Systems (th)at Scale

Jon Crowcroft,

http://www.cl.cam.ac.uk/~jac22

Cloud, Data Center, Networks

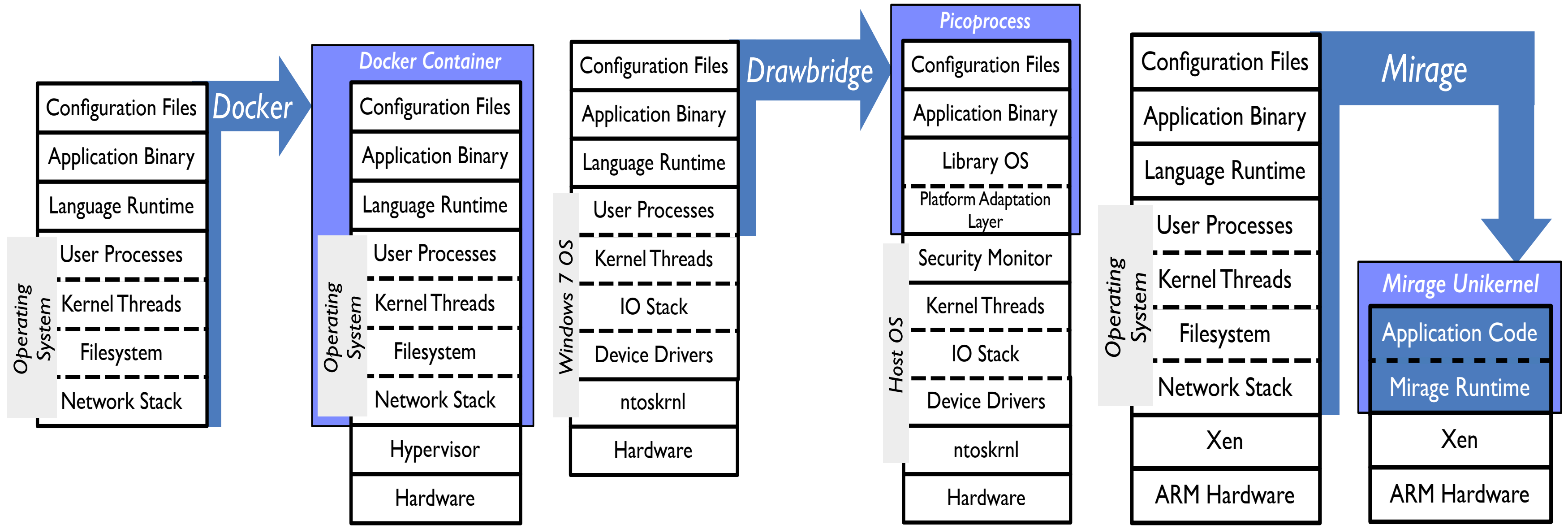
1. New Cloud OS to meet new workloads
   - Includes programming language
   - Collabs incl REMS (w/ P.Gardner/Imperial)
2. New Data Center structure
   - Includes heterogeneous h/w
   - Collabs incl NaaS(Peter Pietzuch Imperial)
   - Trilogy (Mark Handley et al UCL)
3. New Networks (for data centers&)
   - To deal with above☺

What not talking about

- Security
  - (we do that - had another workshop)
- Data
  - Hope Ed folks will!
- Scaling Apps
  - Oxford
- Languages for Apps
  - Ed++

1. Cloud OS

- Unikernels (Mirage, SEL4, ClickOS)



(a) Containers, e.g., Docker. (b) Picoprocesses, e.g., Drawbridge. (c) Unikernels, e.g., MirageOS.

Figure 2: Contrasting approaches to application containment.

Unikernels in OCaml

- But also Go, Scala, Rust etc
- Type safety->security, reliability
- Apps can be legacy or in same languages

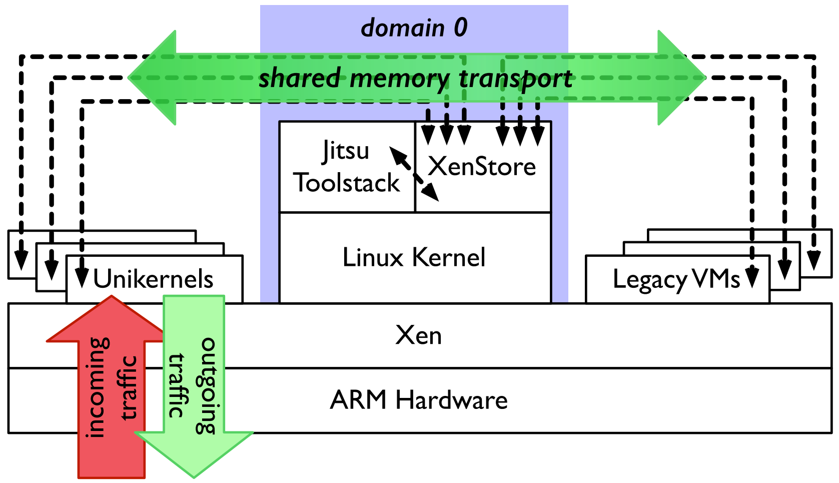


Figure 1: Jitsu architecture: external network connectivity is handled solely by memory-safe unikernels connected to general purpose VMs via shared memory.

