Online Identification of Hierarchical Heavy Hitters

Yin Zhang
yzhang@research.att.com

Joint work with

Sumeet Singh Subhabrata Sen
Nick Duffield Carsten Lund

Internet Measurement Conference 2004

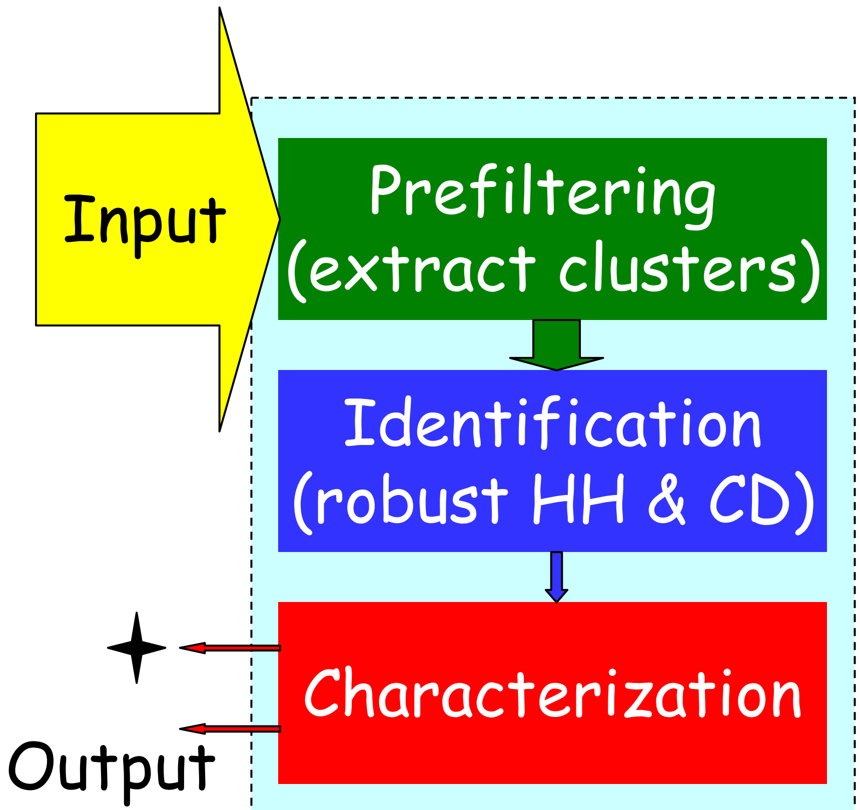
Motivation

- Traffic anomalies are common
  - DDoS attacks, Flash crowds, worms, failures
- Traffic anomalies are complicated
  - Multi-dimensional → may involve multiple header fields
    - E.g. src IP 1.2.3.4 AND port 1214 (KaZaA)
    - Looking at individual fields separately is not enough!
  - Hierarchical → Evident only at specific granularities
    - E.g. 1.2.3.4/32, 1.2.3.0/24, 1.2.0.0/16, 1.0.0.0/8
    - Looking at fixed aggregation levels is not enough!
- Want to identify anomalous traffic aggregates automatically, accurately, in near real time
  - Offline version considered by Estan et al. [SIGCOMM03]

Challenges

- Immense data volume (esp. during attacks)
  - Prohibitive to inspect all traffic in detail
- Multi-dimensional, hierarchical traffic anomalies
  - Prohibitive to monitor all possible combinations of different aggregation levels on all header fields
- Sampling (packet level or flow level)
  - May wash out some details
- False alarms
  - Too many alarms = info "snow" → simply get ignored
- Root cause analysis
  - What do anomalies really mean?

