Approximation in Programmable Data Plane

Minlan Yu
Harvard University

Joint works with Rui Miao, Mohammad Tirmazi, Jiaqi Gao, Sivaramakrishnan Ramanathan, Yuliang Li, Michael Mitzenmacher (Harvard), Ran Ben Basat (UCL), Ennan Zhai, Hongqiang Liu, Ming Zhang (Alibaba), Gianni Antichi (QueenMary), and many others

Programmable Data Plane: Many Applications

High throughput, low latency, low energy and capital cost

Network Telemetry [OpenSketch, FlowRadar, LossRadar, PINT]	Load balancing [SilkRoad]	Security [Jaqen]	Congestion Control [HPCC]	Database [Cheetah]

BAREFOOT NETWORKS

Mellanox TECHNOLOGIES

BROADCOM

Alibaba.com

facebook

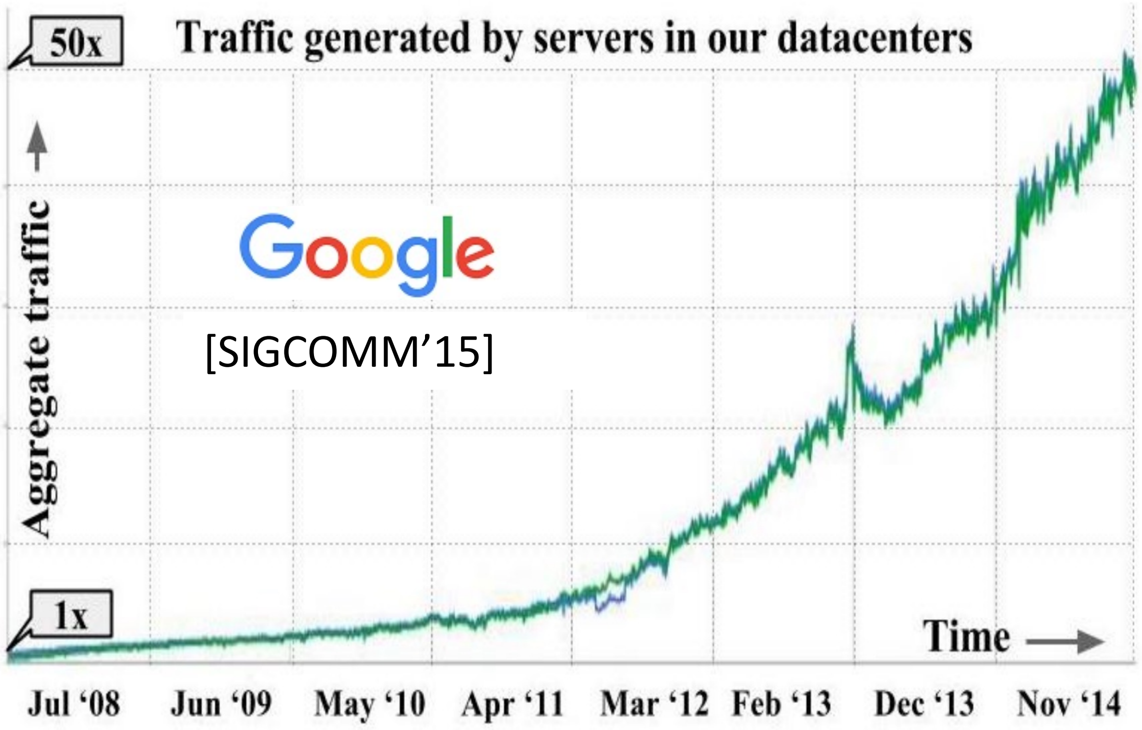
Challenges

- Growing applications
- Switch limitations

Growing applications		Switch limitations
Significant data growth	→	Limited memory
Diverse programs	→	Limited programmability
Increasing line rate	→	Limited per packet processing

Challenge I: Growing Data vs Limited Memory

- Significant data growth



[SIGCOMM'15]

- Slow memory growth

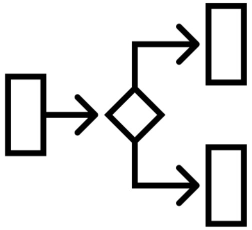
Year	Mem (MB)
2012	10-20
2014	30-60
2016	50-100

SilkRoad [SIGCOMM'17]

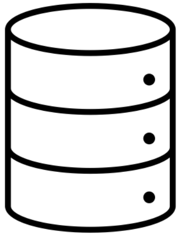
Challenge II:
Program Complexity vs Limited Programmability

