Special Topics: Machine Learning (ML) for Networking

COL867

Holi, 2025

Network Telemetry

Tarun Mangla

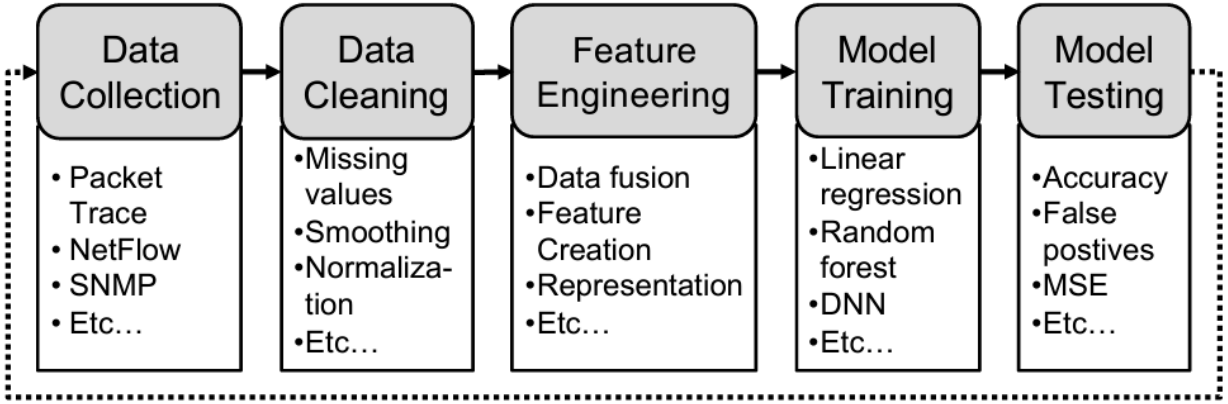
ML for Networks

Module 1: Case studies of specific network learning tasks

Module 2: Task-agnostic automatic ML pipelines for networks

Module 3: Beyond feature engineering and modeling

- Network telemetry
- Robustness and explainability
- Synthetic data generation*
- Data imputation
- Formal verification
- ..



Network Telemetry

- Network telemetry: Collection and analysis of network data

  Fundamental to ML for networking

- Given a network learning task, how would you collect network data?

