Streaming, Sketching and Sufficient Statistics

Graham Cormode
University of Warwick
G.Cormode@Warwick.ac.uk
Data is Massive

- Data is growing faster than our ability to store or index it.
- There are 3 Billion Telephone Calls in US each day (100BN minutes), 30B emails daily, 4B SMS, IMs.
- Scientific data: NASA's observation satellites generate billions of readings each per day.
- IP Network Traffic: can be billions packets per hour per router. Each ISP has many (10s of thousands) routers!
- Whole genome readings for individual humans now available: each is many gigabytes in size.
Small Summaries and Sufficient Statistics

- A summary (approximately) allows answering such questions
- To earn the name, should be (very) small
  - Can keep in fast storage
- Should be able to build, update and query efficiently
- Key methods for summaries:
  - Create an empty summary
  - Update with one new tuple: streaming processing
  - Merge summaries together: distributed processing
  - Query: may tolerate some approximation
- A generalized notion of “sufficient statistics”
The CS Perspective

- **Cynical**: “The price of everything and the value of nothing”
  - Optimize the cost of quantities related to a computation
    - The space required to store the sufficient information
    - The time to process each new item, or answer a query
    - The accuracy of the answer ($\varepsilon$)
    - The amount of “true” randomness
  - In terms of size of input $n$, and chosen parameters

- **Pessimistic**: “A pessimist is never disappointed”
  - Rarely make strong assumptions about the input distribution
  - “the data is the data”: assume fixed input, adversarial ordering
  - Seek to compute a function of the input (not the distribution)
“Probably Approximately Correct”
- Preference for tail bounds on quantities
- Within error $\varepsilon$ with probability $1-\delta$
- Use concentration of measure (Markov, Chebyshev, Chernoff...)

“High price of entr(op)y”: Randomness is a limited resource
- We often need “random” bits as a function of $i$
- Must either store the randomness
- Or use weaker hash functions with small random keys
- Occasionally, assume “fully independent hash functions”

Not too concerned about constant factors
- Most bounds given in $O()$ notation
Data Models

- We model data as a collection of simple tuples.
- Problems hard due to scale and dimension of input.
- Arrivals only model:
  - Example: \((x, 3), (y, 2), (x, 2)\) encodes the arrival of 3 copies of item \(x\), 2 copies of \(y\), then 2 copies of \(x\).
  - Could represent eg. packets on a network; power usage.
- Arrivals and departures:
  - Example: \((x, 3), (y, 2), (x, -2)\) encodes final state of \((x, 1), (y, 2)\).
  - Can represent fluctuating quantities, or measure differences between two distributions.
Part I: Sketches and Frequency Moments

- Frequency distributions and Concentration bounds
- Count-Min sketch for $F_\infty$ and frequent items
- AMS Sketch for $F_2$
- Estimating $F_0$
- Extensions:
  - Higher frequency moments
  - Combined frequency moments
Part II: Advanced Topics

- Sampling and $L_p$ Sampling
  - $L_0$ sampling and graph sketching
  - $L_2$ sampling and frequency moment estimation

- Matrix computations
  - Sketches for matrix multiplication
  - Sparse representation via frequent directions

- Lower bounds for streaming and sketching
  - Basic hard problems (Index, Disjointness)
  - Hardness via reductions
Frequency Distributions

- Given set of items, let $f_i$ be the number of occurrences of item $i$.
- Many natural questions on $f_i$ values:
  - Find those $i$’s with large $f_i$ values (heavy hitters)
  - Find the number of non-zero $f_i$ values (count distinct)
  - Compute $F_k = \sum_i (f_i)^k$ – the $k$’th Frequency Moment
  - Compute $H = \sum_i (f_i/F_1) \log (F_1/f_i)$ – the (empirical) entropy

“Space Complexity of the Frequency Moments”
Alon, Matias, Szegedy in STOC 1996
- Awarded Gödel prize in 2005
- Set the pattern for many streaming algorithms to follow
Concentration Bounds

- Will provide randomized algorithms for these problems
- Each algorithm gives a (randomized) estimate of the answer
- Give confidence bounds on the final estimate $X$
  - Use probabilistic concentration bounds on random variables
- A concentration bound is typically of the form
  \[ \Pr[ |X - x| > \varepsilon y ] < \delta \]
  - At most probability $\delta$ of being more than $\varepsilon y$ away from $x$
Markov Inequality

- Take any probability distribution $X$ s.t. $\Pr[X < 0] = 0$
- Consider the event $X \geq k$ for some constant $k > 0$
- For any draw of $X$, $kI(X \geq k) \leq X$
  - Either $0 \leq X < k$, so $I(X \geq k) = 0$
  - Or $X \geq k$, $\text{lhs} = k$
- Take expectations of both sides: $k \Pr[ X \geq k ] \leq E[X]$
- **Markov inequality**: $\Pr[ X \geq k ] \leq E[X]/k$
  - Prob of random variable exceeding $k$ times its expectation $< 1/k$
  - Relatively weak in this form, but still useful
Sketch Structures

- **Sketch** is a class of summary that is a linear transform of input
  - \( \text{Sketch}(x) = Sx \) for some matrix \( S \)
  - Hence, \( \text{Sketch}(\alpha x + \beta y) = \alpha \text{Sketch}(x) + \beta \text{Sketch}(y) \)
  - Trivial to update and merge

- Often describe \( S \) in terms of hash functions
  - If hash functions are simple, sketch is fast

- Aim for limited independence hash functions \( h: [n] \rightarrow [m] \)
  - If \( \Pr_{h \in H}[ h(i_1)=j_1 \wedge h(i_2)=j_2 \wedge \ldots h(i_k)=j_k ] = m^{-k} \)
    then \( H \) is \( k \)-wise independent family ("\( h \) is \( k \)-wise independent")
  - \( k \)-wise independent hash functions take time, space \( O(k) \)
A First Sketch: Fingerprints

- Test if two (distributed) binary vectors are equal:
  \[ d = (x, y) = 0 \text{ iff } x = y, 1 \text{ otherwise} \]
- To test in small space: pick a suitable hash function \( h \)
- Test \( h(x) = h(y) \): small chance of false positive, no chance of false negative
- Compute \( h(x), h(y) \) incrementally as new bits arrive
  - How to choose the function \( h() \)?
### Polynomial Fingerprints

- Pick \( h(x) = \sum_{i=1}^{n} x_i r^i \mod p \) for prime \( p \), random \( r \in \{1...p-1\} \)
  - Flexible: \( h(x) \) is linear function of \( x \)—easy to update and merge
- For accuracy, note that computation \( \mod p \) is over the field \( \mathbb{Z}_p \)
  - Consider the polynomial in \( \alpha, \sum_{i=1}^{n} (x_i - y_i) \alpha^i = 0 \)
  - Polynomial of degree \( n \) over \( \mathbb{Z}_p \) has at most \( n \) roots
- Probability that \( r \) happens to solve this polynomial is \( n/p \)
- So \( \Pr[ h(x) = h(y) \mid x \neq y ] \leq n/p \)
  - Pick \( p = \text{poly}(n) \), fingerprints are \( \log p = O(\log n) \) bits
- Fingerprints applied to small subsets of data to test equality
  - Will see several examples that use fingerprints as subroutine
Sketches and Frequency Moments

- Frequency distributions and Concentration bounds
- Count-Min sketch for $F_\infty$ and frequent items
- AMS Sketch for $F_2$
- Estimating $F_0$
- Extensions:
  - Higher frequency moments
  - Combined frequency moments
Count-Min Sketch

- Simple sketch idea relies primarily on Markov inequality
- Model input data as a vector $x$ of dimension $U$
- Creates a small summary as an array of $w \times d$ in size
- Use $d$ hash function to map vector entries to $[1..w]$
- Works on arrivals only and arrivals & departures streams
Count-Min Sketch Structure

- Each entry in vector $x$ is mapped to one bucket per row.
- Merge two sketches by entry-wise summation
- Estimate $x[j]$ by taking $\min_k CM[k,h_k(j)]$
  - Guarantees error less than $\varepsilon F_1$ in size $O(1/\varepsilon \log 1/\delta)$
  - Probability of more error is less than $1-\delta$