- Diverse programs

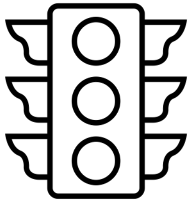
Security



Load balancing



Database

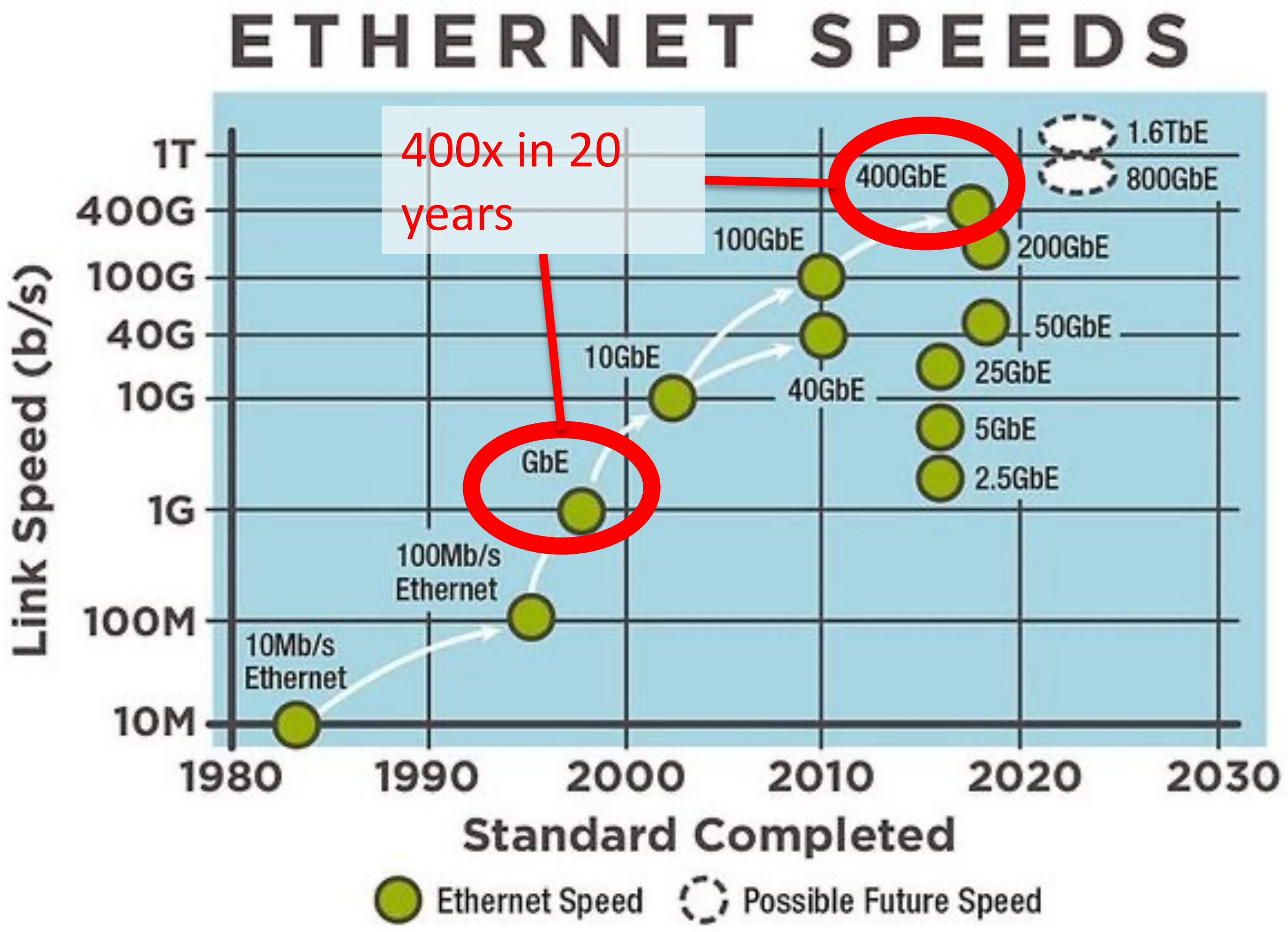


Congestion Control

Network Telemetry

- Limited programmability
  - Was designed for packet processing
  - No floating-point operations
  - Independent operations within a stage
  - Limited state sharing across stages

Challenge III: Increasing BW vs Limited Processing Time



- For each packet
  - More things to do
  - Less time to process

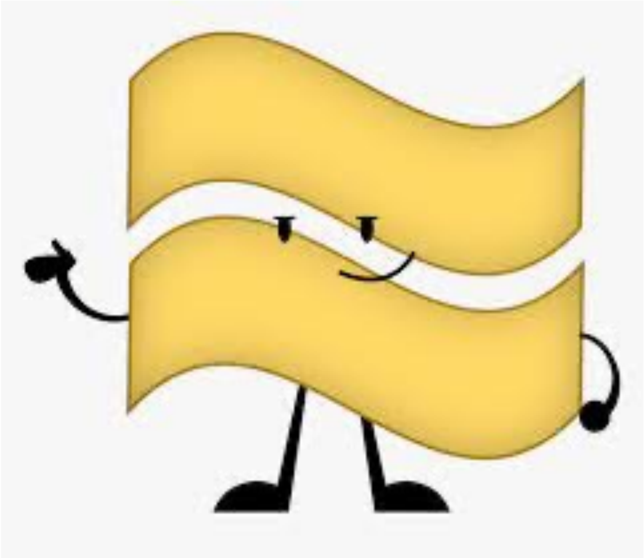
Challenges

- Growing applications
- Switch limitations

Growing applications		Switch limitations
Significant data growth	→	Limited memory
Diverse programs	→	Limited programmability
Increasing line rate	→	Limited per packet processing

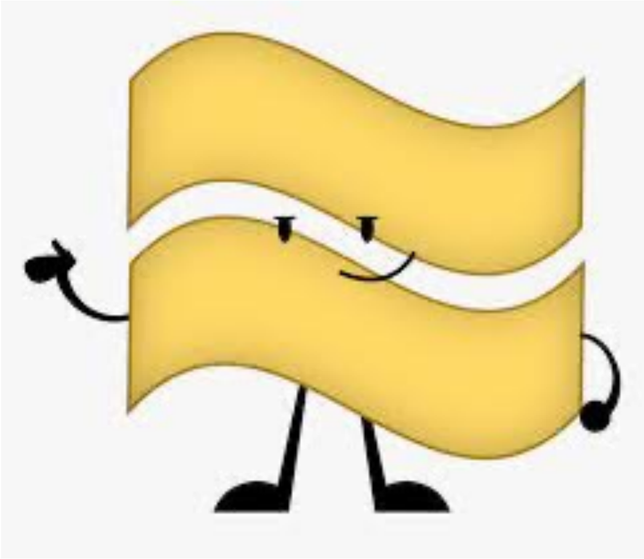
Many Theoretical Techniques on Approximation

- Sampling
  - Randomly select a subset of data
- Sketch
  - Summary data structure for specific query types
- Lossy compression
  - Prune values that ensures approximation bounds
- Coding
  - Combine multiple values across packets
- Distributed algorithms
  - Distributed message passing across nodes



The Gap Between Theory and Practice

- Theory solutions often focus on one constraint
  - Sketch: Reduce memory
  - Coding: Reduce packet bits
  - Distributed algorithms: Reduce #messages
  - How to address multiple limitations in practice?



- Approximation results in practice
  - Will there be errors in the results?
  - What does probabilistic guarantee mean?
  - How to constrain the impact of errors in practice?

Bridge Theory and Practice: Two Examples

- PINT: Probabilistic In-band network telemetry
  - Hashing, coding, sampling, value approximation
  - Handles limited packet bits and programmability
  - Minimize errors through aggregation

- Cheetah: Database queries with switch pruning
  - Sampling, hashing, sketch, lossy compression
  - Handles limited memory, programmability, and packet processing time
  - Deliver accurate results with server processing

PINT
Probabilistic In-band Network Telemetry

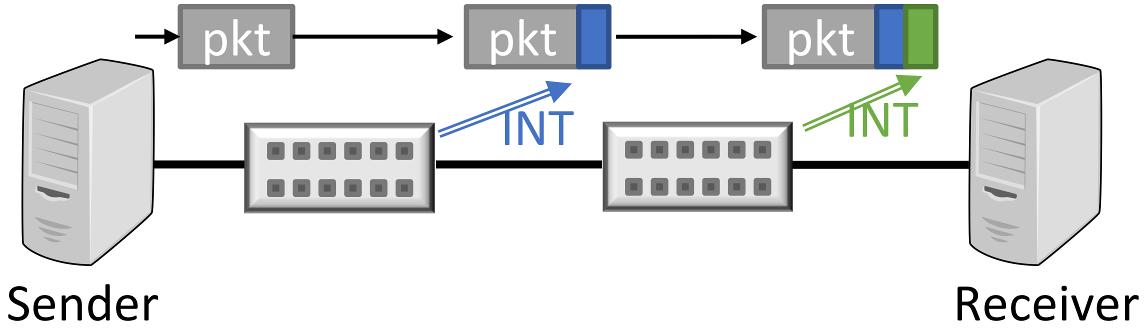
(SIGCOMM’20)

Measuring Packet-level Events

- Diverse queries on packet lifetime
  - Which path do my packets take?
  - Which firewall rules do my packets follow?
  - Which switch/link has the highest latency for my packets?
- Useful for real-time control and feedback loop
  - E.g., congestion control, load balancing, troubleshooting, etc.

