MAD'12

Monitoring the Dynamics of Network Traffic by Recursive Multi-dimensional Aggregation

Midori Kato, Kenjiro Cho, Michio Honda, Hideyuki Tokuda

Keio University
1858
CALAMVS GLADIO FORTIOR
DMC Institute, Keio University
2-17-22 Mita, Minato-ku
Tokyo 108-0073 Japan

IIJ
Internet Initiative Japan

NEC
Empowered by Innovation

Background

- Traffic monitoring is important to detect changes in traffic patterns
  1. Understanding network usage
  2. Detecting abnormal conditions (e.g. flash crowd, DDoS attacks and mis-configuration)

- Concise summary is needed for operators
  - not to overlook possible anomalies but detailed enough to identify anomalies and dynamics
  - produced by efficient flow based traffic analysis
    - The dynamics of flows is good enough to understand network condition

Multi-dimensional Flow Aggregation

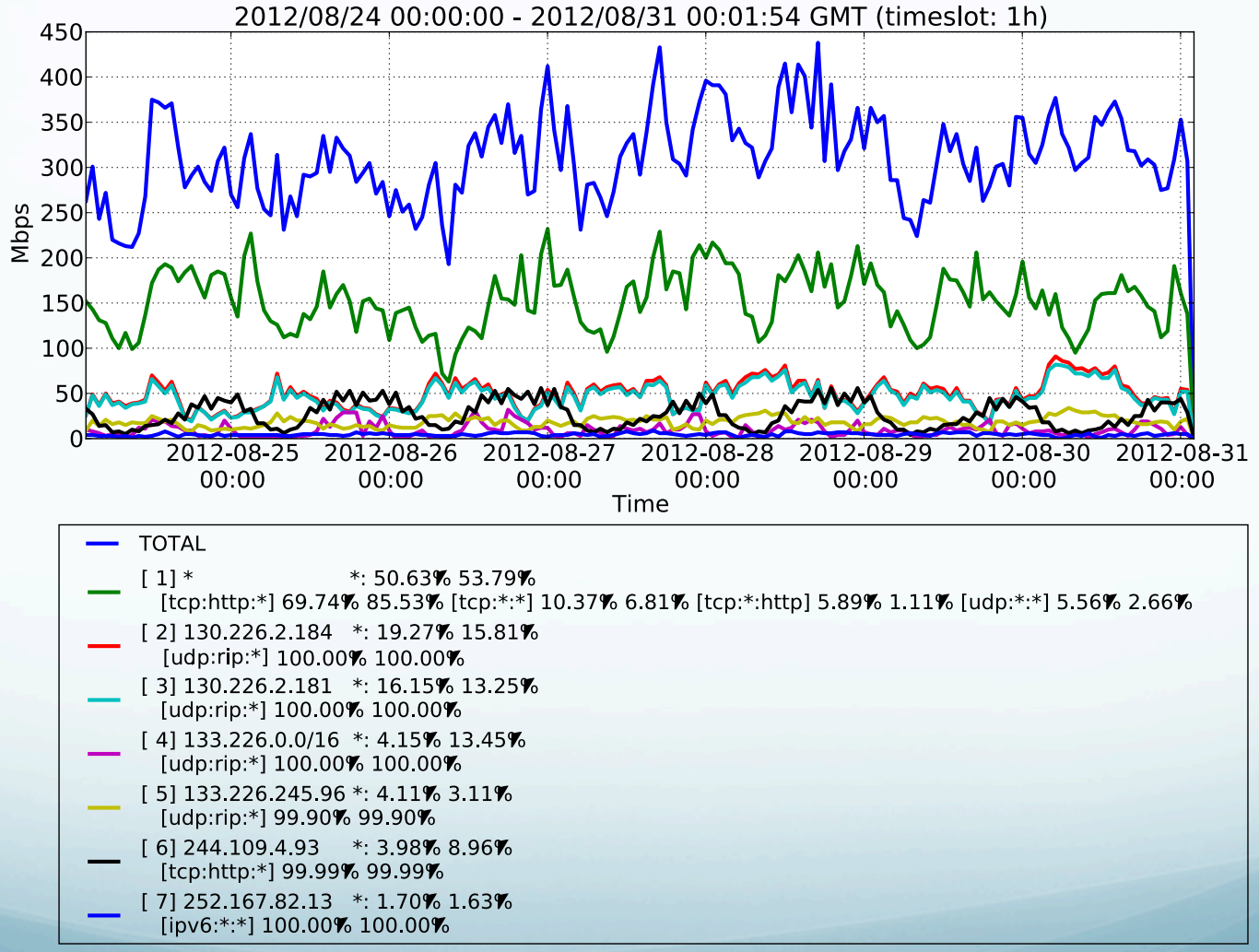
- Extract significant patterns by aggregating flows
  - Flow: packets with unique 5-tuple (source/destination IP address, source/destination port and protocol)

    (e.g. a single TCP connection)
  - Aggregated flow: a set of common attributes in 5-tuple

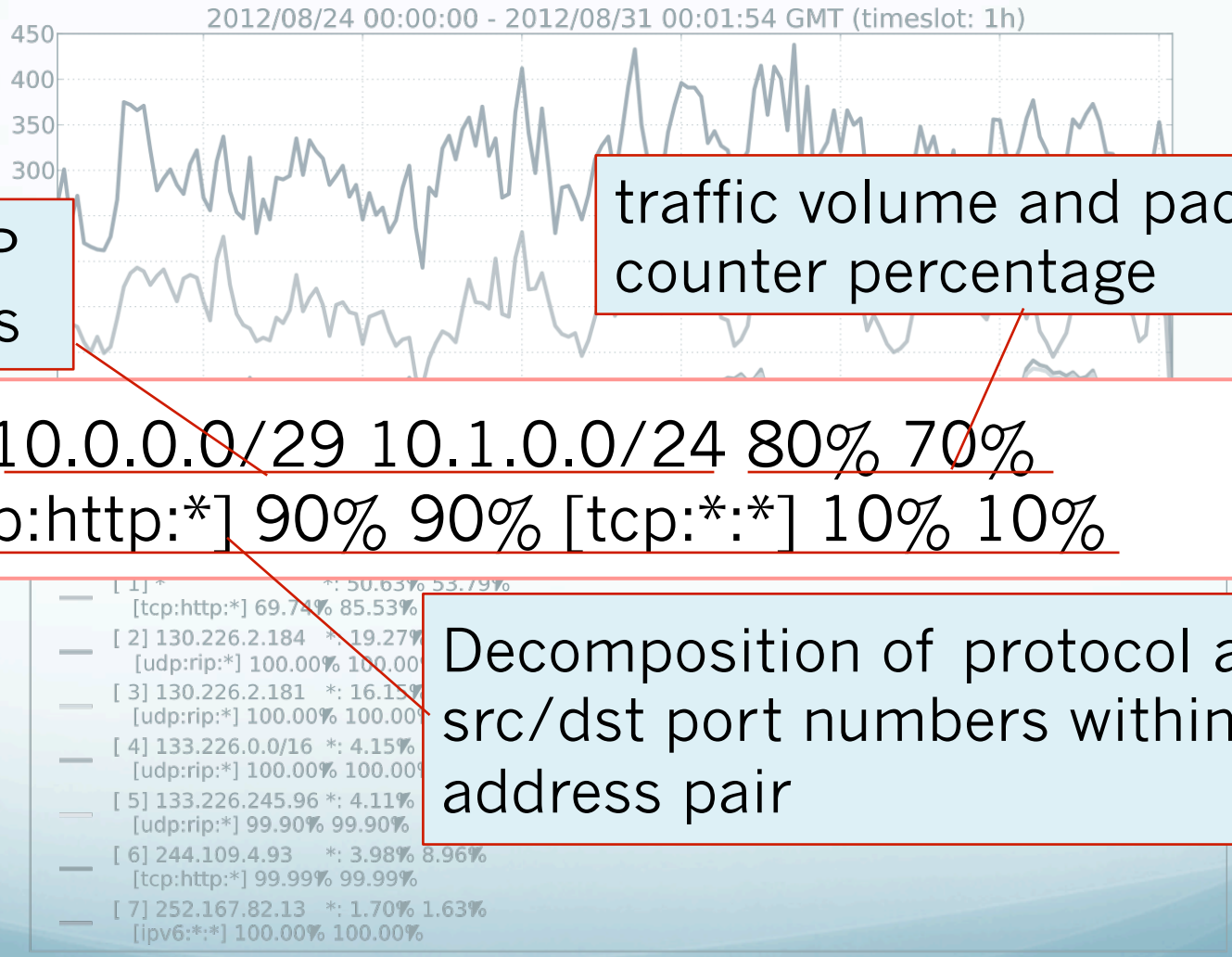
    (e.g. TCP connections directed from 10.0.0.0/29:80 to 10.1.0.0/24:*)

