CSC2420: Lecture 12

- Announcements:
  - Will post L11 and L12 later today or tomorrow
  - Will post some of last assignment today or tomorrow
- Todays agenda :
- Randomized primality
- Expander graphs and rapid mixing
  - Correctness amplification while conserving randomness
- #P Counting problems and randomized estimation

Primality Testing

- I now want to briefly turn attention to one of the most influential randomized algorithms, namely a poly time randomized algorithm for primality (or perhaps better called compositeness) testing.
- History of polynomial time algorithms:
  - 1-sided error with prob[ALG says *N* composite|*N* prime] = 0; prob[ALG say *N* prime|*N* composite] <= *delta* < *1*. Can then repeat.
  - Independently shown by Solovay and Strassen, and Rabin ~ 1974
  - The Rabin test is related to an algorithm by Miller ~1976 that gives a det poly time alg assuming (the unproven) ERH
  - 0-sided error alg (expected poly time) by Goldwasser and Kilian ~1986
  - deterministic poly time alg by Agarwal, Kayal and Saxena ~ 2002

- Even though there is a deterministic alg, it is not nearly as efficient as the 1-sided error algs which are used in practice and which also spurred the interest in this topic, had a major role in various cryptographic developments (which required random primes) and more generally became the impetus for the major interest in randomized algorithms.
- While many of our other examples of randomized algorithms night be considered reasonably natural (even if the analysis might not be easy), the following algorithm requires understanding of the subject matter (a little number theory) and is not something that one can just immediately think of.

Some basic number theory

- We need some number theory results and a basic result from group theory (Lagrange's Theorem). Here is what we need:
- *(Z*_N) = {a in Z_N: gcd(a,N) = 1}* is a (commutative) group under multiplication (*mod N*).
- If *N* is prime, then for *a* not 0 *mod N, a^{N-1} = 1 (mod N)* Fermat's Little Theorem. Furthermore, if *N* is prime then (Z*_N,*) is a cyclic group; that is, there exists a generator *g* such that *{g , g^2, ...,g^{N-1}) = Z_N* which implies that *g^i* is not 1 for *1 <= i < N-1*.
- If *N* is prime, then 1 in *Z*_N* has precisely two square roots {-1,1}
- The Chinese remainder Theorem: Whenever *N_1* and *N_2* are relatively prime, then for all *v_1* and *v_2*, there exists a unique *w < N_1 * N_2* such the *w = v_1 mod N_1* and *w = v_2 mod N_2*.

Simple but not quite correct algorithm

- We need two basic computational facts:

  a^i mod N can be efficiently computed

  gcd(a,b) can be efficiently computed

- Here is a simple algorithm that would work except for an annoying set of numbers (called Carmichael numbers).

  Choose *a* in *Z_N* be uniformly at randomly

  If *gcd(a,N)* not equal 1 then output composite

  If *a^{N-1}* not equal 1, then output composite

  Else output prime.

When does simple algorithm work

- *S = {a| gcd(a,N) = 1 and a^{N-1} = 1}* is a subgroup of *Z^*_N*
- So if there exists an *a* in *Z*_N* such that *gcd(a,N) = 1* and *a^{N-1}* not = 1, then *S* is a proper subgroup and hence */S/* divides *N-1* and thus can be at most half of *N-1*. Then simple algorithm has prob < ½ of error when *N* is composite.
- The only numbers *N* that give us trouble are the Carmichael numbers *N* (false primes) for which *a^{N-1} = 1* for all *a* such that *gcd(a,N) = 1*. It was only (relatively speaking) recently proven (in 1994) that there are an infinite number of Carmichael numbers.
- The first three Carmichael numbers are 561, 1105, 1729
- Aside: Answering a question in class, a byproduct of the next algorithm gives a probabilistic means for testing if a number is a Carmichael number (problem 14.4 in Motwani and Raghavan).

Miller-Rabin 1-sided error algorithm

Let *N-1 = (2^t) u* with *u* odd % since *N* is odd, t >=1
Choose non-zero (possible certificate) *a* randomly in *Z_N*.

x_0 = 2^u % all computation is mod *N*

For *i = 1 ..t*

x_i := x_{i-1}^2

if *x_i =1* and x_{i-1 not in {-1,1} then report composite

End for

If *x_t* not = 1 then report composite *%x_t = x^{N-1}*

Else report prime

We need to show Prob[*a* certifies *N* is composite|N is composite] >=1/2. (Note: this is then what one generally needs to show a set is in RP, namely lots of certificates.)

