ECEN 689
Special Topics in Data Science for Communications Networks

Nick Duffield
Department of Electrical & Computer Engineering
Texas A&M University

Lecture 11
Probabilistic Counting and the Morris Algorithm

Acknowledgement: adapted partly from slides by Edith Cohen

Probabilistic Counting in a Stream

1, 1, 1, 1, 1, 1, 1, 1,

- Simple counting
  - Accumulate count in $\log_2(n)$ bits where n is the current count
- Can we use fewer bits? Important when we have many streams to count, fast memory is scarce (e.g. inside a backbone router)
- Can we reduce storage size if an approximate count suffices?

Probabilistic Counting in a Stream

a, b, b, c, b, b, b, c,

- Counting multiple keys: n_a, n_b , n_c etc.
- Can we tune counting to focus resources on “important” keys
  - Frequent keys
- Example:
  - Packet stream; focus on large flows (high counts n)

Outline

- Morris counting algorithm
- Frequent element counting
- Concise samples
- Counting samples
- Sample and hold

Morris Algorithm 1978

1, 1, 1, 1, 1, 1, 1, 1,

- The first streaming algorithm
  - Stream of positive increments
- Idea
  - Track log n instead of n
  - Use log log n bits instead of log n bits

Deterministic Approach?

1, 1, 1, 1, 1, 1, 1, 1,

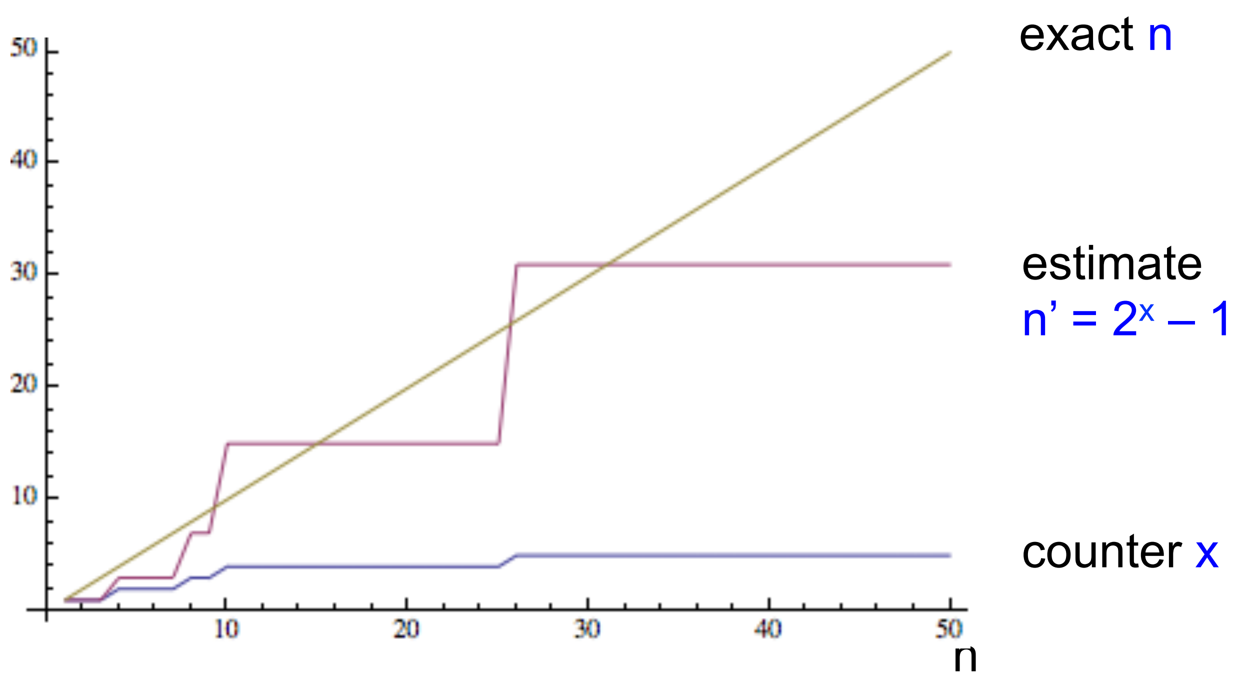
- Can we simply maintain a count of $\log_2 n$?
  - using $\log_2 \log_2 n$ bits
- Problem
  - We are actually maintaining integer part $x = \text{floor}(\log_2 n)$
  - Fractional part of $\log_2 n$ is lost

n	1	2	3	4	5	6	7	8	9
floor($\log_2 n$)	0	1	1	2	2	2	2	3	3

- When to increment x?

