Yokozuna

NoSQL Search Amsterdam 2013

Me

What is Yokozuna?

Source: http://katrinainjapan.files.wordpress.com/2013/08/yokozuna.jpg

Sumo Wrestling Term

“Horizontal rope. The **top rank** in sumo, usually translated Grand Champion. The name comes from the rope a yokozuna wears for the dohyō-iri.”

Source: http://en.wikipedia.org/wiki/Glossary_of_sumo_terms


riak

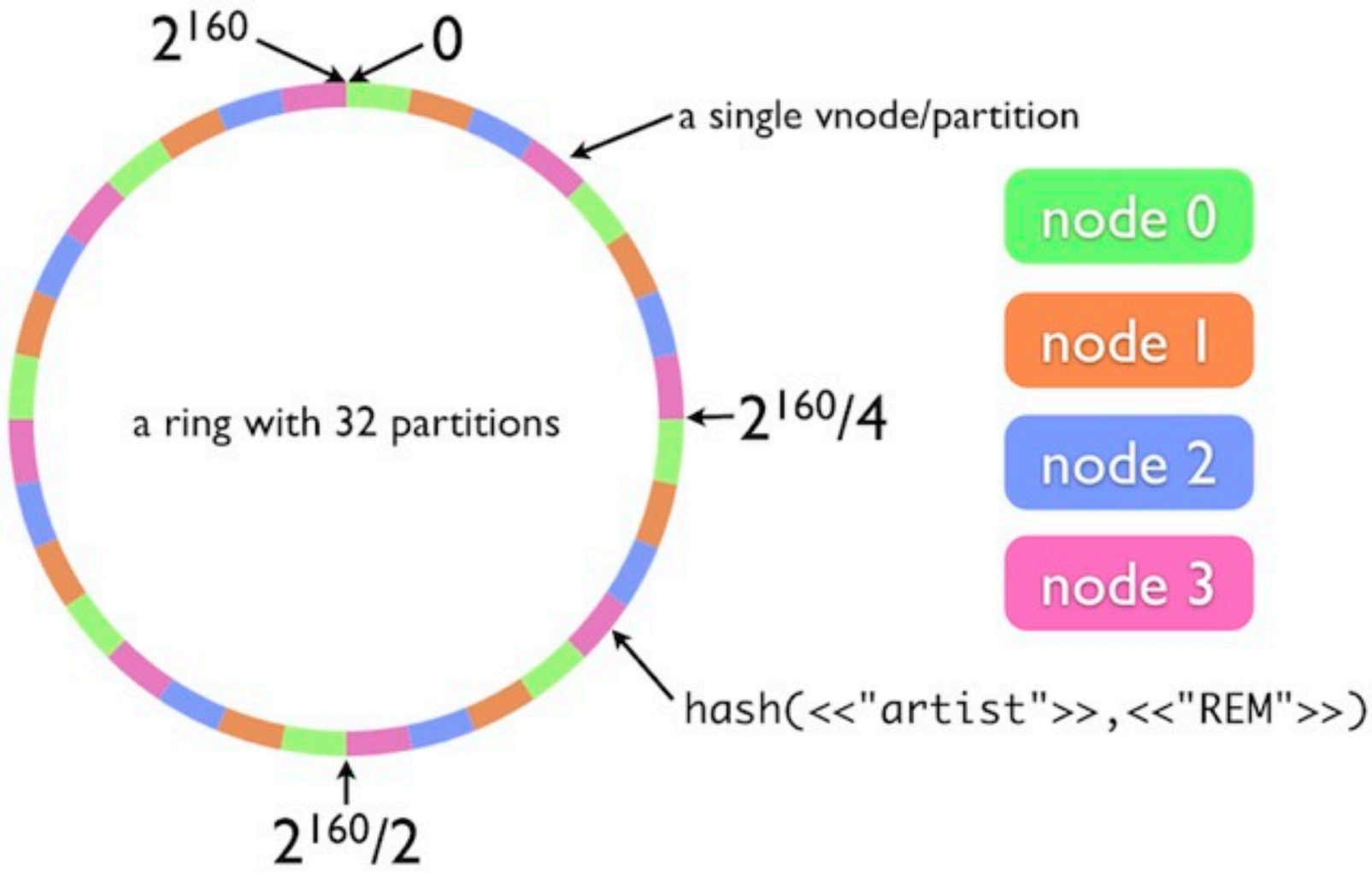

Apache
Solr


riak

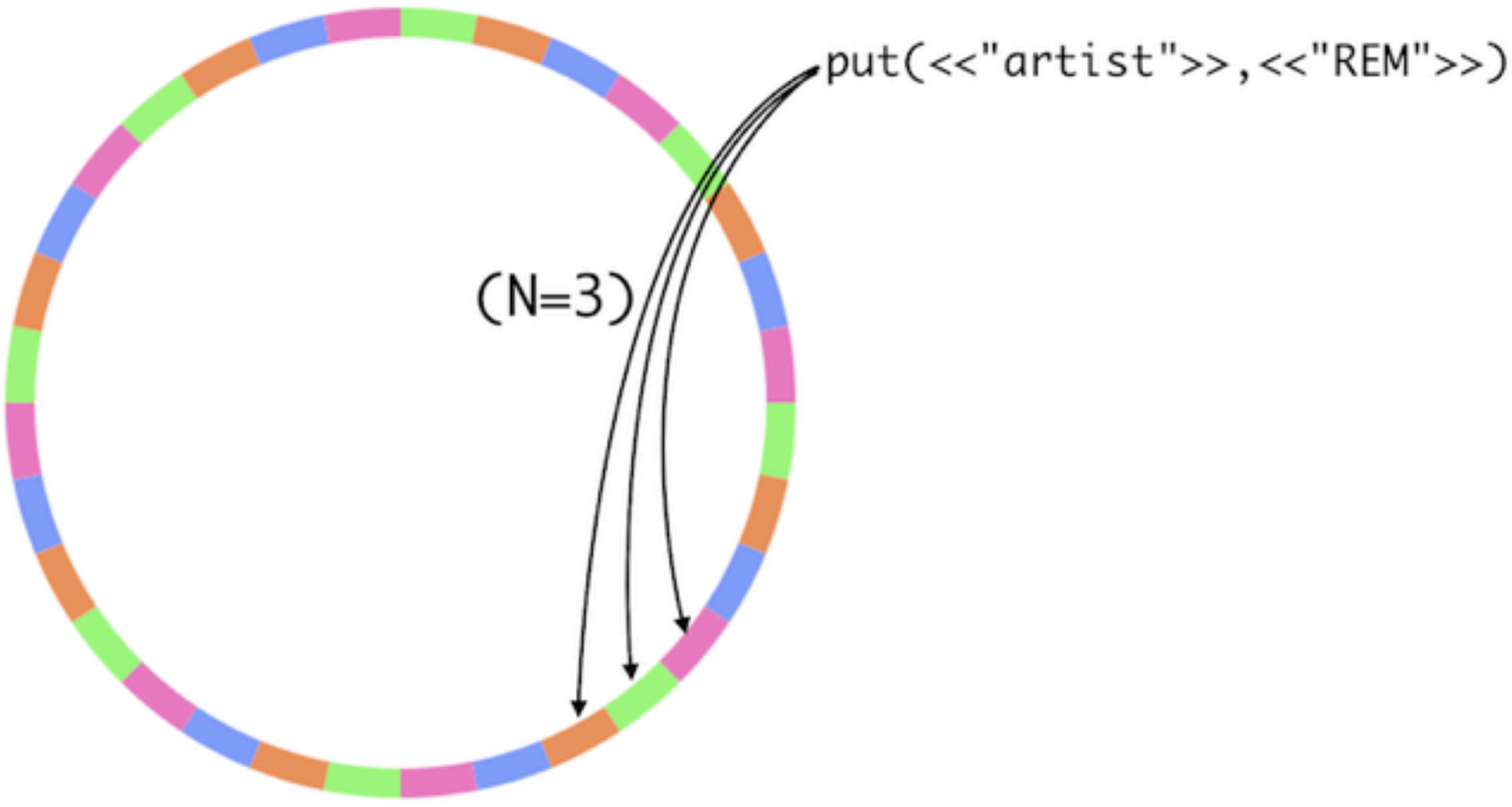
Riak

+ Amazing KV Store
+ Distributed
+ Highly Available
+ Easily Scalable
+ Self Healing
+ Open Source