Approach

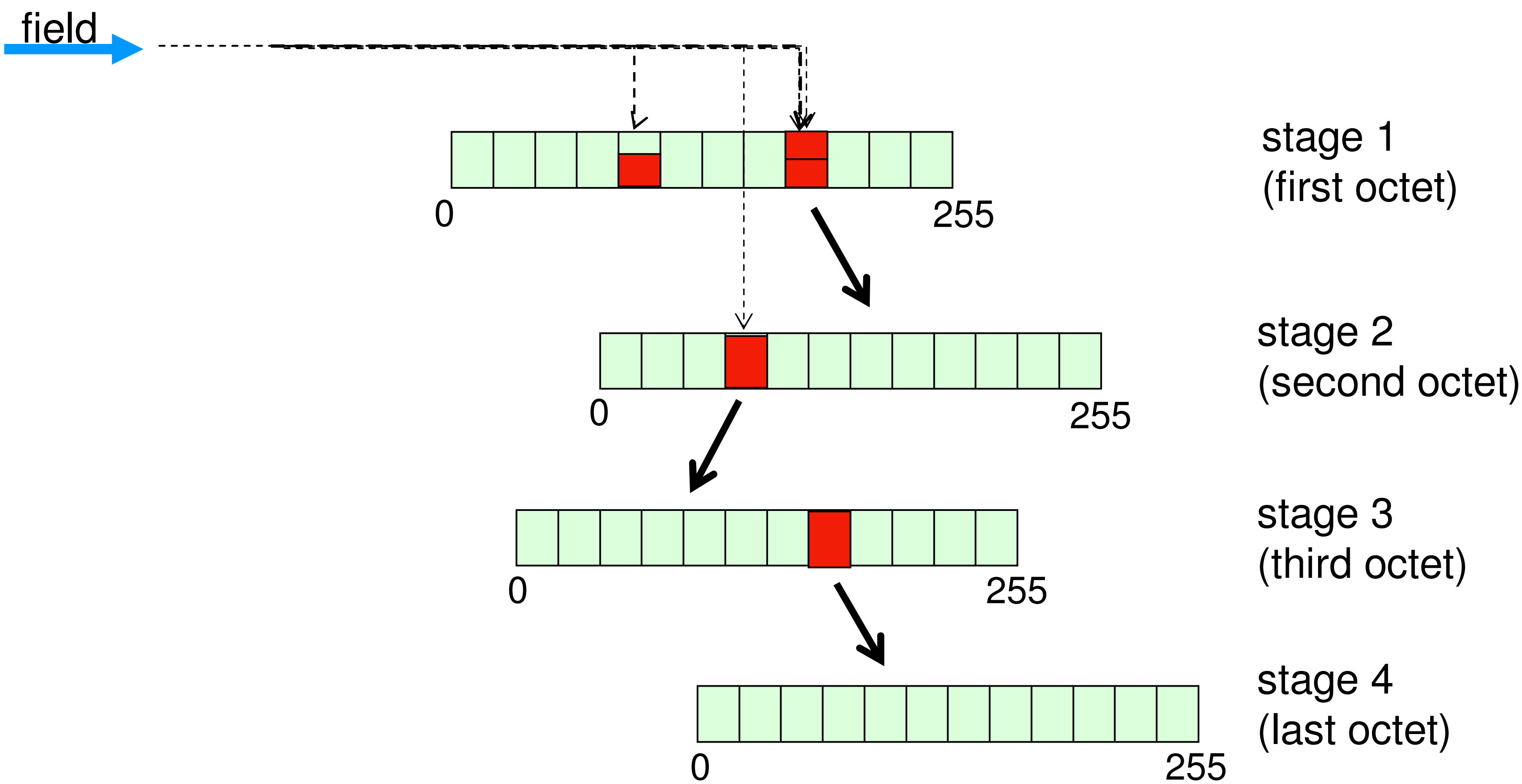


- Prefiltering extracts multi-dimensional hierarchical traffic clusters
  - Fast, scalable, accurate
  - Allows dynamic drilldown
- Robust heavy hitter & change detection
  - Deals with sampling errors, missing values
- Characterization (ongoing)
  - Reduce false alarms by correlating multiple metrics
  - Can pipe to external systems

Prefiltering

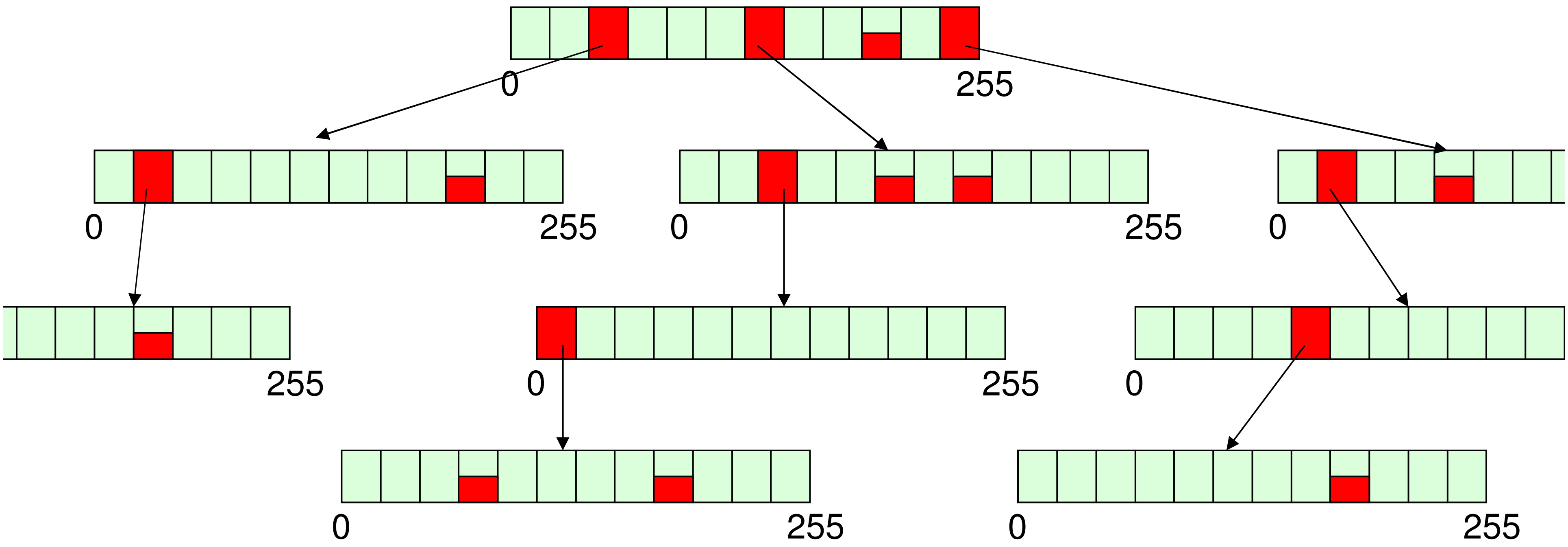
- Input
  - <src_ip, dst_ip, src_port, dst_port, proto>
  - Bytes (we can also use other metrics)
- Output
  - All traffic clusters with volume above (epsilon * total_volume)
    - (cluster ID, estimated volume)
  - Traffic clusters: defined using combinations of IP prefixes, port ranges, and protocol
- Goals
  - Single Pass
  - Efficient (low overhead)
  - Dynamic drilldown capability

Dynamic Drilldown via 1-DTrie



- At most 1 update per flow
- Split level when adding new bytes causes bucket >= T_{split}
- Invariant: traffic trapped at any interior node < T_{split}

1-D Trie Data Structure



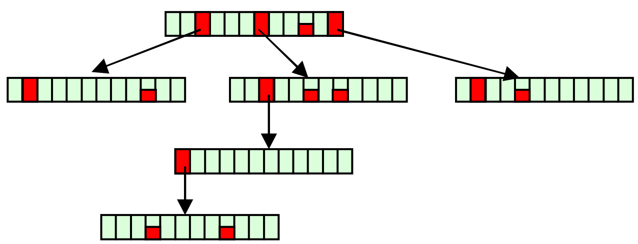
- Reconstruct interior nodes (aggregates) by summing up the children
- Reconstruct missed value by summing up traffic trapped at ancestors
- Amortize the update cost

