In-Network Coflow Scheduling

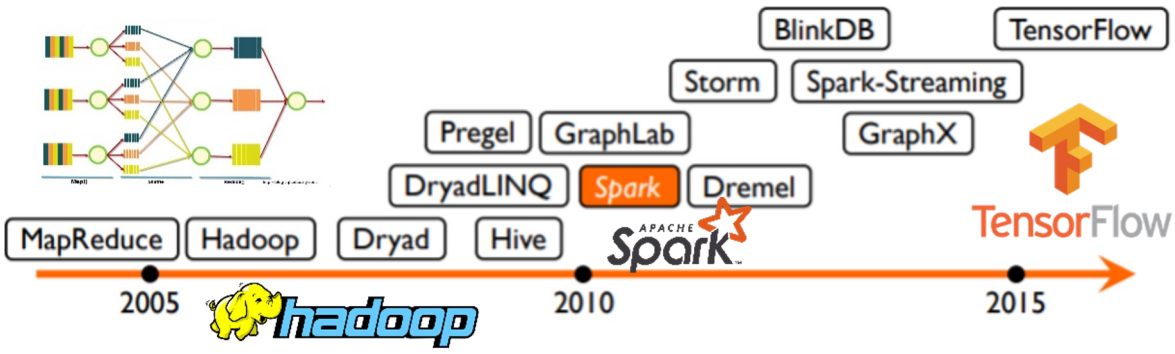
Jin Du and **Kate Ching-Ju Lin**

National Yang-Ming Chiao Tung University

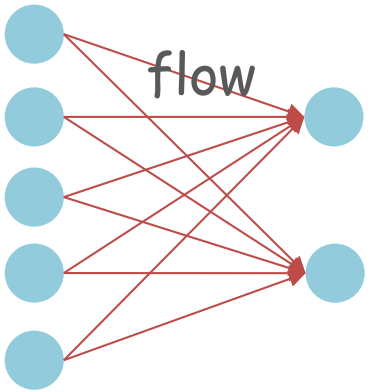
Coflow

- Communication abstraction for data-parallel applications
  - Minimize completion time
  - Meet the deadline
  - Perform fair allocation

Big Datacenters & Data-Parallel Applications



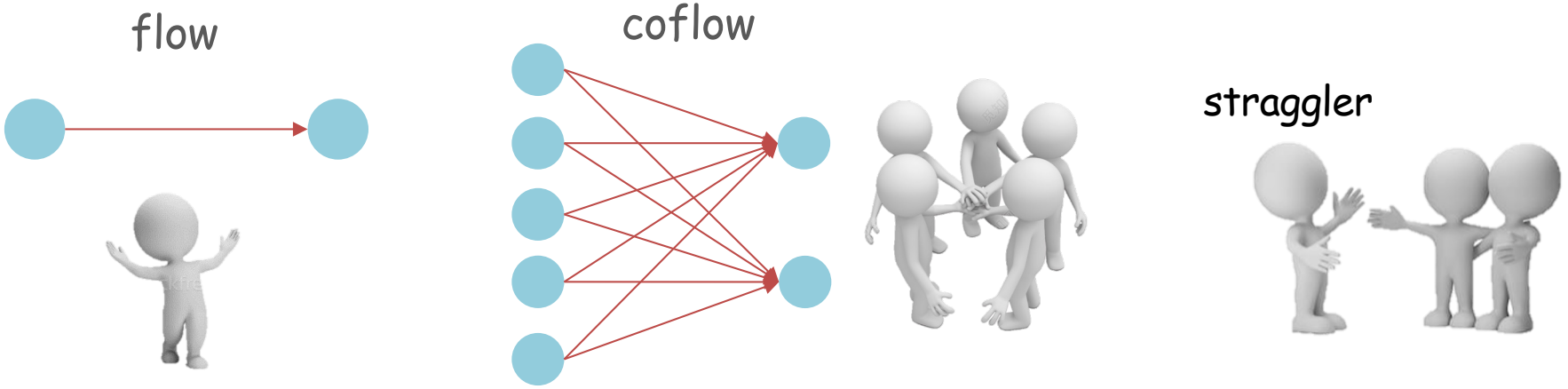
Coflow



1. M. Chowdhury and I. Stoica, Coflow: A Networking Abstraction for Cluster Applications, ACM HotNets, 2012.

Coflow Completion Time (CCT)

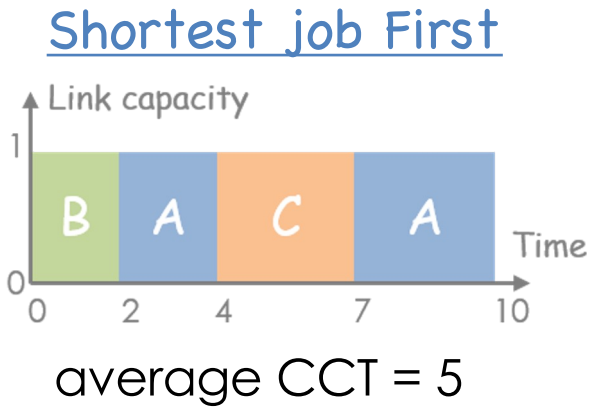
- The completion time of **the last flow** in a coflow
  - Cannot finish until the last flow finishes



How to schedule the coflows to minimize average CCT?

