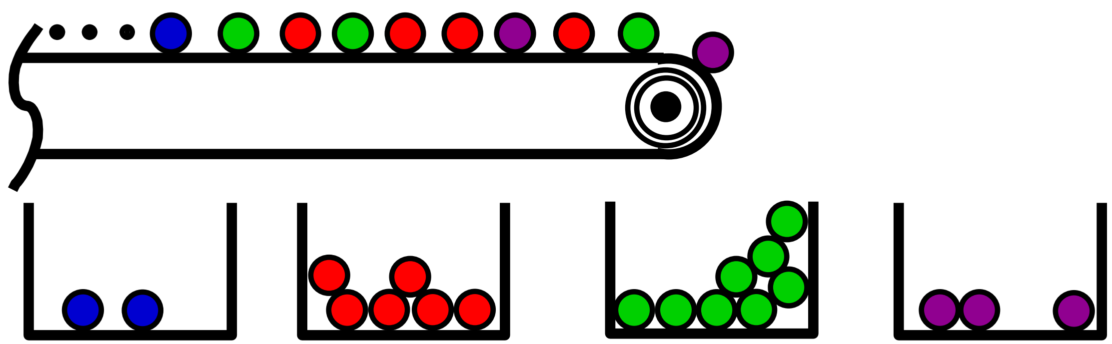
# Space-optimal Heavy Hitters with Strong Error Bounds

**Graham Cormode**
graham@research.att.com

Radu Berinde (MIT)
Piotr Indyk (MIT)
Martin Strauss (U. Michigan)

# The Frequent Items Problem

- The Frequent Items Problem (aka Heavy Hitters): given stream of $N$ items, find those that occur most frequently
- E.g. Find all items occurring more than 1% of the time
- Formally "hard" in small space, so allow approximation
- Find all items with count $\geq \phi N$, none with count $< (\phi - \varepsilon)N$
  - Error $0 < \varepsilon < 1$, e.g. $\varepsilon = 1/1000$
  - Core subproblem: estimate each frequency accurately



# Why Frequent Items?

- A natural question on streaming data
  - Track bandwidth hogs, popular destinations etc.
- The subject of much streaming research
  - Scores of papers on the subject
- A core streaming problem
  - Many streaming problems connected to frequent items (itemset mining, entropy estimation, compressed sensing)
- Many practical applications
  - Search log mining, network data analysis, DBMS optimization

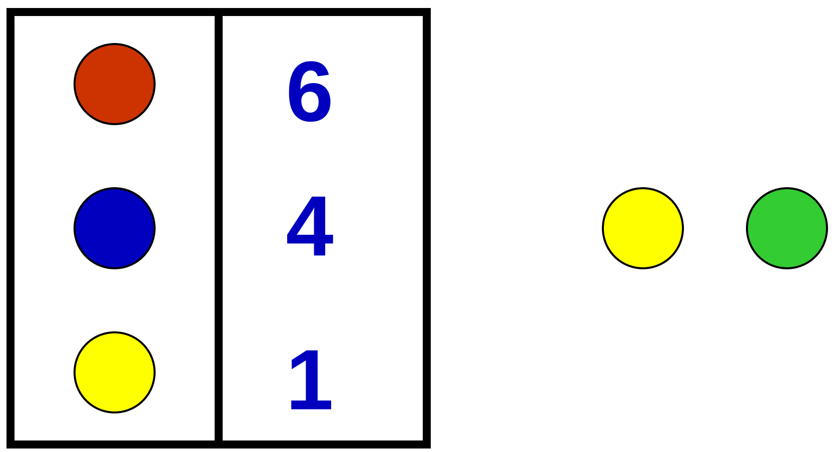
# Prior Work

- Counter-based algorithms accept a stream of arrivals
  - Frequent (1982/2002), Lossy Counting (2002), SpaceSaving (2005)
  - Described in more detail
- Sketch-based algorithms allow arrivals and departures
  - Count Sketch (2002), Count-Min Sketch (2003)
- See survey and experimental study in VLDB 2008
- So why are we still talking about frequent items?

