Coordinated Sampling sans Origin-Destination Identifiers: Algorithms and Analysis

Vyas Sekar, Anupam Gupta, Michael K. Reiter, Hui Zhang

Carnegie Mellon University
Univ. of North Carolina Chapel-Hill
Flow Monitoring is critical for effective Network Management

- Traffic Engineering
- Accounting
- Worm Detection
- Network Forensics
- Analyze new user apps
- Need high-fidelity measurements
- High flow coverage
- Provide network-wide goals
- Respect resource constraints
How do we meet the requirements?

- High flow coverage
- Provide network-wide goals
- Respect resource constraints

Flow Sampling

Network-Wide Coordination & Optimization

cSamp [NSDI’08]
Network-wide coordination

Assign non-overlapping ranges per OD-pair or path
All routers configured with same hash function/key
Generating Sampling Manifests

**Inputs**
- OD-pair info
- Traffic, Path(routers)
- Router constraints e.g., SRAM for flow records

**Outputs**
- Network-wide Optimization (@ NOC)
  - Sampling manifests
    - \{<OD-Pair,Hash-range>\} per router

**Objective:**
\[
\text{Max } \sum_{i \in \text{ODPairs}} \text{Coverage}_i \times \text{Traffic}_i
\]
Subject to achieving maximum
\[
\text{Min}_{i \in \text{ODPairs}} \{ \text{Coverage}_i \}
\]

**Linear Program**
**cSamp algorithm on each router**

1. Get OD-Pair from packet
2. Compute hash (flow = packet 5-tuple)
3. Look up hash-range for OD-pair from sampling manifest
4. Log if hash falls in range for this OD-pair
1. Get OD-Pair from packet

**Why is this challenging?**
OD-pair identification might be ambiguous → Multi-exit peers (and prefix aggregation) (Even with MPLS)

**How does cSamp overcome this?**
Ingresses compute and add this to packet headers

Need to modify packet headers/add shim header
Extra computation on ingresses
May require overhauling routing infrastructure
Can we realize the benefits of cSamp without OD-pair identification?

Use local information to make sampling decisions

“Stitch” coverage across routers on a path
Outline

• Background and Motivation

• Problem Formulation

• Algorithms and Heuristics

• Evaluation
What *local* info can I get from packet and routing table?

{Previous Hop, My Id, NextHop}

*SamplingSpec*
Granularity at which sampling decisions are made

How much to sample for this *SamplingSpec*?

*SamplingAtom*
Discrete hash-ranges, select some to log
“Stitching” together coverage

R1

= 

union

R3

union

R2

= 

union

R4

R5

R6

R7
Problem Formulation

Coverage for path $P_i$

$C_i = \bigcup_{a_k \in P_i} \bigcup_l h(\widehat{g_{kl}})$

Load on router $R_j$

$Load_j = \sum_{a_k \in R_j.\text{specs}} T_k \times \bigcup_l h(\widehat{g_{kl}})$

**Maximize:**

Total flow coverage: $\sum_i T_i C_i$

Minimum fractional coverage: $\min_i \{C_i\}$

**Subject To:**

$\forall$ $j$, $Load_j \leq L_j$
Outline

• Background and Motivation

• Problem Formulation

• *Algorithms and Heuristics*

• Evaluation
Maximize:
Total flow coverage: $\sum_i T_i C_i$
Min. frac coverage: $\min_i \{C_i\}$

Subject To:
$\forall j$, Load$_j \leq L_j$

Total flow coverage:
Submodular maximization with partition-knapsack
$\rightarrow$ Efficient greedy algorithm is near-optimal

Min. fractional flow coverage:
$\rightarrow$ Need “resource augmentation”
Intelligent resource augmentation
Incrementally add OD-pair identifiers

NP-hard!
Min: Hard to approximate!
Leveraging submodularity for $f_{\text{tot}}$

A function $F: 2^V \rightarrow \mathbb{R}$ is **submodular** if $\forall \ A \subseteq A' \subseteq V$, and $\forall \ s \in V,$

$$F(A \cup \{s\}) - F(A) \geq F(A' \cup \{s\}) - F(A')$$

“diminishing returns”

Why does it matter?
Max $F$ s.t $c(A) \leq B$, where $F$ is monotone
Greedy algorithm gives a constant-factor approximation
Can do lazy evaluation to speedup

Maximize: $f_{\text{tot}} = \sum_i T_i C_i$, Subject To: $\forall \ j$, $\text{Load}_j \leq L_j$
Special case of above problem with “partition-knapsack”
What about $f_{\text{min}}$?

$f_{\text{min}} = \min_i \{C_i\}$ is not submodular
Hard to approximate without violating constraints!
But, can get near-optimal, if we violate by a fixed factor

Main idea: Define $f' = \sum_i C'_i$ where $C'_i = \min \{C_i, T\}$
Note that $f' = N \times T$, iff each $C_i \geq T$
Run binary search over $T$ to find best solution
(Each iteration runs greedy with no resource constraints)

Heuristic improvements:
1. Intelligent resource augmentation
2. Upgrade a few ingress to add OD-pairs
Outline

• Motivation
• Problem Formulation
• Algorithms and Heuristics

• Evaluation
Total flow coverage

```
cSamp-T (tuple+) gives near-ideal total flow coverage vs. cSamp
```
Minimum fractional coverage
(with intelligent resource augmentation)

Can get 75% of optimal performance with 1.5X total increase and a 5X max-per-router increase
Summary

• cSamp for efficient flow monitoring
  • Network-wide coordination and optimization
  • But needs OD-pair identification

• How to implement cSamp without OD-pair ids?

• Leverage submodularity for total coverage

• Targeted upgrades for minimum fractional coverage

• cSamp-T makes cSamp’s benefits more immediately available