MassDAL
Massive Data Analysis Lab

What's New: Finding Significant Differences in Network Data Streams

S. Muthukrishnan
muthu@cs.rutgers.edu

Graham Cormode

Network Data Analysis

MassDAL
Massive Data Analysis Lab

Network managers must measure and analyze traffic:

- Maintenance: Failure detection, routing optimization
- Provisioning: Usage monitoring, prediction
- Accounting: Billing, TOS abuse, marketing
- Security: Intrusion detection, attacker identification

The Problem

MassDAL
Massive Data Analysis Lab

Metadata observed while routing packets in IP networks is truly massive.

The size of packet headers seen per hour per router can be gigabytes

Too much information to store or transmit, but each packet is seen as it is processed

→ So try (near) real time analysis of packet streams: make summary based on live traffic, query offline

Challenges

MassDAL
Massive Data Analysis Lab

Many challenges for near-real time analysis:

- Full packet logs not normally kept for later analysis, so cannot backtrack on past data
- Want to record information in network, at line speeds
- Must use small (SRAM) memory, limited memory accesses to keep pace of OC48 speeds.

Network Data Analysis

MassDAL
Massive Data Analysis Lab

Fundamental network management questions often map onto "simple" functions of the data:

- How many distinct host addresses?
- Destinations using most bandwidth?
- *Address with biggest change in traffic overnight?*

The complexity arises from having limited space and fast response requirements.

What's New?

MassDAL
Massive Data Analysis Lab

- Focus on a particular problem, Change Detection.
- Find the item with biggest change in traffic between two measurements
- Could be between difference between traffic on different days, or on different links, etc.
- Many ways to measure 'change' in behavior, we use changes in traffic size per address

Measuring Change

MassDAL
Massive Data Analysis Lab

Call an item (address) with large change a deltoid.
Measure change as:

- Absolute change: find large difference in traffic —
  Find all i so $|x[i] - y[i]| > \phi \, ||x - y||$
  $||x - y||$ is sum of changes, ϕ is threshold < 1
- Relative change: find large percentage difference
- Variational Change: find large variance in readings over several measurements

Change Detection

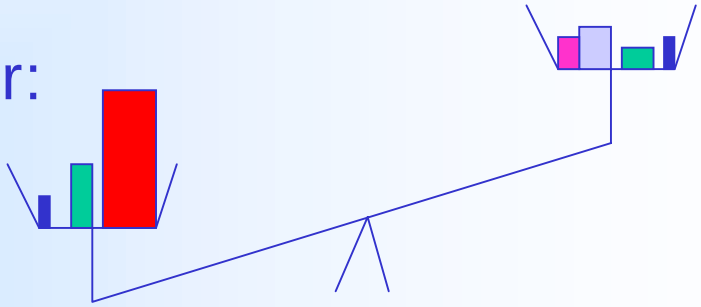
MassDAL
Massive Data Analysis Lab

- Use Non-Adaptive Group Testing: will pick groups of items in a randomized fashion
- Within each group, test for "deltoids": items that have shown a large change in behavior
- Must keep enough information to recover identity of deltoids.
- We separate the structure of the groups from the tests, and consider each in turn.

Groups: Simple Case

MassDAL
Massive Data Analysis Lab

- Suppose there is just one large item, i, whose "weight" is more than half the weight of all items.
- Use a pan-balance metaphor: this item will always be on the heavier side



- Assume we have a test which tells us which group is heavy. The large item is always in that group.
- Arrange these tests to let us identify the deltoid.

Solving the simple case

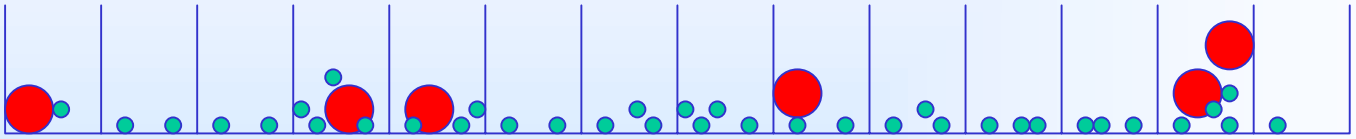
MassDAL
Massive Data Analysis Lab

- Keep a test of items whose identifier is odd, and for even: result of test tells whether i is odd or even
- Similarly, keep tests for every bit position. If there are items 1... n, then need log n tests
- Then can just read off the index of the heavy item
- Now, turn original problem into this simple case…