$w = 2/\varepsilon$

$d = \log 1/\delta$

[C, Muthukrishnan '04]
Approximation of Point Queries

Approximate point query \( x'[j] = \min_k CM[k,h_k(j)] \)

- **Analysis:** In \( k \)'th row, \( CM[k,h_k(j)] = x[j] + X_{k,j} \)
  - \( X_{k,j} = \sum_i x[i] \text{I}(h_k(i) = h_k(j)) \)
  - \( E[X_{k,j}] = \sum_{i \neq j} x[i] \Pr[h_k(i) = h_k(j)] \)
    \( \leq \Pr[h_k(i) = h_k(j)] \sum_i x[i] \)
    \( = \varepsilon F_1/2 \) — requires only pairwise independence of \( h \)
  - \( \Pr[X_{k,j} \geq \varepsilon F_1] = \Pr[X_{k,j} \geq 2E[X_{k,j}]] \leq 1/2 \) by Markov inequality
- **So,** \( \Pr[x'[j] \geq x[j] + \varepsilon F_1] = \Pr[\forall k. X_{k,j} > \varepsilon F_1] \leq 1/2 \log 1/\delta = \delta \)
- **Final result:** with certainty \( x[j] \leq x'[j] \) and
  with probability at least \( 1-\delta \), \( x'[j] < x[j] + \varepsilon F_1 \)
Applications of Count-Min to Heavy Hitters

- Count-Min sketch lets us estimate $f_i$ for any $i$ (up to $\varepsilon F_1$)
- Heavy Hitters asks to find $i$ such that $f_i$ is large ($> \phi F_1$)
- Slow way: test every $i$ after creating sketch
- Alternate way:
  - Keep binary tree over input domain: each node is a subset
  - Keep sketches of all nodes at same level
  - Descend tree to find large frequencies, discard ‘light’ branches
  - Same structure estimates arbitrary range sums
- A first step towards compressed sensing style results...
Application to Large Scale Machine Learning

- In machine learning, often have very large feature space
  - Many objects, each with huge, sparse feature vectors
  - Slow and costly to work in the full feature space
- "Hash kernels": work with a sketch of the features
  - Effective in practice! [Weinberger, Dasgupta, Langford, Smola, Attenberg ‘09]
- Similar analysis explains why:
  - Essentially, not too much noise on the important features
Sketches and Frequency Moments

- Frequency distributions and Concentration bounds
- Count-Min sketch for $F_\infty$ and frequent items
- AMS Sketch for $F_2$
- Estimating $F_0$
- Extensions:
  - Higher frequency moments
  - Combined frequency moments
Chebyshev Inequality

- Markov inequality is often quite weak
- But Markov inequality holds for any random variable
- Can apply to a random variable that is a function of $X$
- Set $Y = (X - E[X])^2$
- By Markov, $Pr[ Y > kE[Y] ] < 1/k$
  - $E[Y] = E[(X-E[X])^2] = Var[X]$
- Hence, $Pr[ |X - E[X]| > \sqrt{k \ Var[X]} ] < 1/k$

**Chebyshev inequality:** $Pr[ |X - E[X]| > k ] < \frac{Var[X]}{k^2}$
- If $Var[X] \leq \varepsilon^2 E[X]^2$, then $Pr[ |X - E[X]| > \varepsilon E[X] ] = O(1)$
**F₂ estimation**

- AMS sketch (for Alon-Matias-Szegedy) proposed in 1996
  - Allows estimation of F₂ (second frequency moment)
  - Used at the heart of many streaming and non-streaming applications: achieves dimensionality reduction
- Here, describe AMS sketch by generalizing CM sketch.
- Uses extra hash functions $g_1...g_{\log \frac{1}{\delta}} \{1...U\} \rightarrow \{+1,-1\}$
  - (Low independence) Rademacher variables
- Now, given update $(j,+c)$, set $CM[k,h_k(j)] += c^*g_k(j)$
Estimate $F_2 = \text{median}_k \sum_i \text{CM}[k,i]^2$

Each row’s result is $\sum_i g(i)^2 x[i]^2 + \sum_{h(i)=h(j)} 2 \ g(i) \ g(j) \ x[i] \ x[j]$

But $g(i)^2 = -1^2 = +1^2 = 1$, and $\sum_i x[i]^2 = F_2$

g(i)g(j) has 1/2 chance of +1 or −1: expectation is 0 ...

$w = 4/\varepsilon^2$

$d=8\log 1/\delta$
**$F_2$ Variance**

- Expectation of row estimate $R_k = \sum_i CM[k,i]^2$ is exactly $F_2$
- Variance of row $k$, $\text{Var}[R_k]$, is an expectation:
  - $\text{Var}[R_k] = E\left( \sum_{\text{buckets } b} (CM[k,b])^2 - F_2\right)^2$
  - Good exercise in algebra: expand this sum and simplify
  - Many terms are zero in expectation because of terms like $g(a)g(b)g(c)g(d)$ (degree at most 4)
  - Requires that hash function $g$ is *four-wise independent*: it behaves uniformly over subsets of size four or smaller
    - Such hash functions are easy to construct
F_2 Variance

- Terms with odd powers of g(a) are zero in expectation
  - g(a)g(b)g^2(c), g(a)g(b)g(c)g(d), g(a)g^3(b)
- Leaves
  \[ \text{Var}[R_k] \leq \sum_i g^4(i) x[i]^4 \]
  \[ + 2 \sum_{j \neq i} g^2(i) g^2(j) x[i]^2 x[j]^2 \]
  \[ + 4 \sum_{h(i) = h(j)} g^2(i) g^2(j) x[i]^2 x[j]^2 \]
  \[ - (x[i]^4 + \sum_{j \neq i} 2x[i]^2 x[j]^2) \]
  \[ \leq F_2^2/w \]

- Row variance can finally be bounded by \( F_2^2/w \)
  - Chebyshev for \( w=4/\varepsilon^2 \) gives probability \( \frac{1}{4} \) of failure:
    \[ \Pr[ |R_k - F_2| > \varepsilon^2 F_2 ] \leq \frac{1}{4} \]
  - How to amplify this to small \( \delta \) probability of failure?
  - Rescaling \( w \) has cost linear in \( 1/\delta \)
Tail Inequalities for Sums

- We achieve stronger bounds on tail probabilities for the sum of independent *Bernoulli trials* via the Chernoff Bound:
  - Let $X_1, \ldots, X_m$ be independent Bernoulli trials s.t. $\Pr[X_i=1] = p$
    ($\Pr[X_i=0] = 1-p$).
  - Let $X = \sum_{i=1}^{m} X_i$, and $\mu = mp$ be the expectation of $X$.
  - Then, for $\varepsilon > 0$, Chernoff bound states:
    $$\Pr[ |X - \mu| \geq \varepsilon \mu] \leq 2 \exp(- \frac{1}{2} \mu \varepsilon^2)$$
  - Proved by applying Markov inequality to $Y = \exp(X_1 \cdot X_2 \cdot \ldots \cdot X_m)$
Applying Chernoff Bound

- Each row gives an estimate that is within $\varepsilon$ relative error with probability $p' > \frac{3}{4}$

- Take $d$ repetitions and find the median. Why the median?
  
