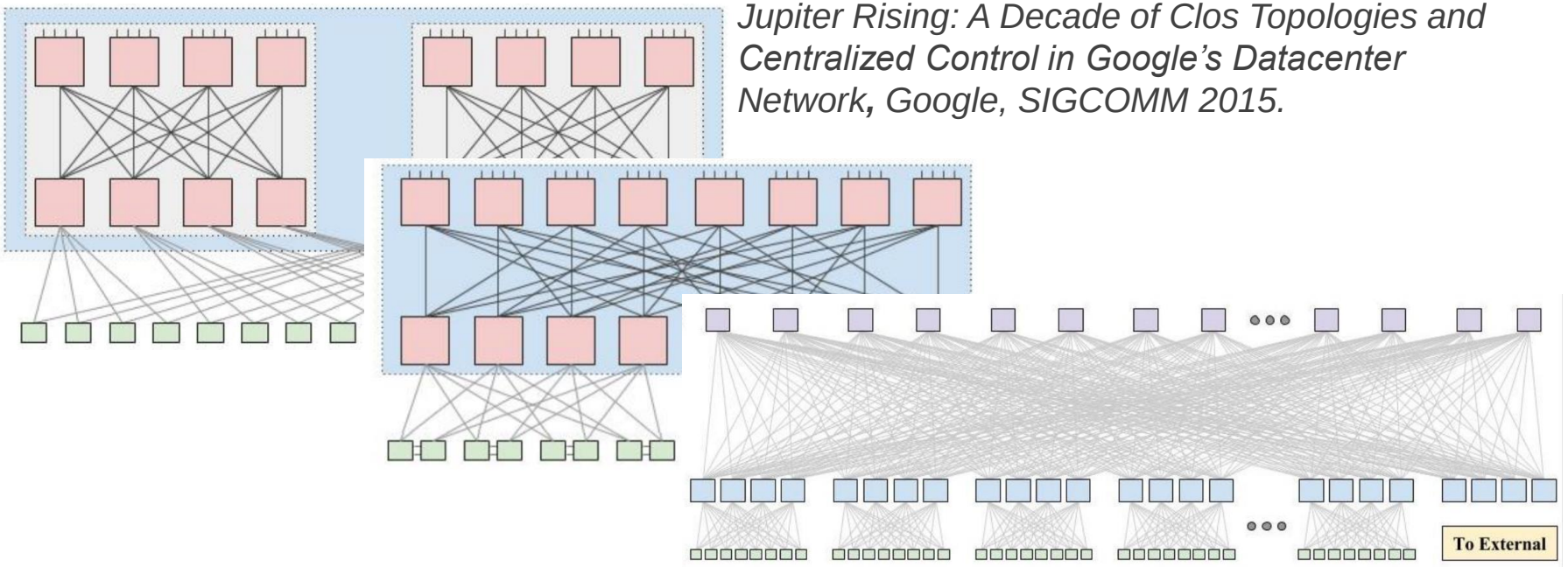
Micro load balancing in data centers with DRILL

Soudeh Ghorbani (UIUC)
Brighten Godfrey (UIUC)
Yashar Ganjali (University of Toronto)
Amin Firoozshahian (Intel)

Where should the load balancing functionality live?

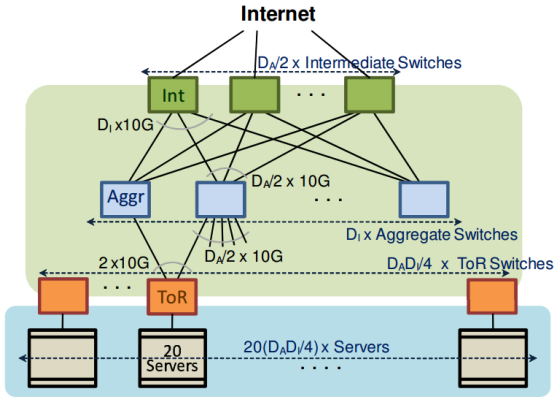
Why *load balancing* in data centers?

Data center apps have demanding network requirements.

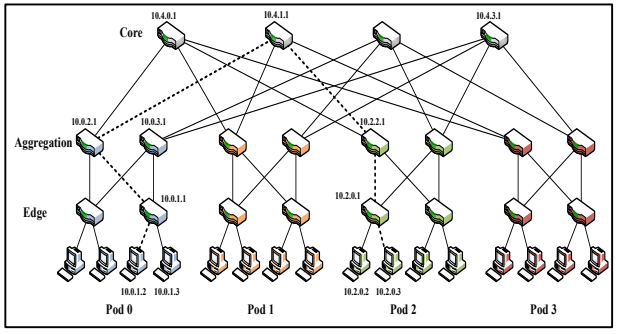


Jupiter Rising: A Decade of Clos Topologies and Centralized Control in Google's Datacenter Network, Google, SIGCOMM 2015.

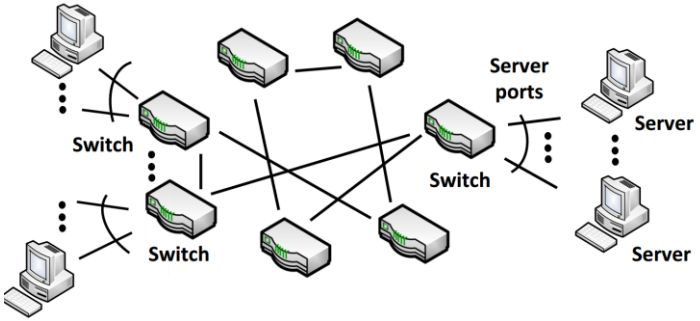
Data center topologies provide high capacity.



VL2: a scalable and flexible data center network, C. Kim et al., SIGCOMM 2009



A scalable, commodity data center network architecture, M. Al-Fares et al., SIGCOMM 2008



Jellyfish: Networking Data Centers Randomly., A. Singla et al., NSDI 2012

But we are still not using the capacity efficiently!

Networks experience high congestion drops as utilization approached 25%[1].

Further improving fabric congestion response remains an ongoing effort[1].

[1] Jupiter Rising: A Decade of Clos Topologies and Centralized Control in Google's Datacenter Network, Google, SIGCOMM 2015.

The gap:

High bandwidth provided via massive *multipathing*.

Balancing load among many paths in real time seems too hard for our "*fast and dumb*" data center fabric.

Congestion happens even when there is spare capacity to mitigate it elsewhere[2].

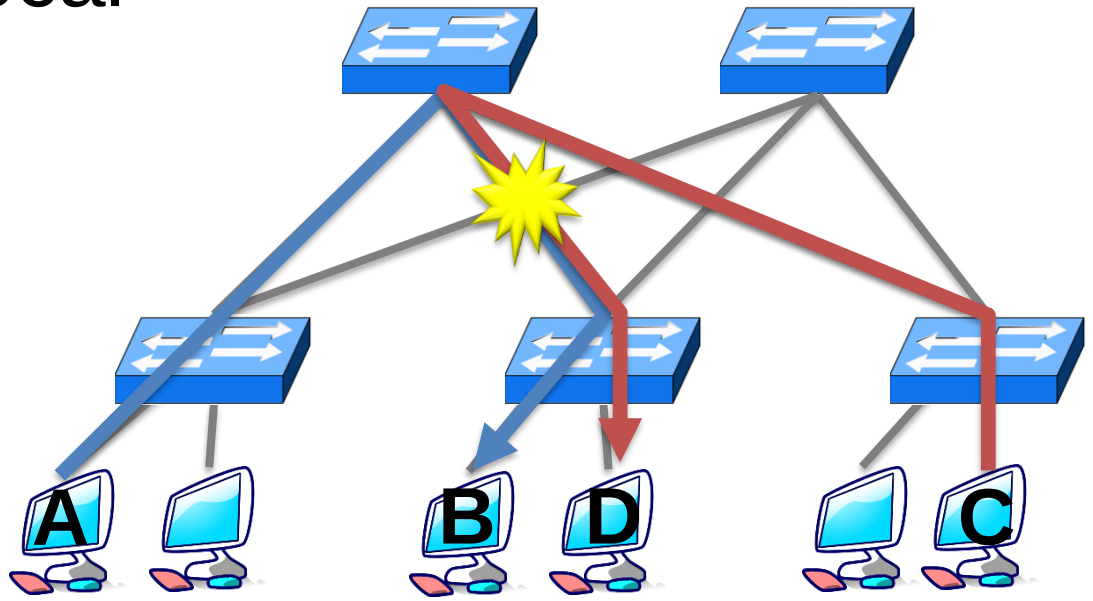
[2] Network Traffic Characteristics of Data Centers in the Wild, T. Benson et al., IMC 2010.

ECMP is not the answer.

Select among equal-cost paths by a hash of 5-tuple

Problems:

- Coarse grained
- Stateless and local



Rethinking the problem:

- Hedera *[NSDI'10]*
- Mahout *[INFOCOM'11]*
- FastPass *[SIGCOMM'14]*
- Plank *[SIGCOMM'14]*
- Presto *[SIGCOMM'15]*
- MPTCP *[NSDI'11]*
- CONGA *[SIGCOMM'14]*
- ...

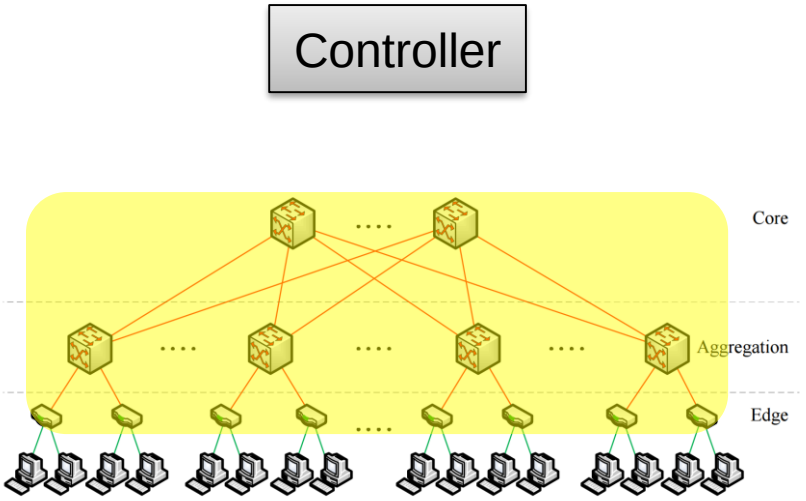
Rethinking the problem

- Let’s move the load balancing functionality out of the core!