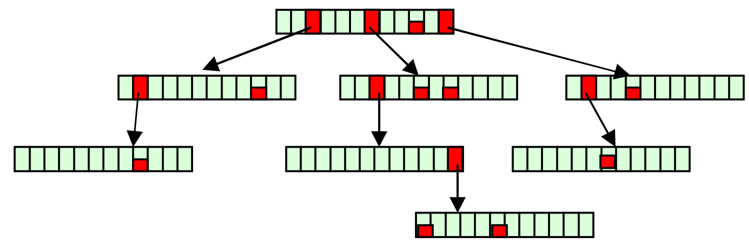
1-D Trie Performance

- Update cost
  - 1 lookup + 1 update
- Memory
  - At most $1/T_{split}$ internal nodes at each level
- Accuracy: For any given $T > d*T_{split}$
  - Captures all flows with metric >= T
  - Captures no flow with metric < $T-d*T_{split}$

Extending 1-D Trie to 2-D: Cross-Producting

Update(k1, k2, value)





p1 = f(k1)

p2 = f(k2)

totalBytes{ p1, p2 } += value

- In each dimension, find the deepest interior node (prefix): (p1, p2)
  - Can be done using longest prefix matching (LPM)
- Update a hash table using key (p1, p2):
  - Hash table: cross product of 1-D interior nodes
- Reconstruction can be done at the end

Cross-Producting Performance

- Update cost:
  - 2 X (1-D update cost) + 1 hash table update.
- Memory
  - Hash table size bounded by $(d/T_{split})^2$
  - In practice, generally much smaller
- Accuracy: For any given $T > d*T_{split}$
  - Captures all flows with metric >= T
  - Captures no flow with metric < $T- d*T_{split}$

HHH vs. Packet Classification

- Similarity
  - Associate a rule for each node
    → finding fringe nodes becomes PC
- Difference
  - PC: rules given a priori and mostly static
  - HHH: rules generated on the fly via dynamic drilldown
- Adapted 2 more PC algorithms to 2-D HHH
  - Grid-of-tries & Rectangle Search
  - Only require $O(d/T_{split})$ memory
- Decompose 5-D HHH into 2-D HHH problems

Change Detection

- Data
  - 5 minute reconstructed cluster series
  - Can use different interval size
- Approach
  - Classic time series analysis
- Big change
  - Significant departure from forecast

Change Detection: Details

- Holt-Winters
  - Smooth + Trend + (Seasonal)
    - Smooth: Long term curve
    - Trend: Short term trend (variation)
    - Seasonal: Daily / Weekly / Monthly effects
  - Can plug in your favorite method
- Joint analysis on upper & lower bounds
  - Can deal with missing clusters
    - ADT provides upper bounds (lower bound = 0)
  - Can deal with sampling variance
    - Translate sampling variance into bounds

Evaluation Methodology

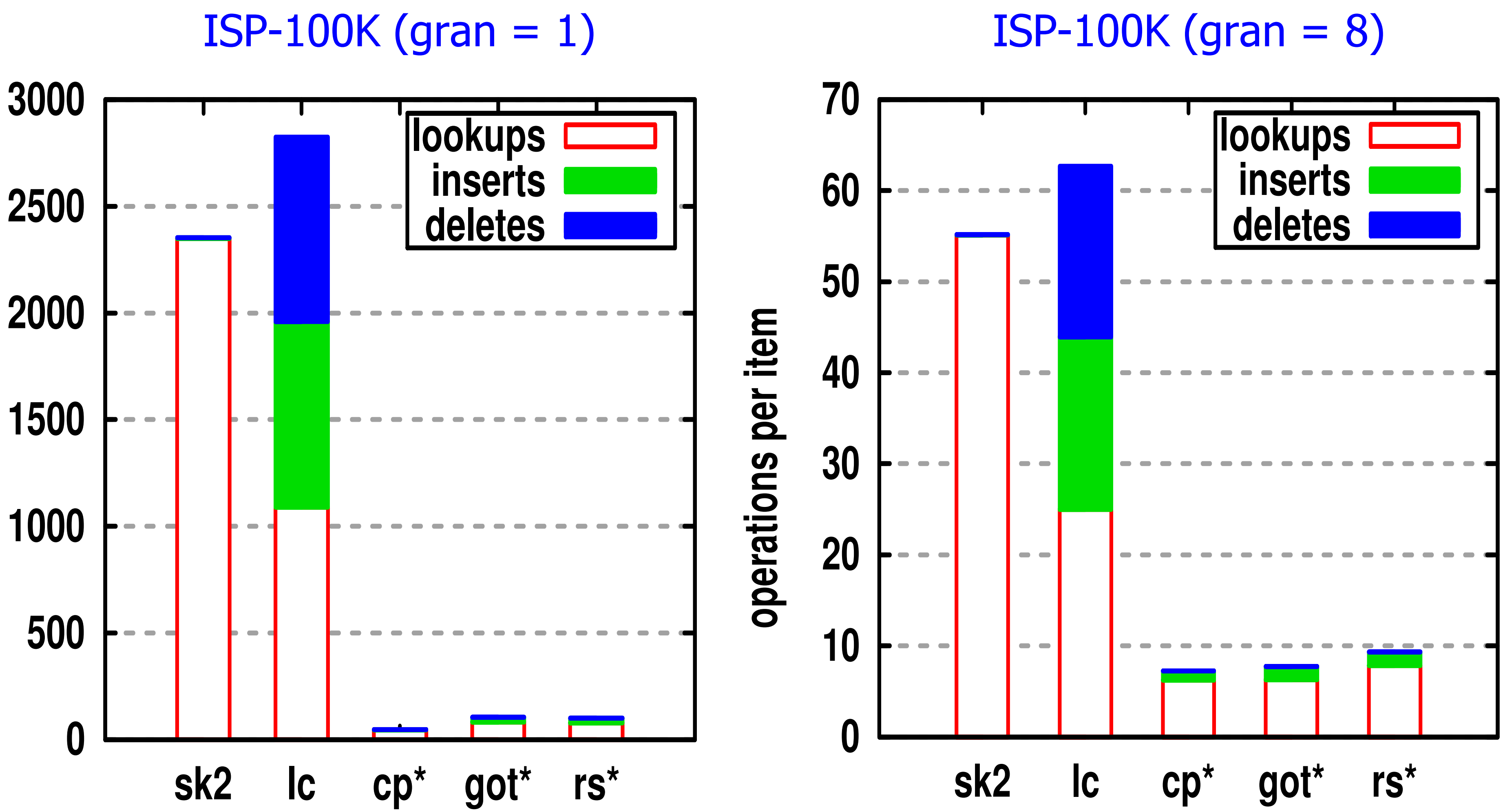
- Dataset description
  - Netflow from a tier-1 ISP