- Two approaches:
  - Endogenous data collection

    Challenge: at scale & flexibility
  - In controlled network conditions

    → No noise

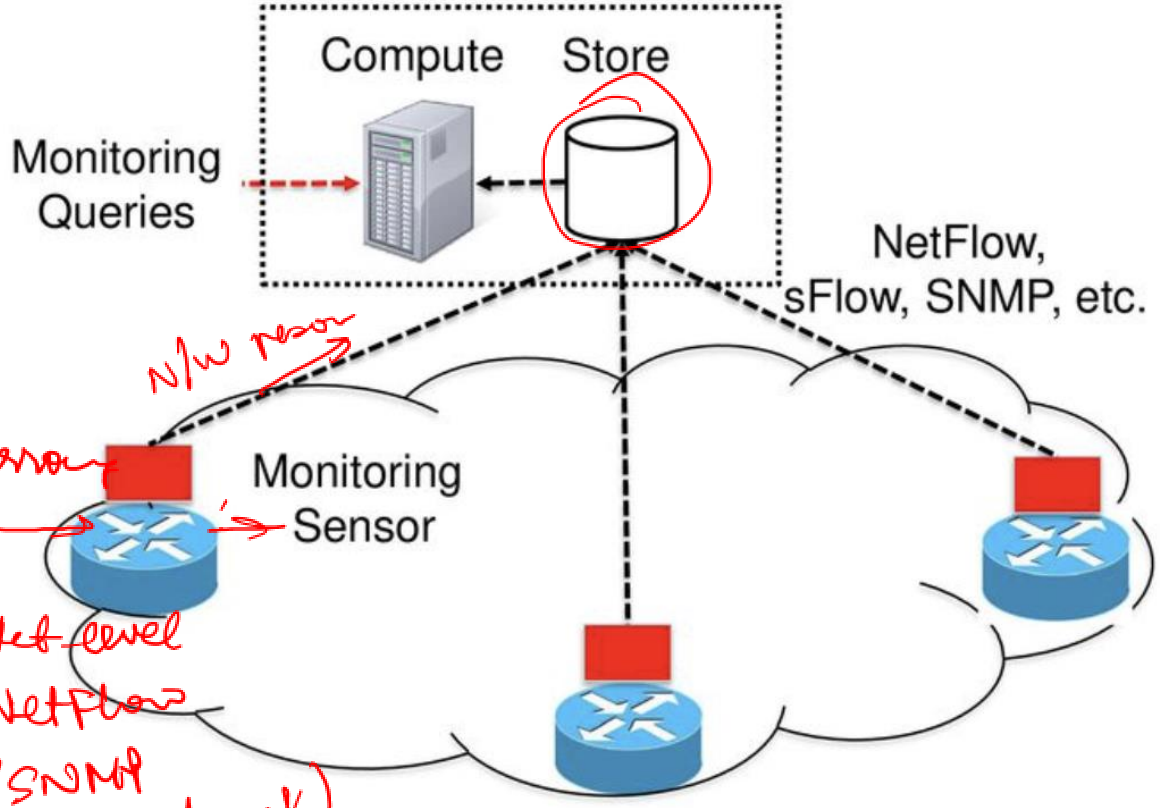
    → Easier to label

    LIMITATION: (1) May not represent the actual distribution

Significant storage & N/w overhead

Conventional Endogenous Network Telemetry

- Bottom-up data collection
  - Monitoring at routers and analysis at centralized servers
- Limitations:
  - → Flexibility vs scalability
- How to mitigate this cost?



N/w resource

TAP / Port mirror

packet-level / NetFlow / SNMP (Router-level)