Brief Analysis

Let *a* be a non witness (non-certificate).

Since a^{N-1} = 1, a*a^{N-2} = 1 and *a* has an inverse so that *a* in *Z*_N*

We now want to show that the non-witnesses are a proper subgroup *Z*_N* which by Lagrange gives us what we want.

Case 1: *N* is not a Carmichael number in which case we are done.

Case 2: For every *b* in *Z*_N, b^{N-1} = 1* i.e. N is Carmichael implying it cannot be a prime power and hence

*N = N_1 * N_2* with *N_1* and *N_2* relatively prime and odd

The non witnesses must include some *b*

b^{(2^i) u} = -1 mod N and hence *b^{(2^i) u} = -1 mod N_1*

By the Chinese Remainder Theorem, there exists

w = v mod N_1 and *w = 1 mod N_2* and hence

w^{(2^I u)} = -1 mod N_1 and *w^{(2^i) u} = 1 mod N_2*

This implies that *w^{2^i u}* is not in *{-1,1} mod N*.

Conserving random bits, expanders and a return to algebraic graph theory

- Lets say that $n = \log N$ and that the compositeness test alg had error prob at most ½ and uses n random bits.
- We can obviously use nk random bits of to achieve error probability at most $(1/2^k)$. Just as a proof of concept (and to start a final problem set), lets see how we could do better than ¼ for $2n$ random bits (staying within poly time). This will not need the machinery of expanders.

Using pairwise independence to amplify correctness

- For *N* prime, let *a,b* be random in Z_*N;* then the set of *r_i* = *ai+b* mod *N* are pairwise indep.
- Consider an RP set A (such as compositeness). We don't have to worry about the case that *x* is not in A, so let *x* be in A. We have a poly time alg with 1-sided error at most ½ and we let r.v. *X_i* = A(*x,r_i*) = 0 if the computation on *x* using *r_i* says no (i.e. is in error) and otherwise A(*x,r_i*) = 1.
- E[*X_i*] is at least ½ and Var[*X_i*] is at most ¼.
- Let *Y* = sum *X*_i for *i = 1 …t* for say any *t* poly in *n*.
- Output *x* not in A (ie. *x* prime) iff *Y* = 0 (i.e. say composite if any test *X_i* = A(*x,r_i*) = 1 .
- Using Chebyshev inequality, we claim:
  Prob[*Y = 0*] <= Prob[|*Y*-**E**[*Y*]| >=*t/2*] <= *1/t*
- Hence with *2n* random bits we can get prob error <= *1/t*

Lecture 11 end: expanders; spectral gap

- Expander graphs have many applications (e.g. to coding theory, random walks, correctness amplification and de-randomization) and there are various combinatorial parameterized definitions.
- Intuitively, expander graphs G = (V,E) satisfy the property that for all (not too large) subsets S of V, the neighbourhood of S is suitably larger than S.
- This is a property (whp) of random graphs and in a sense expander graphs often act as surrogates for random graphs and there has been much research on the explicit construction of small degree random graphs.
- Algebraically, expander graphs can also be characterized as graphs with suitable spectral gap and equivalently as graphs having rapid O(log n) mixing time (to equilibrium in a random walk).

A specific expander definition

- A (*N,d,c*) expander is a d-regular bipartite multi-graph G = (X,Y,E) with ||X|=|Y| = *N*/2 such that for any subset S of X,

$$|Nbhd(S)| \geq (1 + c(1 - \frac{2|S|}{N}))|S|$$

- In general, one wants a small *d* and constant *c > 0*. Most random d-regular bipartite are such expanders but often we need explicitly constructed expanders (which are known).