# Better than advertised

- Experimentally counter algorithms seem better than expected
  - Accuracy much higher than the bounds would suggest
- We can analyze them to show data-dependent bounds
  - For skewed data (common case) much improved guarantees
- Implications for a variety of applications:
  - K-sparse recovery (find best sparse representation)
  - Top-k frequency estimation
  - Estimating confidence of functional dependencies (SIGMOD ’09)

# “Frequent” algorithm

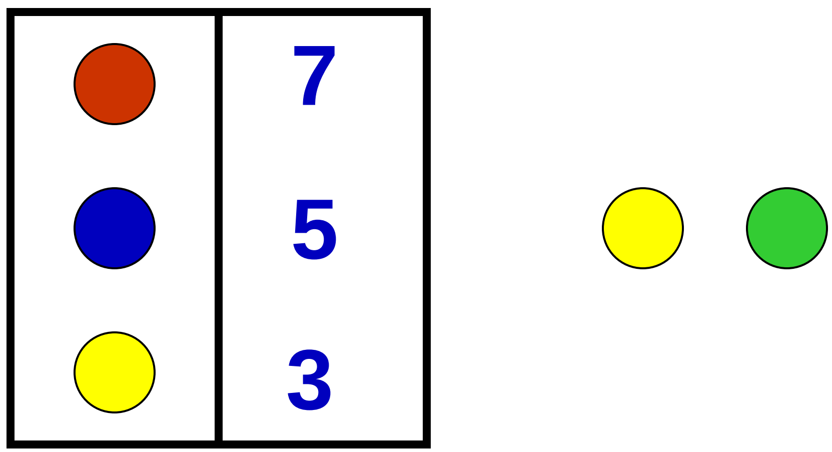



- FREQUENT finds up to k items that occur more than 1/k fraction of the time
- Keep k different candidates in hand.  For each item in stream:
  - If item is monitored, increase its counter
  - Else, if < k items monitored, add new item with count 1
  - Else, decrease all counts by 1

# Frequent Analysis

- Analysis: each decrease can be charged against k arrivals of different items, so no item with frequency N/k is missed
- Moreover, $k=1/\varepsilon$ counters estimate frequency with error $\varepsilon N$
  - Not explicitly stated until later [Bose et al., 2003]
- Some history: First proposed in 1982 by Misra and Gries, rediscovered twice in 2002
  - Later papers showed how to make fast implementations

# SpaceSaving Algorithm




- "SpaceSaving" algorithm [Metwally, Agrawal, El Abaddi 05] has the same space/accuracy bounds
- Keep $k = 1/\varepsilon$ item names and counts, initially zero
  Count first $k$ distinct items exactly
- On seeing new item:
  - If it has a counter, increment counter
  - If not, replace item with least count, increment count

# SpaceSaving Analysis

- Smallest counter value, min, is at most $\varepsilon N$
  - Counters sum to N by induction
  - $1/\varepsilon$ counters, so average is $\varepsilon N$: smallest cannot be bigger
- True count of an uncounted item is between 0 and min
  - Proof by induction, true initially, min increases monotonically
  - Hence, the count of any item stored is off by at most $\varepsilon N$
- Any item x whose true count $> \varepsilon N$ is stored
  - By contradiction: x was evicted in past, with count $\leq min_t$
  - Every count is an overestimate, using above observation
  - So est. count of $x > \varepsilon N \geq min \geq min_t$, and would not be evicted

So: Find all items with count $> \varepsilon N$, error in counts $\leq \varepsilon N$

# Improving the Bounds

- Define a class of "heavy tolerant" counter algorithms
  - An algorithm which stores $m$ items and counts
  - Extra occurrences of an item do not increase estimation error
  - A relatively intuitive and natural property
- Prove that both Frequent and SpaceSaving are heavy tolerant
  - A little intricate, requires careful case analysis
- Show that heavy tolerance implies a k-tail guarantee
  - Define $f_1$ = highest frequency, $f_2$ = second highest, etc.
  - Then define $F_1^{res(k)} = N - (f_1 + f_2 + \ldots f_k)$, $\ll N$ for skewed dbns
  - Accuracy of estimates is $F_1^{res(k)}/(m - Bk)$ for some $B$