Data Centers don't just go fast

- They need to serve applications

1. Latency, not just throughput
2. Face users
    1. Web, video, ultrafast trade/gamers
    2. Face Analytics…
3. Availability & Failure Detectors
4. Application code within network
5. NIC on host or switch – viz

Industry (see pm☺)

Azure

http://conferences.sigcomm.org/sigcomm/2015/pdf/papers/keynote.pdf

Facebook:

http://conferences.sigcomm.org/sigcomm/2015/pdf/papers/p123.pdf

Google:

http://conferences.sigcomm.org/sigcomm/2015/pdf/papers/p183.pdf

2. Deterministic latency bounding

- Learned what I was teaching wrong!
- I used to say:
    - Integrated Service too complex
        - Admission&scheduling hard
    - Priority Queue can't do it
        - PGPS computation for latency?
- I present Qjump scheme, which
    - Uses intserv (PGPS) style admission ctl
    - Uses priority queues for service levels
    - http://www.cl.cam.ac.uk/research/srg/

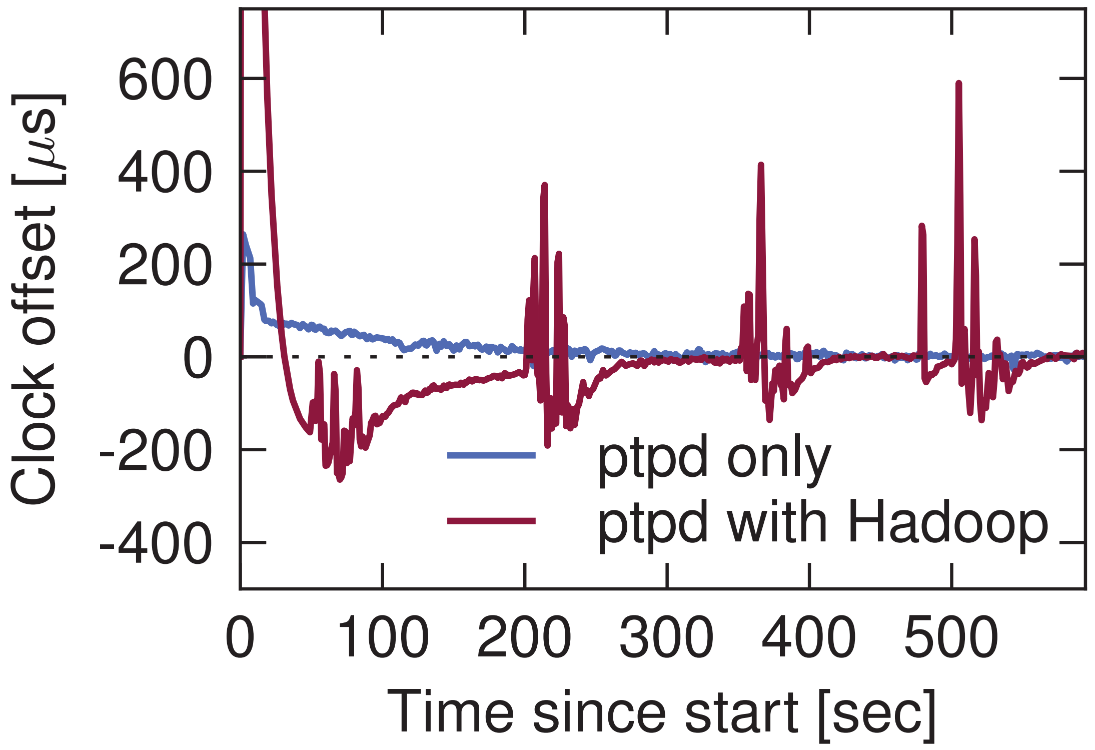
Data Center Latency Problem

- Tail of the distribution,
  - due to long/bursty flows interfering
- Need to separate classes of flow
  - Low latency are usually short flows (or RPCs)
  - Bulk transfers aren't so latency/jitter sensitiv

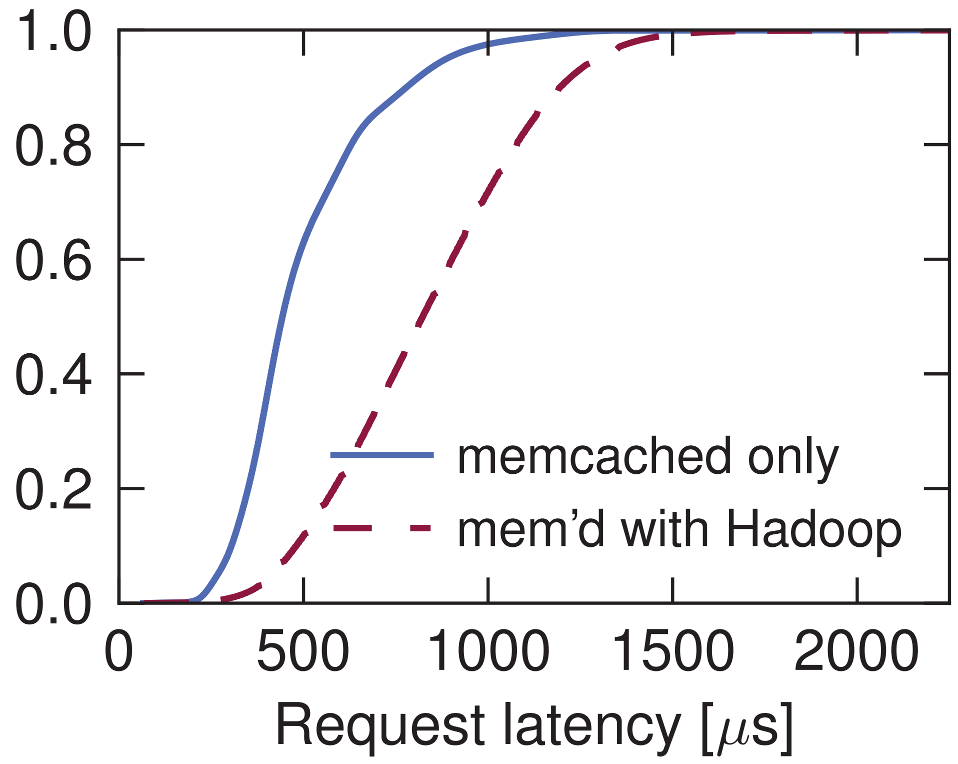
Data Center Qjump Solution

- In Data Center, not general Internet!
  - can exploit topology &
  - traffic matrix &
  - source behaviour knowledge
- Regular, and simpler topology key
- But also largely “cooperative” world…

