Randomized Algorithms 2022-3
Lecture 2

A better exponential time algorithm for satisfiabiltiy, streaming and the prisoner's puzzle[*]

Moni Naor

Watch the remaining parts of Ryan O'Donell Lecture 5 on concentration bounds.

1. `https://www.youtube.com/watch?v=cLczU5-CW70`
2. `https://www.youtube.com/watch?v=zz4C-xECIp4`

1 Sat Algorithms

The main algorithms we saw were for 2-SAT and 3-SAT. The second one is due to Uwing Schöning. You can find a description in Chapter 7 of Mitzenmacher and Upfal. The complexities of the algorithms are roughly $O(n^2)$ and $O((4/3)^n)$ respectively.

A famous conjecture, called the Exponential Time Hypothesis (ETH), states that 3-SAT cannot be solved in sub-exponential time, i.e., in times less than $(1+\alpha)^n$ for some $\alpha > 0$.

2 Streaming Algorithms

Suppose you want to compute a function on a stream of data but do not have enough memory to store it. We will consider single pass algorithm, that is once the data has passed there is no further access to it. We would like as little extra storage as possible.

Several issues come up: Which functions are computable? At what accuracy can they be computed? There is a rich literature one the subject with many interesting algorithms and lower bounds.

For most tasks, if they are doable at all, then randomness is essential.

[*]These notes summarize the material covered in class, usually skipping proofs, details, examples and so forth, and possibly adding some remarks, or pointers. In the interest of brevity, most references and credits were omitted.

2.1 Multiset equality

The problem we addressed can be viewed as a 'streaming' one. We have two multi-sets A and B and they are given in an arbitrary order. Once an element is given it cannot be accessed again (unless it is explicitly stored) and our goal is to have a low memory algorithm. We required a family of functions H that was incremental in nature, in the sense that for a function $h \in H$:

- Given $h, h(A)$ and an element x it is easy to compute $h(A \cup \{x\})$.
- For any two different multi-sets A and B the probability over the choice of $h \in H$ that $h(A) = h(B)$ is small.
- The description of h is short and the output of h is small.

The function we saw was based on treating the set A as defining a polynomial $P_A(x) = \Pi_{a \in A}(x - a)$ over a finite field whose size is larger than the universe from which the elements of A are chosen (say a prime $Q > |U|$). The member of the family of functions is called h_r for $x \in GF[Q]$ and defined as $h_r(A) = P_A(r)$. The probability that two sets collide (i.e. $h_r(A) = h_r(B)$, which in turn means that $P_A(r) = P_B(r)$) is $\max\{|A|, |B|\}/Q$, since this is the maximum number of points that two polynomials whose degree is at most $\max\{|A|, |B|\}$ can agree without being identical.

Storing h_x and storing $h(A)$ as it is computed requires just $O(\log Q)$ bits, so the resulting algorithm never needs to store anything close size to the original sets.

We saw a suggestion by one of the students for such a function that was based on min-count sketch. It is not the most efficient, since the he storage. was inverse in the probability of error (this can be amplified)

Here is another suggestion made a few years ago by a student: The Primes proposal: let $f : N \mapsto N$ be an ordering of the primes, i.e. $f(i)$ returns the ith prime. Now an alternative to the definition of a polynomial P_A define an integer $N_A = \Pi_{a \in A} f(a)$. Claim: for all mutli-sets A and B, if $A \neq B$, then $N_A \neq N_B$. Of course one cannot hope to store N_A explicitly. Instead, just as evaluating $P_A(x)$ at point y can be done on-the-fly, it is possible to compute $N_A \bmod Q$. The hash family now is h_Q where Q is a random prime chosen from a certain size.

Question: Analyze the Primes method. Suggest the appropriate domain from which to chose Q.

Reading and Watching assignment:

- Watch the lecture by David Woodruff on "Adversarially Robust Streaming Algorithms" `https://www.youtube.com/watch?v=9qP3JCWNgnc`
- Tim Roughgarden's Notes on streaming and communication complexity
  `http://timroughgarden.org/w15/l/l1.pdf`

3 Odds and Ends

Pseudo-deterministic Algorithms: An algorithm is called Pseudo-deterministic if it gives the same output with high probability (see Gat and Goldwasser [3]). For a decision problem it simply means that it decides consistently whp. But what about search problems, e.g. if the output is a larger object such as a distributed set. If consistency of the algorithm (e.g. for debugging purposes) is important then we want the same object to be produced with high probability.