Coflow Scheduling Algorithms

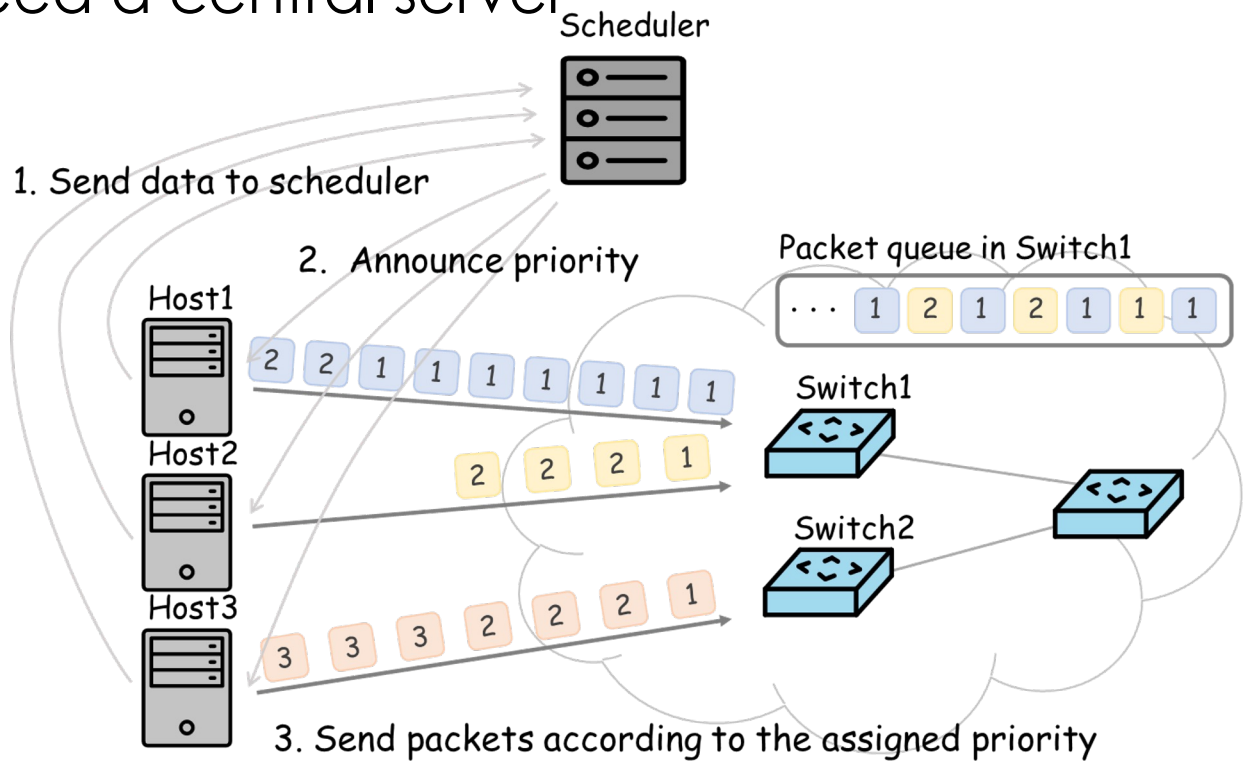
Coflow ID	Arrival Time	Size
A	0	5
B	0	2
C	4	3



- ***Shortest Job First*** (***SJF***) assigns the smallest coflow the highest priority
- Minimize the average CCT

Host-Assisted Coflow Scheduling

- **Information-aware[1]:** Require knowledge about coflow sizes → impractical for many applications
- **Information-agnostic[2]:** Coflows information is unknown → still need a central server



1. M. Chowdhury et al. Efficient coflow scheduling with Varys. In SIGCOMM. 2014.
2. M. Chowdhury et al. Efficient coflow scheduling without prior knowledge. In SIGCOMM. 2015.

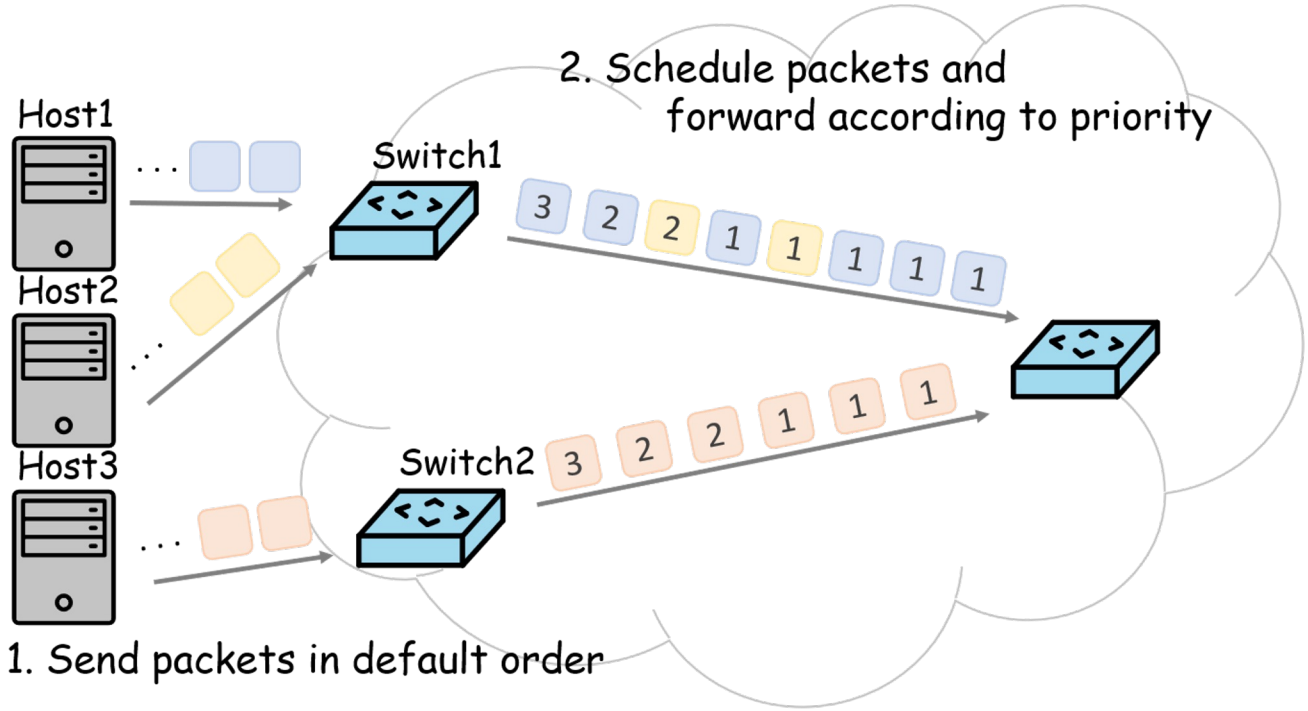
Host-Assisted Coflow Scheduling

- **Information-aware[1]:** Require knowledge about coflow sizes → impractical for many applications
- **Information-agnostic[2]:** Coflows information is unknown

- **High control overhead**
- **Single point failure**
- **Modification of every host**
  - ➢ may not be applicable for public data centers