1-minute summary

Top-down Network Telemetry

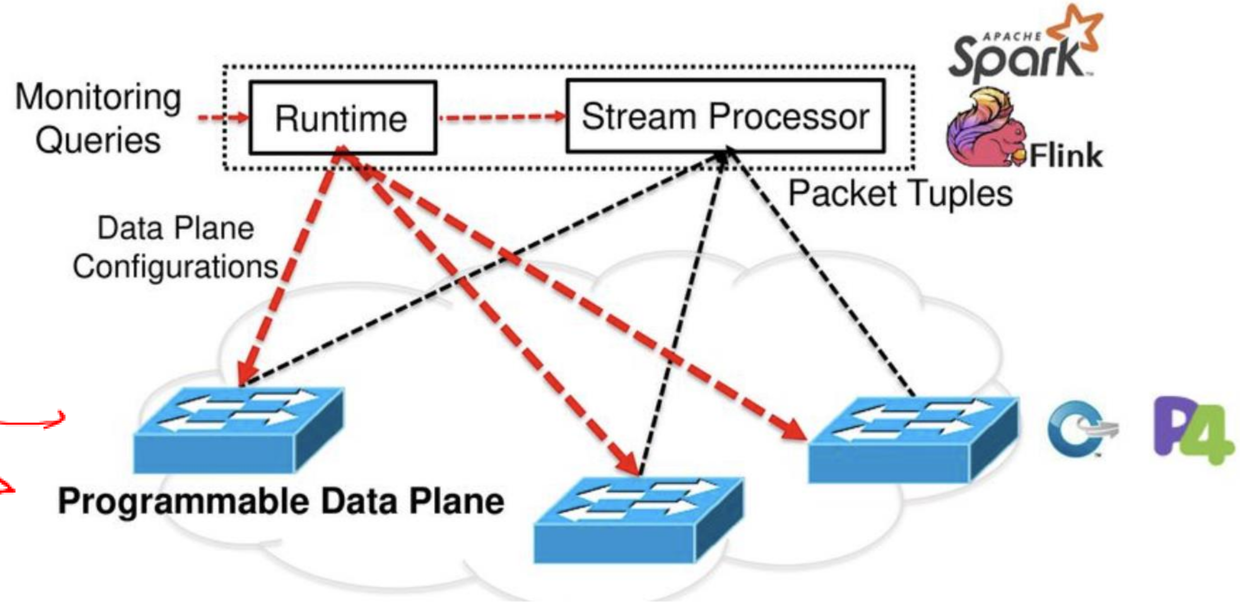
Control Plane

Data Plane

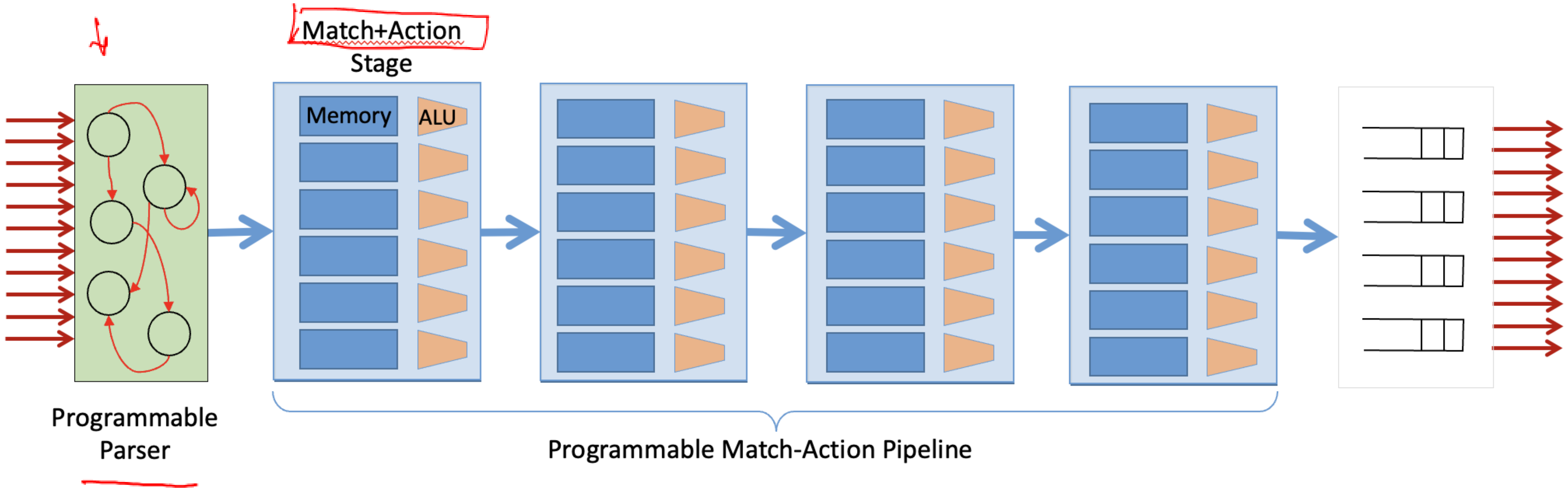
Hardware

- Query-driven network telemetry
- Processing in the data plane
- But aren't routers only forwarding devices?

