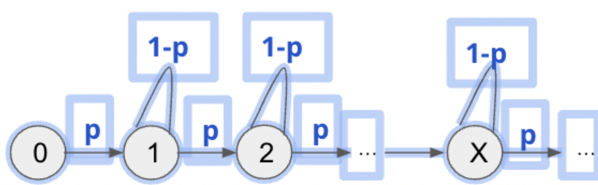
CAS CS 365
Lec 15

Streaming Model (cont.)

1. Algorithms for estimating F_1
   a. $F1 = \sum f1 = m$ (length of stream)
   b. $C \leftarrow 0$
      For x in the stream
         $C \leftarrow c + 1$ (c will be the length of the stream eventually)
   c. $Log_2(m) \rightarrow$ m is huge that $log_2(m)$ is very large as well
   d. Let's say there is a number with 17 digits
   e. Storing the number by only stating the digits (17 digits is also 2 digits)
   f. 17 digits → 2 digits (better but we lose accuracy)
   g. Store the count of digits
      i. 10…0 (16 digits of 0) to 9..9 (17 digits)
   h. Storing the digits of (2) is $log_2(log_2(m))$ bits since $log_2(m) = 17$ and $log_2(log_2(m)) = 2$
2. Morris algorithm
   a. Robert morris
   b. Key idea: instead of maintaining the actual length of the stream m, keep the logarithm
      i. E.g., if m=145, then by knowing the order of magnitude ~10^2, we can tell that our number is between 100 and 999
   c. This allows us to use loglog(m) bits to represent m approximately
3. How to save 1 bit?
   a. Maintain a counter c (aka Morris counter)
   b. int(): $c \leftarrow 0$
   c. process()
      i. For each item in the stream
         1. Increase c with probability ½
         2. o/w keep the same value
      ii. Output estimate 2c → because we increase c with probability of ½
   d. Let z be the value of the counter after m increments
   e. $Z \sim Bin(m, ½)$
      i. $E[z] = m/2$
      ii. $Var[z] = m/4$
      iii. m/2 +- z_score * sqrt(m/4) → sqrt(m) = O(m)
      iv. Space complexity: $Ig(m/2) = lg(m) - 1$ → saved one bit at the cost of accuracy (by halfing the number, we save one digit)

        v. If you want to remove 2 bits, we have to increase c with probs ¼ and etc.

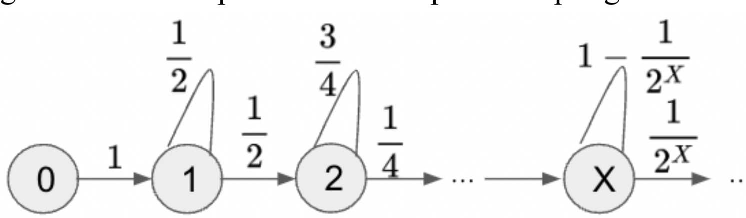
4. How to save k bits?
    a. Maintain a counter c
    b. init(): c ← 0
    c. process()
        i. For each item in the stream
            1. Increase c with probability (½)^k
            2. o/w keep the same value
        ii. Output estimate 2^k * c
    d. Let Z be the value of the counter after m increments
        i. Z~Bin(m, 2^-k)
            1. E[z] = m/2^k
            2. Var[z] ~ m/2^k
            3. Space complexity: lg(m/2^k) = lg(m) - k → saved k bits (in practice, we care about confidence interval → as k increases, there is a higher probability that the error will be large)



    e.
    f. Another perspective as a birth process
    g. Counter values follow a binomial distribution

$$P(C_m = k) = \binom{m}{k} p^k (1-p)^{m-k}$$

5. Morris algorithm → birth process with adaptive sampling



    a.

- Maintain a log-counter **c** (aka Morris counter)
- init(): **c**← 0
- process()
  For each item in the stream
    - Increase **c** with probability $1/2^{c}$
    - o/w keep same value
- Output estimate 2^{c}-1

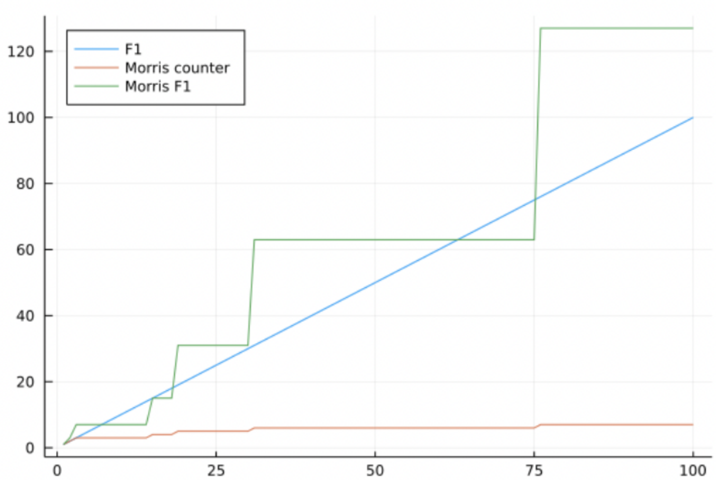
    b.

