Sharding Containers

Andrey Sibiryov

SRE, Uber New York

dockercon 16

The Problem

«It’s complicated» –
John von Neumann.

It’s Not Fast Enough

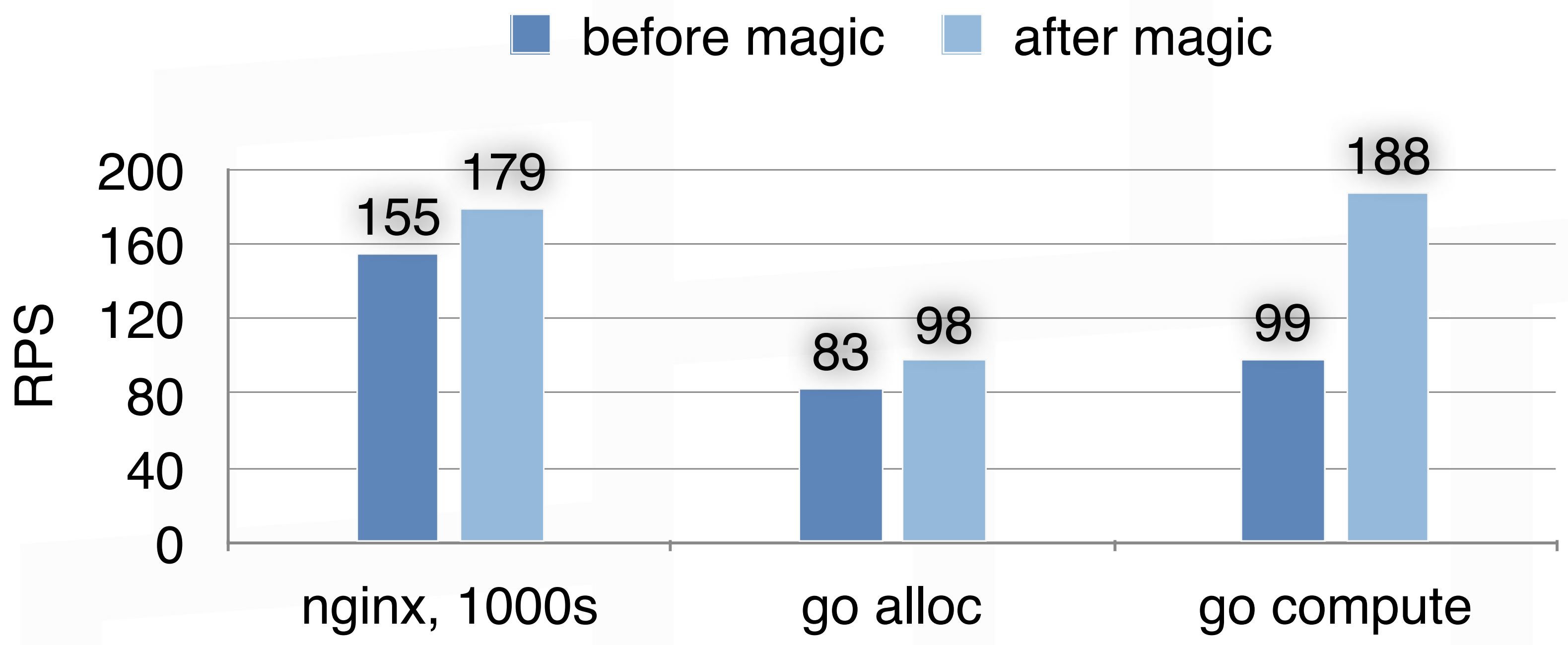
…and we can do better than this.

In Uber, we run more than a thousand microservices in production, written in different languages. At this scale and fanout, performance of each one of them matters.

- **The team I work on runs a very CPU and memory intensive Go service processing millions of requests per second.**
- **We noticed that a relatively large slice of its run time is dedicated to doing useless things – GC, context switching, CPU stalling for memory access and so on.**

Benchmarks!

…and unexpected numbers.