Not true anymore



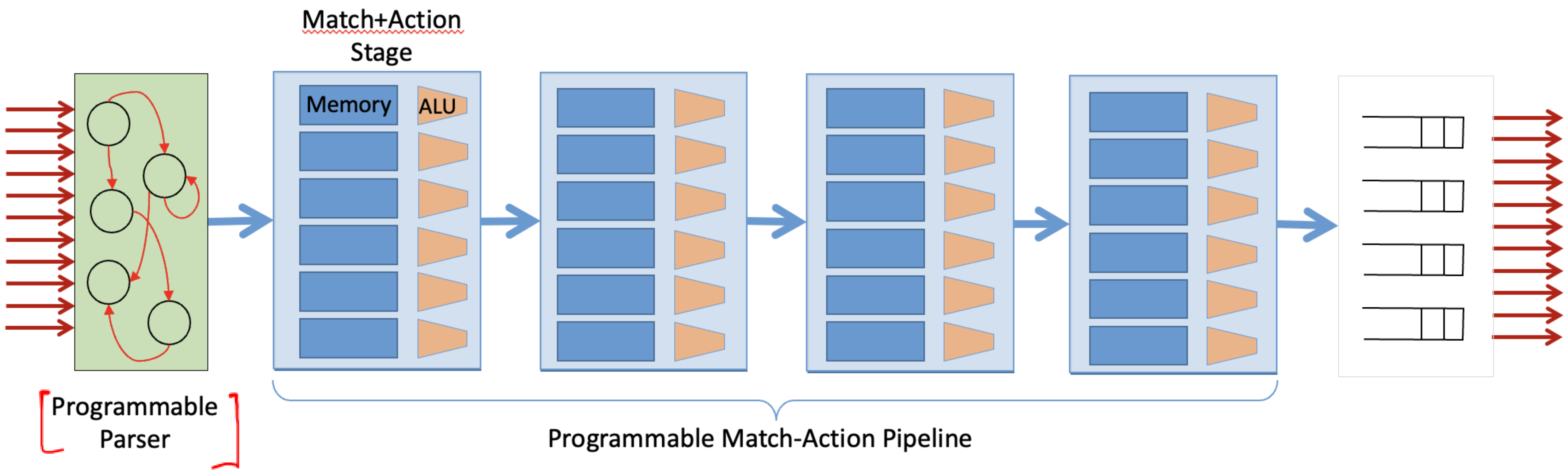
Programmable Data Plane Architecture



Key functionalities

- Programmable parser, extract specific fields of interest
- Matching, to group related packets into a single ‘flow’
- Computation using simple ALUs (addition, subtraction, bitwise operations)
- Storage of information across successive packets using a small number of register arrays
- Communication with a software controller

Telemetry using Programmable Data Plane



How to calculate the number of packets per flow?

① Extract the 5-tuple

② 1 register for each 5 tuple

TCP: Close signal
FIN / RST

UDP:

Programmable Data Plane Constraints

- Memory is the biggest constraint
  - Packets forwarded without going through the main memory
  - Hence, measurements need to be performed in the register arrays
- ALUs available are simple
  - Can not perform complex operations
- Measurement is typically a second-class citizen
  - Gets limited budget compared to routing

How to calculate the number of packets per flow when number of flows are large?

↳ Approximate value → compress the data & store approximate

Is this a New Problem?

- **Big data** caused similar problem
  - Billions of records generated per second (e.g., social media logs)
- Datasets too large to fit in memory
- How did big data folks handle this problem?

Data Streaming Problem

- **Data stream:** a sequence $A = \langle a_1, a_2, \ldots, a_m \rangle$, where the elements of the sequence (called tokens) are drawn from the universe $[n] := \{1, 2, \ldots, n\}$
- **Aim** - compute a function over the stream, eg: median, number of distinct elements, longest increasing sequence, etc.

- **Target Space complexity**
  - Since m and n are “huge,” we want to make s (bits of random access memory) much smaller than these
  - Specifically, we want s to be sublinear in both m and n.

Use Sketch

What are Sketches?

- Data structure that support approximate computing some function of data
- Summarize data into a much smaller space
- Two operations: update and query
- Common sketches
  - Set membership
  - Counting

$|S| : m$

$|U| : m \to O(m)$

Set Membership Task

$\in U$

e

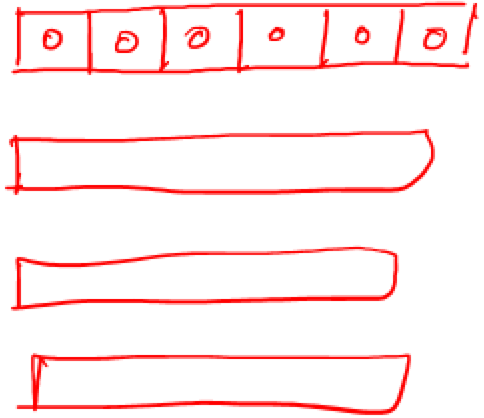
- **Problem statement**: Given a set S, does a given query element belong to S?
- How would you solve this problem in linear space?