Moving LB from fabric to:

Central Controller

- **Hedera** ***[NSDI'10]***
- **Mahout** ***[INFOCOM'11]***
- **FastPass** ***[SIGCOMM'14]***
- **Planck** ***[SIGCOMM'14]***
- Presto *[SIGCOMM'15]*
- MPTCP *[NSDI'11]*
- CONGA *[SIGCOMM'14]*
- ...

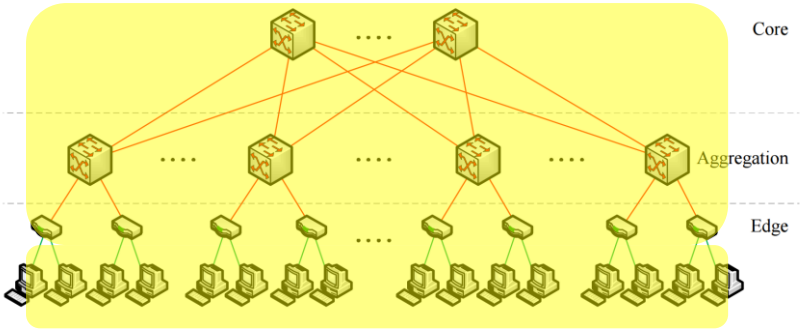


Moving LB from fabric to:

- Hedera [NSDI'10]
- Mahout [INFOCOM'11]
- FastPass [SIGCOMM'14]
- Planck [SIGCOMM'14]
- **Presto** ***[SIGCOMM'15]***
- **MPTCP** ***[NSDI'11]***
- CONGA *[SIGCOMM'14]*
- …

Hosts

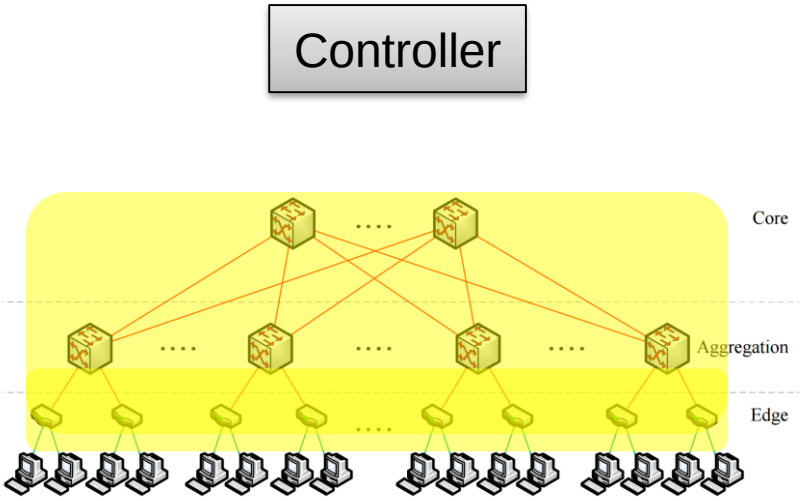
Controller



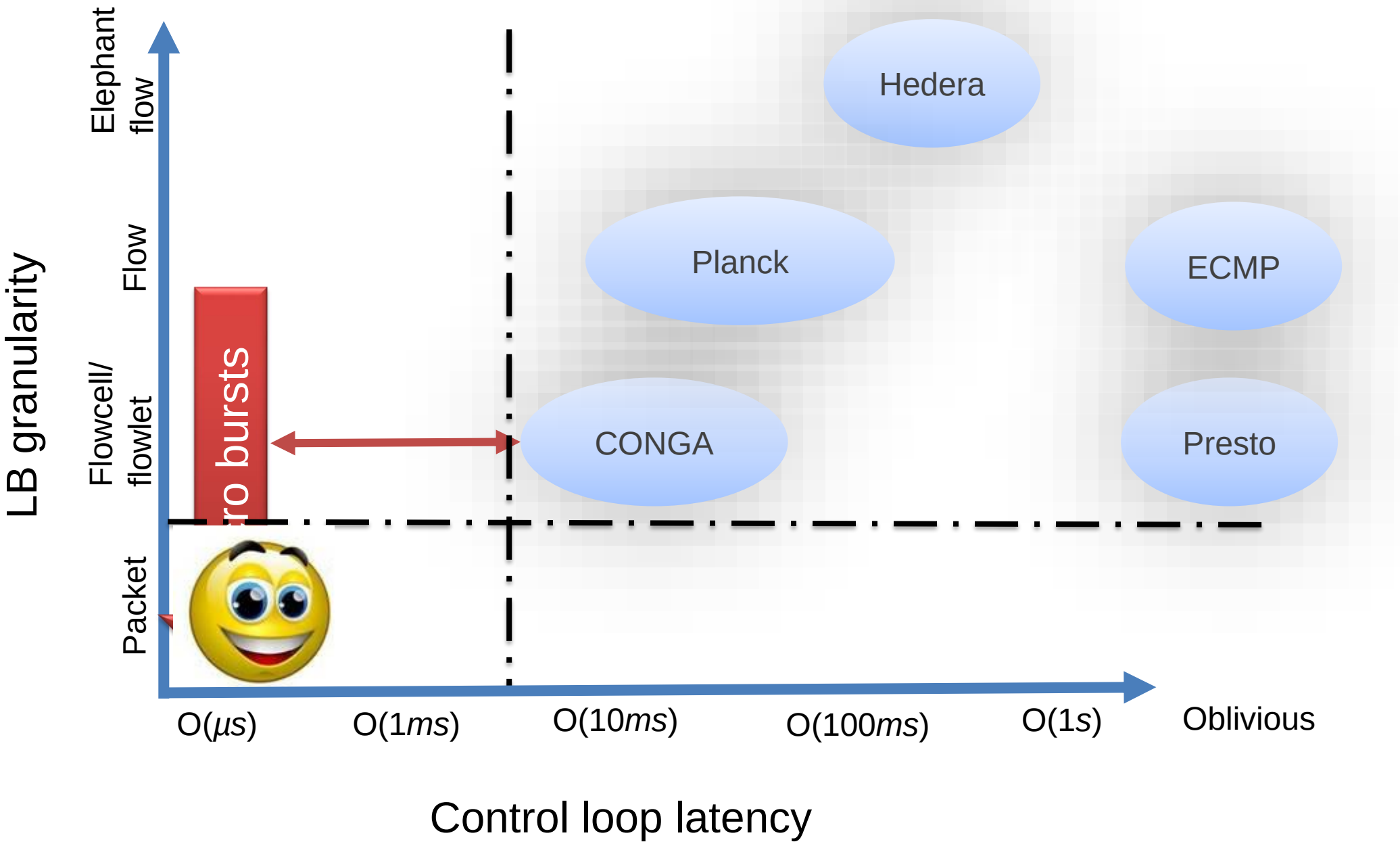
Moving LB from fabric to:

- Hedera *[NSDI'10]*
- Mahout *[INFOCOM'11]*
- FastPass *[SIGCOMM'14]*
- Planck *[SIGCOMM'14]*
- Presto *[SIGCOMM'15]*
- MPTCP *[NSDI'11]*
- **CONGA *[SIGCOMM'14]***
- …

