Coracle

Evaluating Distributed Consensus for Real World Networks & Thoughts on Fixing it

Heidi Howard
University of Cambridge
heidi.howard@cl.cam.ac.uk

Slides: hh360.user.srcf.net/slides/sigcomm.pdf

TL;DR

We want to achieve distributed consensus beyond the typical datacenter.

Existing algorithms not sufficient to achieve this, due (in part) to limited availability.

We can do better.

Coracle → *Unanimous* → *Hydra*

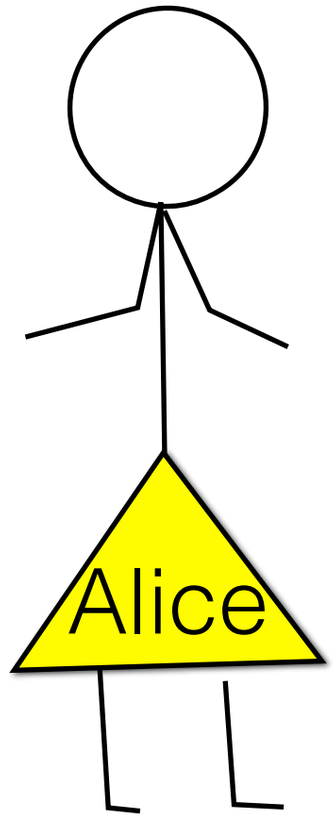
Distributed Consensus

Applications*:

- database transactions
- fault tolerant key-value stores
- distributed lock managers
- terminating reliable broadcast