Space complexity: $O(n)$: $|S|$

- How to solve this in sub-linear space?

$h : U \to \{1 \ldots\ldots w\}$

$w \ll |U|$ S : $h(S) = 1$



K - boolean vector

or K - hash map

Bloom Filter

→ False positive

- **Query:**

K hash function, 1 boolean vector of |w|

h1 : mod 10

h2 : mod ~~..~~ mod 10

update (25)

update (21)

query (22)

. 3 and 1st bit

Bloom Filter

- Consists of:
  - A vector of n Boolean values, initially all set false
  - k independent hash functions, $h_0, h_1, \ldots, h_{\mathrm{k}-1}$, each with range {0, 1, … , n-1}

F	F	F	F	F	F	F	F	F	F
0	1	2	3	4	5	6	7	8	9

n = 10

Bloom Filter → # of distinct element in sub-linear space

O'

- Consists of:
  - A vector of n Boolean values, initially all set false
  - k independent hash functions, $h_0, h_1, \ldots, h_{k-1}$, each with range {0, 1, … , n-1} $\Theta(n)$
- **Update:** For each element s in S, the Boolean value with positions $h_0(s), h_1(s), \ldots, h_{k-1}(s)$ are set true
- **Query:** True if h(s) = T for all hash functions

Else false

F	F	F	F	F	F	F	F	F	F
0	1	2	3	4	5	6	7	8	9

n = 10

Counting

- How to count the number of occurrences of an element s?

Naive algorithm ?

