Yokozuna, Scaling Solr With Riak

Todd Tyree

NoSQL Roadshow Zurich - September 19th 2013

basho

Who Am I?

- Client Services Engineer at Basho EMEA
- Large-scale web application developer (on the heavy-lifting side)
- Worked for Lonely Planet and the BBC prior to Basho
- todd@basho.com

What Is Riak?

- Key value store
- Highly available
- Distributed
- Eventually consistent
- Very scalable

What Is SOLR?

- Most popular [0]
- Lucene based
- Full-text
- Enterprise search engine

[0] http://db-engines.com/en/ranking/search+engine

What Is Yokozuna?

- Tight integration of Riak and SOLR
- Yokozuna is NOT SOLR cloud. *
- https://github.com/basho/yokozuna

*You'll get sick of hearing me say that...

Yokozuna Features

- Each Riak node runs a SOLR instance
- Automatic indexing/repair of data (on configured buckets)
- Supports plain text, XML and JSON
- Will support various binary formats when SOLR Cell is added (soon).

Using Yokozuna

- Uses SOLR query syntax, passed verbatim via Riak.
- If SOLR Distributed Search has it, Yokozuna has it. *
- No SOLR Cloud required.

* http://wiki.apache.org/solr/DistributedSearch

Write it Like Riak

Query it Like SOLR

```
curl -XPUT
     -H 'Content-Type:application/json'
     http://localhost:8098/buckets/shares/keys/2013-10-01
     -d { "adj_close": 118.26,
                "close": 118.26,
                "date": "2013-10-01",
                "high": 120.88,
                 "low": 117.8,
                "open": 119.56,
          "volume": 3536600 }
```

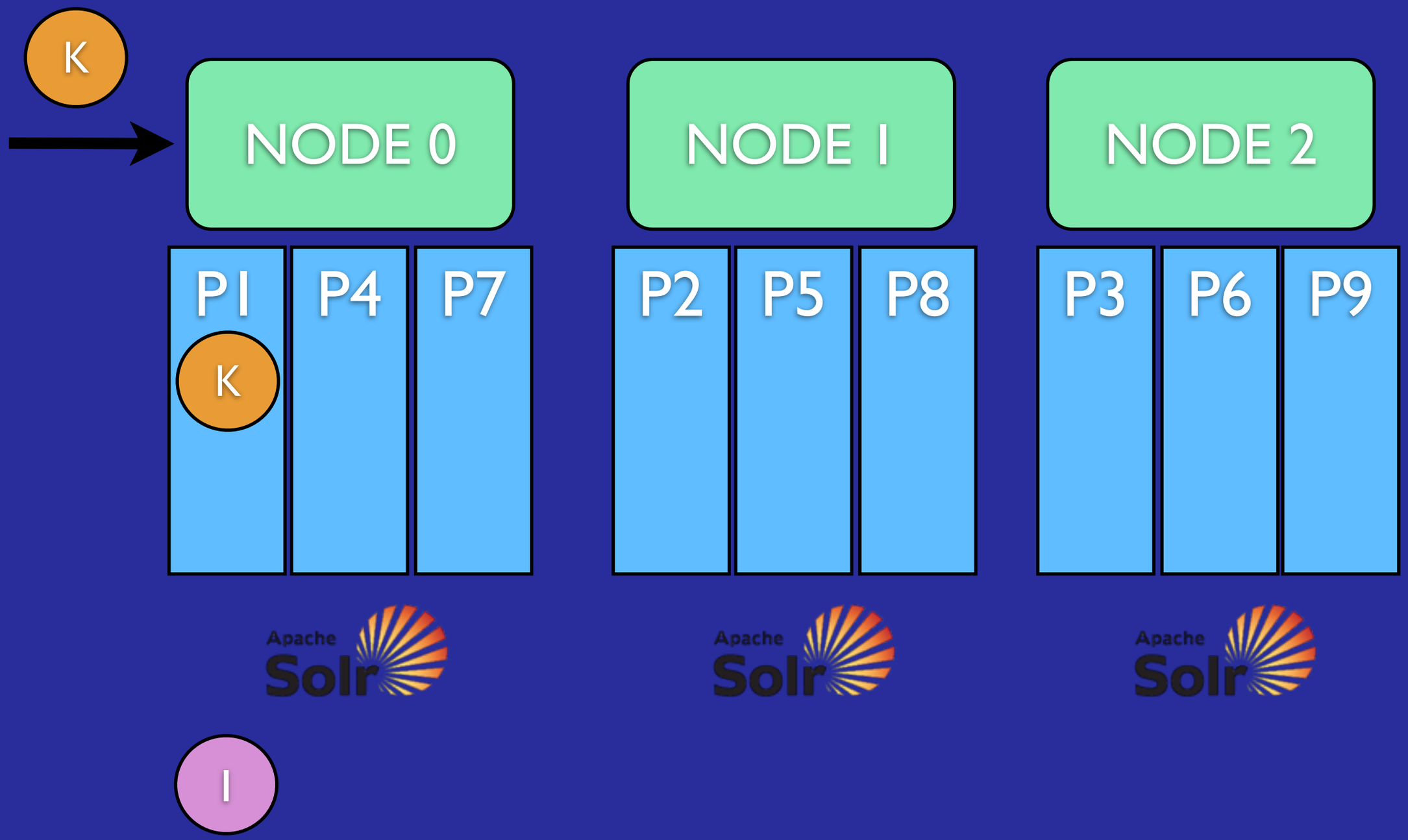
```
curl 'http://localhost:8098/search/shares_index?q=date:2013&wt=json' |
python -mjson.tool
{
    "response": {
        "docs": [
            {
                ...
                "_yz_node": "dev4@127.0.0.1",
                ...
                "_yz_rb": "shares",
                "_yz_rk": "2013-10-01",
                "_yz_vtag": "3HD2fWJwT4xAuyBWZmOuGm"
            },
                    ],
        "maxScore": 0.2972674,
        "numFound": 1,
        "start": 0
    }
...
}
```

