Fastpass

2014

Jonathan Perry, Amy Ousterhout, Hari Balakrishnan, Devavrat Shah, Hans Fugal
M.I.T Computer Science & Artificial Intelligence Lab Facebook

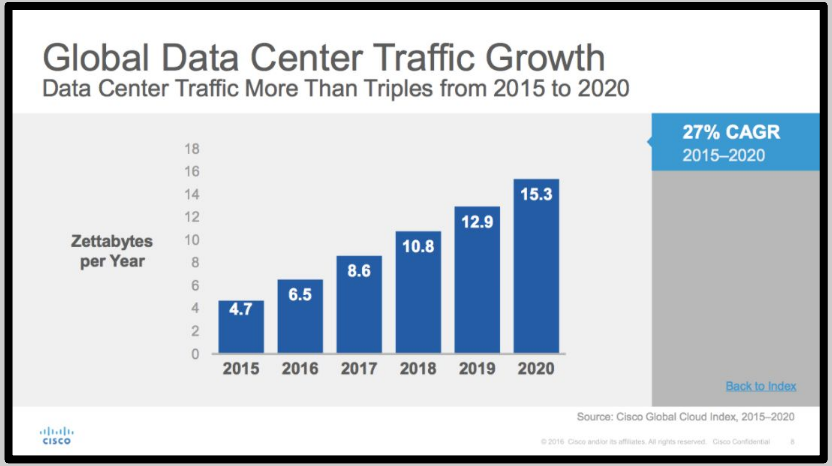
http://fastpass.mit.edu/

Presented By: Norman Ponte (nponte)

Introduction

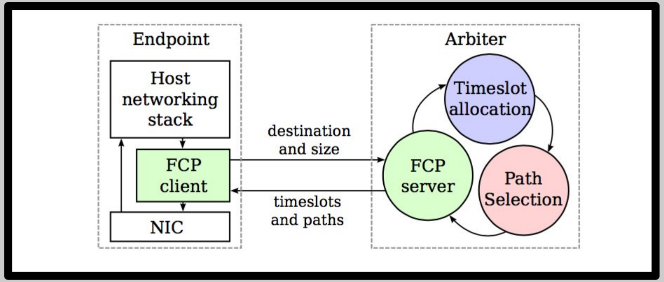
Data Center Scheduling: Why is it Important?

- Data Centers place content near their consumers
- Current Data Center Design inherit the principles from original Internet Design
- Packets spend a lot of time in big memory intensive queues



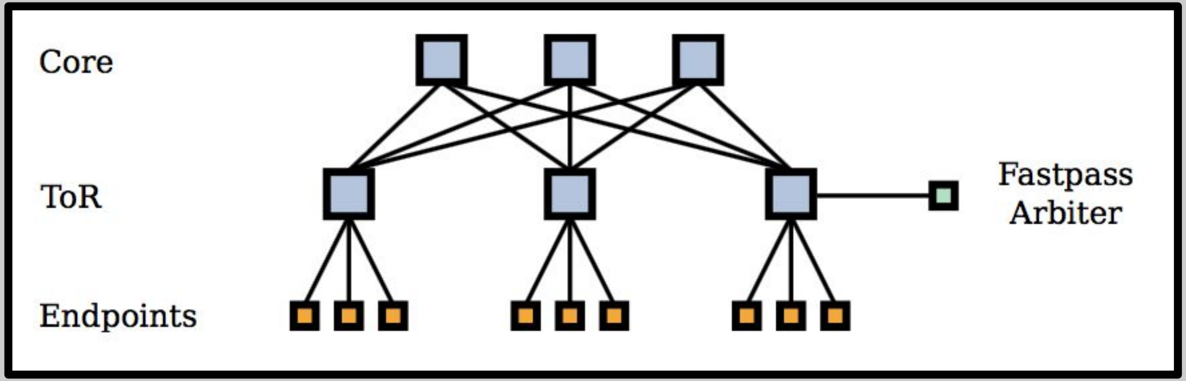
Main Ideas

- **Ideal**: Low median and tail latency, high throughput, fair resource allocation, deadline awareness, congestion avoidance
- Current Centers address these needs but not effectively
- **Goals**: No queuing Delays, High Utilization, Multiple Resource Objectives between flows applications users
- Use of Arbiter to control each packets timing
- Centralized control at granularity of individual packets



Key Insights

- A centralized arbiter can be implemented and work
- Multicore Arbiter
- Arbiter can do a better job at resource allocation for workloads with different performance objectives



Related Work

- Using a *Centralized Controller* but don't provide control over packet latencies or allocations
- **Hedera/TDMA/Mordia** - optimization for elephant flows
- **Orchestra** - Application-level coordinating transfers
- **SWAN** - reconfigure the data plane to match demand, Forwarding Tables
- *Distributed Approaches* set to solve data center problems
- **DCTCP/HULL** - reduce switch queuing, do not eliminate queuing delay
- **MATE/DARD** - reroute traffic selfishly until converging to load balanced solution