INT: In-band Network Telemetry

- INT: add switch states in packets and analyze at the receiver
  - E.g., Switch ID, Queuing delay, link utilization



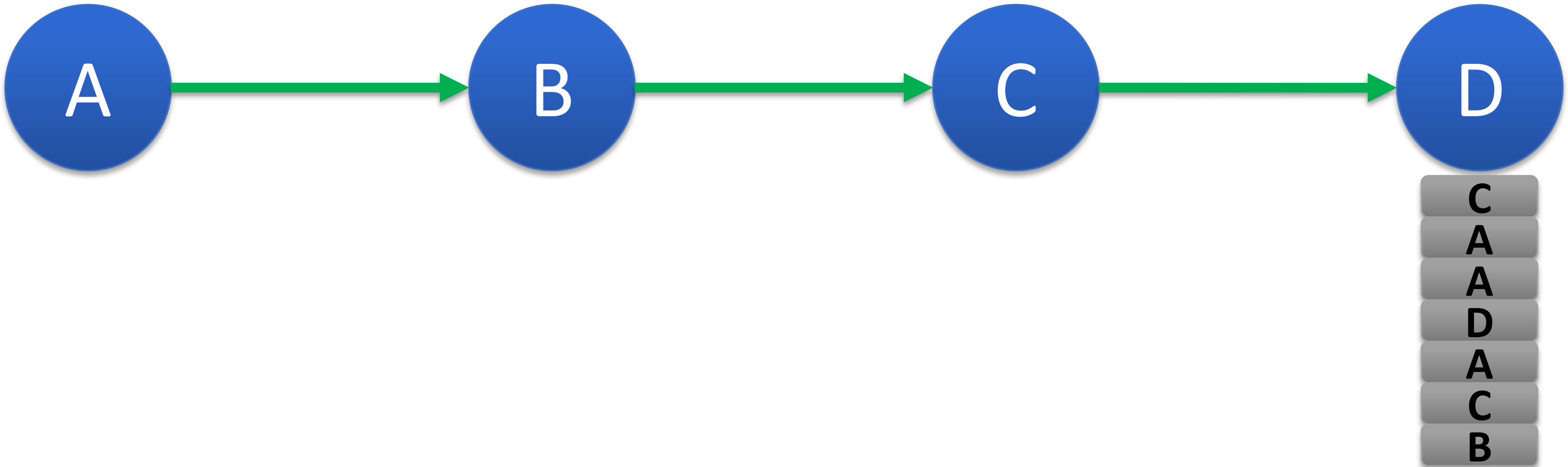
- Key problem: high bit overhead
  - Many switches, many types of information
  - Up to 20% reduction of goodput

PINT: Probabilistic In-band Network Telemetry

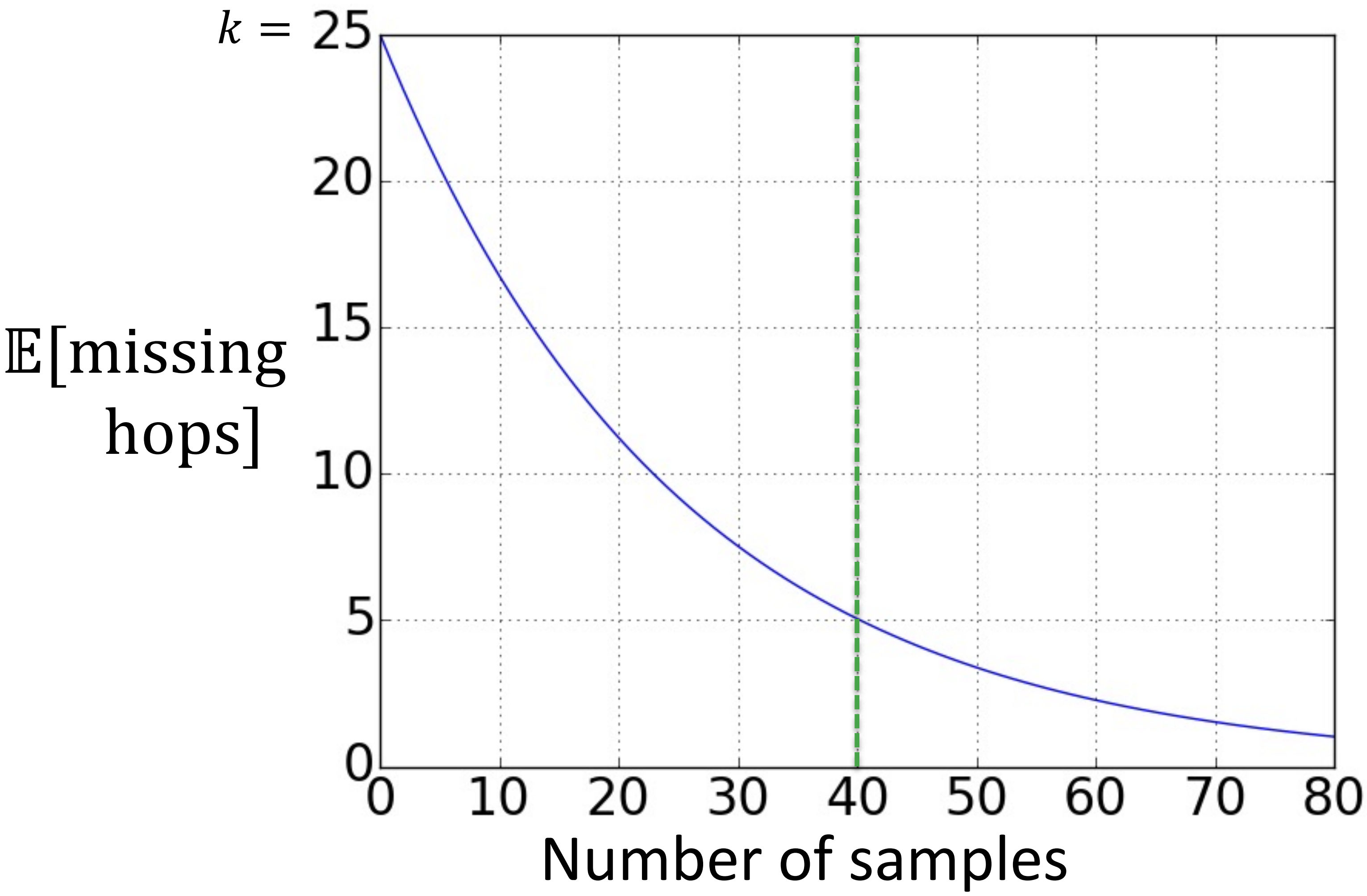
- Goal
  - Encode telemetry information on packets with fewer bits
- Insight
  - Most apps don’t need per-packet per-switch values, but aggregated data
  - Leverage probabilistic solutions to aggregate across packets and flows