$\log_2 n$ bits

n floors, K → $n \log_2 \log_2 K$

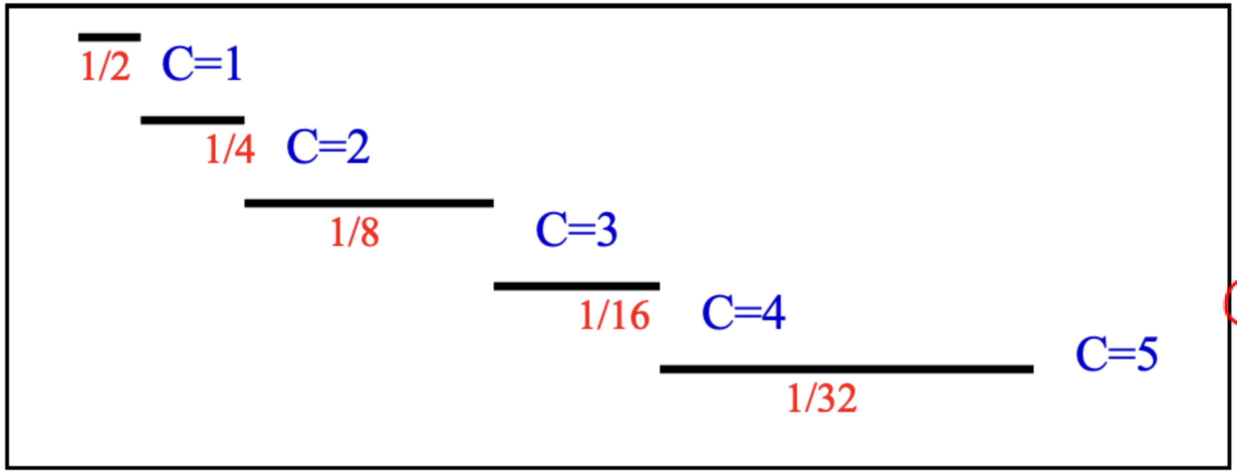
x

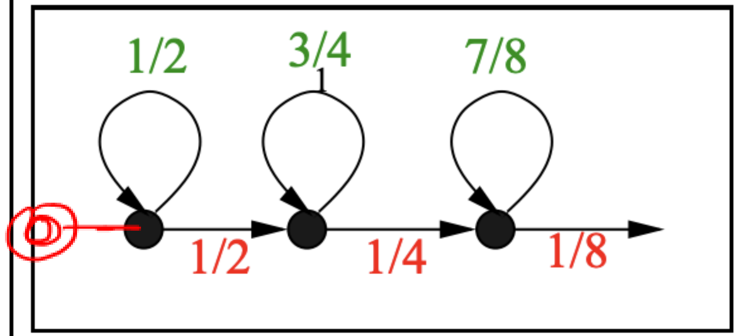
Probabilistic Counting

$\frac{1}{2^x}$

Emulate a counter subject to $X := X + 1$.

Find value of counter was → n





$2^n - 1$

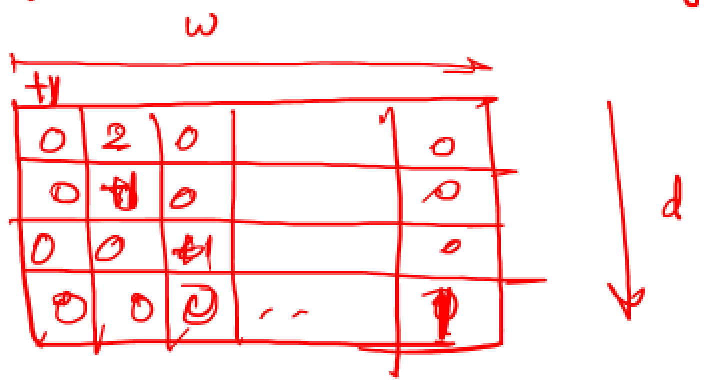
Space: $\log_2 \log_2 n$

$\log_K \log_2 n$

$\frac{1}{k^x}$

Count Min Sketch

d hash functions each of length w



update ($h_j(f_i)$) $\forall j \in \{1, \dots, d\}$

0, 1, 2, $w-1$

Query (f_i)

$h(f_i)$: min ($h_j(f_i)$) $\forall j \in \{1, \dots, d\}$

→ Heavy-hitters

→ Elephant flows (heavy hitters)

→ Mice flow (not important)

$O(d \times w)$

Find Top-K flows

Count Min Sketch

- Two-dimensional array counts with width *w* and depth *d*
- Each row corresponds to a hash function chosen uniformly at random from a pairwise independent family which map vector entry to [1…w].
- **Update:** For an element s, compute the hash values and increment the corresponding entries in the 2-D array
- **Query:** For an element s, compute the hash value and return the min across all values

TRIE h

→ max-heap

Hash function 1 →	0	0	0	0	0
Hash function 2 →	0	0	0	0	0
Hash function 3 →	0	0	0	0	0

Special Topics: Machine Learning (ML) for Networking

COL867

Holi, 2025

Network Telemetry

Tarun Mangla

Recap: Network Telemetry

- Conventional telemetry is bottom up. High overhead
- Top-down telemetry or query-driven telemetry
- **Challenge**: Can’t run arbitrary queries, even on programmable data plane
  - Memory and compute bottleneck
- **Solution (for memory bottleneck)**
  - Use sketches – memory-efficient data structures for summarizing the data
  - Example sketches: bloom filter, probabilistic counting, count min

Implement the following telemetry using sketches

Exercise 2: Count the number of unique sources accessing a web service via port 80

Implement the following telemetry using sketches

Exercise 2: Have you already seen a given unique source *s* accessing your web service via port 80

Bloom filter

Filter on port 80 :

Implement the following telemetry using sketches

Exercise 3: Count the number of unique sources accessing a web service via port 80