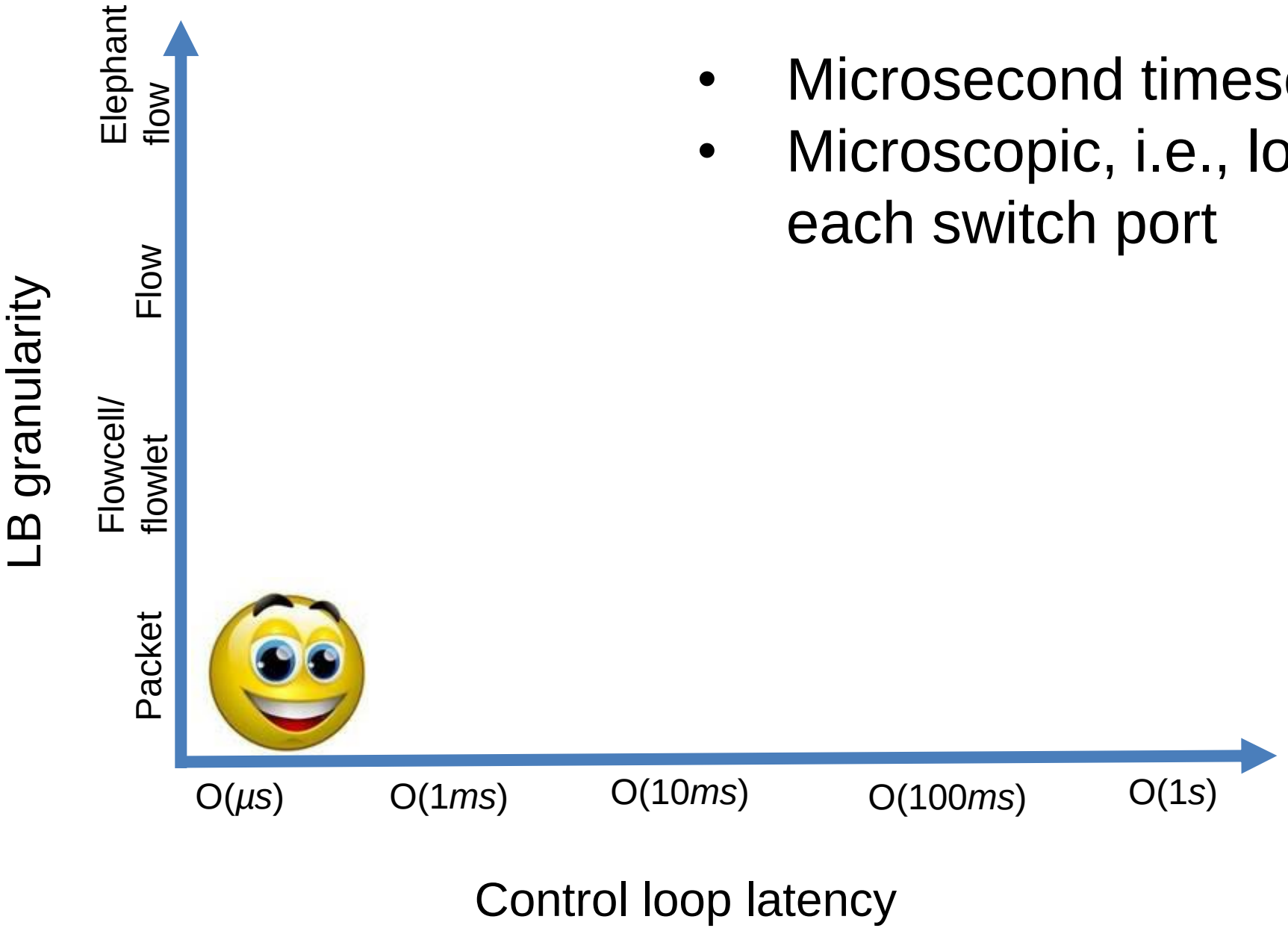
Edge



Scalable LB design space



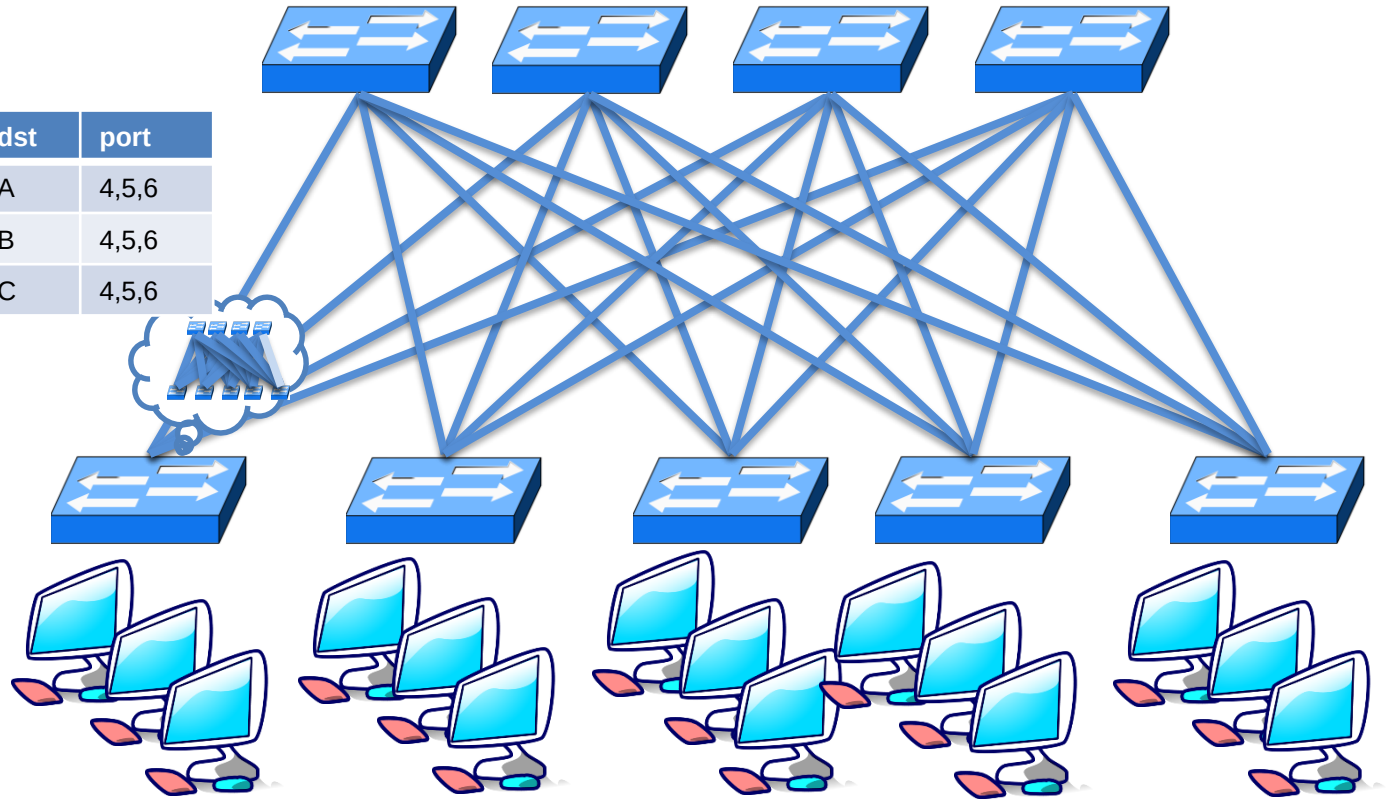
“Micro load balancing”



- Microsecond timescales
- Microscopic, i.e., local to each switch port

Micro LB – A plausible architecture
Symmetric topologies

dst	port
A	4,5,6
B	4,5,6
C	4,5,6



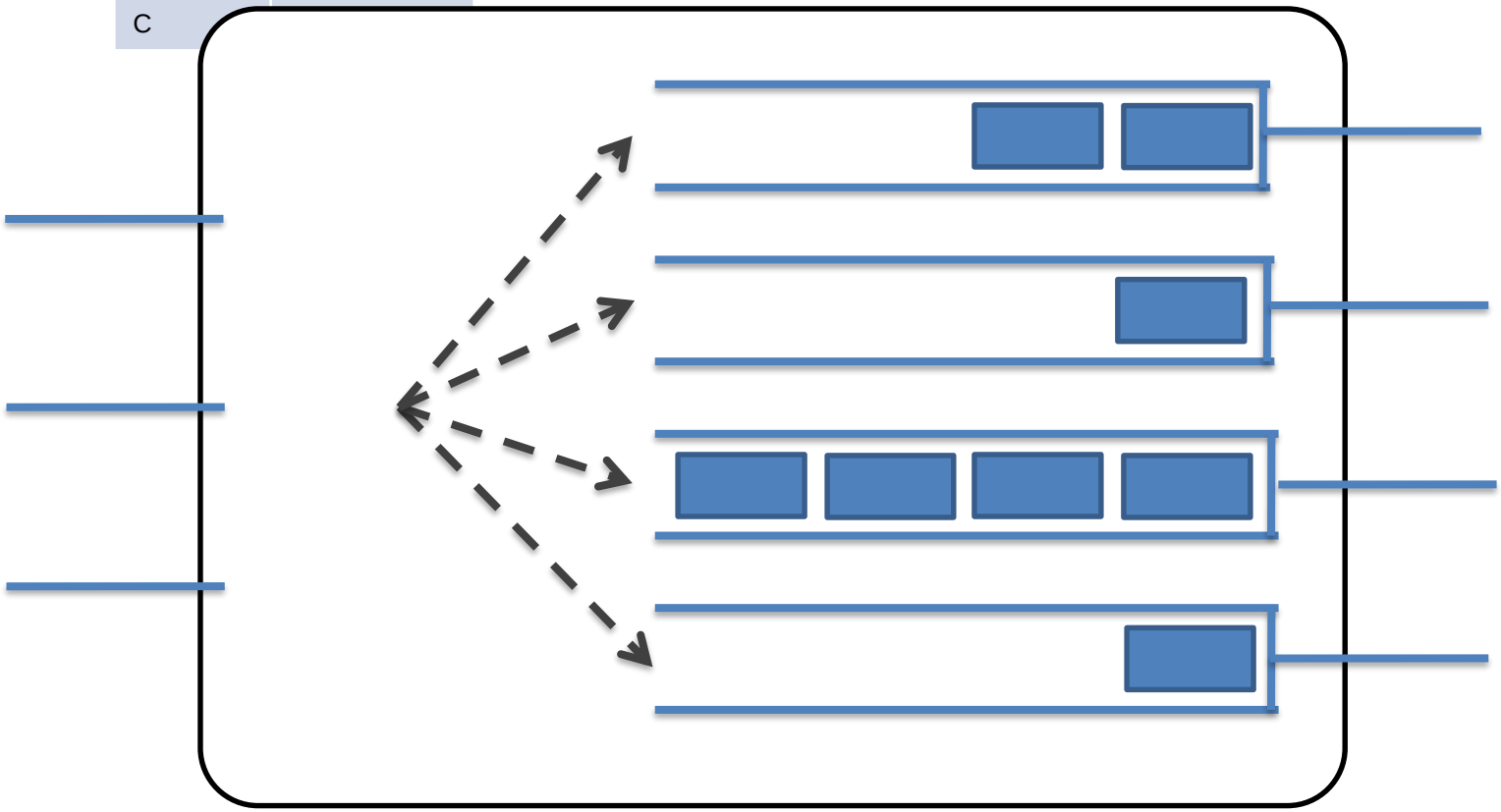
Fabric

1. Discover topology.
2. Compute multiple shortest paths.
3. Install into FIB

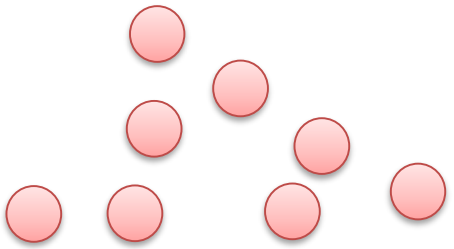
~ ECMP, so far...