Riak Is HIGHLY AVAILABLE

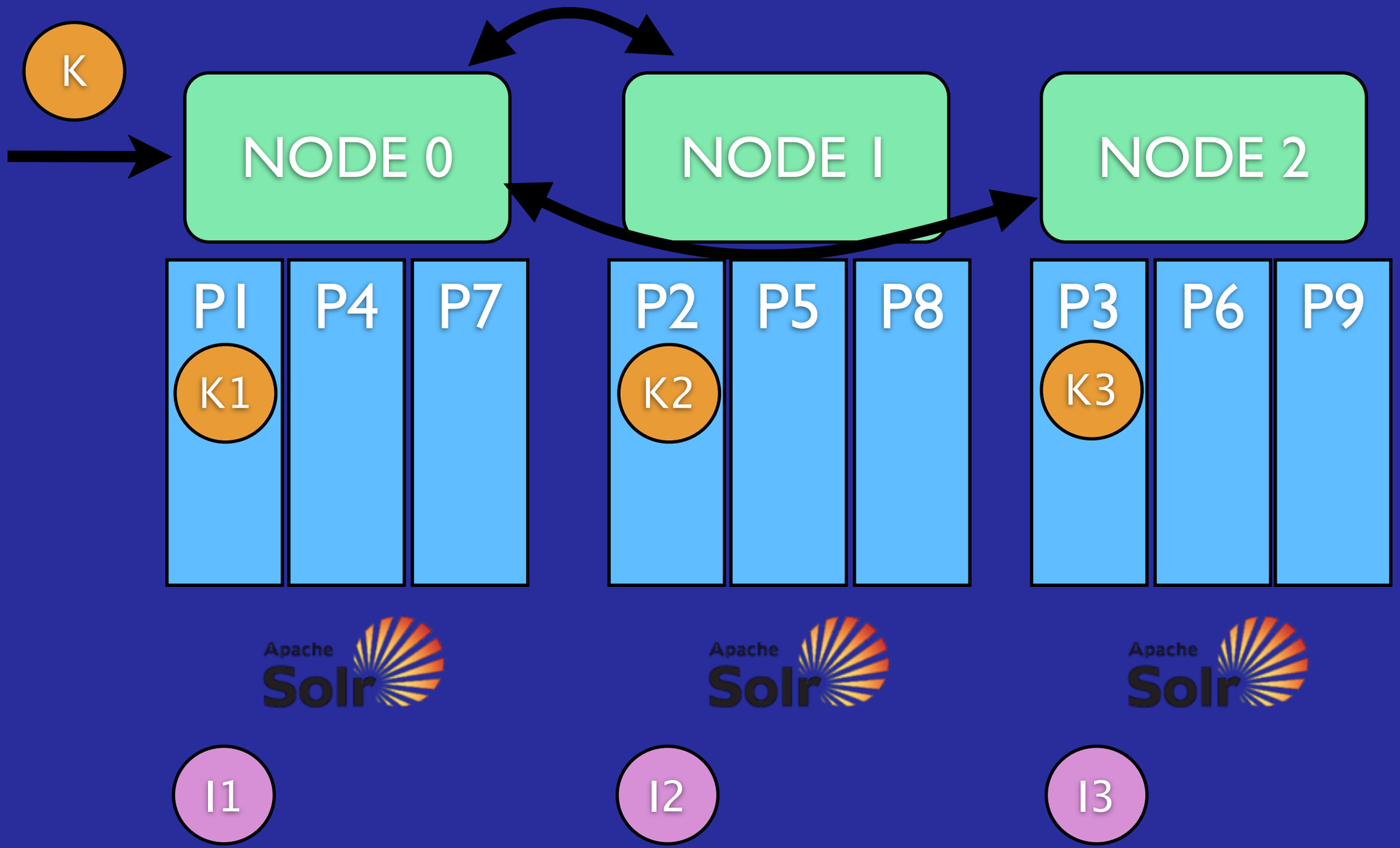
Yokozuna is HIGHLY

AVAILABLE

Writes


K
NODE 0
NODE 1
NODE 2
P1
P4
P7
P2
P5
P8
P3
P6
P9
K
Apache Solr
Apache Solr
Apache Solr
I