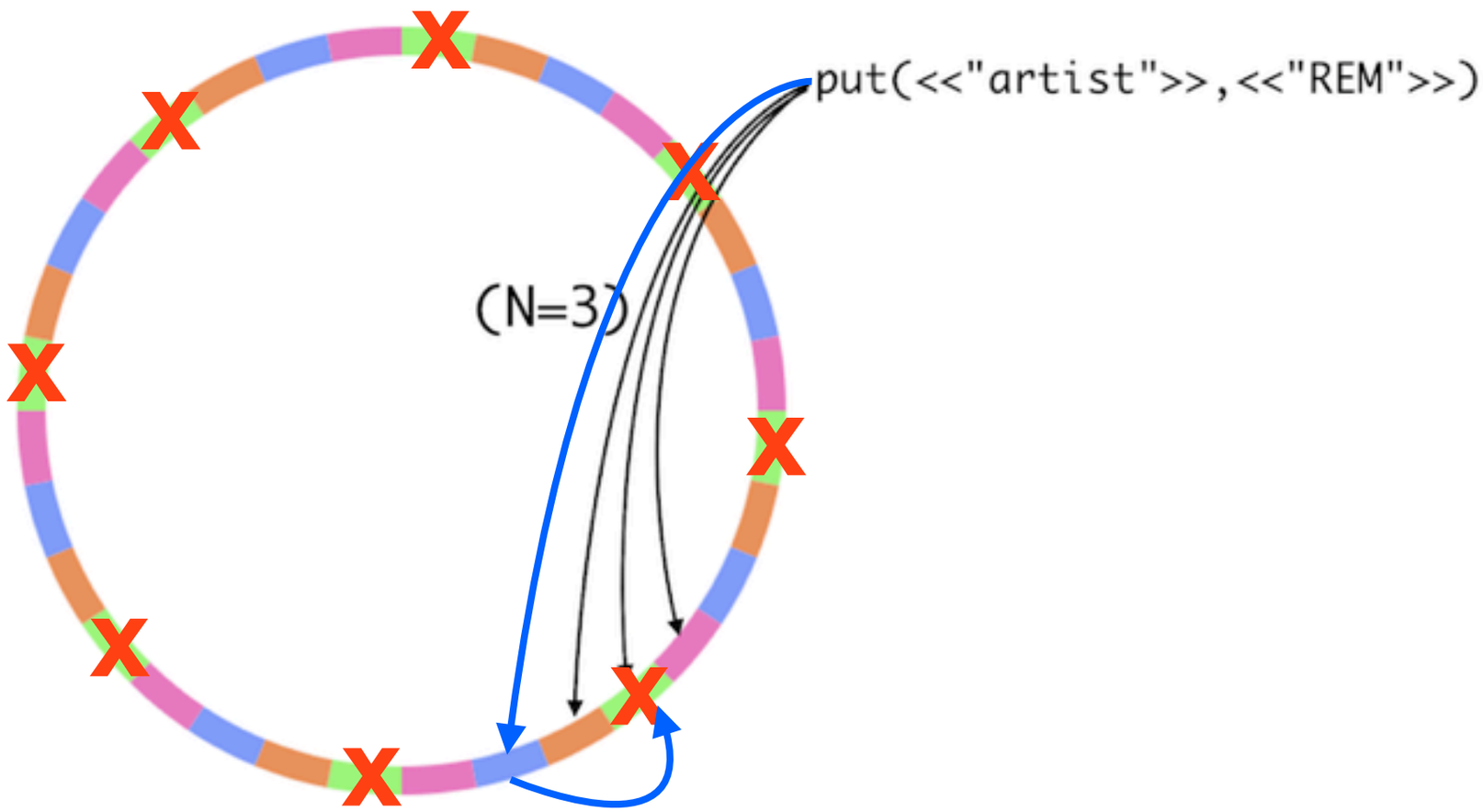
Consistent Hashing



Replication



Self Healing



Riak Questions?

Riak

- Limited Query Ability
- Query Performance
- Index Entropy Repair
- Limited Full Text Search


Apache
Solr

Solr

Not Solr Cloud

+ Amazing Query Support
+ Robust Inverted Index
+ Near Real-time Indexing
+ Sophisticated Analyzers
+ Language Support
+ Features: facets, highlighting, storing, sorting
+ Gold Standard

Solr

Not Solr Cloud

- HA is secondary to search
- Manual everything
- No entropy
- Key value


riak


Apache
Solr

Combine FTW

- Amazing KV Store
- Distributed
- Highly Available
- Easily Scalable
- Self Healing
- Amazing Query Support
- Sophisticated Analyzers
- Language Support
- Great Features

Why Yokozuna?

What about Riak Search?

Riak Search

+ Term-based *sometimes* better
+ Pure Erlang
+ Relatively small code base

Riak Search

- Large result sets (> 100k)
- Memory pressure
- Lack of facet query
- Language support
- Basic analyzers
- Entropy & Repair

Integrate Search

- Riak Search & Basho can't keep pace with Lucene/Solr
- Don't re-invent the search
- Basho's strength is *distributed databases*

What About 2i?

- Query one index [field] at a time
- No notion of ranking
- Range and exact term only
- Must use leveldb or memory
- No full text search
- Basic types - string and int

Goals of Yokozuna

Goals of Yokozuna

- Provide robust query against KV data
- Require minimal work from user
- Don’t concern user with distribution
- Replace Riak Search (and then some)

How does it work?

Yokozuna

- Erlang application like Riak KV
- Erlang supervisor for Solr process

Solr & JVM

- Configurable jvm_args in riak.conf

```
yokozuna.solr_jvm_args = -Xms256m -Xmx256m -XX:
+UseStringCache -XX:+UseCompressedOops
```

Indexing

- Each Riak node runs a Solr instance
- Store schema; create index; associate bucket
- Data is automatically indexed as it is added
- Index repair is provided through AAE
- Extendable through custom extractors

Store Schema

```
<field name="commit_repo" type="string" indexed="true"
stored="true"/>
<field name="commit_hash" type="string" indexed="true"
stored="true"/>
<field name="commit_author" type="string" indexed="true"
stored="true"/>
<field name="commit_dt" type="date" indexed="true"
stored="true"/>
<field name="commit_subject" type="text_general"
indexed="true" stored="true"/>
<field name="commit_body" type="text_general"
indexed="true" stored="true"/>

curl -XPUT -i -H 'content-type: application/xml'
'http://localhost:10018/yz/schema/cls' --data-binary
@cls.xml
```

