Tracking Frequent Items Dynamically:

”What’s Hot and What’s Not”

To appear in PODS 2003

Graham Cormode

graham@dimacs.rutgers.edu

dimacs.rutgers.edu/~graham

S. Muthukrishnan

muthu@cs.rutgers.edu

Everyday Uses of Complexity

Background: A does not believe **B** is telling the truth, so **A** sets a trap.

A: Did you do the one we always called the "Hell Paper". You know the one, where we prove P = NP?

B: I did that! I proved P = NP! I placed near the top of the class, and the professor used my paper as an example!

A: You proved P = NP?

B: Yes!

http://kode-fu.com/shame/2003_04_06_archive.shtml

Outline

- **Problem definition and lower bounds**
- Finding Heavy Hitters via Group Testing
  - Finding a simple majority
  - Non-adaptive Group Testing
- Extensions

Frequent Items

- We see a sequence of items defining a bag
- Bag initially empty
- Items can be inserted or removed
- Problem: find items which occur more than some fraction ϕ of the time

Scenario

- Universe 1…n, represent bag as vector a
- +i means insert item i, so add 1 to a[i]
- -i means remove item i, so decrement a[i]
- Only interested in "hot" entries > $\phi\|a\|_1$

Goal: Small Space, Small Time

- Simple solution: keep a heap, update count of each item as it arrives
- Low time cost, but very costly in space
- Output size is $1/\phi$, so why keep n space?
- Want small space, small time solutions

A Streaming Problem

- The scenario fits into “streaming model”, currently a hot area
- Models data generated faster than our capacity to store and process it
- Streaming algorithms are fast, small space, one pass: useful outside a streaming context
- Related to online algorithms, communication complexity

Arrivals Only

- Recent õ(1/ϕ) space solns for arrivals only:
  Deterministic:Karp,Papadimitriou,Shenker03,Manku, Motwani02, Demaine,LopezOrtiz,Munro02
  Randomized: Charikar, Chen, Farach-Colton 02

- Removals bring new challenges: suppose $\phi=1/5$, and bag has 1 million items.

- Then all but 4 are removed – must recover the 4 items exactly

Challenge of Removals

- Existing arrival-only solutions depend on a monotonicity property
- A new arrival can only make the arriving item hot.
- But a removal of an item can make other items become hot
- Can’t backtrack on the past without explicitly storing the whole sequence

Lower bounds

Encode a bit vector as updates, so $a[i] = \{0,1\}$

Space used by some algorithm for $\phi = ½$ is M

Pick some i, send $||a||_1$ copies of $+i$

i is now a hot item iff $a[i]$ was originally 1

$\Rightarrow$ Can extract the value of any bit.

So $M = \Omega(n)$ bits for vector of dimension n,

similar argument follows for arbitrary ϕ

Our solutions

- Avoid lower bounds using probability and approximation.
- Describe solution based on non-adaptive group testing
- Briefly, extensions and open problems.

Small Space, High Time

- Many stream algorithms use embedding-like solutions, inspired by Johnson-Lindenstrauss lemma
- Alon-Matias-Szegedy sketches can be maintained for vector a
- Keep $Z = a[i]*h(i)$, where $h(i)=\{+1,-1\}$, h drawn from pairwise-independent family
- $E(Z*h(i))=a[i]$, and $Var(Z*h(i)) < \|a\|_2^2$

Problems with this

- Small space, for hot items can make good estimator of frequency, updates are fast
- But… how to retrieve hot items?
- Have to test every i in 1…n – too slow (can you do better?)
- Need a solution with small space, fast update and fast decoding

Outline

- Problem definition and lower bounds
- Finding Heavy Hitters via Group Testing
  - **Finding a simple majority**
  - Non-adaptive Group Testing
- Extensions

