O'REILLY®
oscon

Wishful Thinking

Andrey Sibiryov, Uber SRE

oscon.com
#oscon

Who is this gentleman?

In Uber, we run over 1700 microservices in production, written in different languages. At this scale and fanout, performance of each one of them matters.

- The team I work with runs a very CPU and memory intensive Go service processing millions of requests per second.
- Millions of datapoints per second ingested.
- Millions of datapoints queried – 75 years of data / second.


JESUS
CHRIST
IS COMING!
REPENT
OF YOUR SIN

O'REILLY®
oscon

TWENTY-EIGHT SIGNS OF THE END TIMES

Andrey Sibiryov, Uber SRE

oscon.com
#oscon

O'REILLY®
oscon

The Problem

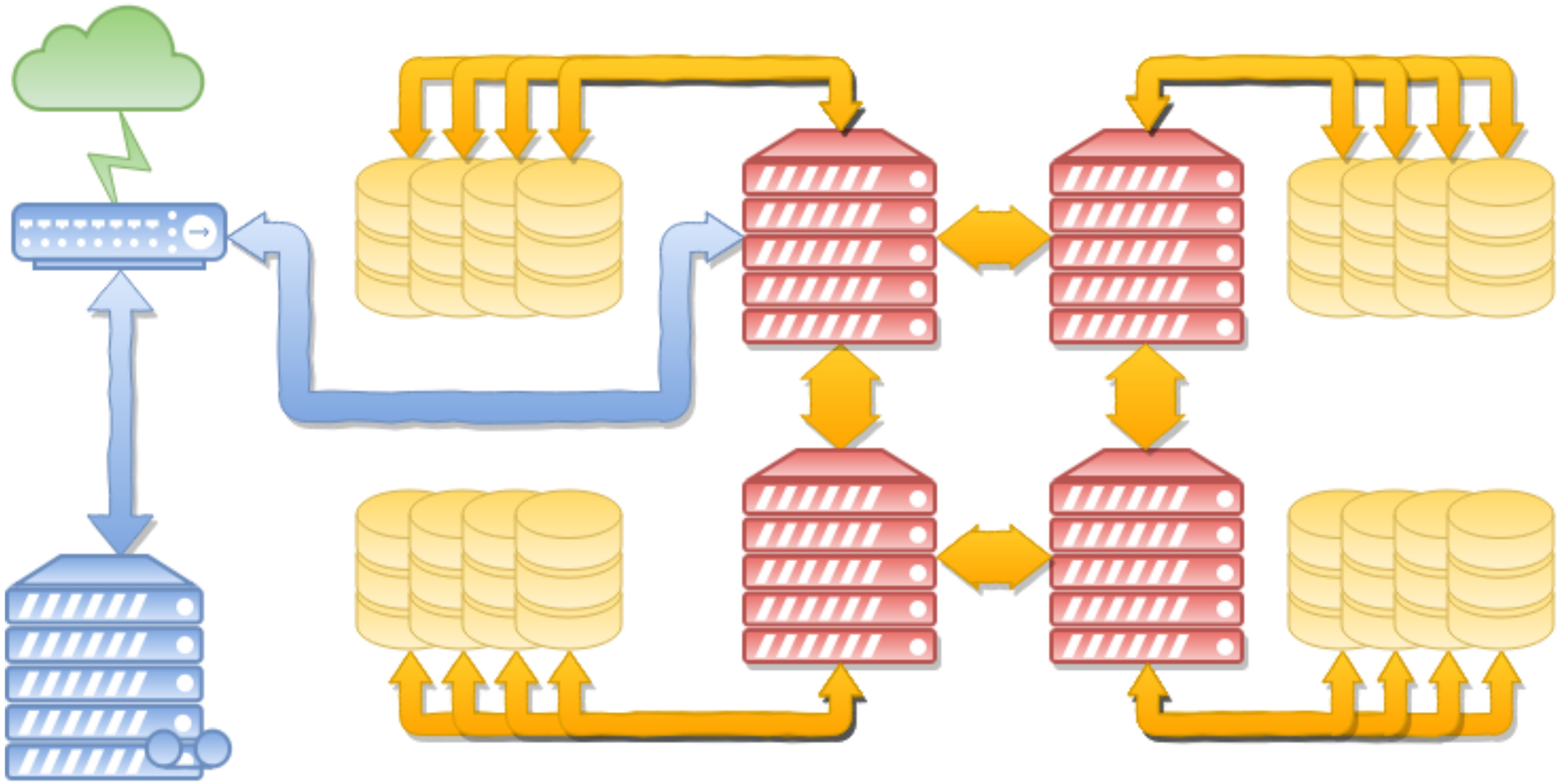
"It's complicated" – Jon von Neumann.