Spread into Buckets

MassDAL
Massive Data Analysis Lab

Allocate items into buckets:



- With enough buckets, we expect to achieve the simple case: each deltoid lands in a bucket where the rest of weight is small
- Repeat enough times independently to guarantee finding all deltoids

Group Structure

MassDAL
Massive Data Analysis Lab

Scheme finds all deltoids with weight at least ϕ of total amount of change, none with less than $\phi - \varepsilon$.

- Use a universal hash function to divide the universe into $2/\varepsilon$ groups, repeat $t = \log 1/\delta$ times.
- Keep a test for each group to determine if there is a deltoid within it. Keep $2\log n$ subgroups in each group based on the bit positions to identify deltoids.

Update procedure: for each update, find the groups the items belongs to and update the corresponding tests.

Group Testing

MassDAL
Massive Data Analysis Lab

- Searching: For each group whose test is positive, read results of tests of subgroups:
  if test j is positive, bit j = 1, test j' positive, bit j = 0
- Avoid false positives: If test j and j' both positive, there are two deltoids in same group, so reject the group (also if j and j' both negative).
- Avoid false positives: Check the recovered item belongs to that group. If so, output it as a deltoid.
- Result: Find all deltoids, if tests gave correct results.

Test for Absolute Changes

MassDAL
Massive Data Analysis Lab

- Non-Adaptive Group testing: Group items in the universe and test for a large change in each group
- Build tests based on keeping sum of traffic of items in each (sub)group
- Tests can fail: false positives and false negatives
- Will use universal hash functions: these give simple guarantees on probability any pair of items collide

Building the Test

MassDAL
Massive Data Analysis Lab

- Suppose i is an absolute change deltoid, then $|x[i] - y[i]| > \phi \|x - y\|$
- For each group G, keep $T[G] = \Sigma_{j \in G} (x[j] - y[j])$
- Test is positive if $|T[G]| > \phi \|x - y\|$
- Argue that in each group i falls in there is a good chance that i will be discovered as a deltoid. Repetitions amplify this probability

Proof outline

MassDAL
Massive Data Analysis Lab

Test will give false positive if

$$|x[i] - y[i]| < (\phi - \varepsilon)\,\|x - y\|$$

and $$|\Sigma_{j \in G} (x[j] - y[j])| > \phi\,\|x - y\|$$

Test may give false negative if

$$|x[i] - y[i]| > (\phi + \varepsilon)\,\|x - y\|$$

and $$|\Sigma_{j \in G} (x[j] - y[j])| < \phi\,\|x - y\|$$

Neither can happen if (stronger condition)

$$Z = \Sigma_{j \in G,\, j \neq i} |(x[j] - y[j])| < \varepsilon\,\|x - y\|$$

Proof Outline

MassDAL
Massive Data Analysis Lab

Expectation of $Z = \Sigma_{j \in G,\, j \neq i} |(x[j] - y[j])|$
$= \Sigma_j \Pr[\text{hash}(i) = \text{hash}(j)] * |x[j] - y[j]|$
$= \varepsilon/2 * \|x - y\|$

$\Pr[Z > \varepsilon \|x - y\|] = \Pr[Z > 2E(Z)]$
$< 1/2$ by Markov inequality

Repetitions give high probability of finding all deltoids.

Additional (verification) tests on each item found give low probability of false positives

Absolute Change Code

MassDAL
Massive Data Analysis Lab

```
For each (item, count)
   For a = 1 to t do
      b = hash(a,item)
      For c = 1 to log n do
          If (bit(item,c)=1)
            T[a,b,c]+=count
```

t can be quite small (3 or 4), can be parallelized

log n typically is 32 for IP addresses, can be reduced at expense of more memory used

Relative Change Test

MassDAL
Massive Data Analysis Lab

Keep different information for each stream.

