PROJECTFIFO

Diabolic Database Design

don’t try this at home!

Who I am?

- Good question, someone tell me if they find out!
- Building cloud orchestration software for SmartOS (Project-FiFo).
- Love solid technology, Illumos & Erlang.
- Please don't take everything I say serious, on occasions I deploy **humor**.

PROJECTFIFO

Why?

- Project-FiFo is cloud orchestration
- that means LOTS of servers and VMs
- it is really helpful to get some metrics on how they work
- existing systems don't really cut it

PROJECTFIFO

What do we build?

- Riak like operations & scale
- Pick the good ideas from Graphite
- Keep it as simple as possible
- ensure lively data
- be open about data loss

PROJECTFIFO

Defining 'a metric'

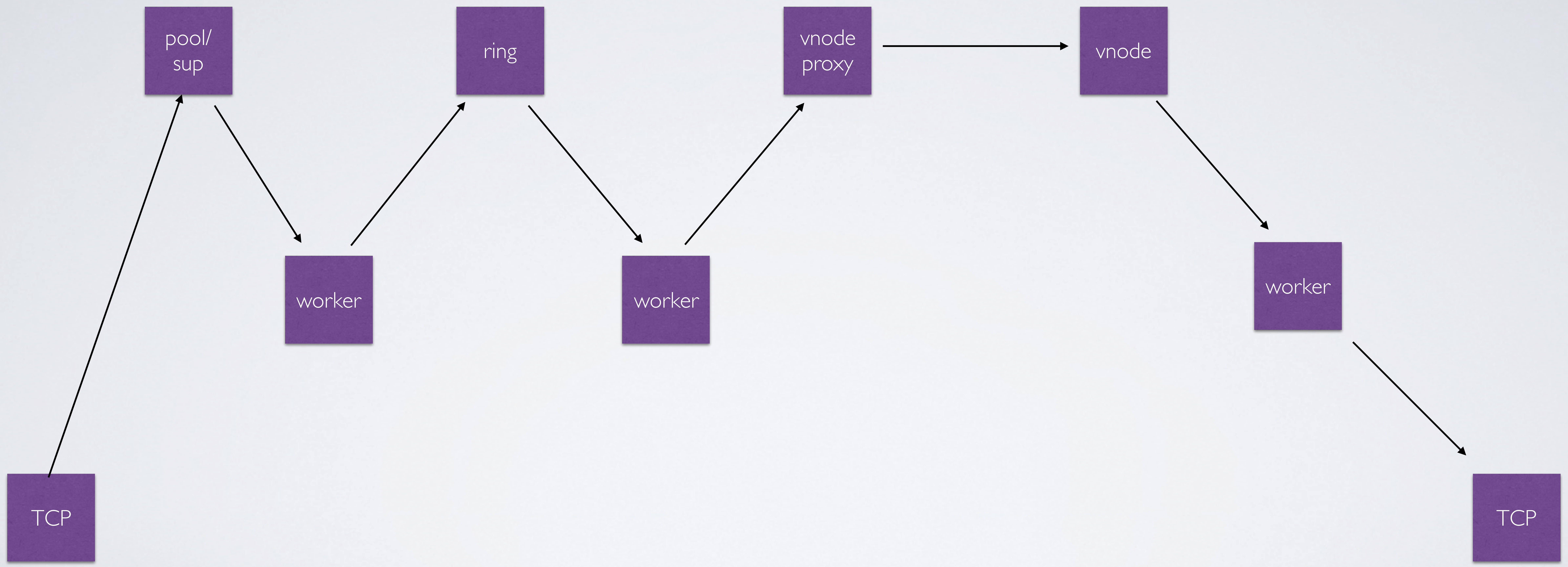
- measurements reported in periodic intervals
- always the same type
- **not an event!**
- usually viewed as agregates
- mostly stable

How many metrics **a second**?

- 5 zones on JPC highio 60.5 (61 GB Ram, 8vCPU, 1 TB Disk, no zfs compression)
- riak_core
- R/N/W=1
- overload with metrics
- 5 nodes (ring_size=64)

PROJECTFIFO

How the data flows



PROJECTFIFO

Get rid of processes!