It’s Complicated

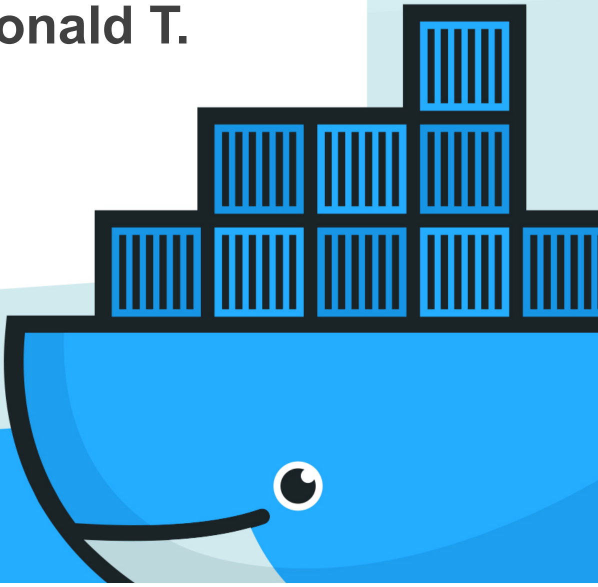
Sockets, Cores, HTs & NUMA.

In just a few years, the modern hardware switched away from growing the CPU power to growing CPU cores and caches.

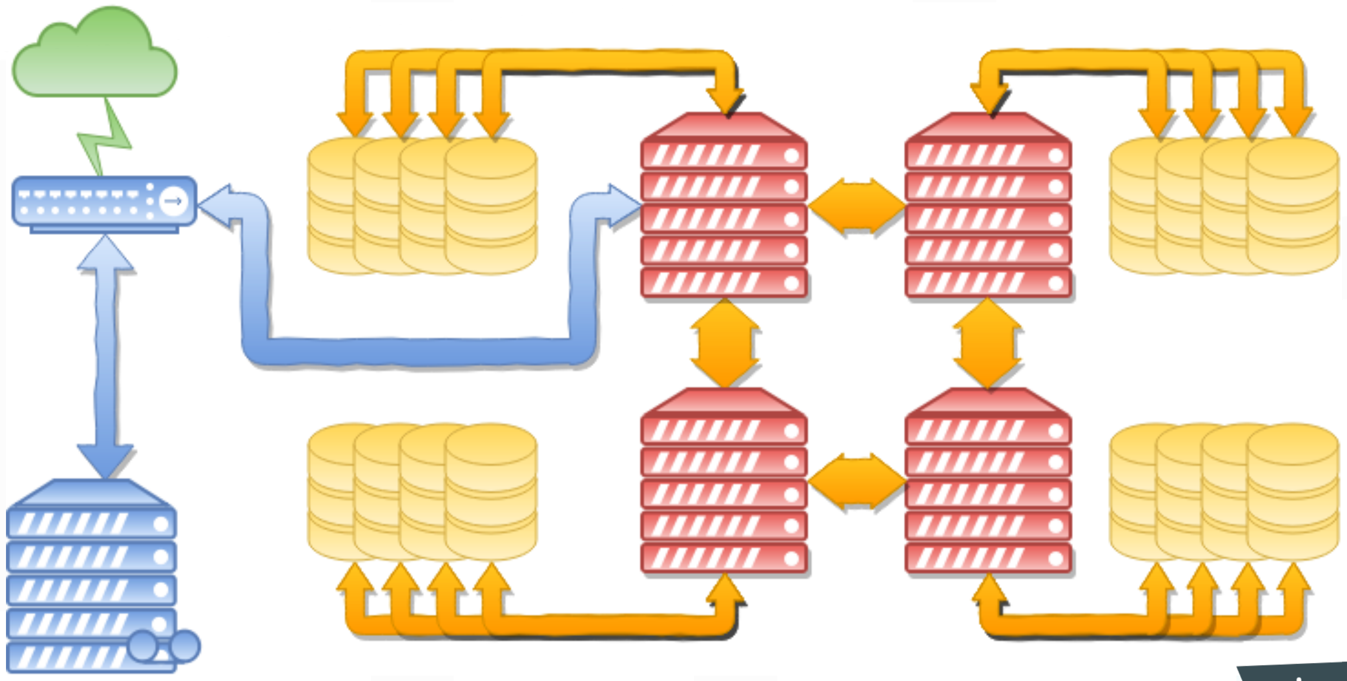
- **Massively multi-core, multi-socket, with deep cache hierarchies and cunning out-of-order execution pipelines.**
- **Same code can have different latency and throughput even when running on the same CPU.**
- **Also, almost nobody uses PMU, PEBS and so on except Brendan Gregg.**

“Crooked Moore’s law doesn’t work anymore!”