Flow	10.0.0.7:80 - 10.1.0.2:3003 TCP
Aggregated Flow (src/dst address)	10.0.0.0/29:80 – 10.1.0.0/24:3003 TCP
Aggregated Flow (+ dst port)	10.0.0.0/29:80 – 10.1.0.0/24:* TCP

Example of visualization



Aggregated flow representation



src/dst IP addresses

traffic volume and packet counter percentage

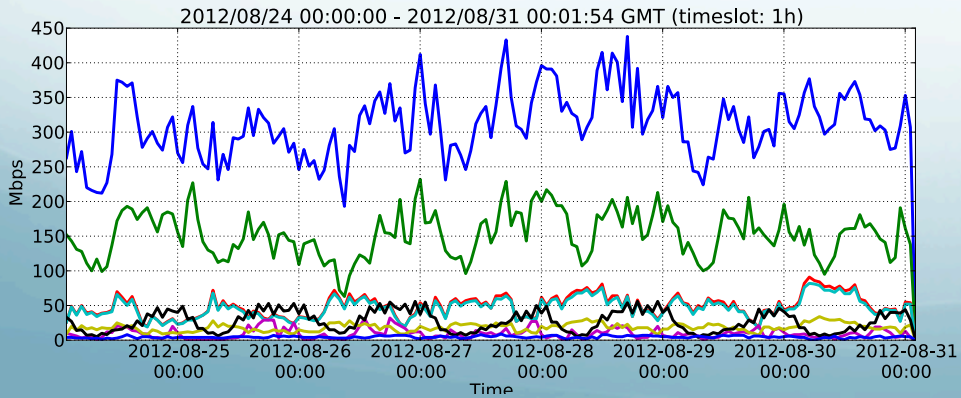
10.0.0.0/29 10.1.0.0/24 80% 70%
[tcp:http:*] 90% 90% [tcp:*:*] 10% 10%

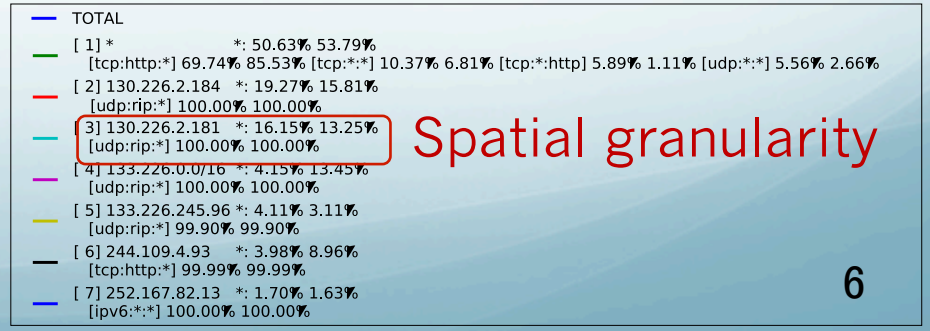
Decomposition of protocol and src/dst port numbers within IP address pair

Problem statement

- Most of aggregated flow visualization lacks interactivity
  - trade-off between performance and flexibility
- Motivation: we enhance interactivity with granularity control of aggregated flows
  - Temporal granularity: a period while significant (aggregated) flows are extracted
  - Spatial granularity: details of addresses and ports

