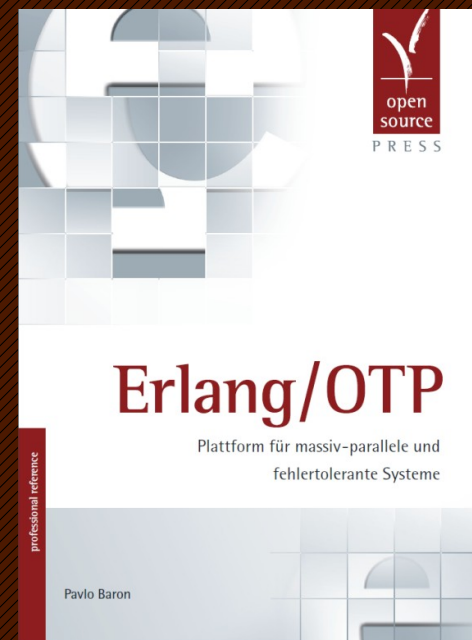


**from
hand
to
mouth**





Geek's
Guide
To The Working Life

Pavlo Baron

pavlo.baron@codecentric.de
@pavlobaron

Forget backend

**Or at least what
you call a backend**

**Your application server
will just slow down
your machines to
keep step with them**

**Or you will slow down
your incoming traffic
to give your application
server a chance to
breathe**

**It's like: order food
at McDonalds, get
queued and receive
the ordered food
through mail 3
days later**

Trying to speed it up
is like: order food at
McDonalds, get queued,
leave the restaurant
with color pictures of
your food and wait 3
days for its delivery

**Every single data
abstraction layer only
helps ruin the
atmosphere through
heating**

Hey, man,
you carry around a damn
USS Enterprise in your
pocket

And you can run a
damn Babylon 5 in
a cluster

And in the end, it's
always store/update/
delete/read/search/
process.

Isn't it?

So why drive a clown car
when you can have
a Ferrari
full of these?



Why not just live
from hand to mouth?



You haz this?

**Mobile clients write from
everywhere, buffer,
read occasionally,
post-processing completely
behind the scenes, like
statistics etc.**

Zoom in!

Immediate, reliable, massive writes. Analytics and processing in a batch afterwards. No need to be exact to the second and 100% data complete

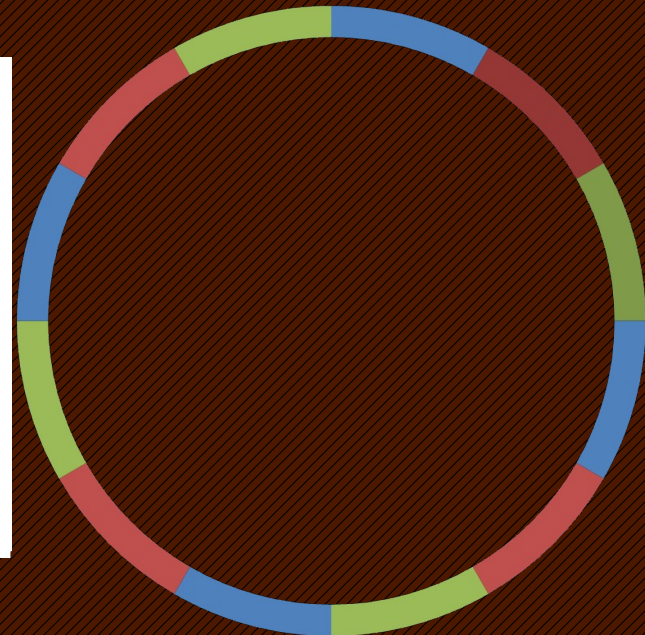
Zoom in!



Zoom in!