It’s complicated

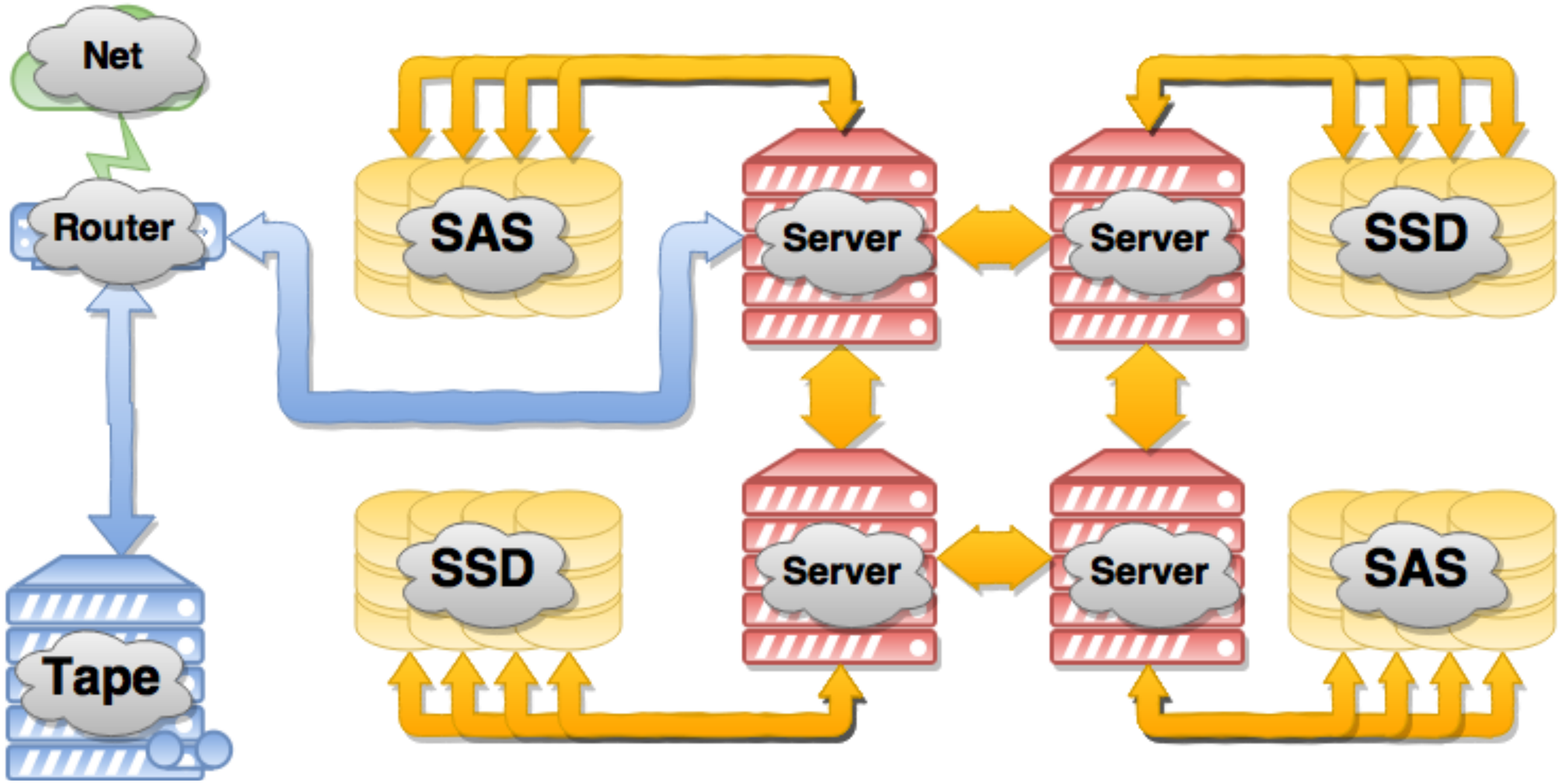
In just a few recent years, the modern hardware switched from growing in quality to growing in quantity.

- Massively multi-core, multi-socket, with deep cache hierarchies and cunning out-of-order execution pipelines.
- Same code can have different latency and throughput even when running on the same CPU.
- And almost nobody uses PMU, PEBS and so on (except Brendan Gregg).