Non-adaptive Group Testing

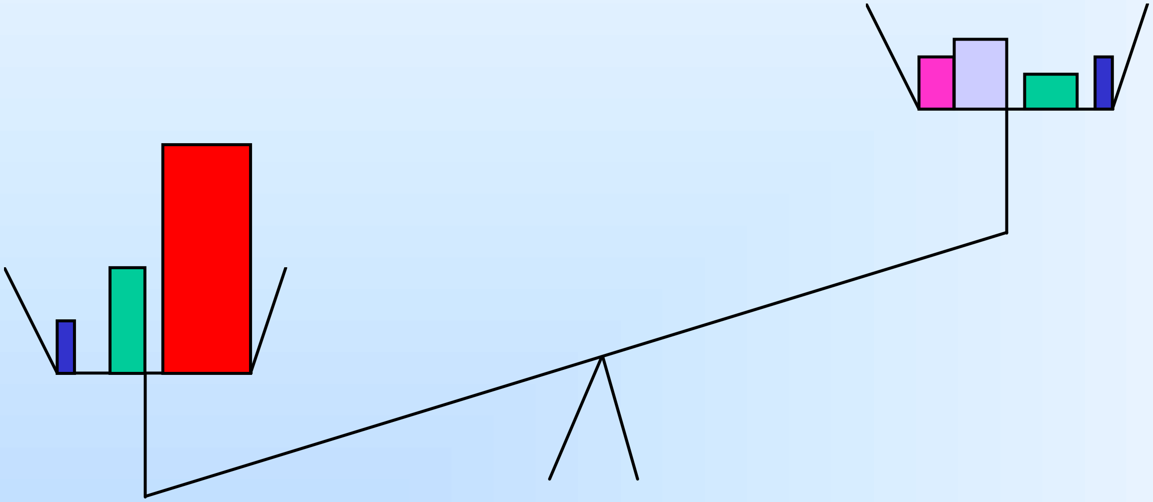
Formulate as group testing.

Arrange items 1..n into (overlapping) groups, keep counts for each group. Also keep $||a||_1$.

Special case: ϕ = ½. At most 1 item a[i] > ½ $||a||_1$

Test: If the count of some group > ½ $||a||_1$ then the hot item must be in that group.

Weighing up the odds



If there is an item with weighing over half the total weight, it will always be in the heavier pan...

Log Groups

- Keep log n groups, one for each bit position
- If j'th bit of i is 1, include item i in group j
- Can read off index of majority item
- log n bits clearly necessary, get 1 bit from each counter comparison.
- Order of arrivals and departures doesn't matter, since addition/subtraction commute

Outline

- Problem definition and lower bounds
- Finding Heavy Hitters via Group Testing
  - Finding a simple majority
  - **Non-adaptive Group Testing**
- Extensions

Group Testing

Extend this approach to arbitrary ϕ

Need a construction of groups so can use "weight" tests to find hot items.

Specifically, want to find up to $k = 1/\phi$ items

Find an arrangement of groups so that the test outcomes allow finding hot items

Additional properties

Want the following three additional properties

- (1) Each item in $O(1/\phi$ poly-log $n)$ groups (small space)

- (2) Generating groups for item is efficient (rapid update)

- (3) Fast decoding, $O(\mathrm{poly}(1/\phi, \log n))$ time (efficient query)

State of the Art

Deterministic constructions use superimposed codes of order k, from Reed-Solomon codes. Brute force $\Omega(n)$ time decoding – fail on (3).

Open Problem 1. Construct efficiently decodable superimposed codes of arbitrarily high order (list decodable codes?).

Open Problem 2. Or, directly construct these “k-separating sets” for group testing.