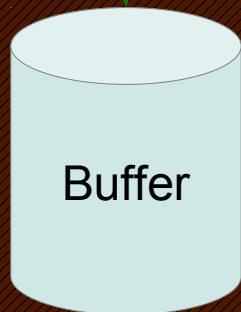


ProtoBufs, REST
Round Robin



Node 1
Node 2
Node 3

local



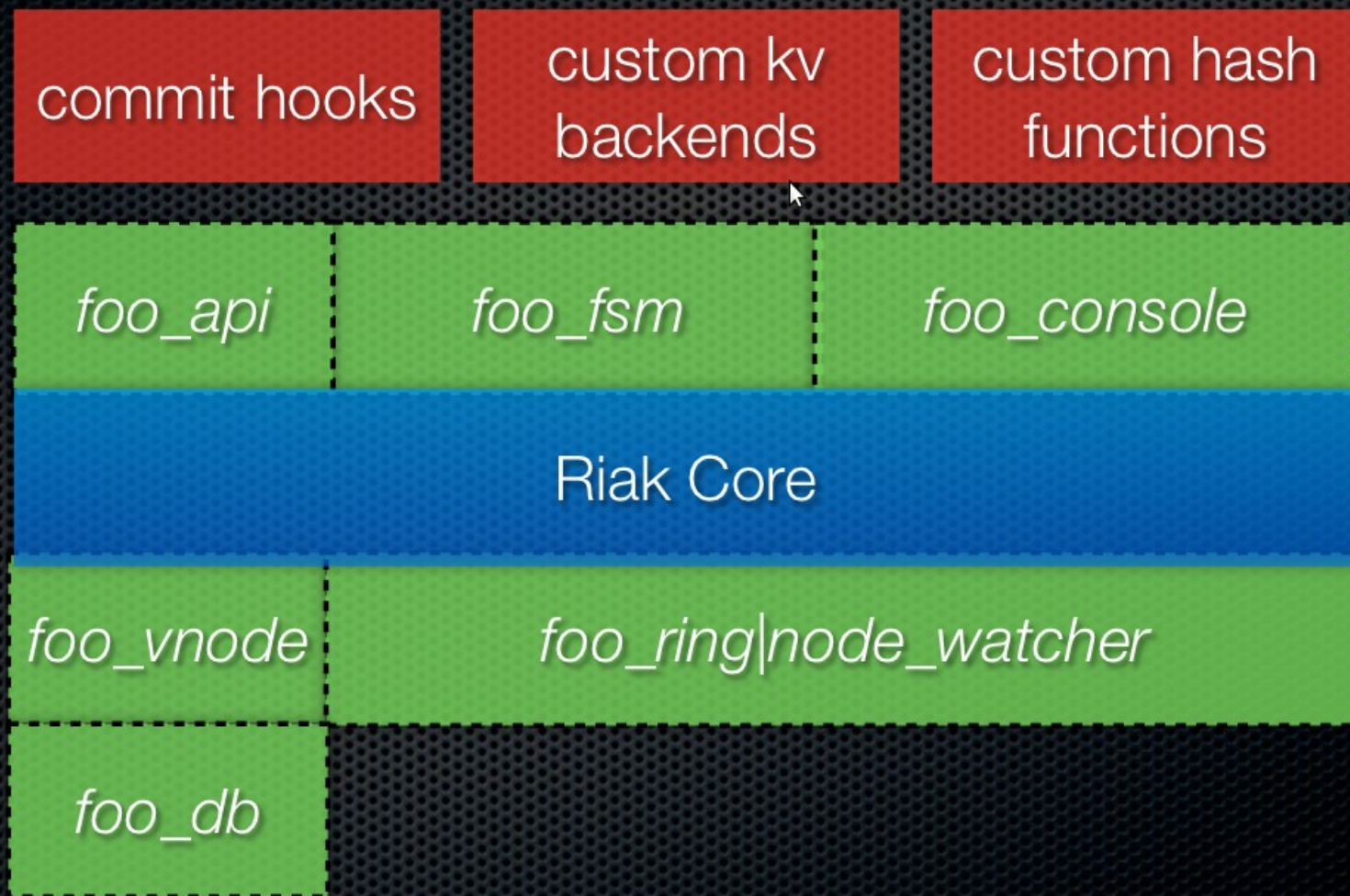
REST, local Erlang

Web Apps,
R,
tools etc.



HTF does it work?

Riak Extension Points



X bit integer space

$$0 \leq N \leq 2^X$$

or: $2 \times \text{Pi}$

$$0 \leq A \leq 2 \times \text{Pi}$$

$$x(N) = \cos(A)$$

$$y(N) = \sin(A)$$

12 partitions (constant)

3 nodes, 4 vnodes each

add node

4 nodes, 3 vnodes each

Alternatives:

