Gems of TCS

Streaming Algorithms

Sasha Golovnev

September 6, 2023

Fruit Game

Credit: Jelani Nelson

(https://www.youtube.com/watch?v=CorP4I23wOo&t=2434s)

Streaming Algorithms

- Massively long stream of data

Streaming Algorithms

- Massively long stream of data
  Instagram, search queries, network packets

Streaming Algorithms

- Massively long stream of data
  Instagram, search queries, network packets
  $x_1, x_2, x_3, \ldots, x_n$

Streaming Algorithms

- Massively long stream of data
  Instagram, search queries, network packets
  $x_1, x_2, x_3, \ldots, x_n$
- Data has grown: we can't afford even storing it

Streaming Algorithms

- Massively long stream of data
  Instagram, search queries, network packets
  $x_1, x_2, x_3, \ldots, x_n$
- Data has grown: we can't afford even storing it
- n inputs, space $\sqrt{n}$; $\log^{10} n$; $\log n$

Streaming Algorithms

- Massively long stream of data
  Instagram, search queries, network packets
  $x_1, x_2, x_3, \ldots, x_n$
- Data has grown: we can't afford even storing it
- n inputs, space $\sqrt{n}$; $\log^{10} n$; $\log n$
- Efficient processing of stream

Streaming Algorithms

- Massively long stream of data
  Instagram, search queries, network packets
  $x_1, x_2, x_3, \ldots, x_n$
- Data has grown: we can't afford even storing it
- n inputs, space $\sqrt{n}$; $\log^{10} n$; $\log n$
- Efficient processing of stream
- Mostly randomized algorithms

Missing Number

Missing Number

- Stream contains n distinct numbers in range $\{0, \ldots, n\}$

Missing Number

- Stream contains n distinct numbers in range $\{0, \ldots, n\}$
- Return the only missing number

Missing Number

- Stream contains n distinct numbers in range $\{0, \ldots, n\}$
- Return the only missing number
- Efficient algorithm?

Streaming Algorithm

- Compute sum of all elements in stream:

$$s = x_1 + \ldots x_n$$

Streaming Algorithm

- Compute sum of all elements in stream:

$$s = x_1 + \dots x_n$$

- Sum of all numbers in range $\{0, \dots, n\}$ is $S = \frac{n(n+1)}{2}$

Streaming Algorithm

- Compute sum of all elements in stream:

$$s = x_1 + \dots x_n$$

- Sum of all numbers in range $\{0, \dots, n\}$ is $S = \frac{n(n+1)}{2}$
- Missing number is $S - s = \frac{n(n+1)}{2} - s$

Streaming Algorithm

- Compute sum of all elements in stream:

$$s = x_1 + \ldots x_n$$

- Sum of all numbers in range $\{0, \ldots, n\}$ is $S = \frac{n(n+1)}{2}$
- Missing number is $S - s = \frac{n(n+1)}{2} - s$
- One pass through stream, efficient processing, $O(\log n)$ space

Two Missing Elements

- Stream contains $n - 1$ distinct numbers in range $\{0, \ldots, n\}$

Two Missing Elements

- Stream contains $n - 1$ distinct numbers in range $\{0, \ldots, n\}$
- Return both missing numbers

Two Missing Elements

- Stream contains $n - 1$ distinct numbers in range $\{0, \ldots, n\}$
- Return both missing numbers
- Efficient algorithm?

Streaming Algorithm

- Compute sum and sum of squares of all elements in stream:

$$s = x_1 + \ldots x_{n-1}$$
$$t = x_1^2 + \ldots x_{n-1}^2$$

Streaming Algorithm

- Compute sum and sum of squares of all elements in stream:

$$s = x_1 + \dots x_{n-1}$$
$$t = x_1^2 + \dots x_{n-1}^2$$

- Sum of all numbers in range $\{0, \dots, n\}$ is $S = \frac{n(n+1)}{2}$
  Sum of squares of all numbers in range $\{0, \dots, n\}$ is $T = \frac{n(n+1)(2n+1)}{6}$

Streaming Algorithm

- If missing numbers are a and b, then

$$\begin{aligned} a + b &= S - s \\ a^2 + b^2 &= T - t \end{aligned}$$

Streaming Algorithm

- If missing numbers are a and b, then

$$
\begin{aligned}
a + b &= S - s \\
a^2 + b^2 &= T - t
\end{aligned}
$$

- One pass through stream, efficient processing, $O(\log n)$ space

Majority Element

Majority Element

- Stream has element occuring $> n/2$ times

Majority Element

- Stream has element occuring $> n/2$ times
- Find it!