  - Because bad estimates are either too small or too large
  - Good estimates form a contiguous group “in the middle”
  - At least $d/2$ estimates must be bad for median to be bad

- Apply Chernoff bound to $d$ independent estimates, $p = 1/4$
  
  - $\Pr[\text{More than } d/2 \text{ bad estimates }] < 2^{\exp(-d/8)}$
  - So we set $d = \Theta(\ln 1/\delta)$ to give $\delta$ probability of failure

- Same outline used many times in summary construction
Applications and Extensions

- $F_2$ guarantee: estimate $\|x\|_2$ from sketch with error $\varepsilon \|x\|_2$
  - Since $\|x + y\|_2^2 = \|x\|_2^2 + \|y\|_2^2 + 2x \cdot y$
    Can estimate $(x \cdot y)$ with error $\varepsilon \|x\|_2 \|y\|_2$
  - If $y = e_j$, obtain $(x \cdot e_j) = x_j$ with error $\varepsilon \|x\|_2$:
    $L_2$ guarantee ("Count Sketch") vs $L_1$ guarantee (Count-Min)

- Can view the sketch as a low-independence realization of the Johnson-Lindenstrauss lemma
  - Best current JL methods have the same structure
  - JL is stronger: embeds directly into Euclidean space
  - JL is also weaker: requires $O(1/\varepsilon)$-wise hashing, $O(\log 1/\delta)$ independence [Kane, Nelson 12]
Sketches and Frequency Moments

- Frequency Moments and Sketches
- Count-Min sketch for $F_\infty$ and frequent items
- AMS Sketch for $F_2$
- Estimating $F_0$
- Extensions:
  - Higher frequency moments
  - Combined frequency moments
**F₀ Estimation**

- F₀ is the number of distinct items in the stream
  - a fundamental quantity with many applications
- Early algorithms by Flajolet and Martin [1983] gave nice hashing-based solution
  - analysis assumed fully independent hash functions
- Will describe a generalized version of the FM algorithm due to Bar-Yossef et. al with only pairwise independence
  - Known as the “k-Minimum values (KMV)” algorithm
\textbf{F_0 Algorithm}

- Let $m$ be the domain of stream elements
  - Each item in data is from $[1...m]$
- Pick a random (pairwise) hash function $h: [m] \rightarrow [m^3]$
  - With probability at least $1-1/m$, no collisions under $h$
- For each stream item $i$, compute $h(i)$, and track the $t$ distinct items achieving the smallest values of $h(i)$
  - \textbf{Note}: if same $i$ is seen many times, $h(i)$ is same
  - Let $v_t = t$'th smallest (distinct) value of $h(i)$ seen
- If $F_0 < t$, give exact answer, else estimate $F'_0 = tm^3/v_t$
  - $v_t/m^3 \approx$ fraction of hash domain occupied by $t$ smallest
Analysis of $F_0$ algorithm

- Suppose $F'_0 = \frac{tm^3}{v_t} > (1+\varepsilon) F_0$  [estimate is too high]

- So for input = set $S \in 2^m$, we have
  - $|\{ s \in S \mid h(s) < \frac{tm^3}{(1+\varepsilon)F_0} \}| > t$
  - Because $\varepsilon < 1$, we have $\frac{tm^3}{(1+\varepsilon)F_0} \leq (1-\varepsilon/2)\frac{tm^3}{F_0}$
  - $Pr[ h(s) < (1-\varepsilon/2)\frac{tm^3}{F_0} ] \approx 1/m^3 \ast (1-\varepsilon/2)\frac{tm^3}{F_0} = (1-\varepsilon/2)t/F_0$

- (this analysis outline hides some rounding issues)
Chebyshev Analysis

- Let \( Y \) be number of items hashing to under \( \frac{tm^3}{(1+\varepsilon)F_0} \)
  - \( E[Y] = F_0 \times Pr[ h(s) < \frac{tm^3}{(1+\varepsilon)F_0} ] = (1-\varepsilon/2)t \)
  - For each item \( i \), variance of the event = \( p(1-p) < p \)
  - \( \text{Var}[Y] = \sum_{s \in S} \text{Var}[ h(s) < \frac{tm^3}{(1+\varepsilon)F_0} ] < (1-\varepsilon/2)t \)
    - We sum variances because of pairwise independence

- Now apply Chebyshev inequality:
  - \( \Pr[ Y > t ] \leq \Pr[|Y - E[Y]| > \varepsilon t/2] \)
    \( \leq 4\text{Var}[Y]/\varepsilon^2 t^2 \)
    \( < 4t/(\varepsilon^2 t^2) \)
  - Set \( t=20/\varepsilon^2 \) to make this \( \text{Prob} \leq 1/5 \)
Completing the analysis

We have shown
\[ \Pr[ F'_0 > (1+\varepsilon) F_0 ] < 1/5 \]

Can show \( \Pr[ F'_0 < (1-\varepsilon) F_0 ] < 1/5 \) similarly
  - too few items hash below a certain value

So \( \Pr[ (1-\varepsilon) F_0 \leq F'_0 \leq (1+\varepsilon)F_0] > 3/5 \) [Good estimate]

Amplify this probability: repeat \( O(\log 1/\delta) \) times in parallel with different choices of hash function \( h \)
  - Take the median of the estimates, analysis as before
**F₀ Issues**

- **Space cost:**
  - Store $t$ hash values, so $O(1/\varepsilon^2 \log m)$ bits
  - Can improve to $O(1/\varepsilon^2 + \log m)$ with additional tricks

- **Time cost:**
  - Find if hash value $h(i) < v_t$
  - Update $v_t$ and list of $t$ smallest if $h(i)$ not already present
  - Total time $O(\log 1/\varepsilon + \log m)$ worst case
Count-Distinct

- Engineering the best constants: Hyperloglog algorithm
  - Hash each item to one of $1/\varepsilon^2$ buckets (like Count-Min)
  - In each bucket, track the function $\max \lceil \log(h(x)) \rceil$
    - Can view as a coarsened version of KMV
    - Space efficient: need $\log \log m \approx 6$ bits per bucket
- Can estimate intersections between sketches
  - Make use of identity $|A \cap B| = |A| + |B| - |A \cup B|$
  - Error scales with $\varepsilon \sqrt{|A| \cdot |B|}$, so poor for small intersections
  - Higher order intersections via inclusion-exclusion principle
Bloom Filters

- **Bloom filters** compactly encode set membership
  - $k$ hash functions map items to bit vector $k$ times
  - Set all $k$ entries to 1 to indicate item is present
  - Can lookup items, store set of size $n$ in $O(n)$ bits

- Duplicate insertions do not change Bloom filters
- Can **merge** by OR-ing vectors (of same size)
Bloom Filter analysis

- How to set \( k \) (number of hash functions), \( m \) (size of filter)?
- False positive: when all \( k \) locations for an item are set
  - If \( \rho \) fraction of cells are empty, false positive probability is \((1-\rho)^k\)
- Consider probability of any cell being empty:
  - For \( n \) items, \( \Pr[\text{cell j is empty}] = (1 - 1/m)^{kn} \approx \rho \approx \exp(-kn/m) \)
  - False positive prob = \((1 - \rho)^k = \exp(k \ln(1 - \rho)) = \exp(-m/n \ln(\rho) \ln(1-\rho))\)
- For fixed \( n, m \), by symmetry minimized at \( \rho = \frac{1}{2} \)
  - Half cells are occupied, half are empty
  - Give \( k = (m/n)\ln 2 \), false positive rate is \( \frac{1}{2}^k \)
  - Choose \( m = cn \) to get constant FP rate, e.g. \( c=10 \) gives < 1% FP
Bloom Filters Applications

- Bloom Filters widely used in “big data” applications
  - Many problems require storing a large set of items
- Can generalize to allow deletions
  - Swap bits for counters: increment on insert, decrement on delete
  - If representing sets, small counters suffice: 4 bits per counter
  - If representing multisets, obtain sketches (next lecture)
- Bloom Filters are an active research area
  - Several papers on topic in every networking conference...
Frequency Moments

- Intro to frequency distributions and Concentration bounds
- Count-Min sketch for $F_\infty$ and frequent items
- AMS Sketch for $F_2$
- Estimating $F_0$

- Extensions:
  - Higher frequency moments
  - Combined frequency moments
Higher Frequency Moments

- $F_k$ for $k>2$. Use a sampling trick [Alon et al 96]:
  - Uniformly pick an item from the stream length $1...n$
  - Set $r =$ how many times that item appears subsequently
  - Set estimate $F'_k = n (r^k - (r-1)^k)$