Temporal granularity

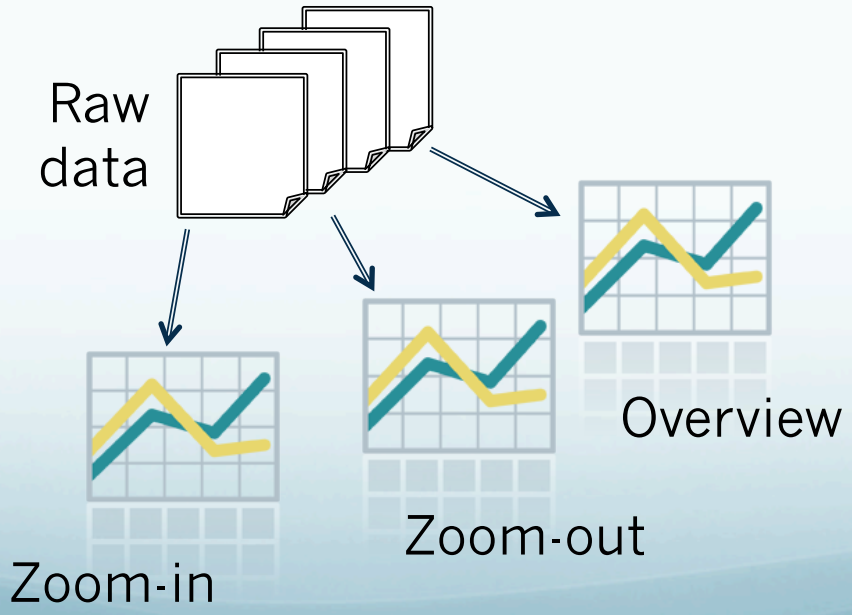




Spatial granularity

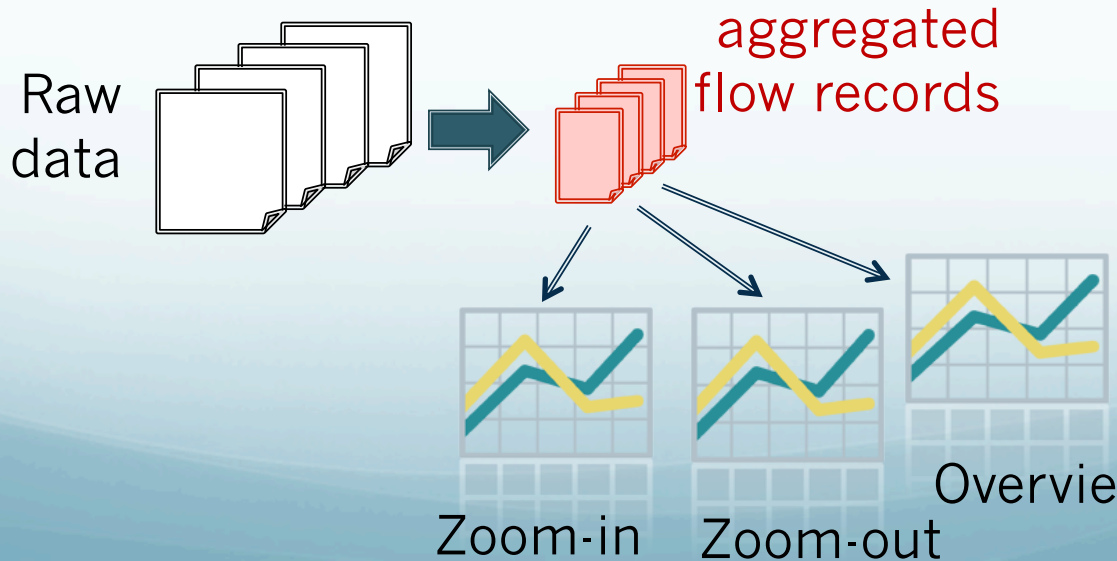
Challenges

- Cluster flows in huge attribute spaces on the fly
  - Millions of flows into several aggregated flows
- Reduce computer overhead when changing views



Agurim

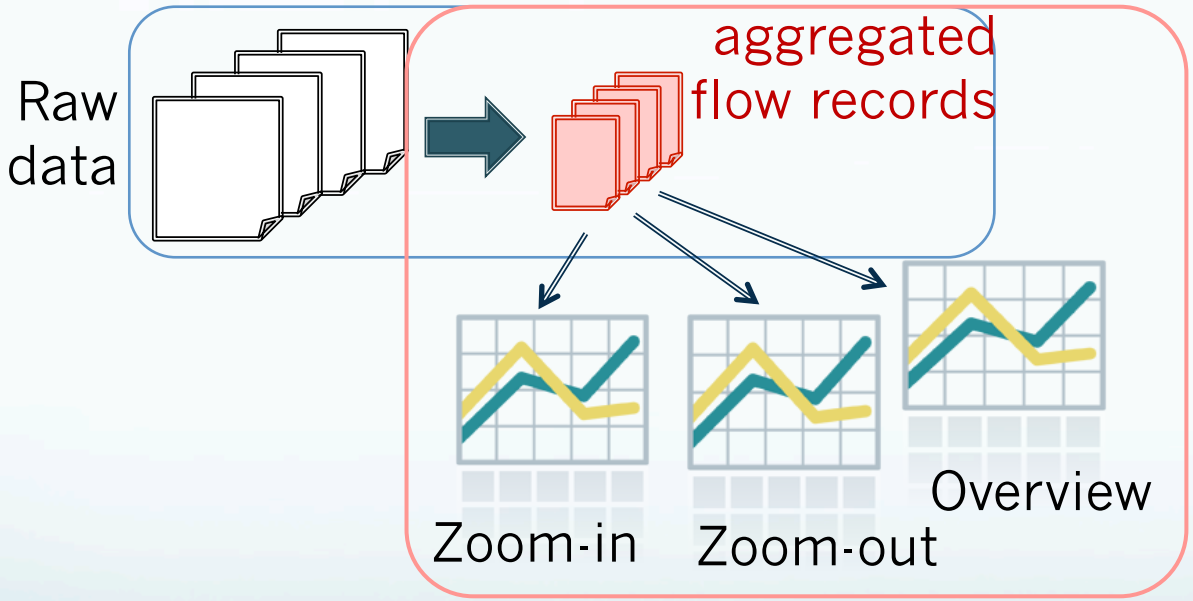
- Agurim: our proposed algorithm for **efficient** and **flexible** multi-dimensional flow aggregation
  - realizes interactive visualization
  - Efficiency: produces reusable and fine-grained aggregated-flow records from raw data
  - Flexibility: aggregates flows to provide a requested view by using these records



Agurim overview: two-staged flow aggregation

Primary Aggregation for efficient data processing

- produce reusable and fine-grained aggregated flow records from raw data



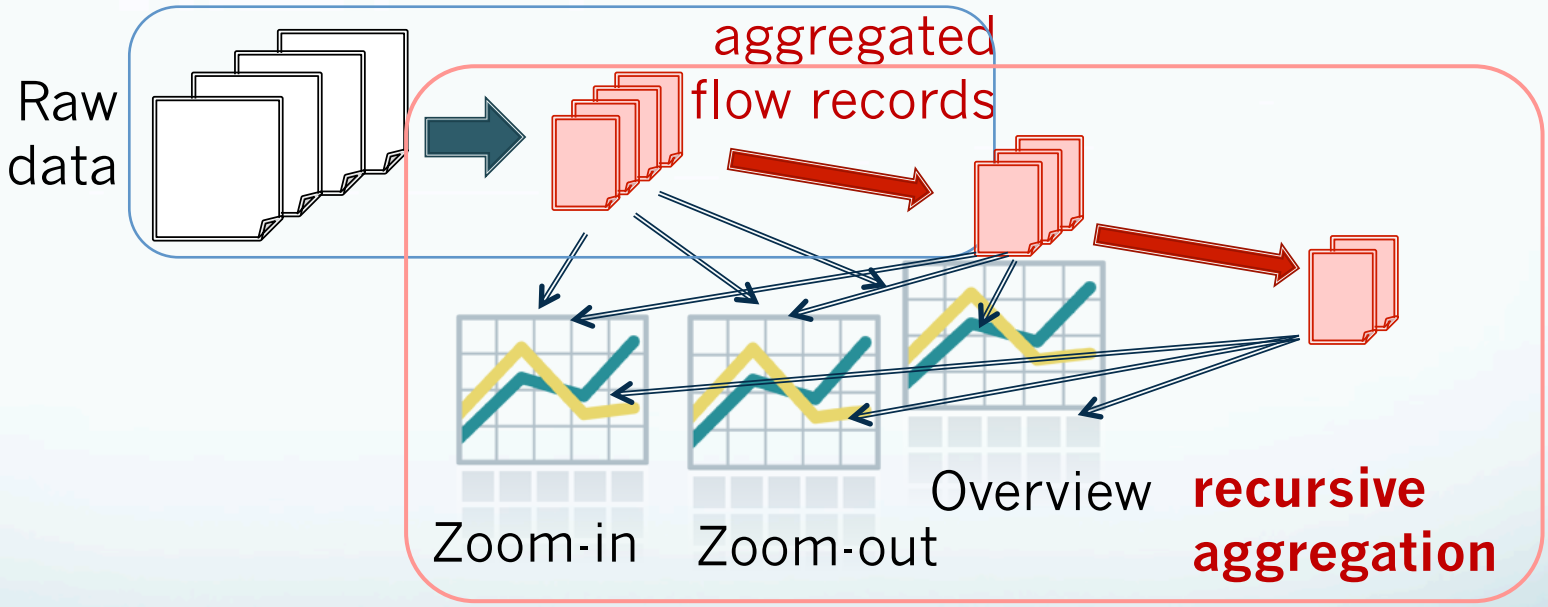
Secondary Aggregation for flexible visualization