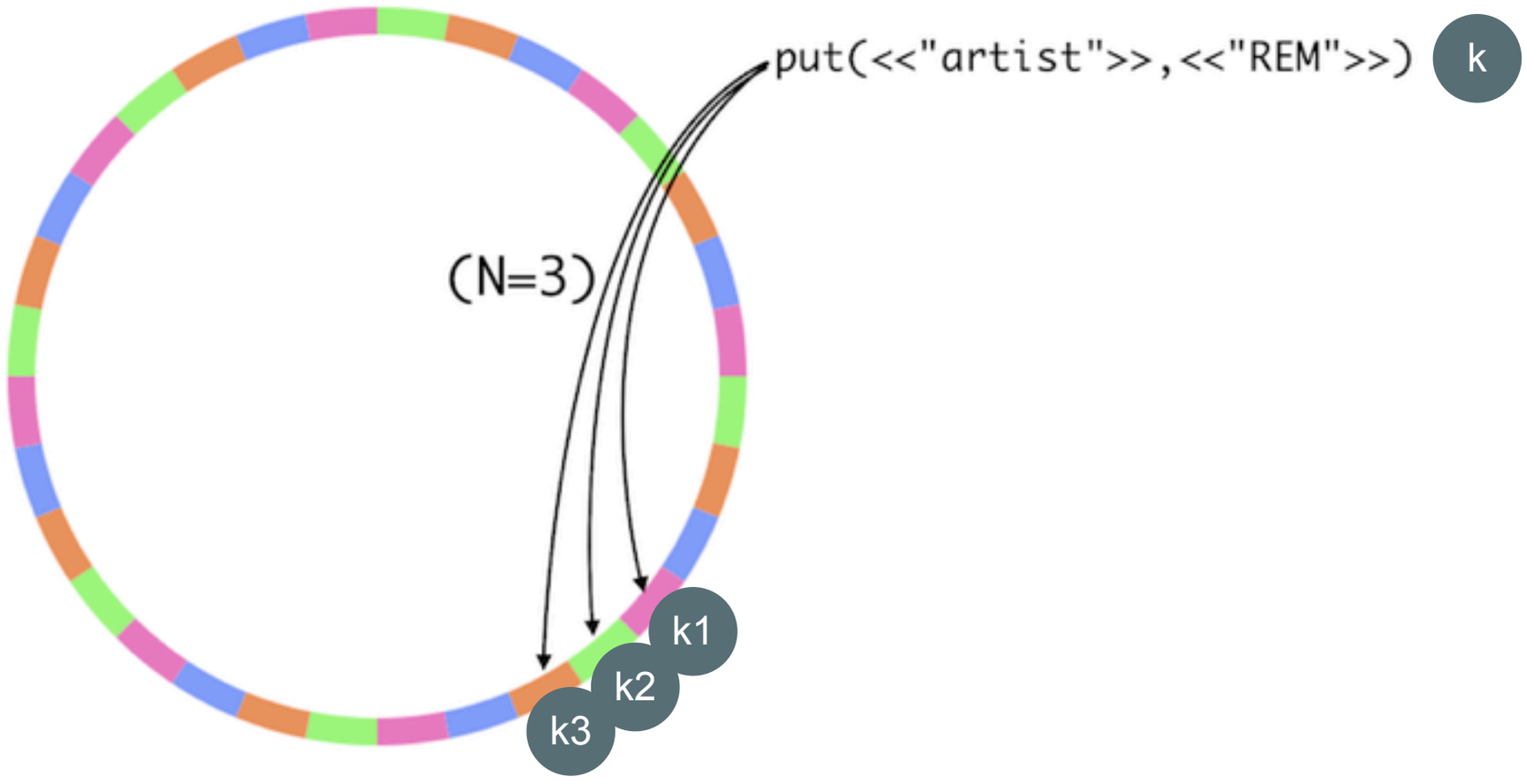
Create Index

```
curl -XPUT -i -H 'content-type: application/json'
'http://localhost:10018/yz/index/cls' -d
'{"schema":"cls"}'
```

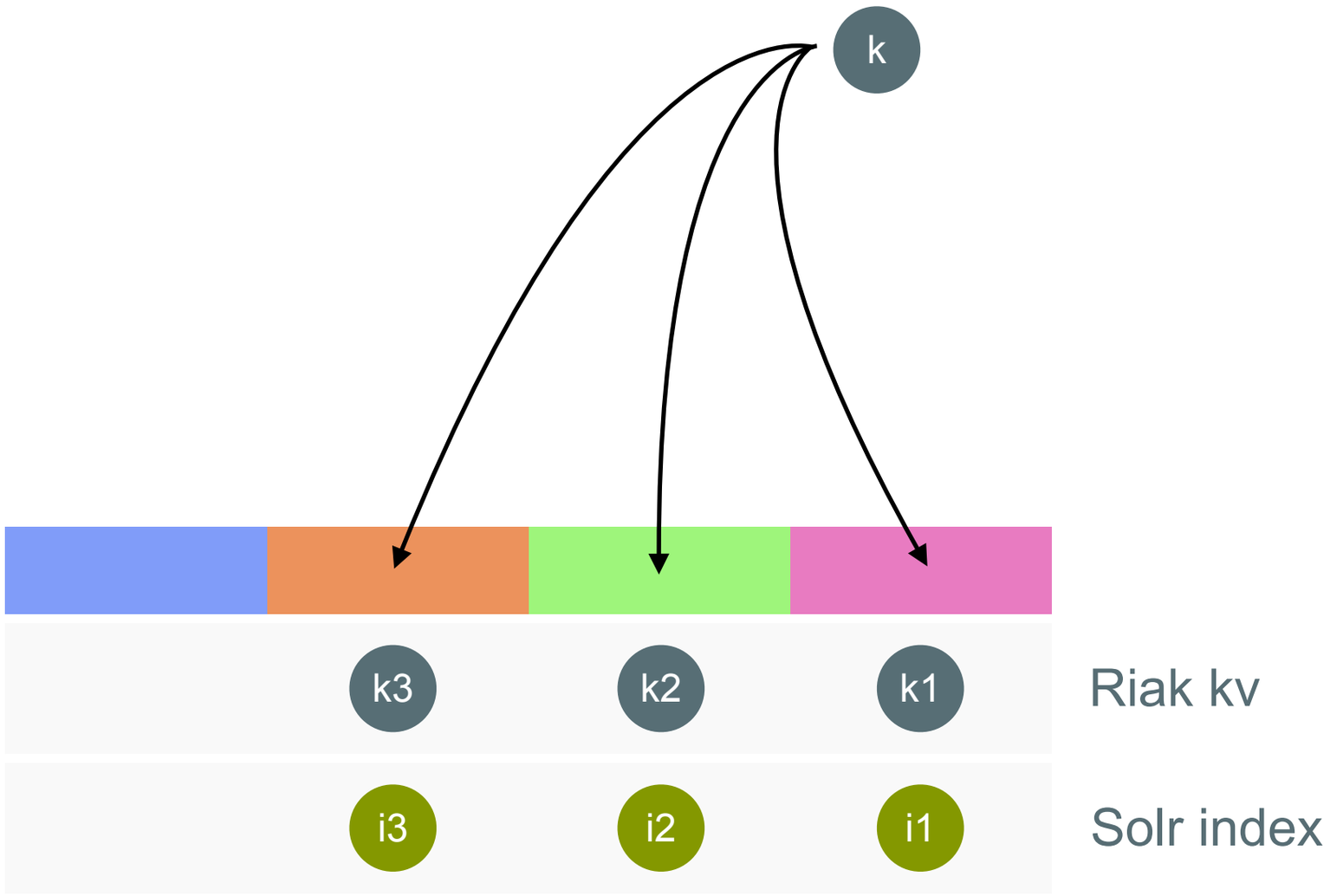
Associate Bucket

```
curl -XPUT -i -H 'content-type: application/json'
'http://localhost:10018/buckets/my_bucket/props' -d
'{"props":{"yz_index":"my_index"}}'
```

Replication



Three Replicas



Features

Solr has it?

Yokozuna has it!*

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

Powerful Analysis

- Full-text to tokens
- Lowercasing
- Stemming
- Synonyms
- Stop-word removal
- Language support

Querying

?q=<field>:<term>

- Single Term
  `?q=commit_repo:riak_kv`
- Boolean (OR, default)
  `?q=commit_repo:riak_kv%20commit_repo:riak_core`
- Boolean (AND)
  `?q=commit_repo:riak_kv%20AND%20commit_author:”Ryan%20Zezeski"`
- Boolean (NOT)
  `?q=commit_repo:riak_kv%20NOT%20commit_author:”Ryan%20Zezeski"`

?q=<field>:<term>

- Range (good for dates; Solr has “date math”)
  `?q=commit_dt:[NOW-1YEAR TO NOW]`
- Wildcard everything (good catch all)
  `?q=*:*`
- Wildcard terms
  `?q=commit_repo:riak_*`
- Wildcard Regex
  `?q=NoExample`

?q=<field>:<term>

- Term (Full Text)
  `?q=commit_subject:vnode%AND%commit_body:vnode`

- Phrase/Proximity (exact match)
  `?q=commit_body:”hinted handoff”`

- Phrase/Proximity (“slop”/“edit distance” of 4)
  `?q=commit_body:”parition vnode”~4`

- Fuzzy (slop at word level for misspellings)
  `?q=commit_body:behaviour~1`

Sort & Rank

- Sorting (good for dates with ranges)
  `?q=commit_dt:[NOW-1YEAR TO NOW]&sort=commit_dt%20asc`

- Ranking
  `?q=commit_body:”hinted handoff”&fl=commit_*,score`

Tagging

- Adds 2i like functionality
- Indexes via object metadata
- Index tags that do not affect the object
- Useful for binary objects