Exercise

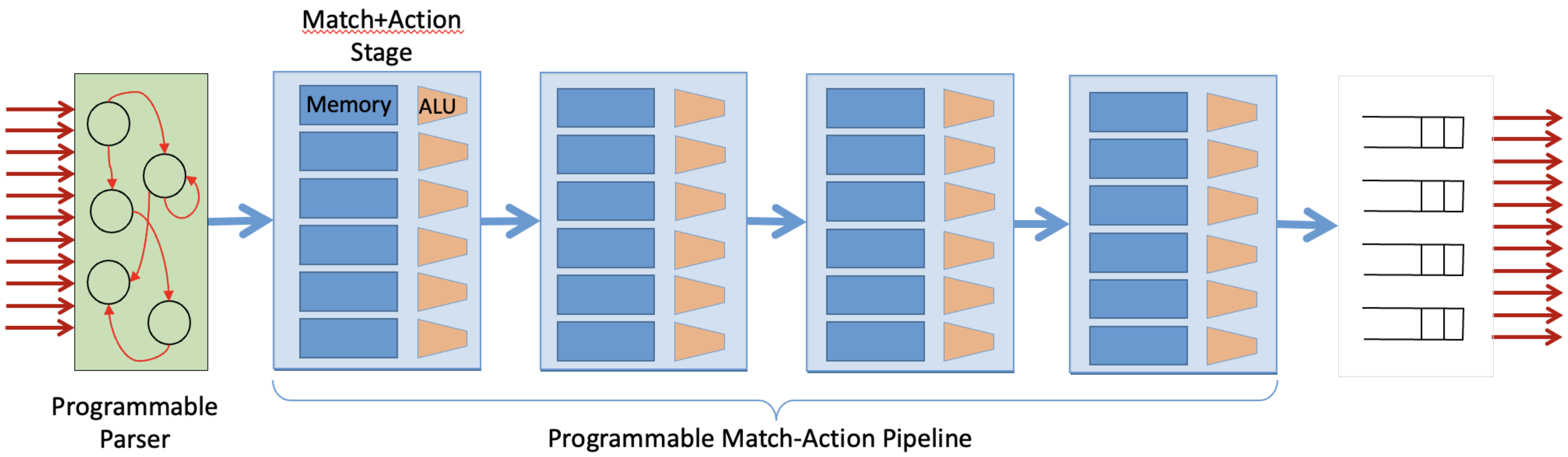
Exercise 4: Identify the heavy hitter flows, i.e., flows that consume more than a fraction T of the link capacity during a time interval

How to implement the sketch in a data plane?

What are the fundamental operations needed for sketch?