- $E[F'_k] = 1/n \cdot n \cdot [ f_1^k - (f_1-1)^k + (f_1-1)^k - (f_1-2)^k + ... + 1^k - 0^k ] + ...$
  - $= f_1^k + f_2^k + ... = F_k$

- $Var[F'_k] \leq 1/n \cdot n^2 \cdot [(f_1^k - (f_1-1)^k)^2 + ...]$
  - Use various bounds to bound the variance by $k \cdot m^{1-1/k} \cdot F_k^2$
  - Repeat $k \cdot m^{1-1/k}$ times in parallel to reduce variance

- Total space needed is $O(k \cdot m^{1-1/k})$ machine words
  - Not a sketch: does not distribute easily. See part 2!
Combined Frequency Moments

- Let $G[i,j] = 1$ if $(i,j)$ appears in input. E.g. graph edge from $i$ to $j$. Total of $m$ distinct edges.
- Let $d_i = \sum_{j=1}^{n} G[i,j]$ (aka degree of node $i$).
- Find aggregates of $d_i$'s:
  - Estimate heavy $d_i$'s (people who talk to many).
  - Estimate frequency moments:
    - number of distinct $d_i$ values, sum of squares
  - Range sums of $d_i$’s (subnet traffic)
- **Approach**: nest one sketch inside another, e.g. HLL inside CM
  - Requires new analysis to track overall error.
Range Efficiency

- Sometimes input is specified as a collection of ranges \([a,b]\)
  - \([a,b]\) means insert all items \((a, a+1, a+2 \ldots b)\)
  - Trivial solution: just insert each item in the range
- **Range efficient** \(F_0\) [Pavan, Tirthapura 05]
  - Start with an alg for \(F_0\) based on pairwise hash functions
  - Key problem: track which items hash into a certain range
  - Dives into hash fns to divide and conquer for ranges
- **Range efficient** \(F_2\) [Calderbank et al. 05, Rusu, Dobra 06]
  - Start with sketches for \(F_2\) which sum hash values
  - Design new hash functions so that range sums are fast
- **Rectangle Efficient** \(F_0\) [Tirthapura, Woodruff 12]
Forthcoming Attractions

- Data Streams Mini Course @Simons
  - Prof Andrew McGregor
  - Starts early October

- Succinct Data Representations and Applications @ Simons
  - September 16-19
Streaming, Sketching and Sufficient Statistics

Graham Cormode
University of Warwick
G.Cormode@Warwick.ac.uk
Recap

- Sketching Techniques summarize large data sets
- **Summarize vectors:**
  - Test equality (fingerprints)
  - Recover approximate entries (count-min, count sketch)
  - Approximate Euclidean norm ($F_2$) and dot product
  - Approximate number of non-zero entries ($F_0$)
  - Approximate set membership (Bloom filter)
Part II: Advanced Topics

- Sampling and $L_p$ Sampling
  - $L_0$ sampling and graph sketching
  - $L_2$ sampling and frequency moment estimation

- Matrix computations
  - Sketches for matrix multiplication
  - Sparse representation via frequent directions

- Lower bounds for streaming and sketching
  - Basic hard problems (Index, Disjointness)
  - Hardness via reductions
**Sampling From a Large Input**

- **Fundamental prob:** sample $m$ items uniformly from data
  - **Useful:** approximate costly computation on small sample

- **Challenge:** don’t know how large total input is
  - So when/how often to sample?

- **Several solutions, apply to different situations:**
  - Reservoir sampling (dates from 1980s?)
  - Min-wise sampling (dates from 1990s?)
Min-wise Sampling

- For each item, pick a random fraction between 0 and 1
- Store item(s) with the smallest random tag [Nath et al.’04]

- Each item has same chance of least tag, so uniform
- Can run on multiple inputs separately, then merge
- Applications in geometry: basic $\varepsilon$-approximations are samples
  - Estimate number of points falling in a range (bounded VC dim)
Sampling from Sketches

- Given inputs with positive and negative weights
- Want to sample based on the overall frequency distribution
  - Sample from support set of $n$ possible items
  - Sample proportional to (absolute) weights
  - Sample proportional to some function of weights
- How to do this sampling effectively?
- Recent approach: $L_p$ sampling
**L_p Sampling**

- **L_p sampling**: use sketches to sample with prob $(1 \pm \varepsilon) \frac{f_i^p}{\|f\|_p^p}$
- “Efficient” solutions developed of size $O(\varepsilon^{-2} \log^2 n)$
  - [Monemizadeh, Woodruff 10] [Jowhari, Saglam, Tardos 11]
- **L_0 sampling** enables novel “graph sketching” techniques
  - Sketches for connectivity, sparsifiers [Ahn, Guha, McGregor 12]
- **L_2 sampling** allows optimal estimation of frequency moments
L₀ Sampling

- L₀ sampling: sample with prob \((1\pm \varepsilon) \frac{f_i^0}{F_0}\)
  - i.e., sample (near) uniformly from items with non-zero frequency

- General approach: [Frahling, Indyk, Sohler 05, C., Muthu, Rozenbaum 05]
  - Sub-sample all items (present or not) with probability \(p\)
  - Generate a sub-sampled vector of frequencies \(f_p\)
  - Feed \(f_p\) to a \(k\)-sparse recovery data structure
    - Allows reconstruction of \(f_p\) if \(F_0 < k\)
  - If \(f_p\) is \(k\)-sparse, sample from reconstructed vector
  - Repeat in parallel for exponentially shrinking values of \(p\)
Sampling Process

- Exponential set of probabilities, \( p=1, \frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}...\ \frac{1}{U} \)
  - Let \( N = F_0 = |\{ i : f_i \neq 0\}| \)
  - Want there to be a level where k-sparse recovery will succeed
  - At level \( p \), expected number of items selected \( S \) is \( Np \)
  - Pick level \( p \) so that \( \frac{k}{3} < Np \leq \frac{2k}{3} \)

- Chernoff bound: with probability exponential in \( k \), \( 1 \leq S \leq k \)
  - Pick \( k = O(\log \frac{1}{\delta}) \) to get \( 1-\delta \) probability
**k-Sparse Recovery**

- Given vector $x$ with at most $k$ non-zeros, recover $x$ via sketching
  - A core problem in compressed sensing/compressive sampling
- **First approach**: Use Count-Min sketch of $x$
  - Probe all $U$ items, find those with non-zero estimated frequency
  - Slow recovery: takes $O(U)$ time
- **Faster approach**: also keep sum of item identifiers in each cell
  - Sum/count will reveal item id
  - Avoid false positives: keep fingerprint of items in each cell
- Can keep a sketch of size $O(k \log U)$ to recover up to $k$ items

<table>
<thead>
<tr>
<th>Sum, $\sum_{i \cdot h(i) = j} i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count, $\sum_{i \cdot h(i) = j} x_i$</td>
</tr>
<tr>
<td>Fingerprint, $\sum_{i \cdot h(i) = j} x_i r_i$</td>
</tr>
</tbody>
</table>
Uniformity

- Also need to argue sample is uniform
  - Failure to recover could bias the process
- $\Pr[ i \text{ would be picked if } k=n ] = 1/F_0$ by symmetry
- $\Pr[ i \text{ is picked } ] = \Pr[ i \text{ would be picked if } k=n \land S \leq k ] \geq (1-\delta)/F_0$
- So $(1-\delta)/N \leq \Pr[ i \text{ is picked } ] \leq 1/N$
- Sufficiently uniform (pick $\delta = \varepsilon$)
Application: Graph Sketching

- Given $L_0$ sampler, use to sketch (undirected) graph properties
- **Connectivity**: want to test if there is a path between all pairs
- **Basic alg**: repeatedly contract edges between components
- Use $L_0$ sampling to provide edges on vector of adjacencies
- **Problem**: as components grow, sampling most likely to produce internal links
Graph Sketching

