Course logistics, Streaming, Sampling

Lecture 1

August 25, 2020

Logistics

- Website has most of the relevant information. Ask if you are unsure. Some information such as Zoom links etc will get updated periodically so check periodically.
- Lectures via Zoom are synchronous: Tue/Thu 9.30-10.45am. Videos available by end of day (modulo technical glitches)
- See instructions on website if you want to be anonymous on video recordings.
- All announcements on Piazza. Check regularly (once a day). Use private posts on Piazza to communicate with course staff for non-urgent matters. Use email to instructor/TA if matter is time-sensitive or confidential.
- All homeworks and project to be submitted via Gradescope
- Exam logistics not finalized yet. Will be announced on Piazza.

Covid-19 and Online Aspects

Unusual situation due to pandemic and remote learning

- Follow a regular schedule as much as possible
- Keep up with lectures and attend office hours as needed, seek out collaborations and discussions with fellow classmates
- Seek help promptly and early if you have any issues or concerns. Do not be shy about contacting course staff for any accommodations that you may need.
- Be kind to yourself and others. Be aware of mental health issues.

Homework, Exams and Grading Policies

Grade based on:

- 4-5 homeworks for 40% (to be submitted on Gradescope)
  - No late submissions by default
  - Will drop few problems to compensate
- 2 midterms for total 40%
- project for 20%

Homework is biweekly but **strongly** encouraged to work each week.

Other important issues

- Mental health
- Anti-racism, inclusivity, bias
- Sexual harassment and reporting
- Academic integrity: be aware of the rules as well as your conscience
- Disability resources: If you have/need DRES accommodations please contact instructor as soon as possible.
- Religious observances
- FERPA rights
- See webpage with links to college of engineering and campus resources and information.

Other important issues

- Mental health
- Anti-racism, inclusivity, bias
- Sexual harassment and reporting
- Academic integrity: be aware of the rules as well as your conscience
- Disability resources: If you have/need DRES accommodations please contact instructor as soon as possible.
- Religious observances
- FERPA rights
- See webpage with links to college of engineering and campus resources and information.

Always feel free to approach the instructor even when you are unsure.

Course Topics

This is a theory course focused on rigorous guarantees and formal analysis of algorithms. Practical applications will be discussed but not the main focus.

- Background in probability/randomized algorithms and some technical tools
- Streaming model and algorithms in the model
  - Sampling
  - Frequency moments
  - Sketching
  - Quantiles and selection
  - Graph streams and sketches
- Dimensionality reduction and related topics
- Similarity estimation, locality sesitivity hashing
- Coresets and clustering
- Fast numerical linear algebra

Applications of course material

- Mining Massive Data Sets by Leskovic, Rajaraman, Ullman. Book, MOOC and Slides at `www.mmds.org`.
- Apache DataSketches: a software library for stochastic streaming algorithms. `datasketches.apache.org`

Part I

Streaming Model

Streaming model

- The input consists of m objects/items/tokens $e_1, e_2, \ldots, e_m$ that are seen one by one by the algorithm.
- The algorithm has "limited" memory say for B tokens where $B < m$ (often $B \ll m$) and hence cannot store all the input
- Want to compute interesting functions over input

Streaming model

- The input consists of m objects/items/tokens $e_1, e_2, \ldots, e_m$ that are seen one by one by the algorithm.
- The algorithm has "limited" memory say for B tokens where $B < m$ (often $B \ll m$) and hence cannot store all the input
- Want to compute interesting functions over input

Some examples:

- Each token in a number from $[n]$
- High-speed network switch: tokens are packets with source, destination IP addresses and message contents.
- Each token is an edge in graph (graph streams)
- Each token in a point in some feature space
- Each token is a row/column of a matrix

Streaming model

- The input consists of m objects/items/tokens $e_1, e_2, \ldots, e_m$ that are seen one by one by the algorithm.
- The algorithm has "limited" memory say for B tokens where $B < m$ (often $B \ll m$) and hence cannot store all the input
- Want to compute interesting functions over input

Some examples:

- Each token in a number from $[n]$
- High-speed network switch: tokens are packets with source, destination IP addresses and message contents.
- Each token is an edge in graph (graph streams)
- Each token in a point in some feature space
- Each token is a row/column of a matrix

Question: What are the tradeoffs between memory size, accuracy, randomness and other resources?