- For stream x, keep $T(x)[j] = \Sigma_{h(i) = j} a(x)[i]$
  sum counts of items in the group
- For stream y, keep $T(y)[j] = \Sigma_{h(i) = j} (1/a(y)[i])$
  sum reciprocal of counts of items in the group
- Test: if $T(x)[j]*T(y)[j] > \phi \Sigma (a(x)[i]/a(y)[i])$
  test if product of counts exceeds threshold
- Must be able to find $(1/a(y)[i])$ – open problem to remove this restriction

Relative Change Test

MassDAL
Massive Data Analysis Lab

- Test has one-sided error, will always say yes if
  $(a(x)[i]/a(y)[i]) > \phi \Sigma (a(x)[i]/a(y)[i])$
- To bound false positives, and ensure true positives are not obscured by noise, need to argue that each test gives good enough estimate of $(a(x)[i]/a(y)[i])$
- In full paper, show that expected error is $\frac{1}{2} \varepsilon \|a(x)\|_1 \|1/a(y)\|_1$. So with constant probability this is good estimate of the change.
- The group structure amplifies this probability to $1-\delta$

Results

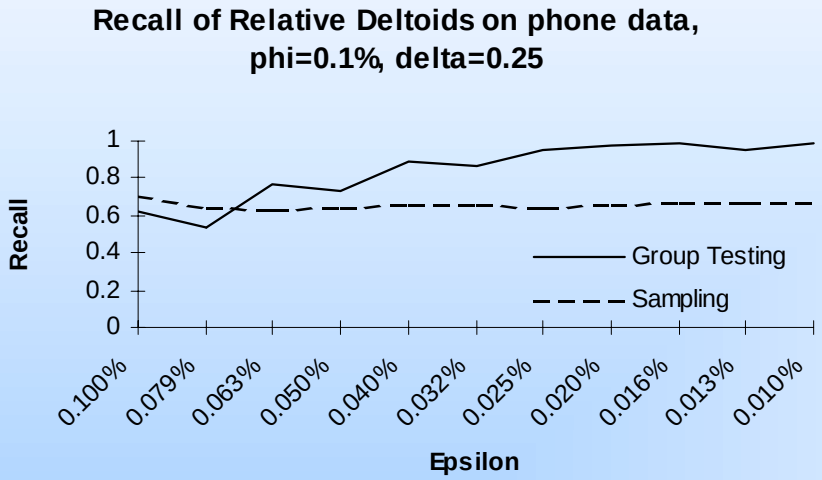
MassDAL
Massive Data Analysis Lab

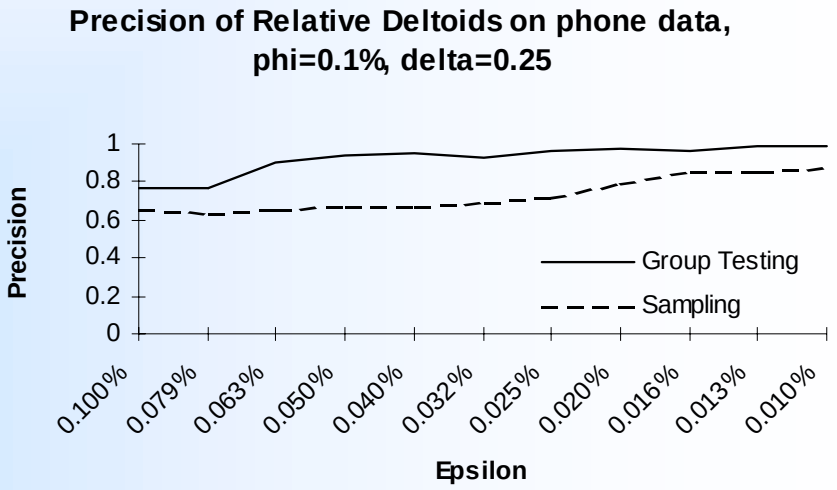
- With probability $1-\delta$, all deltoids are found, no items which are far from being deltoids
- Space is $O(1/\varepsilon \log n \log 1/\delta)$
  Update time is $O(\log n \log 1/\delta)$ per item
  Time to search is linear in the space used
- The same group structure works for different objective functions, if there is an efficient test.