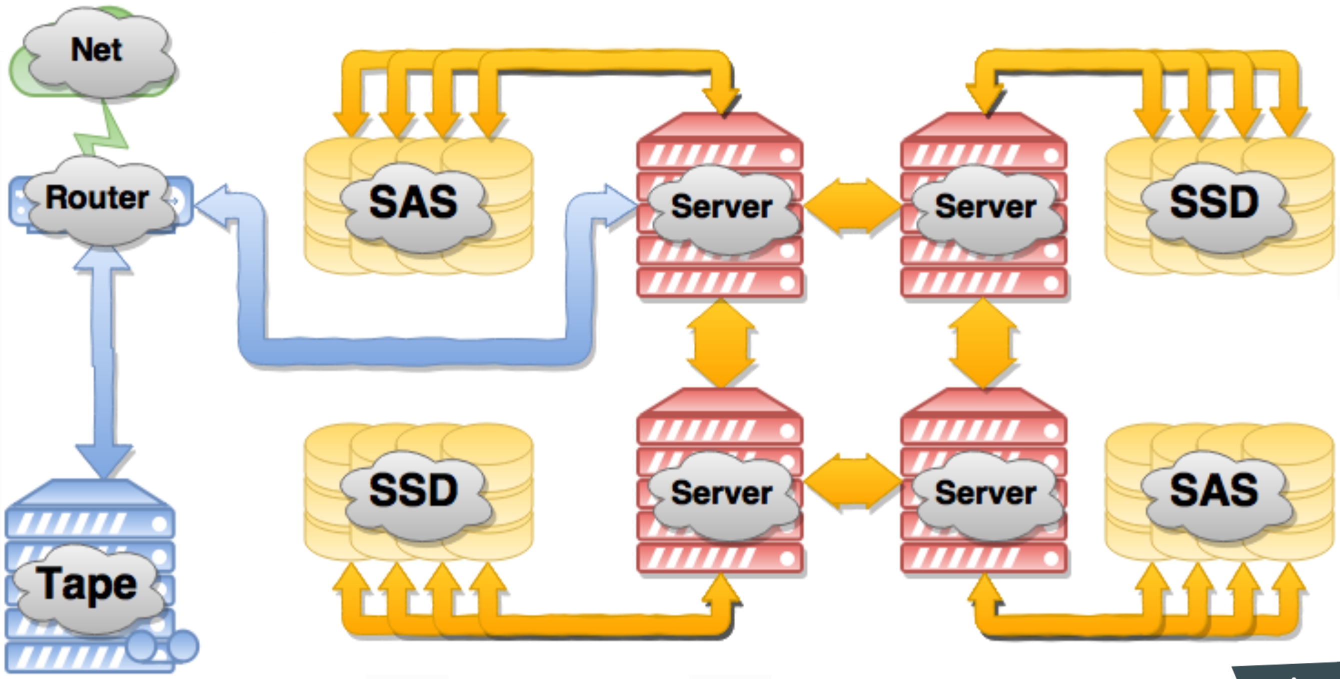
— Donald T.