*not forgetting Greek parliamentary proceedings and generals invading a city

Meet Alice



Consensus + Replication = Fault-tolerant app

Gaios *[Bolosky NSDI'11]* = Paxos + RSM

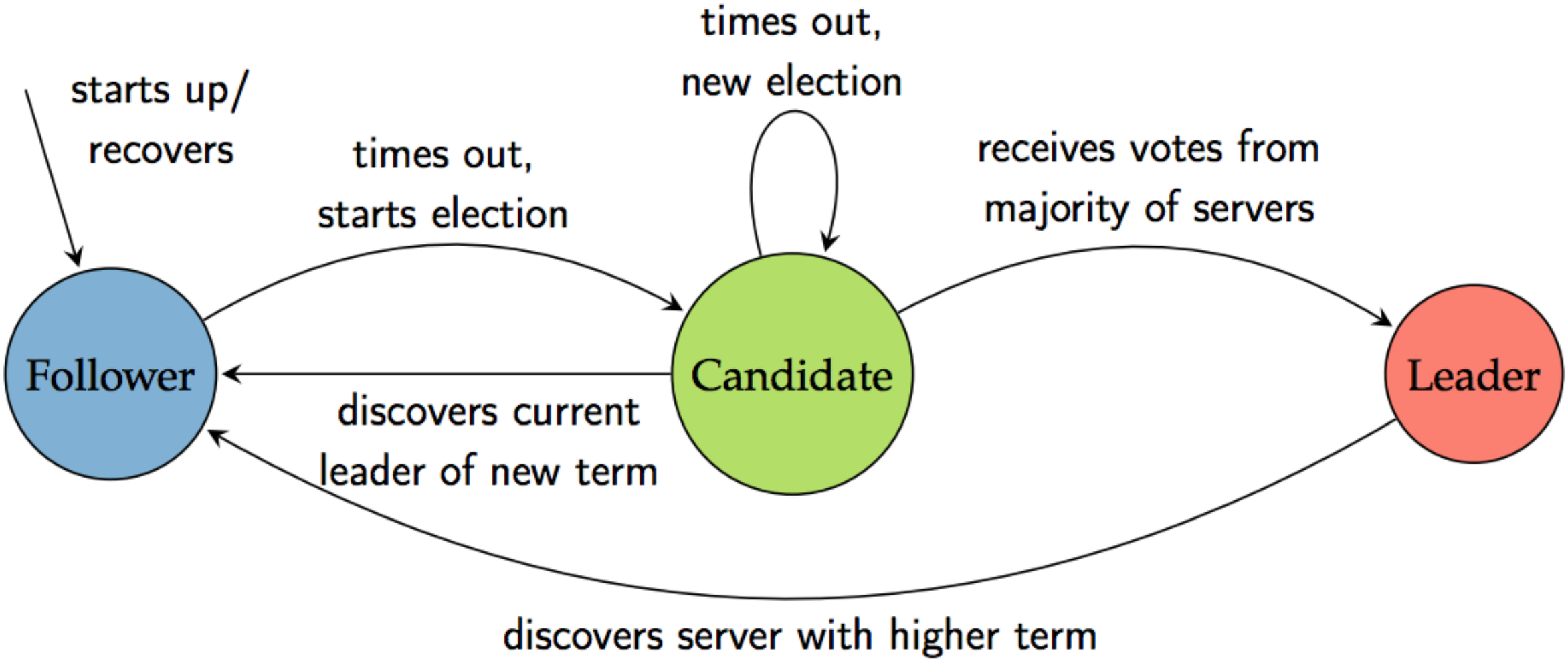
Zookeeper *[Hunt ATC'10]* = Zab + PBR

Raft *[Ongaro ATC'14]* = Raft core + RSM

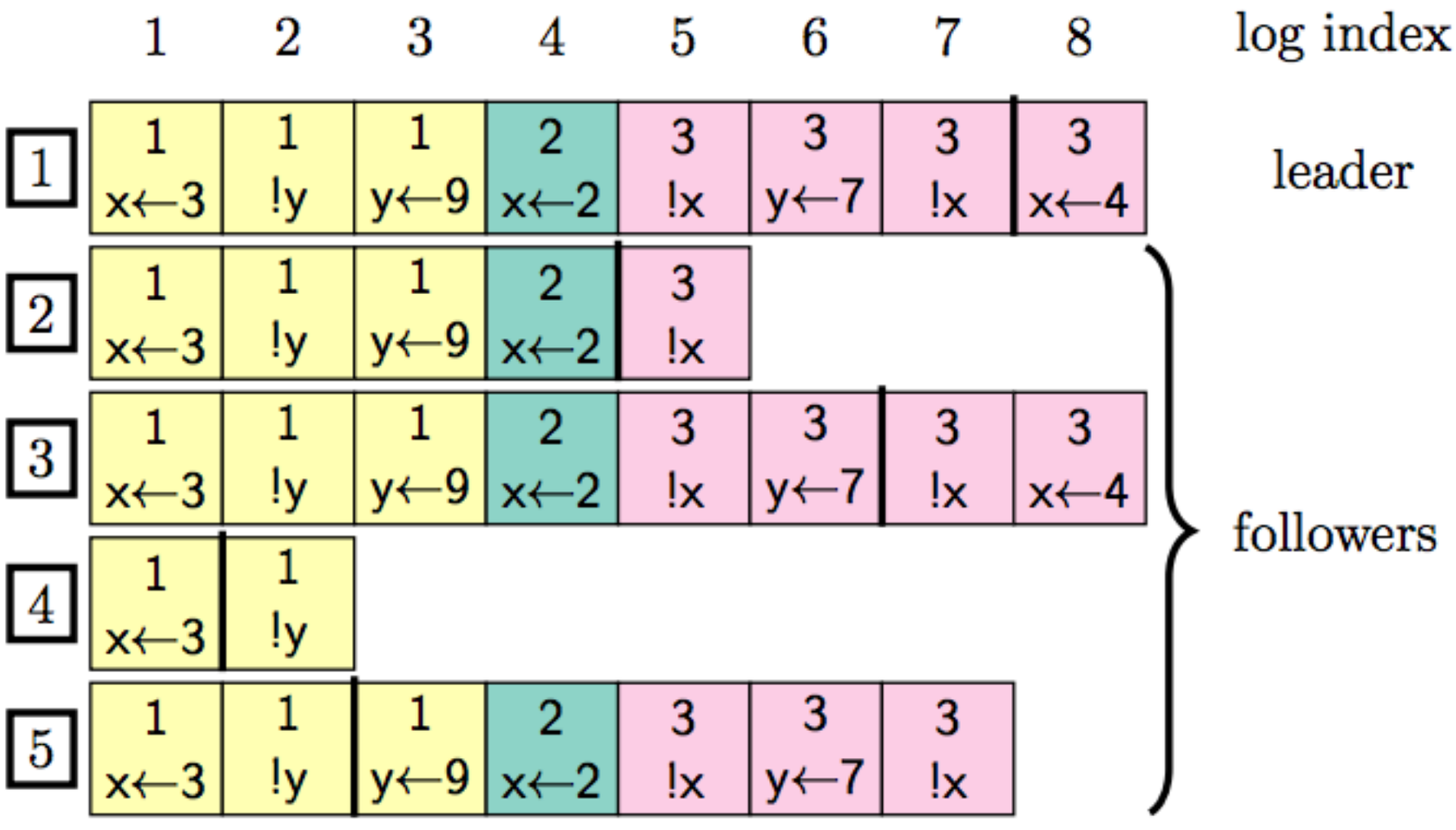
ASIDE: Raft Explained

- Leadership election
  - Modes of operation
  - Terms
- State machine replication (SMR)