Streaming model: motivation/connections

- Very large but slow storage (tape, slow disk) that is suited for sequential access and fast main memory. Read data in one (or more) passes from slow medium.
- Scenarios such as network switches, sensors etc where huge amount of data is flying by and cannot be stored (due to cost or privacy/legal reasons) but one wants only high-level statistics.
- Distributed computing. Data stored in multiple machines. Cannot send all data to central location. Streaming algorithms can simulate a class of algorithms that exchange small amount of data. Leads to sketching.

Streaming model: some early papers

- Munro, J. Ian; Paterson, Mike (1978). "Selection and Sorting with Limited Storage". 19th Annual Symposium on Foundations of Computer Science, 1978.
- Morris, Robert (1978), "Counting large numbers of events in small registers", Communications of the ACM.
- Misra, J.; Gries, David (1982). "Finding repeated elements". Science of Computer Programming.
- Flajolet, Philippe; Martin, G. Nigel (1985). "Probabilistic counting algorithms for data base applications". JCSS.
- Alon, Noga; Matias, Yossi; Szegedy, Mario (1996), "The space complexity of approximating the frequency moments", Proceedings of 28th STOC. *Winner of the Goedal Prize in TCS*.

Streaming: Approximation and Randomization

Question: What are the tradeoffs between memory size, accuracy, randomness and other resources?

Ideal scenario: compute some quantity of interest in very little space compared to input stream length and deterministically.

- Sub-linear: say $\sqrt{m}$ tokens where m is length of stream
- Near-optimal: $O(\text{poly}(\log m))$

Streaming: Approximation and Randomization

Question: What are the tradeoffs between memory size, accuracy, randomness and other resources?

Ideal scenario: compute some quantity of interest in very little space compared to input stream length and deterministically.

- Sub-linear: say $\sqrt{m}$ tokens where m is length of stream
- Near-optimal: $O(\text{poly}(\log m))$

Bad news: For even very simple problems strong lower bounds (essentially linear sapce) if one wants exact answers

Good news: Several interesting and useful results if one allows randomization and approximation

Part II

Sampling

Sampling

Random sampling is a powerful and general tool in data analysis. We will see several variants and applications.

- Pick a small random set S from a large set
- Estimate quantity of interest on S instead of entire data set
- Analysis relies on sampling strategy, sample size, and estimation algorithm

Sampling

Random sampling is a powerful and general tool in data analysis. We will see several variants and applications.

- Pick a small random set S from a large set
- Estimate quantity of interest on S instead of entire data set
- Analysis relies on sampling strategy, sample size, and estimation algorithm

Basic sampling strategy: *uniform* sample of size k from set of size m

- with replacement: pick a uniformly random number $i \in [m]$ and repeat independently k times. same element can be picked multiple times
- without replacement: pick a single set uniformly from all sets of size k (of cardinality $\binom{m}{k}$).

Reservoir Sampling

Question: How do we pick a *single* uniform sample without knowing length of stream in advance?

Reservoir Sampling

Question: How do we pick a *single* uniform sample without knowing length of stream in advance?

How do we pick if we knew the length of stream in advance?

Reservoir Sampling

Question: How do we pick a *single* uniform sample without knowing length of stream in advance?

How do we pick if we knew the length of stream in advance?

- Say length is m
- Pick a random integer r in $\{1, 2, \ldots, m\}$
- Store r'th element of stream as sample

Assumption: Algorithm has access to random numbers/bits.

Reservoir Sampling

Question: How do we pick a *single* uniform sample without knowing length of stream in advance?

How do we pick if we knew the length of stream in advance?

- Say length is m
- Pick a random integer r in $\{1, 2, \ldots, m\}$
- Store r'th element of stream as sample

Assumption: Algorithm has access to random numbers/bits.

Digression: Suppose algorithm has access only to random bits. How can one choose a random integer r in $\{1, 2, \ldots, m\}$?

Digression: Rejection Sampling

Suppose algorithm has access only to random bits. How can one choose a random integer r in $\{1, 2, \ldots, m\}$?