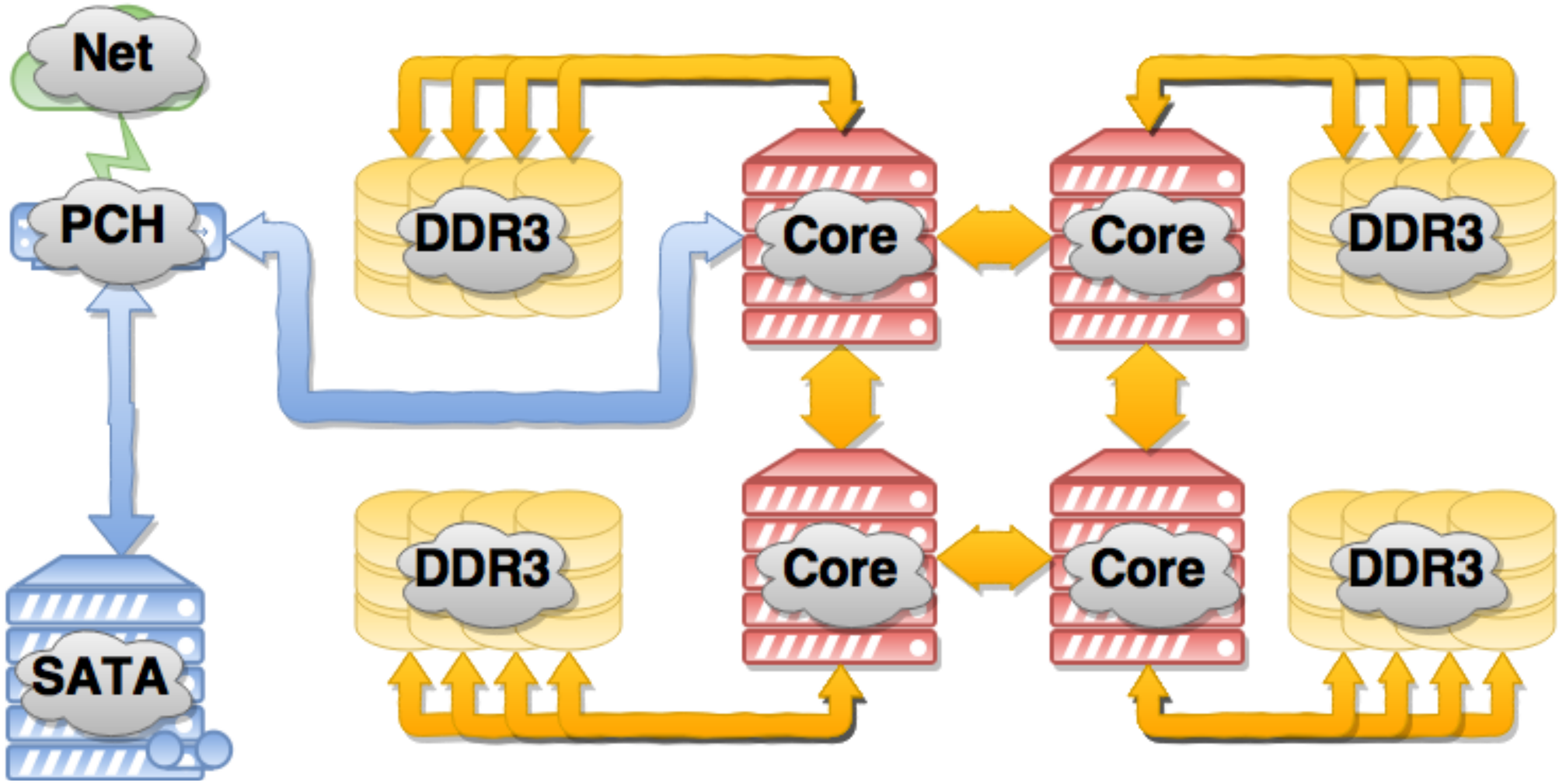
A cryptic diagram



A cryptic diagram



A cryptic diagram



«All problems in computer science can be solved by another level of indirection» – David Wheeler.