- **Idea**: use clever encoding of edges [Ahn, Guha, McGregor 12]
- Encode edge \((i, j)\) as \(((i, j), +1)\) for node \(i < j\), as \(((i, j), -1)\) for node \(j > i\)
- When node \(i\) and node \(j\) get merged, sum their \(L_0\) sketches
  - Contribution of edge \((i, j)\) exactly cancels out
- Only non-internal edges remain in the \(L_0\) sketches
- Use independent sketches for each iteration of the algorithm
  - Only need \(O(\log n)\) rounds with high probability
- **Result**: \(O(\text{poly-log } n)\) space per node for connectivity
Other Graph Results via sketching

- **K-connectivity via connectivity**
  - Use connectivity result to find and remove a spanning forest
  - Repeat $k$ times to generate $k$ spanning forests $F_1, F_2, \ldots, F_k$
  - **Theorem:** $G$ is $k$-connected if $\cup_{i=1}^k F_i$ is $k$-connected

- **Bipartiteness via connectivity:**
  - Compute $c =$ number of connected components in $G$
  - Generate $G'$ over $V \cup V'$ so $(u,v) \in E \Rightarrow (u, v') \in E'$, $(u', v) \in E'$
  - If $G$ is bipartite, $G'$ has $2c$ components, else it has $<2c$ components

- **Minimum spanning tree:**
  - Round edge weights to powers of $(1+\varepsilon)$
  - Define $n_i =$ number of components on edges lighter than $(1+\varepsilon)^i$
  - **Fact:** weight of MST on rounded weights is $\sum_i \varepsilon(1+\varepsilon)^i n_i$
Application: $F_k$ via $L_2$ Sampling

- Recall, $F_k = \sum_i f_i^k$
- Suppose $L_2$ sampling samples $f_i$ with probability $f_i^2/F_2$
  - And also estimates sampled $f_i$ with relative error $\epsilon$
- Estimator: $X = F_2 f_i^{k-2}$ (with estimates of $F_2$, $f_i$)
  - Expectation: $E[X] = F_2 \sum_i f_i^{k-2} \cdot f_i^2 / F_2 = F_k$
  - Variance: $Var[X] \leq E[X^2] = \sum_i f_i^2 / F_2 (F_2 f_i^{k-2})^2 = F_2 F_2k-2$


Rewriting the Variance

- Want to express variance $F_2 F_{2k-2}$ in terms of $F_k$ and domain size $n$
- Hölder’s inequality: $\langle x, y \rangle \leq \|x\|_p \|y\|_q$ for $1 \leq p, q$ with $1/p + 1/q = 1$
  - Generalizes Cauchy-Shwarz inequality, where $p=q=2$.
- So pick $p = k/(k-2)$ and $q = k/2$ for $k > 2$. Then
  $$\langle 1^n, (f_i)^2 \rangle \leq \|1^n\|_{k/(k-2)} \|(f_i)^2\|_{k/2}$$
  $$F_2 \leq n^{(k-2)/k} F_k^{2/k}$$ (1)
- Also, since $\|x\|_{p+a} \leq \|x\|_p$ for any $p \geq 1, a > 0$
  - Thus $\|x\|_{2k-2} \leq \|x\|_k$ for $k \geq 2$
  - So $F_{2k-2} = \|f\|_{2k-2}^{2k-2} \leq \|f\|_k^{2k-2} = F_k^{2-2/k}$ (2)
- Multiply (1) * (2): $F_2 F_{2k-2} \leq n^{1-2/k} F_k^2$
  - So variance is bounded by $n^{1-2/k} F_k^2$
**$F_k$ Estimation**

- For $k \geq 3$, we can estimate $F_k$ via $L_2$ sampling:
  - Variance of our estimate is $O(F_k^2 n^{1-2/k})$
  - Take mean of $n^{1-2/k} \varepsilon^{-2}$ repetitions to reduce variance
  - Apply Chebyshev inequality: constant prob of good estimate
  - Chernoff bounds: $O(\log 1/\delta)$ repetitions reduces prob to $\delta$

- How to instantiate this?
  - Design method for approximate $L_2$ sampling via sketches
  - Show that this gives relative error approximation of $f_i$
  - Use approximate value of $F_2$ from sketch
  - Complicates the analysis, but bound stays similar
**L₂ Sampling Outline**

- For each \( i \), draw \( u_i \) uniformly in the range 0...1
  - From vector of frequencies \( f \), derive \( g \) so \( g_i = f_i / \sqrt{u_i} \)
  - Sketch \( g_i \) vector

- **Sample**: return \((i, f_i)\) if there is unique \( i \) with \( g_i^2 > t = F_2 / \varepsilon \) threshold
  - \( \Pr[ g_i^2 > t \land \forall j \neq i : g_j^2 < t ] = \Pr[ g_i^2 > t ] \prod_{j \neq i} \Pr[ g_j^2 < t ] \)
    - \( = \Pr[ u_i < \varepsilon f_i^2 / F_2 ] \prod_{j \neq i} \Pr[ u_j > \varepsilon f_j^2 / F_2 ] \)
    - \( = (\varepsilon f_i^2 / F_2 ) \prod_{j \neq i} (1 - \varepsilon f_j^2 / F_2 ) \)
    - \( \approx \varepsilon f_i^2 / F_2 \)

- Probability of returning anything is not so big: \( \sum_i \varepsilon f_i^2 / F_2 = \varepsilon \)
  - Repeat \( O(1/\varepsilon \log 1/\delta) \) times to improve chance of sampling
L₂ sampling continued

- Given (estimated) \( g_i \) s.t. \( g_i^2 \geq F_2/\varepsilon \), estimate \( f_i = u_i g_i \)
- Sketch size \( O(\varepsilon^{-1} \log n) \) means estimate of \( f_i^2 \) has error \( (\varepsilon f_i^2 + u_i^2) \)
  - With high prob, no \( u_i < 1/\text{poly}(n) \), and so \( F_2(g) = O(F_2(f) \log n) \)
  - Since estimated \( f_i^2/u_i^2 \geq F_2/\varepsilon \), \( u_i^2 \leq \varepsilon f_i^2/F_2 \)
- Estimating \( f_i^2 \) with error \( \varepsilon f_i^2 \) sufficient for estimating \( F_k \)

- Many details omitted
  - See Precision Sampling paper [Andoni Krauthgamer Onak 11]
Advanced Topics

- Sampling and $L_p$ Sampling
  - $L_0$ sampling and graph sketching
  - $L_2$ sampling and frequency moment estimation
- Matrix computations
  - Sketches for matrix multiplication
  - Sparse representation via frequent directions
- Lower bounds for streaming and sketching
  - Basic hard problems (Index, Disjointness)
  - Hardness via reductions
Matrix Sketching

- Given matrices $A$, $B$, want to approximate matrix product $AB$
- Compute normed error of approximation $C$: $\|AB - C\|$
- Give results for the Frobenius (entrywise) norm $\|\cdot\|_F$
  - $\|C\|_F = (\sum_{i,j} C_{i,j}^2)^{\frac{1}{2}}$
  - Results rely on sketches, so this norm is most natural
Direct Application of Sketches

- Build sketch of each row of $A$, each column of $B$
- Estimate $C_{i,j}$ by estimating inner product of $A_i$ with $B^j$
- Absolute error in estimate is $\varepsilon \|A_i\|_2 \|B^j\|_2$ (whp)
- Sum over all entries in matrix, squared error is
  \[ \varepsilon^2 \sum_{i,j} \|A_i\|_2^2 \|B^j\|_2^2 = \varepsilon^2 (\sum_i \|A_i\|_2^2)(\sum_j \|B^j\|_2^2) \]
  \[ = \varepsilon^2 (\|A\|_F^2)(\|B\|_F^2) \]
- Hence, Frobenius norm of error is $\varepsilon \|A\|_F \|B\|_F$
- Problem: need the bound to hold for all sketches simultaneously
  - Requires polynomially small failure probability
  - Increases sketch size by logarithmic factors
Improved Matrix Multiplication Analysis