Experiments

MassDAL
Massive Data Analysis Lab

Relative Changes





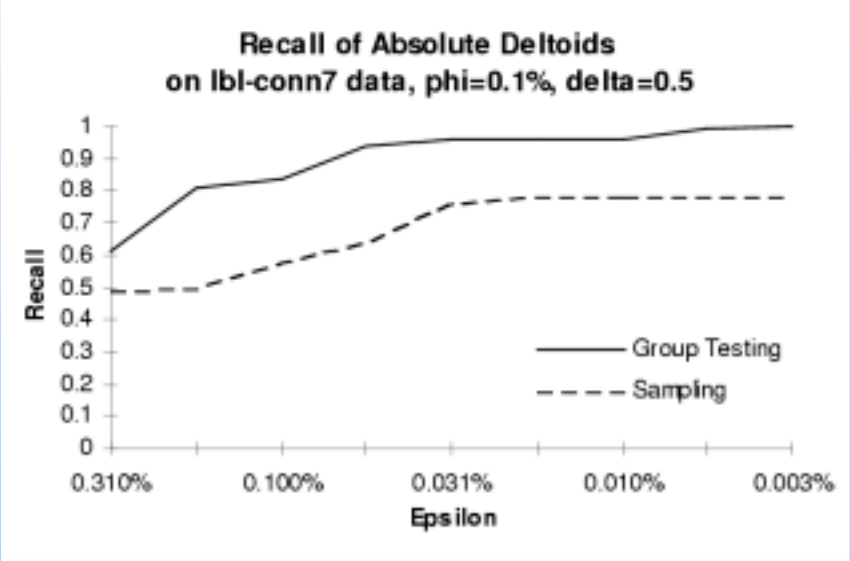
Recall = fraction of deltoids found

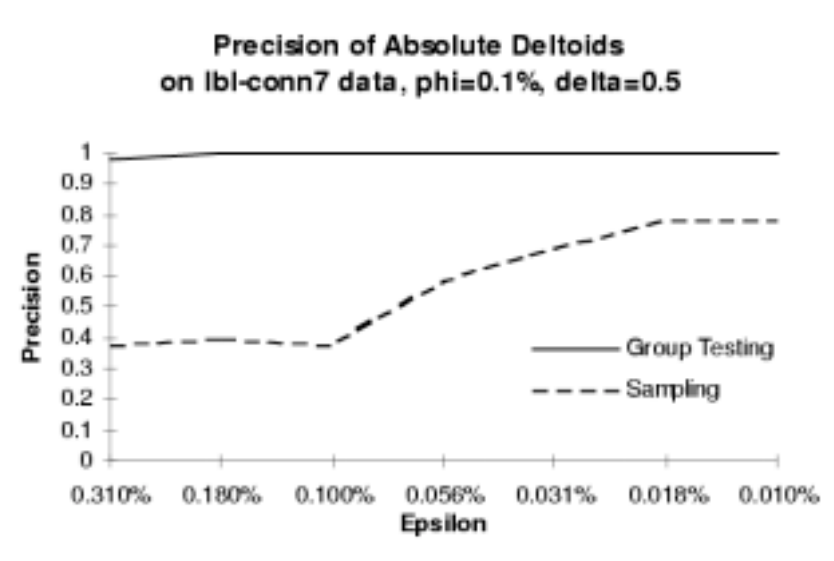
Precision = fraction of returned items that are deltoids