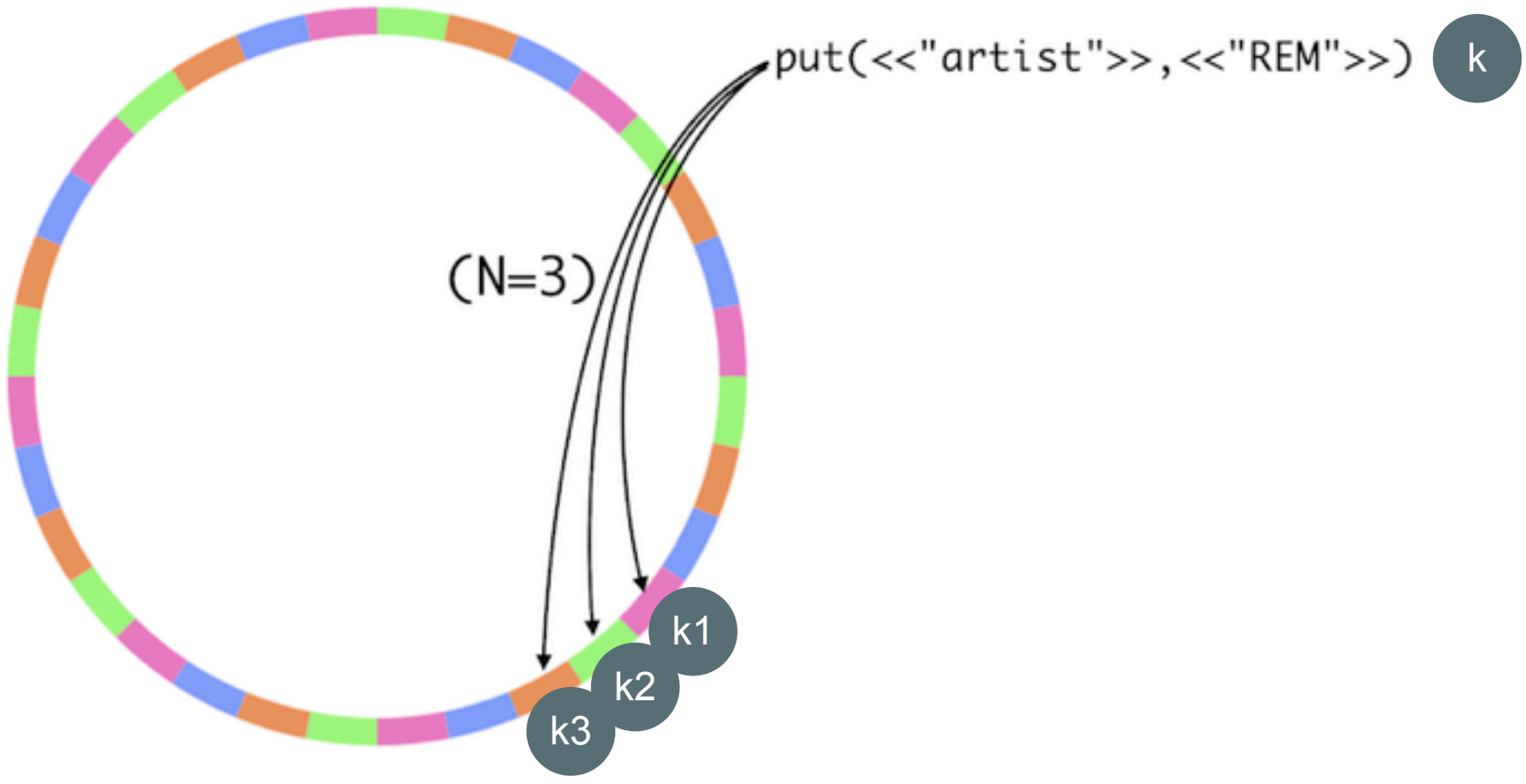
Facets

Highlighting

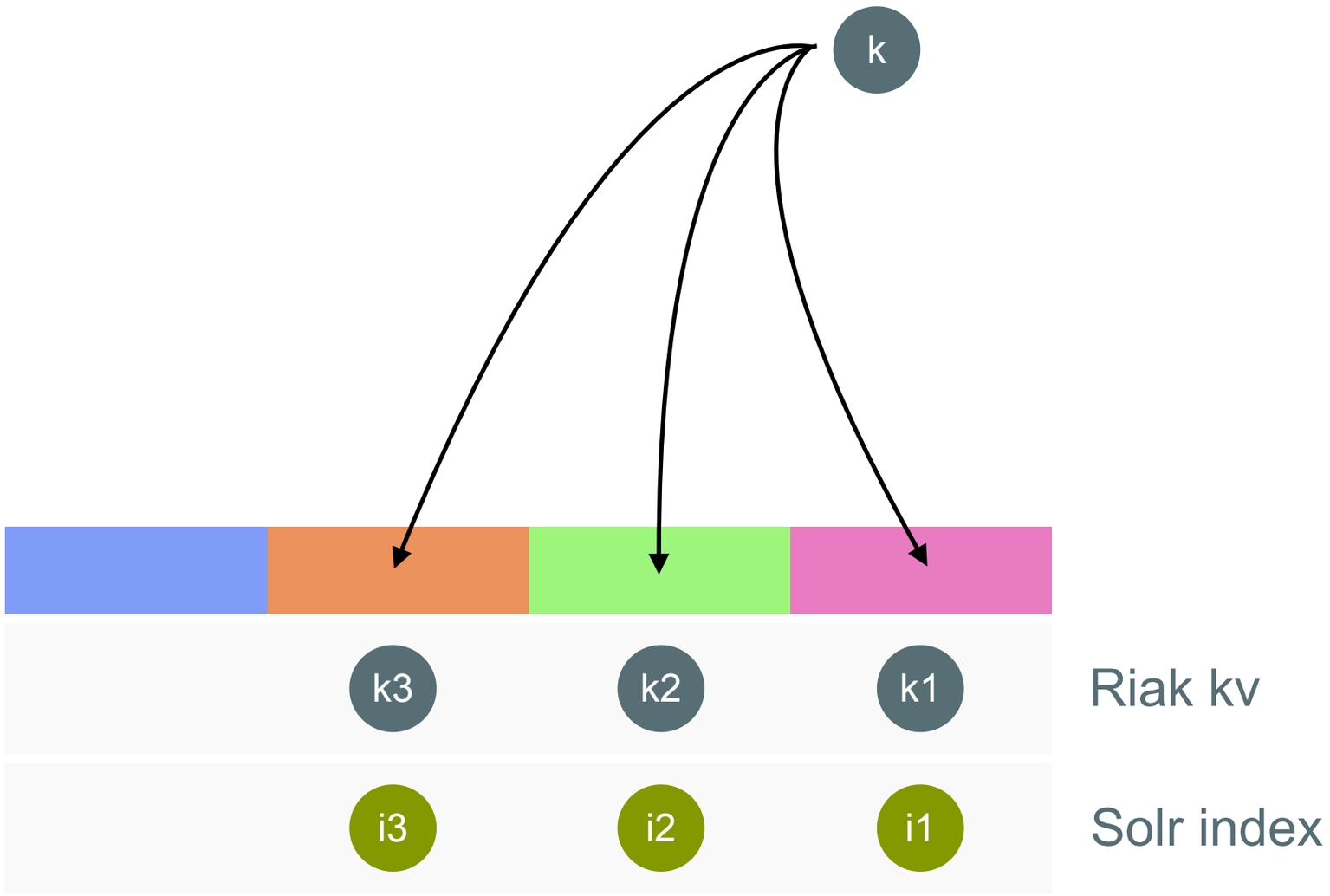
Self Healing

Hinted Handoff

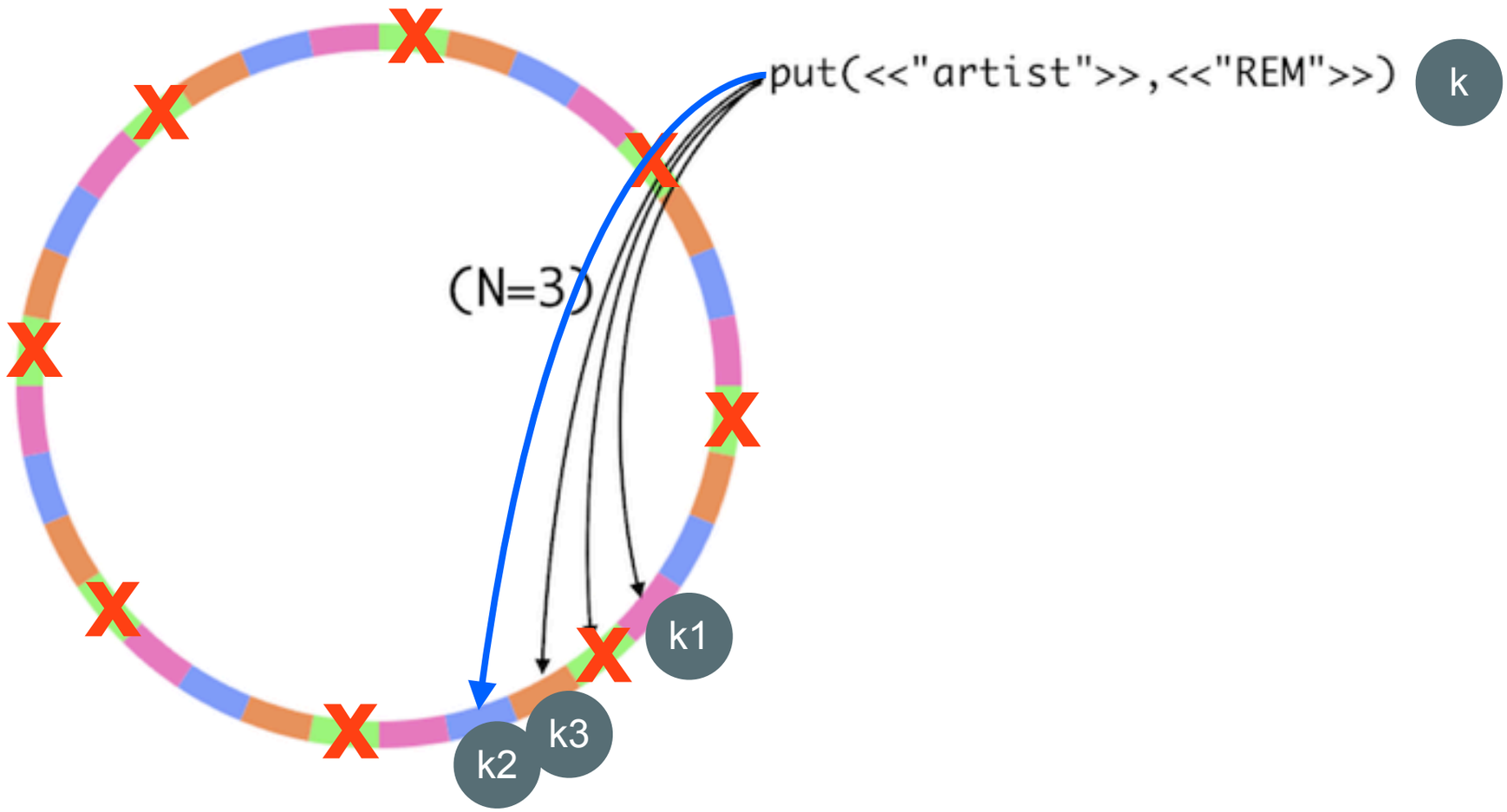
Replication



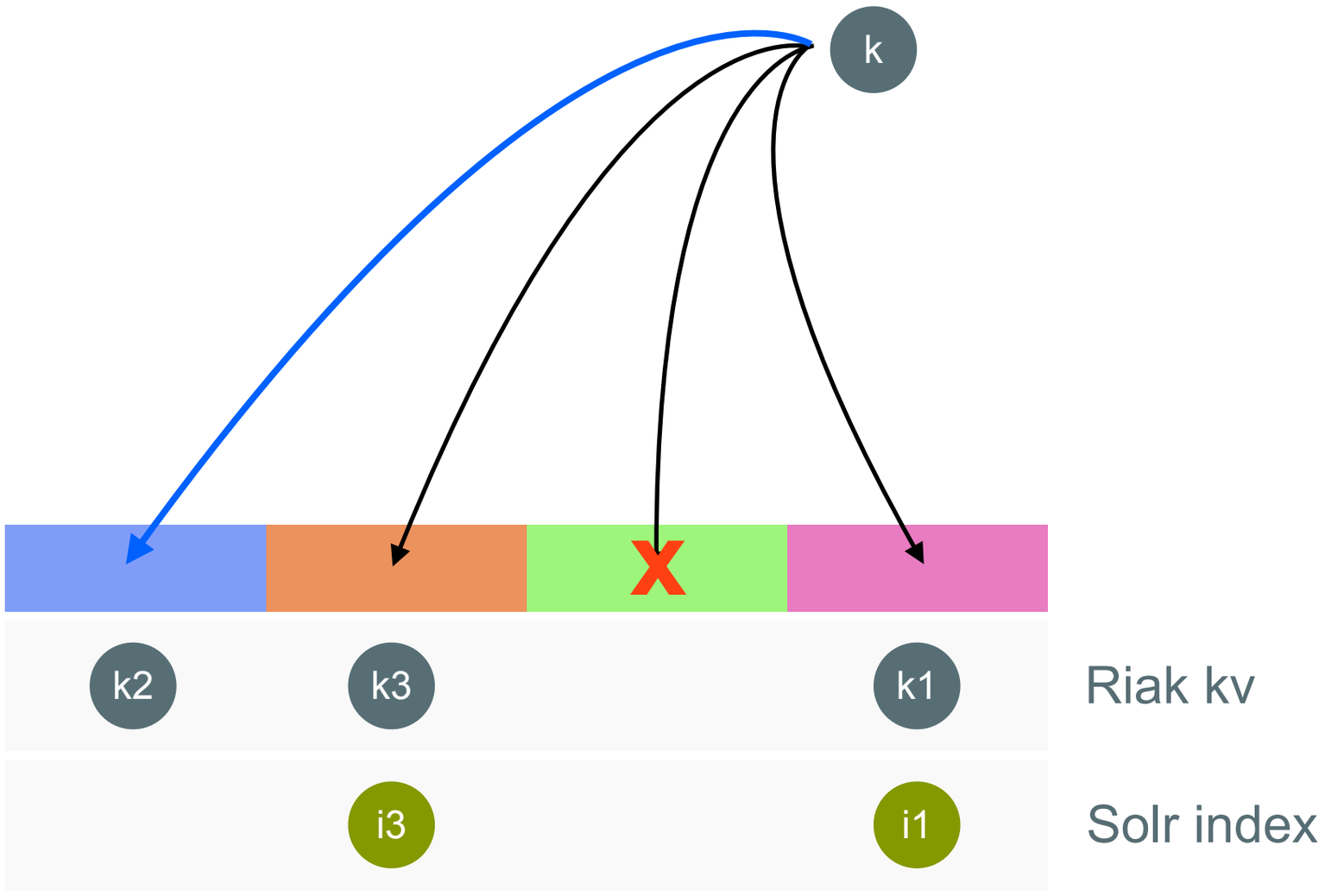
Three Replicas