- a bit unerlangy
- Send directly from the process handling the TCP connection
- Per connection back pressure
- no bottleneck on a pool
- no spawning of new processes

PROJECTFIFO

Cache the Ring

- Uh oh, this isn't exactly the truth … it might have changed but well do we care?
- We don't ask for the correct ring on every message
- We hope that most of the time rings don't change that often (every few seconds)
- remove ring-server as bottleneck

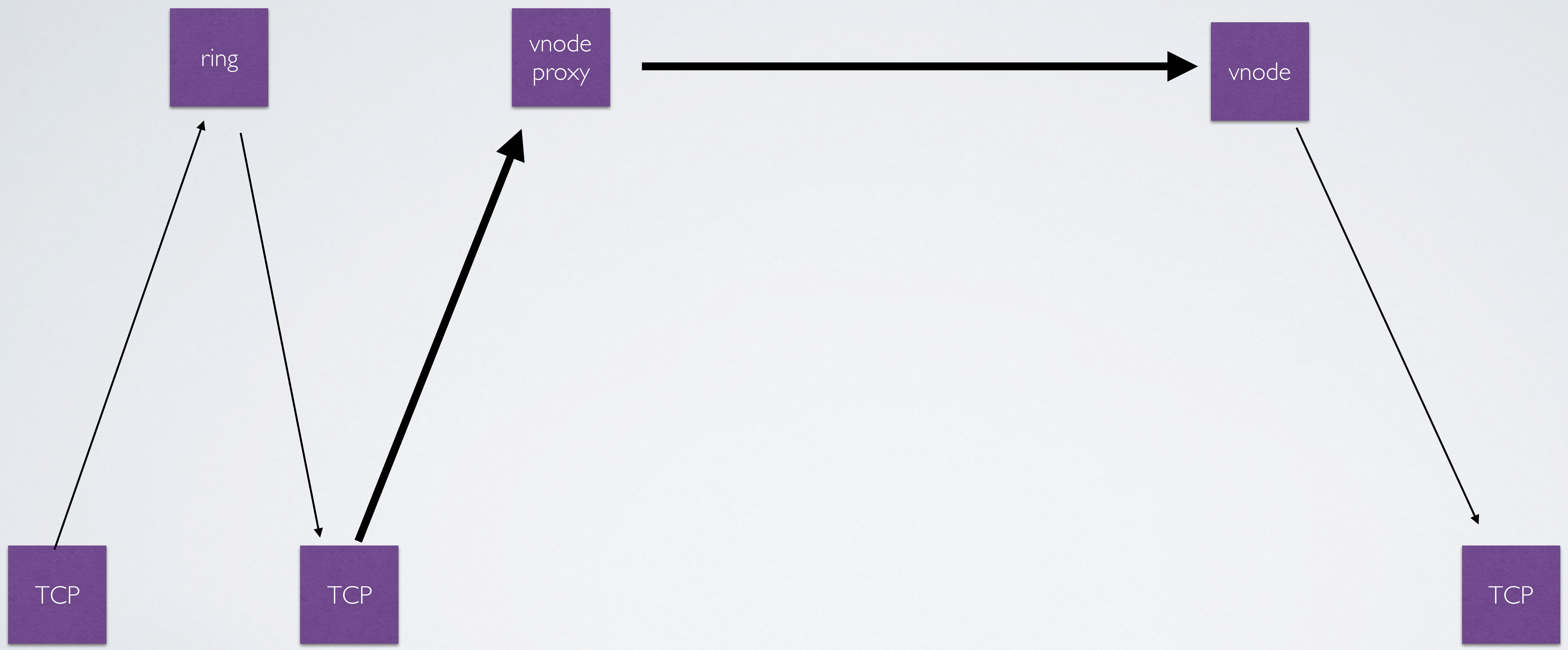
PROJECTFIFO

Introducing bk_dict

- cache the ring
- group the metrics by vnode
- periodically send them in bulk
- will happily loose them if the ring changes while data is send

PROJECTFIFO

How the data flows



PROJECTFIFO

Cache cache cache

- Cache datapoints in VNode (X consecutieve datapoints)
- Mutable memory buffer not binary.
- Don’t require total order
- flush once a datapoint is ‘behind’ the current cache
- bypass cache if a datapoint is ‘before’ the cache

PROJECTFIFO

Danger of the cache

- accept the risk of overwriting data in edge cases
  - overlapping caches after restarts
- memory consumption
- it all goes to flames when the process/node/beam crashes

PROJECTFIFO

Size matters!