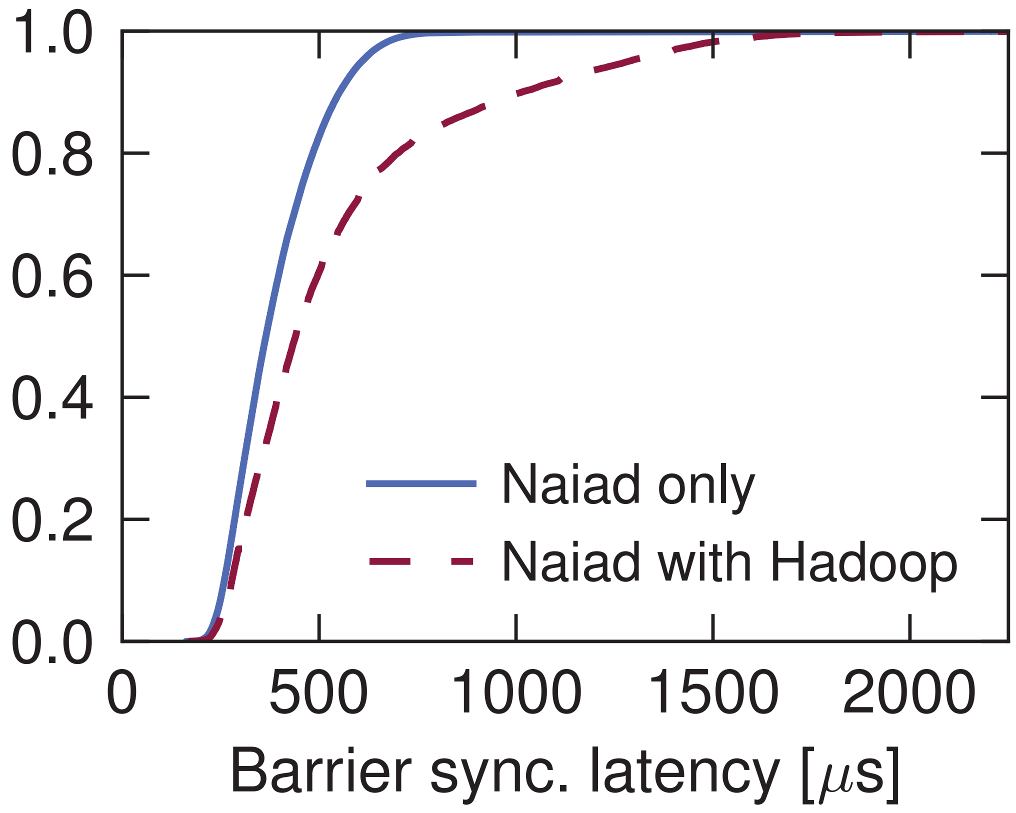
Hadoop perturbs time synch



Hadoop perturbs memcached



Hadoop perturbs Naiad

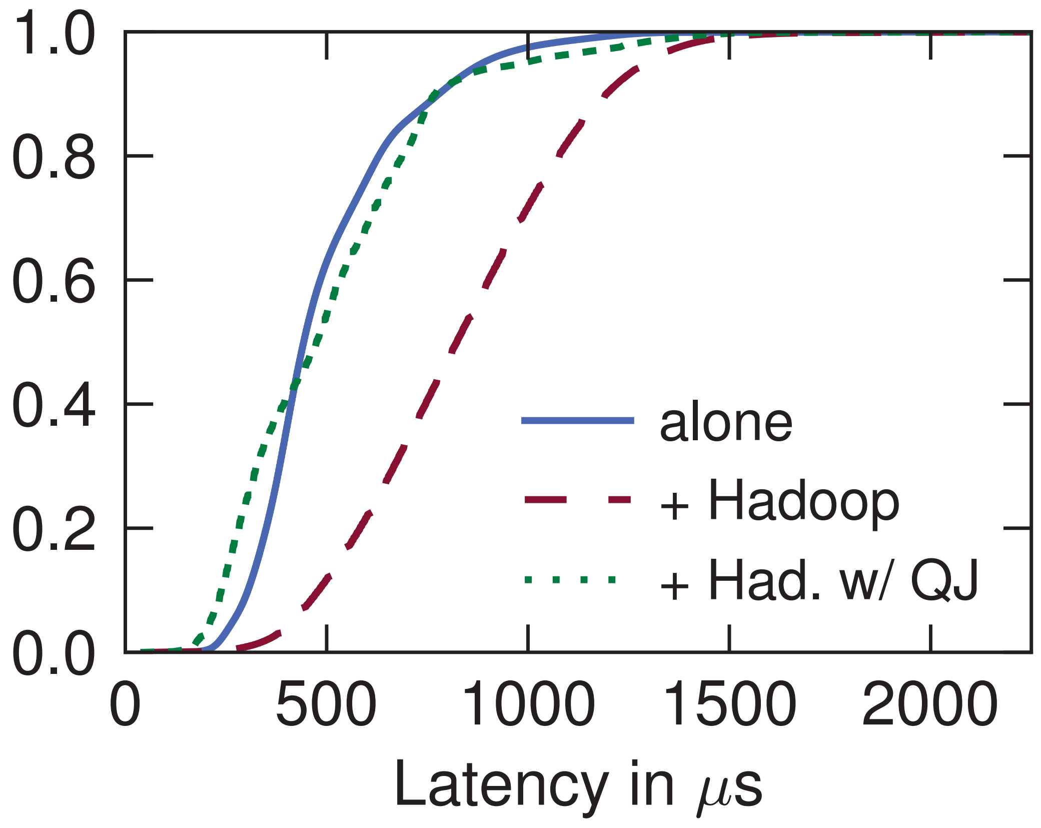


Qjump – two pieces

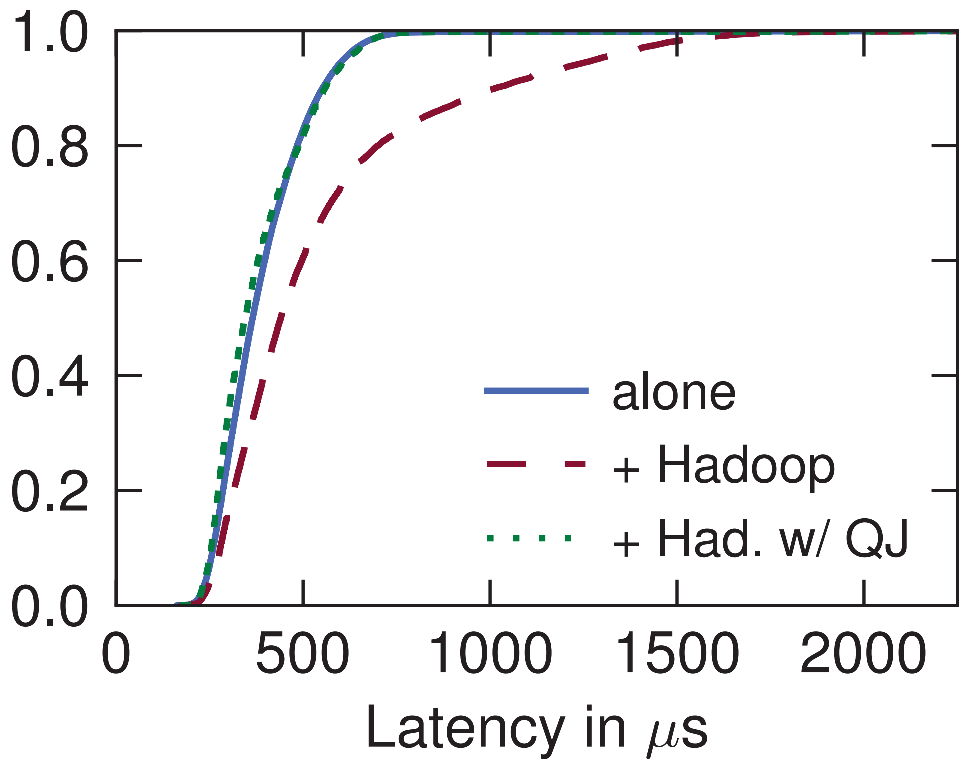
1. At network config time
   - Compute a set of (8*) rates based on
   - Traffic matric & hops => fan in (f)
2. At run time
   - Flow assigns itself a priority/rate class
   - subject it to (per hypervisor) rate limit