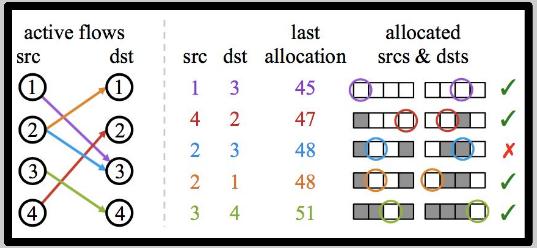
Architecture

Three Key Components

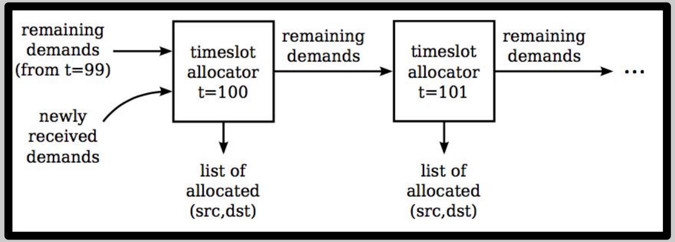
- **Timeslot Allocation Algorithm**
- **Path Assignment Algorithm**
- **Replication Strategy for the Central Arbiter**

Timeslot Allocation

- Choose a matching of endpoints in each timeslot
- Rearrangeably non blocking (RNB) tiers: Any traffic that satisfies the input and output bandwidth constraints
- Allows them to separate timeslot allocation from path selection
- This needs to be fast: greedily allocates a source-destination pair if it doesn't violate bandwidth constraints
- Pipelined timeslot allocation



Max-min fairness



min-FCT

Path Selection

- Assign packets with timeslots to paths through the network that avoiding queuing
- Balance packet load across all available links
- Timeslot allocation guarantees that we can do this
- Path between two ToR can be uniquely specified by a core switch
- Assign a core switch to each packet such that no two packets with the same source ToR or destination ToR assigned same core switch
- **Vertices**:ToR switches, **Edges**: Packets, **Colors**: Core Switch
- Fast Edge-Coloring using Euler-split O(n d log(d)) time : **n** racks **d** nodes

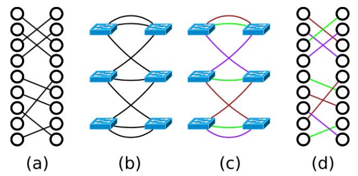


Figure 5: Path selection. (a) input matching (b) ToR graph (c) edge-colored ToR graph (d) edge-colored matching.

Handling Faults

- 3-types:
- Failure of Arbiter
  - Fastpass runs multiple arbiters
  - Backup arbiters receive all requests so no need to share information on failure
- Failure of In-Network Components
  - Usage of package drops to detect network faults
- Packet loss on communication: Endpoints -> Arbiter
  - Fastpass Control Protocol (FCP)
  - Endpoints and Arbiter have an aggregate count of time slots requested non-matching values implies loss in communication