Randomized Construction

- Use randomized group construction (with limited randomness)
- Idea: generate groups randomly which have exactly 1 hot item in whp
- Use previous method to find it
- Avoid false negatives with enough repeats, also try to limit false positives

Randomized Construction

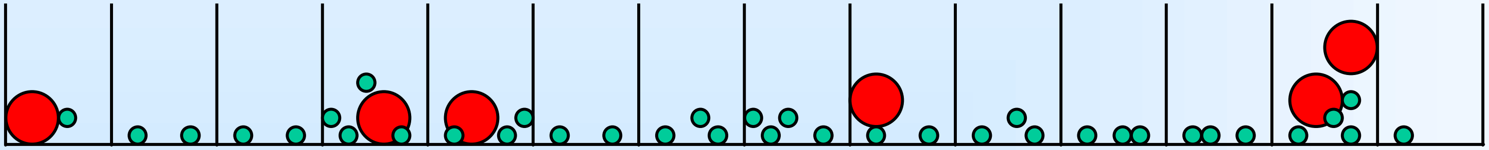
- Partition universe uniformly randomly to c/ϕ groups spreads out hot items, $c > 1$
- Include item i in group j with probability ϕ/c
- Repeat $\log 1/\phi$ times, hot items spread whp
- Storing description of groups explicitly is too expensive

Small space construction

- Pairwise independent hash function suffices
- Range of hash fn is $2/\phi$, defines $2/\phi$ groups, group j holds all items i such that $h(i)=j$
- In each group keep $\log n$ counters as before – easy to update counts for inserts, deletes
- If a hot item is majority in group, can find it

Multiple Buckets

Intuition: Multiple buckets spread out items



- Hot items are unlikely to collide

- Isn’t too much weight from other items

So, there’s a good chance that each hot item will be in the majority for its bucket

Search Procedure

If group count is $> \phi \|a\|_1$ assume hot item is in there, and search subgroups

For each of log n splits, reject some bad cases:

- if both halves of the split $> \phi\|a\|_1$, could be 2 hot items in the same set, so abort
- if both halves of the split $< \phi\|a\|_1$, cannot be hot item in the set, so abort
- Else, find index of candidate hot item

Recap

- Find heavy items using Group Testing
- Spread items out into groups using hash fns
- If there is 1 hot item and little else in a group, it is majority, find using log groups
- Want to analyze probability each hot item lands in such a group (so no false negatives)
- Also want to analyze false positives

Analysis

For each hot item, can identify if its group does not contain much additional weight.

That is, if total other weight $\leq \phi \, ||a||_1$ it is majority

By pairwise independence, linearity of expectation, expected weight in same bucket:

$$E(wt) \leq \Sigma \, a[i]\phi/2 \leq \phi ||a||_1/2$$

By Markov inequality, $\Pr[wt < \phi \, ||a||_1] > ½$

Constant probability of success.

Analysis

Repeat for log $1/(\phi\delta)$ hash functions, gives probability $1 - \delta$ every hot item is in output

Some danger of including an infrequent item in output

Probability of this bounded in terms of the item which is output.

For each candidate, check each group it is in to ensure every one passes threshold.

Time cost

- (1) Space: $O(1/\phi \log(n) \log 1/(\phi\delta))$
- (2) Update time: Compute $\log 1/(\phi\delta)$ hash functions, update $\log(n) \log 1/(\phi\delta)$ counters
- (3) Decode time: $O(1/\phi \log(n) \log 1/(\phi\delta))$
- Can specify $\phi' > \phi$ at query time
- Invariant for order of updates

False Positives

Analysis is similar to before, but guarantees are weaker, eg

Suppose output item w/count $< \phi \, ||a||_1/4$

Every group with that item has $wt > 3 \, ||a||_1/4$

$Pr[wt>3E(wt)/2]<2/3$ in each group, so prob:

$$(2/3)^{-\log \phi\delta} < (\phi\delta)^{0.585} < (\phi\delta)^{1/2}$$

Improved guarantees

False positives may not be a problem, but if they are:

- Probability reduced by increasing the range of hash functions (number of buckets)
- Set number of buckets = $2/\varepsilon$, then probability of outputting any item with frequency less than $(\phi - \varepsilon)$ is bounded by δ
- Increases space, but update time same

Motivating Problems

- Databases need to track attribute values that occur frequently in a column for query plan optimization, approximate query answering.
- Find network users using high bandwidth as connections start and end, for charging, tuning, detecting problems or abuse.
- Many other problems can be modeled as tracking frequent items in a dynamic setting.