A cryptic diagram

IaaS

CaaS

PaaS

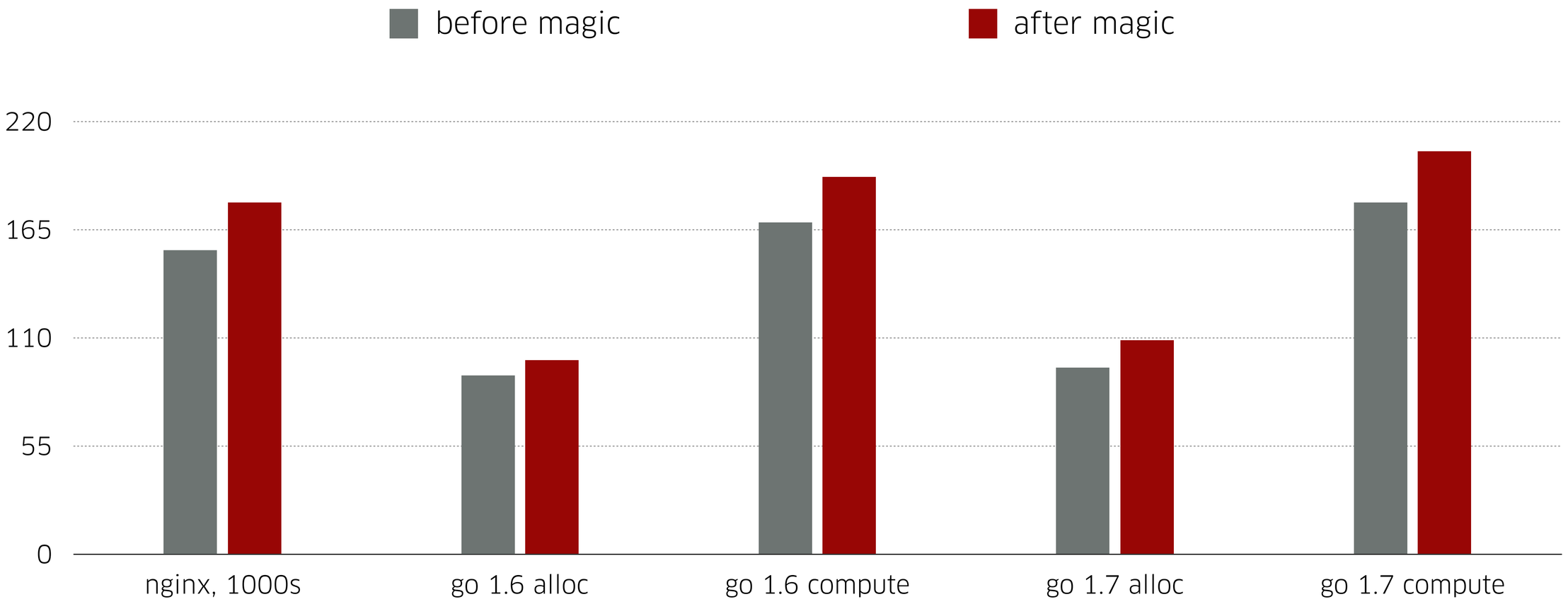
FaaS

Serverless

Clueless

O'REILLY®
oscon

Benchmarks

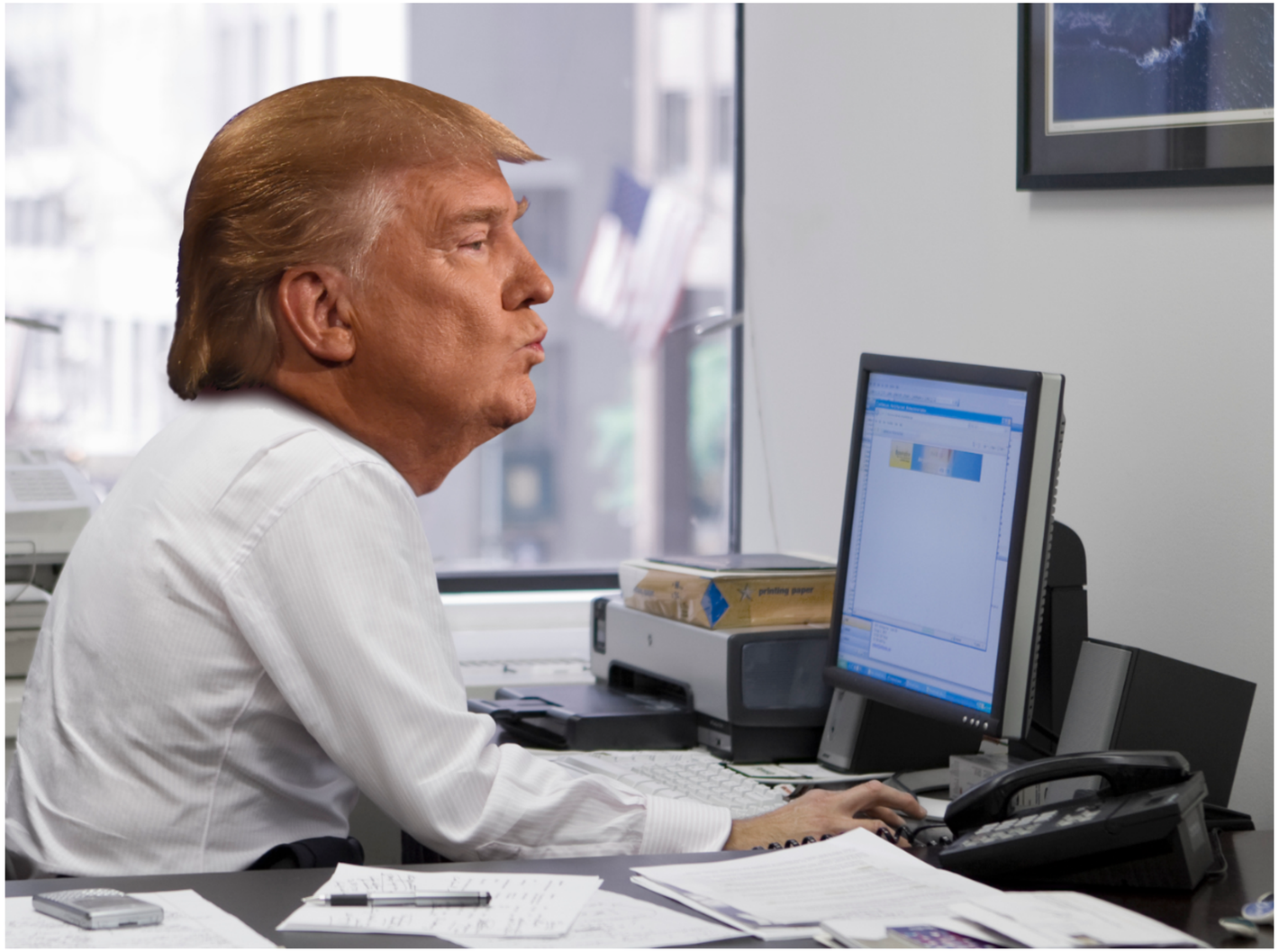

before magic
after magic
220
165
110
55
0
nginx, 1000s
go 1.6 alloc
go 1.6 compute
go 1.7 alloc
go 1.7 compute

O'REILLY®
oscon

The «Solution»

The challenge of avoiding challenges.

oscon.com
#oscon



