Tackling the Challenges of Big Data
Big Data Analytics

Piotr Indyk
Professor
Massachusetts Institute of Technology

Tackling the Challenges of Big Data
Big Data Analytics
Fast Algorithms II & Streaming and Sampling
Introduction

Piotr Indyk
Professor
Massachusetts Institute of Technology

Streaming & Sampling

Two approaches to dealing with massive data using limited resources

Sampling: pick a random sample of the data, and perform the computation on the sample

8 2 1 9 1 9 2 4 6 3 9 4 2 9 4 9 3 9 5 9 5 6 …

Streaming: make a single pass over the whole data; maintain a `sketch' of the data set from which the desired properties can be inferred

8 2 1 9 1 9 2 4 6 3 9 4 2 9 4 9 3 9 5 9 5 6 …

Streaming vs. Sampling

Sampling pros:

- Computation is performed only on the sample - reduced storage and computation time
- Only the sampled data elements are needed; the remaining elements do not even need to be materialized

Streaming pros:

- Every data element is seen at least once, so `no element is left behind'.
- E.g., does the data set contain any 1 ?
- Reduced storage, computation time near-linear in the data size

Rest of This Lecture

Streaming:

- Estimate the number of distinct elements in the stream in limited space, up to $1 \pm \varepsilon$ error
- Need only 128 bytes for all works of Shakespeare with error $\varepsilon \approx 10\%$
- Other problems solvable using streaming algorithms

Sampling:

- Recent algorithms for sparse Fourier Transform that estimate large coefficients in the spectrum of a signal using few signal samples

Tackling the Challenges of Big Data
Big Data Analytics
Fast Algorithms II & Streaming and Sampling

Introduction

THANK YOU

PROFESSIONAL EDUCATION

CSAIL

Tackling the Challenges of Big Data
Big Data Analytics

Piotr Indyk
Professor
Massachusetts Institute of Technology

Tackling the Challenges of Big Data
Big Data Analytics

Fast Algorithms II & Streaming and Sampling

Streaming Algorithms

Piotr Indyk
Professor
Massachusetts Institute of Technology

© 2014 Massachusetts Institute of Technology

Streaming Algorithms

Single pass over the data: $i_1, i_2, \ldots, i_n$

- Typically, we assume the number of data items n is known, at least approximately

Bounded storage (e.g. $n^{1/2}$ or $\log^2(n)$)

- Units of storage: bits, bytes or data elements

Fast processing time per element

Streaming Algorithms, ctd.

- Most algorithms are randomized (they use pseudo-random numbers) and approximate
- That is, they report an estimate such that
  $\Pr[\text{Estimate}=\text{Truth } (1\pm\varepsilon)]>1-P$
- This is often (but not always) necessary

Counting Distinct Elements

- Stream elements: integer numbers from 1…m
  8 2 1 9 1 9 2 4 6 3 9 4 2 9 4 9 3 9 5 9 5 6 …
- Goal: estimate the number of distinct elements DE in the stream
  - Up to $1\pm\varepsilon$
  - With probability 1-P
- Simpler goal: for a given T>0, provide an algorithm which, with probability 1-P:
  - Answers `DE>T' if DE> $(1+\varepsilon)$T
  - Answers `DE<T' if DE< $(1-\varepsilon)$T

Tackling the Challenges of Big Data
Big Data Analytics
Fast Algorithms II & Streaming and Sampling
Streaming Algorithms

THANK YOU

Tackling the Challenges of Big Data
Big Data Analytics

Piotr Indyk
Professor
Massachusetts Institute of Technology

Tackling the Challenges of Big Data
Big Data Analytics
Fast Algorithms II & Streaming and Sampling
Distinct Elements

Piotr Indyk
Professor
Massachusetts Institute of Technology

Counting Distinct Elements

- Stream elements: integer numbers from 1…m
- Goal: estimate the number of distinct elements DE in the stream
  - Up to 1±ε
  - With probability 1-P
- Simpler goal: for a given T>0, provide an algorithm which, with probability 1-P:
  - Answers `DE>T' if DE> (1+ε)T
  - Answers `DE<T' if DE< (1-ε)T

Count Vector Interpretation

- Stream: 8 2 1 9 1 9 2 4 4 9 4 2 5 4 2 5 8 5 2 5

Vector c:

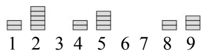
- Initially, c=0
- Arrival of i is interpreted as

$$c_i = c_i + 1$$

- Want to estimate the number of non-zero entries in c, denoted by NZ(c)