Replicated Writes

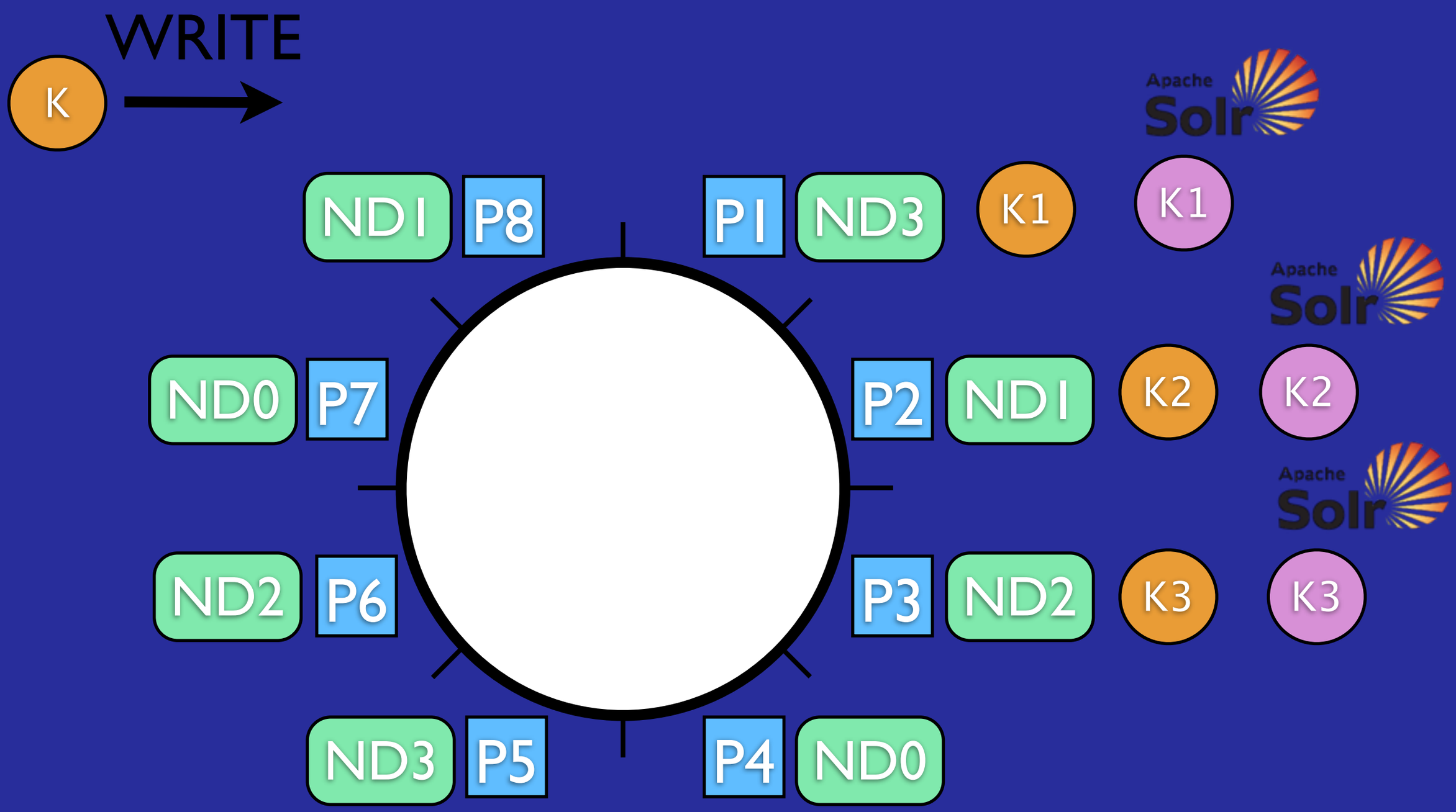


Self Healing

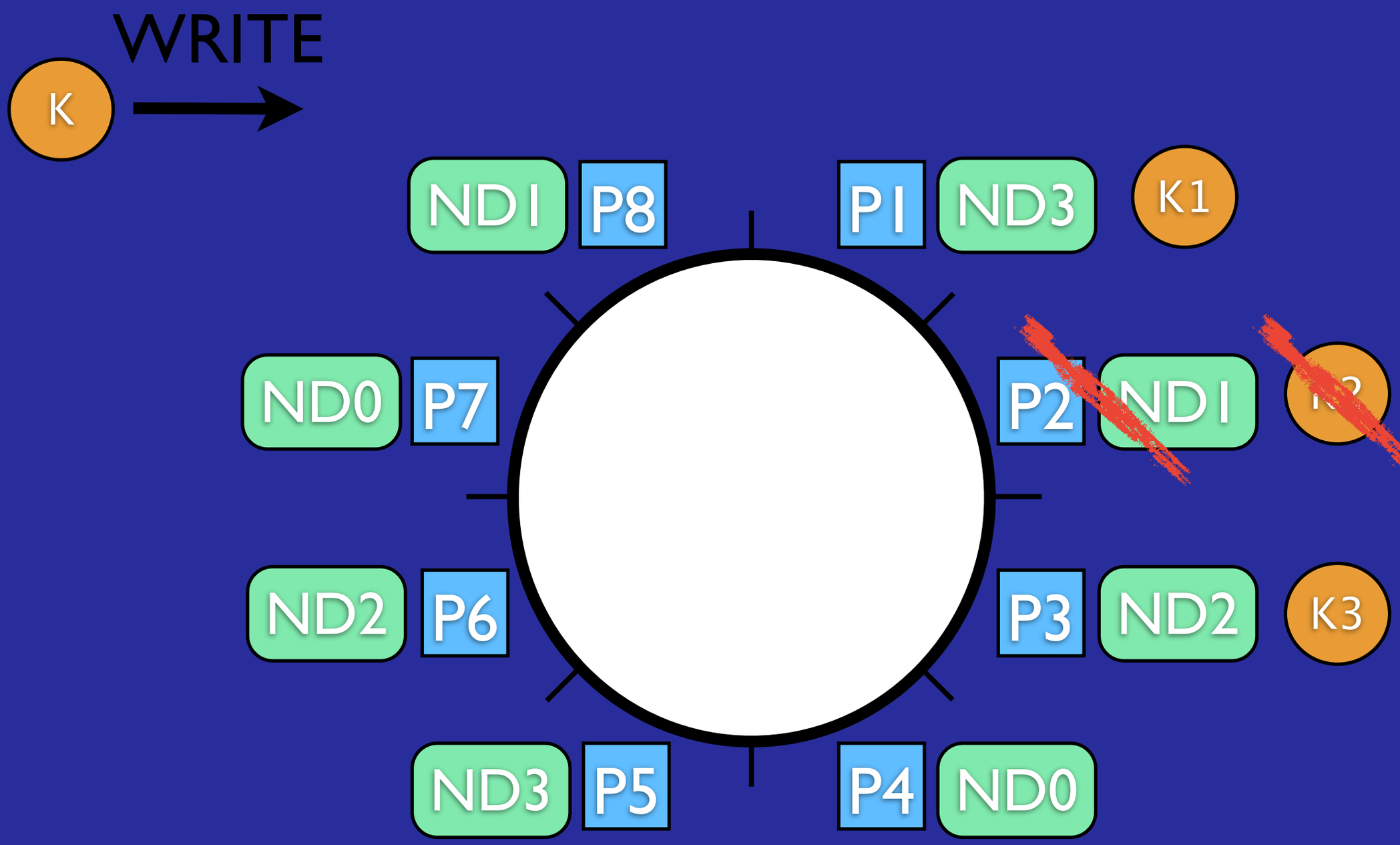
Hinted Handoff

- When a node goes down, data is written/read to/from other nominated nodes (‘fallback partitions’).
- When the node comes back up, data is handed back (‘primary partition’).
- As data is handed back, it gets indexed on its destination node.