- generate plot data for desired temporal and spatial granularity by using flow records

Agurim overview: two-staged flow aggregation

Primary Aggregation for efficient data processing

- produce reusable and fine-grained aggregated flow records from raw data



Secondary Aggregation for visualization

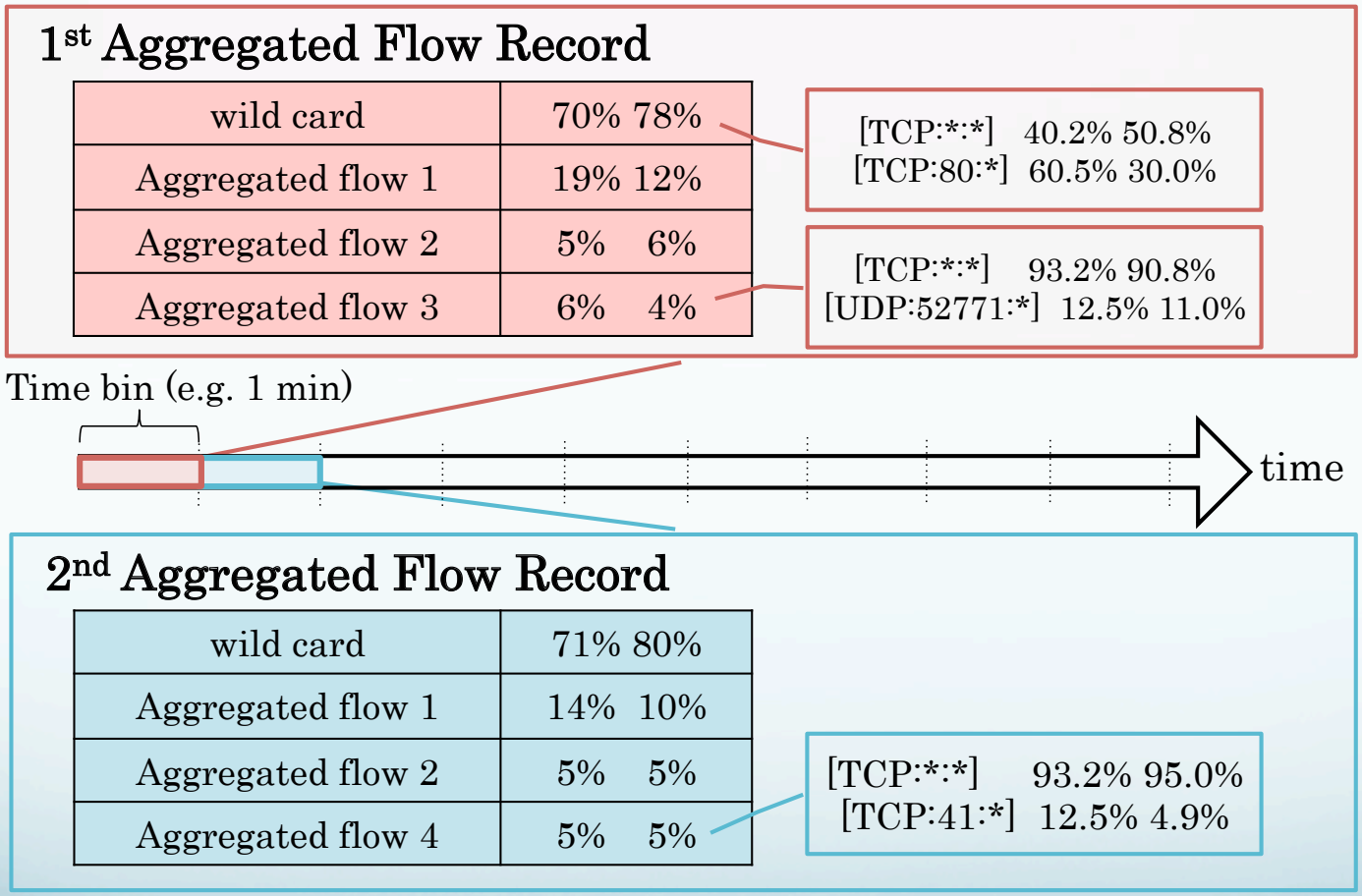
- generate plot data for desired temporal and spatial granularity
- recursive aggregation: re-aggregate flow records

Previous Work

	Monitoring tool	Multi-dimensional Flow Aggregation?	Recursive Aggregation?
1	Aguri	no	yes
2	AutoFocus	yes	no
3	ProgME	yes	no
4	HHH	yes	no
5	Multi-dimensional HHH	yes	no
	Agurim	**yes**	**yes**

[1] K. Cho, R. Kaizaki, and A. Kato. "Aguri: An aggregation-based traffic profiler" In Quality of Future Internet Services, 2001.
[2] C. Estan, S. Savage, and G. Varghese. "Automatically inferring patterns of resource consumption in network traffic" In ACM SIGCOMM 2003.
[3] L. Yuan, C.-N. Chuah, and P. Mohapatra. "Progme: towards programmable network measurement" In ACM SIGCOMM,2007
[4] Y. Zhang, S. Singh, S. Sen, N. Duffield, and C. Lund. "Online identification of hierarchical heavy hitters: algorithms, evaluation, and applications" In ACM IMC 2004
[5] G. Cormode, F. Korn, S. Muthukrishnan, and D. Srivastava. "Diamond in the rough: finding hierarchical heavy hitters in multidimensional data". In ACM SIGMOD, 2004

Primary Aggregation Overview



Agurim produces fine-grained aggregated flow records for each time bin

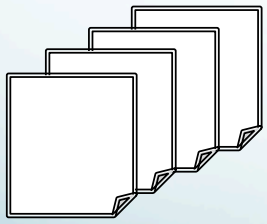
Primary Aggregation overview (cont')

- processes 5-attribute space by two-pass algorithm

[1st-pass] Aggregate each attribute separately

[2nd-pass] Match each packet against aggregated attributes

Raw data



Aggregated attributes for src address

$S_1, S_2, \ldots, S_m$

Aggregated attributes for dst address

$D_1, D_2, \ldots, D_n$

# of attributes: 1/100 – 1/1000

dst\src

	S_0	S_1	S_2		S_m
D_0	300	12	77	⋯	⋯
D_1	28	0	14		
D_2	59	0	0		
⋮				⋱	
D_n					⋱

(unit: KB)