* 8 arbitrary – but often h/w supported☺

Memcached latency redux w/ QJ

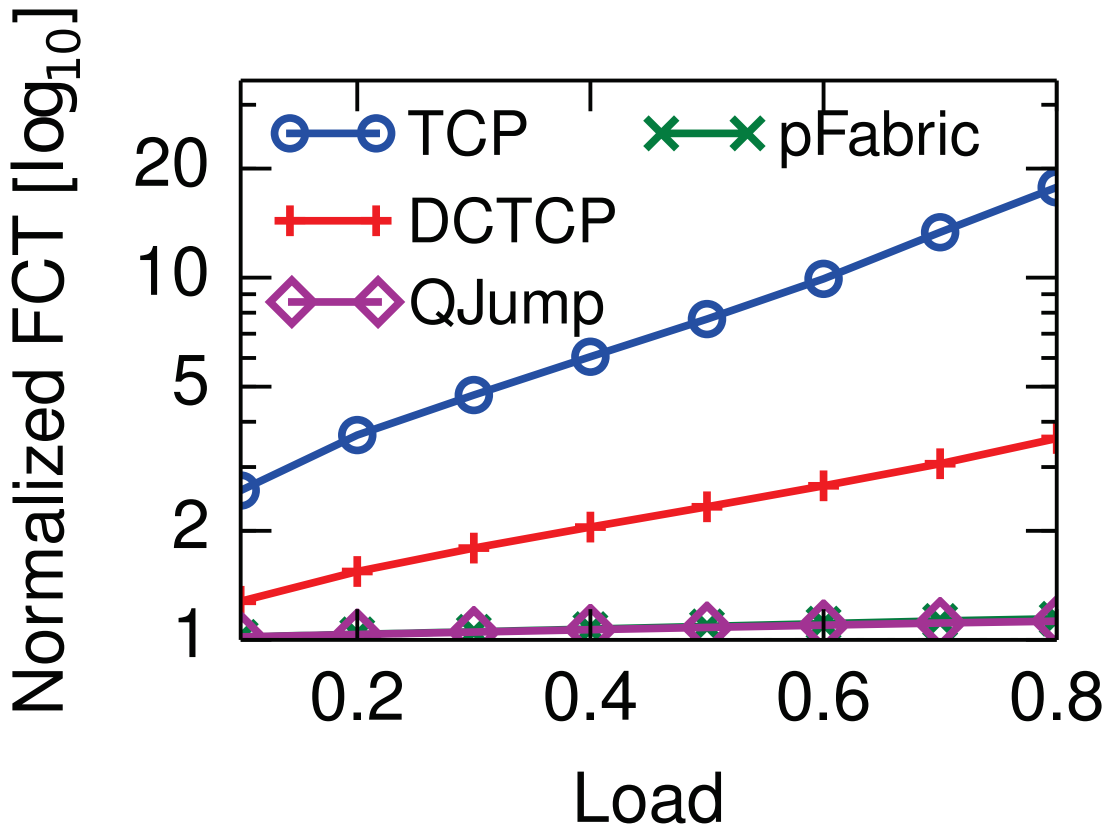


QJ naiad barrier synch latency redux

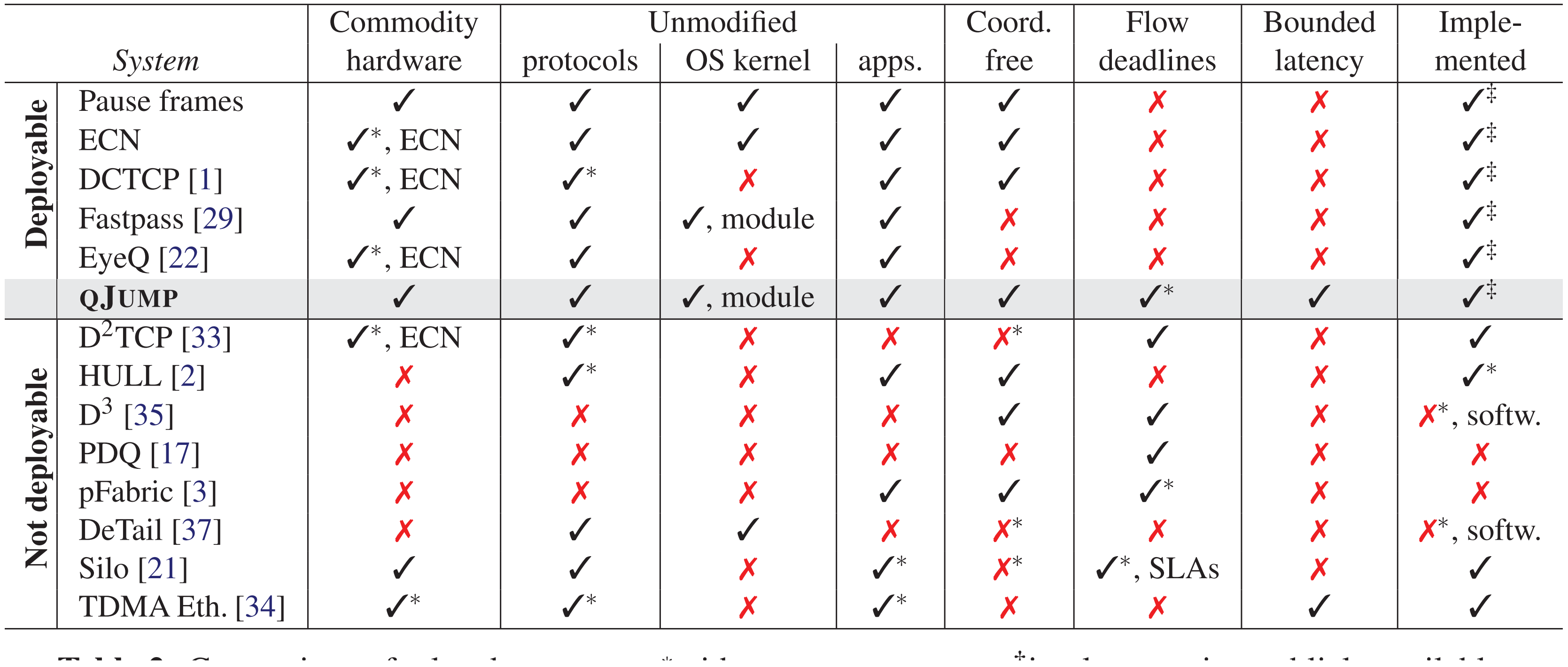


a) QJ fixes Naiad barrier synchronization

Web search FCT100Kb ave



Big Picture Comparison – Related work…

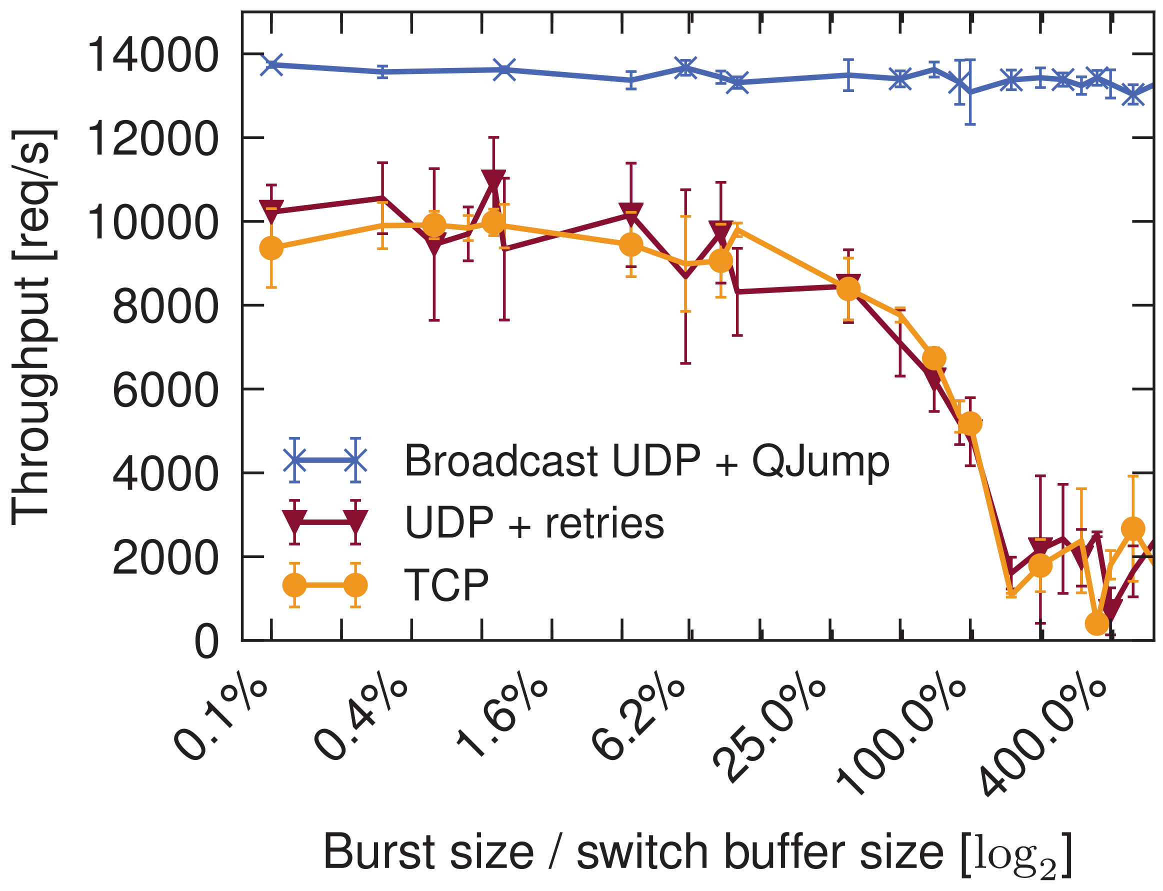


		Commodity	Unmodified			Coord.	Flow	Bounded	Imple-
	System	hardware	protocols	OS kernel	apps.	free	deadlines	latency	mented
Deployable	Pause frames	✓	✓	✓	✓	✓	✗	✗	✓‡
	ECN	✓*, ECN	✓	✓	✓	✓	✗	✗	✓‡
	DCTCP [1]	✓*, ECN	✓*	✗	✓	✓	✗	✗	✓‡
	Fastpass [29]	✓	✓	✓, module	✓	✗	✗	✗	✓‡
	EyeQ [22]	✓*, ECN	✓	✗	✓	✗	✗	✗	✓‡
	QJUMP	✓	✓	✓, module	✓	✓	✓*	✓	✓‡
Not deployable	D^2TCP [33]	✓*, ECN	✓*	✗	✗	✗*	✓	✗	✓
	HULL [2]	✗	✓*	✗	✓	✓	✗	✗	✓*
	D^3 [35]	✗	✗	✗	✗	✓	✓	✗	✗*, softw.
	PDQ [17]	✗	✗	✗	✗	✗	✓	✗	✗
	pFabric [3]	✗	✗	✗	✓	✓	✓*	✗	✗
	DeTail [37]	✗	✓	✓	✗	✗*	✗	✗	✗*, softw.
	Silo [21]	✓	✓	✗	✓*	✗*	✓*, SLAs	✗	✓
	TDMA Eth. [34]	✓*	✓*	✗	✓*	✗	✗	✓	✓

Failure Detectors

- 2PC & CAP theorem
- Recall CAP (Brewer's Hypothesis)
  - Consistency, Availability, Partitions
  - Strong& weak versions!
  - If have net&node deterministic failure detector, isn't necessarily so!