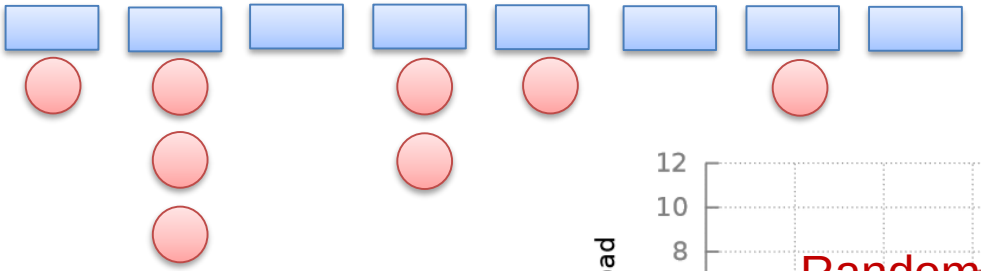
Inside a switch…

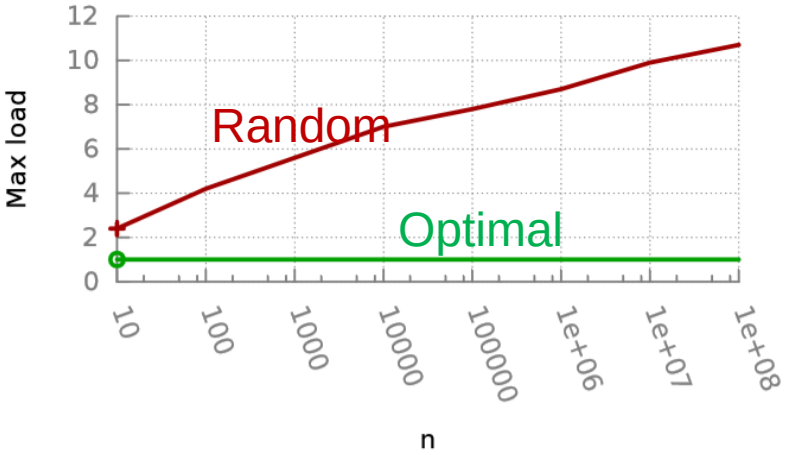
dst	port
A	4,5,6,7
B	4,5,6,7
C	



The power of 2 choices







- *n* bins and *n* balls
- Each ball choosing a bin *independently* and uniformly at *random*
- Max load:

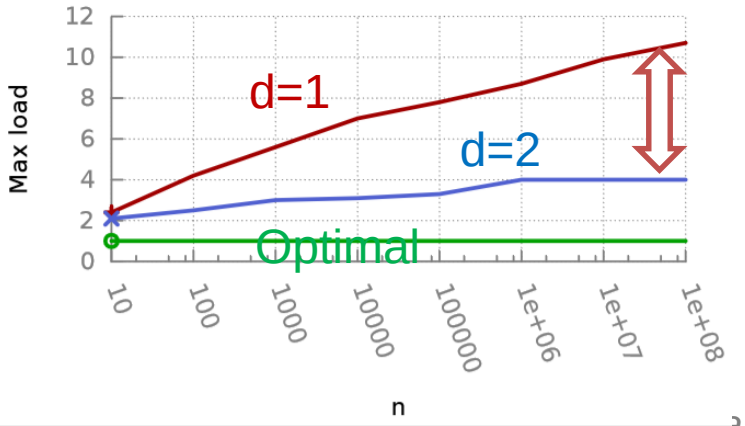
$$\theta(\log n / \log \log n)$$

Y. Azar, A. Z. Broder, A. R. Karlin, and E. Upfal. Balanced allocations. SIAM Journal on Computing, 1994.

The power of 2 choices

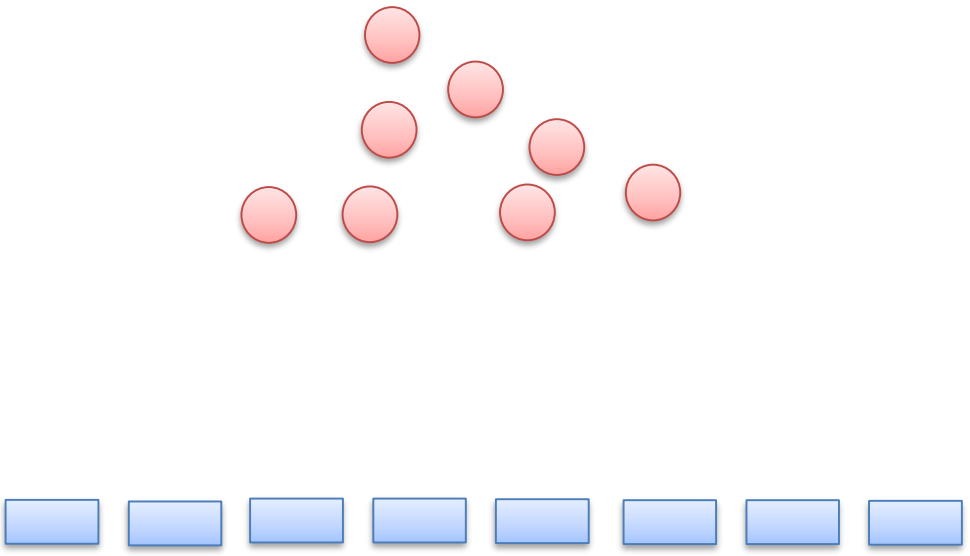
- n bins and n balls
- Balls placed *sequentially*,
- in the *least loaded of* $d \geq 2$ *random bins*
- Max load:

$$\mathcal{O}\left(\log \log n / \log d\right)$$



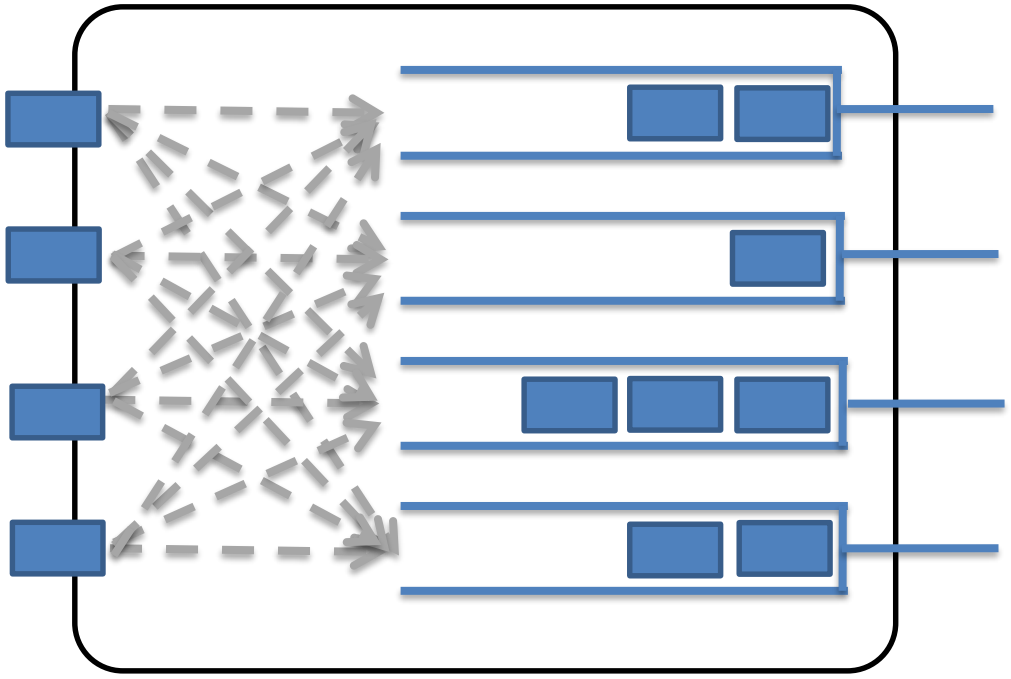
Y. Azar, A. Z. Broder, A. R. Karlin, and E. Upfal. Balanced allocations. SIAM Journal on Computing, 1994.

What we want:



- *Queues* instead of bins.
- Each ball chooses a bin *independently, no coordination.*

What we want:



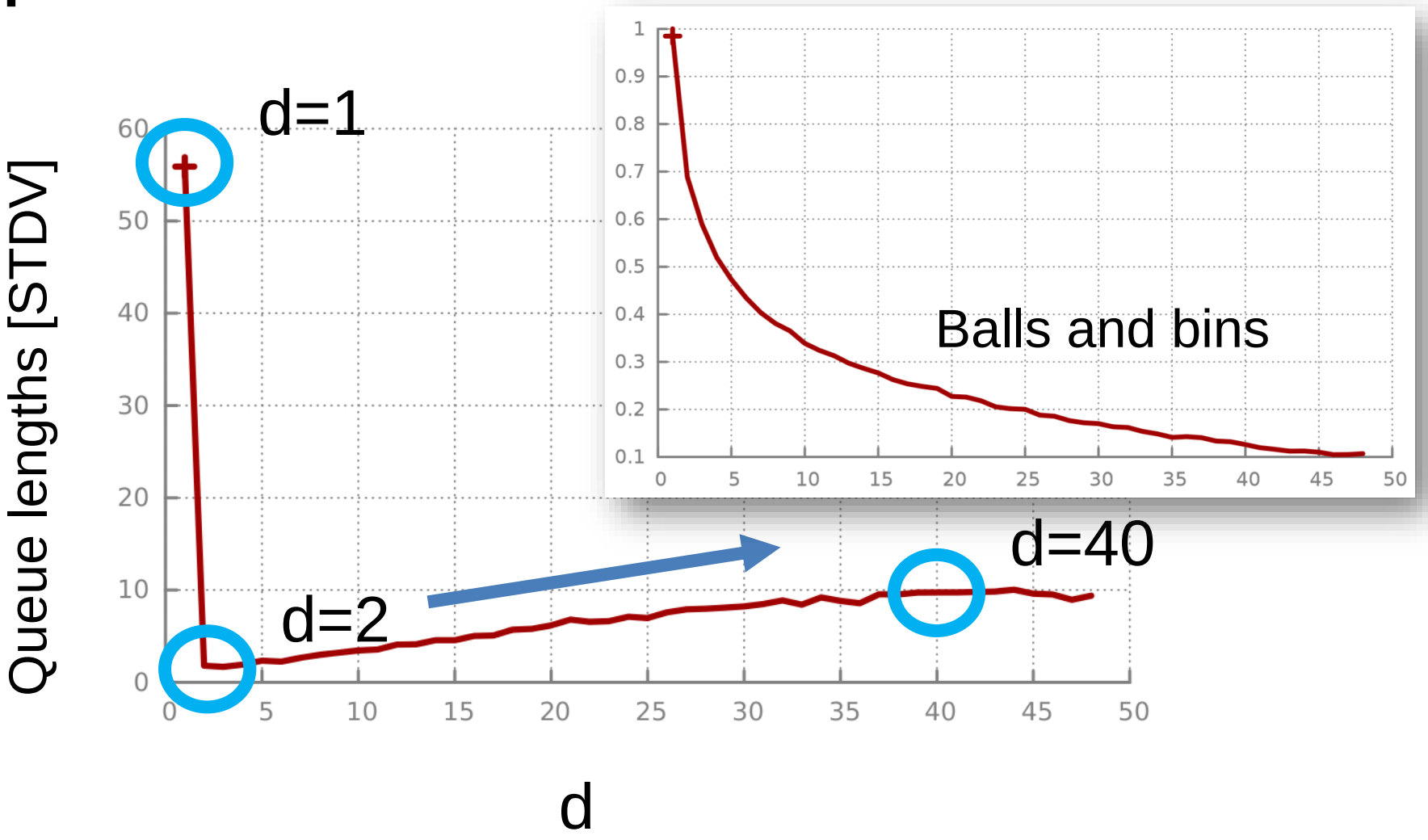
dropped

- *Queues* instead of bins.
- Each ball chooses a bin *independently, no coordination.*