aggregated flow records

* Protocol/port space are processed in a similar way

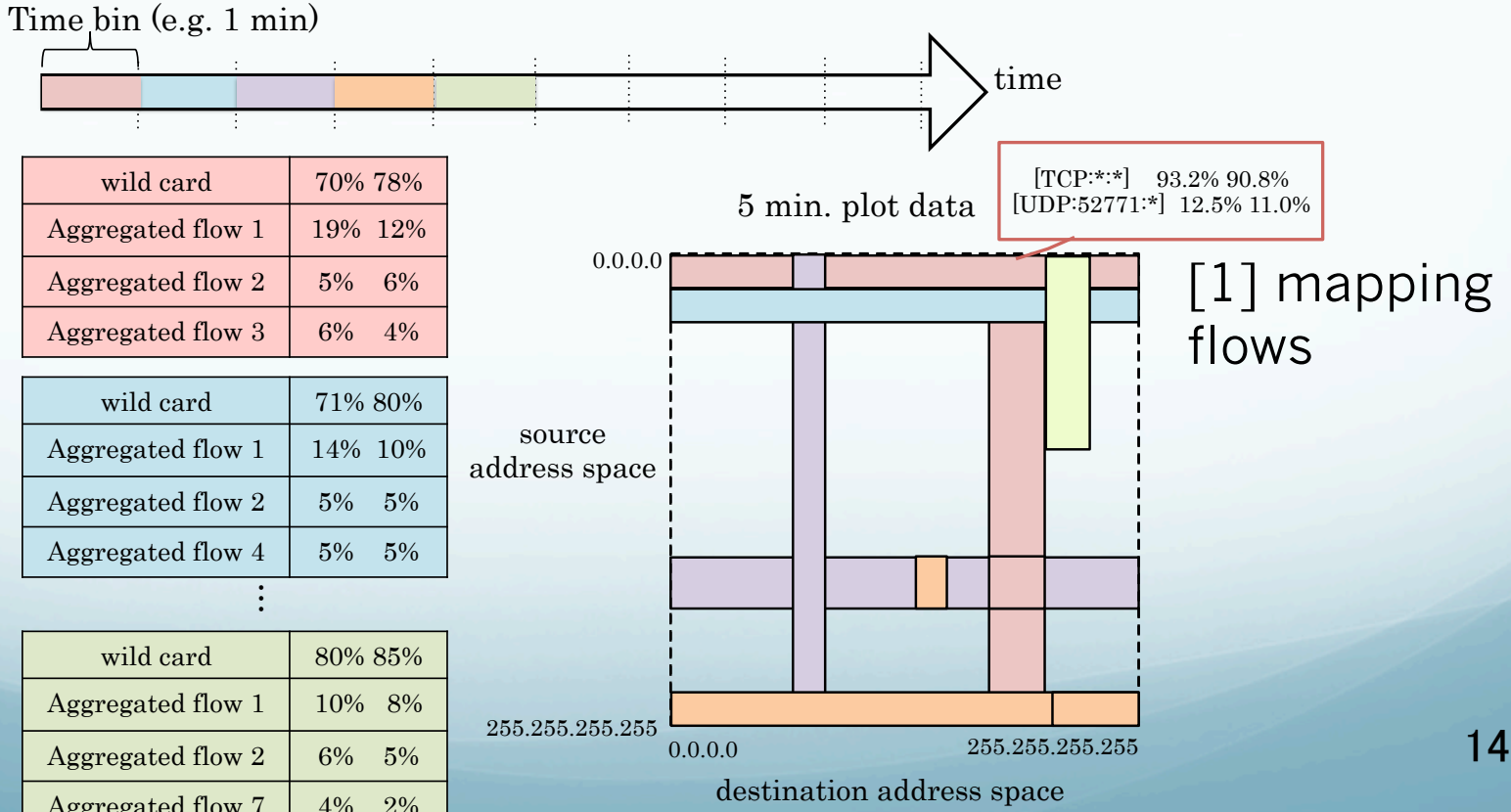
Secondary Aggregation Overview

- Aggregated flow records still have redundant information to visualize
- In the secondary aggregation, Agurim

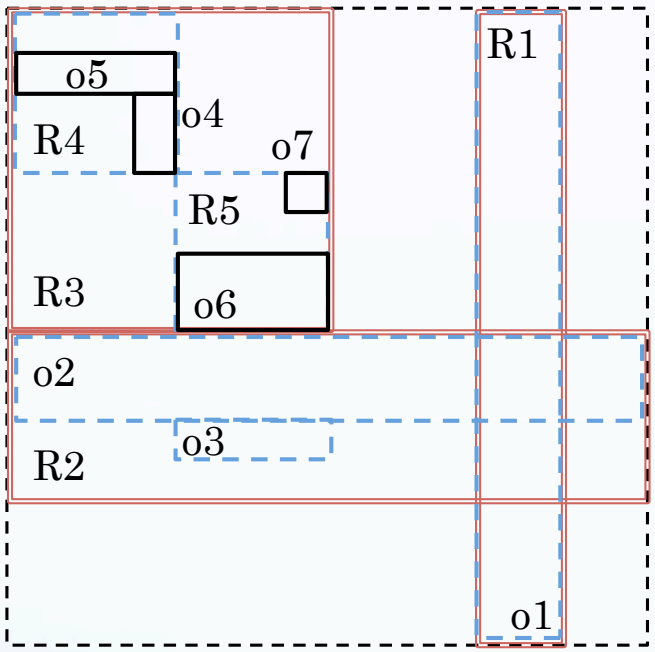
[1] maps all the flow records into address space for a specified duration

[2] aggregates small flows to find superset for visualization

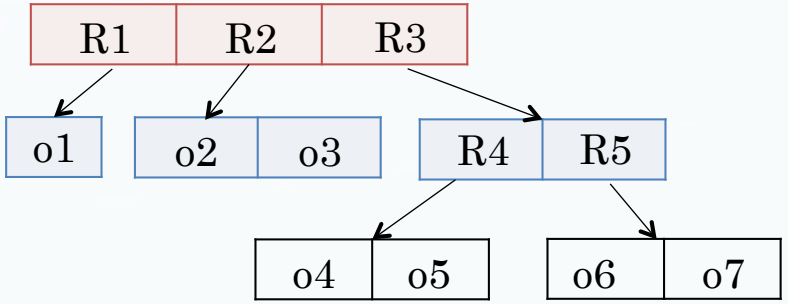
[3] re-aggregate flows to reduce computation overhead



Secondary Aggregation Overview (cont')