An open problem is *pseudo-deterministic algorithms* for min-cut. That is, we want an algorithm that outputs the one particular global min with reasonably high probability. Can you have such an algorithm and at what cost. The question is whether it is possible to obtain pseudo-deterministic algorithms that are as efficient as the best algorithms for the min-cut problem. What we said was that you can find all min cuts and output the lexicographically smallest, but this is not particularly efficient.

Sunflowers: We mentioned the *Sunflower Lemma* of Erdöos and Rado that says that in a large enough set system $\mathcal{F}$ there are r subsets that form a "sunflower". I.e. the r subsets have a common core (could be empty) that is their intersection and there are no other common elements between any pair of subsets in the sunflower. I.e. $S_1, S_2, \ldots S_r$ form a sunflower if for all $1 \le i < j \le r$ we have $S_i \cap S_j = S_1 \cap S_2 \cap \cdots \cap S_r$. The smallest interesting case is when $r = 3$, since for $r = 2$ every pair of subsets forms a sunflower.

The original bound of Erdös and Rado was that if the sets in $\mathcal{F}$ are of size at most w and there are at least $w!(r-1)^w$ sets in F, then there is a sunflower of size r. While the $(r-1)^w$ is necessary (there are examples of this size without an r-sunflower), it is not clear that the $w!$ factor is needed.

In a recent breakthrough [1] this was improved to about $(\log w)^w$. We saw a nugget from the proof:

Definition 1. *(Satisfying set system). Let $0 < \alpha, \beta < 1$. Let $\mathcal{F}$ be a set system on a ground set X. Suppose that we choose a subset Y of X so that each element of X is chosen to be in Y with probability α independently of the other elements. We say that the set system $\mathcal{F}$ is (α, β)-satisfying if*

$$\Pr[\exists S \in \mathcal{F} s.t.\ S \subset Y] > 1 - \beta].$$

What we showed is that if F is a $(1/3, 1/3)$-satisfying set system, then F contains 3 pairwise disjoint subsets. The idea was to color each element of X with one of three colors with equal probability. For each color the probability of having a monochromatic subset in $\mathcal{F}$ is exactly the same probability as when selecting elements with probability 1/3 of completely selecting one of the subsets. And from the definition of $(1/3, 1/3)$-satisfying set system it is larger than 2/3. Therefore the probability that all three events occur simultaneously is large than 0[1] . Which must mean that there are three disjoint subsets, since two monochromatic sets colored in different colors sets cannot intersect.

The 100 Prisoners Puzzle: we discussed the 100 Prisoners Puzzle due to Peter Bro Miltersen. Here is a paper on it [2]. There is an unresolved issue regarding a malicious warden who knows the

[1] This is an application of the **union bound**: we have three bad events (not being monochromatic) each with probability less than 1/3 and therefore the probability that any of the bad events happens is strictly less than 1

method and sets a permutation of the slips so as to make the participants fail.

1. Is it true that if the participants have no secret information then the warden wins with high probability?
2. If the participants have a secret random permutation then they get the same probability of success as in the original puzzle. But is there a small family of permutations that is easy to store and evaluate where this is true?

Recently a popular video about it was produced: `https://www.youtube.com/watch?v=iSNsgj1OCLA`

References

[1] Ryan Alweiss, Schachar Lovett, Kewen Wu Jiapeng Zhang, Improved bounds for the sunflower lemma, STOC 2020.

[2] Eugene Curtin and Max Warshauer, The Locker Puzzle, Mathematical Inteligenicer. `https://www.cl.cam.ac.uk/~gw104/Locker_Puzzle.pdf`

[3] Gat and Goldwasser, Probabilistic Search Algorithms with Unique Answers and Their Cryptographic Applications ECCC R11-136 `https://eccc.weizmann.ac.il/report/2011/136/`

[4] Jon Kleinberg and Eva Tardos, **Algorithm Design**. Addison Wesley, 2006. The relevant chapter 13.

[5] Dick Lipton's blog, "Rabin Flips a Coin", March 2009 `https://rjlipton.wordpress.com/2009/03/01/rabin-flips-a-coin/`

[6] Michael Oser Rabin, *Probabilistic algorithms.* In Algorithms and complexity: New Directions and Recent Results, pages 21 - 39. Academic Press, New York.

[7] Rene Schoof, *Four primality testing algorithms*, `http://arxiv.org/abs/0801.3840`