- Hashing
- Match
- Probabilistic counter

Programmable Data Plane Architecture

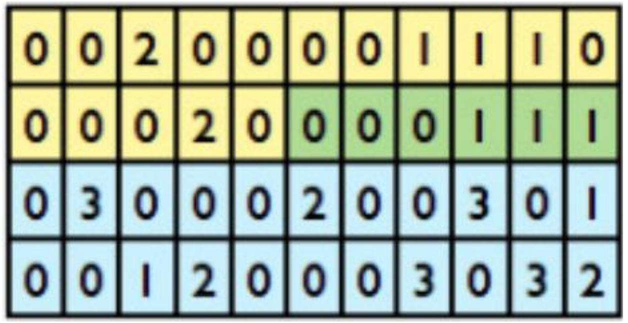


Key functionalities

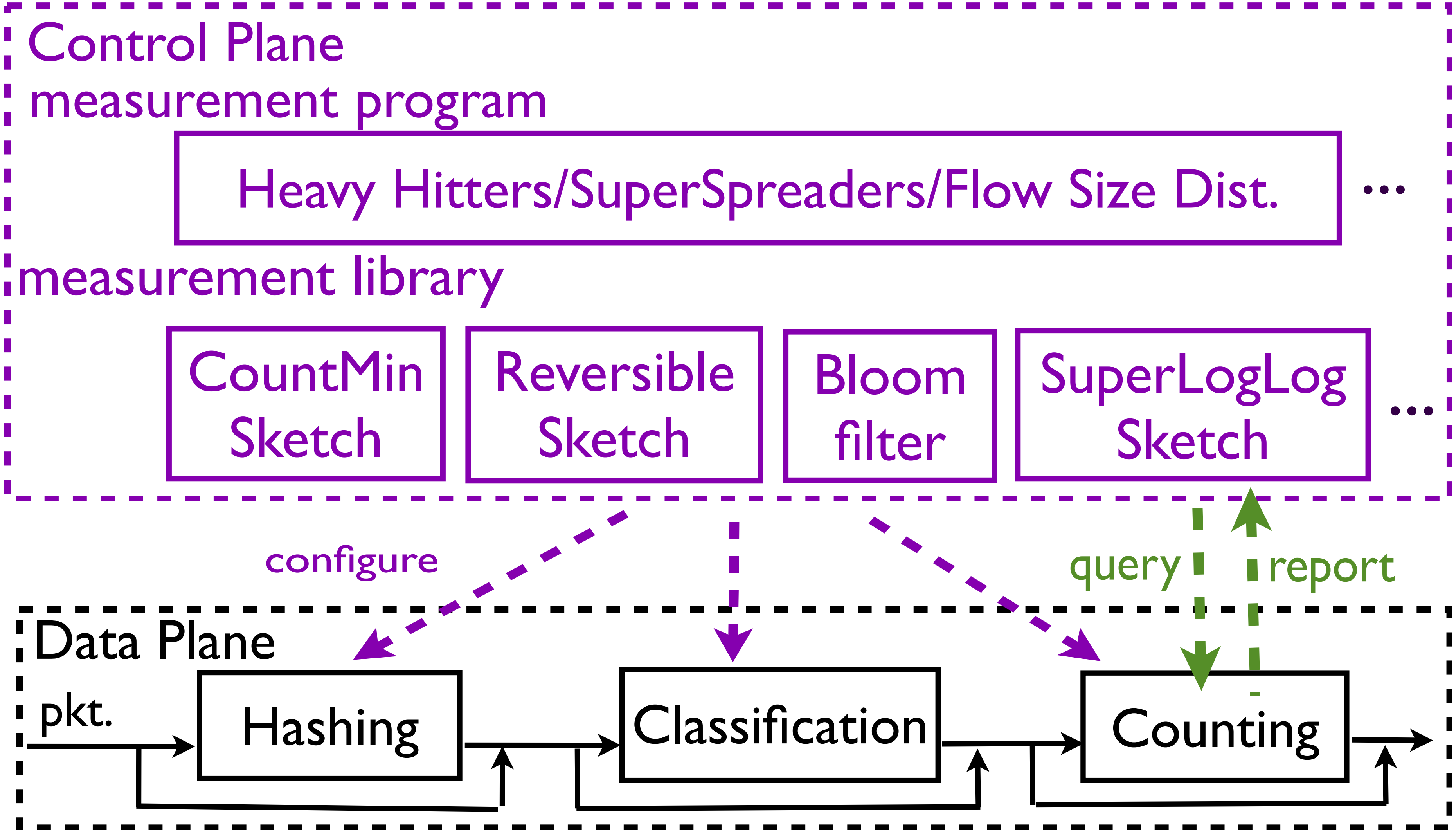
- Programmable parser, extract specific fields of interest
- Matching, to group related packets into a single ‘flow’
- Computation using simple ALUs (addition, subtraction, bitwise operations)
- Storage of information across successive packets using a small number of register arrays
- Communication with a software controller

Build on Existing Switch Components

- A few simple hash functions
  - 4-8 three-wise or five-wise independent hash functions
  - Leverage traffic diversity to approx. truly random func.
- A few TCAM entries for classification
  - Match on both packets and hash values
  - Avoid matching on individual micro-flow entries
- Flexible counters in SRAM
  - Logical tables with flexible indexing
  - Access counters by addresses



OpenSketch Architecture



Discussion

- What about performance metrics like latency?
  - Nice survey paper: Compact Data Structures for Network Telemetry

- What is the next logical step?