- Simple analysis is too pessimistic [Clarkson Woodruff 09]
  - It bounds probability of failure of each sketch independently
- A better approach is to directly analyze variance of error
  - Immediately, each estimate of (AB) has variance $\varepsilon^2 \|A\|_F^2 \|B\|_F^2$
  - Just need to apply Chebyshev inequality to sum... almost
- Problem: how to amplify probability of correctness?
  - ‘Median’ trick doesn’t work: what is median of set of matrices?
  - Find an estimate which is close to most others
    - Estimate $\|A\|_F^2 \|B\|_F^2 := d$ using sketches
    - Find an estimate that’s closer than $d/2$ to more than $\frac{1}{2}$ the rest
    - We find an estimate with this property with probability $1-\delta$
More directly approximate matrix multiplication:
- use more powerful hash functions in sketching
- obtain a single accurate estimate with high probability

Linear regression given matrix $A$ and vector $b$:
find $x \in \mathbb{R}^d$ to (approximately) solve $\min_x \|Ax - b\|
- Approach: solve the minimization in “sketch space”
- Require a summary of size $O(d^2/\varepsilon \log 1/\delta)$
Frequent Items and Frequent Directions

- A deterministic algorithm for tracking item frequencies
  - With a recent analysis of its performance
  - Unusually, it is deterministic
- Inspiring an algorithm for tracking matrix properties
  - Due to [Liberty 13], extended by [Ghashami Phillips 13]
Misra-Gries Summary (1982)

- **Misra-Gries (MG) algorithm** finds up to $k$ items that occur more than $1/k$ fraction of the time in the input.

- **Update**: Keep $k$ different candidates in hand. For each item:
  - If item is monitored, increase its counter.
  - Else, if $< k$ items monitored, add new item with count 1.
  - Else, decrease all counts by 1.
Streaming MG analysis

- $N =$ total weight of input
- $M =$ sum of counters in data structure
- **Error** in any estimated count at most $(N - M)/(k + 1)$
  - Estimated count a lower bound on true count
  - Each decrement spread over $(k + 1)$ items: 1 new one and $k$ in MG
  - Equivalent to deleting $(k + 1)$ distinct items from stream
  - At most $(N - M)/(k + 1)$ decrement operations
  - Hence, can have “deleted” $(N - M)/(k + 1)$ copies of any item
  - So estimated counts have at most this much error
Merging two MG Summaries [ACHPWY ‘12]

Merge algorithm:
- Merge the counter sets in the obvious way
- Take the \((k+1)\)th largest counter \(= C_{k+1}\), and subtract from all
- Delete non-positive counters
- Sum of remaining counters is \(M_{12}\)

This keeps the same guarantee as Update:
- Merge subtracts at least \((k+1)C_{k+1}\) from counter sums
- So \((k+1)C_{k+1} \leq (M_1 + M_2 - M_{12})\)
- By induction, error is
  \[
  ((N_1-M_1) + (N_2-M_2) + (M_1+M_2-M_{12}))/\!(k+1) = ((N_1+N_2) - M_{12})/(k+1)
  \]
  (prior error) (from merge) (as claimed)
A Powerful Summary

- MG summary with update and merge is very powerful
  - Builds a compact summary of the frequency distribution
  - Can also multiply the summary by any scalar
  - Hence can take (positive) linear combinations: $\alpha x + \beta y$
  - Useful for building models of data

- Ideas recently extended to matrix computations
Frequent Directions

- **Input**: An \( n \times d \) matrix \( A \), presented one row at a time
- Find \( k \times d \) matrix \( Q \) so for any vector \( x \), \( Qx \) approximates \( Ax \)
- **Simple idea**: use SVD to focus on most important directions
- Given current \( k \times d \) matrix \( Q \)
  - Replace last row with new row \( a_i \)
  - Compute SVD of \( Q \) as \( U\Sigma V \)
  - Set \( \Sigma' = \text{diag}( \sqrt{\sigma_1^2 - \sigma_k^2}, \sqrt{\sigma_2^2 - \sigma_k^2}, \ldots, \sqrt{\sigma_{k-1}^2 - \sigma_k^2}, \sqrt{\sigma_k^2 - \sigma_k^2}) = 0) \)
  - Rescale: \( Q' = \Sigma'V^T \)
- At step \( i \), have introduced error based on \( \delta_i = \Sigma_{k,k} = \sigma_k \)
Frequent Directions Analysis

- Error (in Frobenius norm) introduced at each step at most $\delta_i^2$
  - Let $v_j$ be $j$'th column of $V_j$ and pick any $x$ such that $\|x\|_2 = 1$
  - $\|Qx\|_2^2 = \sum_{j=1}^{k} \sigma_j^2 (v_j \cdot x)^2 = \sum_{j=1}^{k} (\sigma'_j^2 + \delta_i^2) (v_j \cdot x)^2$
    $= \sum_{j=1}^{k} \sigma'_j^2 (v_j \cdot x)^2 + \sum_{j=1}^{k} \delta_i^2 (v_j \cdot x)^2$
    $\leq \|Q'x\|_2^2 + \delta_i^2$

- Observe that $\|Q'\|_F^2 - \|Q\|_F^2 = \delta_i^2 + \delta_i^2 + ... = k \delta_i^2$
- Adding row $a_i$ causes $\|Q\|_F^2$ to increase by $\|a_i\|_2^2$
- Hence, $\|A\|_F^2 = \sum_i \|a_i\|_2^2 = k \sum_i \delta_i^2$
- Summing over all steps, $0 \leq \|Ax\|_2^2 - \|Qx\|_2^2 \leq \sum_i \delta_i^2 = \|A\|_F/k$
  - “Relative error” bounds follow by increasing $k$ [Ghashami Phillips 13]
Advanced Topics

- Sampling and $L_p$ Sampling
  - $L_0$ sampling and graph sketching
  - $L_2$ sampling and frequency moment estimation

- Matrix computations
  - Sketches for matrix multiplication
  - Sparse representation via frequent directions

- Lower bounds for streaming and sketching
  - Basic hard problems (Index, Disjointness)
  - Hardness via reductions
Streaming Lower Bounds

- Lower bounds for summaries
  - Communication and information complexity bounds
  - Simple reductions
  - Hardness of Gap-Hamming problem
  - Reductions to Gap-Hamming

1 0 1 1 1 0 1 0 1 ...

Alice

Bob
Computation As Communication

- Imagine Alice processing a prefix of the input
- Then takes the whole working memory, and sends to Bob
- Bob continues processing the remainder of the input
Suppose Alice’s part of the input corresponds to string $x$, and Bob’s part corresponds to string $y$...

...and computing the function corresponds to computing $f(x,y)$...

...then if $f(x,y)$ has communication complexity $\Omega(g(n))$, then the computation has a space lower bound of $\Omega(g(n))$

Proof by contradiction:
If there was an algorithm with better space usage, we could run it on $x$, then send the memory contents as a message, and hence solve the communication problem
Alice has string \( x \), Bob has string \( y \), want to test if \( x = y \)

Consider a deterministic (one-round, one-way) protocol that sends a message of length \( m < n \)

There are \( 2^m \) possible messages, so some strings must generate the same message: this would cause error

So a deterministic message (sketch) must be \( \Omega(n) \) bits
  – In contrast, we saw a randomized sketch of size \( O(\log n) \)
Hard Communication Problems

- **INDEX**: Alice’s $x$ is a binary string of length $n$
  Bob’s $y$ is an index in $[n]$
  **Goal**: output $x[y]$
  **Result**: (one-way) (randomized) communication complexity of **INDEX** is $\Omega(n)$ bits

- **AUGINDEX**: as **INDEX**, but $y$ additionally contains $x[y+1]...x[n]$
  **Result**: (one-way) (randomized) complexity of **AUGINDEX** is $\Omega(n)$ bits

- **DISJ**: Alice’s $x$ and Bob’s $y$ are both length $n$ binary strings
  **Goal**: Output 1 if $\exists i: x[i]=y[i]=1$, else 0
  **Result**: (multi-round) (randomized) communication complexity of **DISJ** (disjointness) is $\Omega(n)$ bits
Hardness of INDEX

