Randomized Algorithms 2018/9A
Lecture 5

Maximal Independent Set, Analysis of Randomized Quicksort, Extractors and Probabilistic Constructions [*]

Moni Naor

1 Maximal Independent Set in Distributed Graphs

We considered a distributed model of computing where nodes in a graph contain processors and they can talk to their immediate neighbors in a synchronous manner. The goal now is to collectively compute some labeling of the graph (e.g. coloring, maximal independent set or maximal matching). In particular for the maximal independent set problem the goal is to a pick a set of nodes S such that S is a maximal independent set of the underlying graph and each node determines whether or not it is in S. In this model we may need randomness simply to break symmetry (think of a cycle), but even if processors have unique id's (so the greedy algorithm can be executed) no fast local deterministic algorithm is known (one that take polylog in n time, regardless of the graph).

The algorithm we discussed lets each node choose a random number in $[0, 1]$ and only local minima are kept and enter the MIS; their neighbors are removed and the process repeats. It is a variant of a 1986 algorithm by Mike Luby [3], see the lecture notes by Wattenhaffer [5]. The key point in the analysis was to consider the number of edges removed, rather than the number of nodes. The expected number of edges removed in each phase is at least $1/3$ and the expected number of rounds until all nodes know whether they are in the MIS or not is $O(\log n)$.

A question I raised in class is whether there are *Pseudo-deterministic* algorithm for MIS. An algorithm is called Pseudo-deterministic if it gives the same output with high probability (see Gat and Goldwasser [1]). 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. However, the problem of coming up with a pseudo-deterministic algorithm for MIS is rather hard in the following sense:

Claim 1. *If there is a Pseudo-deterministic algorithm for MIS then there is also a deterministic one that operates in the same amount of time (but perhaps with longer messages).*

Proof. Suppose that we have a pseudo-deterministic algorithm that takes t steps and finds whp

[*]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.

(for our purposes it just has to be higher than 1/2 a specific set S that is an MIS. I.e. an execution of the algorithm produces S with probability larger than 1/2. The decision of whether a node is in the MIS or not is a function of the id's of the nodes of distance t and their choices of random coins. For a node v, given the id's at distance t rom v it is possible simulate what will happen in the algorithm and hence it is possible to compute the probability that node v will enter the set. This is not necessarily efficiently computable but in this model we ignore computation costs at the nodes. In the deterministic algorithm node v decides to enter the MIS iff the probability of v ending in the MIS is larger than 1/2. This defines an MIS: it cannot be the case that the probability of two adjacent nodes entering the be larger than 1/2. On the other hand, we are promised that there is an MIS that is output with high probability, so all the nodes of that set decide to enter. □

2 Probabilistic Constructions

An important reason to use randomization is to argue about the existence of combinatorial objects of certain type. One example we saw was a tournament withe Schutte property: a tournament is a directed graph where for every pair of nodes x and (Y exactly one of the edges (x, y) and (y, x) exists. The k-schutte property says that for any set of k nodes there exists a node x such that x has directed edges to all the nodes in S (think of domination). Can we construct for every k and large enough n such a tournament?

Consider a tournament on n nodes chosen at random, that is for every pair of nodes we choose at random whether to have the orientation (x, y) or (y, x). What is the probability that it has the desired property? For every set S of size k (there are $\binom{n}{k}$ such sets), for every node x outside the set (there are $n-k$ such nodes) the probability that x does not dominate S is $1-1/2^k$. The probability that no node dominates S is $(1-1/2^k)^{n-k}$. We now appeal to the *union bound*: the probability of a bad event that is the union of several bad ones is bounded by the sum of the probabilities of those events. In our case the bad event was that set S has no dominator. If set S has no dominator, then a relatively close set (with one element exchanged say) is also likely not to have a dominator, so we have no independence. But to appeal to the union bound we do not need independence. So we need to have

$$\binom{n}{k}(1-1/2^k)^{n-k} < 1$$

to argue that such a tournament exists. This gives k

A known explicit construction for such tournaments is the **Paley tournament**. Let $P = 3 \bmod 4$ be a prime. A quadratic residue mod$P$ is an integer w such that there exist integer z s.t. $z^2 = w \bmod P$. Then we know that -1 is not a quadratic residue mod$P$ and this means that for every w we have that w is a quadratic residue iff $-w$ is not a quadratic residue. Now the tournament is defined by have and edge from x to y iff $x - y$ is a quadratic residue mod$P$. From the condition on P this is indeed a tournament, i.e. (x, y) is an edge iff (y, x) is not an edge. The fact that it satisfies the property follows from properties related to a construction of small biased probability spaces that we may consider later in the course.