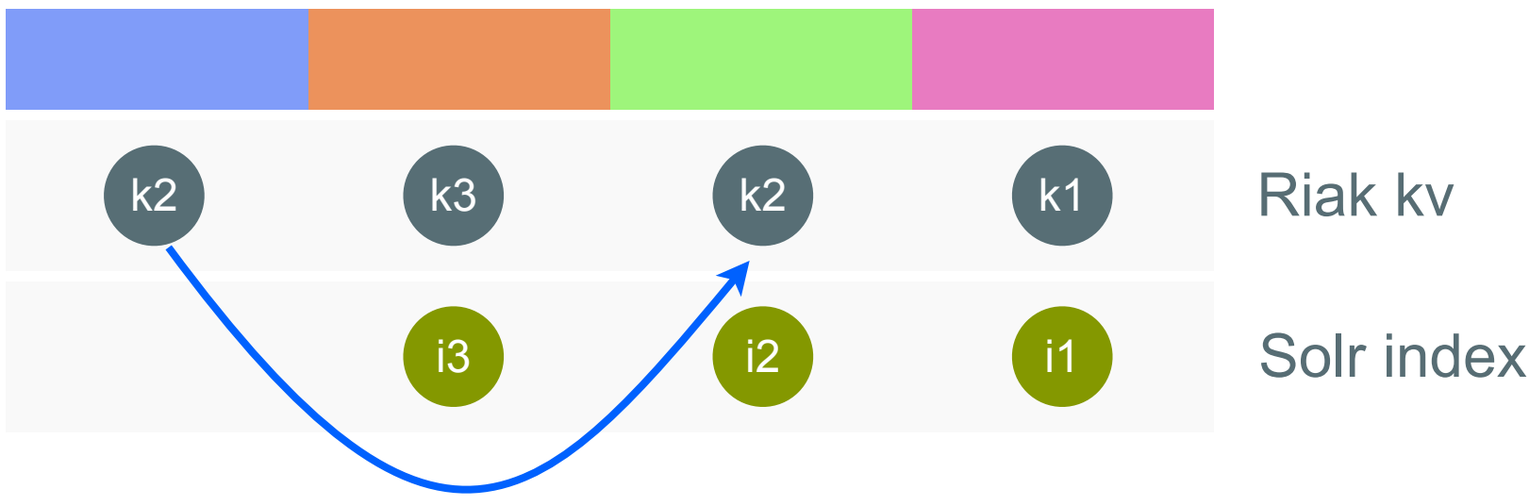
Node Failure



Fallback Replica



Hinted Handoff



Hinted Handoff

- When a node in Riak fails, fallbacks are used
- When the node returns, data is handed back
- As data is “handed-off” from fallback to primary, it is indexed on the primary