Digression: Rejection Sampling

Suppose algorithm has access only to random bits. How can one choose a random integer r in $\{1, 2, \ldots, m\}$?

- Let $k = \lceil \log m \rceil$
- Use k random bits to generate an integer r uniformly in $\{1, 2, \ldots, 2^k\}$
- If $r \in \{1, 2, \ldots, m\}$ output r Else reject r and repeat

Digression: Rejection Sampling

Suppose algorithm has access only to random bits. How can one choose a random integer r in $\{1, 2, \ldots, m\}$?

- Let $k = \lceil \log m \rceil$
- Use k random bits to generate an integer r uniformly in $\{1, 2, \ldots, 2^k\}$
- If $r \in \{1, 2, \ldots, m\}$ output r Else reject r and repeat

Question: What is expected number of iterations to generate a "good sample"?

Digression: Rejection Sampling

Suppose algorithm has access only to random bits. How can one choose a random integer r in $\{1, 2, \ldots, m\}$?

- Let $k = \lceil \log m \rceil$
- Use k random bits to generate an integer r uniformly in $\{1, 2, \ldots, 2^k\}$
- If $r \in \{1, 2, \ldots, m\}$ output r Else reject r and repeat

Question: What is expected number of iterations to generate a "good sample"? At most 2. Why?

Reservoir Sampling

Question: How do we pick a *single* uniform sample without knowing length of stream in advance?

Reservoir Sampling

Question: How do we pick a *single* uniform sample without knowing length of stream in advance?

UNIFORMSAMPLE:
$s \leftarrow$ null
$m \leftarrow 0$
While (stream is not done)
  $m \leftarrow m + 1$
  e_m is current item
  Toss a biased coin that is heads with probability $1/m$
  If (coin turns up heads)
    $s \leftarrow e_m$
endWhile
Output s as the sample

Reservoir Sampling: Claim

Lemma

Let m be the length of the stream. The output of the algorithm s is uniform. That is, for any $1 \leq j \leq m$, $\Pr[s = e_j] = 1/m$.

Reservoir Sampling: Claim

Lemma

Let m be the length of the stream. The output of the algorithm s is uniform. That is, for any $1 \leq j \leq m$, $\Pr[s = e_j] = 1/m$.

Proof.

We observe that $s = e_j$ if e_j is chosen when it is considered by the algorithm (which happens with probability $\frac{1}{j}$), and none of $e_{j+1}, \ldots, e_m$ are chosen to replace e_j. All the relevant events are independent and we can compute:
$\Pr[s = e_j] = \frac{1}{j} \times \prod_{i>j}(1 - 1/i) = 1/m$. □

Can also prove by induction on m.

Reservoir Sampling: k samples

Want to pick k samples for $k > 1$. How?

Reservoir Sampling: k samples

Want to pick k samples for $k > 1$. How?

- With replacement. Easy, simply run single sample algorithm independently in parallel and store the k samples.

Reservoir Sampling: k samples

Want to pick k samples for $k > 1$. How?

- With replacement. Easy, simply run single sample algorithm independently in parallel and store the k samples.
- Without replacement?

k samples without replacement

```
Sample-without-Replacement(k):
S[1..k] ← null
m ← 0
While (stream is not done)
   m ← m + 1
   e_m is current item
   If (m ≤ k) S[m] ← e_m
   Else
         r ← uniform random number in range [1..m]
         If (r ≤ k) S[r] ← e_m
endWhile
Output S
```

Exercise: Prove correctness of algorithm.

k samples without replacement: alternative

```
SAMPLE-WITHOUT-REPLACEMENT(k):
S[1..k] ← null
m ← 0
While (stream is not done)
    m ← m + 1, e_m is current item
    Pick random real number θ_m ∈ (0, 1)
    Store in S the min{k, m} items with largest θ values
endWhile
Output S
```

Exercise: How will you implement in streaming setting with $O(k)$ space? Prove correctness of algorithm.

Weighted Sampling

Stream has m items $e_1, \ldots, e_m$. Each item has weight $w_i > 0$. Want to pick item i in proportion to weight (useful in various settings). Formally $\Pr[e_i \text{ is chosen}] = w_i/W$ where $W = \sum_{i=1}^m w_i$.