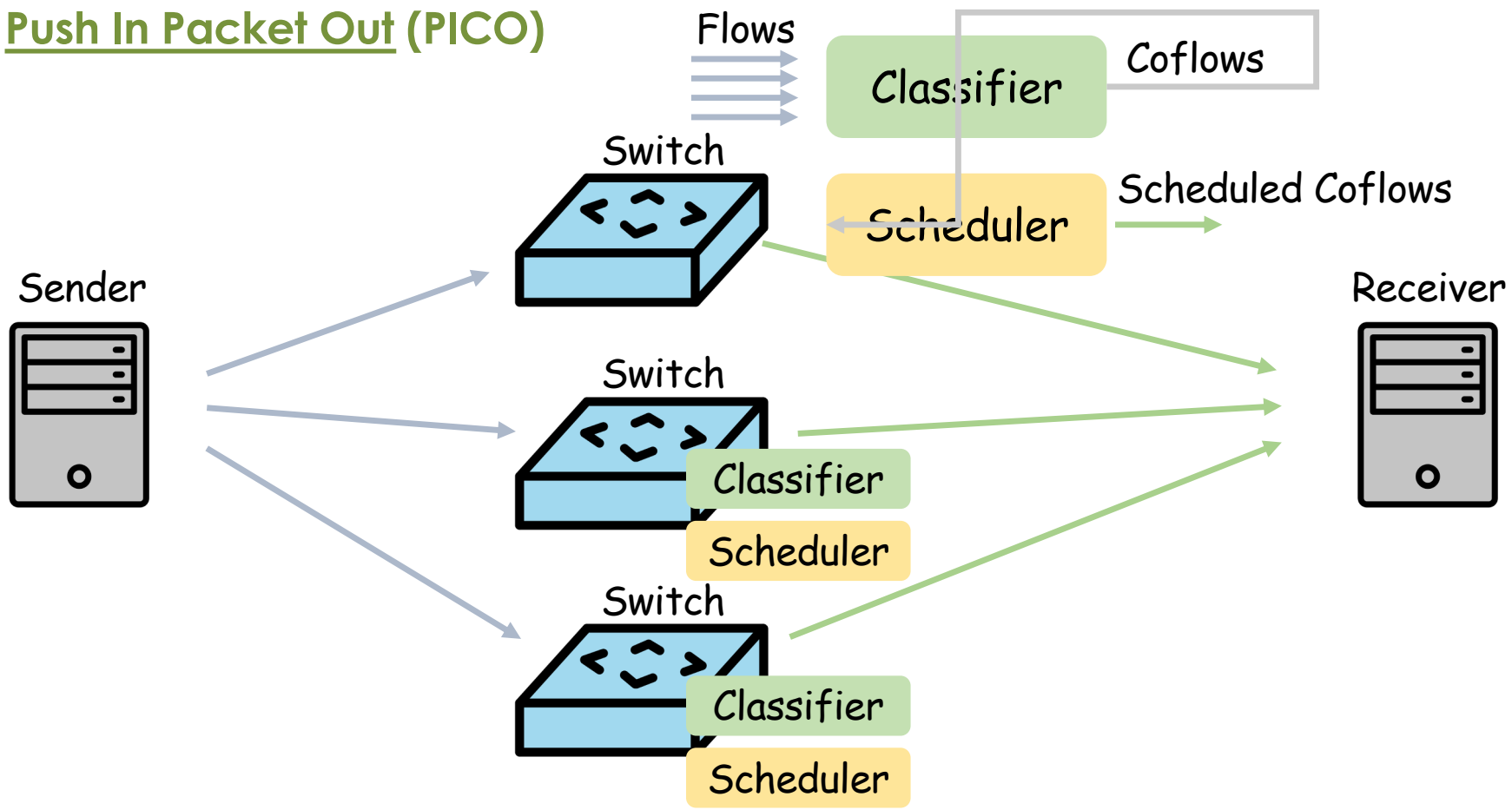
1. M. Chowdhury et al. Efficient coflow scheduling with Varys. In SIGCOMM. 2014.
2. M. Chowdhury et al. Efficient coflow scheduling without prior knowledge. In SIGCOMM. 2015.