- What can we use CAP-able system for?

2b 2PC throughput with and without QJump



Consistent, partition tolerant app?

- Software Defined Net update!
  - Distributed controllers have distributed rules
  - Rules change from time to time
  - Need to update, consistently
  - Need update to work in presence of partitions
    - By definition!
  - So Qjump may let us do this too!

3. Application code -> Network

- Last piece of data center working for application
- Switch and Host NICs have a lot of smarts
  - Network processors,
  - like GPUs or (net)FPGAs
  - Can they help applications?
  - In particular, avoid pathological traffic patterns (e.g. TCP incast)

Application code

- E.g. shuffle phase in map/reduce
  - Does a bunch of aggregation
  - (min, max, ave) on a row of results
  - And is cause of traffic "implosion"
  - So do work in stages in the switches in the net (like merge sort!)
- Code very simple
- Cross-compile into switch NIC cpus

Other application examples

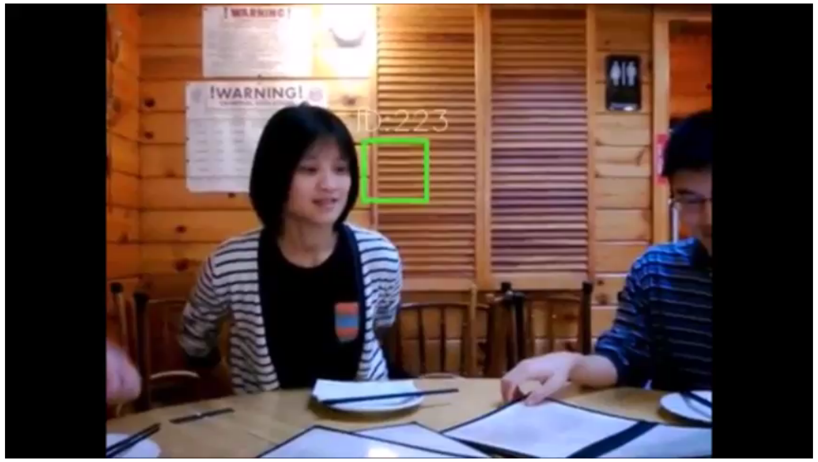
- Are many …
- Arose in Active Network research
  - Transcoding
  - Encryption
  - Compression
  - Index/Search
- Etc etc