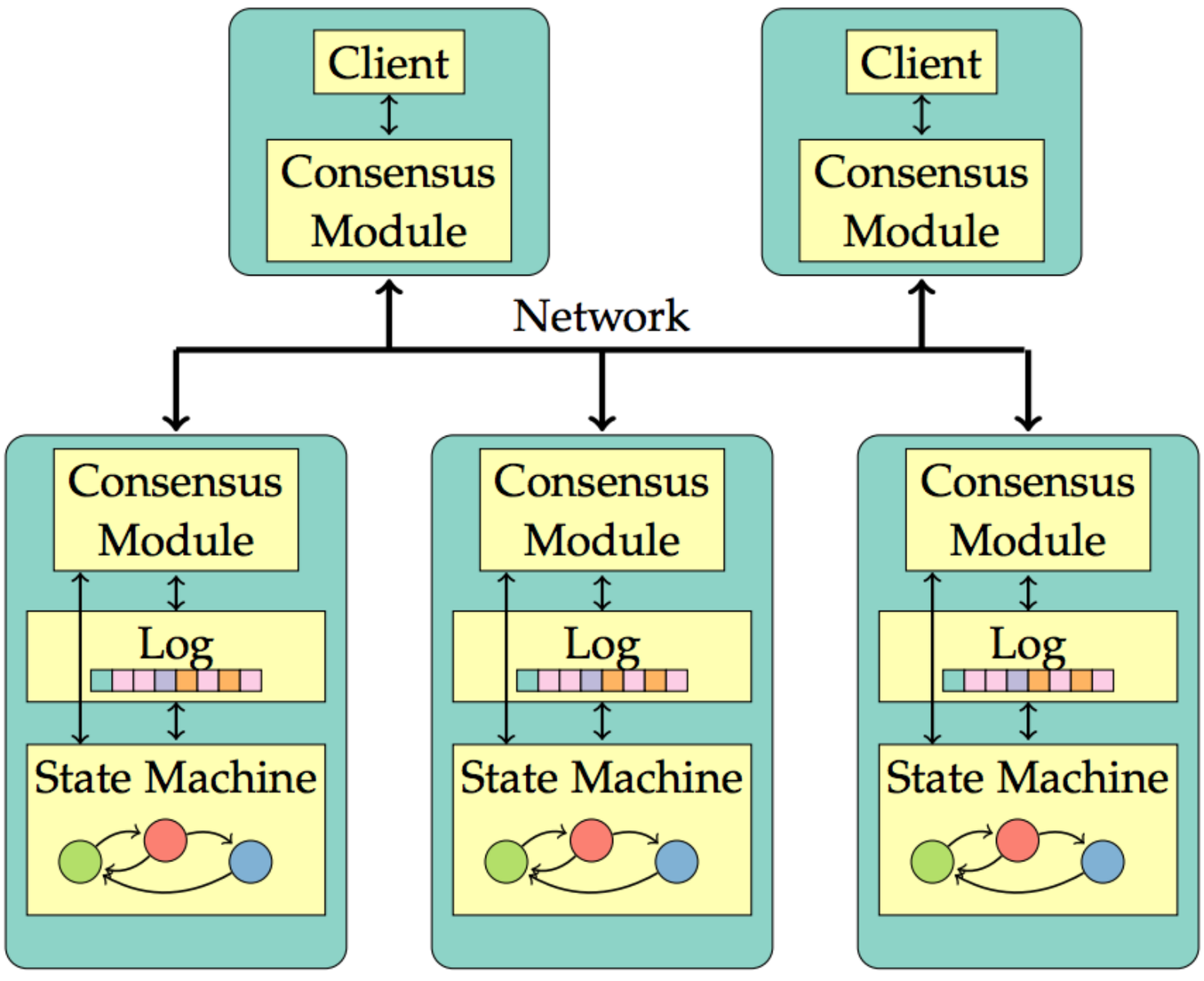
ASIDE: Raft Explained



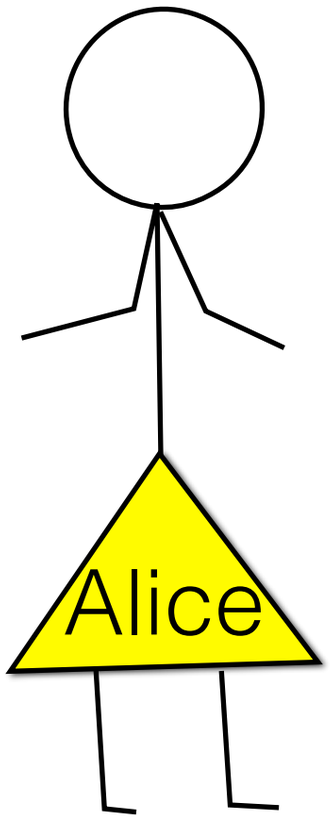
ASIDE: Raft Explained



ASIDE: Raft Explained



Returning to Alice



Alice deploys Raft consensus

Raft is proven correct

Thus, Alice can sleep well

Specified Assumptions

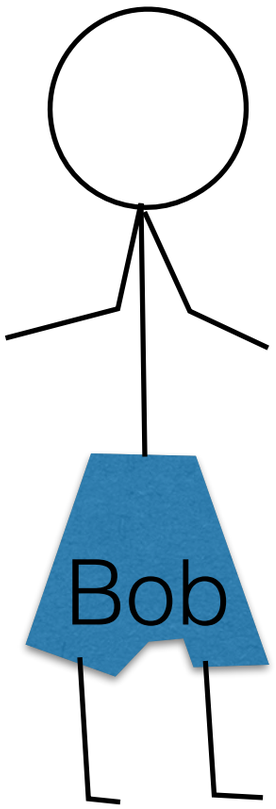
- Network communication is unreliable.
- Nodes have persistent storage that cannot be corrupted and any write will be completed before crashing.
- Asynchronous environment with faulty clocks, no bound for message delay and nodes may operate at arbitrary speeds.
- No Byzantine failures.