I: A Cryptic Diagram

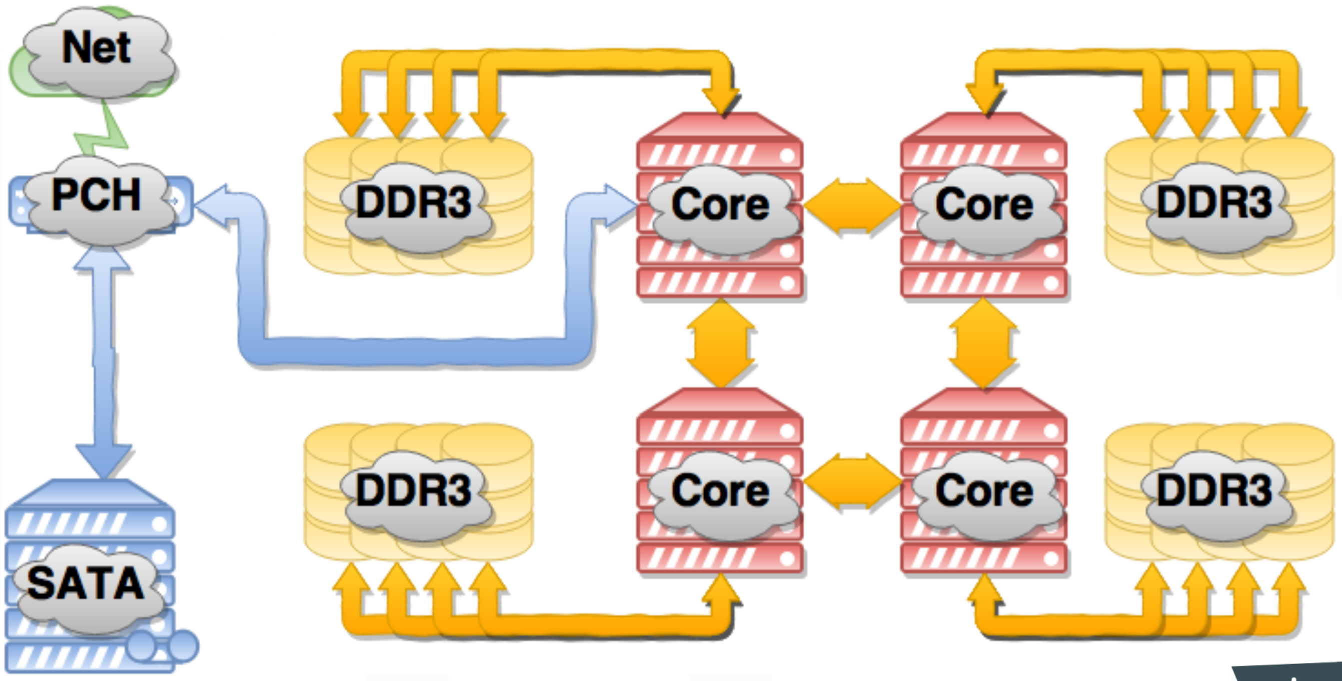


II: A Cryptic Diagram



dockercon 16

III: A Cryptic Diagram



It’s Complicated

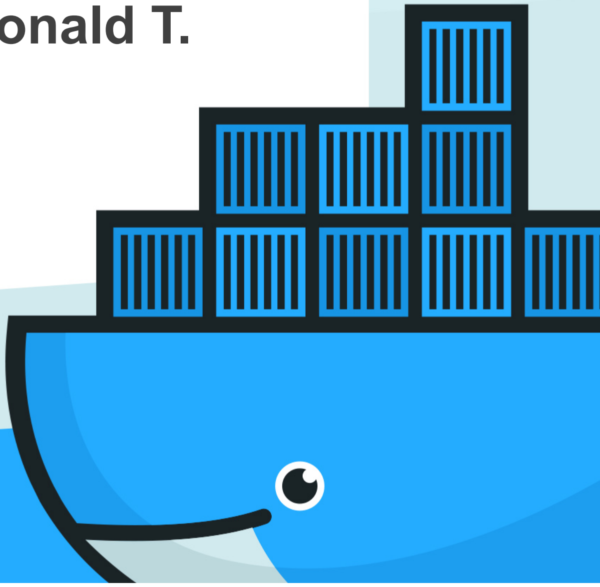
Devices, Interrupts & Latency.

Growing gap between performance characteristics of different buses and components and multi-level caching end up introducing more and more hidden lag to all operations.

- **A single core in not capable of processing input from a NIC.**
- **Some applications are even forced to switch to userspace-based polling to achieve full performance.**
- **Computers grow in complexity: adding queues, buffers & offload techniques at expense of transparency.**

“Computers are networks-on-a-chip, literally!”

— Donald T.



The “Solution”

The challenge of
avoiding challenges.