Trace	Duration	#routers	#records	Volume
ISP-100K	3 min	1	100 K	66.5 MB
ISP-1day	1 day	2	332 M	223.5 GB
ISP-1mon	1 month	2	7.5 G	5.2 TB

- Algorithms tested

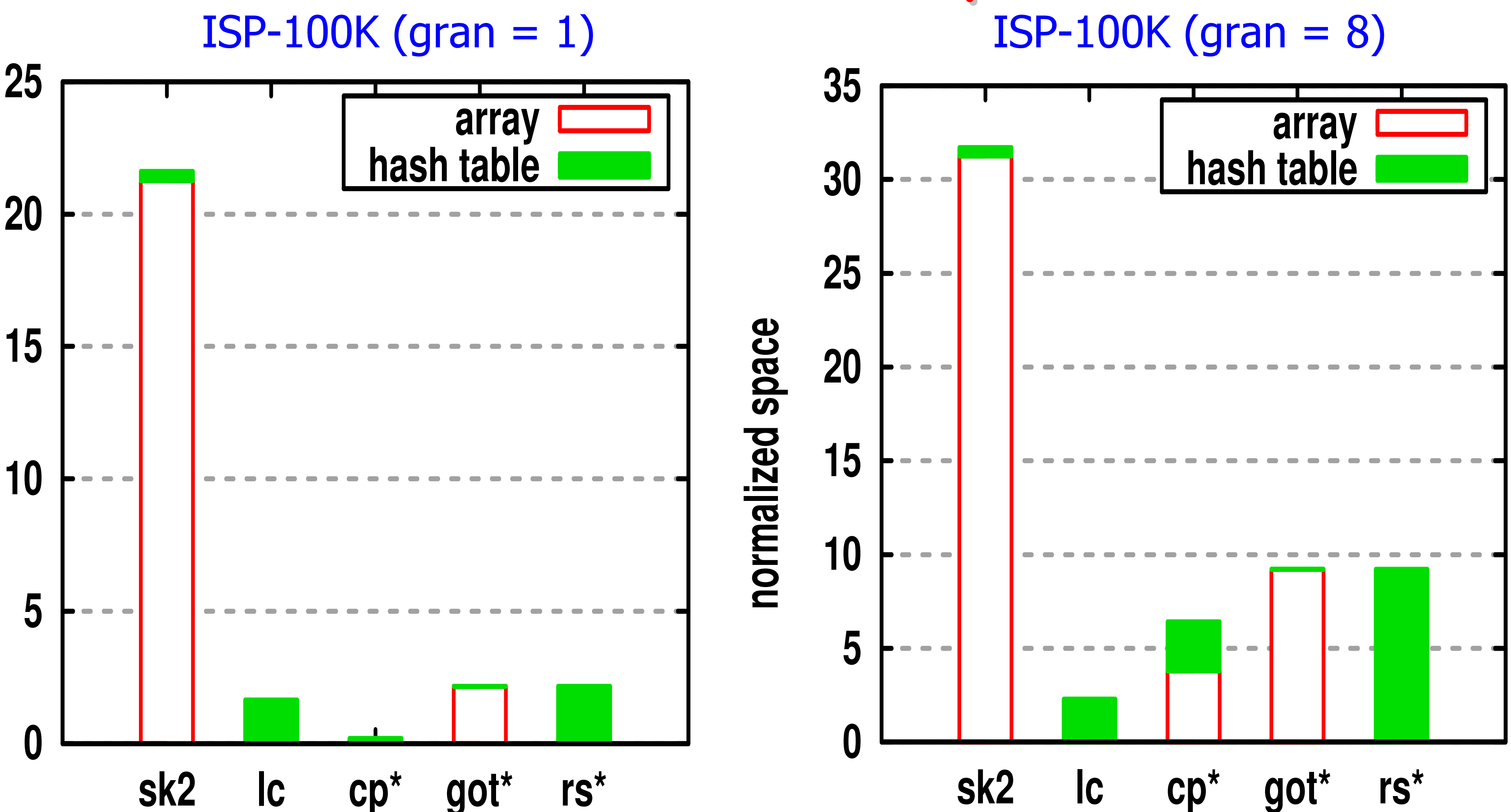
Baseline (Brute-force)	Sketch	sk, sk2
	Lossy Counting	lc
Our algorithms	Cross-Producting	cp
	Grid-of-tries	got
	Rectangle Search	rs