3 nodes, $2 \times 5 + 1 \times 2$ vnodes

container based

Da quorum

V: vnodes holding a key

W: write quorum

R: read quorum

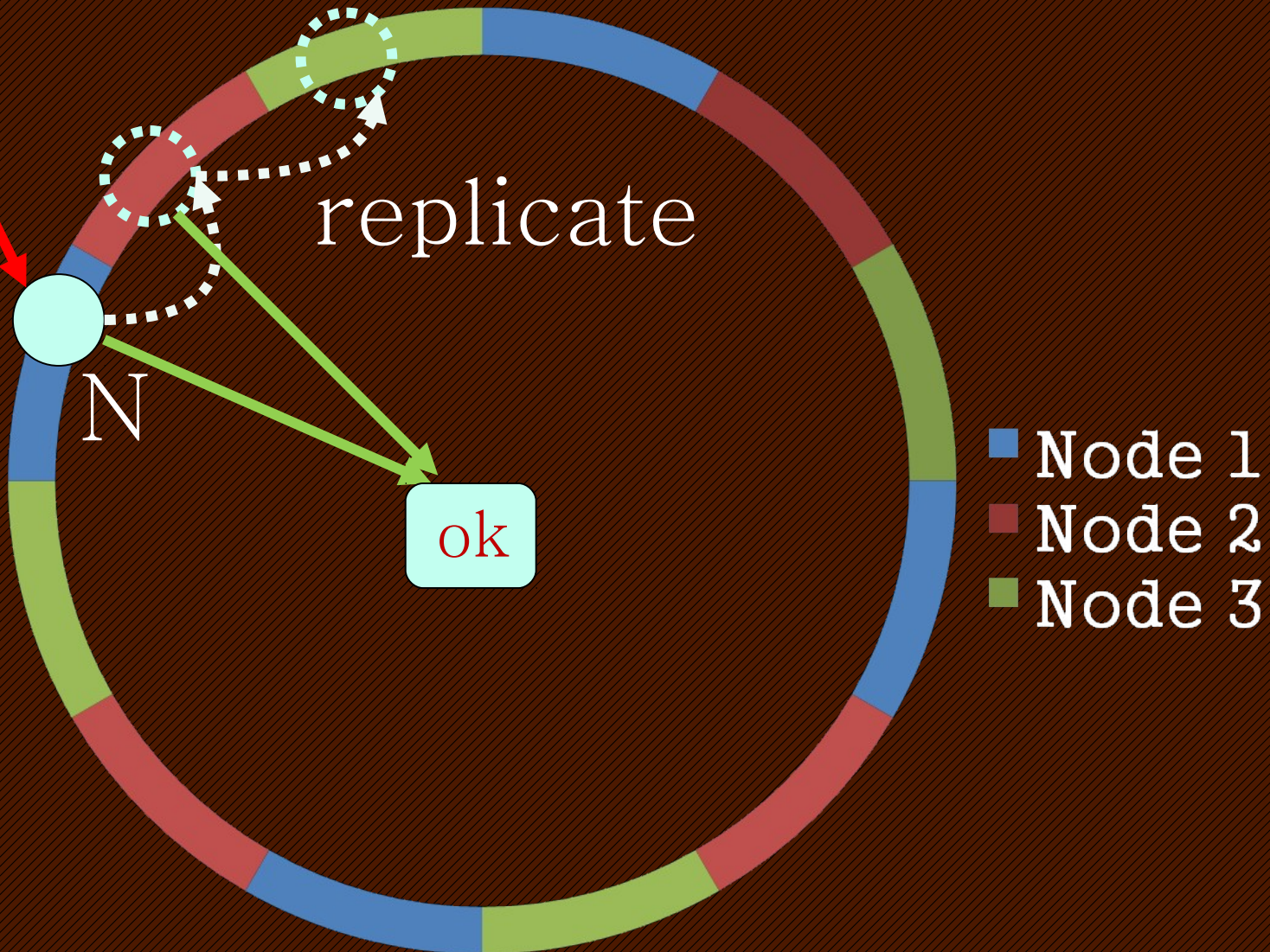
DW, PW, PR

$$W > 0.5 * V$$

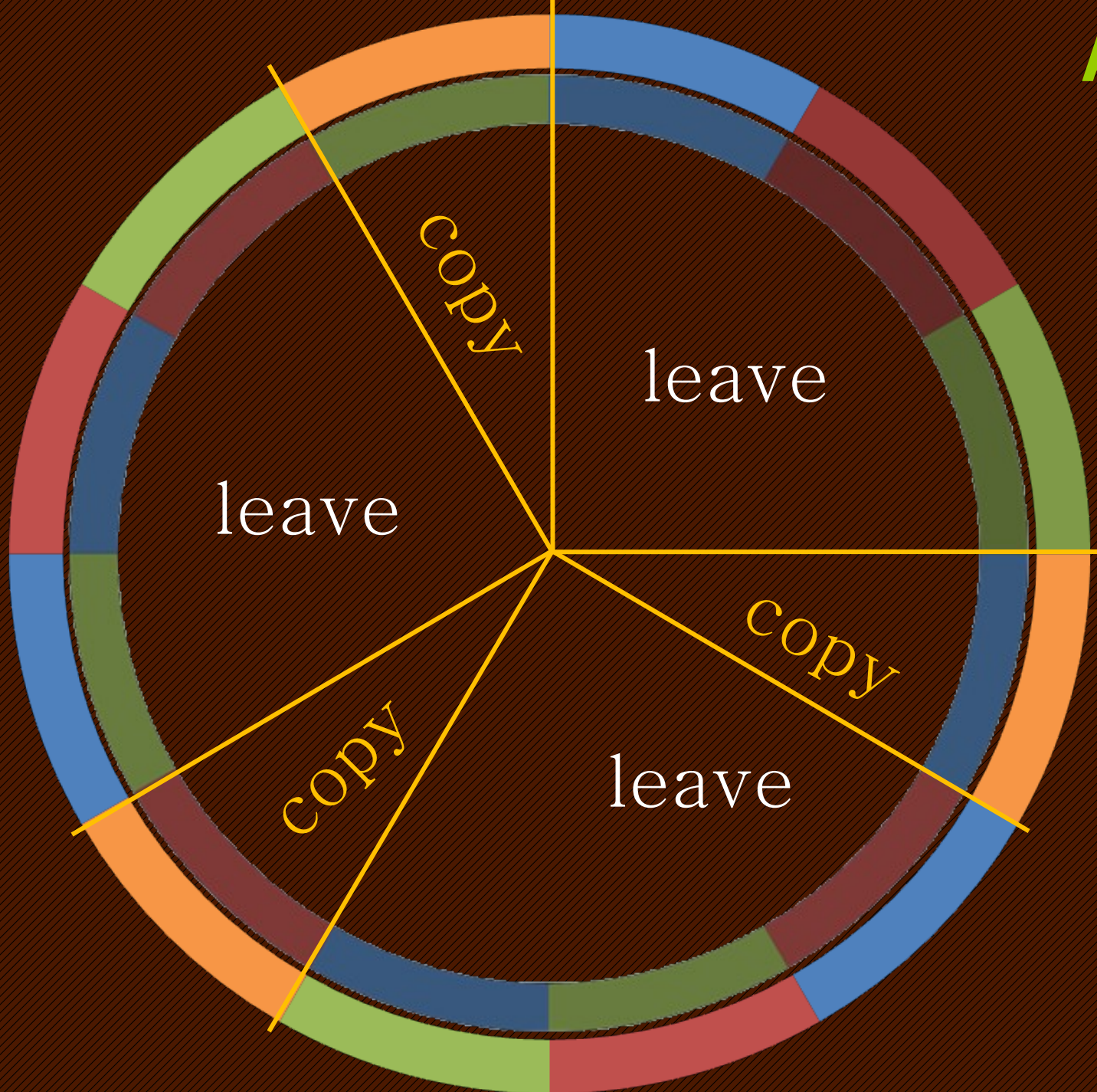
