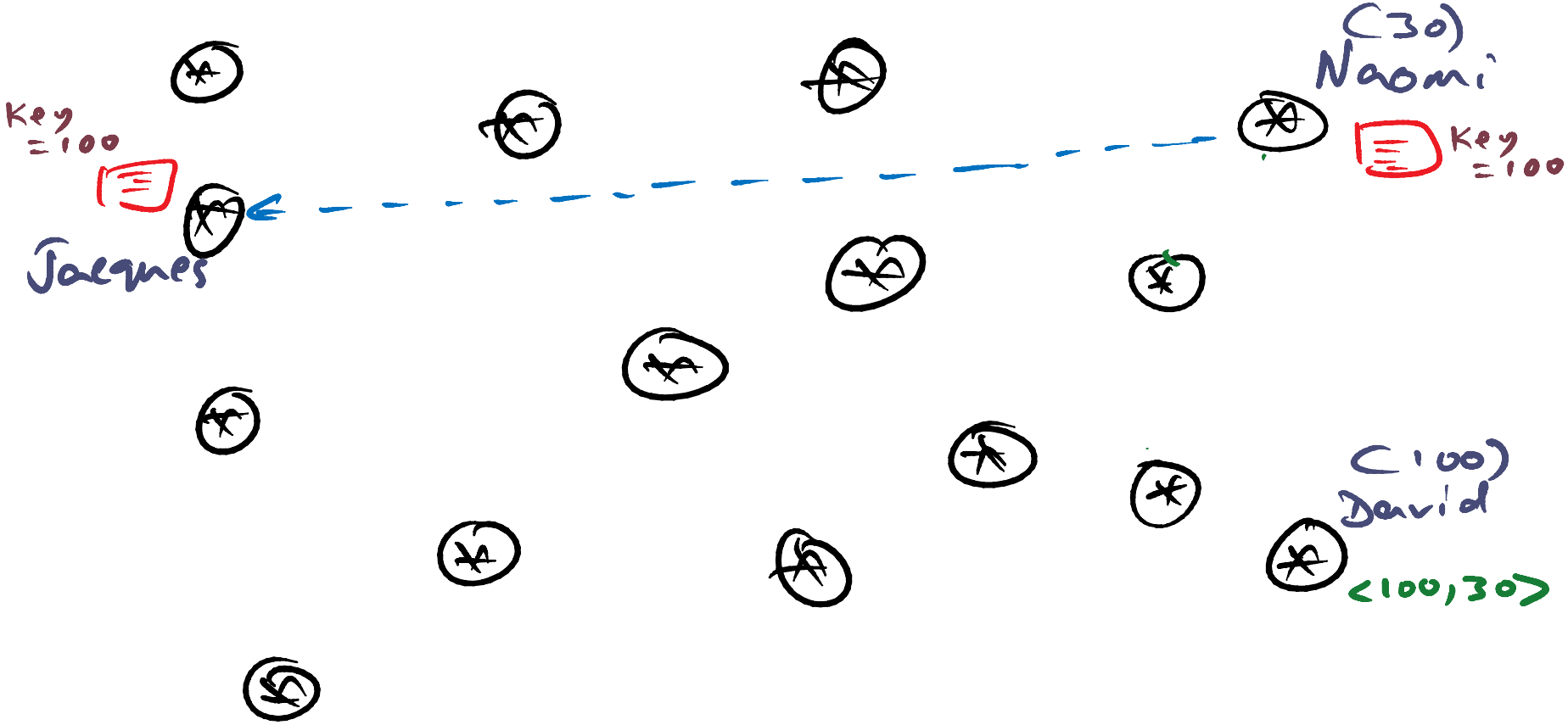
Coral in action



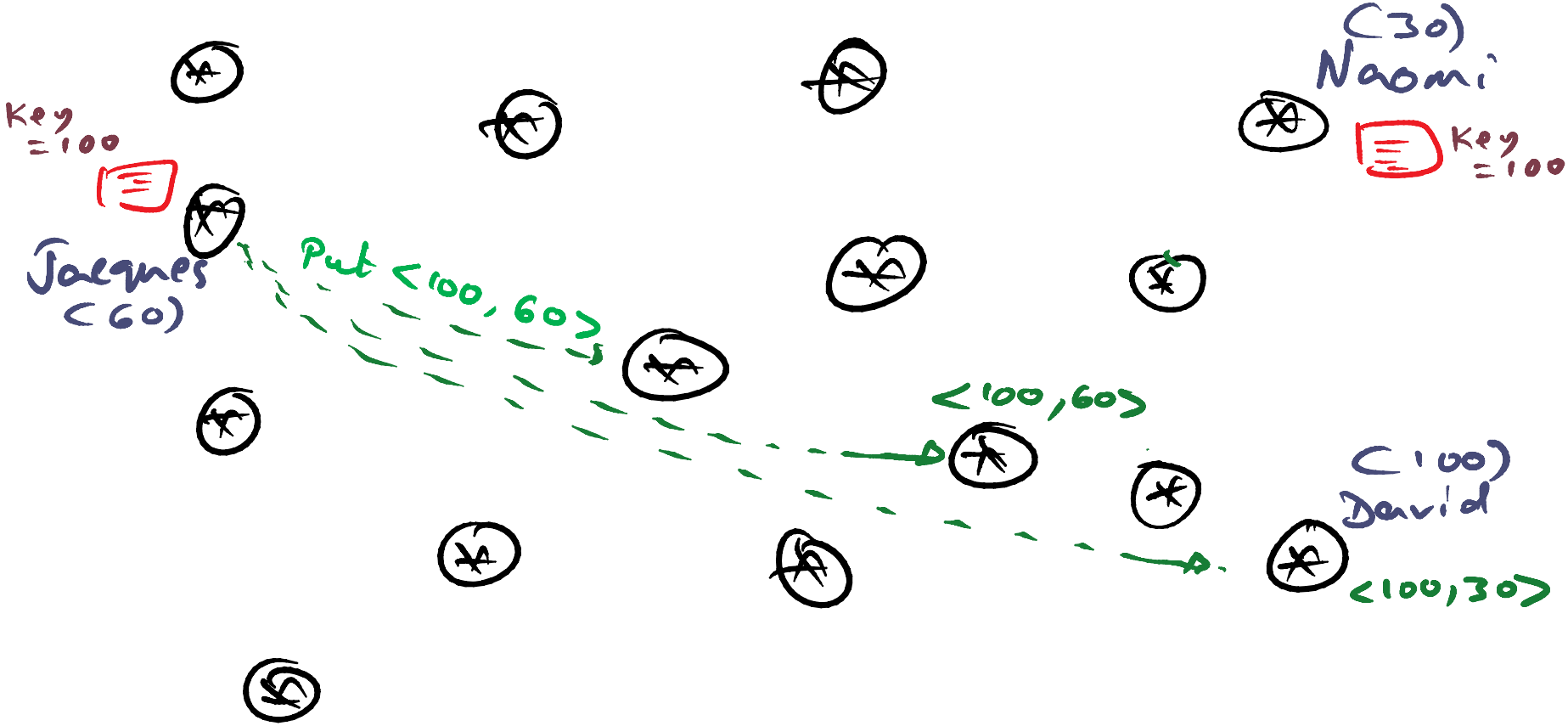
Coral in action



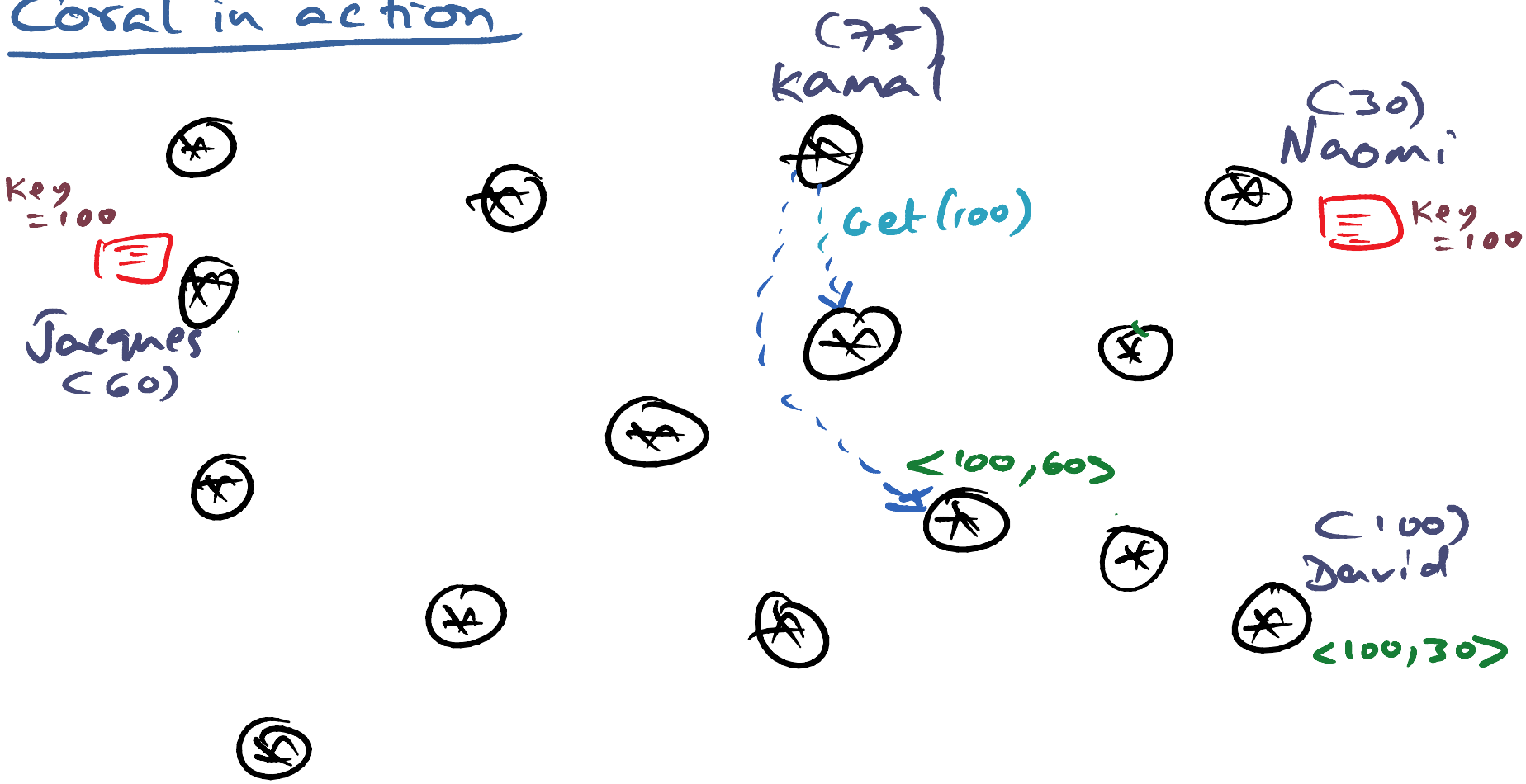