In-Network Coflow Scheduling

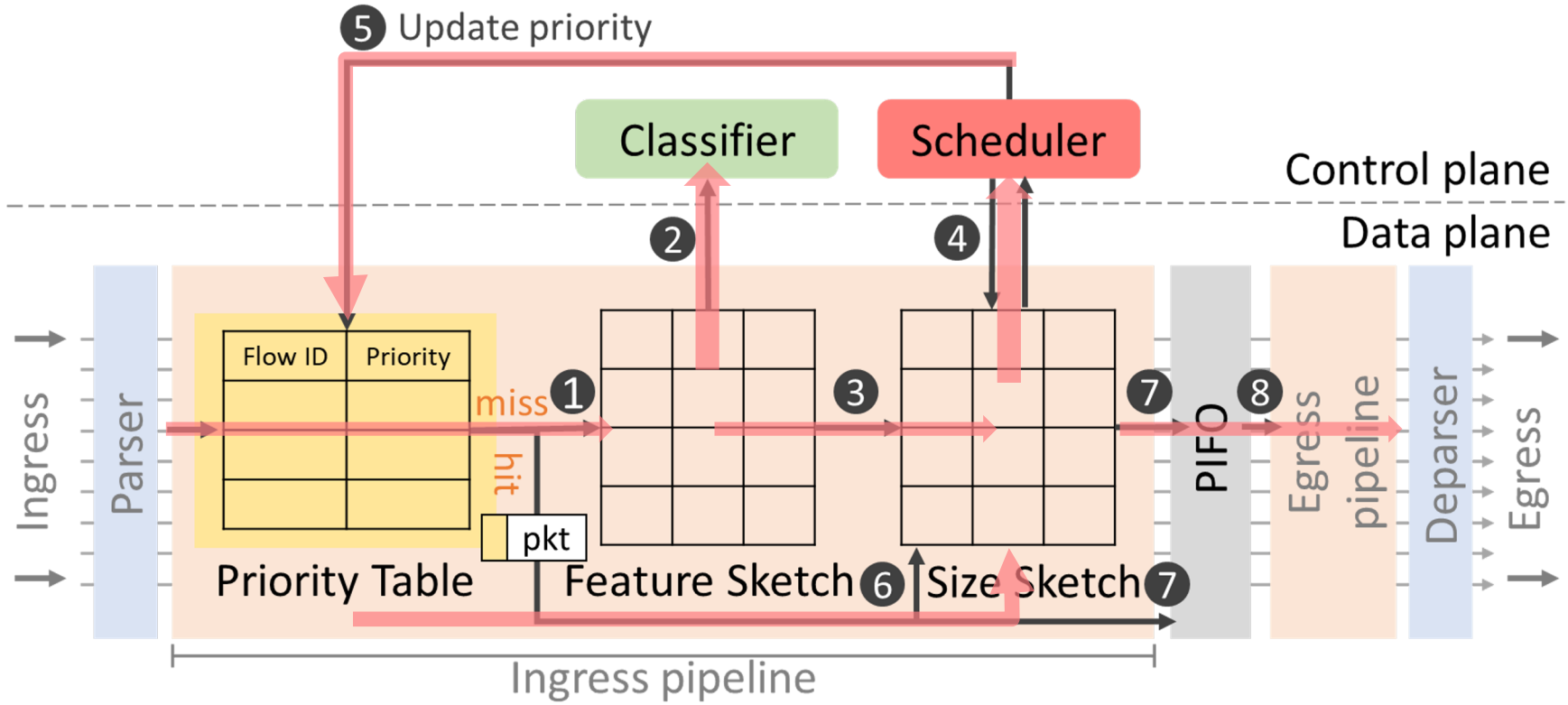


Goal: Identify and schedule coflows
without assistance of controllers and hosts

Overview of PICO



PICO Pipeline



1. **Priority Table**: Match the priority of each flow
2. **Feature Sketch**: Extract flow feature
3. **Size sketch**: Track the coflow size
4. **PIFO**[1]: Order packets based on priority

1. A. Sivaraman et al. Programmable packet scheduling at line rate. In SIGCOMM. 2016.

Challenges

1. How to identify whether arrival flows belong to the same coflow in a distributed manner?
   - Fast in-network coflow detection

2. How to monitor the coflow sizes on the fly?
   - Real-time coflow monitoring and scheduling

PICO Designs

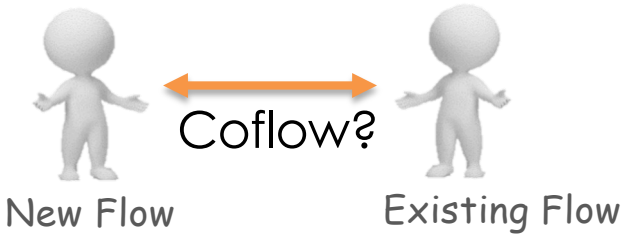
1. How to identify whether arrival flows belong to the same coflow in a distributed manner?
   - Fast in-network coflow detection

   Pairwise Coflow Classification

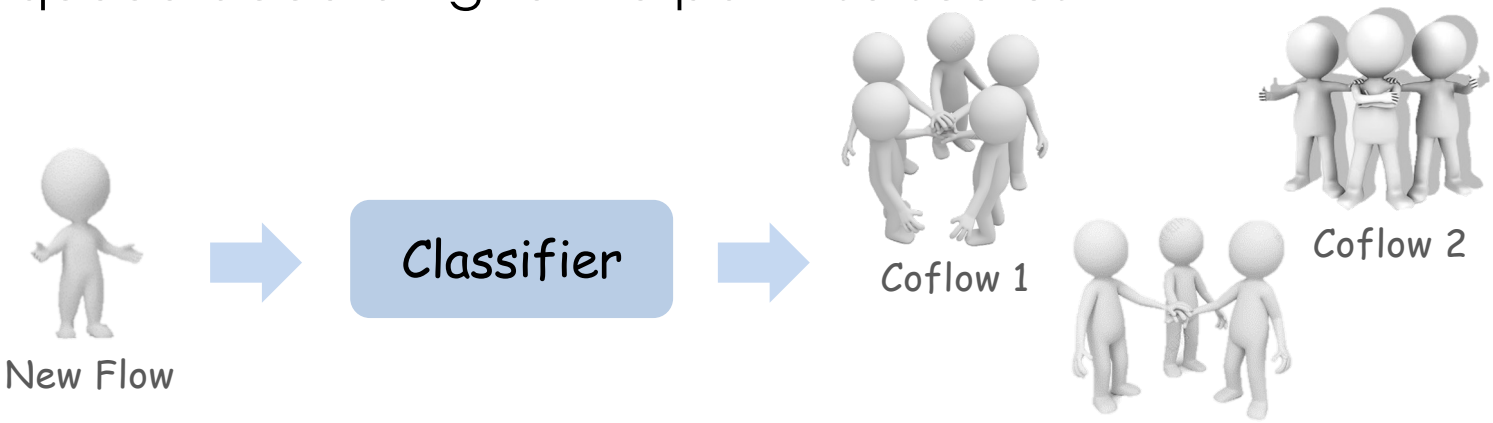
2. How to monitor the coflow sizes on the fly?
   - Real-time coflow monitoring and scheduling

Pairwise Coflow Classification

- **Fast coflow detection**
  - Detect whether any two flows belong to the same coflow
  - Classify whenever a new flow arrives
  - Do not wait until all flows arrive

