Sketching

CS 584: Big Data Analytics

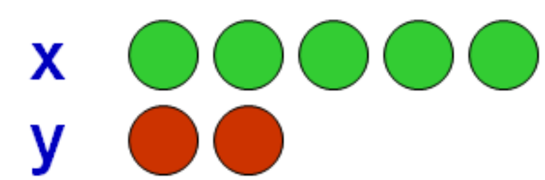
Material adapted from Graham Cormode (https://simons.berkeley.edu/sites/default/files/docs/529/cormodeslides.pdf), Andrew McGregor (https://people.cs.umass.edu/~mcgregor/slides/10-jhu1.pdf), & Minos Garafalkis (http://www.cs.berkeley.edu/~brewer/cs262/Minos-StreamingData2)

Data is Massive

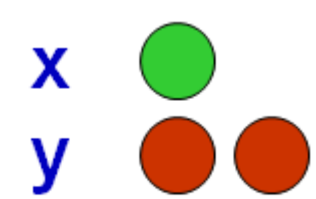
- Data is growing faster than our ability to store or index it
- Traditional DBMS (finite and persistent) vs data streams (distributed, continuous, time-varying, …)
- Variety of modern applications
  - Network monitoring & sensor networks
  - Web logs and click streams
  - Genome sequences
  - Other massive data sets

Streaming Data Models

- Data is a collection of simple tuples
- Problems hard due to scale and dimension of input
- Arrivals only (Cash register model):
    - Represent packets in a network, power usage, …
    - Example: (x, 3), (y, 2), (x, 2)



- Arrivals and departures (Turnstile model):
    - Represent fluctuating quantities or differences between distributions
    - Example: (x, 3), (y, 2), (x, -2)