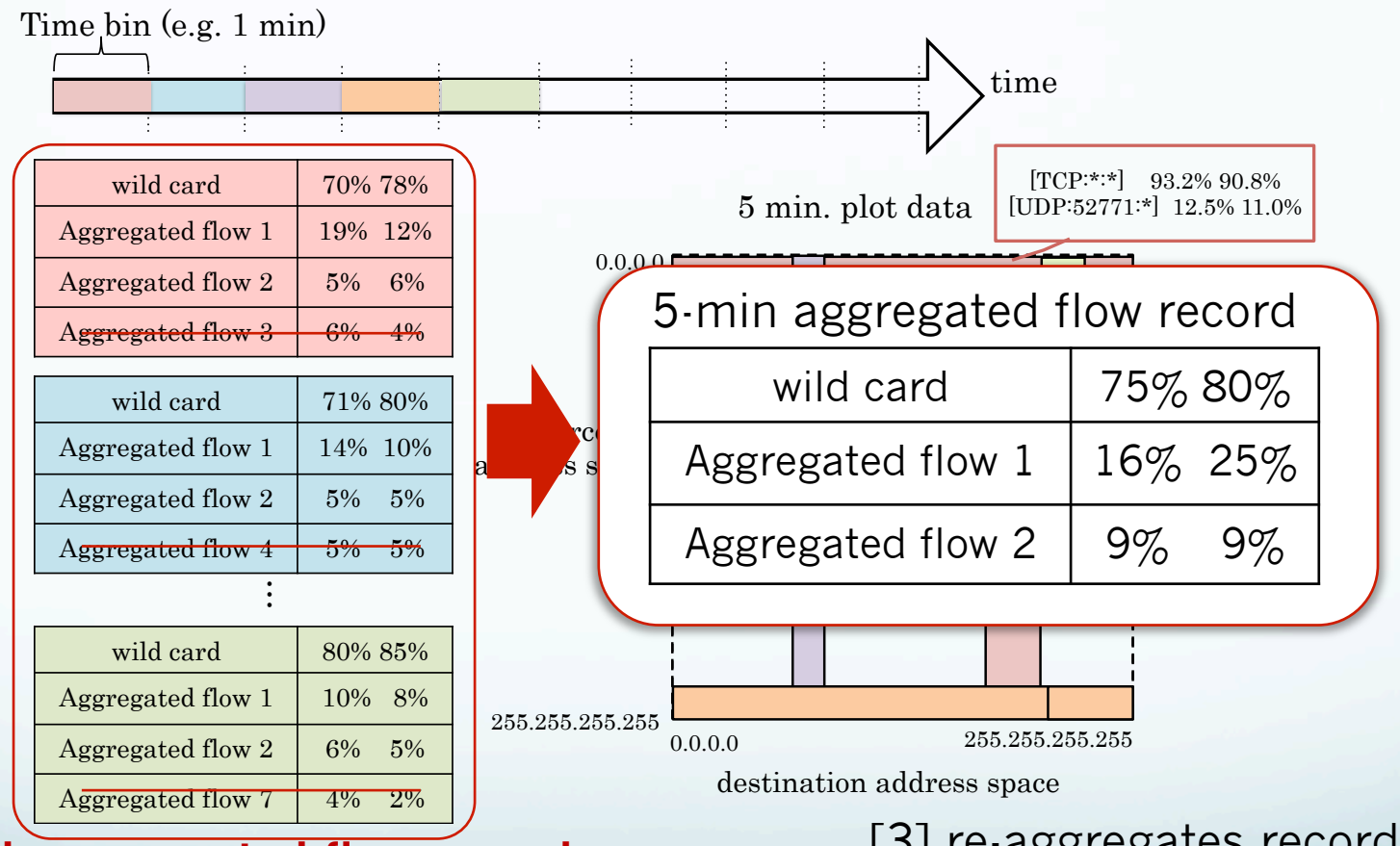


[2] aggregating small flows



- R-tree enables spatial data access
  - Parent node holds the entire region of child nodes
  - Aggregate neighbors based on minimum boundary region
- We use R-tree not to answer range queries but to find superset

Secondary Aggregation Overview (cont')