Weighted Sampling

Stream has m items $e_1, \ldots, e_m$. Each item has weight $w_i > 0$. Want to pick item i in proportion to weight (useful in various settings). Formally $\Pr[e_i \text{ is chosen}] = w_i / W$ where $W = \sum_{i=1}^{m} w_i$.

<u>SINGLE WEIGHTED SAMPLE:</u>

$s \leftarrow$ null, $m \leftarrow 0$, $W = 0$
While (stream is not done)
   $m \leftarrow m + 1$, $W \leftarrow W + w_m$
   e_m is current item
   Toss a biased coin that is heads with probability w_m / W
   If (coin turns up heads)
       $s \leftarrow e_m$
endWhile
Output s as the sample

Weighted Sampling: k samples

With replacement is easy. Without replacement? What does sampling without replacement mean?

Weighted Sampling: k samples

With replacement is easy. Without replacement? What does sampling without replacement mean?

If $k = 0$ do nothing. Else sample one item in proportion to weight, remove from set and recurse with $k - 1$.

Weighted Sampling: k samples

With replacement is easy. Without replacement? What does sampling without replacement mean?

If $k = 0$ do nothing. Else sample one item in proportion to weight, remove from set and recurse with $k - 1$.

How to implement above in streaming without knowing full sequence in advance?

Weighted Sampling: k samples

Offline algorithm.

WEIGHTED-SAMPLE-WITHOUT-REPLACEMENT(k):

For $i = 1$ to m do
    $\theta_i \leftarrow$ uniform random number in interval $(0, 1)$
    $w'_i = \theta_i^{1/w_i}$
endFor
Sort items in decreasing order according to w'_i values
Output the first k items from the sorted order

Weighted Sampling: k samples

Offline algorithm.

WEIGHTED-SAMPLE-WITHOUT-REPLACEMENT(k):

For $i = 1$ to m do
  $\theta_i \leftarrow$ uniform random number in interval $(0, 1)$
  $w'_i = \theta_i^{1/w_i}$
endFor
Sort items in decreasing order according to w'_i values
Output the first k items from the sorted order

Exercise: describe a streaming implementation with $O(k)$ space.

Analysis

Lemma

For $1 \leq j \leq m$ *let* $X_j = \theta_j^{1/w_j}$. *Then* $\Pr[X_i = \max_j X_j] = w_i/W$.

Analysis

Lemma

For $1 \leq j \leq m$ *let* $X_j = \theta_j^{1/w_j}$. *Then* $\Pr[X_i = \max_j X_j] = w_i/W$.

Assuming lemma: picking top k values amongst $X_1, \ldots, X_m$ is same as picking in sequence without replacement due to independence in the choice of θ_i values.

More formally

$$\Pr[X_{i'} \text{ is second largest} \mid X_i \text{ is largest}] = w_{i'}/(W - w_i)$$

Analysis

Lemma

For $1 \leq j \leq m$ *let* $X_j = \theta_j^{1/w_j}$. *Then* $\Pr[X_i = \max_j X_j] = w_i/W$.

Assuming lemma: picking top k values amongst $X_1, \ldots, X_m$ is same as picking in sequence without replacement due to independence in the choice of θ_i values.

More formally

$$\Pr[X_{i'} \text{ is second largest} \mid X_i \text{ is largest}] = w_{i'}/(W - w_i)$$

A simpler claim

Claim

Let r_1, r_2 *be independent unformly distributed random variables over* $[0,1]$ *and let* $X_1 = r_1^{1/w_1}$ *and* $X_2 = r_2^{1/w_2}$ *where* $w_1, w_2 \geq 0$*. Then*

$$\Pr[X_2 \geq X_1] = \frac{w_2}{w_1 + w_2}.$$

Suppose $X = r^{1/w}$ where $w > 0$ is fixed and r is chosen uniformly at random from $[0,1]$. What are the cumulative density function F_X and density function f_X of X? Note that $X \in [0,1]$.