``They [Raft and other protocols] are fully functional (available) as long as any majority of the servers are operational and can communicate with each other and with clients. Thus, a typical cluster of five servers can tolerate the failure of any two servers.``

DEMO TIME

join in at consensus-oracle.github.io/coracle/ and click “Take me to the DEMO"

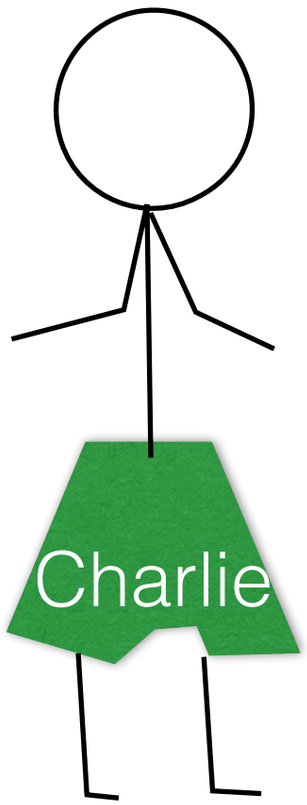
Meet Bob



Use case: Google cloud permutable VMs

Problems: node failures are common, machine migration

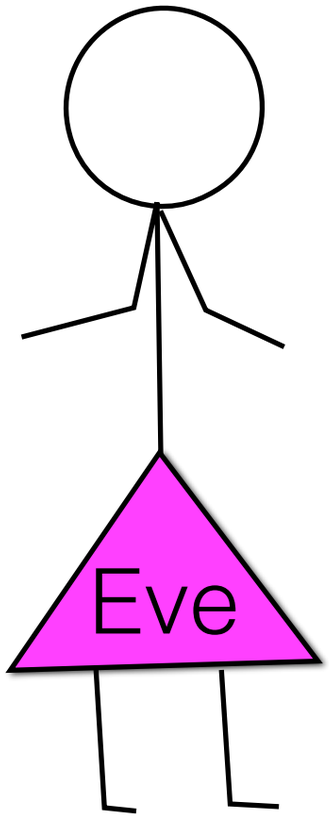
Meet Charlie



Use case: Geo-replicated datacentres

Problems: heterogeneous latency, high latency links, node clustering