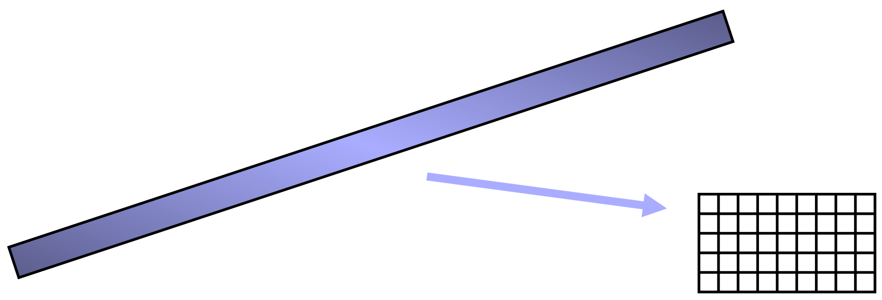


Sketches

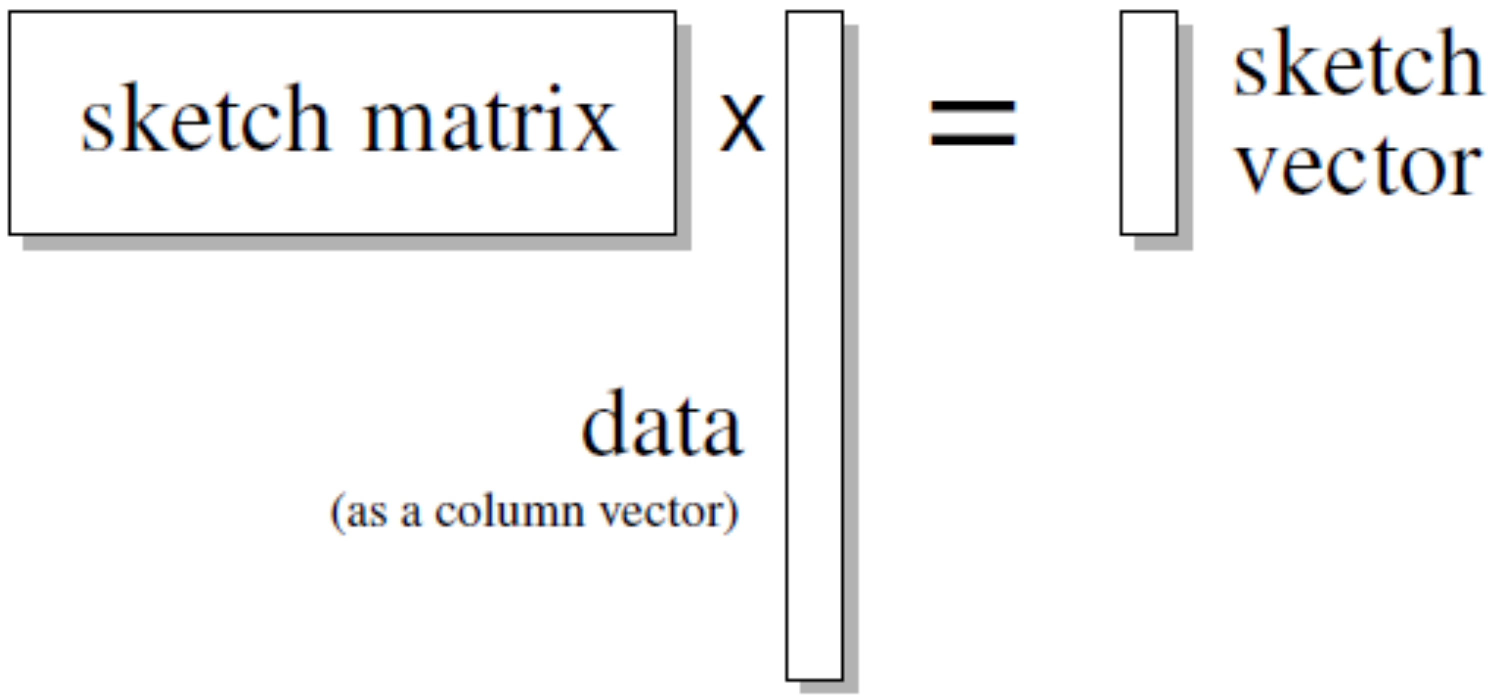
- Not every problem can be solved with sampling (e.g., counting number of distinct items in the stream)

- General technique for processing streams where an algorithm can "see" all the data even if it can't remember it all

- Basic idea: apply a linear projection "on the fly" that takes high-dimensional data to a smaller dimensional space



Linear Sketching

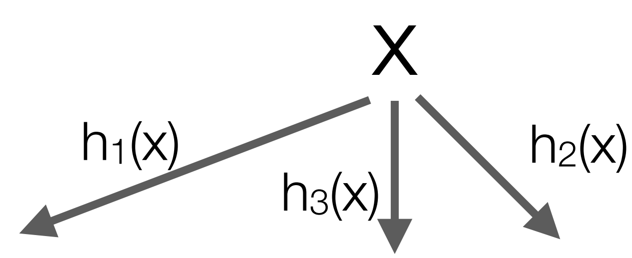


- Sketch is linear transform of input (Sketch(x) = Sx)
- Trivial to update and merge
- Often describe S in terms of hash functions: if hash functions are simple, sketch is fast

Bloom Filter

Compactly encode set membership

- k hash functions map to bit vector k times
- Set the bits h(x) to 1 for each hash function for input x
- Can lookup items, store set of size n in O(n) bits



1	2	3	4	5	6	7	8	9	10
0	0	1	0	1	1	0	0	0	0

Bloom Filter Analysis

- How to set k (number of hash functions) and m (size of filter)?
- False positive: When all k locations for an item are set
  - Probability of empty cell with n items:

$$P(\text{cell empty}) = (1 - 1/m)^{kn} \approx \exp(-kn/m) = \rho$$

  - Probability of false positive:

$$P(\text{FP}) = (1 - \rho)^k = \exp(-m/n \ln(\rho) \ln(1 - \rho))$$

Bloom Filter Applications

- Widely used in “big data” applications
  - Many problems require storage of large set of items
  - Small false positive rates and if it says “never seen”, then it is truly new
- Can be cleared every so often to decrease false positive probability
- Can be generalized to allow deletions and represent multisets