$$R + W > V$$

Key = "foo"
= N, W = 2

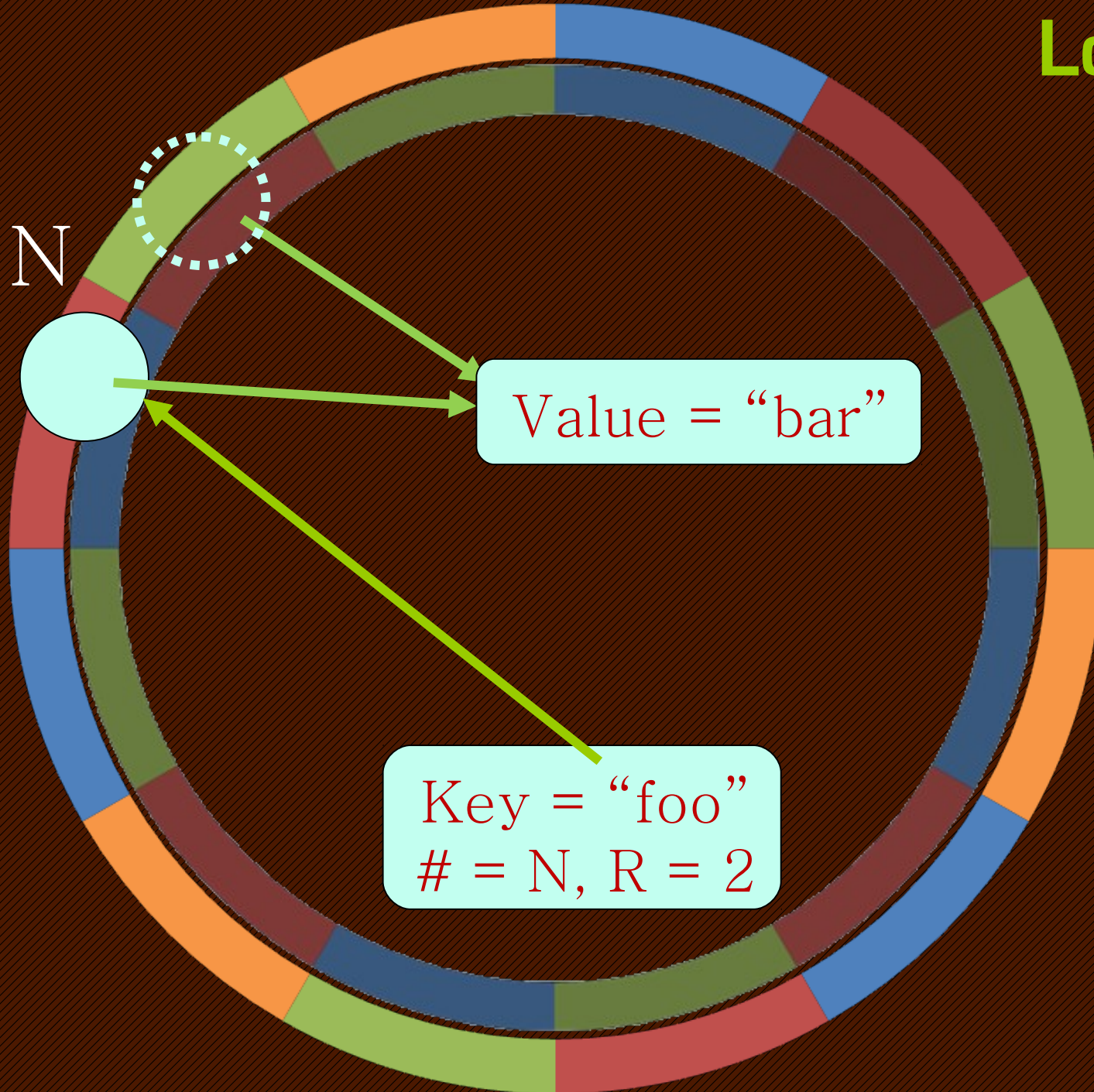
Insert key
(sloppy quorum)



Add node



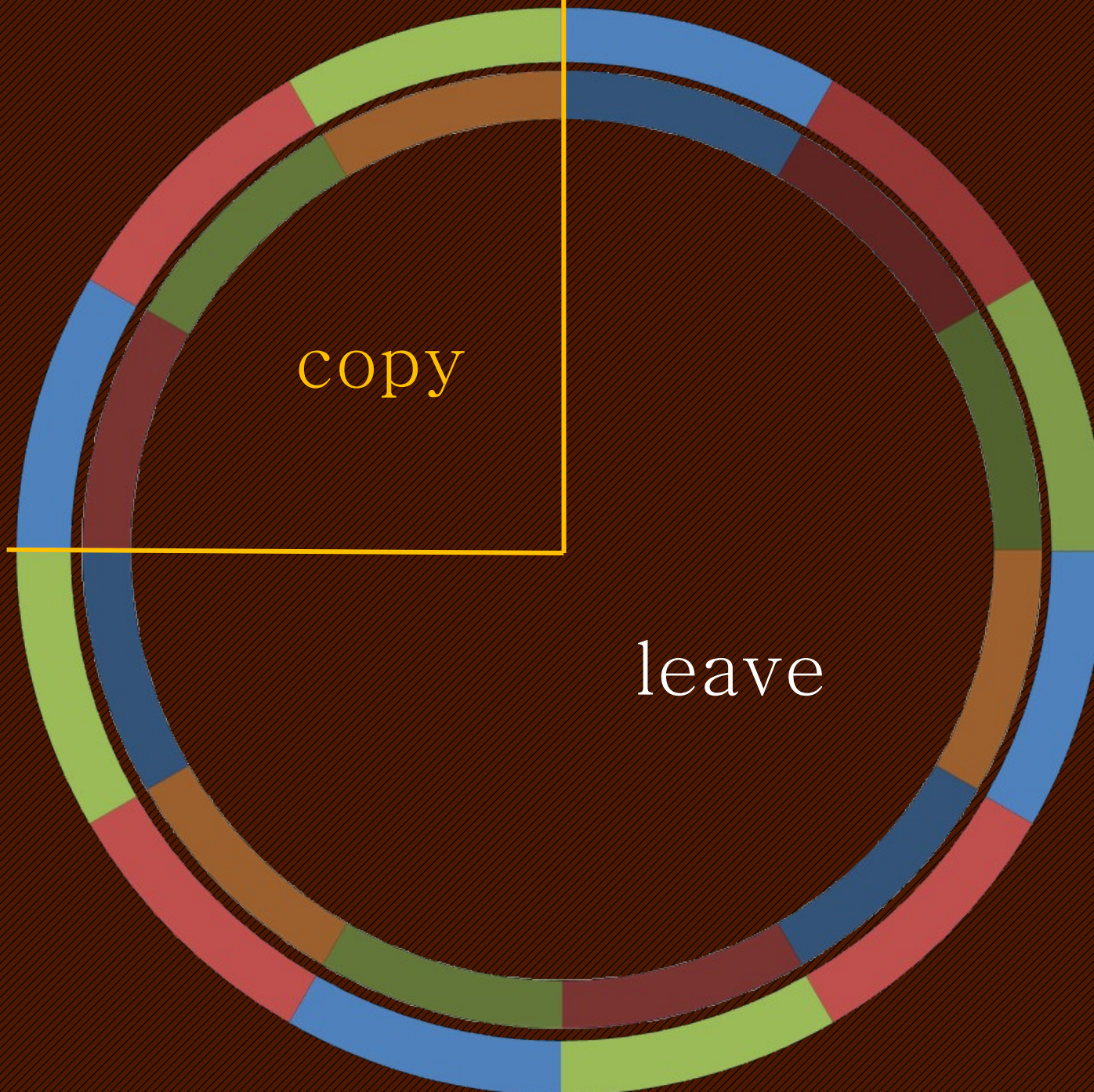
- Node 1
- Node 2
- Node 3
- Node 4



Lookup key
(sloppy
quorum)