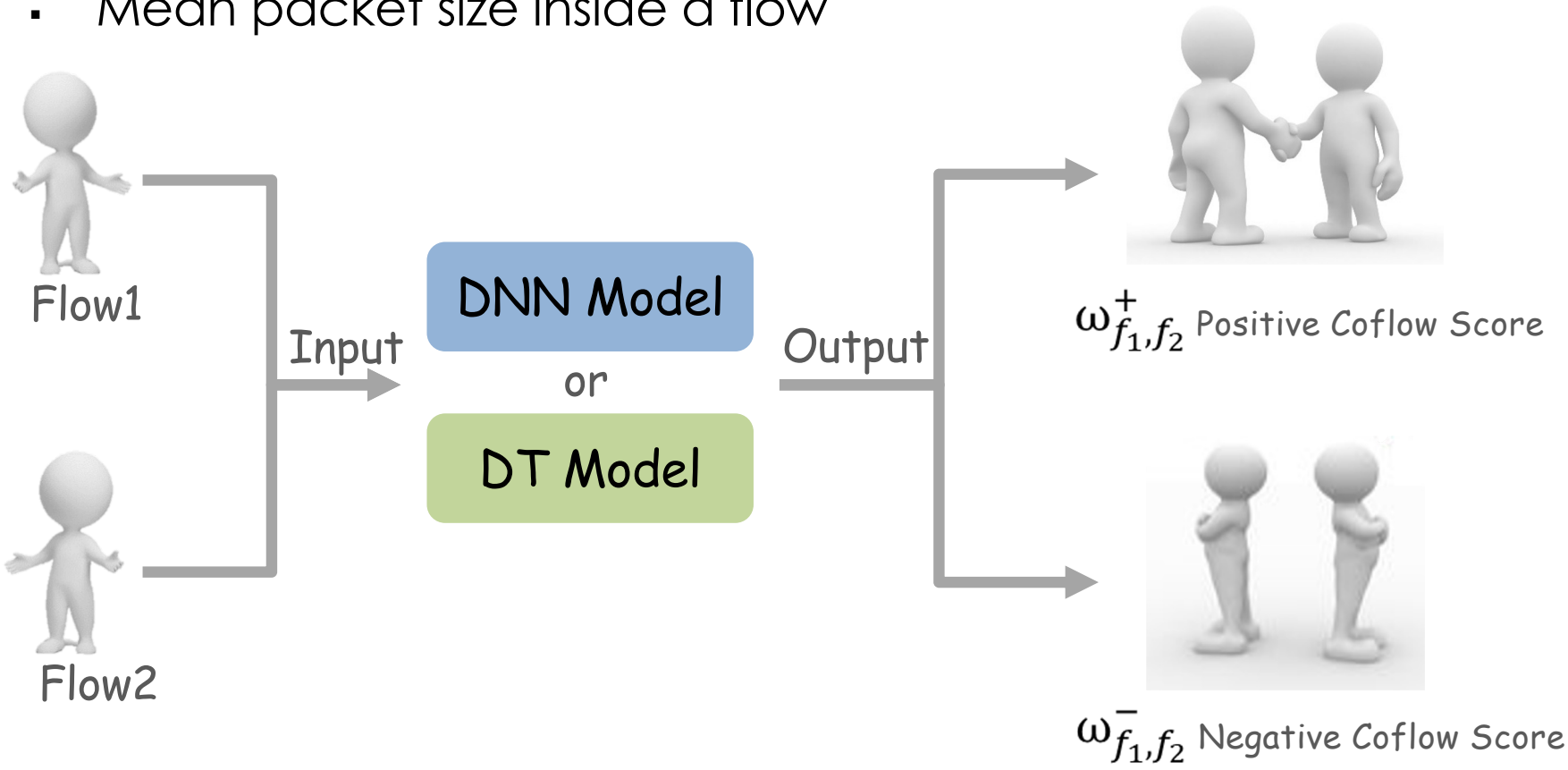


- **Score-based coflow classification**
  - Insert the flows of a coflow ***sequentially*** into the same queue according to the pairwise scores



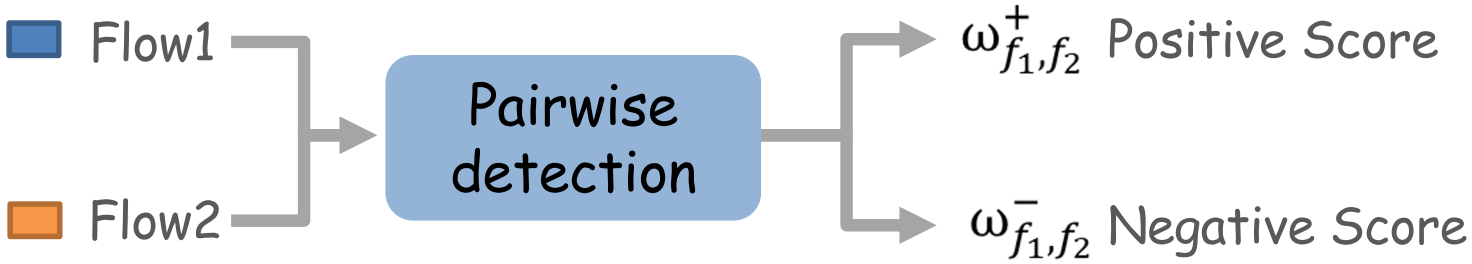
1. Pairwise Coflow Detection

- Extract features from two flows to train DNN or DT
  - Flow start time
  - Mean packet size inside a flow

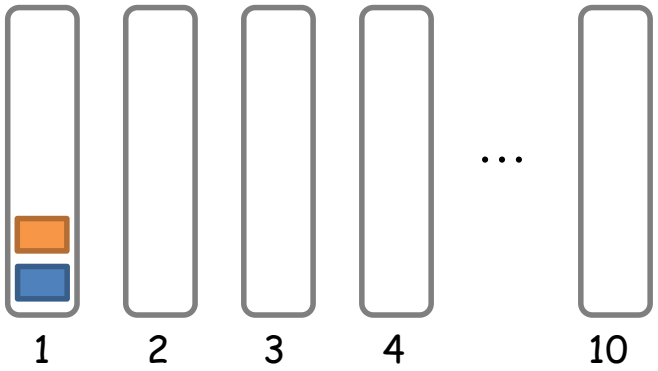


2. Score-based Coflow Classification

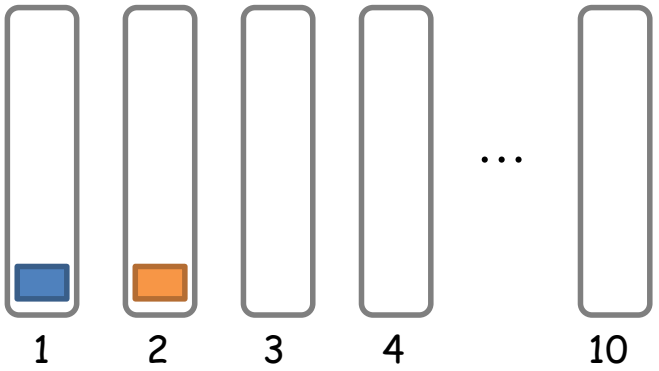
- Compare each newly arrived flow with the flows that have been cached in a queue