Need language to express these

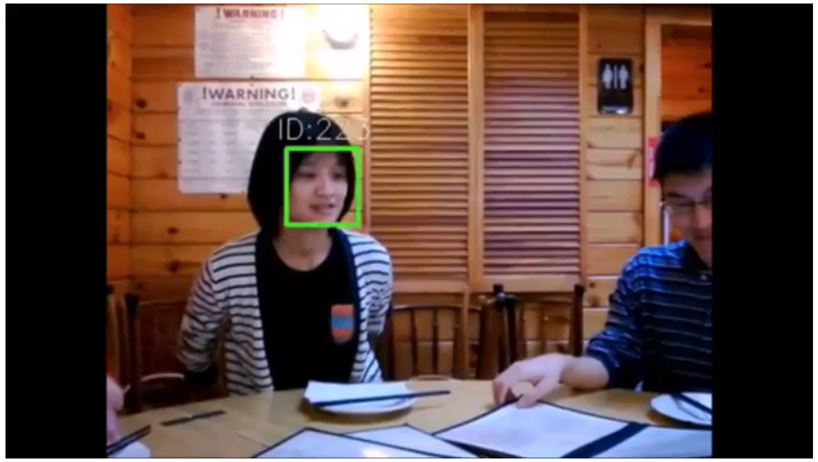
- Finite iteration
- (not Turing-complete language)
- So design python- with strong types!
- Work in progress in NaaS project at Imperial and Cambridge…

Cloud Computing Isn't For Everything!

Latency effect on facial recognition Source: Glimpse project, MIT, 2014



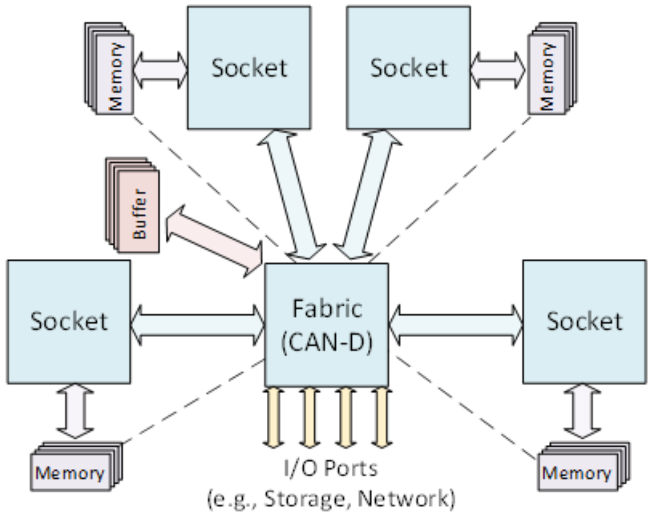
Remote Processing

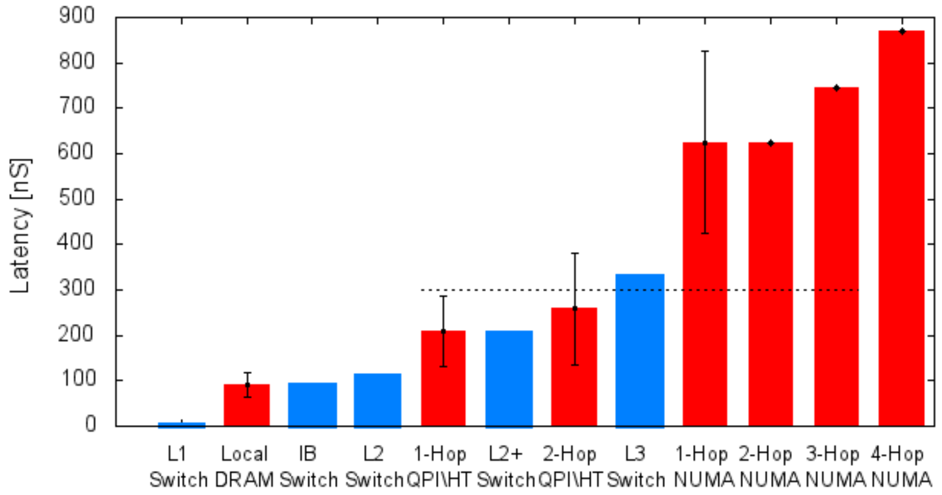


Local Processing

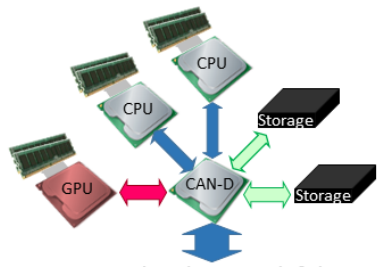
- "being fast really matters…half a second delay caused a 20% drop in traffic and it killed user satisfaction" - Marissa Mayer @ Web 2.0 (2008)
- "A millisecond decrease in a trade delay may boost a high-speed firm's earnings by about 100 million per year" – SAP, 2012
- "It's simply not appropriate to just drag and drop our databases into a cloud platform" – Thomas Kadlec, Tesco, 2015