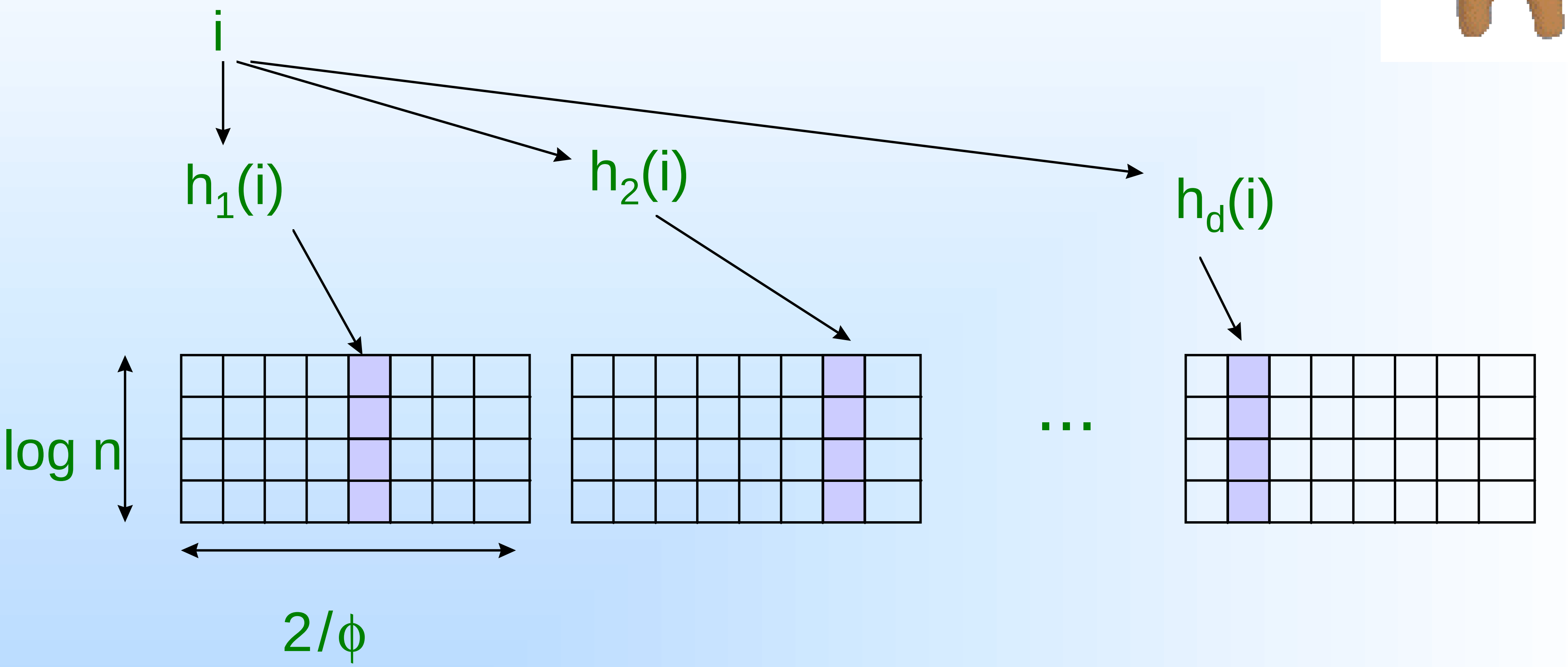
Implementation Issues

- Want solutions to work fast – at packet speeds in networks?

- Estan, Varghese 02 describe hardware solutions for inserts only, fixed threshold case based on fully independent hashes

- Group Testing is suited for hardware implementation: each hash function can be parallelized.