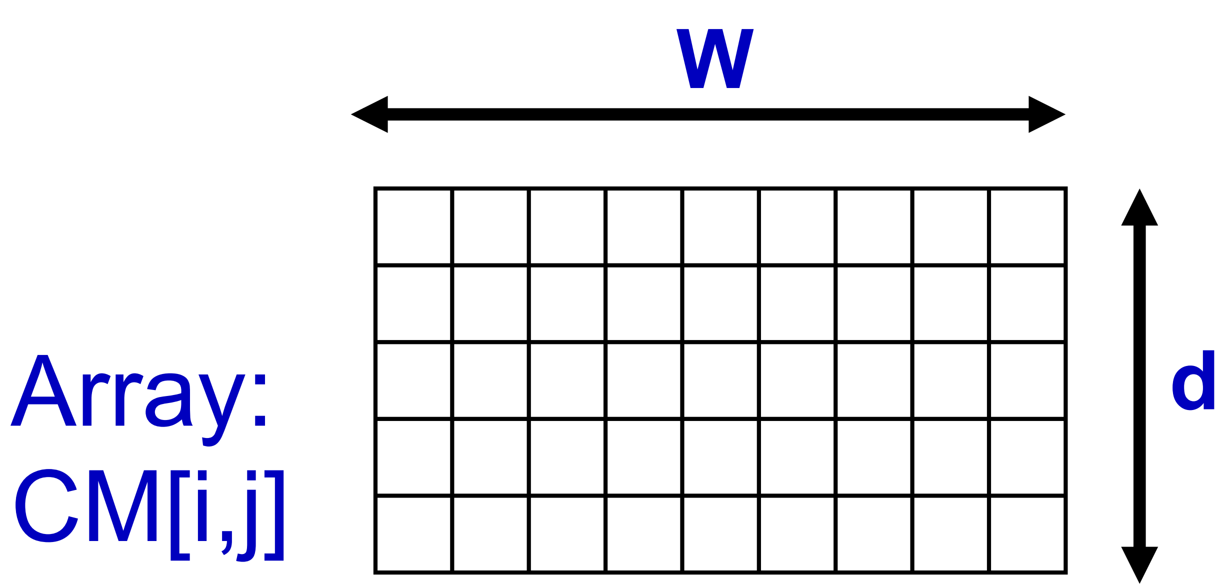
Count-Min Sketch [Cormode & Muthukrishnan, 2004]

- Simple technique to summarize large amounts of frequency data
  - Basis of many different stream mining tasks
- Useful multi-purpose sketch
  - Heavy hitters: Find all i such that $f_i \geq \phi m$
  - Range sums: Estimate $\sum_{i \leq k \leq j} f_k$
  - Find k-quantiles: Find values such that