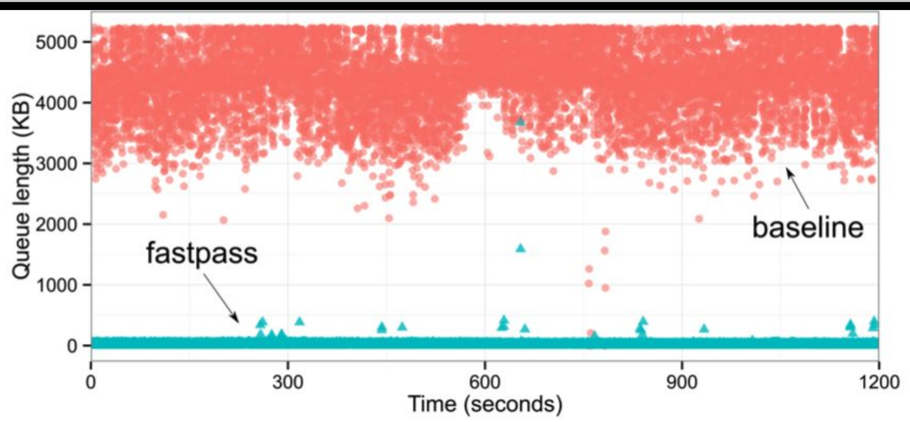
Results

Slightly Less Throughput and Much Less Queueing

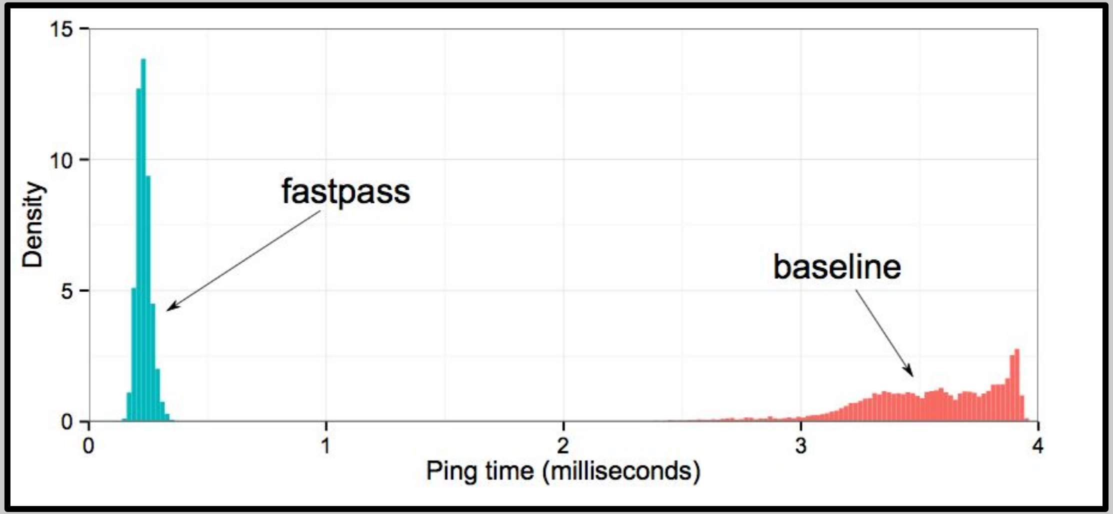
Fastpass	9.28 Gbits/s
Baseline	9.43 Gbits/s

Due to FCP (Fastpass Control Protocol) use of traffic

	Median	90th %ile	99th	99.9th
Baseline (Kbytes)	4351	5097	5224	5239
Fastpass (Kbytes)	18	36	53	305

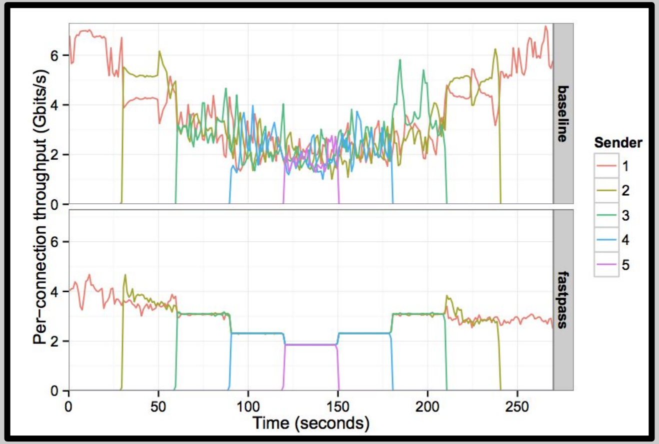


High Load Latency Improvement 15.5x



	Median	90^{th} %ile	99^{th}	99.9^{th}
Baseline (ms)	3.56	3.89	3.92	3.95
Fastpass (ms)	0.23	0.27	0.32	0.38

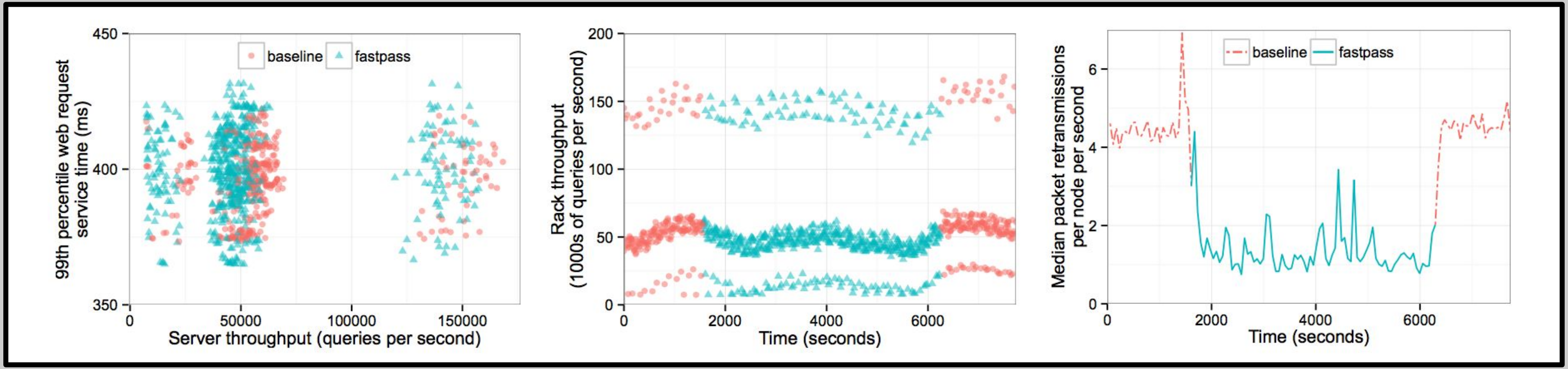
5200x standard deviation of throughput: Fairness



#connections	Baseline	Fastpass	Improvement
3	543.86	15.89	34.2×
4	628.49	0.146	4304.7×
5	459.75	0.087	5284.5×

Facebook Deployment

- Able to test their algorithm on Facebook data centers
- Almost no benefit except 2x lowering of TCP retransmits
- Latency-sensitive service - response path for use web requests
-



Going Forward

Real world Value & Evaluation

- The authors admit that scalability is a concern
- Arbiter would have to handle large volume of traffic : Custom Hardware?
- Facebook Concerns
- "Zero Queue" - movement of queues to the arbiter
- Open door for high-performance, tightly-integrated, predictable networks

Questions?