- <<Int:56>> looks good?

PROJECTFIFO

Size matters!

- <<Int:56>> looks good?
- <<Int:64>> looks better?

PROJECTFIFO

Size matters!

- <<Int:56>> looks good?
- <<Int:64>> looks better?
- Nononononon.

PROJECTFIFO

Size matters!

- <<Int:56>> looks good?
- <<Int:64>> looks better?
- Nonononononon.
- BEAM treats:
  - 60 bit or less integers as native
  - 61 bit or more as bigint (10% slower)

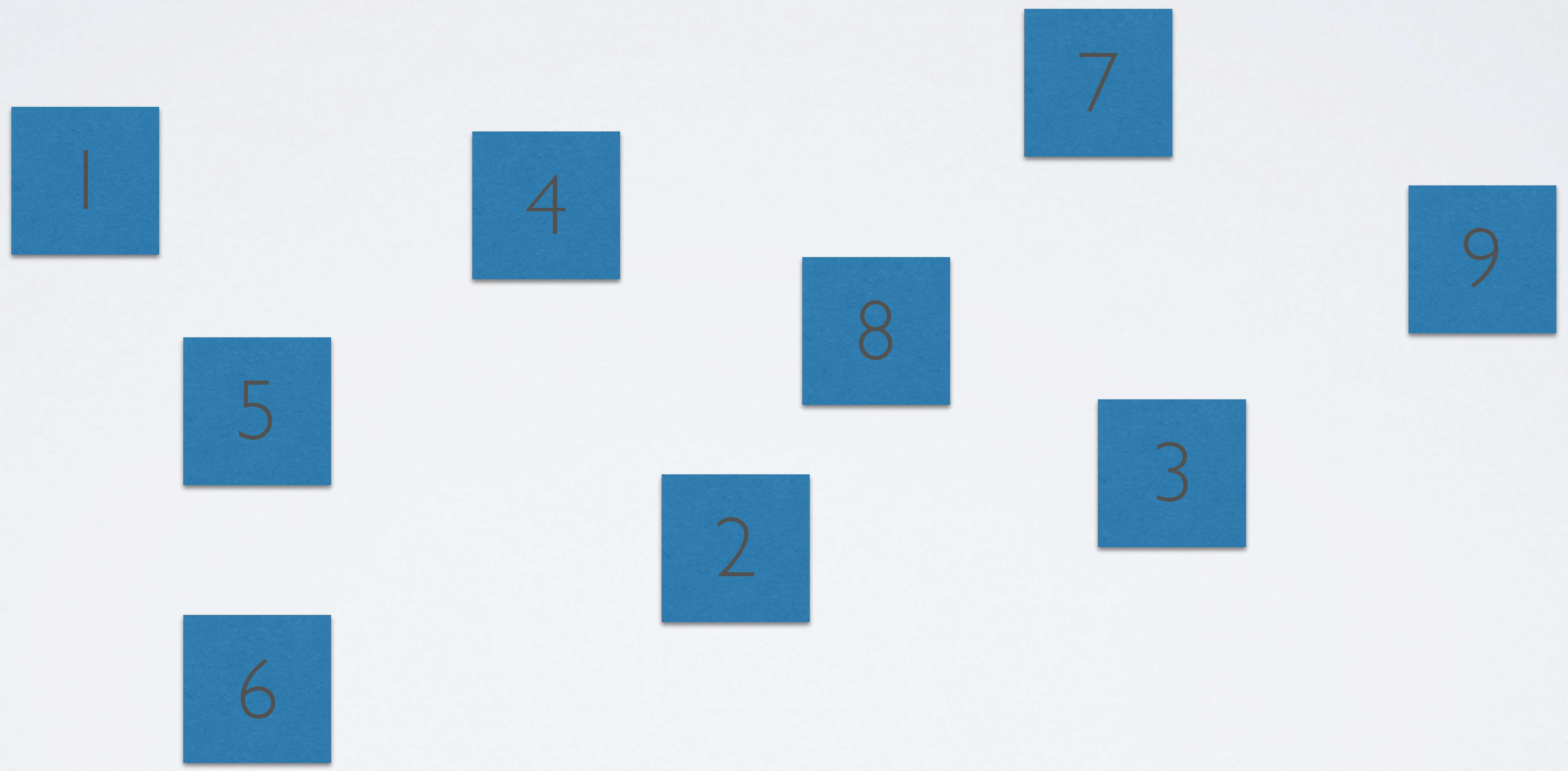
PROJECTFIFO

Split IO and Cache

- An IO Process for each VNode (cache)
- Send async as long as the IO-message queue is not growing out of hand
- Don't block VNode (cache) with disk IO
- Pass read on directly (use gen_server:reply)

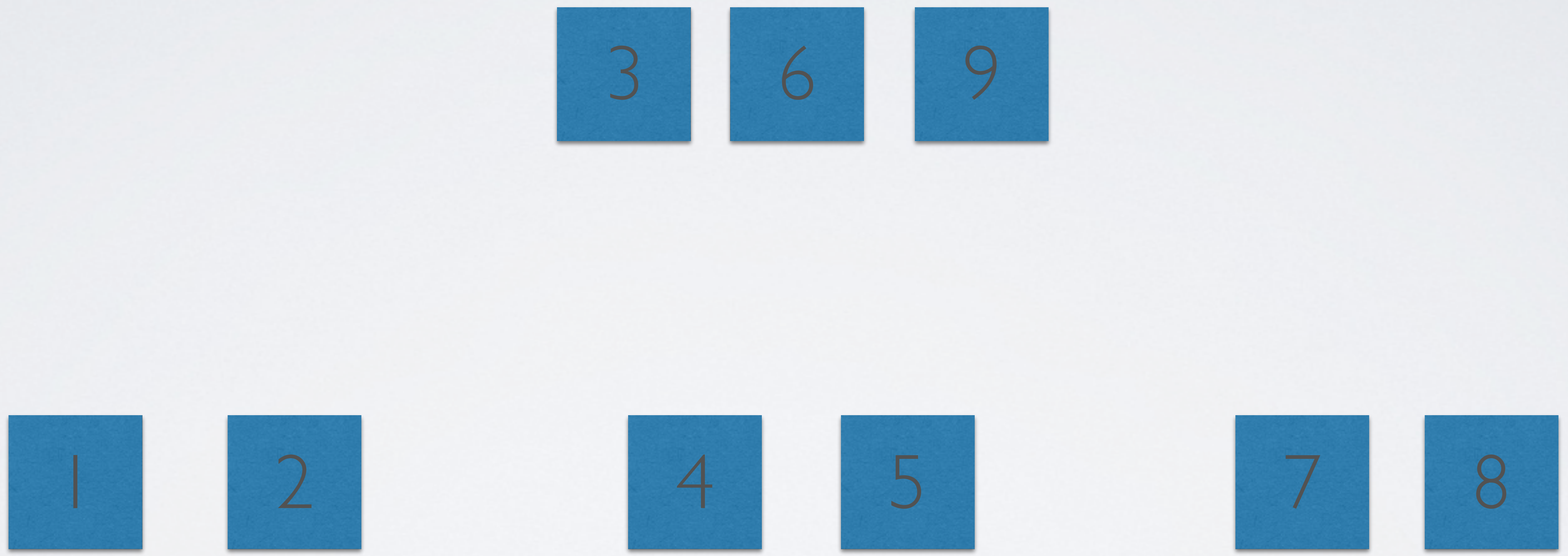
PROJECTFIFO

Storing data



PROJECTFIFO

Storing data - Tree



PROJECTFIFO

Storing data - if only …

- we had a data structure
- optimized for sequential data
- that is simple and well understood
- has constant access times for access and write

PROJECTFIFO

How a file is written

- Each file contains a fixed number of points
- each file contains as many metrics as needed
- this turns all reads and writes is **serial IO**

How many metrics a second?

- ~ 9.000.000 metrics every second
- ~ 1.5-2.000.000 per node scaling linear

Links and stuff

- @heinz_gies / @project_fifo
- DalmatinerDB: https://dalmatiner.io
- Project-FiFo: https://project-fifo.net
- Docs: https://docs.dalmatiner.io
- dFiFo driven public cloud: https://vrocket.io