Streaming Algorithms

Meng-Tsung Tsai

02/23/2018

Course goals

- Getting familiar with how to process huge amounts of data using small working space, including:

  (a) algorithm design, and (b) implementation.

- Understanding the limit of algorithms that use small working space.

Course materials

- Slides on e3new.nctu.edu.tw
- Online documents and reference books.
- There is no main textbook.

Office hours

- 14:20 - 15:10 on Thursdays at EC 336, or by appointment.
- TA hours: TBA.

Grading policy

- Grade in percentages, 0 - 100.
- Your final grade is based on the 3 written assignments (30%), 3 programming assignments (30%), and 1 final project (40%).
- In the written assignments, writing down "I don't know" for a problem gives you 25% credits.
- Collaboration is encouraged, but the writeups/programs should be your own. The consequence of cheating or plagarism can be very serious, including failing this course.

Make-up work

If you fail this course at the end of this semester, don't feel hesitate to contact me. You may have make-up work if you participate in class regularly.

Streaming Model

Streaming Model (1/3)

Input: A stream (sequence) S of data is given one by one to algorithms.

Requirement: Algorithms are allowed to use o(|S|) space.

Output: Write the solution to a write-only stream.

O(poly log |S|) = O($\log^k$ |S|) for some constant k.

In the literature, streaming algorithms are defined to be those algorithms that use O(poly log |S|) space. However, this requirement may be relaxed for some domains.

Streaming Model (2/3)

while (cin $\gg$ x){ // assume that data are integers

Some codes that use memory sublinear to the input size, in which you may cout $\ll$ something;

}

cout $\ll$ something;

Programs are not allowed to read the input in the reverse direction by ungetc() or something equivalent.

Streaming Model (3/3)

Algorithms may scan the input from the beginning to the end multiple times, say p times, which are called p-pass algorithms.

Example 1 - Min

Input: A stream (sequence) S of integers is given one by one to algorithms.

Requirement: Algorithms are allowed to use $o(|S|)$ space.

Output: Write $\min_{x \in S} x$ to a write-only stream.

Yes, $O(1) = o(|S|)$ space suffice.

How to find the k-th smallest integers in S where $k \ll |S|$?

Exercise 1

Design an algorithm that finds the k-th smallest value of the given n integers using $O(k)$ space.

Example 2 - 2Sum

Input: A stream (sequence) S of integers is given one by one to algorithms.

Requirement: Algorithms are allowed to use $o(|S|)$ space.

Output: Write "Yes" to a write-only stream if there exist a, b in S so that $a+b = 0$; otherwise, write "No".

Claim. Any p-pass streaming algorithm that solves 2Sum requires $\Omega(|S|/p)$ space.

Exercise 2

Design an algorithm that solves 2Sum optimally.

Example 3 - approximate shortest paths

Input: A sequence of edges in an undirected graph G is given one by one to algorithms where each edge has a non-negative weight.

Requirement: Algorithms are allowed to use $O(n^{1.5})$ space where n denotes the number of nodes in G.

Output: Write a subgraph H of G to a write-only stream so that

$$d_H(u, v) \leq 3\, d_G(u, v) \text{ for all } u, v \text{ in } G \text{ and } |E(H)| \leq n^{1.5}$$

where $d_G(u, v)$ denotes the distance between u and v in G.

Algorithms that use O(n polylog n) space for graph problems are called semi-streaming algorithms.

Probablistic Method (Warm-up)

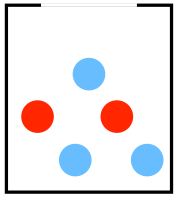
What is probablistic method?

Use probablistic statements to show some event must happen.

What is probablistic method?

Use probablistic statements to show some event must happen.

Example.



Given a box of balls, draw a ball from the box uniformly at random.

The sampled ball has blue color with probability 1/2, and has red color with probability 1/3. What can you infer?