Enqueue if $\omega^{+}_{f_1,f_2} \geq \omega^{-}_{f_1,f_2}$

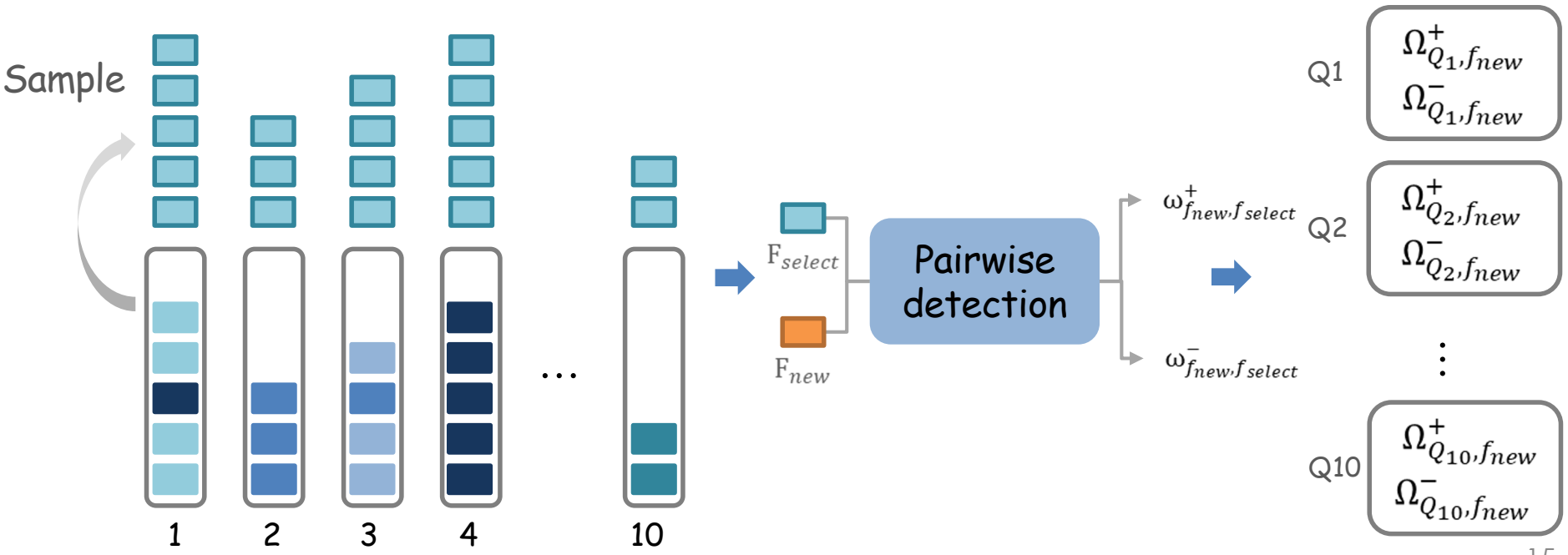


New queue if $\omega^{+}_{f_1,f_2} < \omega^{-}_{f_1,f_2}$



2. Score-based Coflow Classification

- Randomly select at most *k* flows from each queue
  - Calculate the coflow scores w^+ and w^-
- Find the queue with the highest average positive score w^+ and non-negligiable negative score w^-
  - Otherwise, insert it into a new queue or the smallest queue



PICO Designs

1. How to identify whether arrival flows belong to the same coflow in a distributed manner?
   - Fast in-network coflow detection

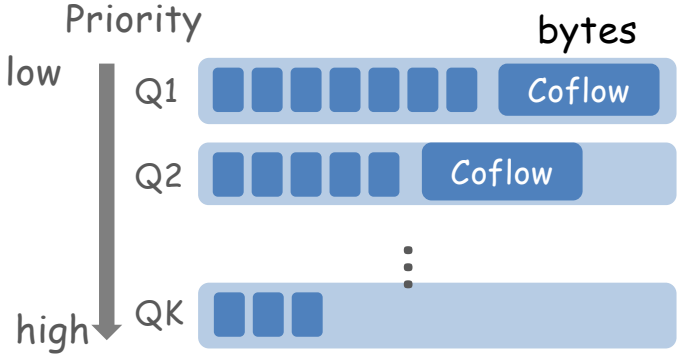
   Pairwise Coflow Classification

2. How to monitor the coflow sizes on the fly?
   - Real-time coflow monitoring and scheduling

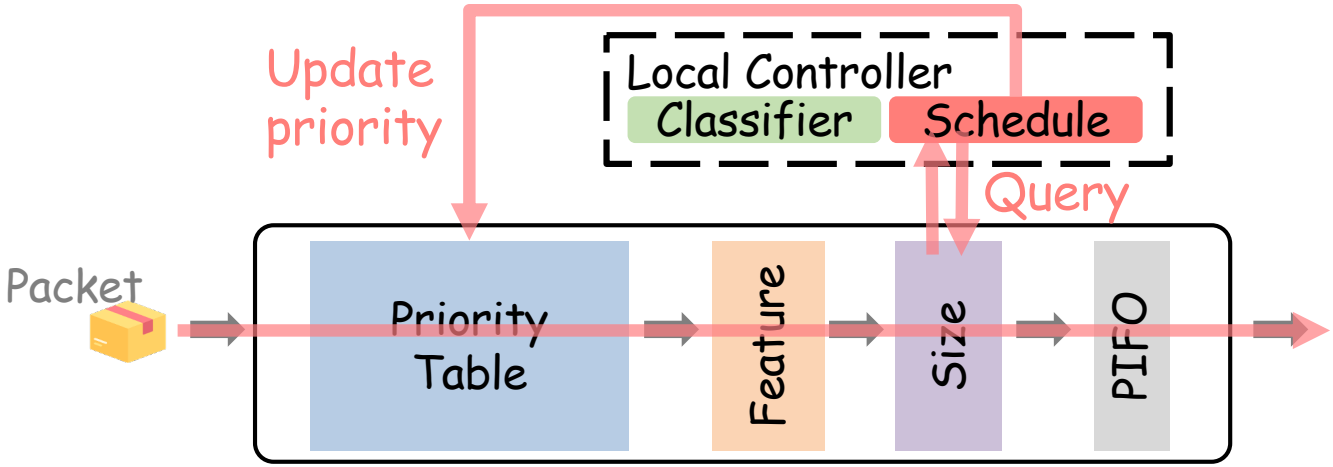
   Coflow Tracking and Scheduling

Scheduler

- **D-CLAS**[1]
  - Prioritize based on number of bytes a coflow **has already sent**
  - A multi-level scheduler, each queue with all the coflows of the same priority