Runtime Costs



We are an order of magnitude faster

Normalized Space

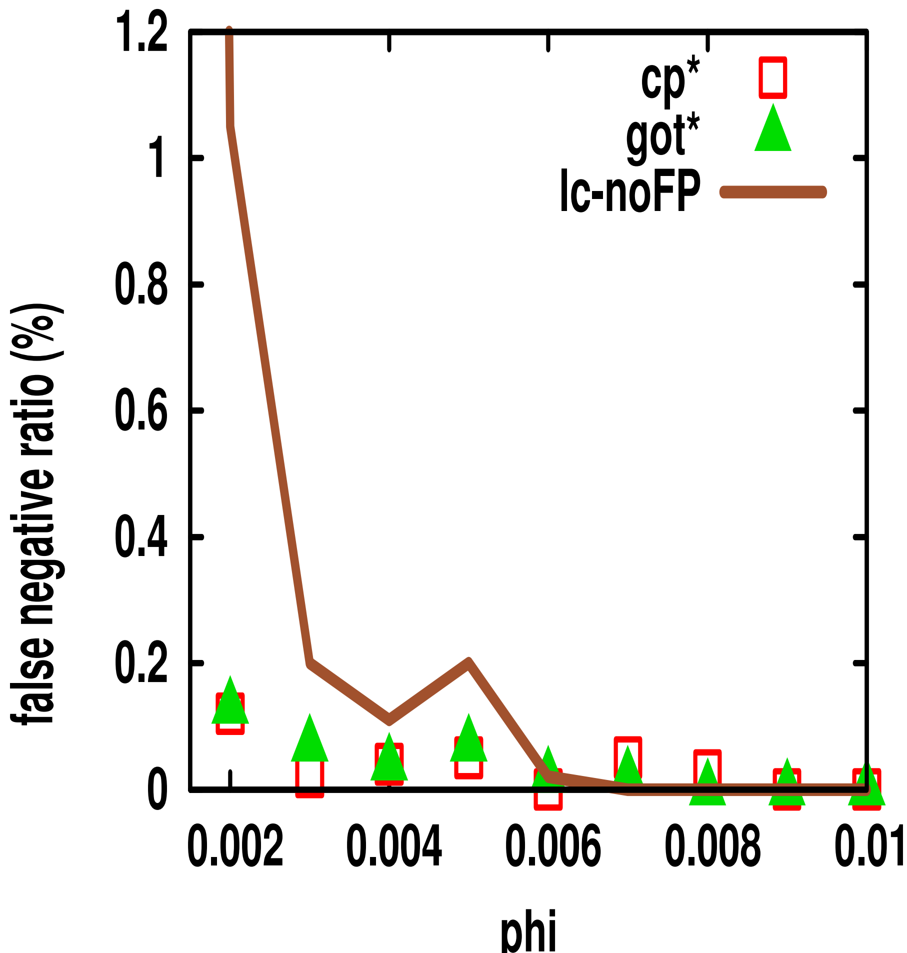


normalized space = actual space / [(1/epsilon) (32/gran)^2]

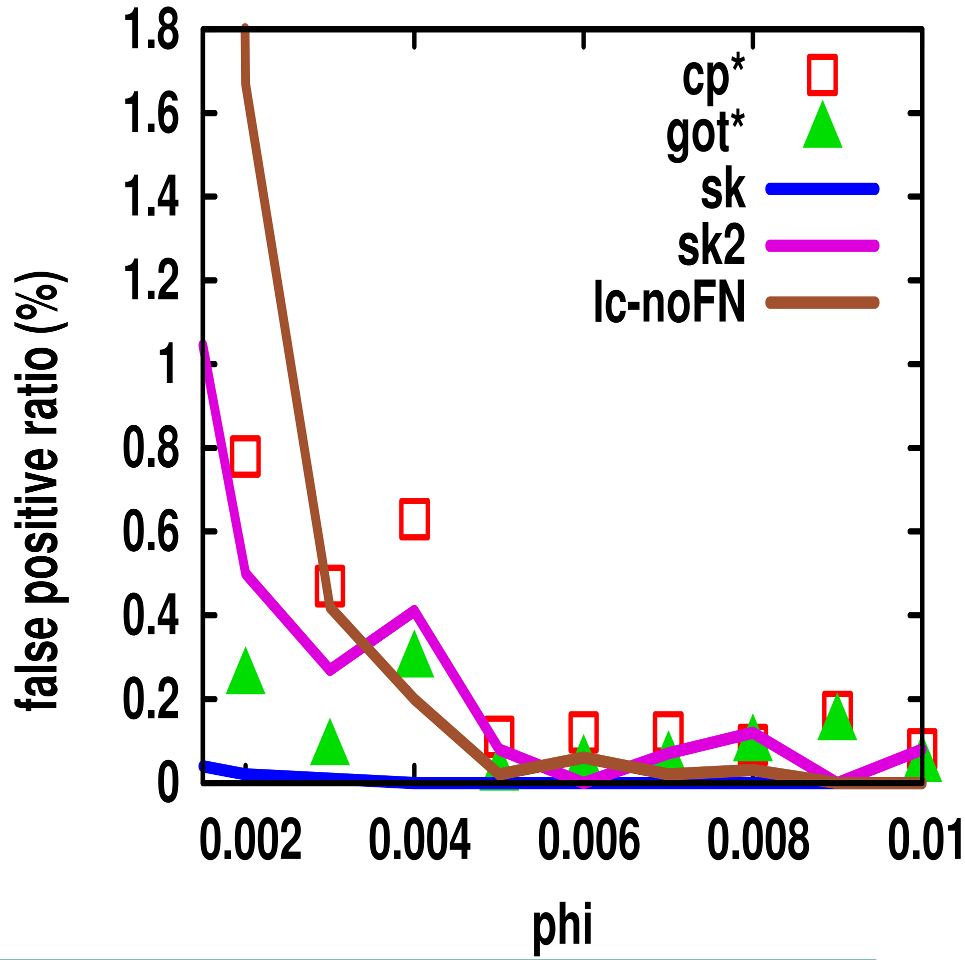
gran = 1: we need less / comparable space
gran = 8: we need more space but total space is small