- Node 1
- Node 2
- Node 3
- Node 4

**Remove
node**



- Node 1
- Node 2
- Node 3

write

RM2

Da Gossip

Clock table

Update log

Replica clock

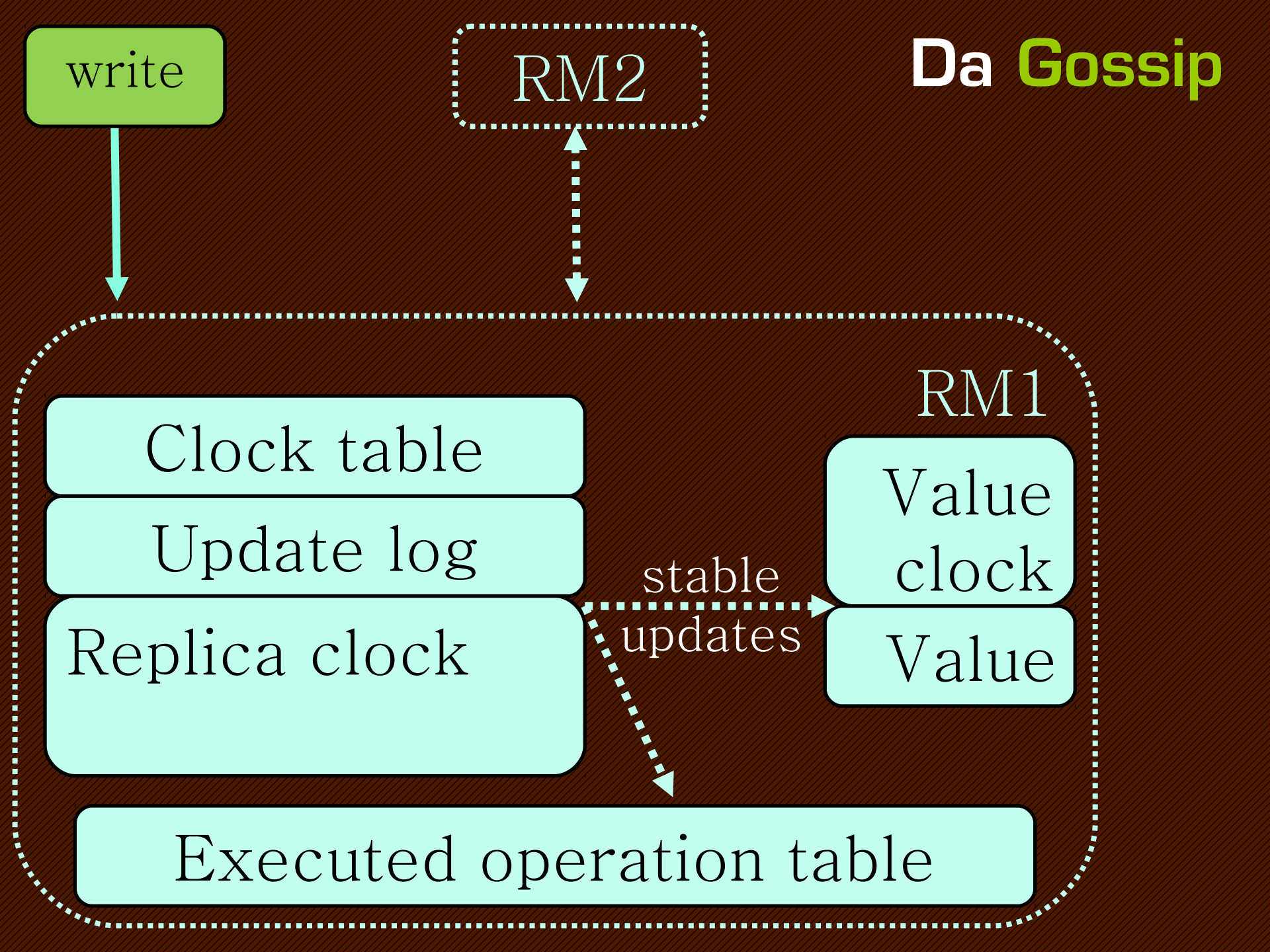
RM1

Value
clock

Value

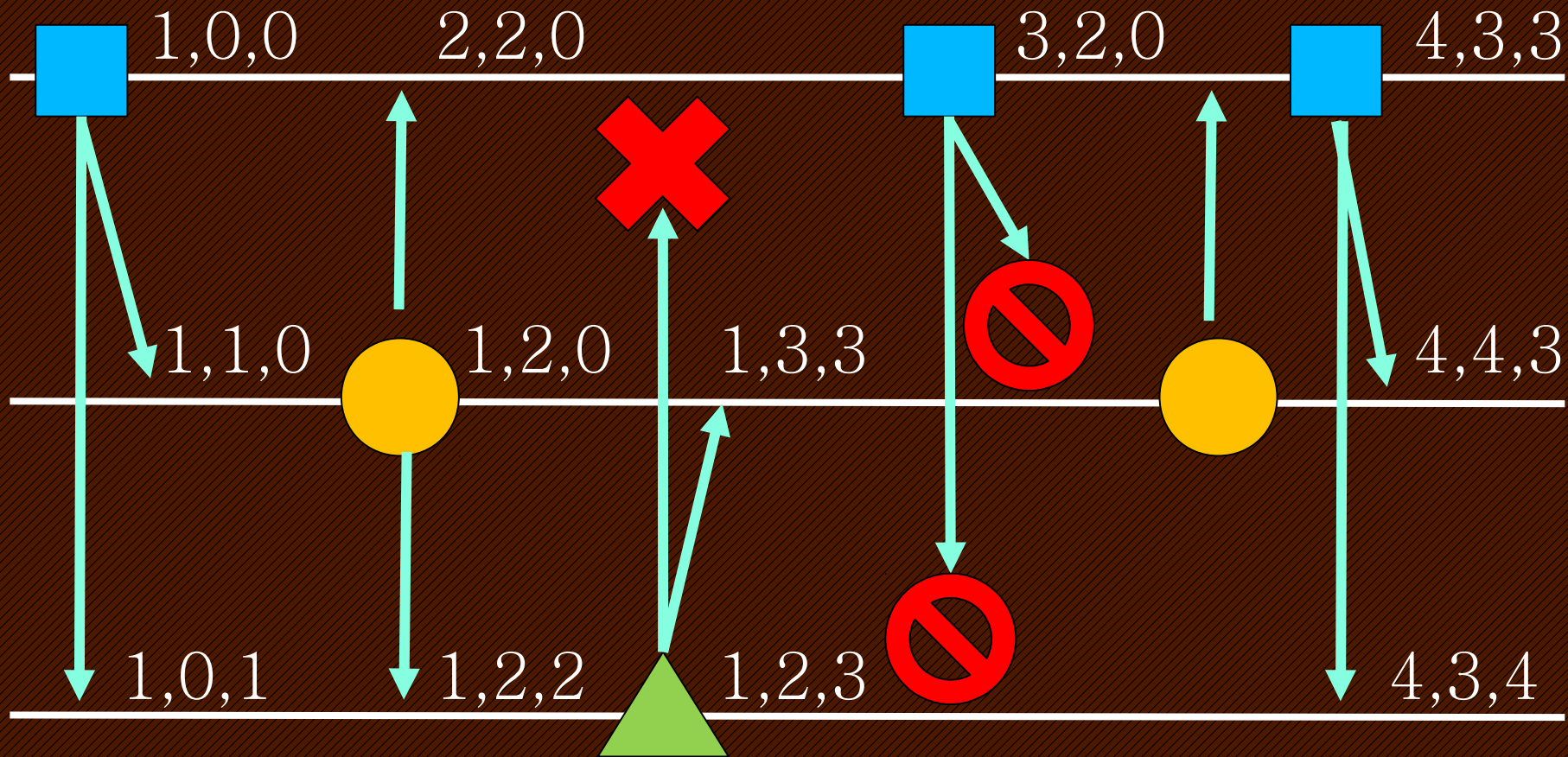