Simulation methodology

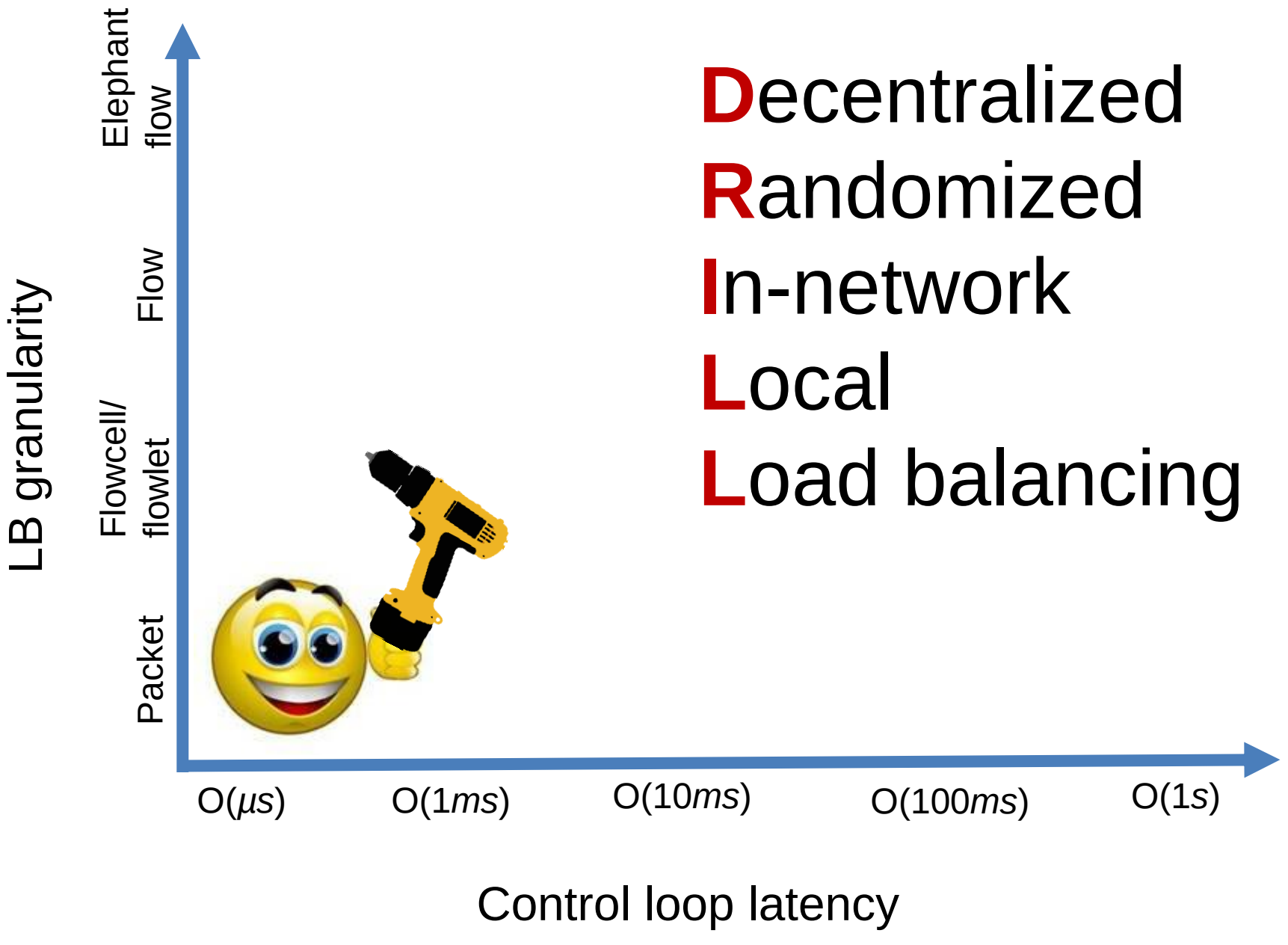
- OMNET++, INET framework
- Linux 2.6 TCP
- Leaf/spine topologies
- Datacenter traces from DevoFlow *[SIGCOMM'11]*