Expanders and spectral gap

- The relation between this type of expander and the spectral gap $d = \lambda_1 > \lambda_2$ is the following seminal result of Alon:
- If G is a (*N,d,c*) expander then

$$|\lambda_2| \leq d - \frac{c^2}{1024+2c^2}$$

- If $|\lambda_2| \leq d - \epsilon$ then G is an (*N,d,c*) expander with $c \geq \frac{2d\epsilon - \epsilon^2}{d^2}$

Random walks on expanders

- To view the transition matrix of a random walk on an expander G, we normalize the adjacency matrix to form P = A(G)/*d*. Furthermore since G is bipartite we don't have a stationary distribution so instead we consider the transition matrix Q = (I+P)/2 meaning that with probability ½ we stay in the same state. This is now an aperiodic Markov chain which has a stationary distribution. The new eigenvalues $\{\lambda_i'\}$ now satisfy:
- $$\lambda_i' = \frac{1+\lambda_i/d}{2} \quad \lambda_2 = d - \epsilon \implies \lambda_2' = 1 - \epsilon/d$$
  $$1 = \lambda_1' \geq \lambda_2' \ldots \geq \lambda_n' = 0$$
  with first eigenvalue $e_1' = \frac{1}{\sqrt{n}}(1, 1, \ldots, 1)$

Convergence to Q's stationary distribution

- Claim: Q is doubly stochastic which implies that π, the stationary distribution is the uniform distribution.
- We want to show that for the walk determined by Q with $\lambda_2' = 1 - \frac{\epsilon}{2d}$ convergence to the stationary distribution is fast in the following sense:

$$D(t) = max_j \frac{|q_j^t - \pi_j|}{\pi_j} \leq N^{1.5}(\lambda_2')^t$$

- Here q_j^t is the probability of being in state j at time t. With the second eigenvalue is bounded away from 1, this implies approximate convergence in *t* = O(*n*) steps. See Motwani and Raghavan section 6.7.2

Back to correctness amplification

- Recall that naïve repeated application of a randomized (even 2-sided error) algorithm with error (say) 1/100 using n bits can be amplified to obtain exponentially small error c^{-*k*} using *kn* bits by taking the majority of the *k* outcomes.
- Suppose we have an explicit expander with say *d* at most 8. Given that the stationary distribution on a random walk of the graph determined by Q is the uniform distribution, the idea is to do such a random walk and sample O(*k*) nodes sampling every b steps (for some appropriate constant *b*) to approximate sampling from a uniform distribution on a set of *N* nodes each representing a setting of *n* random bits.
- Starting at a random node (using *n* bits), we will only need *n* + O(*k*) to do obtain enough trials to obtain an exponentially small error *c*^{-*k*}.

Sketch of amplification correctness

- We again let A(x,r_i) = 0 (resp. 1) if the algorithm outputs 0 (resp 1) using *r_i* as the random bits.
- Suppose we have a say small degree *d* = 7 expander choose *b* so that $(\lambda_2')^b \leq 1/10$
- We run A(*x,r_i*) with *r_i* determined by the node X_{*b*i*} reached at the *t* = *b*i* th step (for *i* = 1, …, 7*k*) in a random walk (wrt Q) started at a random node X_0 (i.e. starting with the uniform distribution). We then take the majority of these 7*k* {A(*x,r_i*)} trials. The claim is that the probability of error in this majority test is at most ½^*k*.

Amplification analysis continued

- Let p_i be the probability distribution vector for X_{*b*i*} so that *p_i* = *p_0* (Q^*b*) with *p_0* being the uniform distribution. Let W (resp W') be the 0,1 diagonal matrix whose ith entry = 1 iff A(*x,r_i*) = 1 (resp. A(*x,r_i*) = 0). It follows that ||p_i W||_1 is the probability that A(*x,r_i*) = 1 and similarly ||*p_i* W'||_1 is the probability that A(*x,r_i*) = 0.
- Consider the "random strings" *r_1, …, r_*{7*k*} which determines the sequence T = (S_*1*, …,S_{*7k*}) where S_*i* = W (reps W') if A(*x,r_i*) = 1 (resp. 0) .
- Clearly the probability of any such given sequence is ||*p_0* (BS_*1*) (B S_*2*) …(BS_{7*k*})||_1 where B = Q^b

Now to exploit the initial 1/100 error

- The initial probability error yields the following for any probability distribution:
- ||*p*BW|| <= ||*p*|| (Euclidean norm)
- ||*p*BW'|| <= (1/5) ||*p*||
- Using the fact that the majority of the A(*x,r_i*) are correct, if T is a bad sequence, then the prob(T) is at most (1/5)^{7*k*} ||*p_0*||_1 which is at most sqrt{*N*} (1/5)^{7*k*} ||*p_0*||.
- There are less than 2^{*7k*} bad sequences so that the error probability for the majority test is at most 2^{-k}