6. Why this estimator?
    a. Claim: Define Xn to be the value of the counter after n increments. Then, E[2^Xn] = n + 1
    b. **Proof (induction)**

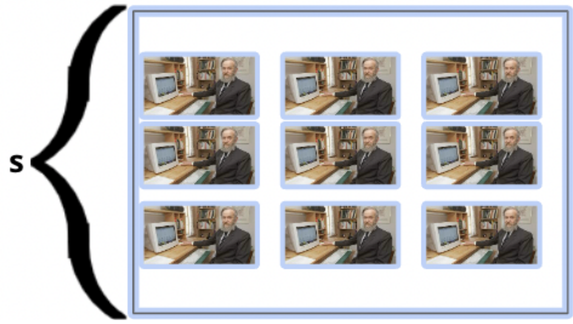
       Base case: If n=0, X_n=0 and thus the claim holds.

       Inductive step: By conditional expectation rule $E[2^{Xn+1}]=E[E[2^{Xn+1}\,|\,X_n]]$ and the inductive hypothesis, we obtain the following expression:

$$
\begin{aligned}
E\left[2^{X_{n+1}}\right] &= \sum_{j=0}^{+\infty} P(X_n = j) E\left[2^{X_{n+1}} \mid X_n = j\right] \\
&= \sum_{j=0}^{+\infty} P(X_n = j)\left[2^j\left(1 - \frac{1}{2^j}\right) + 2^{j+1}\frac{1}{2^j}\right] \\
&= \sum_{j=0}^{+\infty} P(X_n = j)(2^j + 1) = E\left[2^{X_n}\right] + \sum_{j=0}^{+\infty} \Pr(X_n = j) \\
&= E\left[2^{X_n}\right] + 1
\end{aligned}
$$

    c. Base: E[2^0] = 0 + 1 = 1
    d. Ind hyp: E[2^Xn] = n + 1
    e. We need to prove E[2^X_{n+1}] = (n+1) + 1 = n + 2
7. Properties of Morris algorithm
    a. The expectation of the variable Z=2^Xm satisfies the following
        i. E[Z] = m + 1
    b. Corollary: Morris algorithm outputs an unbiased estimator of m
        i. The variance of Z is equal to Var[z] = m*(m-1)/2
    c. Observation: No improvement in terms of concentration as m grows since Var(z)/E(z)^2 is constant
8. Morris algorithm
    a. 

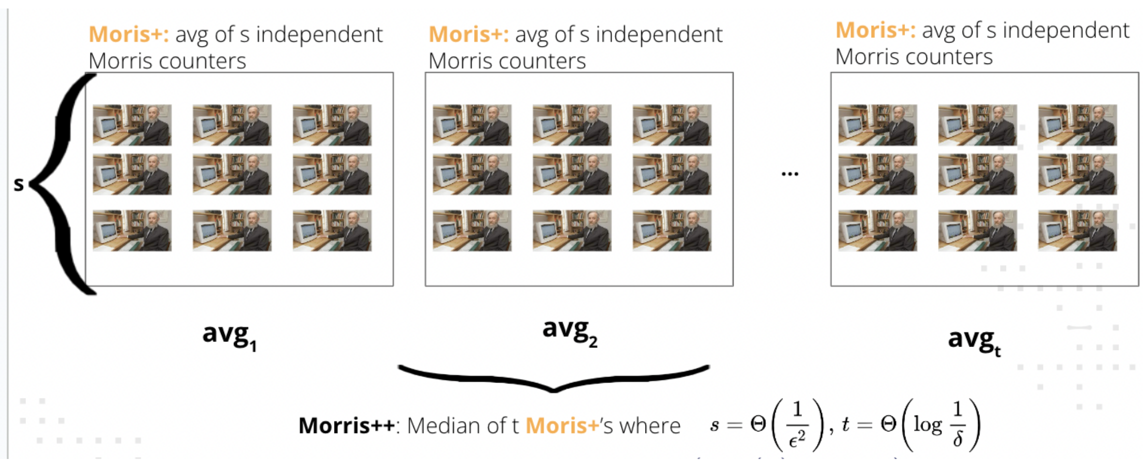
9. Morris+

a. 

b. Suppose we run s Morris counters and we output the average $\frac{1}{s}\sum_{i=1}^{s} m_i$

   i. The average of this estimator remains the same but the variance scales down by a factor of 1/s

c. Reducing variance: Morris ++

d. 

e. Claim: the space complexity with probability 1-δ is $O\left(\frac{1}{\epsilon^2}\lg\left(\frac{1}{\delta}\right)\ \lg\lg\left(\frac{n}{\epsilon\delta}\right)\right)$ and we obtain an (epsilon, delta) – an approximation scheme to F1 fort insert-only streams

10. Optimal Algorithm for F1

   a. Nelson and Yu proved recently that Morris algorithm is optimal by

      i. Tightening the analysis of the space complexity $O\left(\log\log n + \log\frac{1}{\epsilon} + \log\log\frac{1}{\delta}\right)$ for an (epsilon, beta) - approximation scheme to F1

      ii. Proving a tight lower bound, and thus practically nailing down the problem

11. How to set a?
    a. Set a = 2*e^2*(delta) and apply ChebyShev's inequality

$$\Pr(|Z - m| \geq \epsilon m) \leq \frac{Var(Z)}{\epsilon^2 m^2} = \frac{\frac{m(m-1)}{2} 2\delta\epsilon^2}{\epsilon^2 m^2} \leq \delta$$

    b. Space complexity: $O\left(\log\log n + \log\frac{1}{\epsilon} + \log\frac{1}{\delta}\right)$
12. Turnstile model
    a. Attempt: suppose we have a strict turnstile model. Can we use one Morris counter for insertions, and one for deletions and somehow combine them?
    b. No! If z+, z- are the approximate histograms of x+, x- of additions and deletions respectively, |z+-z-|, can be O(epsilon*m) off in terms of additive error from the desired |x+-x-|