# Results on Tail Bounds

- General result: for all counter-based algorithms, $B \leq 2$
- Specific results: $B = 1$ for SpaceSaving and Frequent
- With $m$ counters, get accuracy $F_1^{res(k)}/(m - Bk)$ for any $k < m$
  - So with $m = O(k)$ counters, get accuracy $F_1^{res(k)}/k$
  - Much better than prior $F_1/k$ accuracy for skewed distributions
  - Only need $O(\varepsilon^{-1/z})$ counters for $\varepsilon N$ accuracy on Zipfian ($z$) data
- k-tail bounds are optimal: can force $F_1^{res(k)}/2m$ error

# Frequent Tail Bound Analysis

- Conceptually, each arrival increments a counter for that item
- Over the stream, $d$ times an element decrements $d$ counters
- Sum of counters $C = N - d(m+1)$
- Error in estimated count of an item is at most $d$
- Consider sum of estimated counts of $k$ most frequent items:
  - $N - d(m+1) \geq \sum_{i=1}^{k} (f_i - d) = -dk + (N - F_1^{res(k)})$
  - Rearranging and simplifying, $d(m+1-k) \leq F_1^{res(k)}$
- So $d$, error in count, is at most $F_1^{res(k)}/(m+1-k)$ : $k$ tail with $B=1$

# Implications

- k-Sparse recovery: recover a vector f' that approximates f for $p \geq 1$
  - With $m = k(B + 3/\varepsilon)$ counters, top-k counter values define f'
  - Show that $\|f - f'\|_p \leq \varepsilon F_1^{res(k)} k^{1/p-1} + (F_p^{res(k)})^{1/p}$
  - Smallest possible error is $(F_p^{res(k)})^{1/p}$
- m-Sparse recovery: recover a vector f' that approximates f
  - With $m = k(B + 1/\varepsilon)$ counters, all m counter values define f'
  - Show that $\|f - f'\|_p \leq (1+\varepsilon) F_1^{res(k)} (k\varepsilon^{-1})^{1/p-1}$
  - Converges with previous result for $p=1$
- Estimate $F_1^{res(k)}$
  - With $m = k(B + 1/\varepsilon)$ counters, top-k counter values define f'
  - Show that $N - \|f'\|_1 \in (1 \pm \varepsilon) F_1^{res(k)}$

# Weighted Updates

- Weighted case: find items whose total weight is high
  - Sketch algorithms adapt easily, counter algs with effort
- Simple solution: all weights are integer multiples of small $\delta$
- Full solution: define appropriate generalizations of counter algs to handle real valued weights
  - Straightforward to extend SpaceSaving analysis to weighted case
  - Frequent more complex, action depends on smallest counter value
- Result: both algorithms still provide B=1 tail guarantees
  - Even on real valued non-negative update streams

# Mergability of Summaries

- Want to merge summaries, to summarize the union of streams
- Sketches with shared hash fns are easy to merge together
- Counter-based algorithms need new analysis:
  - Merging two summaries preserves accuracy, but space may grow
  - With pruning of the summary, can merge indefinitely
  - Space remains bounded, accuracy degrades by at most a constant
- Result: Given m counters, algorithms provide similar guarantees
  - Accuracy behaves like $3/(m - (B+1)k)\ F_1^{res(k)}$ on merged summaries
  - Grow summaries by a constant factor to get same accuracy

# Conclusions

- Finding the frequent items is one of the most studied problems in data streams
- We analyzed a broad class of counter-based algorithms, and showed improved (optimal) worst-case bounds
  - Can replace sketches with deterministic summaries in many cases
  - Results much more compact, accurate, reliable
- For gory details of analysis, see the paper