A simpler claim

Claim

Let r_1, r_2 be independent unformly distributed random variables over $[0, 1]$ and let $X_1 = r_1^{1/w_1}$ and $X_2 = r_2^{1/w_2}$ where $w_1, w_2 \geq 0$. Then

$$\Pr[X_2 \geq X_1] = \frac{w_2}{w_1 + w_2}.$$

Suppose $X = r^{1/w}$ where $w > 0$ is fixed and r is chosen uniformly at random from $[0, 1]$. What are the cumulative density function F_X and density function f_X of X? Note that $X \in [0, 1]$.

$$F_X(t) = \Pr[X \leq t] = \Pr[r^{1/w} \leq t] = \Pr[r \leq t^w] = t^w.$$

A simpler claim

Claim

Let r_1, r_2 be independent unformly distributed random variables over $[0,1]$ and let $X_1 = r_1^{1/w_1}$ and $X_2 = r_2^{1/w_2}$ where $w_1, w_2 \geq 0$. Then

$$\Pr[X_2 \geq X_1] = \frac{w_2}{w_1 + w_2}.$$

Suppose $X = r^{1/w}$ where $w > 0$ is fixed and r is chosen uniformly at random from $[0,1]$. What are the cumulative density function F_X and density function f_X of X? Note that $X \in [0,1]$.

$$F_X(t) = \Pr[X \leq t] = \Pr[r^{1/w} \leq t] = \Pr[r \leq t^w] = t^w.$$

Hence $f_X(t) = \frac{d}{dt} F_X(t) = w t^{w-1}$.

Proof of Claim

$$
\begin{aligned}
\Pr[X_1 \le X_2] &= \int_0^1 F_{X_1}(t) f_{X_2}(t)\,dt \\
&= \int_0^1 t^{w_1} w_2 t^{w_2-1}\,dt \\
&= \frac{w_2}{w_1 + w_2}.
\end{aligned}
$$

Proof of Lemma

$$\begin{aligned}
\mathbf{Pr}[X_i \text{ is max}] &= \int_0^1 \left(\prod_{j \neq i} F_{X_j}(t) \right) f_{X_i}(t) dt \\
&= \int_0^1 t^{W - w_i} w_i t^{w_i - 1} dt \\
&= \int_0^1 t^{W-1} w_i dt \\
&= \frac{w_i}{W}.
\end{aligned}$$

Part III

Mean and Median via Sampling

Mean and Median

Suppose we have a list of n numbers $a_1, a_2, \ldots, a_n$

- Mean: average value $= \sum_{i=1}^{n} a_i / n$
- Median: middle number after sorting

Mean and Median

Suppose we have a list of n numbers $a_1, a_2, \ldots, a_n$

- Mean: average value $= \sum_{i=1}^{n} a_i / n$
- Median: middle number after sorting

Two important statistics about numerical data. Can be computed in $O(n)$ time. Mean is trivial. Median is not so obvious.

Mean and Median

Suppose we have a list of n numbers $a_1, a_2, \ldots, a_n$

- Mean: average value $= \sum_{i=1}^{n} a_i / n$
- Median: middle number after sorting

Two important statistics about numerical data. Can be computed in $O(n)$ time. Mean is trivial. Median is not so obvious.

Can we compute them in streaming setting? How do we estimate if data is not easily accessible or very large?

Median estimation via Sampling

- Sample k elements from $a_1, a_2, \ldots, a_n$. Let S be sample.
- Compute median of S and output it

Median estimation via Sampling

- Sample k elements from $a_1, a_2, \ldots, a_n$. Let S be sample.
- Compute median of S and output it

Will see soon proof of the following.

Theorem

If $k = \Omega(\frac{1}{\epsilon^2} \log \frac{1}{\delta})$ algorithm outputs an ϵ-approximate median with probability at least $(1 - \delta)$.

Mean estimation via Sampling

Assume $a_1, \ldots, a_n > 0$

- Sample k elements from $a_1, a_2, \ldots, a_n$. Let S be sample.
- Compute mean of S and output it

Question: Can uniform sampling give a good estimate?

Mean estimation via Sampling

Assume $a_1, \ldots, a_n > 0$

- Sample k elements from $a_1, a_2, \ldots, a_n$. Let S be sample.
- Compute mean of S and output it

Question: Can uniform sampling give a good estimate?
Mean is sensitive to outliers. How do we overcome this?

- Show that estimation works when there are no outliers
- Use *importance* sampling if/when possible