Active Anti-Entropy

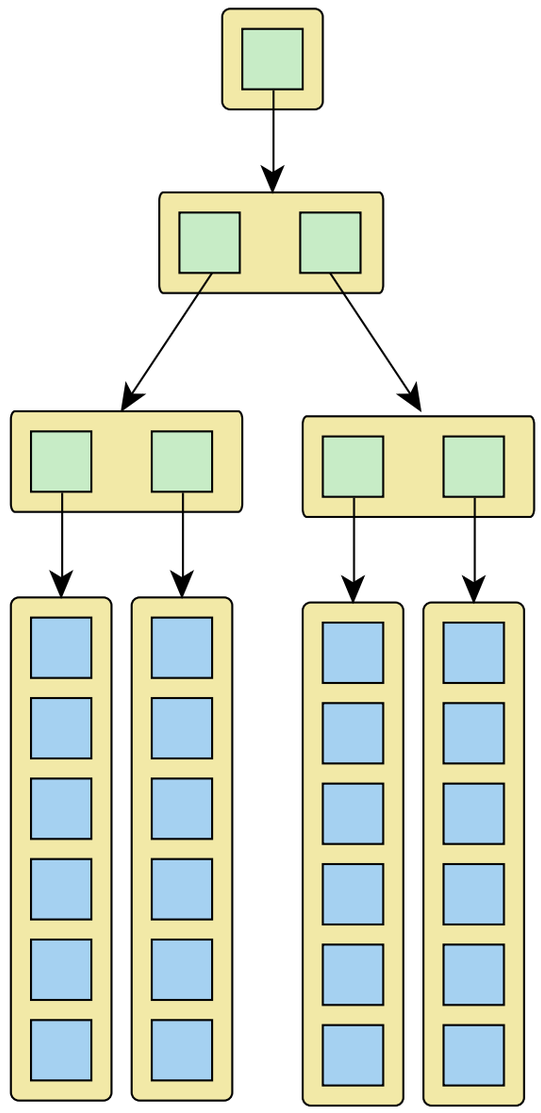
AAE

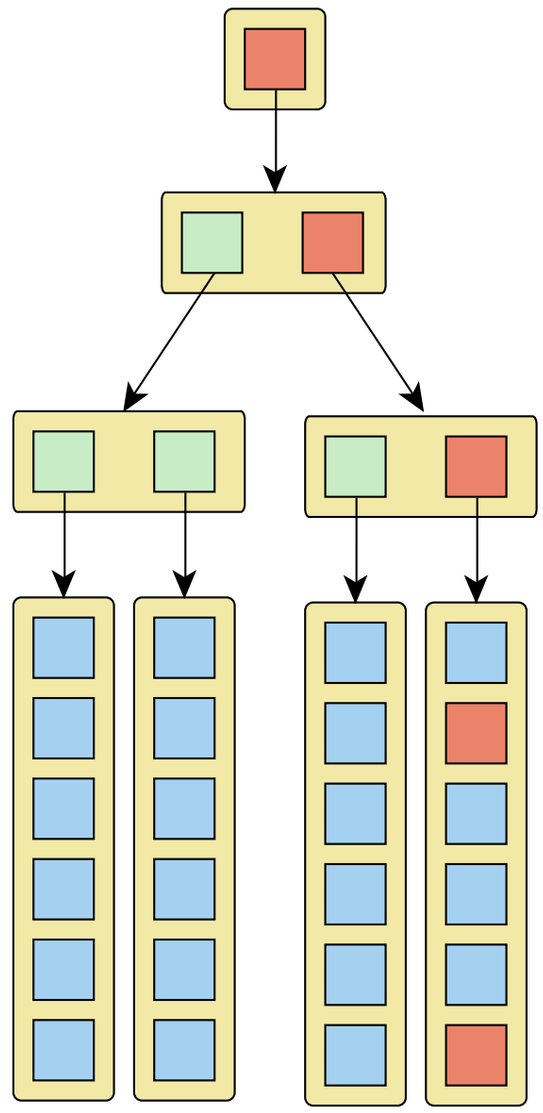
- Two systems (Riak & Solr) increase chances of inconsistency
- Files can become corrupted/truncated
- Solr indexes could be accidentally removed
- Handles malformed KV data

AAE

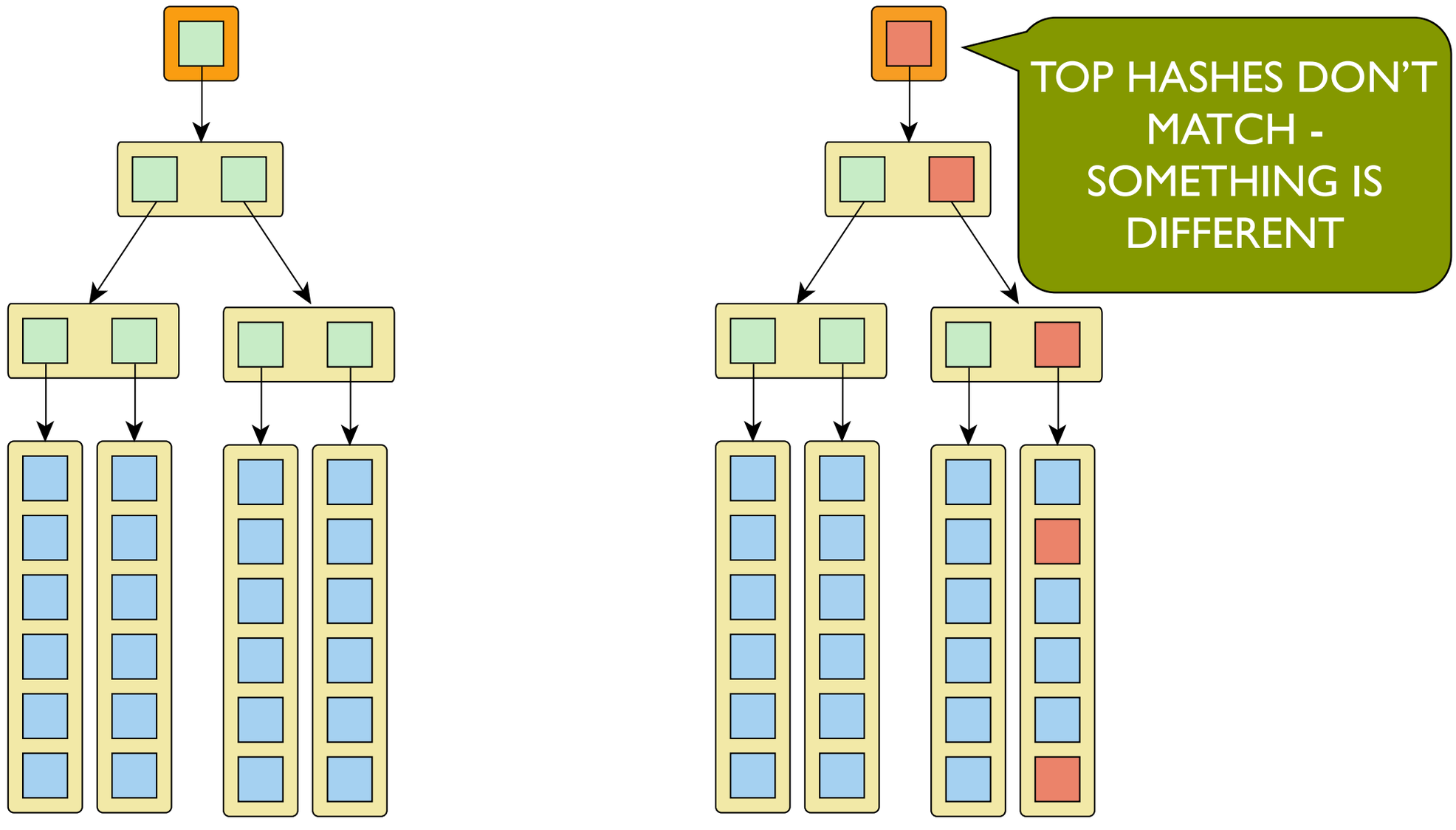
- It uses hash trees
- Updates in real time
- It’s non-blocking
- Periodically exchanged
- Periodically expired and rebuilt
- It invokes read-repair and re-index on divergence