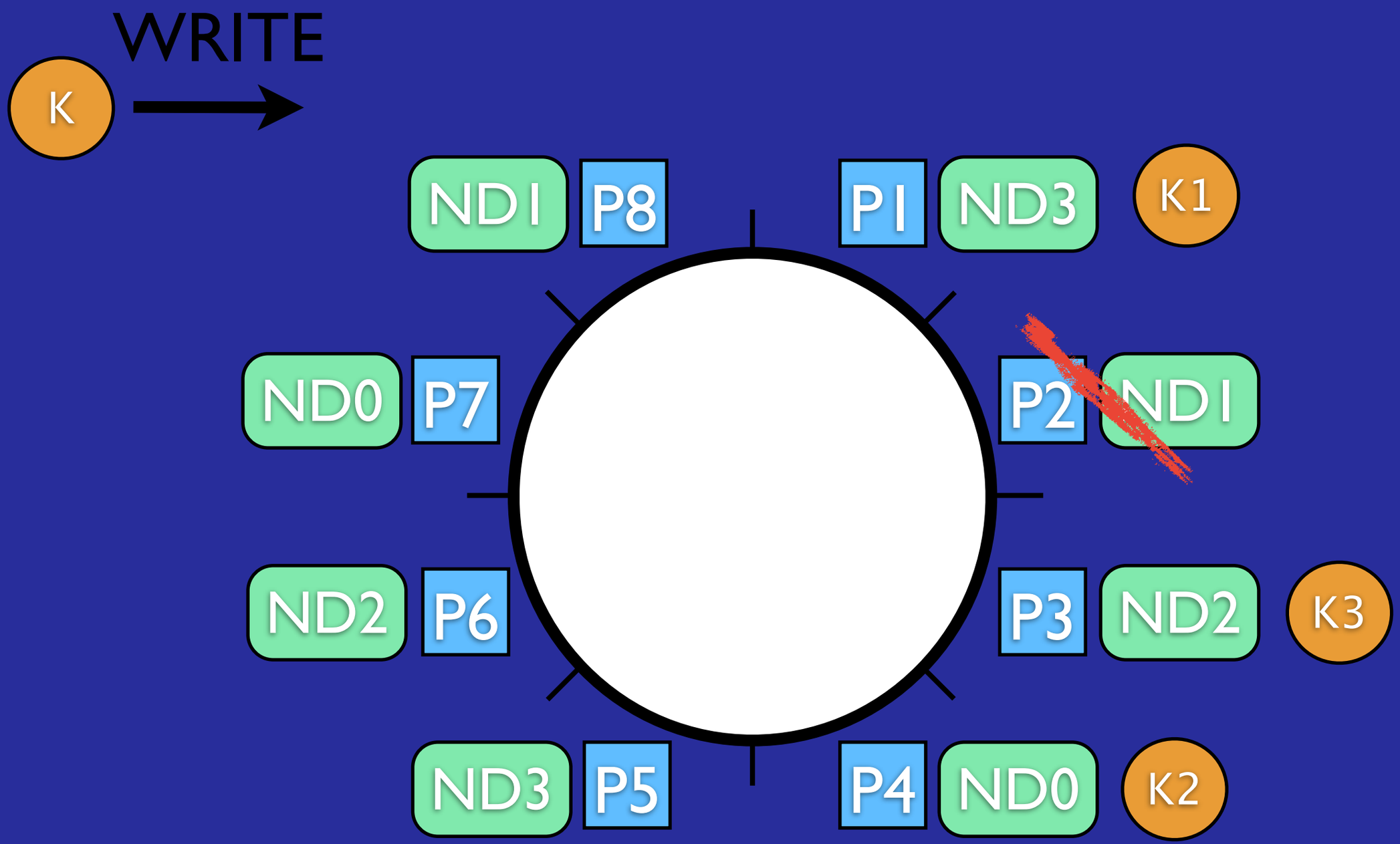
Hinted Handoff



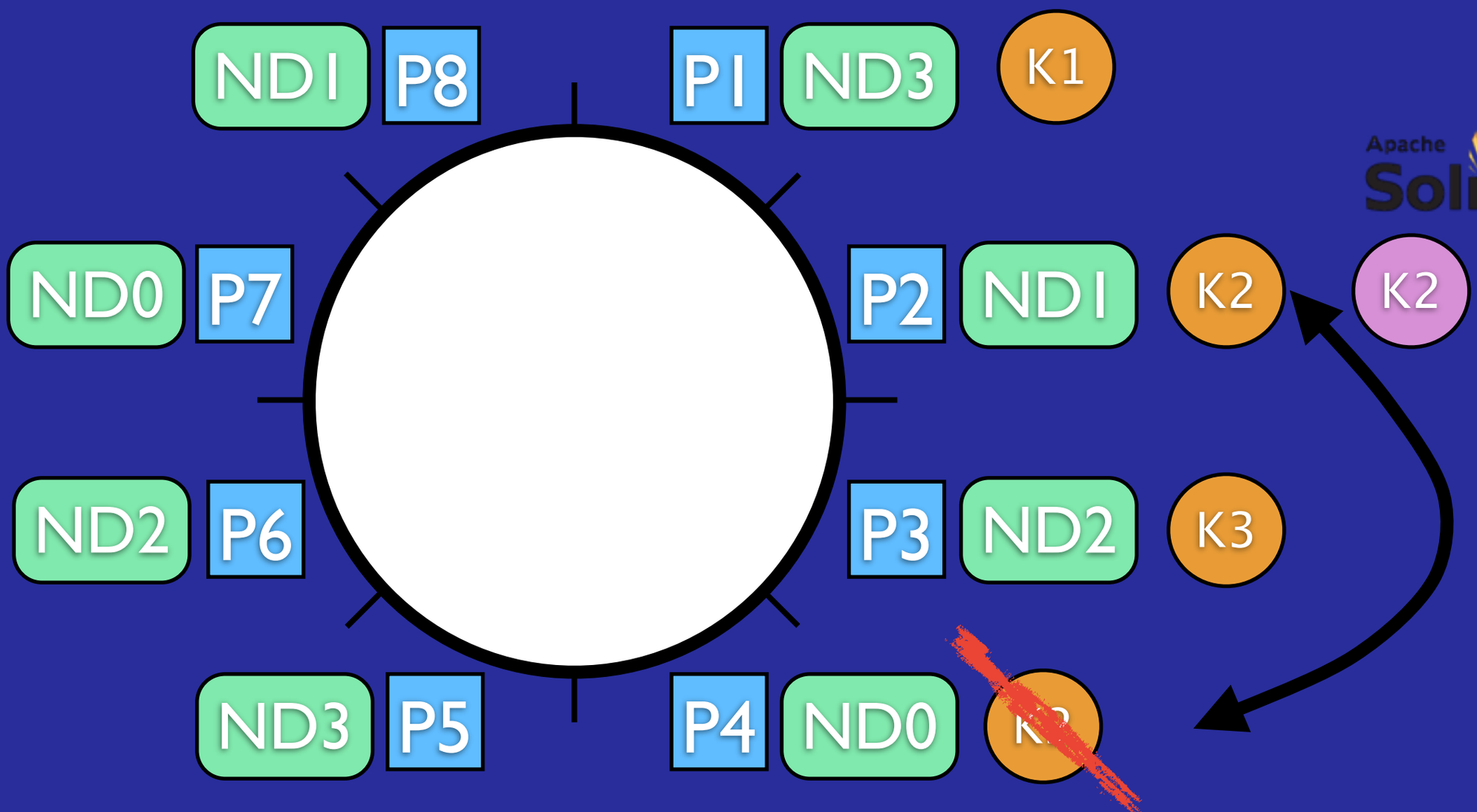
Hinted Handoff



Hinted Handoff



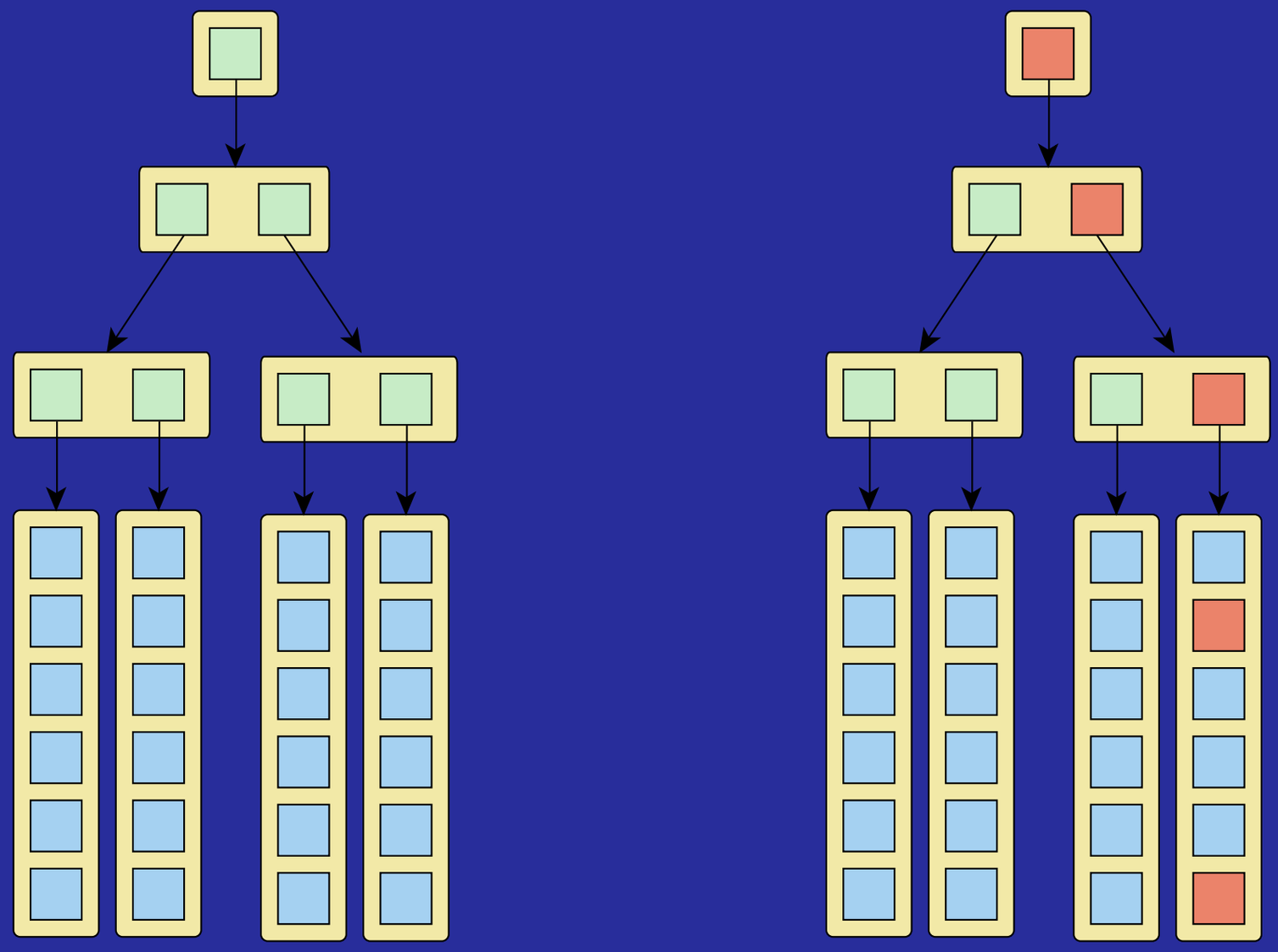
Hinted Handoff