Meet Eve

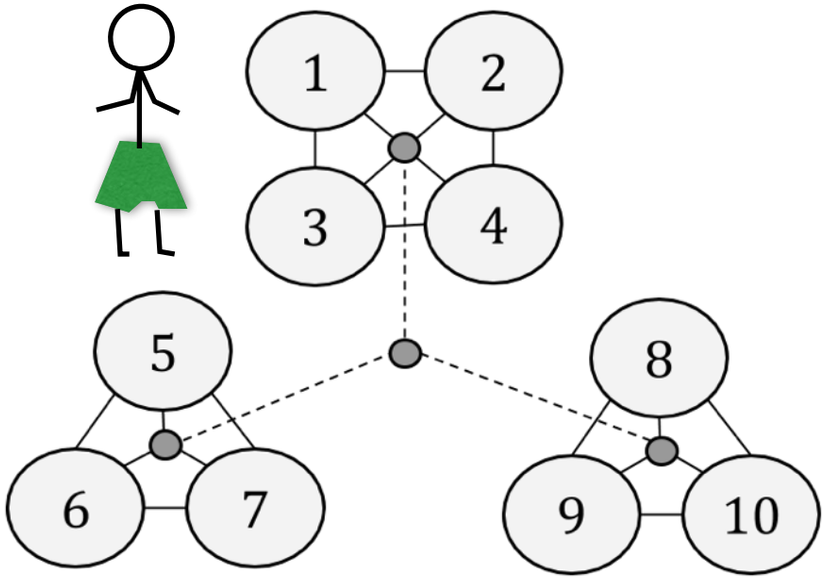


Use case: Internet edge

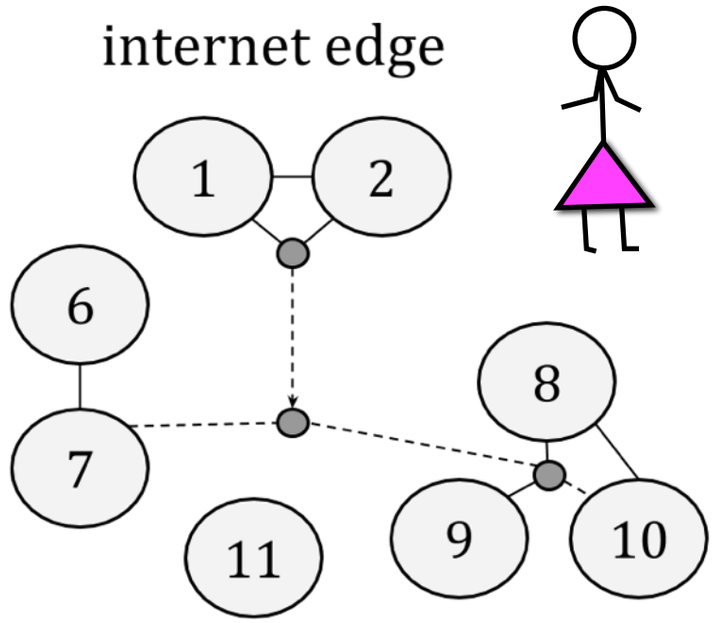
Problems: many…

heterogeneous

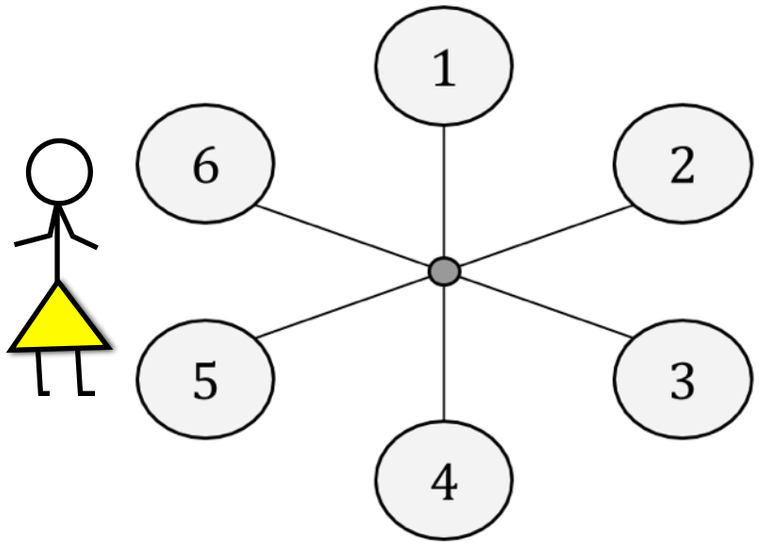
geo-replicated datacenters



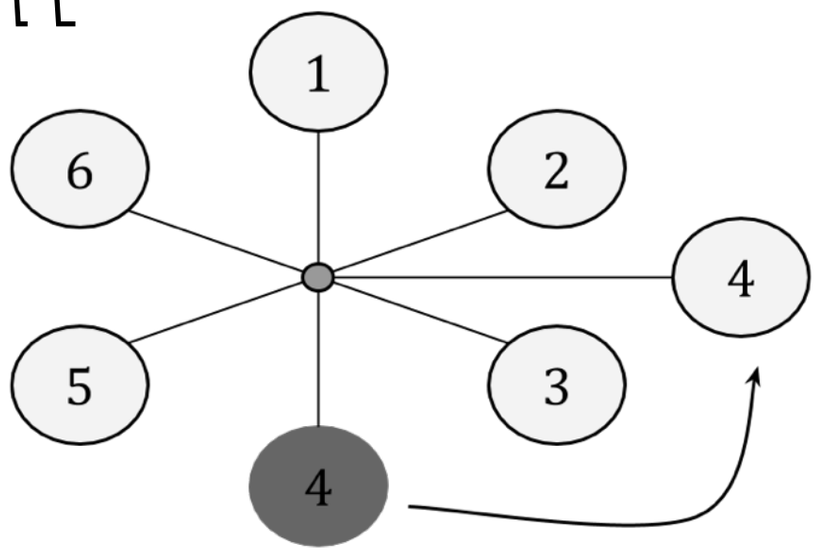
internet edge



datacenter



preemptible/
spot instance VM's



homogeneous

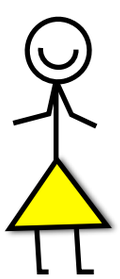
structured

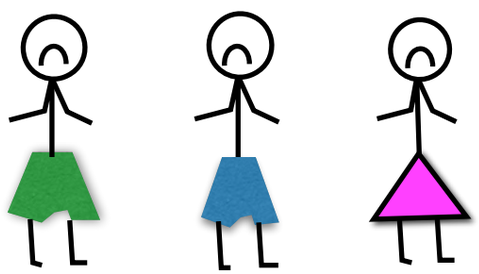
semi-structured

unstructured

New context

- Node failures are commonplace
- Network latency is unstructured and heterogeneous
- Partition are regular, possibly permanent
- Reachability between nodes may be asymmetric and non-transitive