Tiny Terabit Datacentre
An End-Host Networked-Server Architecture

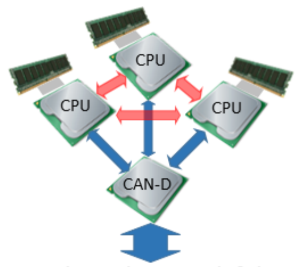




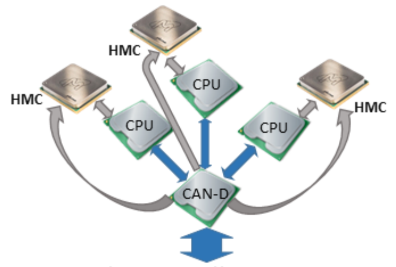
- ✓ **High Performance**
- ✓ **Resource Isolation**
- ✓ **Flexible Implementation**
- ✓ **Predictable Latency**
- ✓ **Low Latency Interconnect**
- ✓ **Affordable**



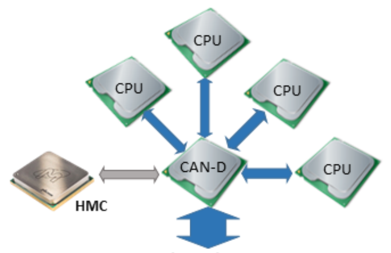
Centralized network fabric



Distributed network fabric

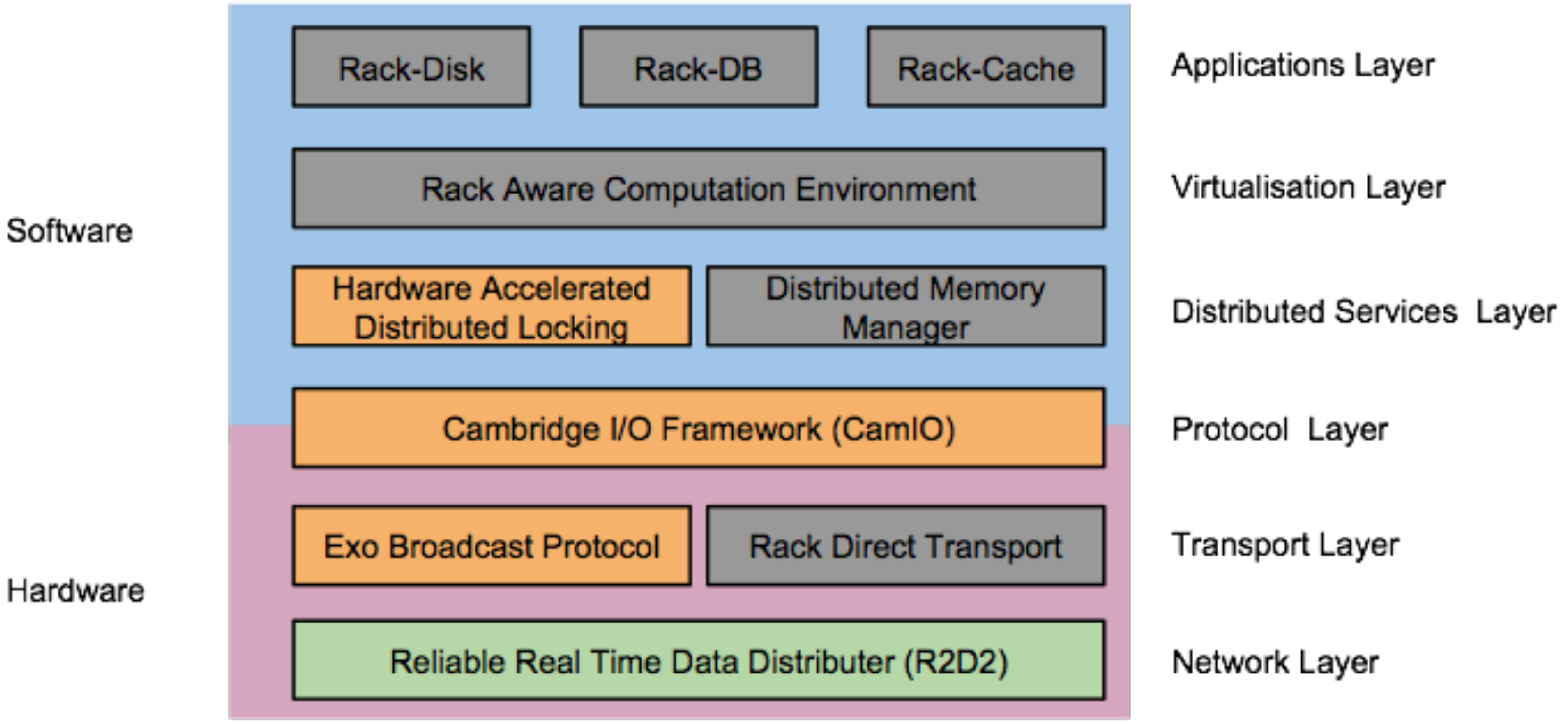


Multi-controller memory



Centralized memory

Networks, Interfaces and Transports for Rack-Scale Operating Systems


Rack-Disk
Rack-DB
Rack-Cache
Applications Layer
Rack Aware Computation Environment
Virtualisation Layer
Software
Hardware Accelerated Distributed Locking
Distributed Memory Manager
Distributed Services Layer
Cambridge I/O Framework (CamIO)
Protocol Layer
Exo Broadcast Protocol
Rack Direct Transport
Transport Layer
Hardware
Reliable Real Time Data Distributer (R2D2)
Network Layer

🟩 Operational 🟧 First Prototype ⬜ Design stage

Conclusions/Discussion

- Data Center is a special case!
- Its important enough to tackle
  - We can hard bound latency easily
  - We can detect failures and therefore solve some nice distributed consensus problems
  - We can optimise applications pathological traffic patterns
  - Integrate programming of net&hosts
  - Weird new h/w…
- Plenty more to do…