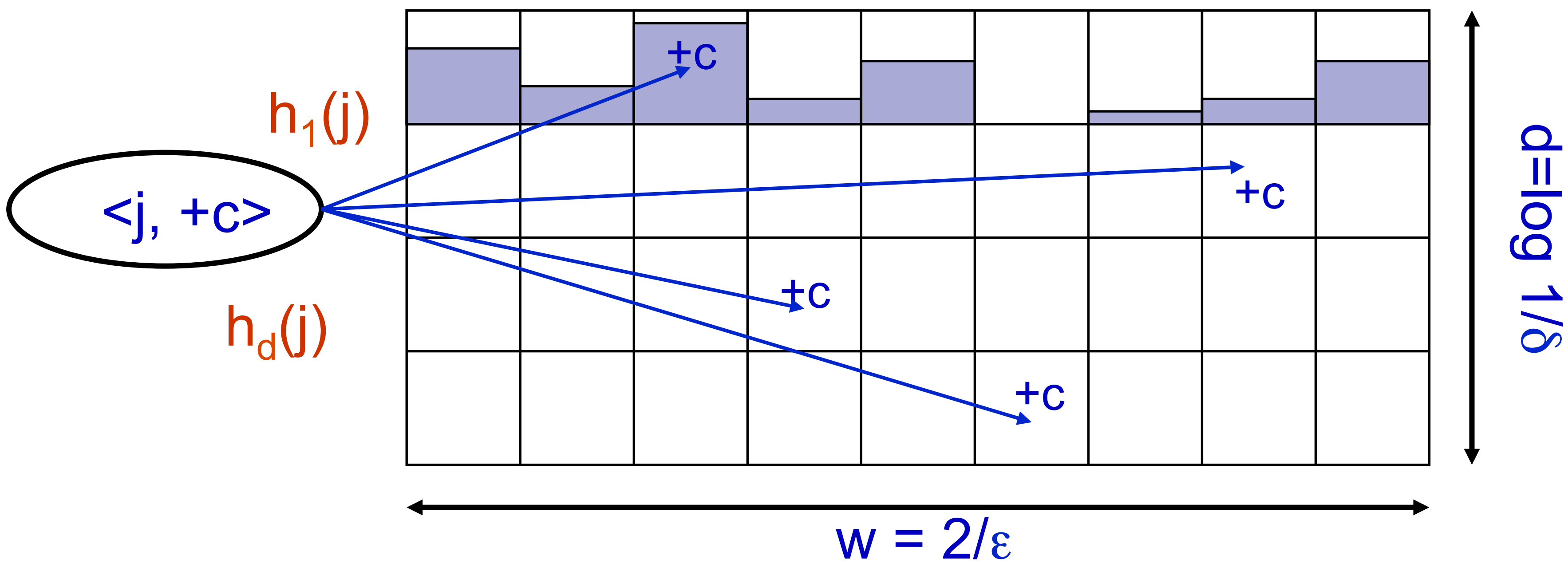
$$q_0 = 0, q_k = n, \sum_{i \leq q_i - 1} f_i < \frac{jm}{k} \leq \sum_{i \leq q_i} f_i$$

Count-Min Algorithm

- Model input data as a vector x of dimension U
- Create a small summary as a w x d array
- Use d hash functions to map vector entries to [1, w]
- Works on arrivals only and arrivals and departure streams



Count-Min Sketch Structure



- Each entry in input 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 $\varepsilon||A||_1$ in size $O(1/\varepsilon \log 1/\delta)$

Count-Min Approximate Query

- Point query: $Q(i) = \min_k \mathrm{CM}[k, h_k(i)]$
- Range query:

$$Q(i,j) = \sum_{\ell=i}^{j} \min_k \mathrm{CM}[k, h_k(\ell)]$$

- Inner product query:

$$Q(\mathbf{a}, \mathbf{b}) = \sum_{i=1}^{n} (\min_k \mathrm{CM}[k, h_k(a_i)])(\min_k \mathrm{CM}[k, h_k(b_i)])$$

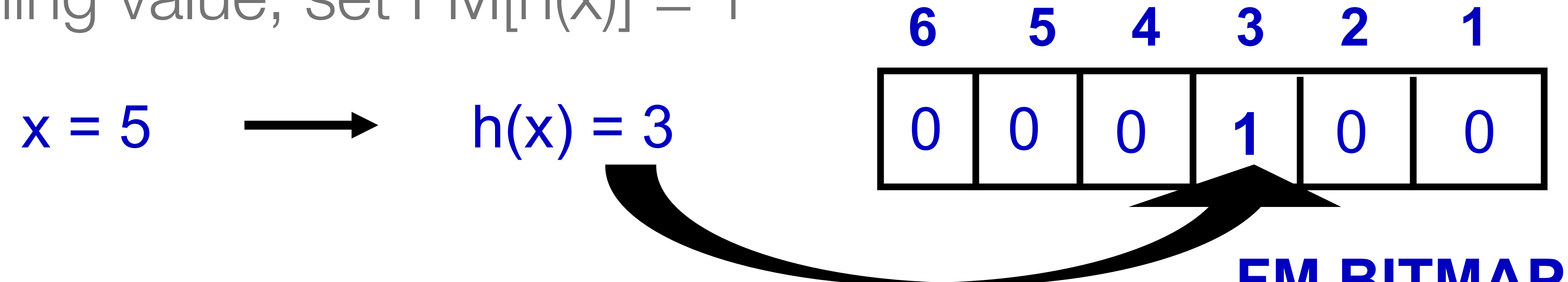
Counts are biased (overestimates) due to collisions