stable
updates

Executed operation table

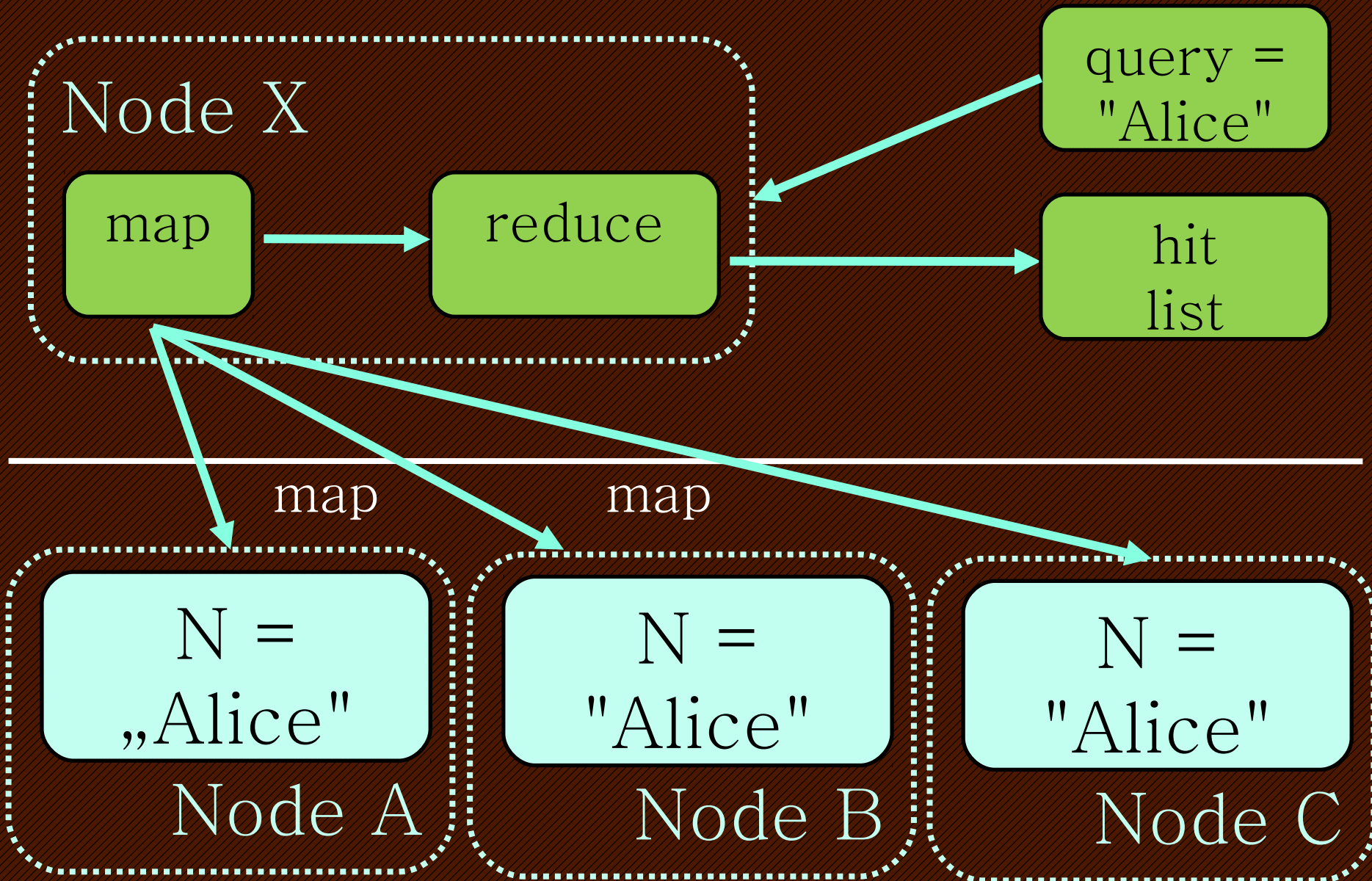


Da **vector** clocks

Node 1
Node 2
Node 3



Da MapReduce



Da hinted handoff

N: node, G: group including N

node(N) is unavailable

- replicate to G or