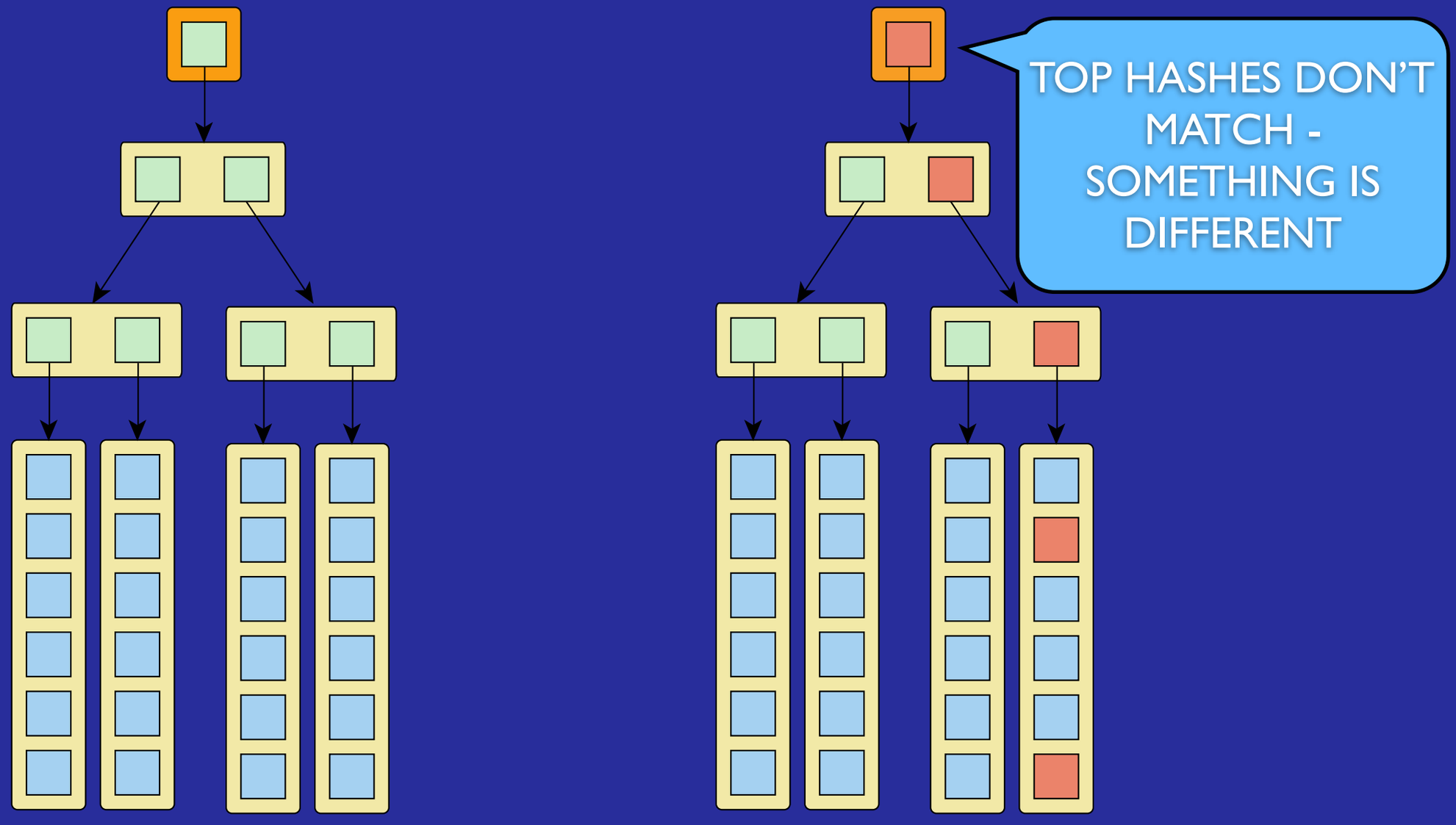
Active Anti Entropy

- 2 Systems - a greater chance for inconsistency
- Files get truncated/corrupted/lost
- Segfaults happen at EXACTLY the wrong moment