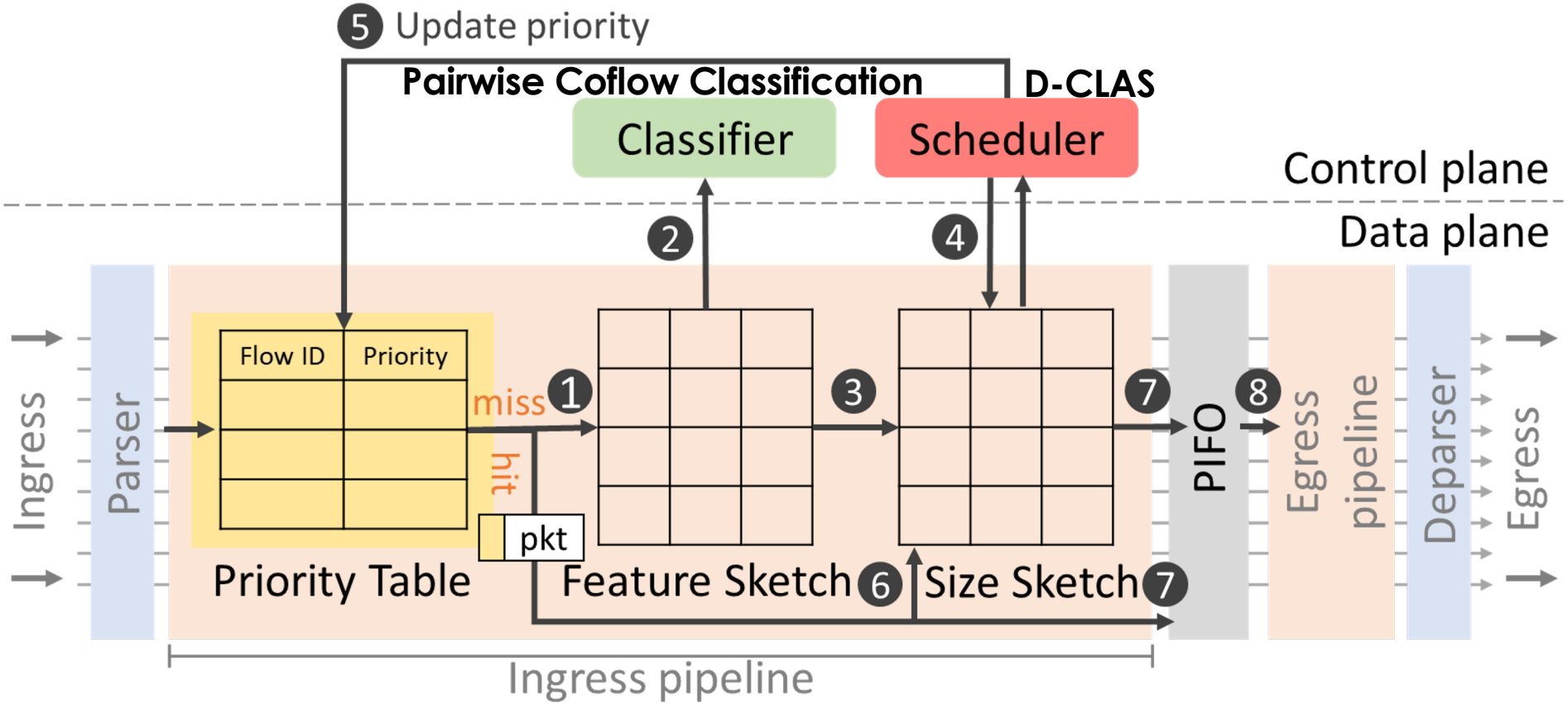


- **Coflow Size Estimation**
  - All packets are redirected to the size sketch
  - Local controller **periodically** queries the size sketch



1. M. Chowdhury et al. Efficient coflow scheduling without prior knowledge. In SIGCOMM. 2015.

Recap


5 Update priority
Pairwise Coflow Classification
D-CLAS
Classifier
Scheduler
Control plane
Data plane
2
4
Ingress
Parser
Flow ID
Priority
miss
1
3
7
8
PIFO
Egress pipeline
Deparser
Egress
hit
pkt
Priority Table
Feature Sketch
6
Size Sketch
7
Ingress pipeline

Performance Evaluation

Experiment Setting

- **Dataset**: Facebook trace
  - 526 coflows (7×10^5 flows)
  - From a 3000-machine 150-rack MapReduce cluster
- **Pairwise detection model**
  - DNN model: 2 hidden layers and 16 neural in each layer
  - Decision Tree: depth of 7 and 99 leaves
- **Comparison Schemes**

Algorithm	Coflow Size	Coflow ID	Scheduling
Ideal (Varys)	V	V	Global
Host (Aalo)	X	V	Global
Host-PD	X	X	Global
PICO	X	X	In-network

Metrics

- Classification
  - Recall, precision, F1 score and accuracy
- Scheduling
  - FI (Factor of Improvement) in CCT

$$\mathrm{FI} = \frac{CCT_{fairness}}{CCT_{other}}$$

- Flow size estimation
  - ARA (Average relative accuracy)

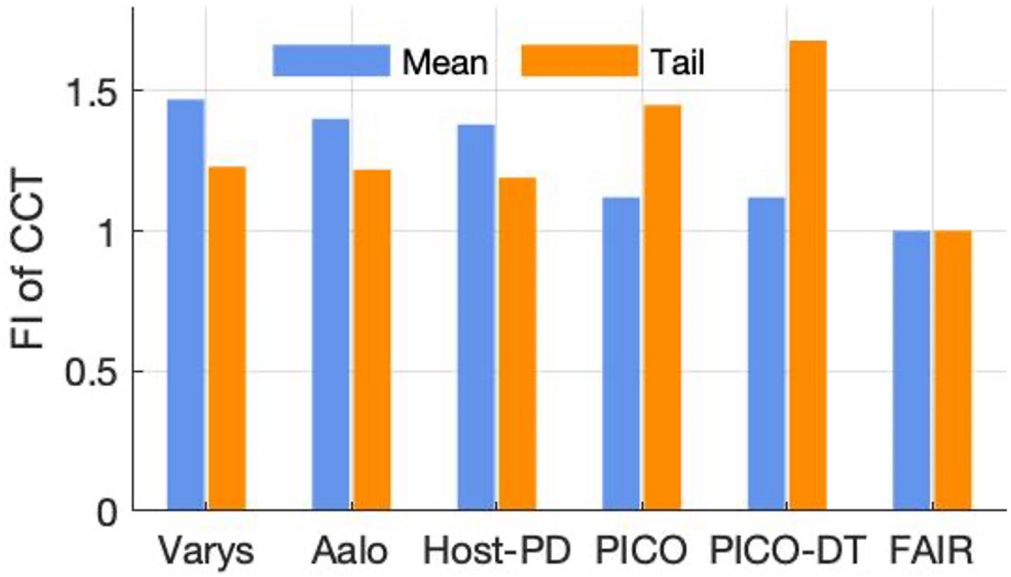
$$ARA = 1 - \frac{1}{n}\sum_{i=1}^{n} \frac{|f_i - \hat{f}_i|}{f_i}$$