- Show hardness of INDEX via Information Complexity argument
  - Makes extensive use of Information Theory
- Entropy of random variable $X$: $H(X) = - \sum_x \Pr[X=x] \, \lg \Pr[X=x]$  
  - (Expected) information (in bits) gained by learning value of $X$  
  - If $X$ takes on at most $N$ values, $H(X) \leq \lg N$
- Conditional Entropy of $X$ given $Y$: $H(X|Y) = \sum_y \Pr[y] \, H[X|Y=y]$  
  - (Expected) information (bits) gained by learning value of $X$ given $Y$
- Mutual Information: $I(X : Y) = I(Y : X) = H(X) - H(X | Y)$  
  - Information (in bits) shared by $X$ and $Y$  
  - If $X$, $Y$ are independent, $I(X : Y) = 0$ and $I(XY : Z) \geq I(X : Z) + I(Y : Z)$
Information Cost

- Use Information Theoretic properties to lower bound communication complexity
- Suppose Alice and Bob have random inputs \( X \) and \( Y \)
- Let \( M \) be the (random) message sent by Alice in protocol \( P \)
- The cost of (one-way) protocol \( P \) is \( \text{cost}(P) = \max |M| \)
  - Worst-case size of message (in bits) sent in the protocol
- Define information cost as \( \text{icost}(P) = I(M : X) \)
  - The information conveyed about \( X \) in \( M \)
  - \( \text{icost}(P) = I(M : X) = H(M) - H(M | X) \leq H(M) \leq \text{cost}(P) \)
Information Cost of INDEX

- Give Alice random input $X = n$ uniform random bits
- Given protocol $P$ for INDEX, Alice sends message $M(X)$
- Give Bob input $i$. He should output $X_i$
- \[ \text{icost}(P) = I(X_1 X_2 \ldots X_n : M) \geq I(X_1 : M) + I(X_2 : M) + \ldots + I(X_n : M) \]
- Now consider the mutual information of $X_i$ and $M$
  - Have reduced the problem to $n$ instances of a simpler problem
- **Intuition:** $I(X_j : M)$ should be at least constant, so $\text{cost}(P) = \Theta(n)$
Fano’s Inequality

- When forming estimate $X'$ from $X$ given (message) $M$, where $X, X'$ have $k$ possible values, let $E$ denote $X \neq X'$. We have:
  \[ H(E) + \Pr[E] \log(k-1) \geq H(X \mid M) \]
  where $H(E) = -\Pr[E] \lg \Pr[E] - (1-\Pr[E]) \lg(1-\Pr[E])$

- Here, $k=2$, so we get $I(X : M) = H(X) - H(X \mid M) \geq H(X) - H(E)$
  - $H(X) = 1$. If $\Pr[E]=\delta$, we have $H(E) < \frac{1}{2}$ for $\delta<0.1$
  - Hence $I(X_i : M) > \frac{1}{2}$

- Thus $\text{cost}(P) \geq \text{i}cost(P) > \frac{1}{2} n$ if $P$ succeeds w/prob $1-\delta$
  - Protocols for INDEX must send $\Omega(n)$ bits
  - Hardness of AUGINDEX follows similarly
Outline for DISJOINTNESS hardness

- Hardness for **DISJ** follows a similar outline
- Reduce to $n$ instances of the problem "**AND**"
  - "**AND**" problem: test whether $X_i = Y_i = 1$
- Show that the information cost of **DISJ** protocol is sufficient to solve all $n$ instances of **AND**
- Show that the information cost of each instance is $\Omega(1)$
- Proves that communication cost of **DISJ** is $\Omega(1)$
  - Even allowing *multiple rounds* of communication
Simple Reduction to Disjointness

- $F_\infty$: output the highest frequency in the input
- Input: the two strings $x$ and $y$ from disjointness instance
- Reduction: if $x[i]=1$, then put $i$ in input; then same for $y$
  - A streaming reduction (compare to polynomial-time reductions)
- Analysis: if $F_\infty=2$, then intersection; if $F_\infty\leq 1$, then disjoint.
- Conclusion: Giving exact answer to $F_\infty$ requires $\Omega(N)$ bits
  - Even approximating up to 50% relative error is hard
  - Even with randomization: $\text{DISJ}$ bound allows randomness

$x: 1\ 0\ 1\ 1\ 0\ 1 \rightarrow 1,\ 3,\ 4,\ 6$

$y: 0\ 0\ 0\ 1\ 1\ 0 \rightarrow 4,\ 5$
Simple Reduction to Index

- \( F_0 \): output the number of items in the stream
- **Input**: the strings \( x \) and index \( y \) from  **INDEX**
- **Reduction**: if \( x[i]=1 \), put \( i \) in input; then put \( y \) in input
- **Analysis**: if \((1-\varepsilon)F'_0(x \cup y) > (1+\varepsilon)F'_0(x)\) then \( x[y]=1 \), else it is 0
- **Conclusion**: Approximating \( F_0 \) for \( \varepsilon < 1/N \) requires \( \Omega(N) \) bits
  - Implies that space to approximate must be \( \Omega(1/\varepsilon) \)
  - Bound allows randomization
Matrix-Multiplication: approximate $A^T B$ with error $\varepsilon^2 \|A\|_F \|B\|_F$

- For $r \times c$ matrices. $A$ encodes string $x$, $B$ encodes index $y$

\[
\begin{pmatrix}
  +1 & -1 & -2 & -2 & \ldots & \pm 2^k & \pm 2^k & \ldots & 0 & 0 & 0 & 0 \\
  -1 & -1 & -2 & +2 & \ldots & \pm 2^k & \pm 2^k & \ldots & 0 & 0 & 0 & 0 \\
  +1 & +1 & +2 & -2 & \ldots & \pm 2^k & \pm 2^k & \ldots & 0 & 0 & 0 & 0 \\
  -1 & -1 & +2 & +2 & \ldots & \pm 2^k & \pm 2^k & \ldots & 0 & 0 & 0 & 0 \\
\end{pmatrix}
\]

$A^T B$ “reads off” $j$’th column of $A^T$

- Bob uses suffix of $x$ in $y$ to remove heavy entries from $A$
  $\|B\|_F = 1 \quad \|A\|_F = cr/\log (cn) \ast (1 + 4 + \ldots 2^{2k}) \leq 4cr2^{2k}/3\log (cn)$

- Choose $r = \log(cn)/8\varepsilon^2$ so permitted error is $c 2^{2k} / 6\varepsilon^2$
  - Each error in sign in estimate of $(A^T B)$ contributes $2^{2k}$ error
  - Can tolerate error in at most $1/6$ fraction of entries

- Matrix multiplication requires space $\Omega(rc) = \Omega(c/\varepsilon^2 \log (cn))$
Streaming Lower Bounds

- Lower bounds for data streams
  - Communication complexity bounds
  - Simple reductions
  - Hardness of Gap-Hamming problem
  - Reductions to Gap-Hamming
Gap Hamming

**Gap-Hamming** communication problem:

- Alice holds $x \in \{0,1\}^N$, Bob holds $y \in \{0,1\}^N$
- **Promise**: $\text{Ham}(x,y)$ is either $\leq N/2 - \sqrt{N}$ or $\geq N/2 + \sqrt{N}$
- Which is the case?
- **Model**: one message from Alice to Bob
- Sketching upper bound: need relative error $\epsilon = \sqrt{N/F_2} = 1/\sqrt{N}$
  - Gives space $O(1/\epsilon^2) = O(N)$

Requires $\Omega(N)$ bits of one-way randomized communication

[Indyk, Woodruff’03, Woodruff’04, Jayram, Kumar, Sivakumar ’07]
Reduction starts with an instance of **INDEX**
- Map string $x$ to $u$ by $1 \rightarrow +1$, $0 \rightarrow -1$ (i.e. $u[i] = 2x[i] - 1$)
- Assume both Alice and Bob have access to public random strings $r_j$, where each bit of $r_j$ is iid $\{-1, +1\}$
- Assume w.l.o.g. that length of string $n$ is odd (important!)
- Alice computes $a_j = \text{sign}(r_j \cdot u)$
- Bob computes $b_j = \text{sign}(r_j[y])$