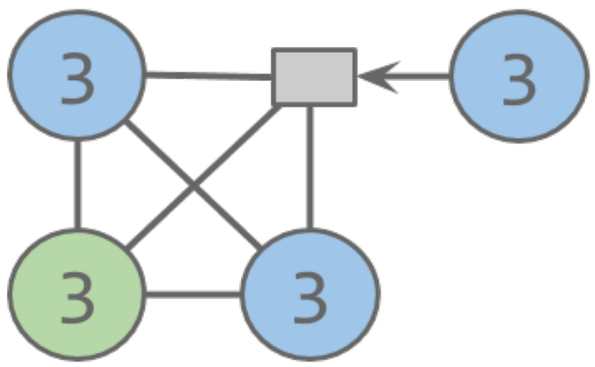


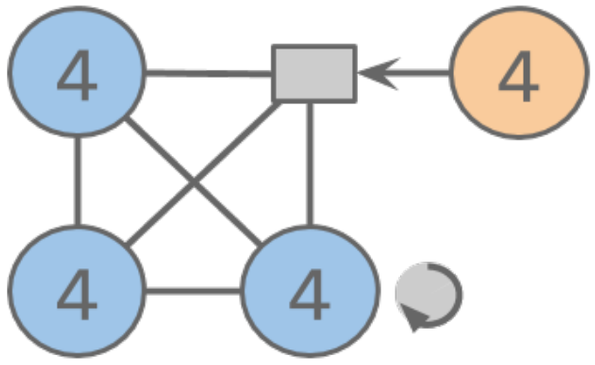


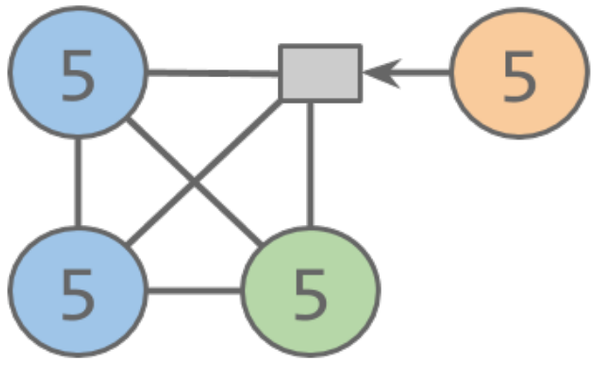
DEMO TIME

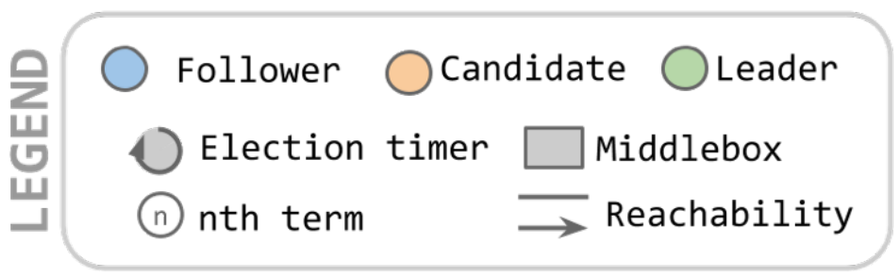
join in at consensus-oracle.github.io/coracle/ and click “Take me to the DEMO"