HHH Accuracy

False Negatives (ISP-100K)

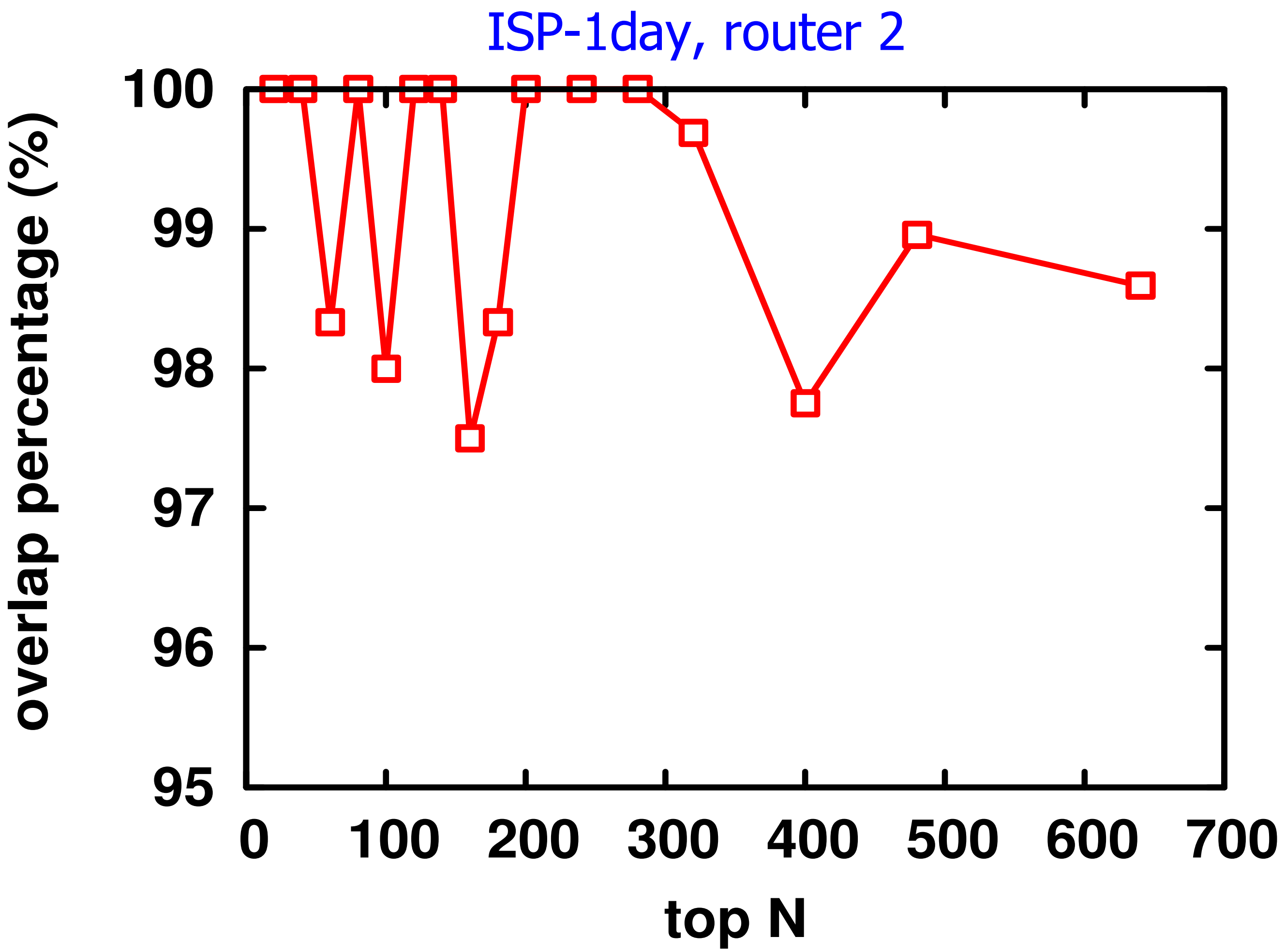


False Positives (ISP-100K)