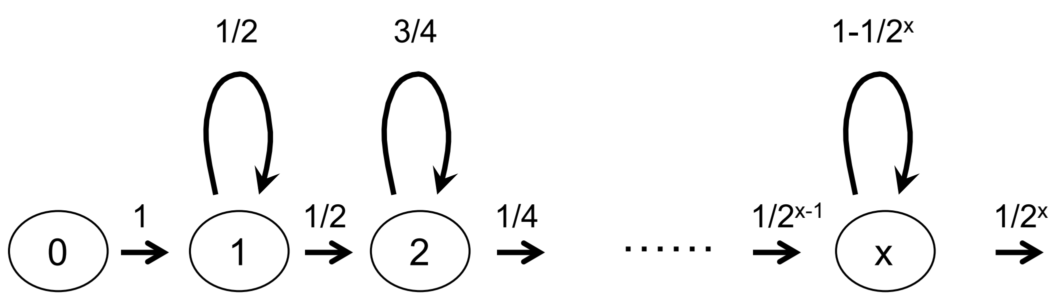
Morris Algorithm

- Maintain a "log" counter x
- Initialize to 0
- Each arrival:
  - increment with probability $p_x = 2^{-x}$
- Query: output estimate $n' = 2^x - 1$



Morris Algorithm: Birth Process

- Let X(n) denote count after arrival n
- Pure birth process
  - Transition x→x+1 with probability 2^{-x}



Morris Algorithm: Unbiasedness

- Initialize $x = 0$; increment w.p. $p_x = 2^{-x}$; estimate $n' = 2^x - 1$

- $n = 1$
  - before: $x = 0$ $p_0 = 1$;
  - prob. 1: $x \rightarrow 1$
  - estimate $n' = 2^1 - 1 = 1 = n$
- $n = 2$;
  - before: $x = 1$; $p_1 = ½$
  - prob. ½: x stays at 1; $n' = 2^1 - 1 = 1$
  - prob. ½: $x \rightarrow 2$. $n' = 2^2 - 1 = 3$
  - $E[n'] = ½ * 1 + ½ * 3 = 2 = n$

Morris algorithm: general case

- Let $X(n)$ denote random counter x after n^{th} arrival
- Initialize $X(0) = 0$; increment w.p. $p_x = 2^{-x}$
- Estimate $n' = 2^{X(n)} - 1$

- $$\begin{aligned} E[2^{X(n)}] &= \Sigma_{j=1,\ldots,n-1} \Pr[X(n-1) = j]\, E[2^{X(n)} \mid X(n-1) = j] \\ &= \Sigma_{j=1,\ldots,n-1} \Pr[X(n-1) = j]\, (p_j 2^{j+1} + (1- p_j)\, 2^j) \\ &= \Sigma_{j=1,\ldots,n-1} \Pr[X(n-1) = j]\, (2^j + 1) \\ &= E[2^{X(n-1)}] + 1 \end{aligned}$$

- Iterating: $E[2^{X(n)}] = E[2^{X(0)}] + n = 1 + n$
- Therefore: $E[2^{X(n)} - 1] = n$

- Conclusion: $n' = 2^{X(n)} - 1$ is an unbiased estimator of n

Morris algorithm: variance

- $Var[n'] \quad = Var[2^{X(n)} - 1] = Var[2^{X(n)}]$
  $= E[2^{2X(n)}] - E[2^{X(n)}]^2$
  $= E[2^{2X(n)}] - (n+1)^2$

- $E[2^{2X(n)}] \quad = \Sigma_{j=1,\ldots,n-1} Pr[X(n-1) = j\,]\, E[2^{2X(n)} \mid X(n-1) = j\,]$
  $= \Sigma_{j=1,\ldots,n-1} Pr[X(n-1) = j\,]\, (\, p_j\, 2^{2j+2} + (1- p_j)\, 2^{2j})$
  $= \Sigma_{j=1,\ldots,n-1} Pr[X(n-1) = j\,]\, (\, 2^{j+2} + 2^{2j} - 2^{j})$
  $= \Sigma_{j=1,\ldots,n-1} Pr[X(n-1) = j\,]\, (\, 3*2^{j} + 2^{2j})$
  $= 3\, E[2^{X(n-1)}] + E[2^{2X(n-1)}]$
  $= 3\, n + E[2^{2X(n-1)}]$