Flow-level Path Tracing

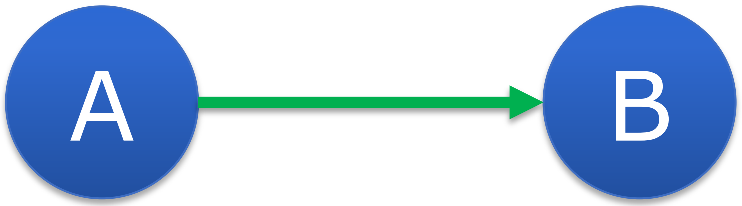
- Baseline solution: write a sampled ID on each packet
  - We can use the TTL field to get the hop number and run Reservoir Sampling (Sattari et al., 2010).
  - A *Coupon Collector* process. For k hops it will take $k \ln k\,(1 + o(1))$ packets to detect the path.



Coupon Collector Process

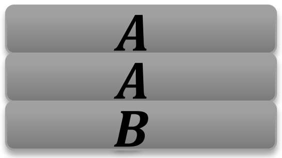


The Power of Coding



Baseline:

- Get information on the first packet.
- Require 2 packets on average to get the second hop ID

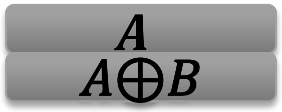


  - Overall: $1 + 2 = 3$ packets in expectation.

Coding solution:

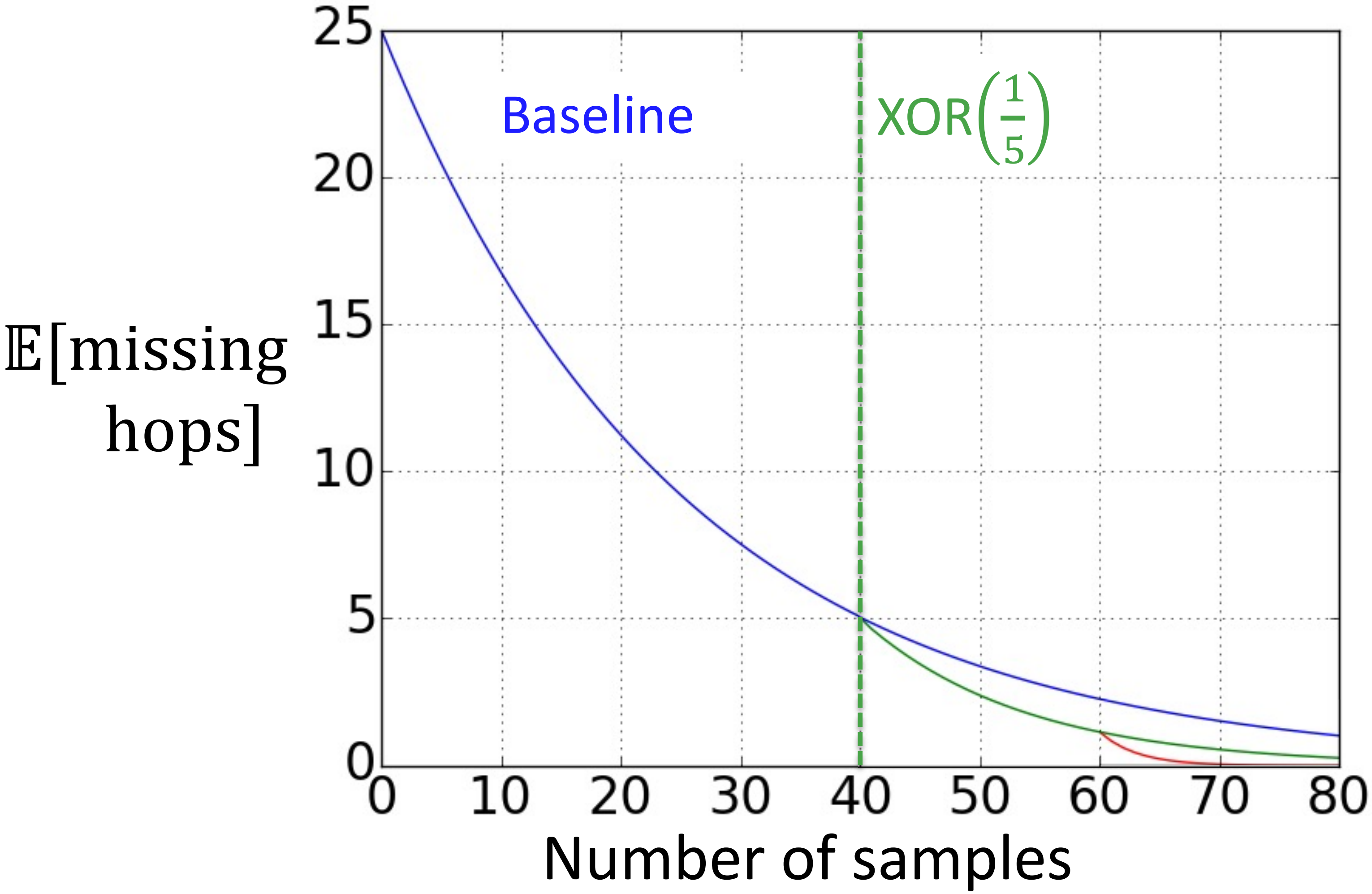
Consider baseline sampling with probability 0.5, and writing $A \oplus B$ otherwise.