dockercon 16

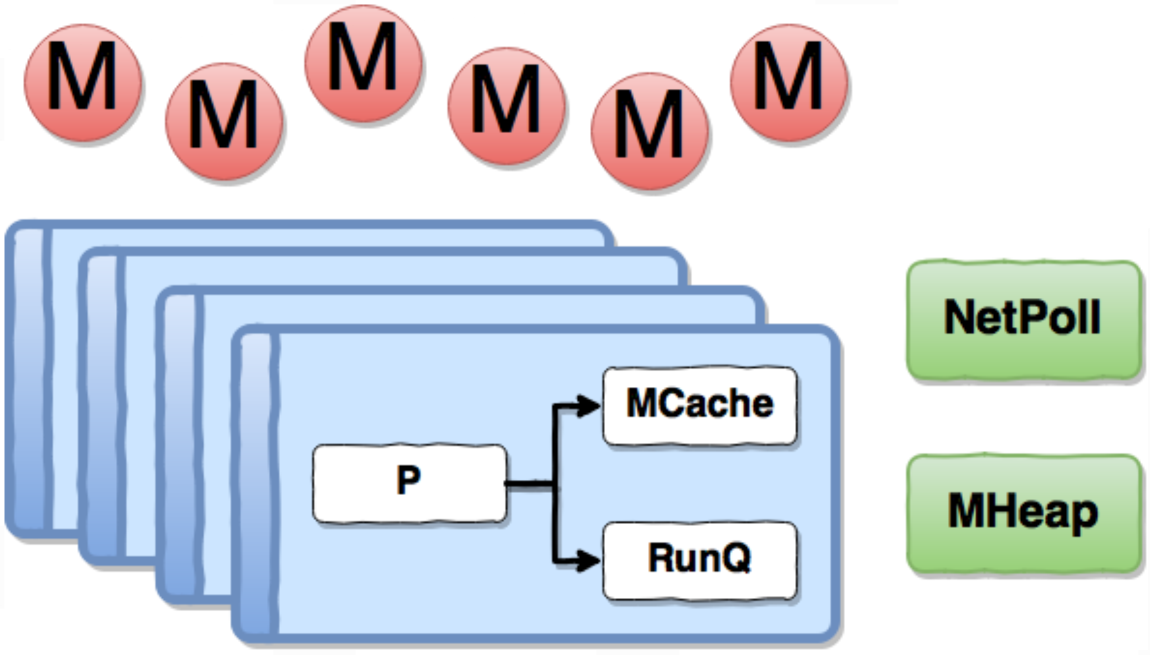
Wishful Thinking

…somebody already thought this through, right?

Modern language runtimes and VMs chose the way of least resistance in hope that OSes will take care of the complexity of the underlying hardware. You'll never believe what happened next:

- **Golang Issue #14406 – «… GC makes sufficient accesses to memory to trick Linux's NUMA logic into moving physical memory pages to be closer to the GC workers».**
- **<Maybe another example?>**

IV: A Cryptic Diagram



Wishful Thinking

We've put a VM into your VM so you can stall while you stall.

Multi-layer abstractions, over-engineering and multiple indirections are the current trends of software development. The level of abstraction engineers work on now is as remote from real hardware as we are from planting potato on Mars right now.

- **VMs & transpilation, garbage collectors, abstract hardware models.**
- **Running in multiple nested virtual machines with virtualized networking.**

The Workaround

Computer-Friendly
Engineering.

dockercon 16

Sharding

Databases is not the only thing you can shard.

Shard (n.) – A **shard** is a horizontal partition of data in a database or search engine. Each shard is held on a separate server instance, to spread load.

- **In fact, we can shard whatever we want.**
- **In fact, we already do this: load balancing is essentially sharding of your whole backend infrastructure.**

Load Balancing

It's not only for networks.

A **load balancer** distributes workloads across multiple computing resources, such as computers, a computer cluster or network links. It aims to optimize resource use, maximize throughput, minimize response time, and avoid overload of any single resource.

- **Normally, load balancers are used to distribute traffic across network nodes.**
- **In fact, we can use a network LB to distribute load across physical CPU cores.**

Pinning

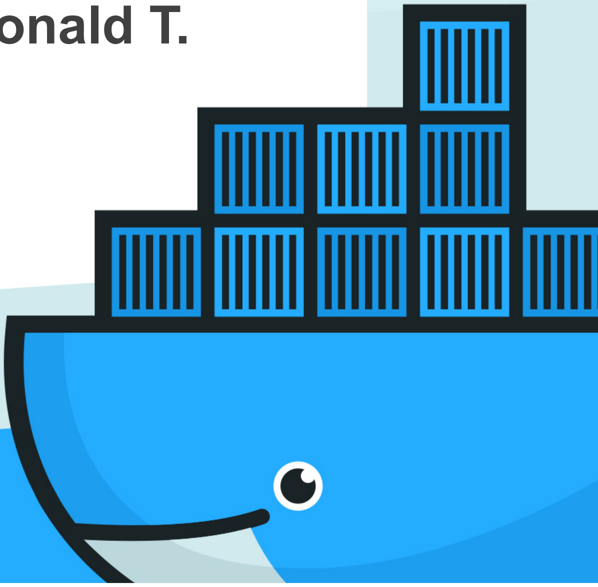
Tactics 101: use the terrain.

Network topology is the arrangement of the various elements (links, nodes, etc.) of a computer network.

- **In modern computers, each core is essentially a separate network node.**
- **Docker supports CPU pinning. This way, we can spin up multiple instances of the same container and pin them to separate cores.**
- **We can even pin linked components closer to each other.**

“Make programs computer-friendly again!”

— Donald T.



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 Gorb for seamless local load balancer setup & configuration.**

github://kobolog/tesson

Thank you!