The pitfalls of choice



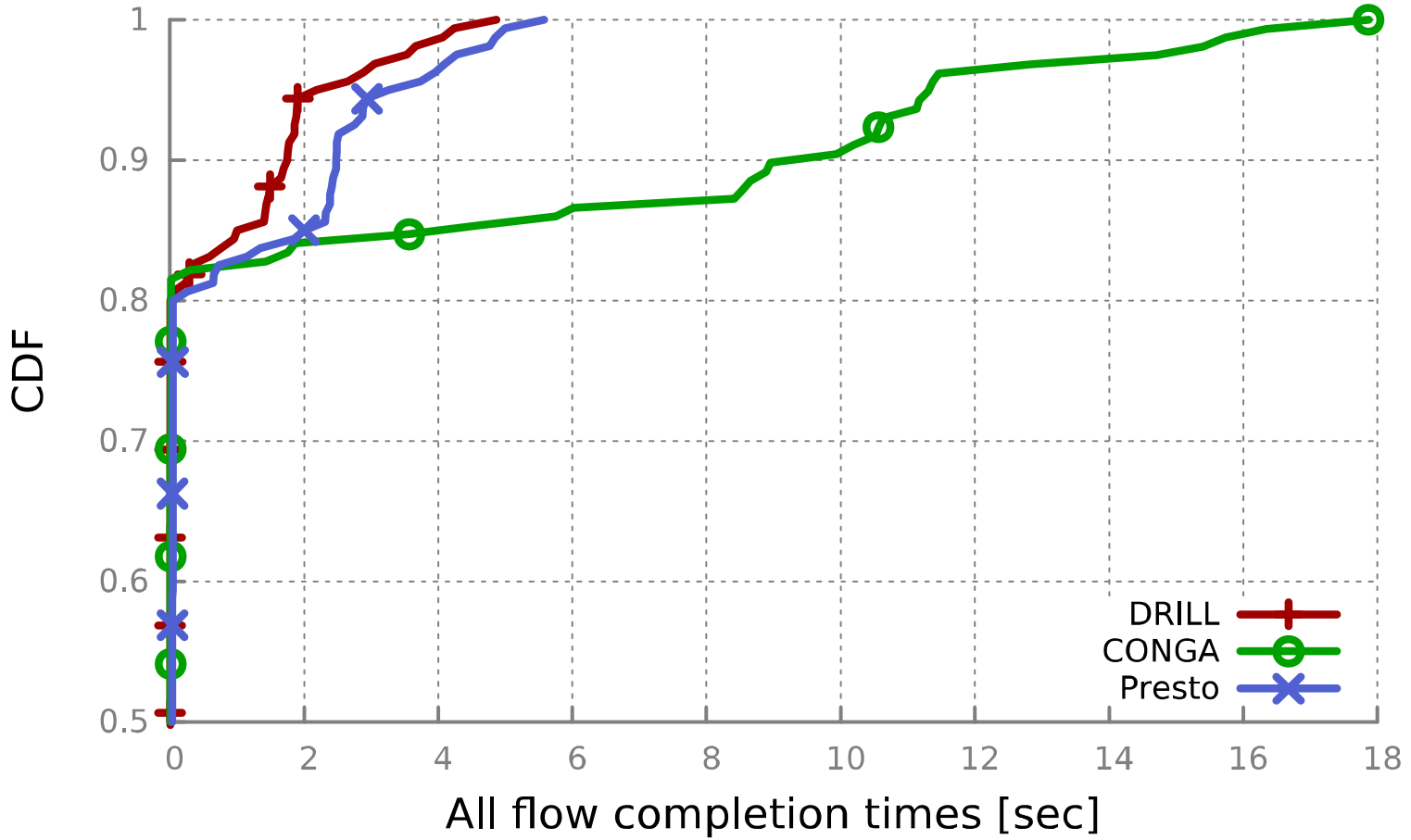
- Setting parameters
- Stability

DRILL

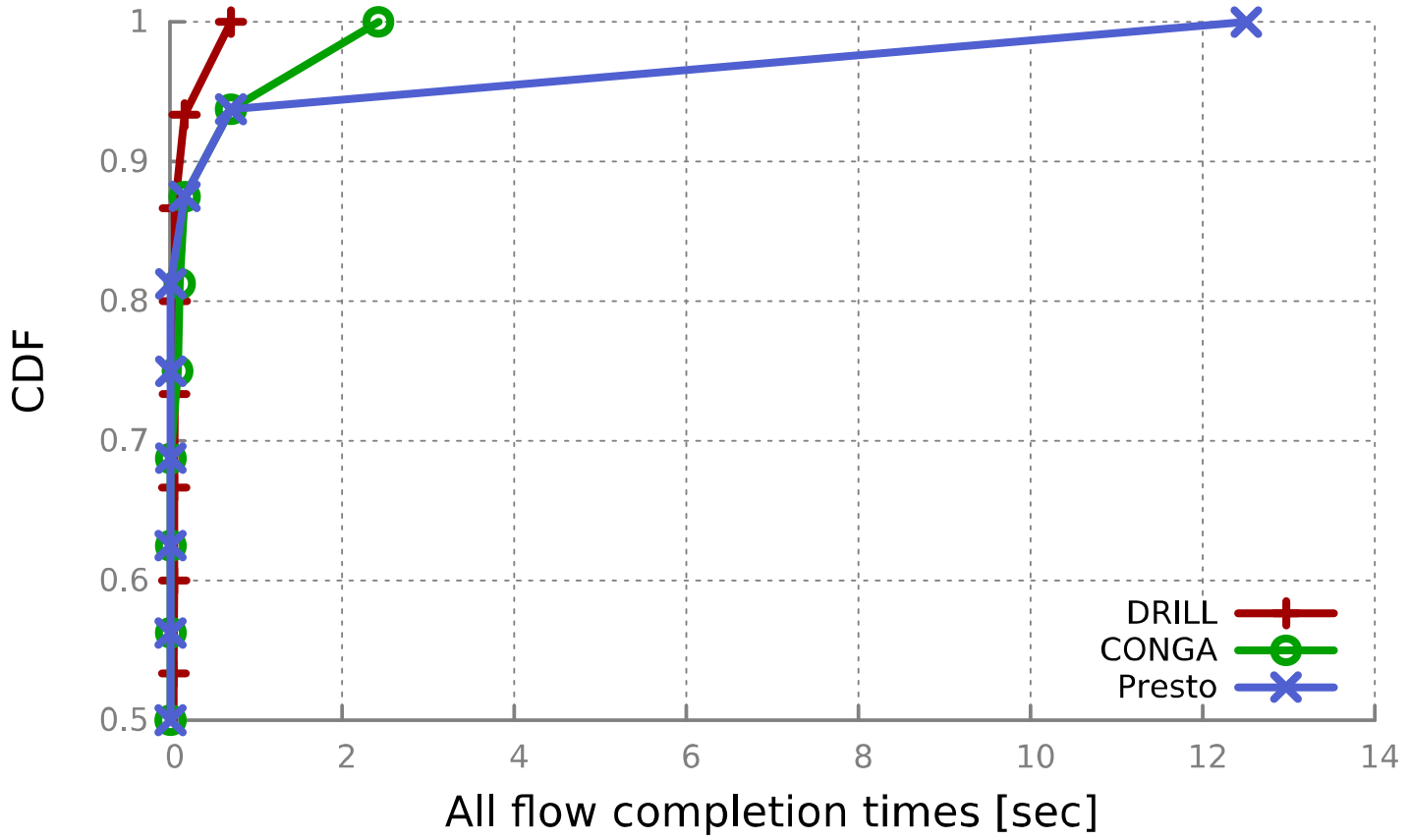


Decentralized
Randomized
In-network
Local
Load balancing

Substantial improvement over prior work.

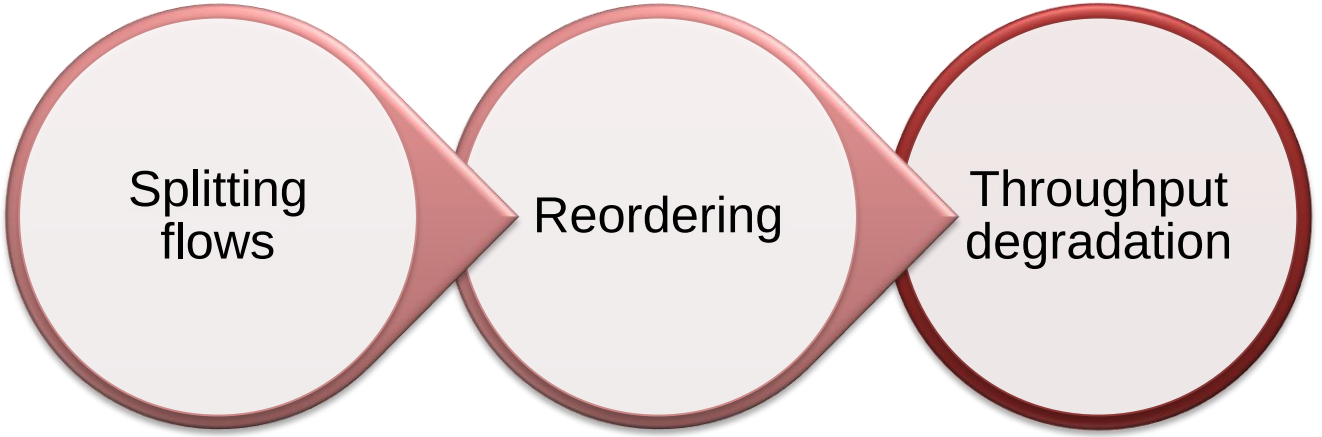


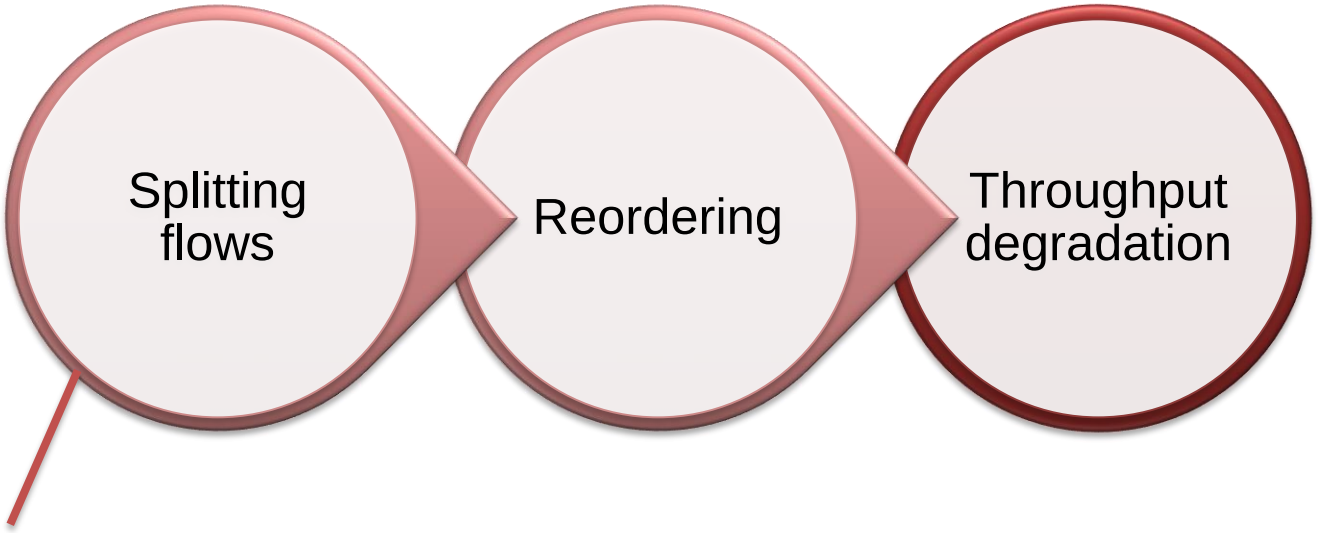
Substantial improvement over prior work.