- If the first packet is an ID (e.g., A), we need $4/3$ more packets on average.
- If the first packet is $A \oplus B$, we need 2 more packets

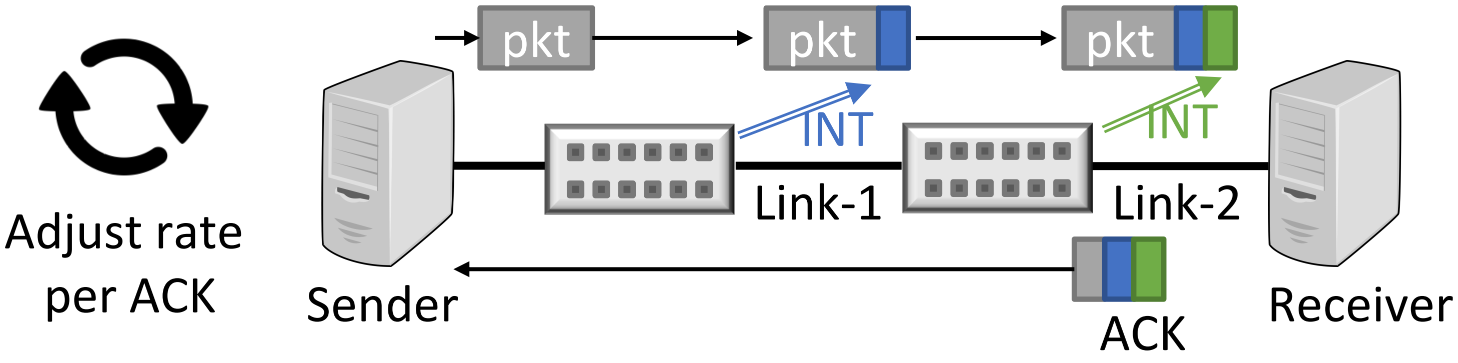


  - Overall: $1 + \frac{4/3+2}{2} = 8/3$ packets in expectation.

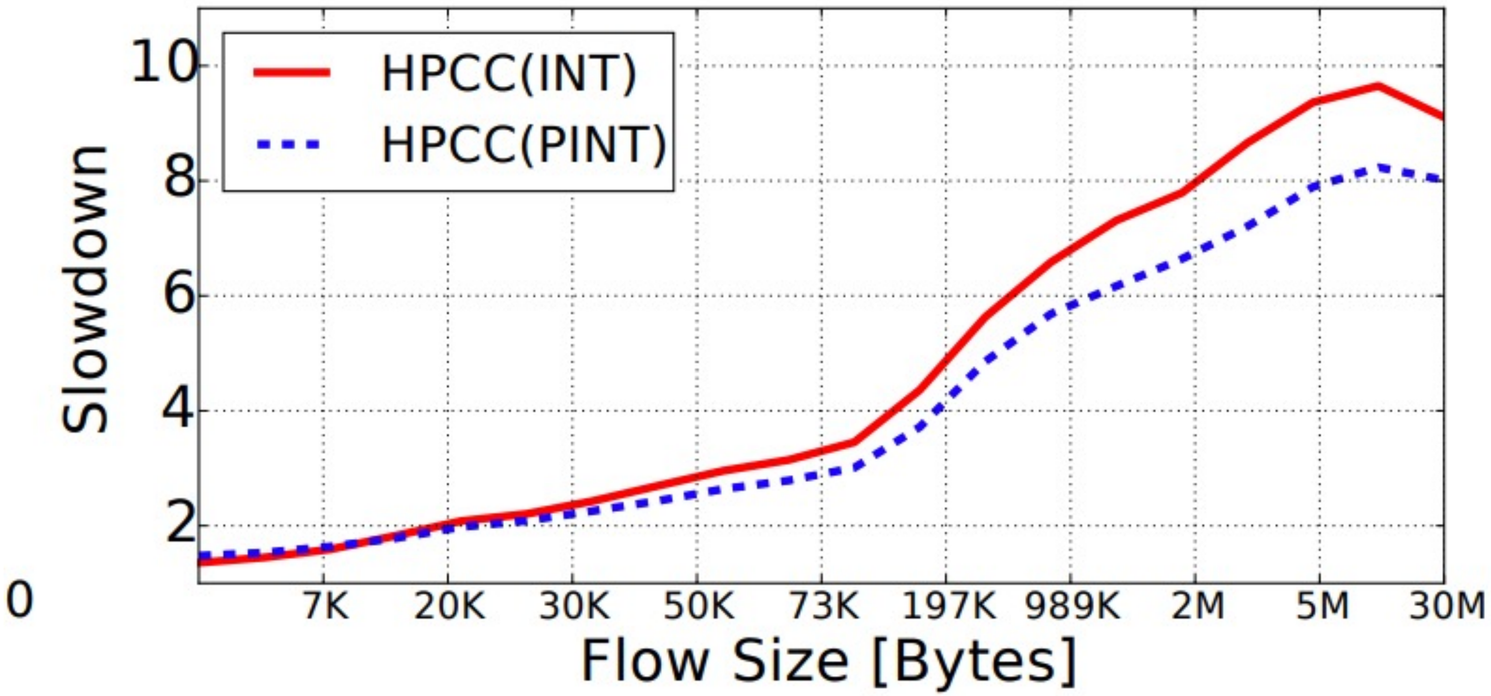
Improving the Coupon Collector



High Precision Congestion Control over PINT



HPCC (SIGCOMM'19)

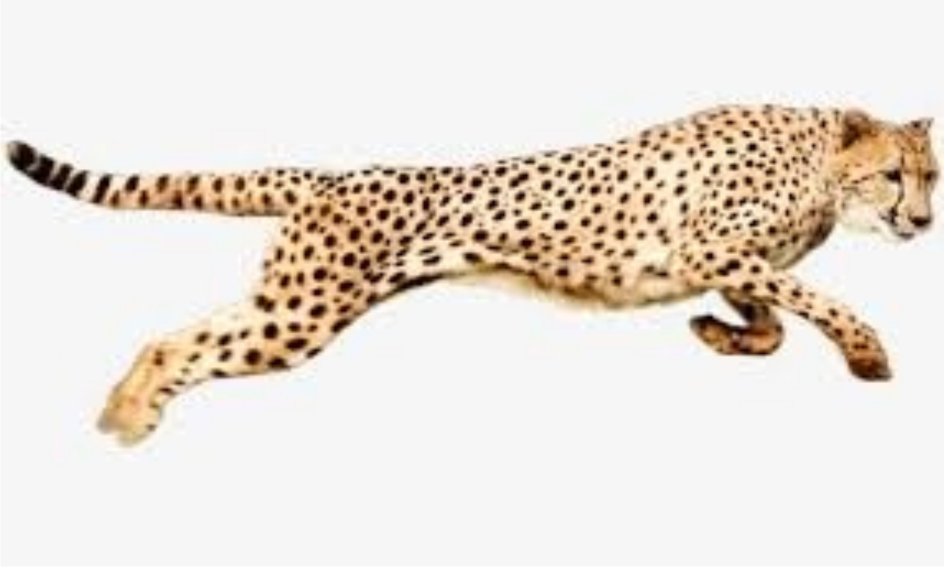


PINT Conclusion

- Approximation to reduce packet overhead
  - Coding, hashing, sampling, value approximation
  - Provable guarantees on #packets and #bits for high accuracy
- Support a variety of aggregation queries
  - Path query
  - Max queue length, median and tail latency etc.
  - And a mix of these queries