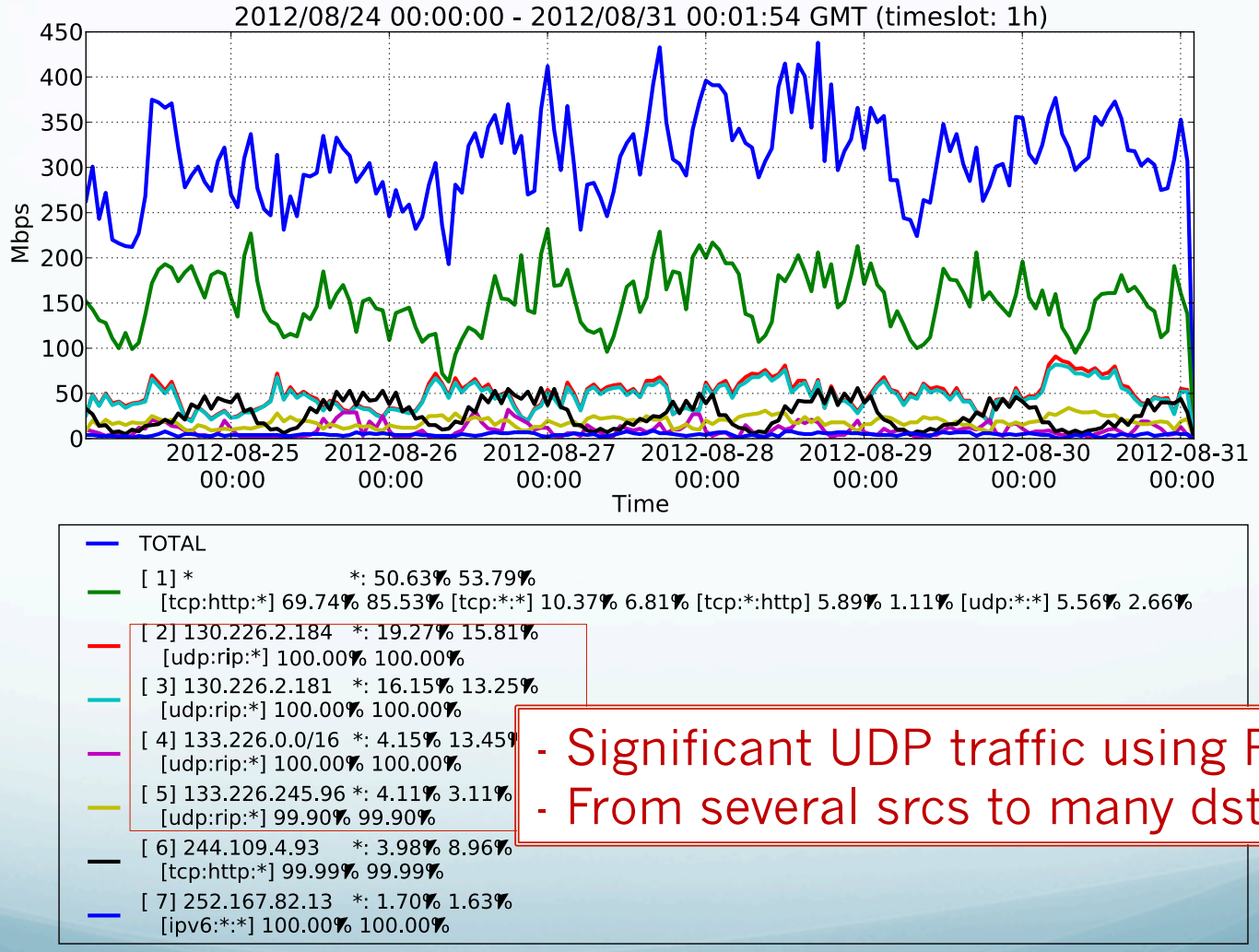
5 x 1min aggregated flow records

[3] re-aggregates records to reduce computation overhead

- Agurim aggregates short and significant flows by flow re-aggregation

- Demo

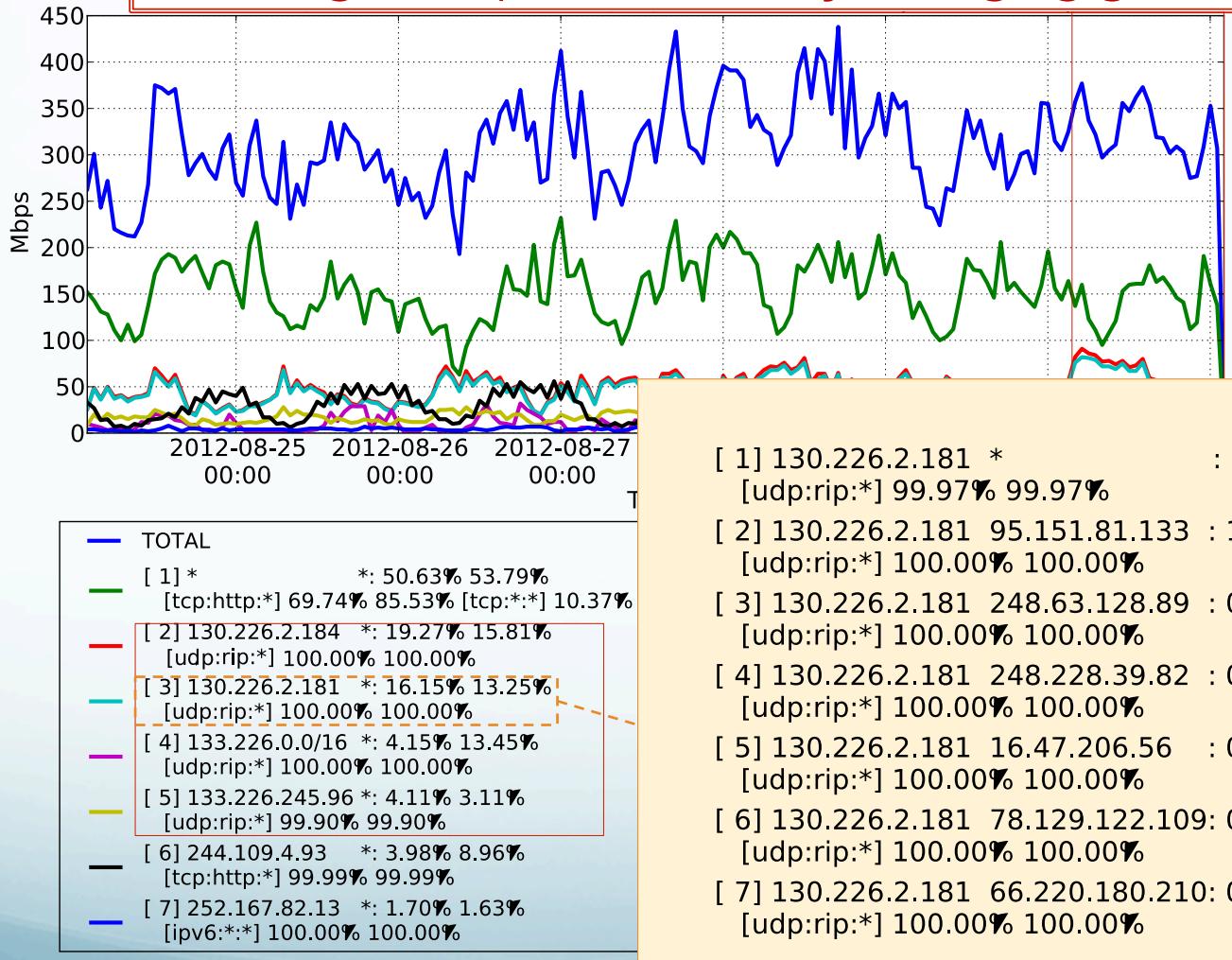
Agurim visualization



- Significant UDP traffic using RIP port
- From several srcs to many dsts

Agurim visualization

zooming a suspicious flow by changing granularity



[1] 130.226.2.181 * : 94.65% 94.65%
[udp:rip:*] 99.97% 99.97%

[2] 130.226.2.181 95.151.81.133 : 1.46% 1.46%
[udp:rip:*] 100.00% 100.00%

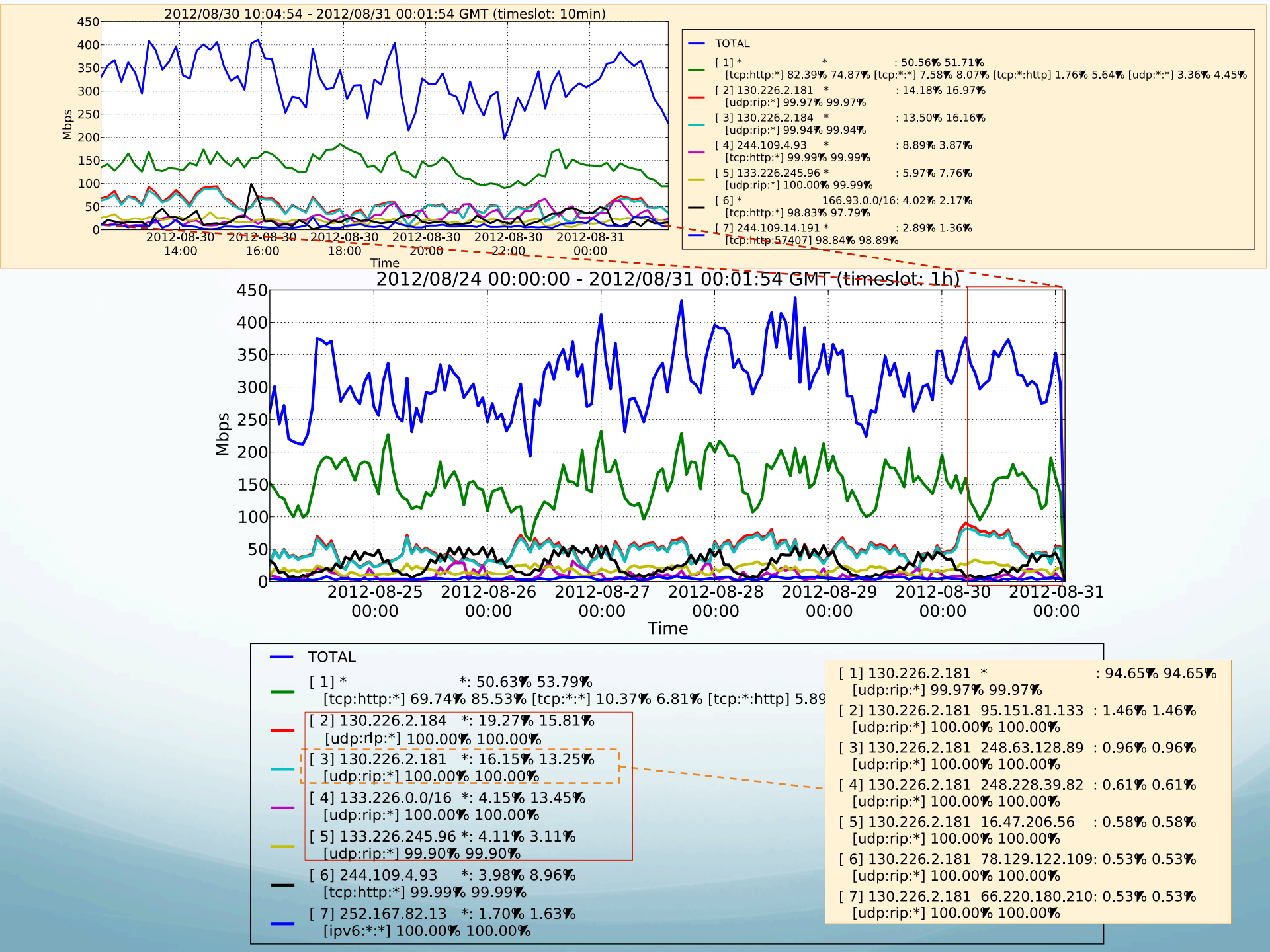
[3] 130.226.2.181 248.63.128.89 : 0.96% 0.96%
[udp:rip:*] 100.00% 100.00%