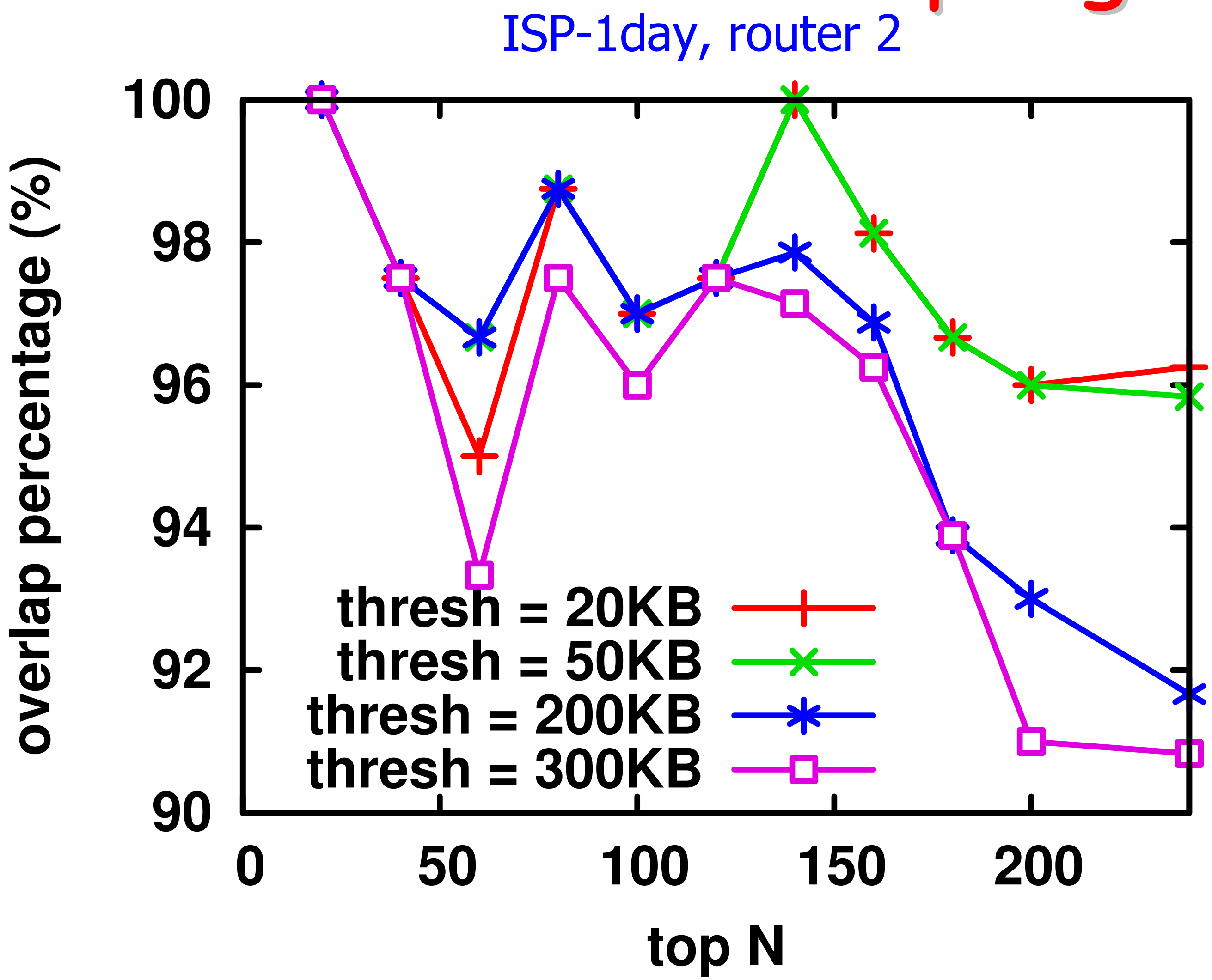
HHH detection accuracy comparable to brute-force

Change Detection Accuracy



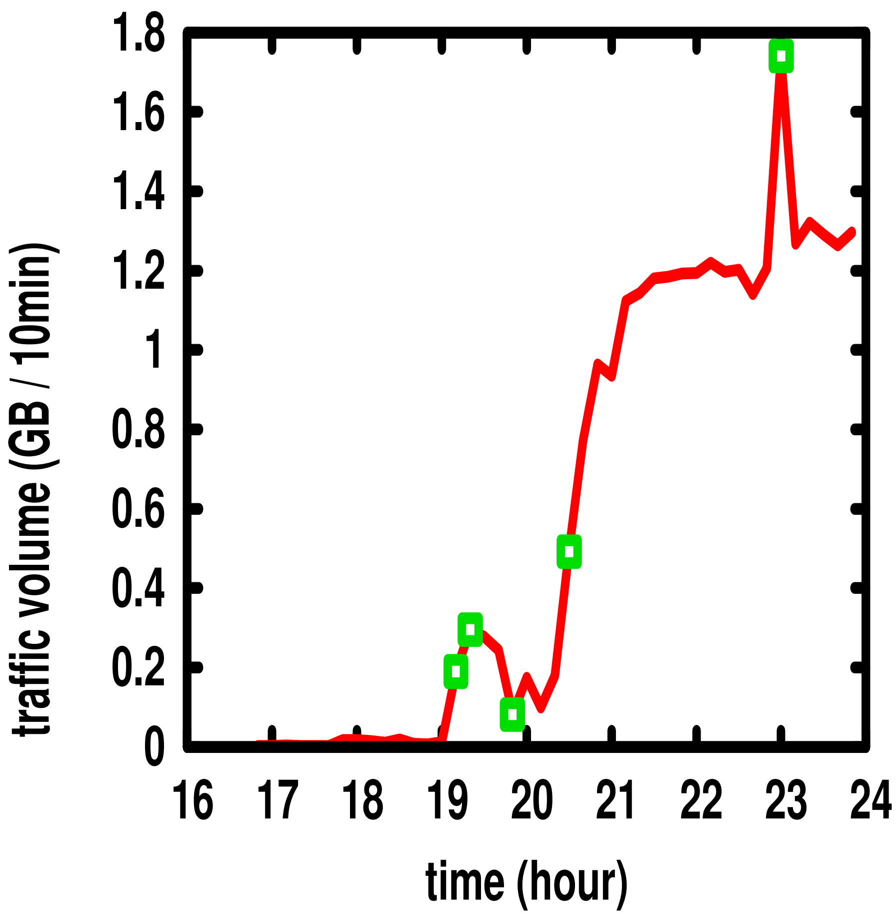
Top N change overlap is above 97% even for very large N