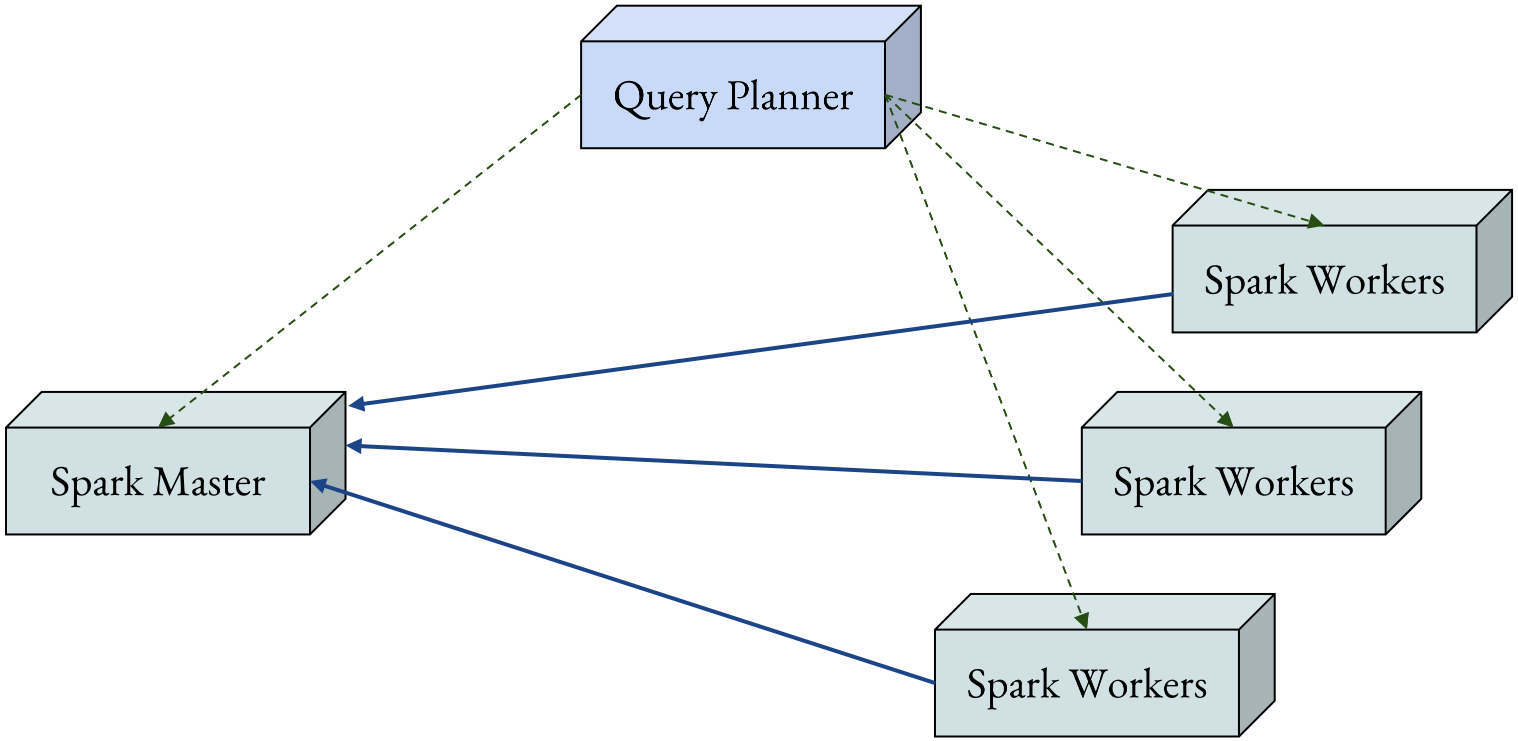
Cheetah:
Accelerating Database Queries with Switch Pruning