Donald the Engineer

«**Somebody else totally did a tremendous job already**» – Donald the Engineer.

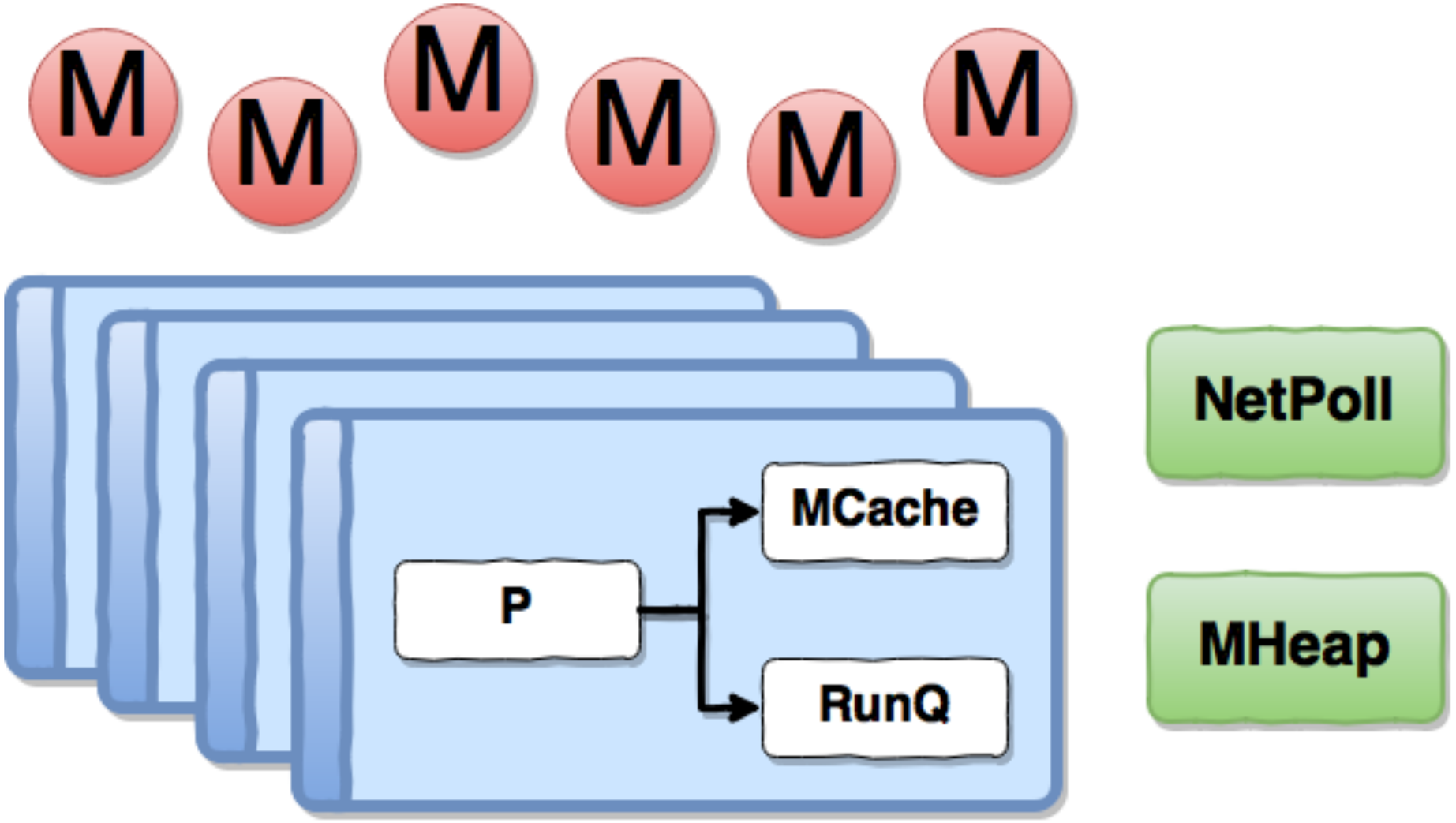

ASK ME
WHY YOU
DESERVE
HELL

«There is no code to optimize malloc for NUMA architectures, coordinate thread locality, sort threads by core, etc. It is assumed that the kernel will handle those issues sufficiently well» – libc.