- store data(N) locally

- hint handoff for later

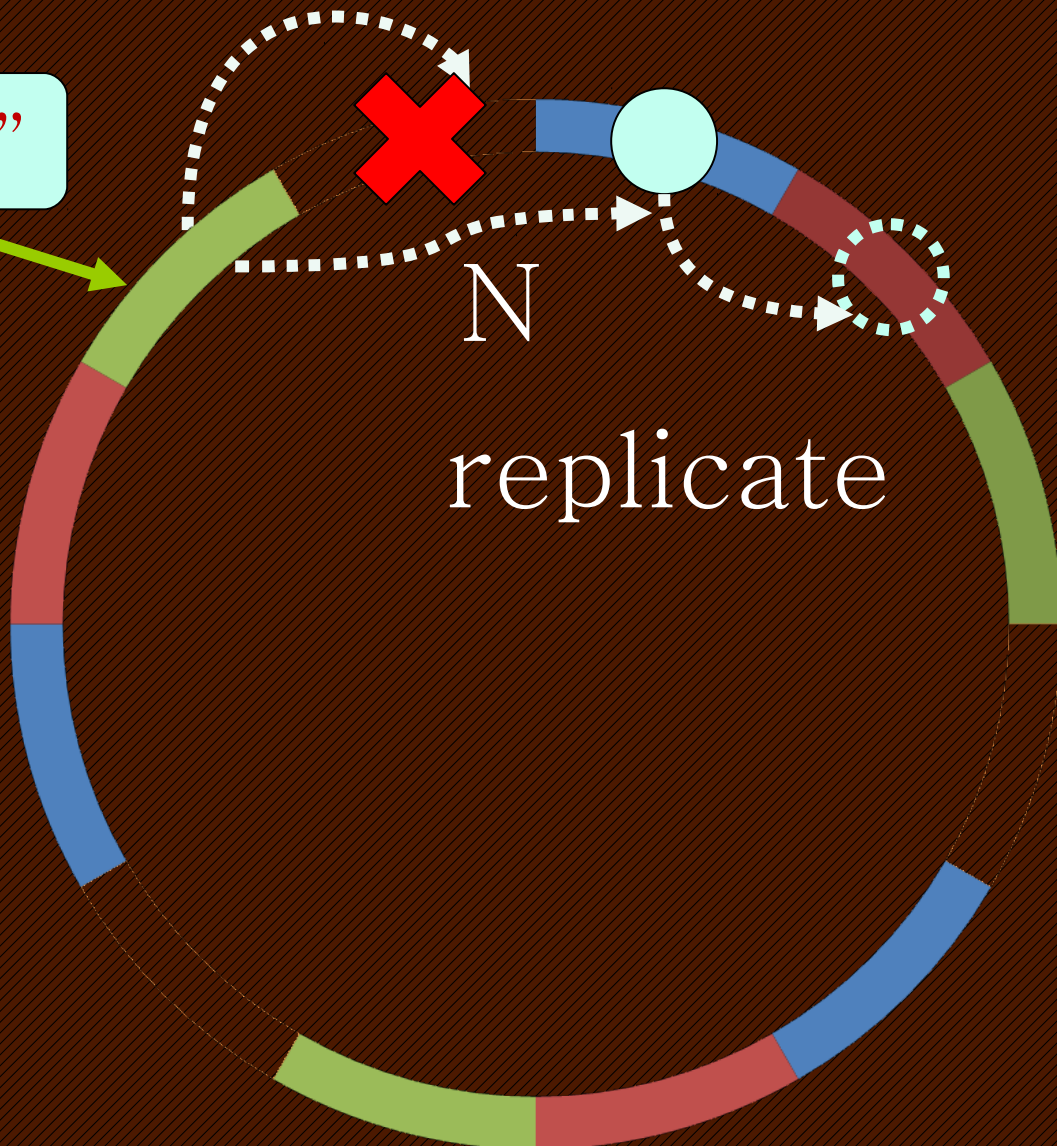
node(N) is alive

- handoff data to node(N)

Key = "foo", # = N ->
handoff hint = true

Replica
fails

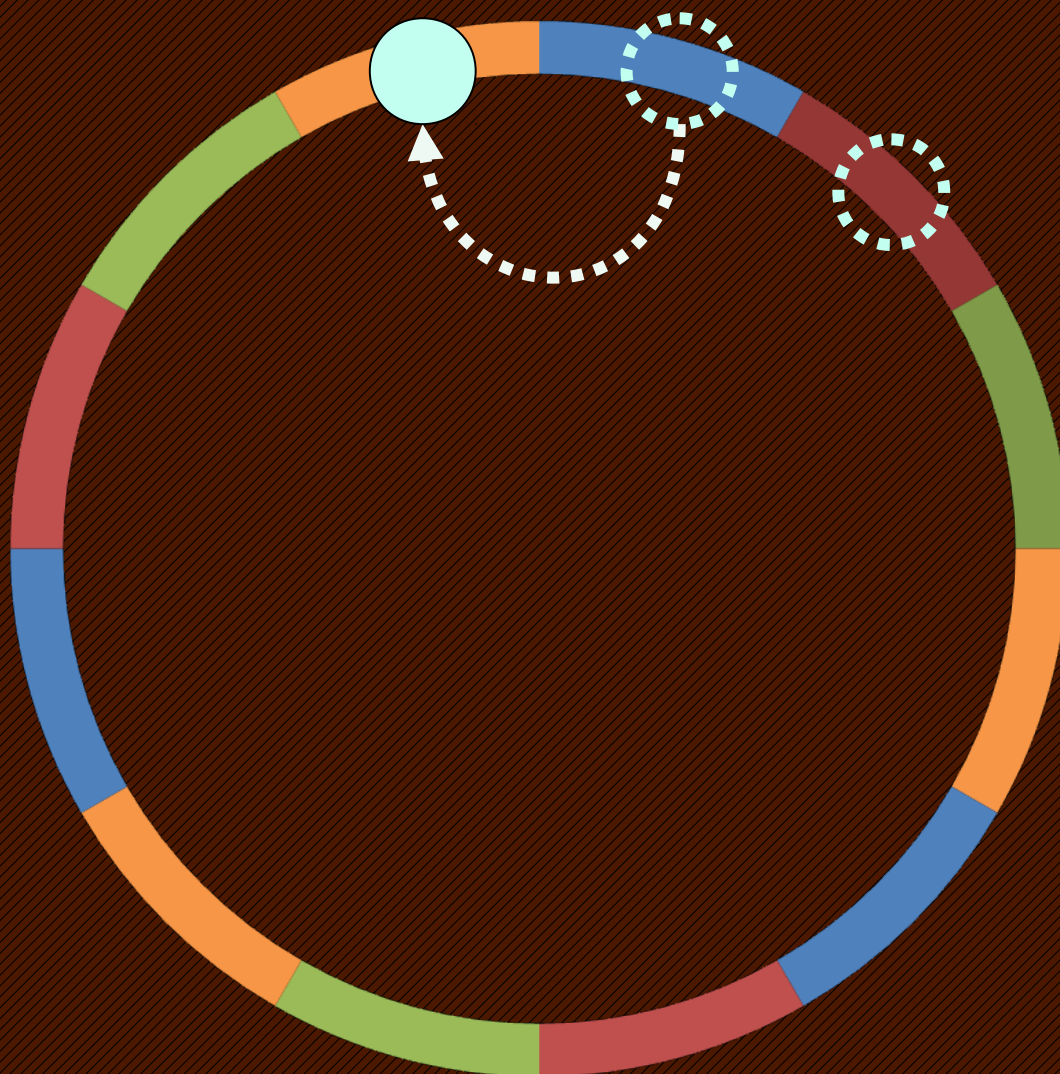
Key = "foo"