Experiments

MassDAL
Massive Data Analysis Lab

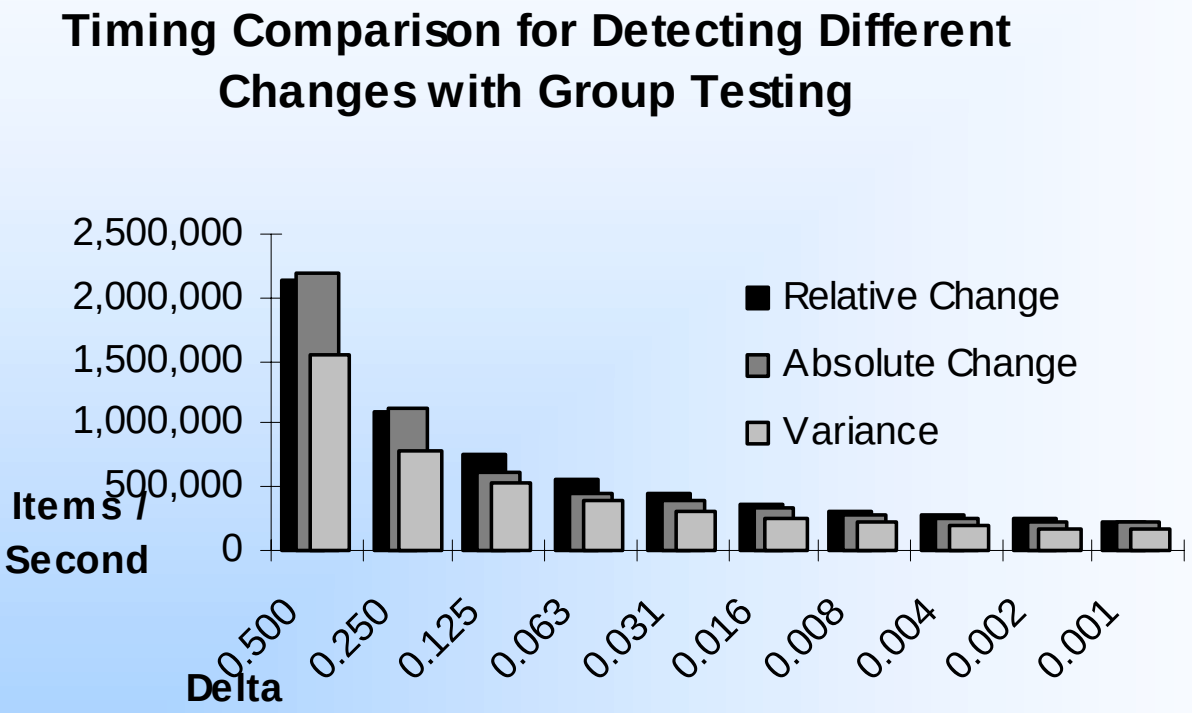
Absolute Changes





Experiments

MassDAL
Massive Data Analysis Lab



Experiments run on lightly loaded 2.4GHz PC

Conclusions

MassDAL
Massive Data Analysis Lab

- Fast, efficient way to keep summaries of observed traffic.
- Items with large change in behavior can be recovered easily.
- Easy to add, subtract, scale summaries to find changes from average or other prediction models.
- Gives a new tool for network data analysis

MassDAL
Massive Data Analysis Lab

Probability Calculation

MassDAL
Massive Data Analysis Lab

- Error variable $X_{ij} = T(x)[j]*T(y)[j] - (a(x)[i]/a(y)[i])$

and let $p = Pr[h(i) = h(j)] = 1/\#groups = \varepsilon/2$

$$E(X_{ij}) = E(T(x)[j]*T(y)[j] - (a(x)[i]/a(y)[i]))$$
$$= (a(x)[i] + a(x)[j] \mid h(j) = h(i))*$$
$$(1/a(y)[i] + 1/a(y)[j] \mid h(j) = h(i))$$
$$- (a(x)[i]/a(y)[i])$$

$$\leq a(x)[i]*p*\Sigma\, 1/a(y)[j] + 1/a(y)[i]*p*\Sigma\, a(x)[j]$$
$$+ p*(\Sigma_{j\neq i}\, a(x)[j])*(\Sigma_{j\neq i}\, 1/a(y)[j])$$

$$\leq p(\Sigma a(x)[i])*(\Sigma 1/a(y)[i]) = \varepsilon \|a(x)\|_1 \, \|1/a(y)\|_1/2$$

Details

MassDAL
Massive Data Analysis Lab

- Error term is $\varepsilon \|a(x)\|_1 \|1/a(y)\|_1$ not $\Sigma\ (a(x)[i]/a(y)[i])$ — but the latter is not possible in small space
- Requires one of the streams to be aggregated and reformatted, to compute $1/a(y)$.
- No problem if streams are naturally aggregated (eg SNMP data)
- Scenario: enough space to capture one stream, then "compress" into Group Testing data structure for later comparison and analysis with new streams

Data Stream Model

MassDAL
Massive Data Analysis Lab

- Stream defines a vector x[1..U], initially all 0
  In networks $U = 2^{32}$ or 2^{64}, too big to store

- Stream of updates (i, c_j): $x[i] = x[i] + c_j$
  Each packet is an update: i = IP address, c_j = size