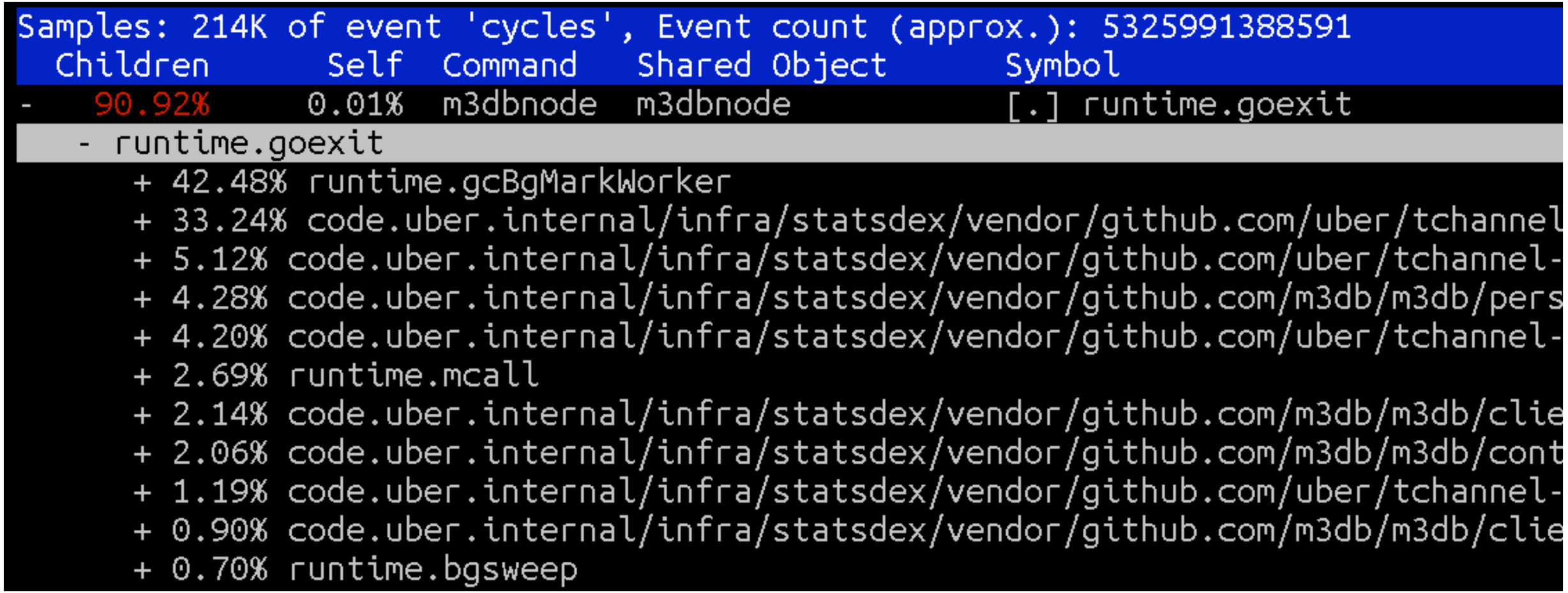
«Runtime does not try hard to ensure any locality, resources are pooled. <...> Runtime is not aware of system topology» – golang.