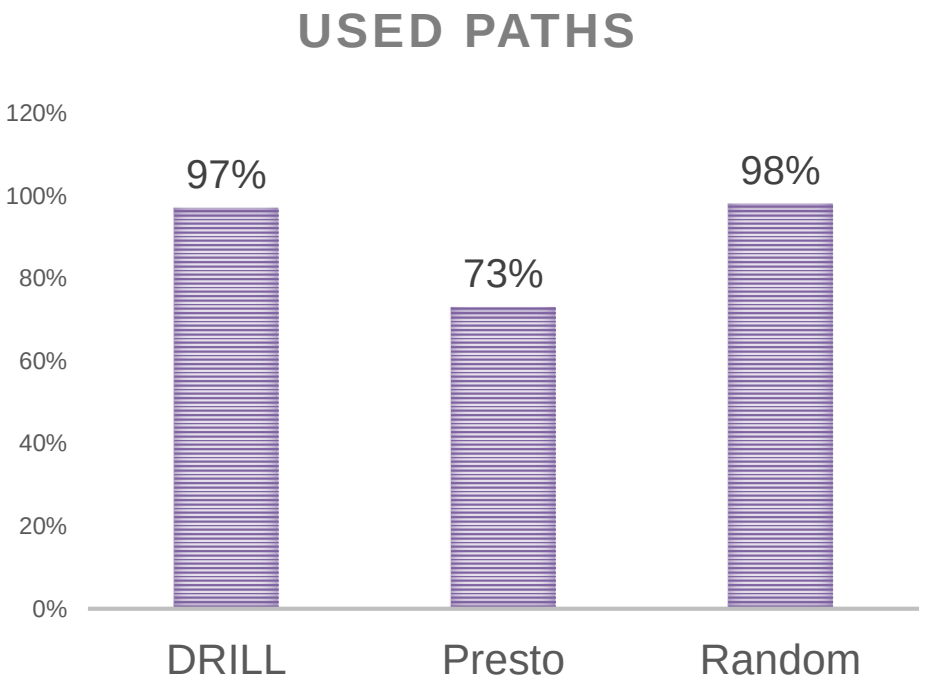
An **incast** application

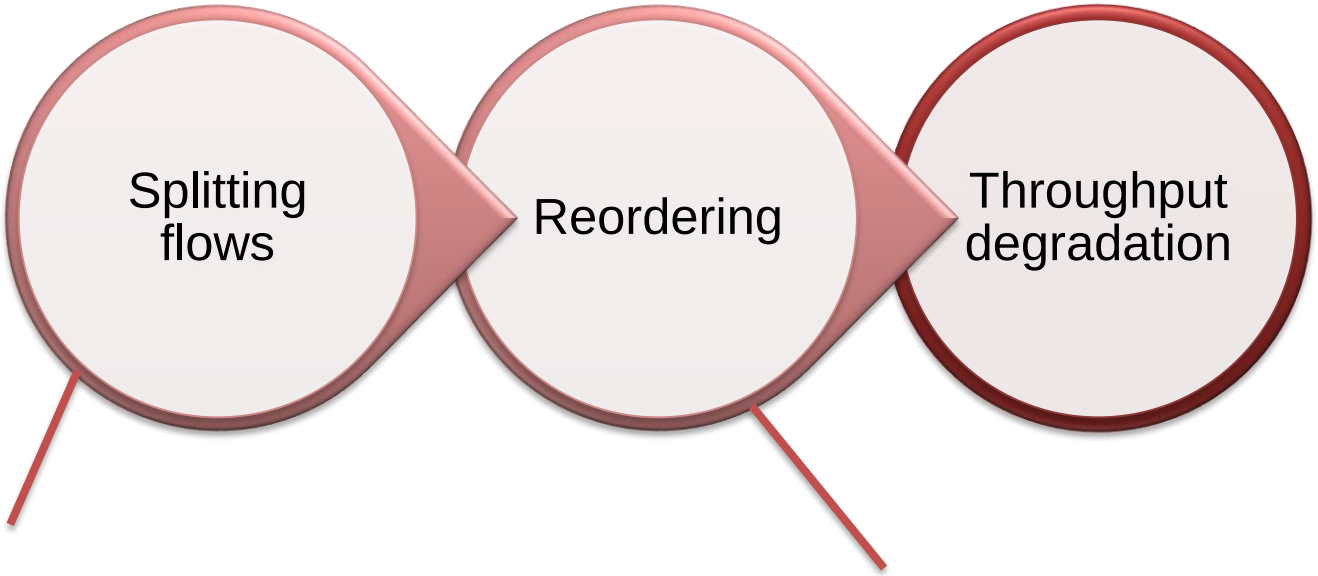
Thou shalt not split flows!





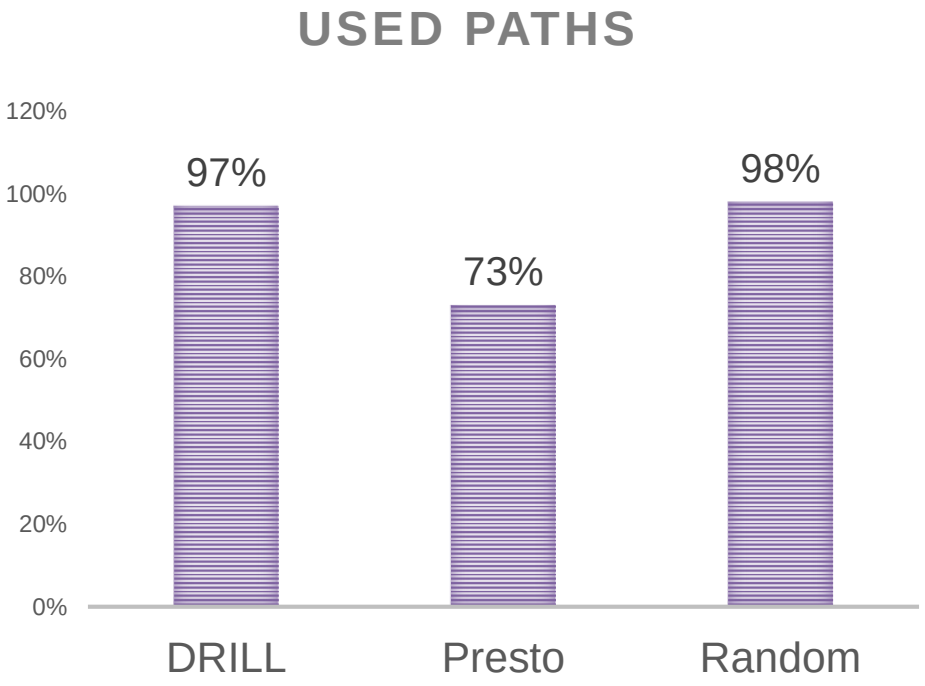
DRILL splits flows along many paths

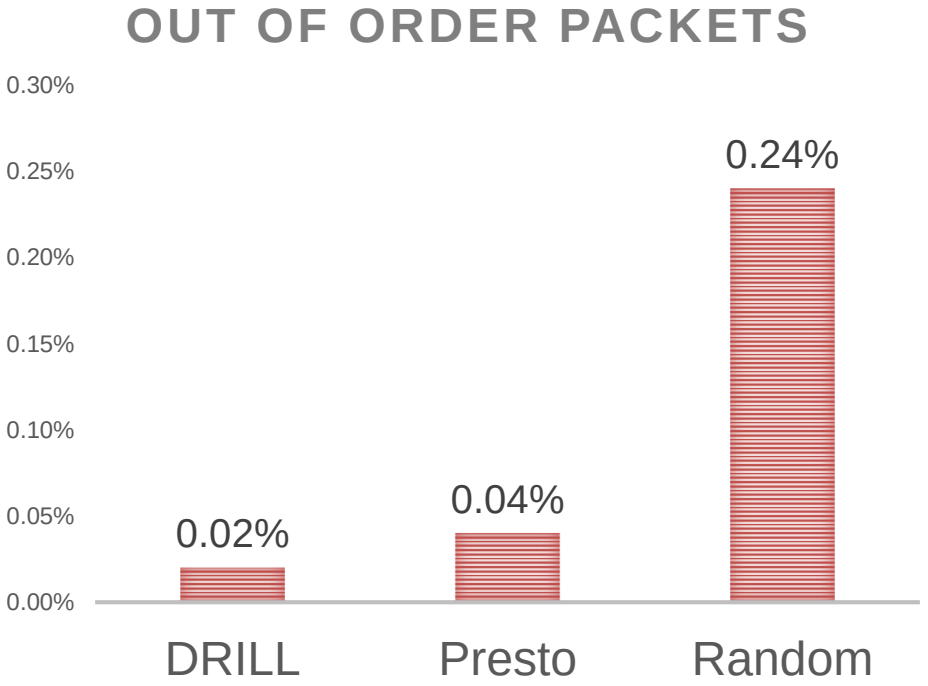


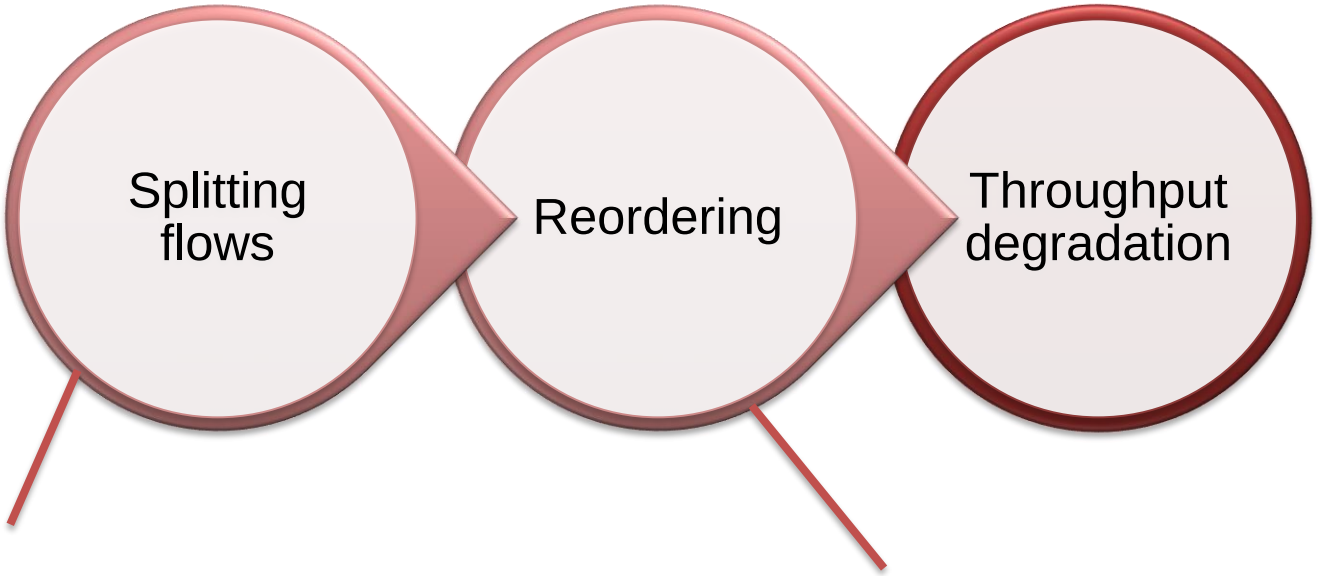


DRILL splits flows along many paths

yet causes little reordering!