- Node 1
- Node 2
- Node 3
- Node 4

handoff

Replica
recovers



- Node 1
- Node 2
- Node 3
- Node 4

And that's by far not all:

You can search through
index with data locality.

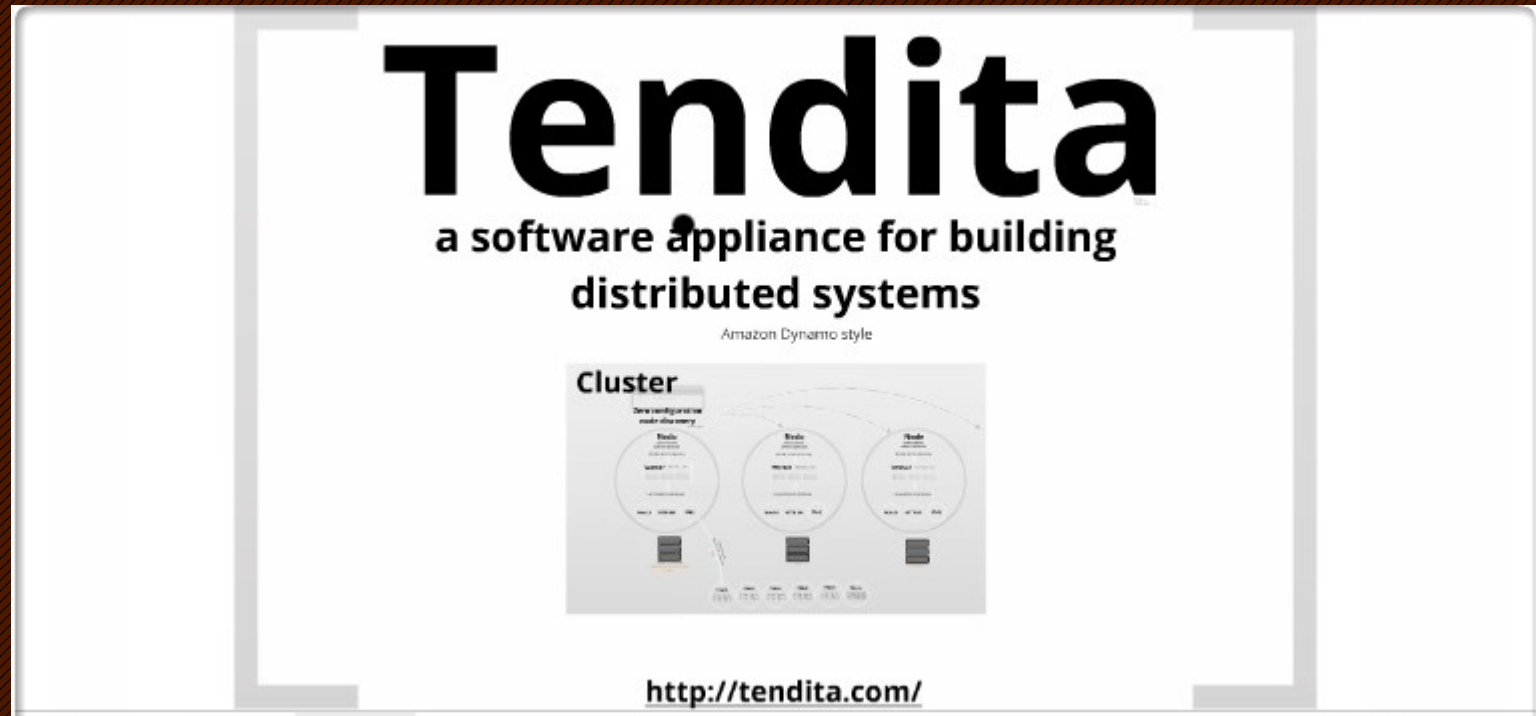
You can tag objects and
query through 2i.

You can add hooks

You don't even need to store anything.

**Just distribute your
calculation, search, batch**

Have a look at tendita.com



Tendita
a software appliance for building
distributed systems
Amazon Dynamo style

Cluster

Three nodes are shown, each with a 'Node' label and a 'Cluster' label. The nodes are connected by dashed lines, indicating a distributed system architecture. Below the nodes, there is a table with columns for 'Node', 'IP', 'CPU', 'MEM', and 'DISK'.

Node	IP	CPU	MEM	DISK
Node 1	10.0.0.1	100%	100%	100%
Node 2	10.0.0.2	100%	100%	100%
Node 3	10.0.0.3	100%	100%	100%

<http://tendita.com/>

So it's even comfy to load
your clown car
with these



Thank you



Most images originate from
istockphoto.com

except few ones taken
from Wikipedia or Flickr (CC)
and product pages/
publicly available presentations
or generated through public
online generators