SIGMOD'20

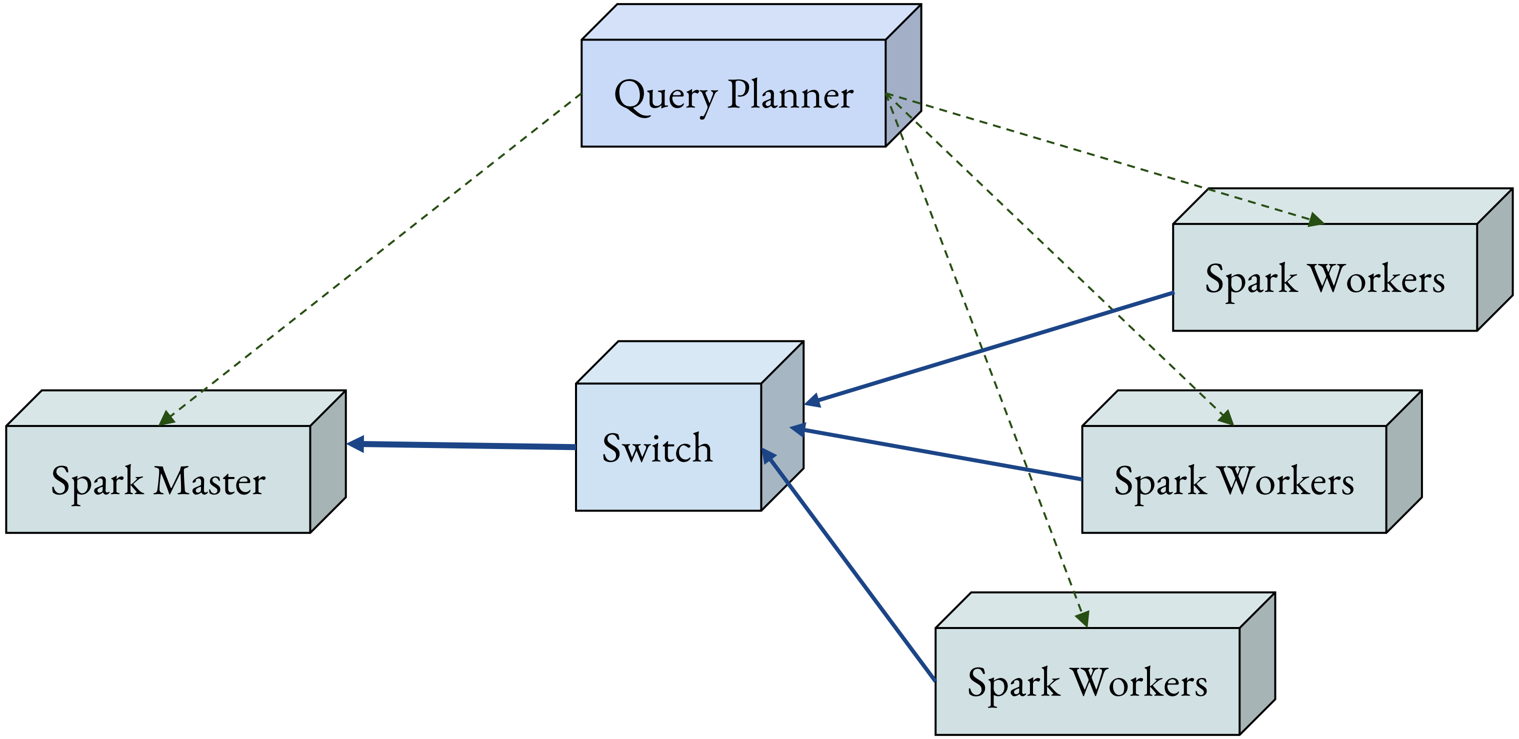


Database Operations

- Large amount of data
  - Over 8 billion queries/day in Alibaba Cloud
- Highly optimized for performance
  - Parallelize data processing at workers



Why Programmable Switches?



Already in the network.

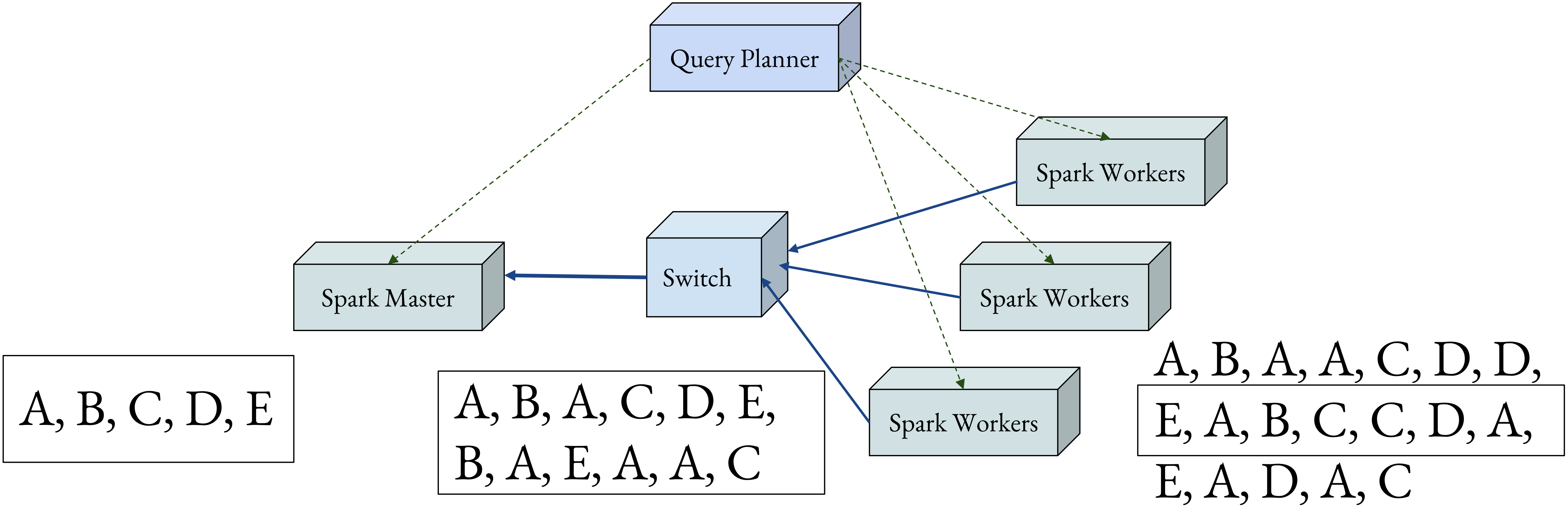
Process Tbps of data

Process **cross-partition** data.

Key Challenge: Switches have limited programmability and limited memory

The Pruning Abstraction

Query on **pruned** dataset = Query on original dataset

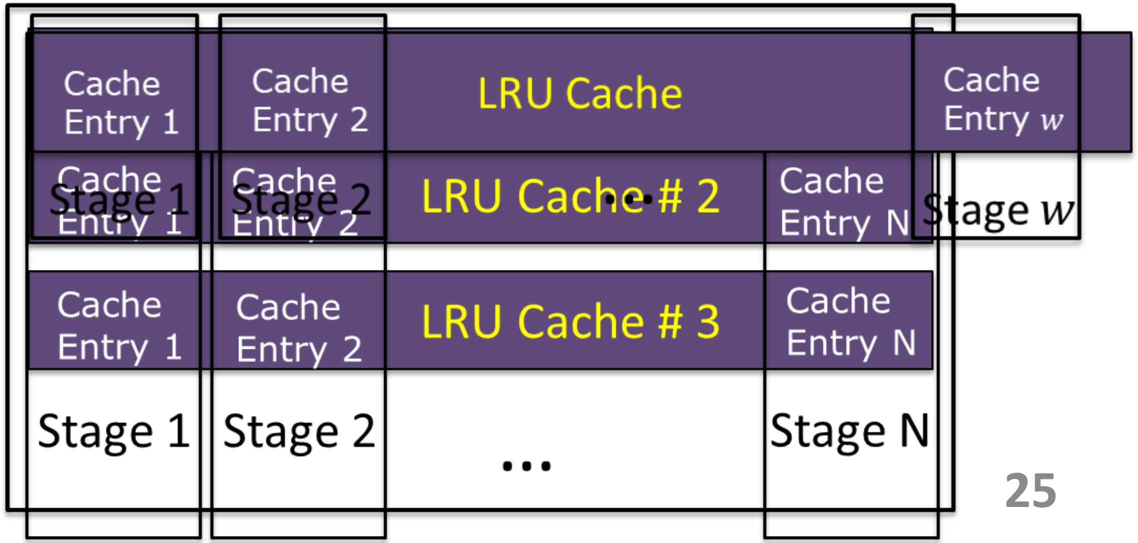


Pruned data set
Using switches

Original dataset

Example: Distinct Query Pruning

- Selects all the distinct values
- Strawman solution: Bloom Filters
  - Problem: have false positives which may drop distinct entries
- But a cache works!
  - Implement LRU with a rolling replacement across stages
- Our solution: Multi-row LRU cache
  - Reduce #per-packet comparison