O'REILLY® oscon

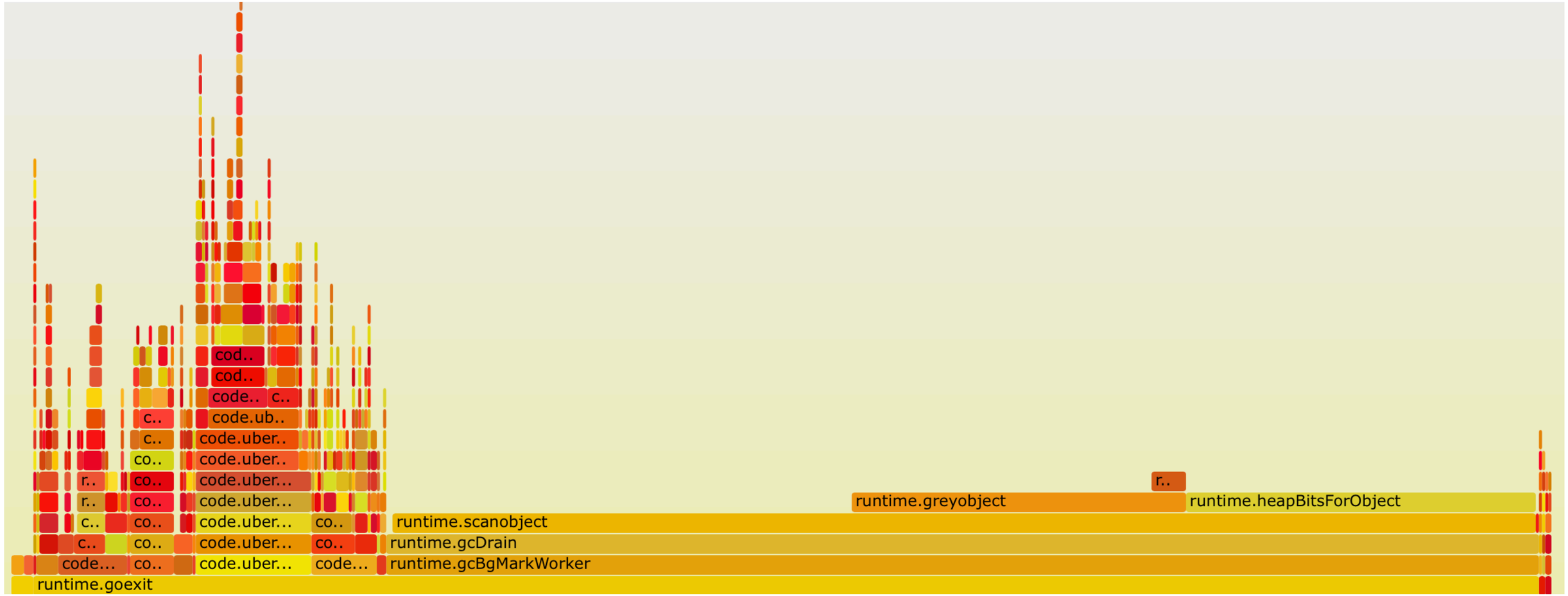
A cryptic diagram



«GC makes sufficient accesses to memory to trick kernel memory manager into moving physical memory pages to be closer to the GC workers» – kernel.


Samples: 214K of event 'cycles', Event count (approx.): 5325991388591
Children Self Command Shared Object Symbol
- 90.92% 0.01% m3dbnode m3dbnode [.] runtime.goexit
- runtime.goexit
+ 42.48% runtime.gcBgMarkWorker
+ 33.24% code.uber.internal/infra/statsdex/vendor/github.com/uber/tchannel
+ 5.12% code.uber.internal/infra/statsdex/vendor/github.com/uber/tchannel-
+ 4.28% code.uber.internal/infra/statsdex/vendor/github.com/m3db/m3db/pers
+ 4.20% code.uber.internal/infra/statsdex/vendor/github.com/uber/tchannel-
+ 2.69% runtime.mcall
+ 2.14% code.uber.internal/infra/statsdex/vendor/github.com/m3db/m3db/clie
+ 2.06% code.uber.internal/infra/statsdex/vendor/github.com/m3db/m3db/cont
+ 1.19% code.uber.internal/infra/statsdex/vendor/github.com/uber/tchannel-
+ 0.90% code.uber.internal/infra/statsdex/vendor/github.com/m3db/m3db/clie
+ 0.70% runtime.bgsweep

perf top


cod..
cod..
code..
c..
code.ub..
c..
code.uber..
co..
code.uber..
r..
co..
code.uber...
r..
r..
co..
code.uber...
c..
c..
co..
code.uber...
co..
c..
co..
code.uber...
co..
code...
co..
code.uber...
code...
r..
runtime.greyobject
runtime.heapBitsForObject
runtime.scanobject
runtime.gcDrain
runtime.gcBgMarkWorker
runtime.goexit

go-torch

«It is a future where applications scale effortlessly along with hardware and as hardware becomes more powerful the GC will not be an impediment to better, more scalable software» – golang.


THE
END
IS
NIGH

O'REILLY®
oscon

The Workaround

Computer-friendly engineering.