Estimating NZ(c)

Vector c:



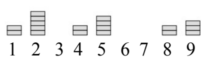
- Preprocessing:
  - Select k random sets $S_0 \ldots S_{k-1}$ of coordinates such that, for each i, we have

    $$\Pr[i \in S_j] = 1/T$$

  - Important: do not store S explicitly. Instead, to test if $i \in S_j$:
    - Set the pseudo-random seed to j+k*i
    - Generate a pseudo-random number R(i,j)
    - Check if R(i,j) mod T = 0

Estimating NZ(c), continued

Vector c:



- Computation: For each j, compute $sum_j = \Sigma_{i \in S_j} c_i$
  - For each j, let $sum_j=0$
  - For each stream element i
    - For each j, if $i \in S_j$ then $sum_j=sum_j+1$

Estimating NZ(c), continued

Vector c:

- After we computed $sum_j = \Sigma_{i \in S_j} c_i$ we estimate:
  - Let Z = number of sum_j that are equal to 0
  - If Z>k/e then report `NZ < T' else report `NZ>T'

- Intution: if NZ is small compared to T, then the non-zero coordinates are not likely to belong to sets S_j. Therefore Z will be large.

Tackling the Challenges of Big Data
Big Data Analytics
Fast Algorithms II & Streaming and Sampling
Distinct Elements

THANK YOU

Tackling the Challenges of Big Data
Big Data Analytics

Piotr Indyk
Professor
Massachusetts Institute of Technology

Tackling the Challenges of Big Data
Big Data Analytics
Fast Algorithms II & Streaming and Sampling
Distinct Elements - Analysis

Piotr Indyk
Professor
Massachusetts Institute of Technology

Performance

- The described algorithm is a variant of [Flajolet-Martin, FOCS'83]
- Best theoretical result: $O(1/\varepsilon^2 + \log n)$ bits [Kane-Nelson-Woodruff, PODS'10]
- Practice: need only 128 bytes for all works of Shakespeare, with error $\varepsilon \approx 10\%$
- LogLog, HyperLogLog [Durand-Flajolet, ESA'03]

Other Tasks

- **Heavy hitters** (a.k.a. elephants) [Misra-Gries'82, Charikar-Chen-FarachColton'02, Estan-Varghese'03, Cormode-Muthukrishnan'04,'05, Cormode-Hadjieleftheriou'07,…]
  - Finds coordinates i such that $|c_i|$ is "large"
  - Estimates $c^*_i = c_i \pm \varepsilon n$
- **Entropy** [DoBa-Chakrabarti-Muthukrishnan'05, Guha-McGregor-Venkatasubramanian'05, Chakrabarti-Cormode-McGregor'06, Bhuvanagiri-Ganguly'06, Harvey-Nelson-Onak'08]
- **Independence testing** [Indyk-McGregor'08]
- **Median, quantiles, histograms** [Munro-Paterson'80, Manku-Rajagopalan-Lindsay'98,'99, Greenwald-Khanna'02, Gilbert-Guha-Indyk-Kotidis-Muthukrishnan-Strauss'02,…]

Tackling the Challenges of Big Data
Big Data Analytics
Fast Algorithms II & Streaming and Sampling
Distinct Elements - Analysis

THANK YOU

PROFESSIONAL EDUCATION
CSAIL

Tackling the Challenges of Big Data
Big Data Analytics

Piotr Indyk
Professor
Massachusetts Institute of Technology

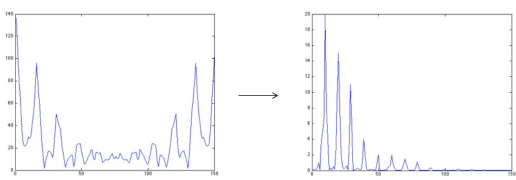
PROFESSIONAL EDUCATION
CSAIL

Tackling the Challenges of Big Data
Big Data Analytics
Fast Algorithms II & Streaming and Sampling
Sampling Algorithms for the Sparse Fourier Transform

Piotr Indyk
Professor
Massachusetts Institute of Technology

Discrete Fourier Transform

- Given a signal, compute its spectrum



Applications: Audio, Video, GPS, Radar, Sequencing,…

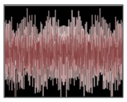
Computing Fourier Transform

- Naïve Algorithm $O(n^2)$
- In 1965, Cooley and Tukey introduced the FFT which computes the frequencies in $O(n \log n)$
- But … FFT is too slow for BIG Data problems

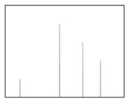
Can we design a sublinear Fourier algorithm?