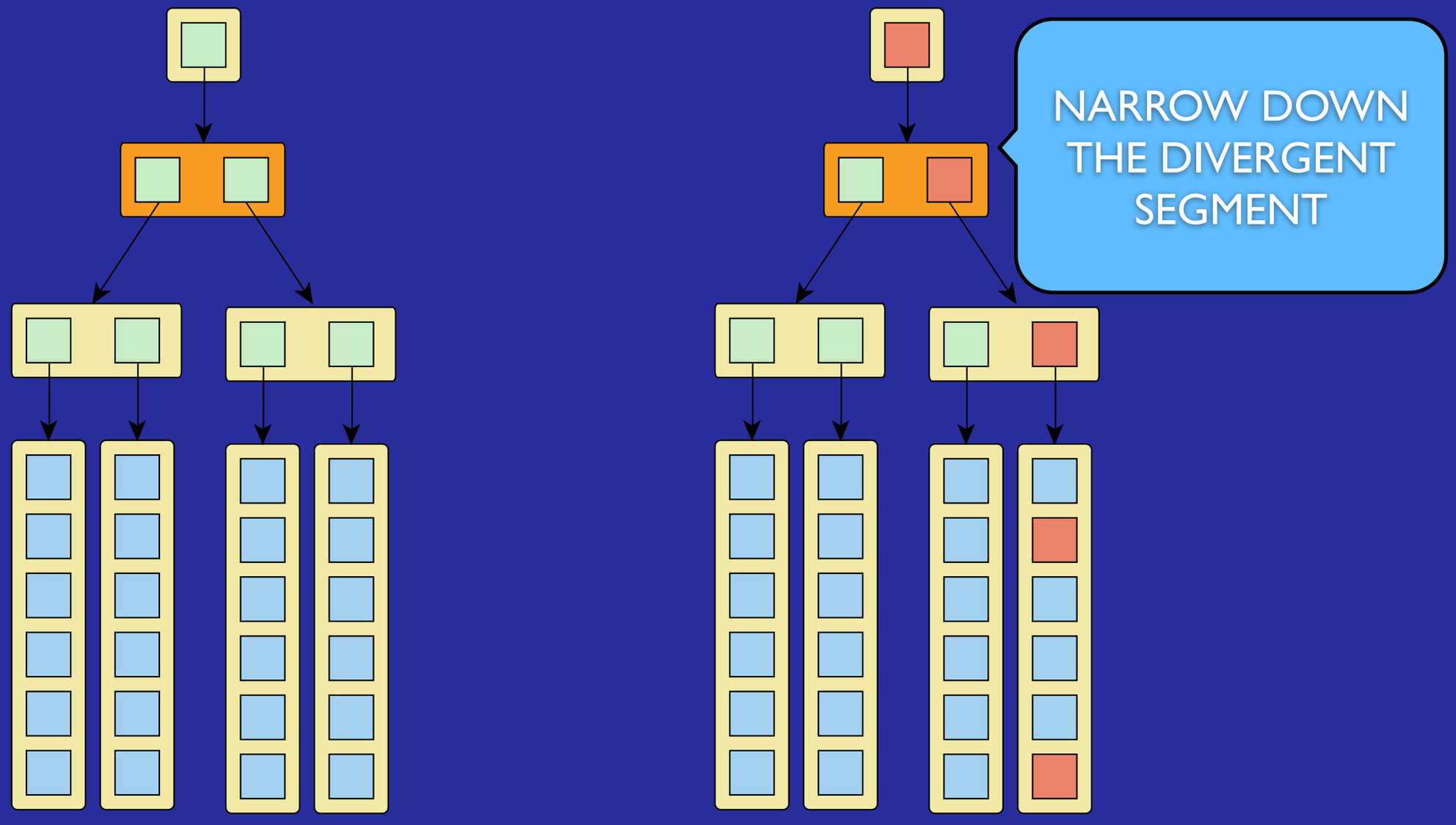
AAE - Exchange



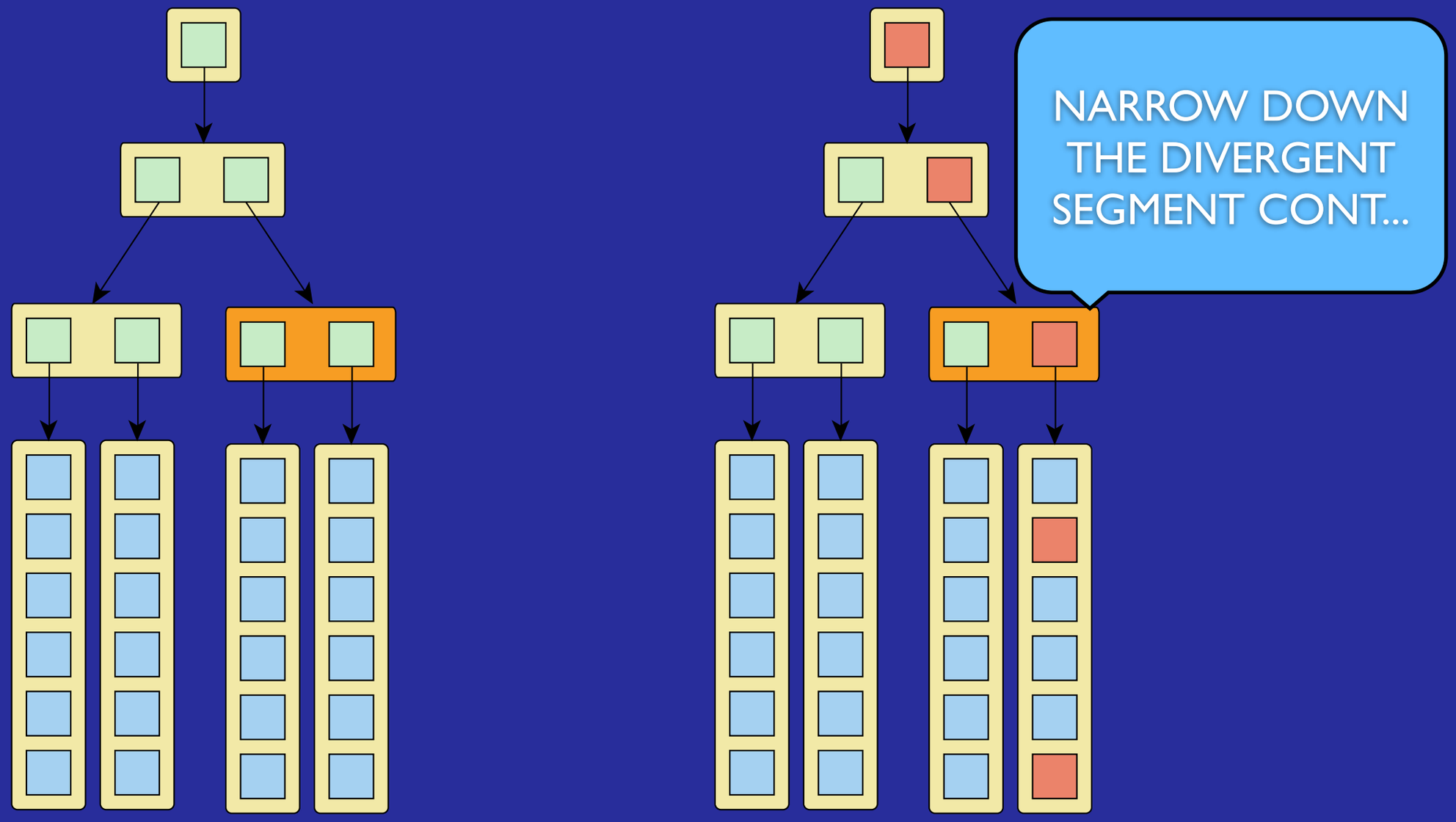
AAE - Exchange



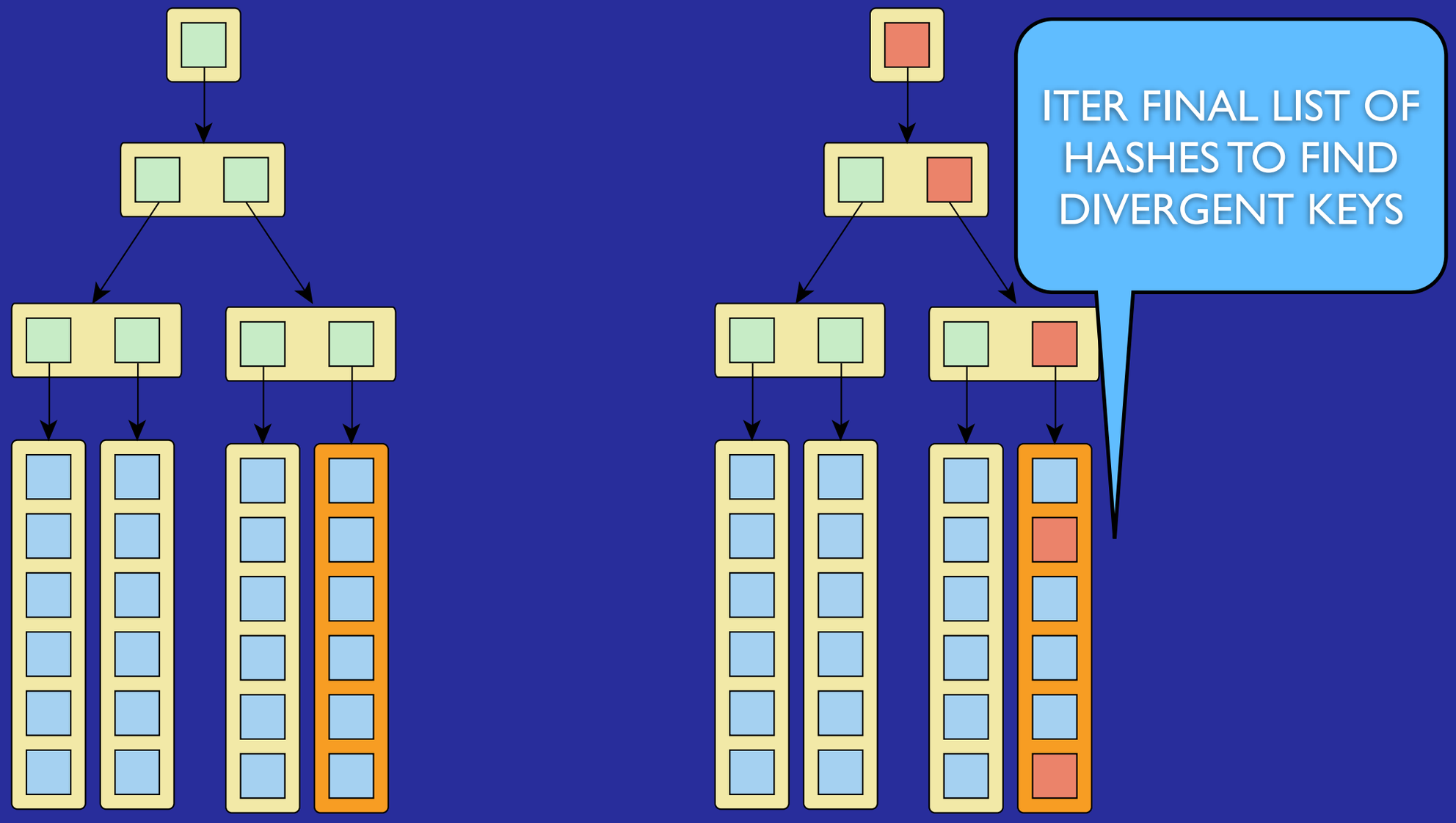
AAE - Exchange



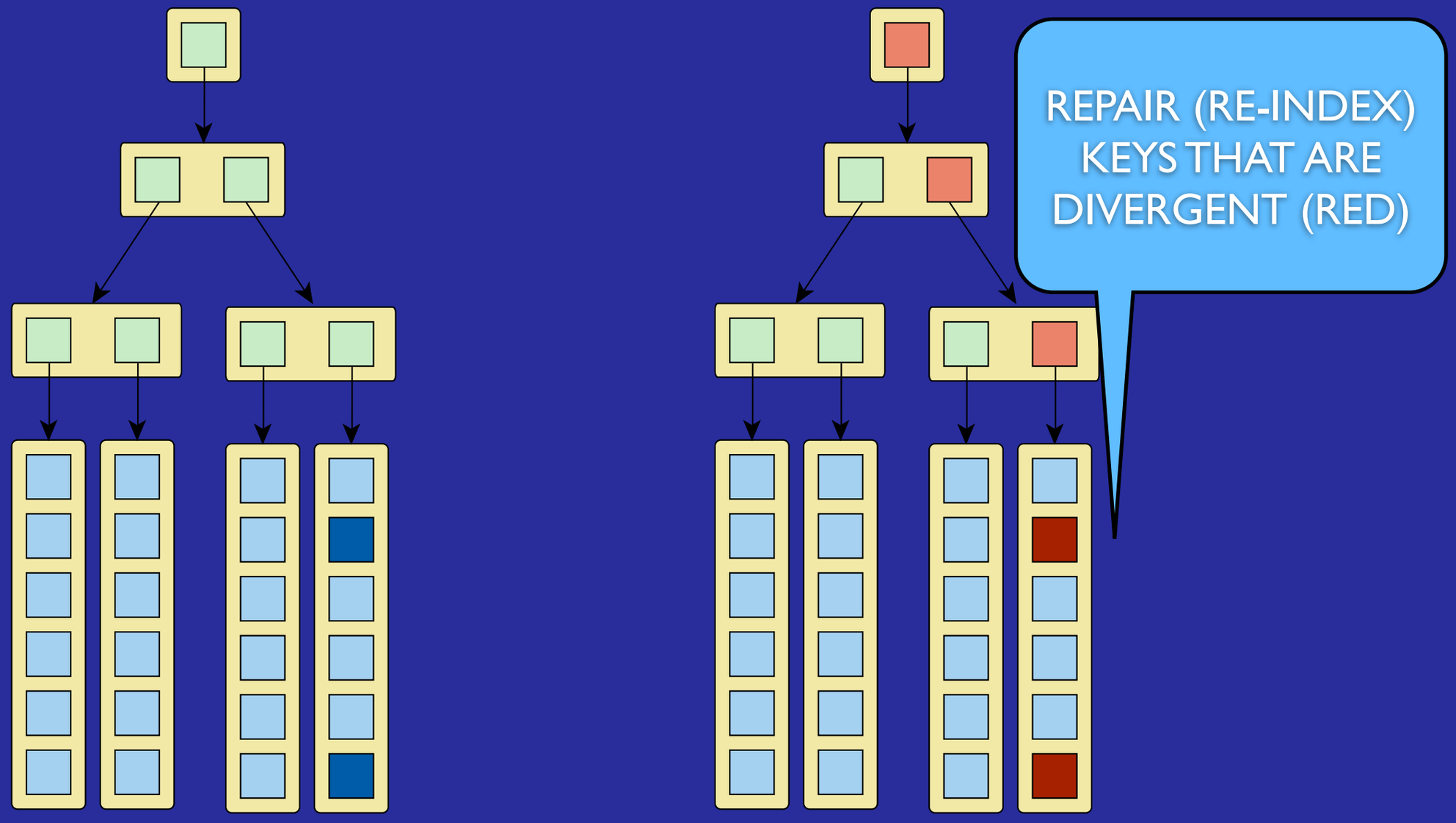
AAE - Exchange



AAE - Exchange



AAE - Exchange



Read Repair

- Replicas may not agree or may get lost.
- Replica values are checked during read.
- If they disagree, they are fixed.
- New values are sent to each replica.

Nodes Go Down

- Replication allows for query availability
- You only need one index replica
- Yokozuna will refuse the query without one replica

There are myriad failure scenarios from the completely

OBVIOUS

to the nearly

INVISIBLE

Develop for
DETECTION
and
REPAIR,
not for
PREVENTION

Thank You!

https://github.com/basho/yokozuna