Hardware Issues



Could fully parallelize operation in hardware, with sufficiently flexible memory

Experiments

Wanted to test the recall and precision of the different methods

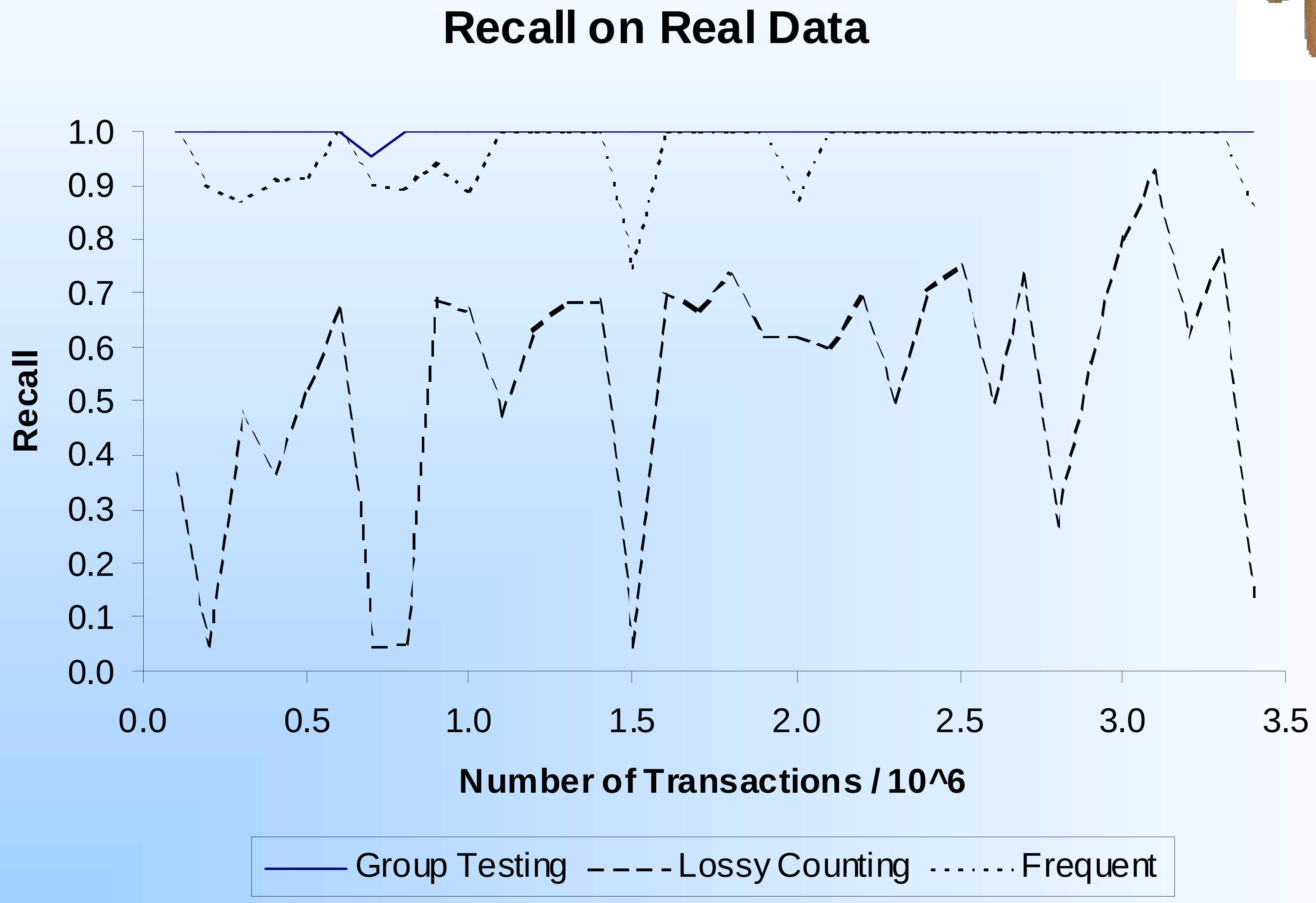
Recall = % of frequent items found

Precision = % of found items frequent

A relatively small experiment... processed a few million phone calls (from one day)