Cheetah Results

- Support a wide variety of database queries
  - Join, Group-By, Having, Skyline, Top-K, and Filtering
  - And their combinations

- Approximation algorithms for switch pruning
  - Sampling, hashing, sketch, lossy compression
  - Expected pruning rates

- Integrated with Spark and Tofino switches
  - 40-75 % faster completion time on database benchmarks

Bridge Theory and Practice

- PINT: Probabilistic In-band network telemetry
  - Hashing, coding, sampling, value approximation
  - Bridge the gap of limited packet bits and programmability

- Cheetah: Database queries with switch pruning
  - Sampling, hashing, sketch, lossy compression
  - Bridge the gap of limited memory, programmability, and packet processing time

How to make it easier to build the bridge?

Challenges of Programming in the Data Plane

Portability

Migrate program across switches

Extensibility

Distribute program across multiple switches

Composition

Fit multiple programs into one switch

Chip Vendors

BROADCOM

CISCO

BAREFOOT NETWORKS

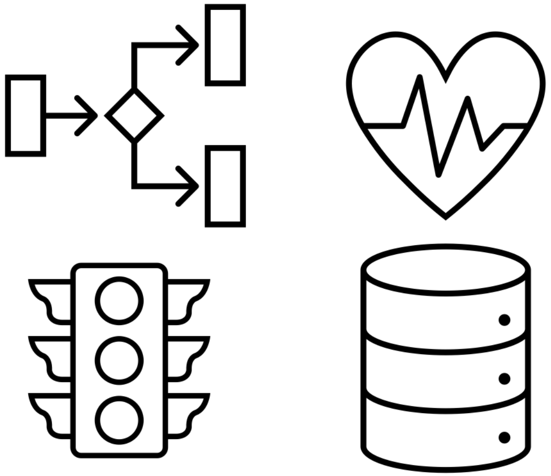
Mellanox TECHNOLOGIES

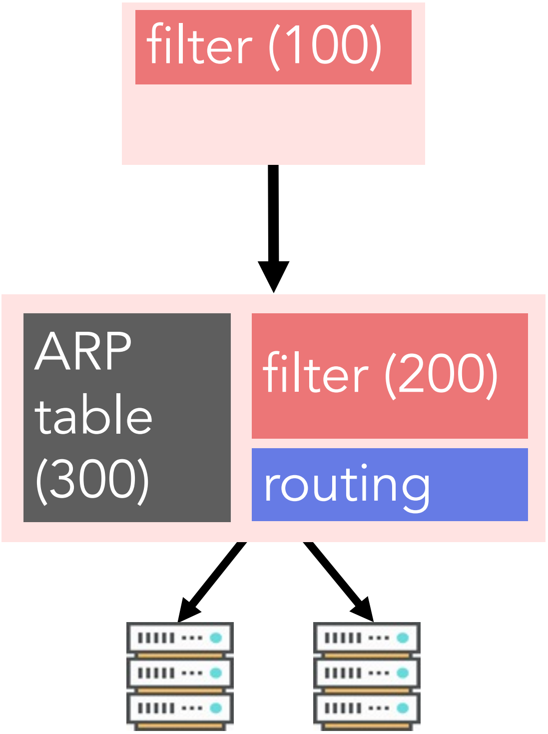
Languages

NPL

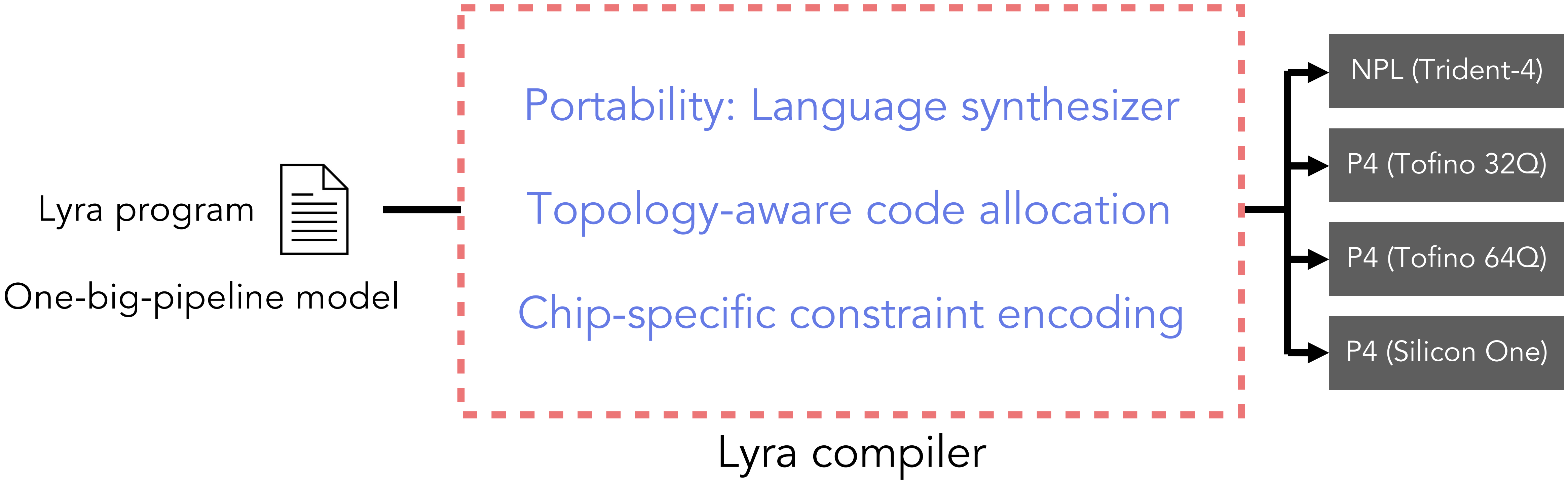
P4

Programs





Lyra: A Data Plane Language & Compiler (SIGCOMM'20)



From assembly language to "C" language

Going Forward: Bridge Theory and Practice

- From practice to theory
  - A theoretical model for programmable data plane
  - Computation model, communication model, resource constraints and tradeoffs
- From theory to practice
  - Libraries for approximation operations and data structures
  - Automatic compilation to diverse data planes

From "C" language to "MapReduce" models

Thank you!