Backup: Example 1

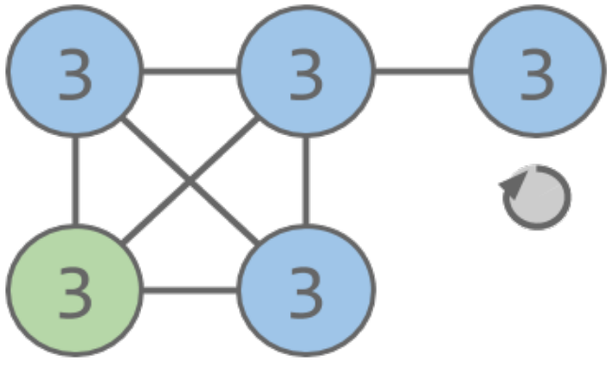


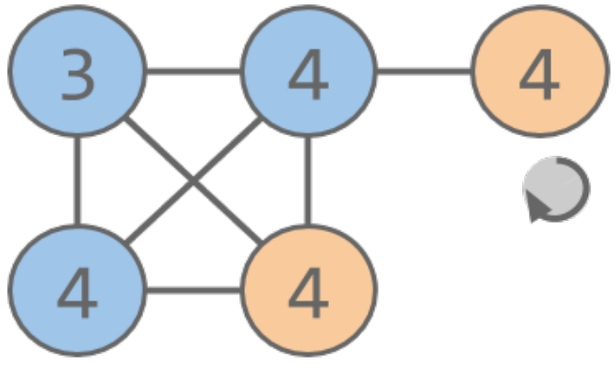


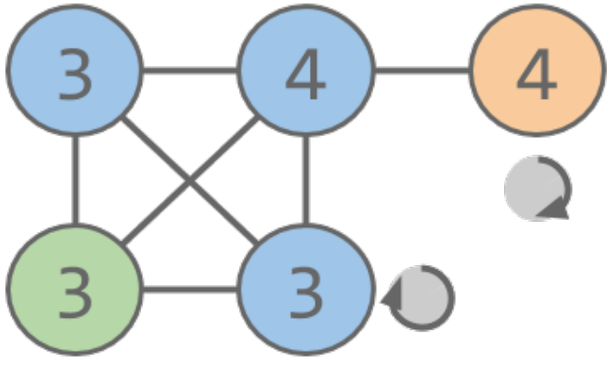


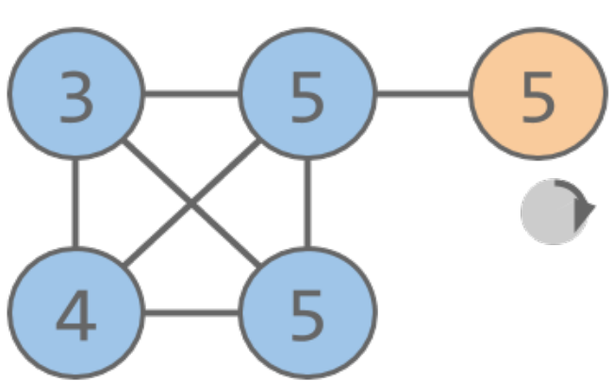


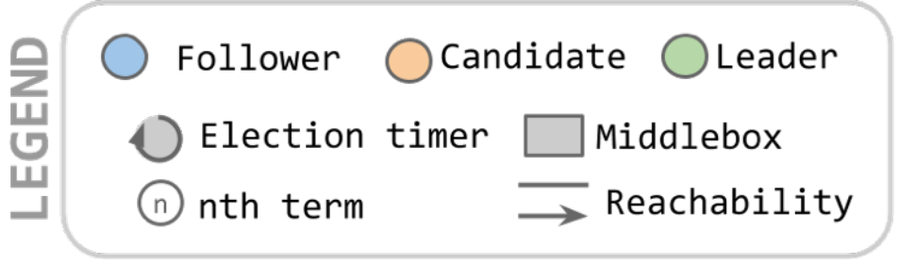
Backup: Example 2



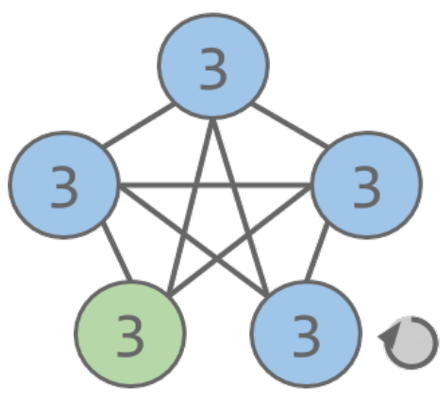


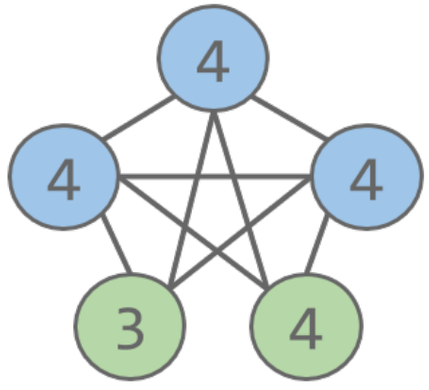


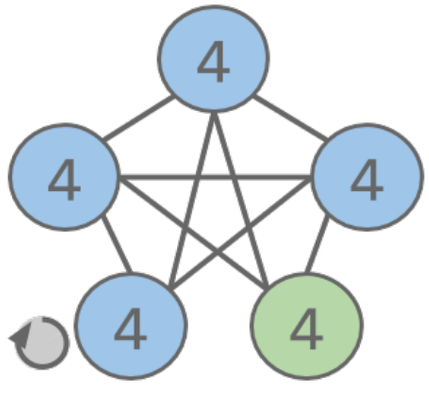


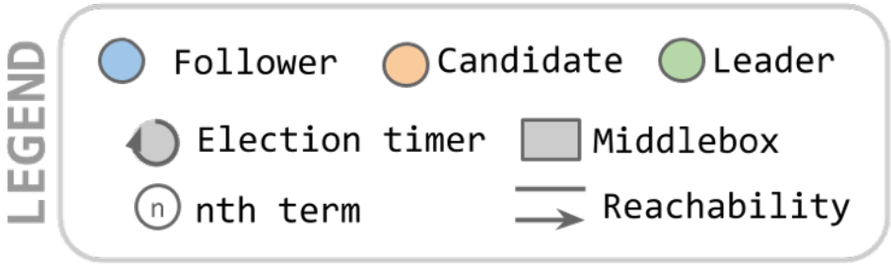


Backup: Example 3





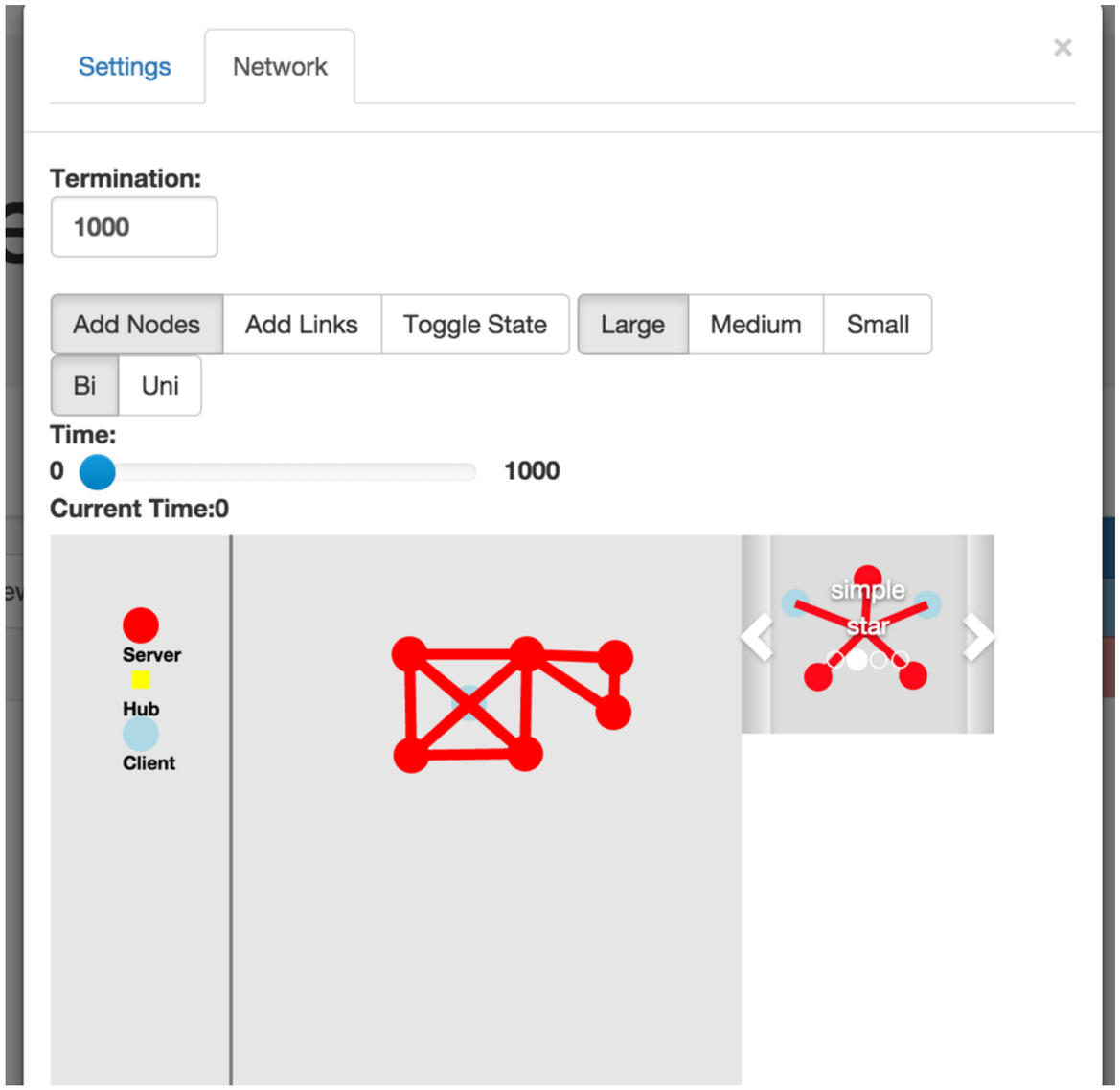




Coracle

Event based simulation of consensus algorithms on interesting networks with:

- pure protocol implementations with Unix & MirageOS support
- test suite of interesting and realistic examples



Backends

Common abstraction

Frontends

Raft

EPaxos

VRR