3 Nuts and Bolts and QuickSort

The nuts and bolts puzzle asks for finding a matching between nut and bolts where you can only compare nuts to bolts but not nuts to nuts or bolts to bolts. Can you think of a probabilistic algorithm for it, with as few comparisons as possible?

Question: why is $n \log n$ a lower bound on randomized algorithms for the nuts and bolts question?

We saw a slick argument showing that the *expected* time is $2n \ln n + O(n)$. The argument was based on analyzing the probability that elements i and j are compared (this probability is $2/(j-i+1)$ and then by the *linearity of expectation* it is a Harmonic sum. Note that this is the true complexity (in terms of comparisons) and not merely an upper bound.

For both QuickSort and the MIS algorithm above in order to get a high probability result we need a Chernoff type concentration bound. In this case when we say 'high probability' we mean that the desired run time ($O(n \log n)$ and $O(\log n)$ respectively) is achieved with probability at least $1 - 1/n^c$.

The are several results called 'Chrenoff'. These bounds talk about the probability that a random variable which is a sum of small independent random variables is far away from its expectation. Here is the one we used:

Theorem 2. *Let $X_1, X_2, \ldots X_n$ be independent Poisson trials such that* $\Pr[X_i = 1] = p_i$ *and* $\Pr[X_i = 0] = 1 - p_i$. *Let $X = \sum_{i=1}^{n} X_i$ and $\mu = E[X]$. For $0 < \delta < 1$ we have*

$$\Pr[|X - \mu| \geq \delta\mu] \leq 2e^{-\mu\delta^2/3}.$$

See Chapters 4.2 and 4.3 of Mitzenmacher and Upfal.

We also used a *union bound*: the probability of a bad event that is the union of several bad ones is bounded by the sum of those events. In case of QuickSort the bad event was that the algorithms takes too long and smaller events are that element i took a long time. These smaller events are not independent (in fact they are *positively correlated* for close i and j) but the union bound does not need that. The nice thing about it is that it is always applicable. The downside is that we may not really need such small probability for each small event and thus get a pessimistic bound.

4 Extractors

An extractor is a function that 'purifies' a not-so-random source: it takes a string that was chosen from a distribution that is far from the uniform and produces one that is (close to) uniform. There are several types of extractors, depending if there is an auxiliary randomness (the seed) and whether the seed is part of the output or not (weak or strong extractor).

We saw one such extractor, for the case of a biased coin, whose bias is unknown. The von Neumann 'trick' of turning a biased coin into a fair one is from the period where randomized algorithms started [4]. A famous quote from that paper is "*Any one who considers arithmetical methods for producing random digits is of course in a state of sin*" and then goes on to discuss such methods (what we would call today pseudorandom generators).

For a general source we argued that it is impossible to construct a deterministic (no seed) extractor, even if the entropy is high. In general the **min-entropy** of a source X, which is

$$H_\infty(X) = -\log \max_x \Pr(x)$$

is a bound on the amount of randomness we can extract from a source[1].

References

[1] Eran Gat and Shafi Goldwasser, *Probabilistic Search Algorithms with Unique Answers and Their Cryptographic Applications*, Electronic Colloquium on Computational Complexity (ECCC) 18: 136 (2011), `http://eccc.hpi-web.de/report/2011/136/`

[2] Janos Komlos, Yuan Ma and Endre Szemerredi, *Matching nuts and bolts in $O(n \log n)$ time*, SODA 1996.

[3] Michael Luby, *A Simple Parallel Algorithm for the Maximal Independent Set Problem*, SIAM Journal on Computing 15 (4): 10361053, 1986.

[4] John von Neumann, *Various techniques used in connection with random digits*, National Bureau of Standards Applied Math Series 12: 36-38, 1951. See `https://dornsifecms.usc.edu/assets/sites/520/docs/VonNeumann-ams12p36-38.pdf`

[5] Roger Wattenhofer, `http://dcg.ethz.ch/lectures/fs10/podc/lecture/mis.pdf`

[1]The other notion of entropy we mentioned is Shannon Entropy which is $\sum_x - \Pr(x) \log \Pr(x)$. It is always the case that the Shannon entropy is at least as large than the min-entropy.