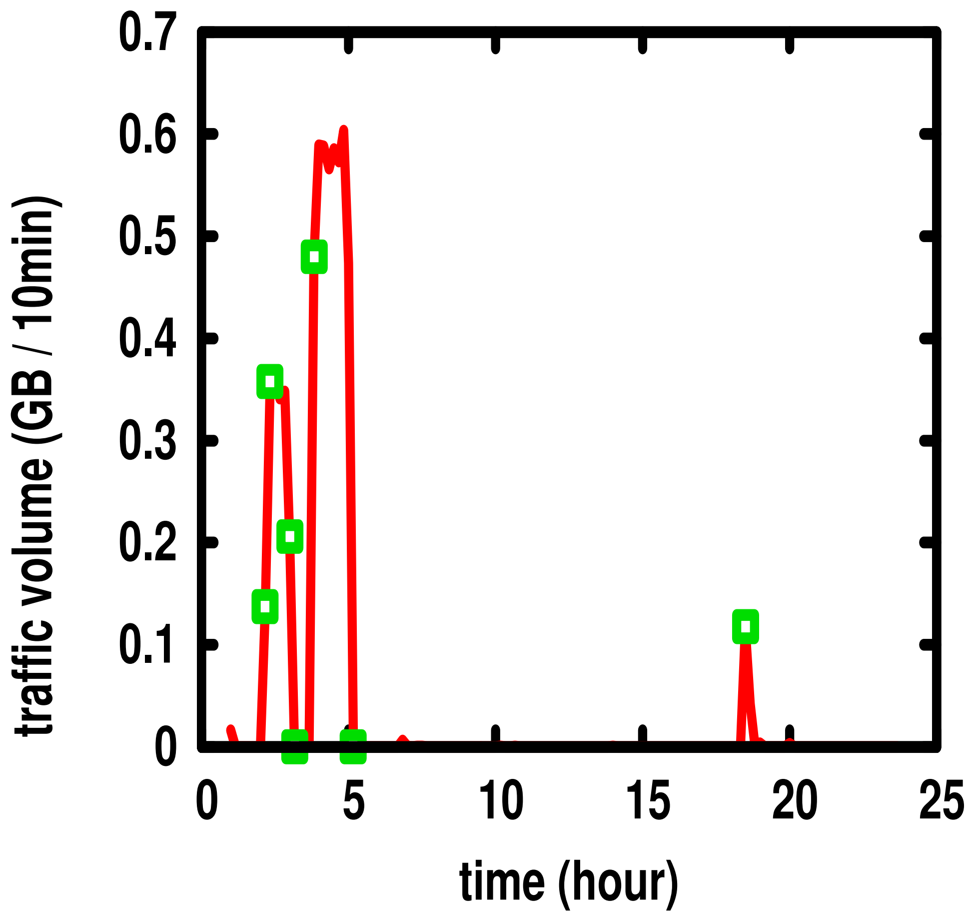
Effects of Sampling



Accuracy above 90% with 90% data reduction

Some Detected Changes





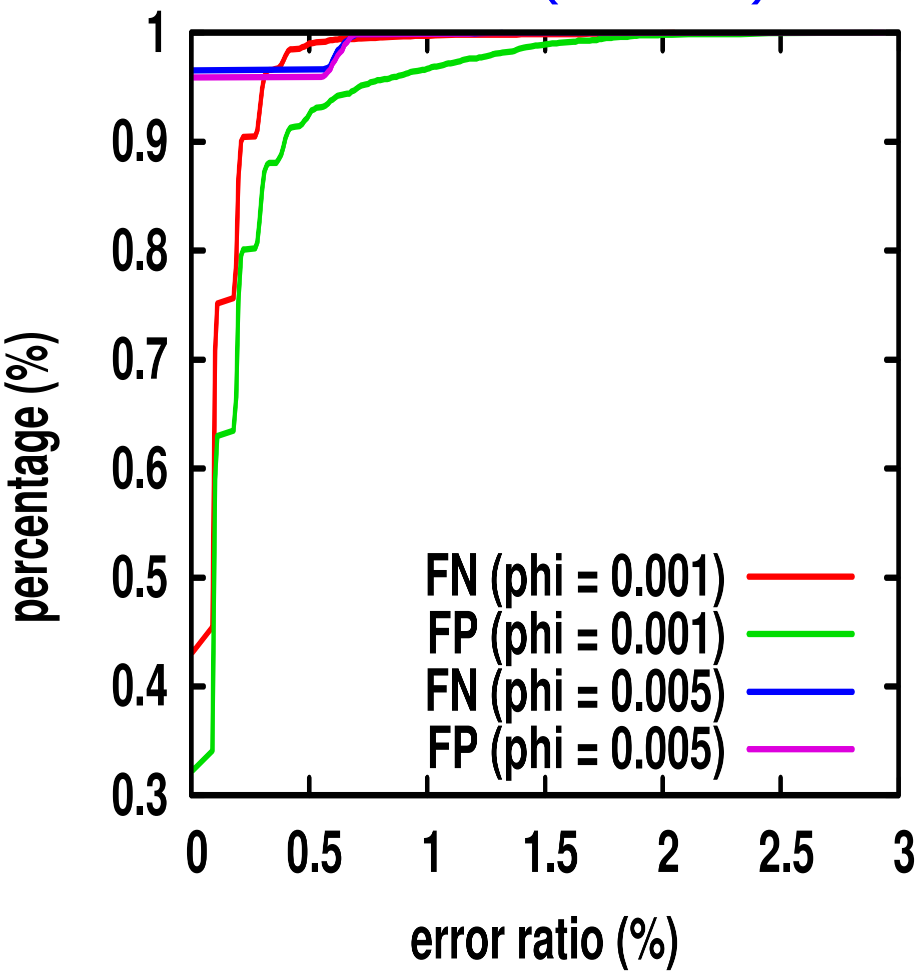
Next Steps

- Characterization
  - Aim: distinguish events using sufficiently rich set of metrics
    - E.g. DoS attacks looks different from flash crowd (bytes/flow smaller in attack)
  - Metrics:
    - # flows, # bytes, # packets, # SYN packets
      - ↑ SYN, ↔ bytes ➔ DoS??
      - ↑ packets, ↔ bytes ➔ DoS??
      - ↑ packets, in multiple /8 ➔ DoS? Worm?
- Distributed detection
  - Our summary data structures can easily support aggregating data collected from multiple locations

Thank you!

HHH Accuracy Across Time

ISP-1mon (router 1)



ISP-1mon (router 2)