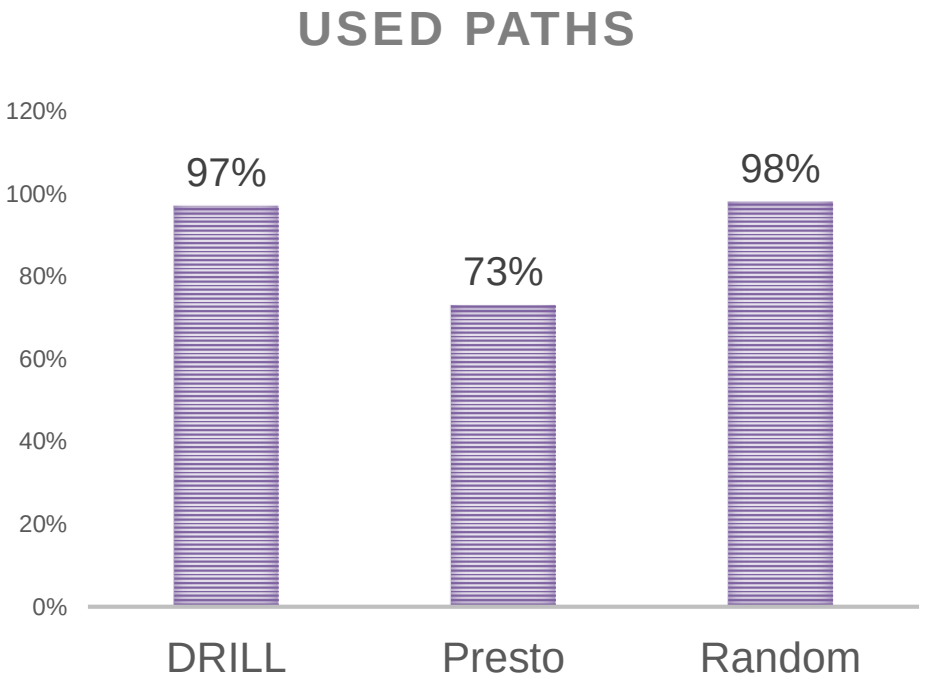


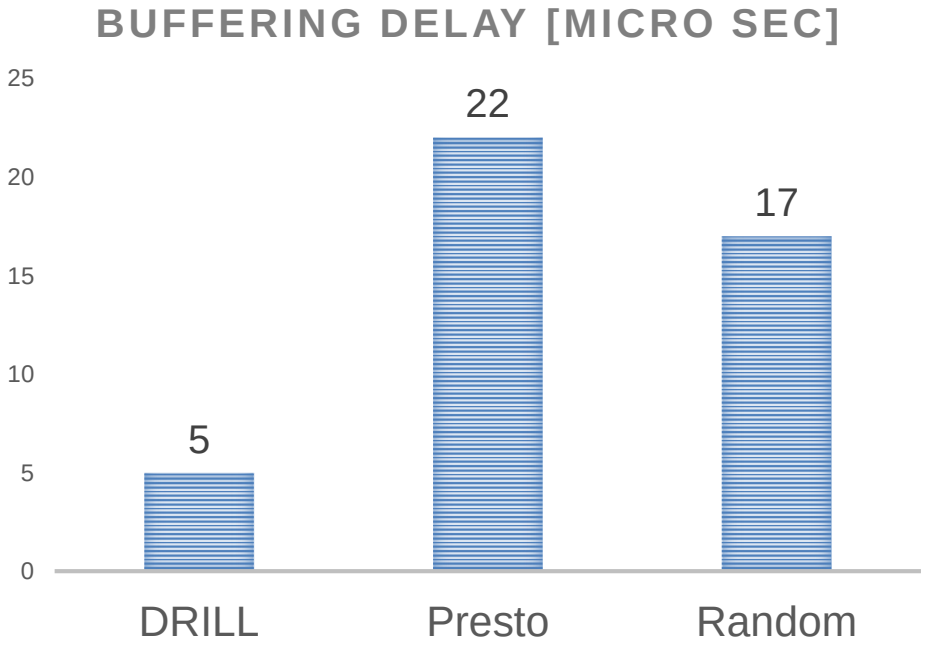


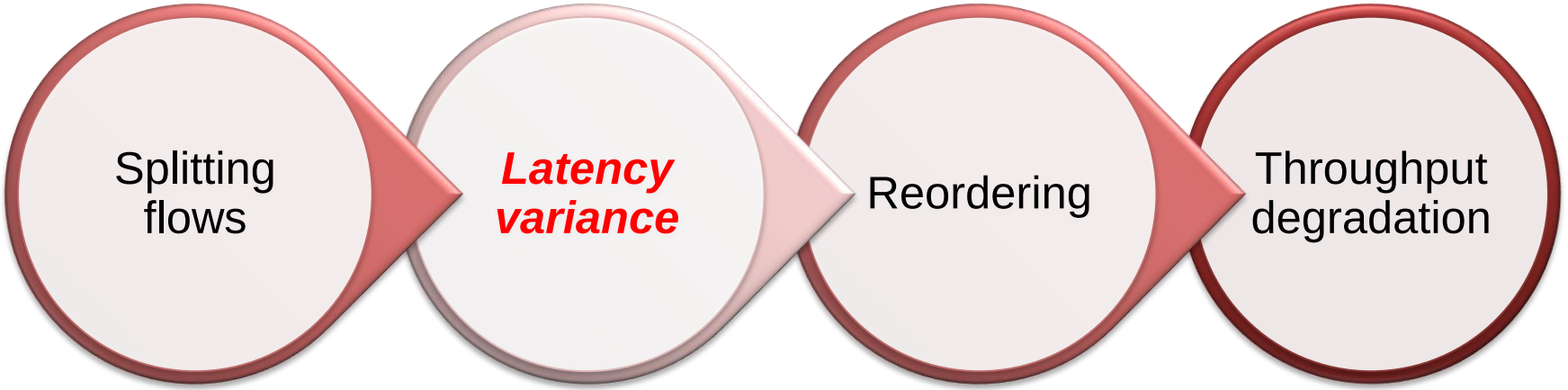


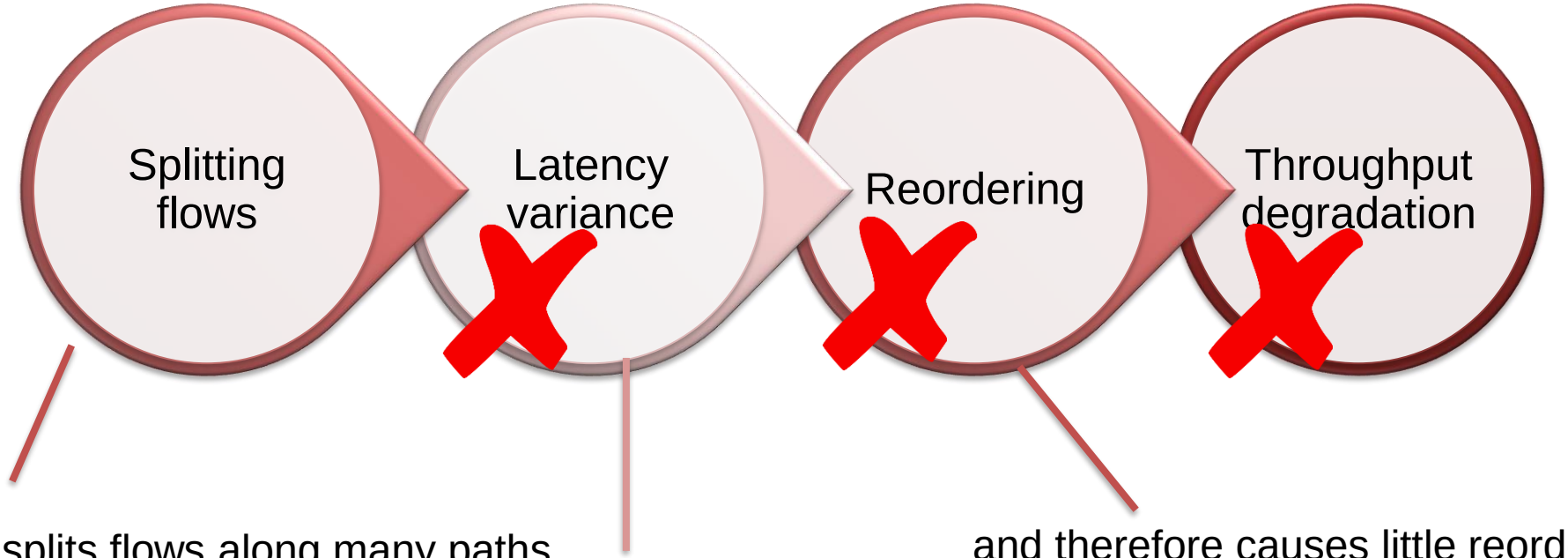
DRILL splits flows along many paths

yet causes little reordering!





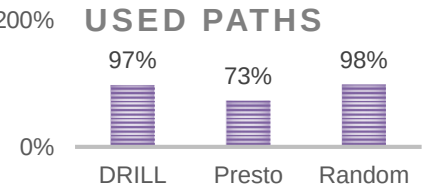

Splitting flows
Latency variance
Reordering
Throughput degradation

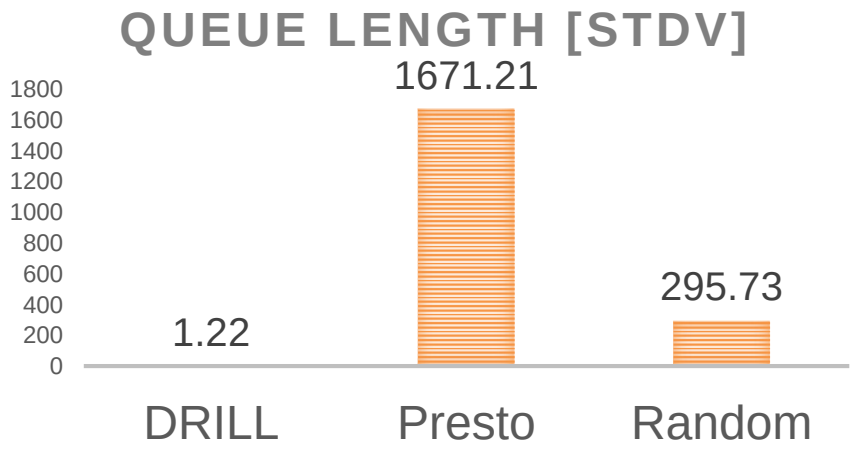


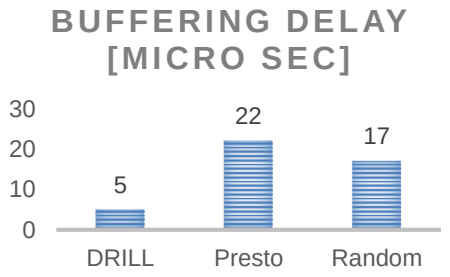
DRILL splits flows along many paths

but has low queueing delay variance

and therefore causes little reordering!







Insight: Queueing delay variance is so small that it doesn't matter what path the packet takes.

Ongoing and future work

Efficient handling of asymmetry

- Failures
- Irregular topologies with non-equal cost paths

Converged Ethernet

Micro Load Balancing with DRILL: Conclusion

- Microscopic, microsecond decisions yield lowest latency load balancing
- Splitting flows is splitting hairs
- Strong candidate for augmenting data center switching hardware