oscon.com
#oscon

Intuition is a lie | Confirmation Bias

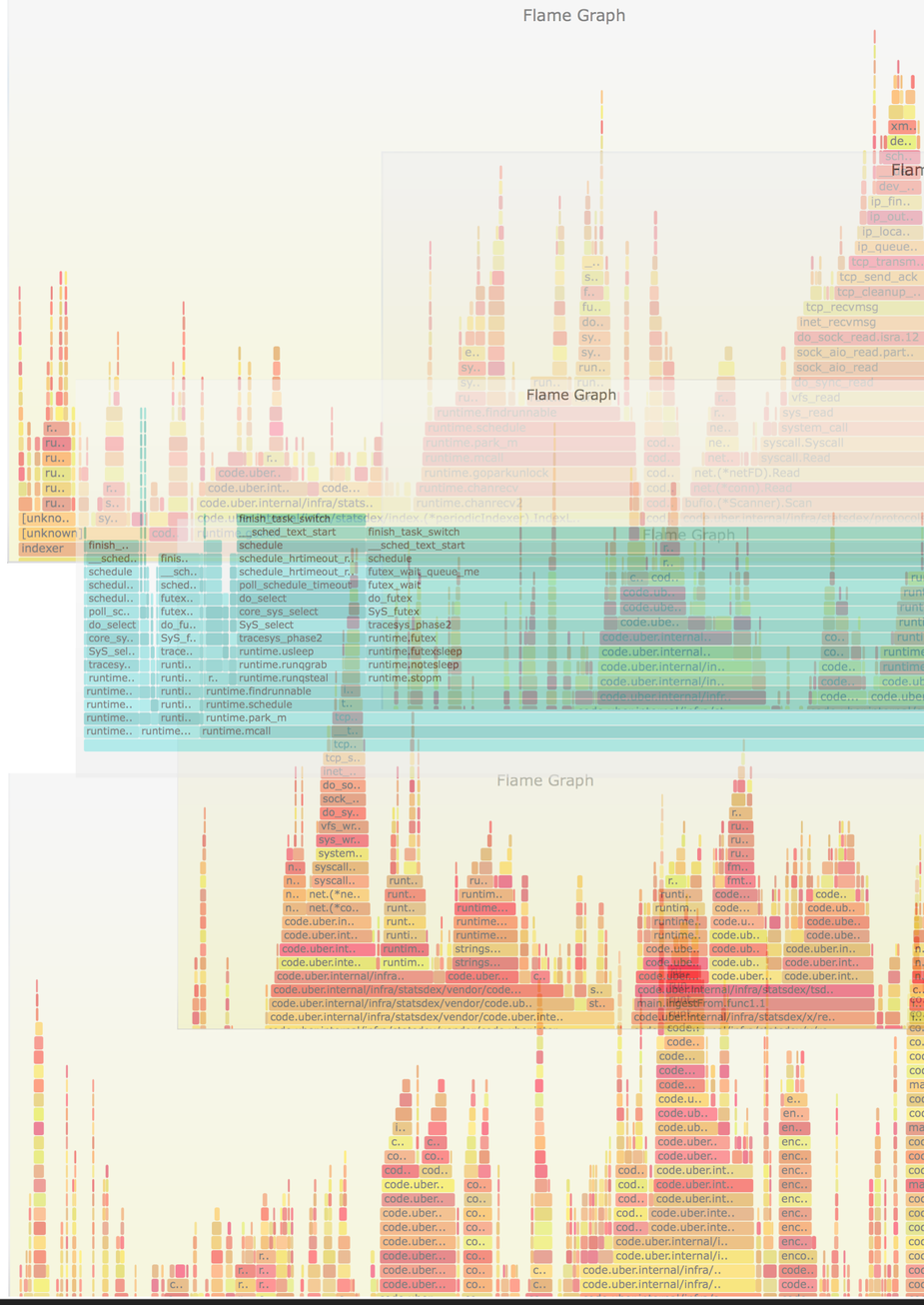
If premature optimization is the root of all evil, then engineer's intuition is a department store where you can purchase enough roots for a lifetime and get a free home delivery.

- Humans are unbelievably terrible at guessing, predicting and reasoning about computers.
- User feedback is your number one code profiling tool.
- The best way to make your code harder, better, faster & stronger is to ship it.

Profiling | Don't Be Smart

Code profiling and analytics is your only source of truth. There are tons of tools that can provide breathtaking insights into your code, both on-CPU and off-CPU.

- Generic: eBPF, Linux Perf, DTrace, Intel V-Tune.
- Language-specific: Golang's PProf, Python's cProfile.
- App-level: Prometheus, Jaeger, Datadog & more.



Sharding, Load Balancing and Topology Awareness

We can build on top of the idea that each server is a tiny network in a box!

- Let's imagine that each core is essentially a separate network node.
- We can spin up multiple instances of the same service and pin them to separate cores using e.g. Docker & containers.
- We can use a network load balancer to distribute load across physical CPU cores.
- We can even pin linked components closer to each other.

Project Tesson | Let's shard all the things!

Tesson is a tool that automatically analyzes your hardware topology to utilize it as much as possible by spawning & pinning multiple instances of your app behind a local load balancer.

- Supports different granularities: core, NUMA node, etc.
- Integrates with your favorite load balancer for seamless setup & configuration.

github://kobolog/tesson

Thank you!

Andrey Sibiryov, Uber New York SRE

kobolog@uber.com @kobolog