Repeat $N$ times with different random strings, and consider the Hamming distance of $a_1...a_N$ with $b_1...b_N$
- Argue if we solve **Gap-Hamming** on $(a, b)$, we solve **INDEX**
Consider the pair $a_j = \text{sign}(r_j \cdot u)$, $b_j = \text{sign}(r_j[y])$

Let $w = \sum_{i \neq y} u[i] r_j[i]$

- $w$ is a sum of $(n-1)$ values distributed iid uniform $\{-1, +1\}$

**Case 1:** $w \neq 0$. So $|w| \geq 2$, since $(n-1)$ is even

- so $\text{sign}(a_j) = \text{sign}(w)$, independent of $x[y]$
- Then $\Pr[a_j \neq b_j] = \Pr[\text{sign}(w) \neq \text{sign}(r_j[y])] = \frac{1}{2}$

**Case 2:** $w = 0$

So $a_j = \text{sign}(r_j \cdot u) = \text{sign}(w + u[y]r_j[y]) = \text{sign}(u[y]r_j[y])$

- Then $\Pr[a_j \neq b_j] = \Pr[\text{sign}(u[y]r_j[y]) = \text{sign}(r_j[y])]$
- This probability is 1 is $u[y]=+1$, 0 if $u[y]=-1$
- Completely biased by the answer to **INDEX**
Finishing the Reduction

- So what is $\Pr[w=0]$?
  - $w$ is sum of $(n-1)$ iid uniform $\{-1,+1\}$ values
  - Then: $\Pr[w=0] = 2^{-n}(n \text{ choose } n/2) = c/\sqrt{n}$, for some constant $c$

- Do some probability manipulation:
  - $\Pr[a_j = b_j] = \frac{1}{2} + c/2\sqrt{n}$ if $x[y]=1$
  - $\Pr[a_j = b_j] = \frac{1}{2} - c/2\sqrt{n}$ if $x[y]=0$

- Amplify this bias by making strings of length $N=4n/c^2$
  - Apply Chernoff bound on $N$ instances
  - With prob $>2/3$, either $\text{Ham}(a,b)>N/2 + \sqrt{N}$ or $\text{Ham}(a,b)<N/2 - \sqrt{N}$

- If we could solve $\text{Gap-Hamming}$, could solve $\text{INDEX}$
  - Therefore, need $\Omega(N) = \Omega(n)$ bits for $\text{Gap-Hamming}$
Streaming Lower Bounds

- Lower bounds for data streams
  - Communication complexity bounds
  - Simple reductions
  - Hardness of *Gap-Hamming* problem
  - Reductions to *Gap-Hamming*
Gap-Hamming instance—Alice: $x \in \{0,1\}^N$, Bob: $y \in \{0,1\}^N$

Entropy estimation algorithm $A$

- Alice runs $A$ on $\text{enc}(x) = \langle (1,x_1), (2,x_2), \ldots, (N,x_N) \rangle$
- Alice sends over memory contents to Bob
- Bob continues $A$ on $\text{enc}(y) = \langle (1,y_1), (2,y_2), \ldots, (N,y_N) \rangle$

<table>
<thead>
<tr>
<th>Alice</th>
<th>Bob</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1,0)</td>
<td>1</td>
</tr>
<tr>
<td>(2,1)</td>
<td>1</td>
</tr>
<tr>
<td>(3,0)</td>
<td>0</td>
</tr>
<tr>
<td>(4,0)</td>
<td>0</td>
</tr>
<tr>
<td>(5,1)</td>
<td>1</td>
</tr>
<tr>
<td>(6,1)</td>
<td>1</td>
</tr>
<tr>
<td>(1,1)</td>
<td>0</td>
</tr>
<tr>
<td>(2,1)</td>
<td>0</td>
</tr>
<tr>
<td>(3,0)</td>
<td>0</td>
</tr>
<tr>
<td>(4,0)</td>
<td>1</td>
</tr>
<tr>
<td>(5,1)</td>
<td>0</td>
</tr>
<tr>
<td>(6,0)</td>
<td>0</td>
</tr>
</tbody>
</table>
Lower Bound for Entropy

- Observe: there are
  - $2\text{Ham}(x,y)$ tokens with frequency 1 each
  - $N\text{-Ham}(x,y)$ tokens with frequency 2 each
- So (after algebra), $H(S) = \log N + \text{Ham}(x,y)/N = \log N + \frac{1}{2} \pm \frac{1}{\sqrt{N}}$
- If we separate two cases, size of Alice’s memory contents $= \Omega(N)$
  Set $\epsilon = \frac{1}{(\sqrt{N} \log N)}$ to show bound of $\Omega(\epsilon/\log 1/\epsilon)^{-2}$

<table>
<thead>
<tr>
<th>Alice</th>
<th>0</th>
<th>1</th>
<th>0</th>
<th>0</th>
<th>1</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(1,0)</td>
<td>(2,1)</td>
<td>(3,0)</td>
<td>(4,0)</td>
<td>(5,1)</td>
<td>(6,1)</td>
</tr>
<tr>
<td>Bob</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>(1,1)</td>
<td>(2,1)</td>
<td>(3,0)</td>
<td>(4,0)</td>
<td>(5,1)</td>
<td>(6,0)</td>
</tr>
</tbody>
</table>
Lower Bound for $F_0$

- Same encoding works for $F_0$ (Distinct Elements)
  - 2$\text{Ham}(x,y)$ tokens with frequency 1 each
  - $N$-$\text{Ham}(x,y)$ tokens with frequency 2 each

- $F_0(S) = N + \text{Ham}(x,y)$

- Either $\text{Ham}(x,y) > N/2 + \sqrt{N}$ or $\text{Ham}(x,y) < N/2 - \sqrt{N}$
  - If we could approximate $F_0$ with $\varepsilon < 1/\sqrt{N}$, could separate
  - But space bound = $\Omega(N) = \Omega(\varepsilon^{-2})$ bits

- Dependence on $\varepsilon$ for $F_0$ is tight

- Similar arguments show $\Omega(\varepsilon^{-2})$ bounds for $F_k$
  - Proof assumes $k$ (and hence $2^k$) are constants
Summary of Tools

- Vector equality: fingerprints
- Approximate item frequencies:
  - Count-min, Misra-Gries ($L_1$ guarantee), Count sketch ($L_2$ guarantee)
- Euclidean norm, inner product: AMS sketch, JL sketches
- Count-distinct: k-Minimum values, Hyperloglog
- Compact set-representation: Bloom filters
- Uniform Sampling
- $L_0$ sampling: hashing and sparse recovery
- $L_2$ sampling: via count-sketch
- Graph sketching: $L_0$ samples of neighborhood
- Frequency moments: via $L_2$ sampling
- Matrix sketches: adapt AMS sketches, frequent directions
Summary of Lower Bounds

- Can’t deterministically test equality
- Can’t retrieve arbitrary bits from a vector of $n$ bits: $\text{INDEX}$
  - Even if some unhelpful suffix of the vector is given: $\text{AUGINDEX}$
- Can’t determine whether two $n$ bit vectors intersect: $\text{DISJ}$
- Can’t distinguish small differences in Hamming distance: $\text{GAP-HAMMING}$

These in turn provide lower bounds on the cost of
- Finding the maximum frequency
- Approximating the number of distinct items
- Approximating matrix multiplication
Current Directions in Streaming and Sketching

- **Sparse representations** of high dimensional objects
  - Compressed sensing, sparse fast Fourier transform
- **Numerical linear algebra** for (large) matrices
  - k-rank approximation, linear regression, PCA, SVD, eigenvalues
- **Computations on large graphs**
  - Sparsification, clustering, matching
- **Geometric** (big) data
  - Coresets, facility location, optimization, machine learning
- **Use of summaries in** distributed computation
  - MapReduce, Continuous Distributed models
Forthcoming Attractions

- Data Streams Mini Course @Simons
  - Prof Andrew McGregor
  - Starts early October

- Succinct Data Representations and Applications @ Simons
  - September 16-19