Consensus Protocols

SMR

DES

OCaml

Cmdline

JS GUI

IO

Unix

MirageOS

Key-value store

Lock service

Distributed Applications

Next Steps

- **Coracle:** Supporting more consensus protocols and studying real networks

- **Unanimous**: New consensus algorithm for real world networks, focused on availability.

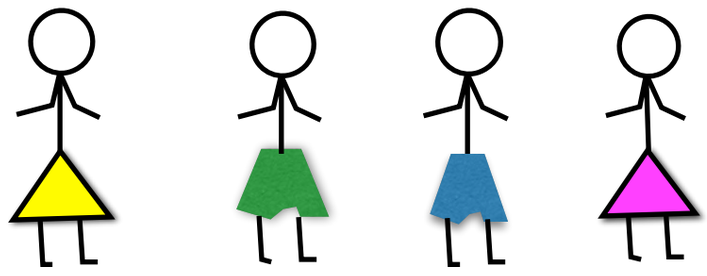
- **Hydra**: Self-scaling, self-healing services using Jitsu *[Madhavapeddy NSDI '15]* and MirageOS *[Madhavapeddy ASPLOS '13]*

Fin.

Coracle demo: consensus-oracle.github.io/coracle/

Coracle source*: github.com/consensus-oracle/coracle

Slides**: hh360.user.srcf.net/slides/sigcomm.pdf



*Code is open source under the MIT license.
**Material are released under CC Attribution 4.0 International license.