Streaming Algorithm

- count $\leftarrow 0$;   m $\leftarrow \perp$

Streaming Algorithm

- count $\leftarrow 0$; m $\leftarrow \perp$
- For each element x_i of Stream:

Streaming Algorithm

- $\mathsf{count} \leftarrow 0; \quad \mathsf{m} \leftarrow \perp$
- For each element x_i of Stream:
  - If $\mathsf{count} = 0$, then $\mathsf{m} \leftarrow x_i$ and $\mathsf{count} \leftarrow 1$

Streaming Algorithm

- $\mathsf{count} \leftarrow 0$; $\mathsf{m} \leftarrow \perp$
- For each element x_i of Stream:
  - If $\mathsf{count} = 0$, then $\mathsf{m} \leftarrow x_i$ and $\mathsf{count} \leftarrow 1$
  - ElseIf $x_i = \mathsf{m}$, then count ++

Streaming Algorithm

- $\mathsf{count} \leftarrow 0; \quad \mathsf{m} \leftarrow \perp$
- For each element x_i of Stream:
    - If $\mathsf{count} = 0$, then $\mathsf{m} \leftarrow x_i$ and $\mathsf{count} \leftarrow 1$
    - ElseIf $x_i = \mathsf{m}$, then count ++
    - Else count --

Streaming Algorithm

- $\mathsf{count} \leftarrow 0$; $\mathsf{m} \leftarrow \perp$
- For each element x_i of Stream:
  - If $\mathsf{count} = 0$, then $\mathsf{m} \leftarrow x_i$ and $\mathsf{count} \leftarrow 1$
  - ElseIf $x_i = \mathsf{m}$, then count++
  - Else count--
- Return m

Example

Proof

Another View

Misra-Gries Algorithm

Misra-Gries Algorithm

- $\mathsf{count}_1, \ldots, \mathsf{count}_k \leftarrow 0; \quad \mathsf{m}_1, \ldots, \mathsf{m}_k \leftarrow \perp$
- For each element x_i of Stream:
  - If $x_i = \mathsf{m}_j$, then count_j ++
  - Else
    - Let count_j be min in $\mathsf{count}_1, \ldots \mathsf{count}_k$
    - If $\mathsf{count}_j = 0$, then $\mathsf{m}_j = x_i; \quad \mathsf{count}_j = 1$
    - Else count_1 --$, \ldots, \mathsf{count}_k$ --
- Return $\mathsf{m}_1, \ldots, \mathsf{m}_k$

Approximate Counting

- Router receives stream of network packages

- Router receives stream of network packages
- Want to count number of packages from IP "1.2.3.4"

- Router receives stream of network packages
- Want to count number of packages from IP "1.2.3.4"
- Efficient algorithm?

- Router receives stream of network packages
- Want to count number of packages from IP "1.2.3.4"
- Efficient algorithm?
- Efficient approximate algorithm?

Overview

Morris Algorithm

Morris Algorithm

- $c \leftarrow 0$

Morris Algorithm

- $c \leftarrow 0$
- When see next element:
    - with probability $\frac{1}{2^c}$ increment c
    - with probability $1 - \frac{1}{2^c}$ do nothing

Morris Algorithm

- $c \leftarrow 0$
- When see next element:
    - with probability $\frac{1}{2^c}$ increment c
    - with probability $1 - \frac{1}{2^c}$ do nothing
- Return $2^c - 1$

ANALYSIS

Probability of Success

- $\mathbb{E}[\text{output}] = n$

Probability of Success

- $\mathbb{E}[\text{output}] = n$
- By Markov's, $\Pr[\text{output} \geq 2n] \leq 1/2$

Probability of Success

- $\mathbb{E}[\text{output}] = n$
- By Markov's, $\Pr[\text{output} \geq 2n] \leq 1/2$
- Similar inequalities show that $\Pr[\text{output} \in [n - O(n), n + O(n)]] \geq 0.9$

Probability of Success

- $\mathbb{E}[\text{output}] = n$
- By Markov's, $\Pr[\text{output} \geq 2n] \leq 1/2$
- Similar inequalities show that $\Pr[\text{output} \in [n - O(n), n + O(n)] \geq 0.9$
- Again, repeating Algorithm several times significantly amplifies probability of success

Summary

- One pass through stream may be sufficient

Summary

- One pass through stream may be sufficient
- Use Randomness and Approximation

Summary

- One pass through stream may be sufficient
- Use Randomness and Approximation
- Markov's ineqaulity: from Expectation to Probability

Summary

- One pass through stream may be sufficient
- Use Randomness and Approximation
- Markov's ineqaulity: from Expectation to Probability
- Amplify probability by Repetitions