Coral in action



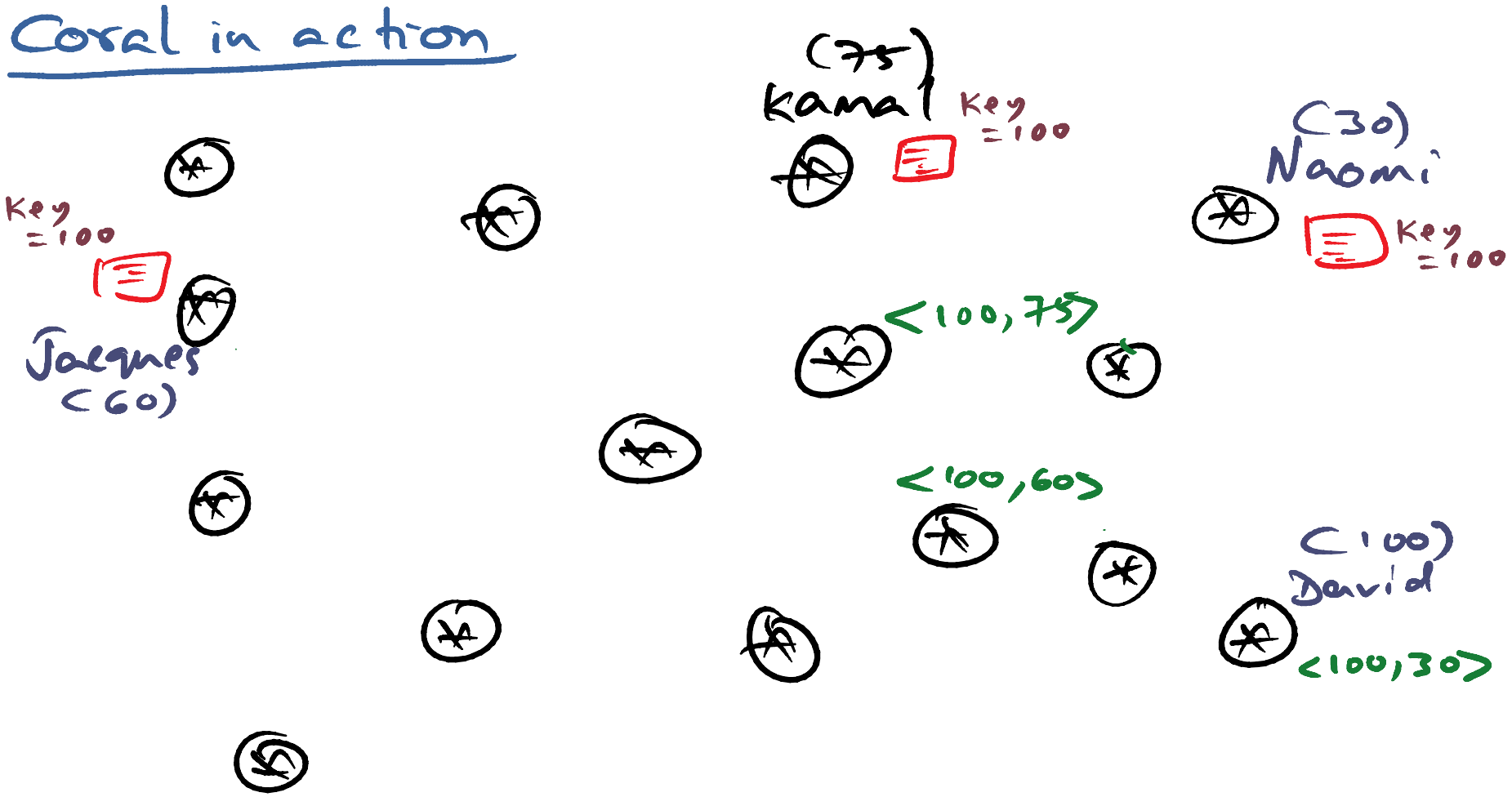
Coral in action



Coral in action



Coral in action



Coral in action

Origin server not stressed
mete server distributed

Key takeaways

- How to avoid overloading the meta-server and origin server
- Coral system democratize content generation, storage, and distribution
- Of course, commercial CDNs like Akamai do not operate this way. They are in it for the money!