Idea: Leverage Sparsity

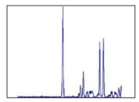
Often the Fourier Transform is dominated by a few peaks



Time Signal



Sparse Freqs.



Approximately Sparse Freqs.

Sparsity appears in video, audio, seismic data, telescope/satellite data, medical tests, genomics

Sparse FFT runs in sub-linear time on sparse data

Benefits of Sparse Fourier Transform

- Faster computation
  ◊Scalable to larger datasets
- Use only samples of the data
  ◊ Lower acquisition time
  ◊ Less communication bandwidth
- Lower power consumption

Tackling the Challenges of Big Data
Big Data Analytics
Fast Algorithms II & Streaming and Sampling

Sampling Algorithms for the Sparse Fourier Transform

THANK YOU

Tackling the Challenges of Big Data © 2014 Massachusetts Institute of Technology

Tackling the Challenges of Big Data
Big Data Analytics

Piotr Indyk

Professor

Massachusetts Institute of Technology

Tackling the Challenges of Big Data
Big Data Analytics
Fast Algorithms II & Streaming and Sampling
Sparse Fourier Transform - Algorithm and Performance

Piotr Indyk

Professor

Massachusetts Institute of Technology

Basic Ideas

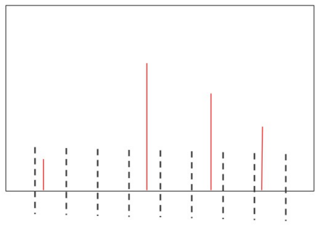
1- Bucketize

Divide spectrum into a few buckets

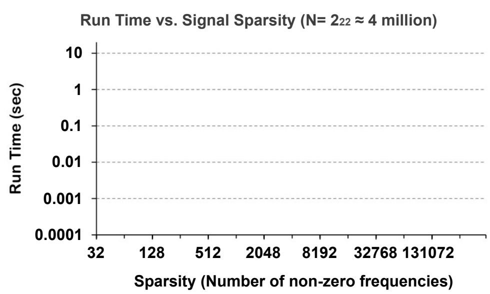
◊ Can ignore empty bucket

2- Estimate

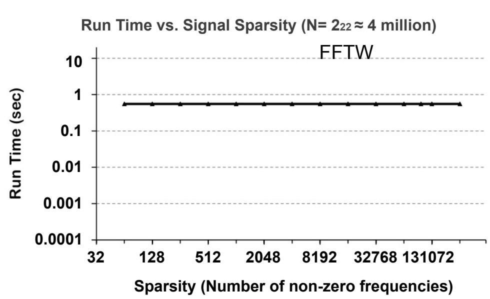
Estimate the large coefficient in each non-empty bucket



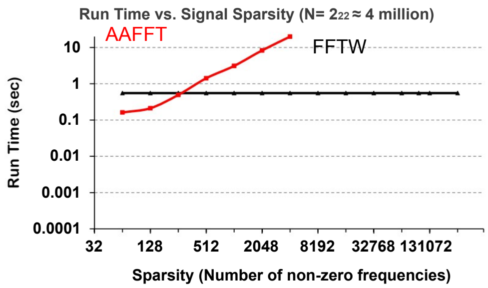
Results



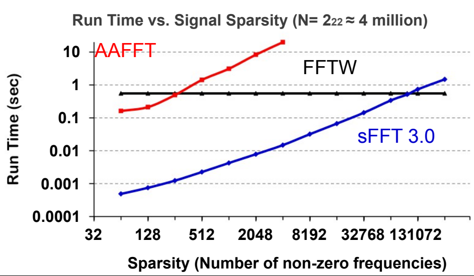
Results



Results



Results



- **Linear in sparsity, not signal length**
- **Orders of magnitude faster than FFT on sparse d**

Current Directions

- Hardware for a million-point Fourier Transform
- Applications:
  - GPS
  - Smaller and cheaper 3D cameras
  - Medical Imaging - NMR, and MRI

References

- SFFT 1.0, 2.0 [Hassanieh-Indyk-Katabi-Price, SODA'12]
- SFFT 3.0 [Hassanieh-Indyk-Katabi-Price, STOC'12]
- Sparse Fourier Transform website: http://groups.csail.mit.edu/netmit/sFFT/

Tackling the Challenges of Big Data
Big Data Analytics
Fast Algorithms II & Streaming and Sampling
Sampling Algorithms for the Sparse Fourier Transform

THANK YOU