Counting Distinct Elements

- Find the number of distinct values in a stream
  - Equivalent to F_0 frequency moment or L_0 (hamming) stream norm
- Hard problem for random sampling [Charikar et al., 2000]
  - Must sample almost the entire table to guarantee the estimate is within a factor of 10 with a probability greater than half, regardless of ester used

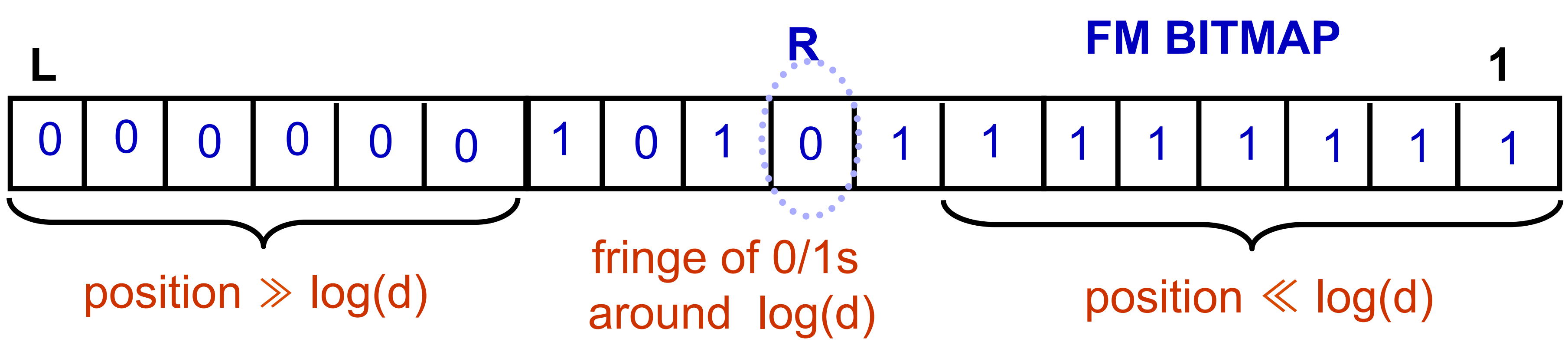
FM Sketch [Flajolet & Martin, 1983]

- Uses hash function to map input items to i with probability 2^{-i}
  - P[h(x) = 1] = 1/2, P[h(x)] 2] = 1/4, …
  - Easy to construct h from a uniform hash function by counting trailing zeroes
- Maintain bitmap array of L = log N bits
  - Initialize bitmap to all 0s
  - Each incoming value, set FM[h(x)] = 1



FM Sketch Analysis

- With d distinct values, expect d/2 map to FM[1], d/4 map to FM[2], …



- Let R = position of rightmost zero in FM, indicator of log(d)

- Basic estimate is $d = c\ 2^R$ with scaling constant $c = 1.3$

Counting Distinct Elements Algorithm

Many algorithms have been introduced to solve this problem

- Flajolet & Martin: O(log n) space for fixed ε in random oracle model
- Alon, Matias, and Szegedy: O(log n) space/update time for fixed ε with no oracle
- Gibbons & Tirthapura: $O(\varepsilon^{-2} \log n)$ space and $O(\varepsilon^{-2})$ update time
- Bar-Yossef et al.: $O(\varepsilon^{-2} \log n)$ space and $O(\log 1/\varepsilon)$ update time
- Kane, Nelson, and Woodruff: $O(\varepsilon^{-2} + \log n)$ space and O(1) update and reporting time

Other Sketches

- Different sketches for other frequency moments
  - AMS sketch for F_2 (second frequency moment)
  - Higher frequency moments (F_k for $k > 2$)
  - Combined frequency moments
- Graph sketching (connectivity, minimum spanning tree, …)
- Matrix multiplication (given A, B, approximate AB)
- Lower bounds for streaming and sketching (communication and information complexity bounds)

Summary

- Sampling and sketching are heart of many stream mining algorithms
- Sample is a general representation of the data set, sketches are specific to a particular purpose
  - Traditional sampling does not work in the turnstile (arrivals & departure models)
- Algorithms are quite simple and very fast
  - Limiting factor in practice is often I/O related