- Iterate: $E[2^{2X(n)}] \quad = 3\Sigma_{m=1,\ldots,n}\, m \quad + E[2^{2X(0)}]$
  $= 3n(n+1)/2 + 1$
- $Var[n'] = 3n(n+1)/2 + 1 - (n+1)^2 \quad = n(n-1)/2$

Morris algorithm

- Coefficient of Variation = StdDev / Mean ≈ $1/\sqrt{2}$:
  - doesn't improve as n grows
- How to improve?

Morris Algorithm: Reducing Variance 1

- Change base of logarithms $2 \rightarrow b > 1$
- Instead of counting $\log_2(n)$, count $\log_b(n)$
- Increment counter x with probability b^{-x}
  - Method of base 2 analysis caries through
- $E[b^{X(n)}] = (b-1)n + 1$
  - $n' = (E[b^{X(n)}] - 1)/(b-1)$ is an unbiased estimator of n
- $Var[n'] = (b-1)n(n-1)/2$
- By decreasing b closer to 1
  - Decrease variance
  - Increase size of storage needed
    - $b \rightarrow \log_b(n)$ increases

Morris Algorithm: Reducing Variance 2

- Familiar approach
  - Multiple independent estimates
- Mean of estimates
- Median of means

Frequent Element Counting

a, b, b, c, b, b, b, c,

- Elements occur multiple times
- Want to find which elements occur most often
- Stream size n
- m distinct elements

Frequent Elements

a, b, b, c, b, b, b, c,

- Applications
  - Networking: find “elephant” flows
  - Search: find the most frequent queries

- Pareto Principle
  - Typical frequency distributions are highly skewed
  - Small proportion of elements are very frequent

- Zipf’s Law
  - Rank elements by frequency
  - Frequency of rank k element proportional to $1/k^s$, some $s > 1$

Frequent Elements: exact solution

a, b, b, c, b, b, b, c,

- Maintain counter for each distinct element
  - Instantiate on first occurrence
  - Increment on every occurrence
- Problem
  - Need to maintain m counters
  - Generally only have room for $k \ll m$ counters

Frequent Elements: Misra & Gries 1982

a, b, b, c, b, b, b, c,

- Processing an element x
  - If: already have counter for x, increment it
  - Else if: no counter for x, but fewer than k counters, create a counter for x and initialize it to 1
  - Else: decrease all counters by 1. Remove counters containing 0.
- Query: how many times did x occur?
  - If: we have a counter for x, return counter value
  - Else: return 0
- Clearly an underestimate

Misra & Gries: Analysis

a, b, b, c, b, b, b, c,

- For each x: true value – counter = # decrements
- How many possible decrements to counter for x?
- Suppose sum of counters is $n' < n$ = length of stream
- Each decrement step removes k counts
  - Also did not count the current arrival
- Therefore k+1 undercounts from each decrement
  - There are at most $d = (n-n')/(k+1)$ decrement steps

Misra & Gries: Analysis

a, b, b, c, b, b, b, c,

- There are at most $d = (n-n')/(k+1)$ decrement steps
- Counter for x is smaller than count by at most d
  - Good estimates when counter(x) >> d
  - Error bound inversely proportional to k
  - Track n by count (or estimate)
- Works since typical distributions have few frequent elements

Bibliography

- Approximate counting (Morris Algorithm)
  - Robert Morris. Counting Large Numbers of Events in Small Registers. Commun. ACM, 21(10): 840-842, 1978.
    - http://www.inf.ed.ac.uk/teaching/courses/exc/reading/morris.pdf
  - Philippe Flajolet Approximate counting: A detailed analysis. BIT 25 1985
    - http://algo.inria.fr/flajolet/Publications/Flajolet85c.pdf
- Frequent element summaries
  - J. Misra and David Gries, Finding Repeated Elements. Science of Computer Programming 2, 1982
    - http://www.cs.utexas.edu/users/misra/scannedPdf.dir/FindRepeatedElements.pdf