AAE - Exchange

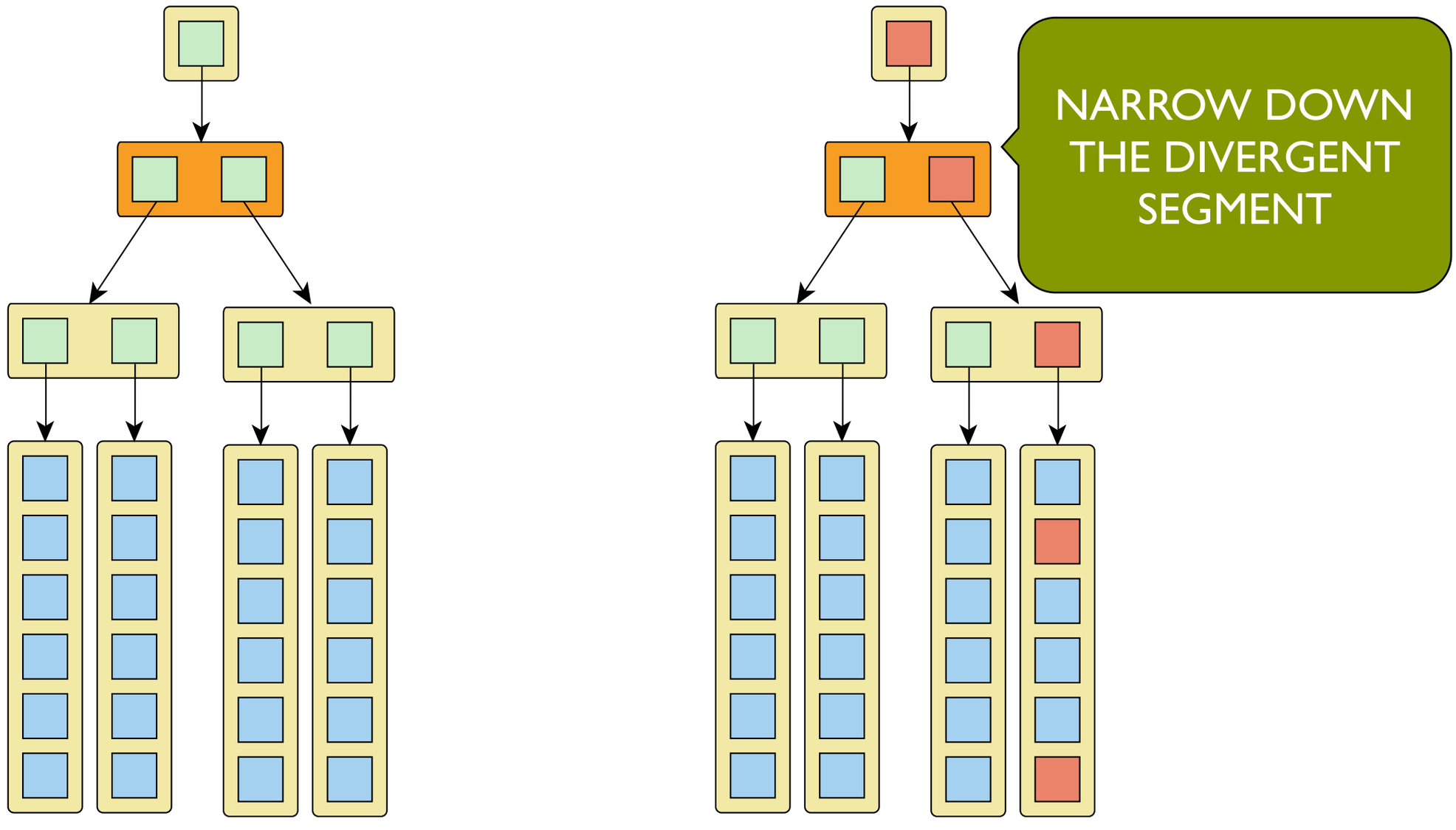




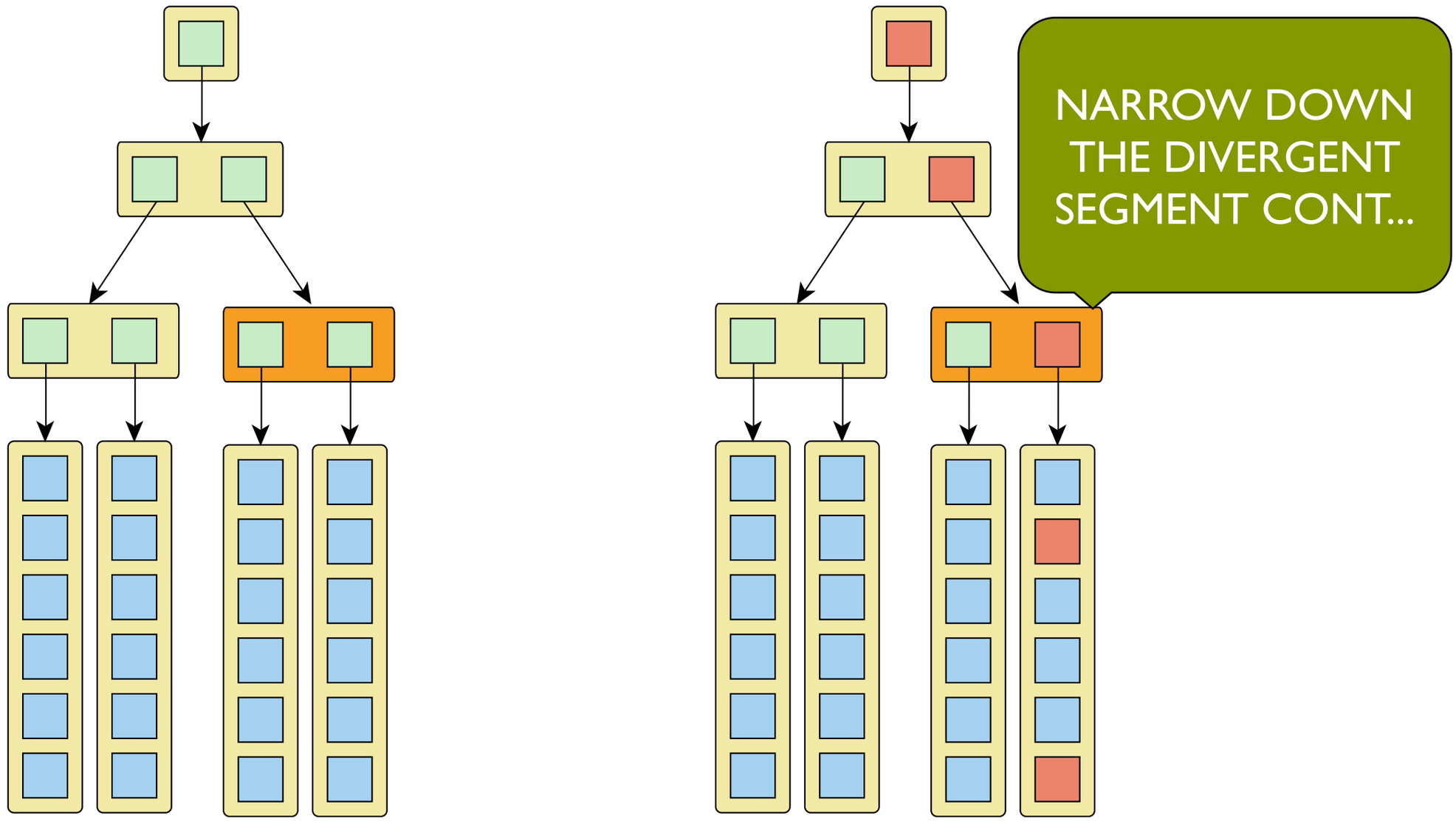
AAE - Exchange



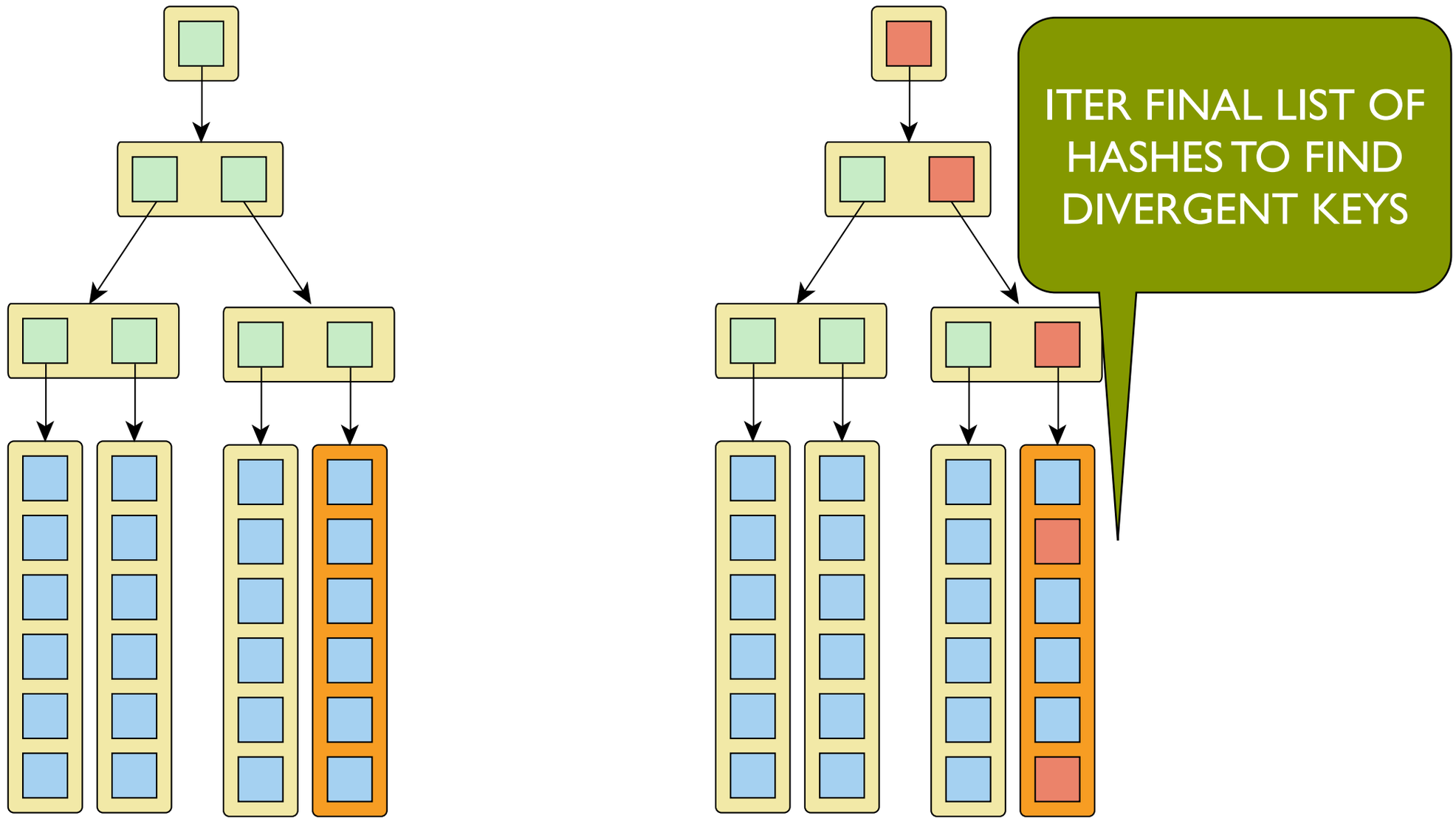
AAE - Exchange



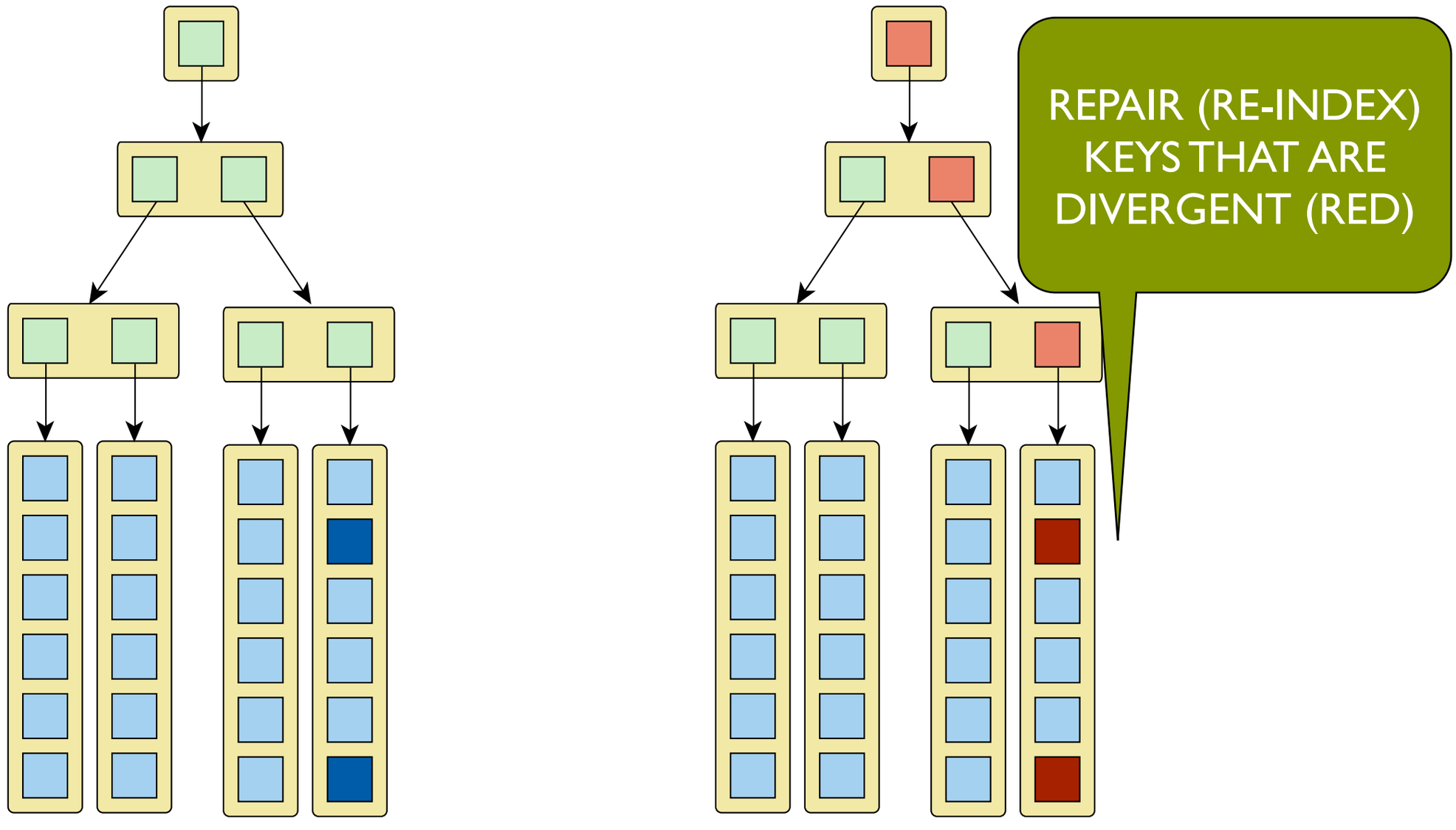
AAE - Exchange



AAE - Exchange



AAE - Exchange



Learn More

- Mailing list at docs.basho.com
- #riak IRC room on irc.freenode.net
- http://bit.ly/riak-2-0

Questions?

Thanks very much
dbrown@basho.com