Classification Accuracy

	F score	Precision	Recall	Accuracy
PICO	98.9 %	99.4 %	98.4 %	99.7 %
PICO-DT	97.8 %	99.3 %	96.3 %	99.4 %
DKmeans	59.4 %	86.8 %	49.2 %	52.6 %
Kmeans	26.4 %	33.3 %	22.6 %	68.6 %
Host-PD	93.2 %	99.6 %	87.5 %	99.2 %

- PICO achieves accuracy of around 99%, much higher than K-means
- Centralized algorithms perform worse than **sequential distributed algorithms**
  - Interfered by irrelevant data

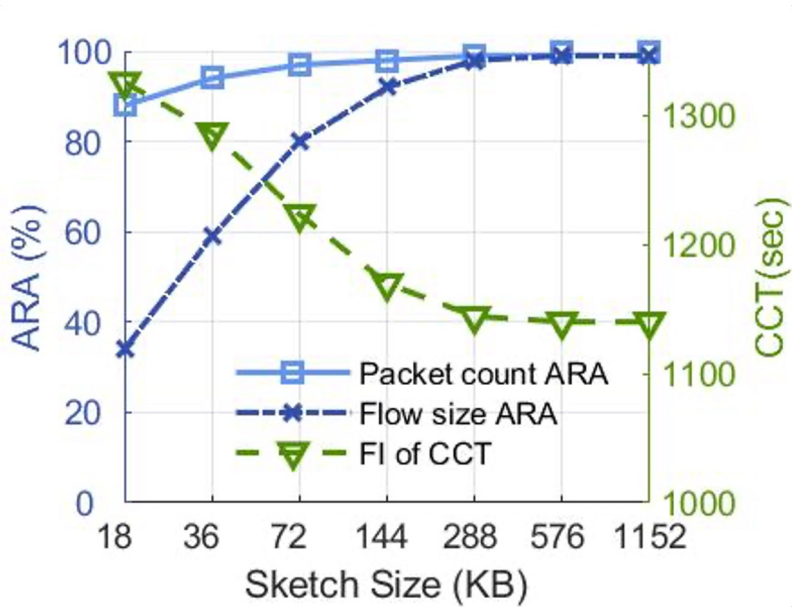
CCT Improvement

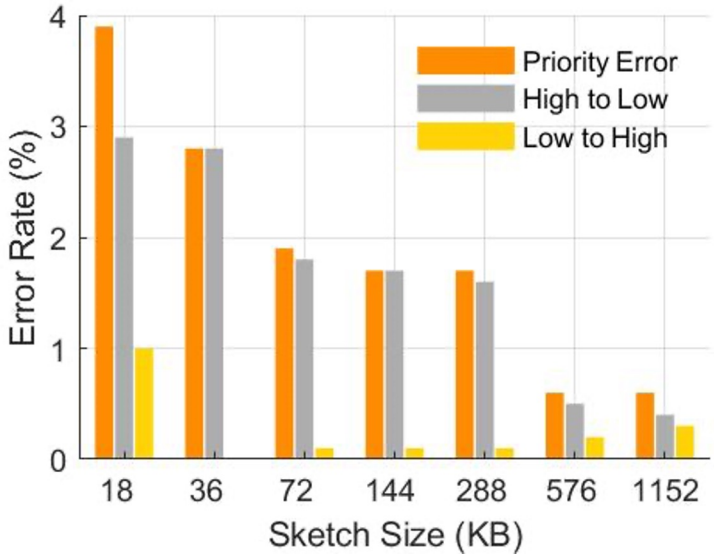


Facebook trace

- Host-PD produces the average CCT just slightly worse than Aalo (with full coflow ID information)
- PICO increases the average CCT slightly but achieves much better tail CCT
  - Smaller errors for large coflows

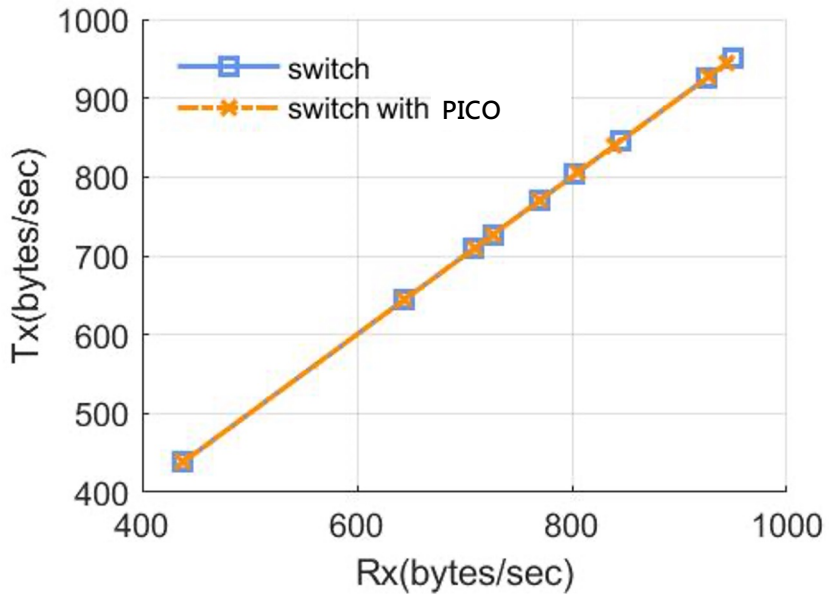
Impact of Size Estimation Errors





- When the memory sizes is sufficiently large, the sketch errors are acceptably small
- Most errors are **"high-to-low"**
  - Larger error for smaller coflows
  - Still assign large coflows the lowest priority

Packet Forwarding Rate



Forwarding rate in Tofino switches

- PICO's pipeline ensures nearly no drop in the forwarding rate

Conclusion

- **PICO**: in-network coflow scheduling system
  - Minimizing CCT without involving a controller and modifying hosts/applications
  - A pairwise coflow classification algorithm that sequentially identifies coflows
  - A data-plane pipeline that enables real-time coflow scheduling on the fly

- Evaluation results show that
  - Pairwise classicization performs no worse than centralized clustering
  - Producing an average CCT only slightly worse than centralized scheduling