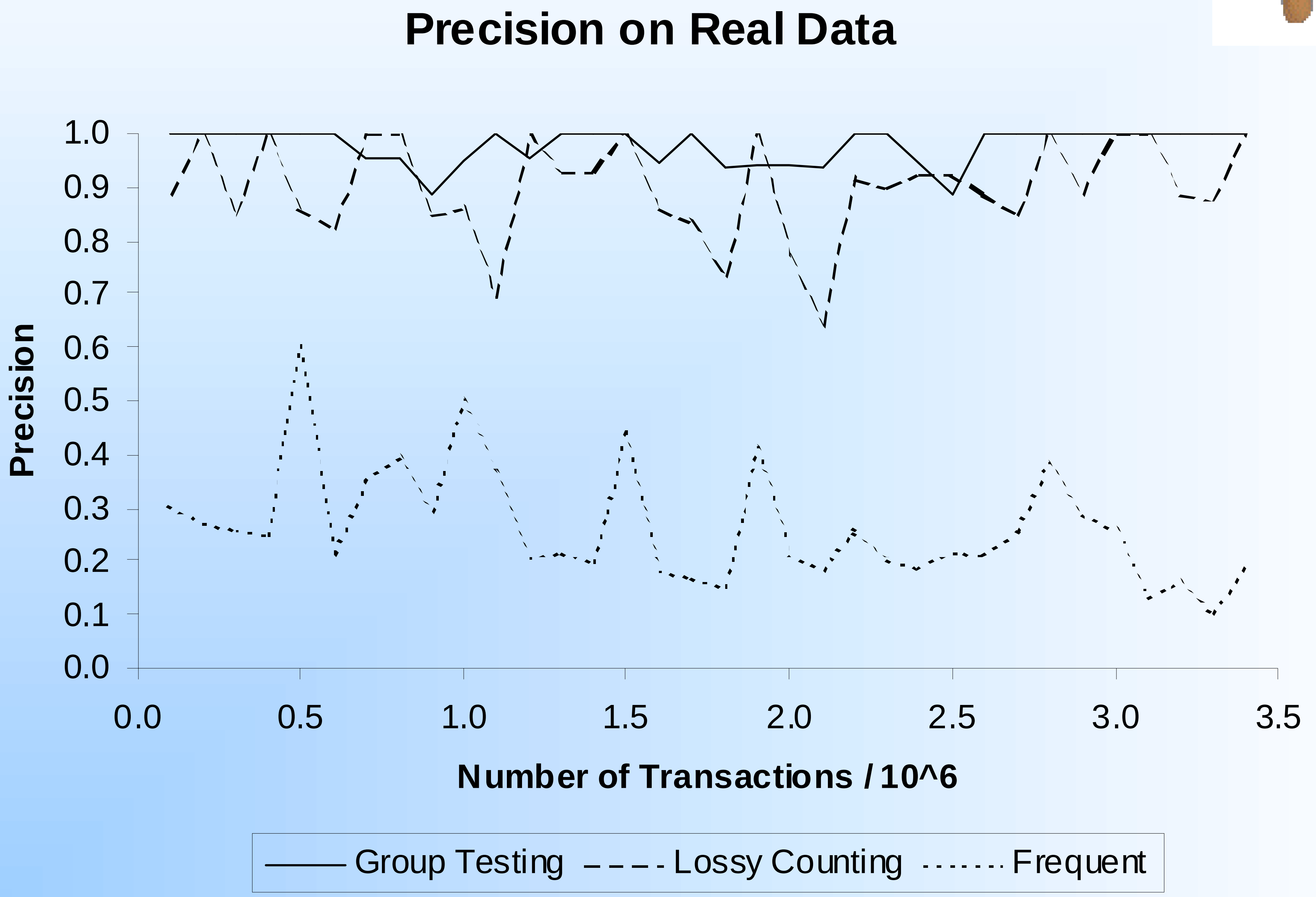
Compared to algorithms for inserts only, modified to handle deletions heuristically.