[4] 130.226.2.181 248.228.39.82 : 0.61% 0.61%
[udp:rip:*] 100.00% 100.00%

[5] 130.226.2.181 16.47.206.56 : 0.58% 0.58%
[udp:rip:*] 100.00% 100.00%

[6] 130.226.2.181 78.129.122.109: 0.53% 0.53%
[udp:rip:*] 100.00% 100.00%

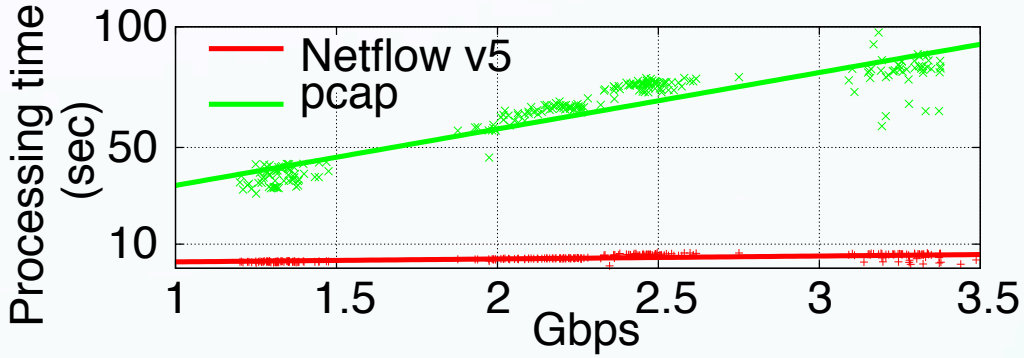
[7] 130.226.2.181 66.220.180.210: 0.53% 0.53%
[udp:rip:*] 100.00% 100.00%


2012/08/30 10:04:54 - 2012/08/31 00:01:54 GMT (timeslot: 10min)
Mbps
Time
2012-08-30 14:00
2012-08-30 16:00
2012-08-30 18:00
2012-08-30 20:00
2012-08-30 22:00
2012-08-31 00:00
TOTAL
[1] * * : 50.56% 51.71%
[tcp:http:*] 82.39% 74.87% [tcp:*:*] 7.58% 8.07% [tcp:*:http] 1.76% 5.64% [udp:*:*] 3.36% 4.45%
[2] 130.226.2.181 * : 14.18% 16.97%
[udp:rip:*] 99.97% 99.97%
[3] 130.226.2.184 * : 13.50% 16.16%
[udp:rip:*] 99.94% 99.94%
[4] 244.109.4.93 * : 8.89% 3.87%
[tcp:http:*] 99.99% 99.99%
[5] 133.226.245.96 * : 5.97% 7.76%
[udp:rip:*] 100.00% 99.99%
[6] * 166.93.0.0/16: 4.02% 2.17%
[tcp:http:*] 98.83% 97.79%
[7] 244.109.14.191 * : 2.89% 1.36%
[tcp:http:57407] 98.84% 98.89%
2012/08/24 00:00:00 - 2012/08/31 00:01:54 GMT (timeslot: 1h)
Mbps
Time
2012-08-25 00:00
2012-08-26 00:00
2012-08-27 00:00
2012-08-28 00:00
2012-08-29 00:00
2012-08-30 00:00
2012-08-31 00:00
TOTAL
[1] * *: 50.63% 53.79%
[tcp:http:*] 69.74% 85.53% [tcp:*:*] 10.37% 6.81% [tcp:*:http] 5.89
[2] 130.226.2.184 *: 19.27% 15.81%
[udp:rip:*] 100.00% 100.00%
[3] 130.226.2.181 *: 16.15% 13.25%
[udp:rip:*] 100.00% 100.00%
[4] 133.226.0.0/16 *: 4.15% 13.45%
[udp:rip:*] 100.00% 100.00%
[5] 133.226.245.96 *: 4.11% 3.11%
[udp:rip:*] 99.90% 99.90%
[6] 244.109.4.93 *: 3.98% 8.96%
[tcp:http:*] 99.99% 99.99%
[7] 252.167.82.13 *: 1.70% 1.63%
[ipv6:*:*] 100.00% 100.00%
[1] 130.226.2.181 * : 94.65% 94.65%
[udp:rip:*] 99.97% 99.97%
[2] 130.226.2.181 95.151.81.133 : 1.46% 1.46%
[udp:rip:*] 100.00% 100.00%
[3] 130.226.2.181 248.63.128.89 : 0.96% 0.96%
[udp:rip:*] 100.00% 100.00%
[4] 130.226.2.181 248.228.39.82 : 0.61% 0.61%
[udp:rip:*] 100.00% 100.00%
[5] 130.226.2.181 16.47.206.56 : 0.58% 0.58%
[udp:rip:*] 100.00% 100.00%
[6] 130.226.2.181 78.129.122.109: 0.53% 0.53%
[udp:rip:*] 100.00% 100.00%
[7] 130.226.2.181 66.220.180.210: 0.53% 0.53%
[udp:rip:*] 100.00% 100.00%

Performance of Primary Aggregation

- Measure processing time using pcap(tcpdump) and Netflow[6] data

Dataset: 4-hour trace data collected on 10Gbps link from Tier 1 ISP to CAIDA
timebin: 1 min.
CPU: Intel Core i5 @2.5GHz



- Netflow: Agurim takes less than 10-second to process 1-minute-long 3Gbps traffic
  - Because trace data has been aggregated to some extent
- Pcap: Agurim takes 40-second to process 1-minute-long 1Gbps traffic
  - Main contributor: attirubute lookup for each packet
  - Possible optimization: flow cache to store frequently-accessed items

[6] T. Zseby and S. Zander and G. Carle“RFC3334: Policy-Based Accounting”

Performance of Secondary Aggregation

- Measure processing time until Agurim generates plot data

Dataset: 7-day-long aggregated flow records collected on 150Mbps transit link of WIDE backbone
Time bin of records: 1 min
Time resolution of plot: 1 hour

time period in the entire view	observed unique aggr. flows	processing time
12-hour	2,178	0.44 sec
1-day	3,796	1.35 sec
3-day	9,858	13.46 sec
1-week	23,065	75.77 sec

- Processing time increases exponentially with the number of flows
  - Main contributor: loop of finding and aggregating small flows until the requested # of flows is reached
  - On-going work: optimization with flow re-aggregation
    - Pre-aggregate small flows
    - Recursive aggregation: Prepare coarser data (e.g., 1-hour, 1-day) for coarse grained views

Conclusion

- Agurim
  - achieve dynamic granularity control by multi-dimensional flow re-aggregation
  - produce 1-day-long plot data in 1.5 seconds
- Agurim will help to manage complex and dynamic today's networks
- Future work
  - Release the code as open source software
  - Improve system performance
  - Evaluate the aggregation accuracy
- Demo in the demo session