Recall



Recall on Real Data
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Recall
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
Number of Transactions / 10^6
Group Testing
Lossy Counting
Frequent

Precision



Precision on Real Data
Precision
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
Number of Transactions / 10^6
Group Testing
Lossy Counting
Frequent

Outline

- Problem definition and lower bounds
- Finding Heavy Hitters via Group Testing
  - Finding a simple majority
  - Non-adaptive Group Testing
- **Extensions**

Tracking Changes

- We see two sequences of updates, a and b, representing (say) two day's items

- Which items had biggest absolute change, |a[i] – b[i]|?

- Solved in 2 passes (using AMS-like sketches) by Charikar, Chen, Farach-Colton '02

Absolute Changes

- Can only be $1/\phi$ items with change greater than $\phi \, ||a - b||_1$
- Non-adaptive group testing solution should work immediately, in one pass.
- Replace argument about expected weight with expected absolute change

Relative Changes

- Which had biggest relative change, a[i]/b[i]? (open problem in CCFC02)
- If have b explicitly, set (1/b)[i]=1/b[i]
- Aim to find i where a[i]*(1/b[i]) = a[i]/b[i] is “large”
- Use sketches to approximate $\pi_i(a) \bullet \pi_i(1/b)$ for carefully chosen projections π_i

Relative Changes

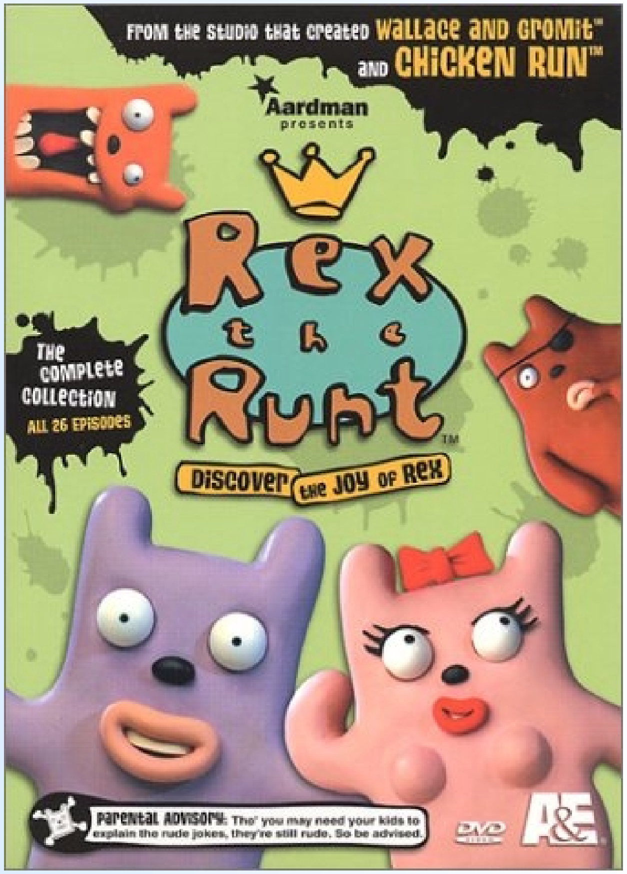
- Open Problem 3. Find large relative changes when input not nicely presented
- What about other notions of changes?
- Work in progress: find items which have highest variance in counts over K days

Open Problems

- Derandomization of these methods – is randomness really necessary?
- Particularly, fast group testing decoding
- Hot items used by practitioners to isolate “outliers” – is this the right notion?
- How to find with high variance, unusual distribution, changes in distribution instead?

Rex the Runt

- British animation from Aardman Animations
- Available on DVD
- Highly recommended by me!


FROM THE STUDIO THAT CREATED WALLACE AND GROMIT™
AND CHICKEN RUN™
Aardman
presents
Rex the Runt™
THE COMPLETE COLLECTION
ALL 26 EPISODES
DISCOVER THE JOY OF REX
PARENTAL ADVISORY: Tho' you may need your kids to explain the rude jokes, they're still rude. So be advised.
DVD
A&E