#P Counting problems

- Recall that an NP set L can be defined by L = {*x*|there exists *y* : R(*x,y*) = 1} where R is a poly time verification alg and *y* is a poly length certificate. (Similarly RP sets are those where the fraction of certificates is some fraction > 0.)
- A #P counting problem #L is one can be defined as the number of certificates for an NP set L. More specifically, if say L is the set of formulas, then #SAT is the counting problem that given any formula *x* = F, outputs the number of satisfying assignments.
- Clearly if #L is poly computable then so is L. But even if L is poly time, it does not show that #L is poly time.
- For example, given a (proper) DNF formula F, we know immediately that F is satisfiable and given a bipartite graph G, we can efficently determine if G has a perfect matching. Yet both #DNF-SAT and #perfect-match are #P complete and it is strongly believed that such problems are not poly time computable.

How well can we approximate some #P complete counting problems?

- What would be almost as good as computing an exact count, would be to compute an estimate ALG(x) in the range [(1-eps) #x, (1+eps) #x] for every instance x where #x is the number of certificates for instance x. For a randomized algorithm we would want such an estimate say with probability at least delta.
- Given that we can encode eps in log(1/eps) bits we might hope for such an algorithm to have time bounded by a polynomial in n, log(1/eps), and for randomized algs, log (1/delta). But is is not hard to see that this show P = #P (or BPP = #P) so we settle for algorithms that are poly in n, (1/eps) and log(1/delta). We call this an (eps,delta) FPRAS.

Approximate counting

- Unlike the BPP = P question, it turns out that there are some problems (volume estimation of a convex body in n dimensions) for which randomization provably helps. We will restrict attention to randomized approximate counting.
- When the underlying problem L is in P, there is natural randomized approach ("basic Monte Carlo sampling") to consider. Namely, sample from the space of possible inputs and then let the fraction of good inputs in this sample be an estimate of the fraction of all inputs that are good.

The natural approach and its limitation

- Here is the abstract estimation problem: let f be a Boolean function over a universe U such that f(*u*) can be efficiently computed for any *u* in U. Assume that U can be sampled uniformly at random. We want to estimate the size of G = {*u*|f(*u*) = 1}
- Let Y_i be 1 iff f(*u_i*) = 1 where *u_i* is the *i*th sampled input. Choose M random samples and then estimate |G| by Z = |U| sum Y_i/M. Let $\rho = |G|/|U|$
- The basic Monte Carlo estimate is an FPRAS if $M \geq \frac{4}{\epsilon^2 \rho} ln \frac{2}{\delta}$ So what is wrong?

Importance sampling

- When rho is small (as it can be very #DNF-SAT when a formula might only have poly many satisfying assignments) this is clearly inefficient. Instead we need to try to define a skewed (rather than uniform) sampling.
- Here is the idea for #DNF-SAT. We let H_*j* be the set of satisfying assignments for the *j*th clause C_*j*. Let U = {(*v,j*)|*v* satisfies C_*j*}. We let f(*v,j*) = 1 iff *v* satisfies C_*j* and *v* does not satisfy C_*i* for any $i < j$.
- Our goal then becomes one of estimating the size of G = {(*v,j*)|f(*v,j*) = 1}. Note that f is efficiently computable.
- If there are *m* clauses, then rho = |G|/|U| >= 1/*m*
- It remains to show how to sample uniformly from U; do this by uniformly choosing *j* and then a satisfying assignment in C_*j*.

Estimating number of perfect matchings in high degree (*n,n*) bipartite graphs

- We will just roughly sketch the approach to estimating the number of perfect matchings when the minimum degree is at least *n/2*. This is still a #P complete problem. (This restriction is used to guarantee that *r_k* = *m_k/m_{k-1}* is at least 1/*n^2* where *m_k* is the number of size *k* matchings.)
- Let M_*k* be the set of size *k* matchings. The estimation method consists of "nearly uniform" sampling from M_*k* and then using such sampling to recursively estimating *m_k* and hence to estimate the number of perfect matchings = *m_n*

Using uniform sampling from M_*k* to recursively estimate *m_k*

- Let *e* be an edge and *m'_k* (resp *m''_k*) be the number of matchings in M_*k* containing (resp. not containing) edge *e*. As long as there are more than *k* edges, there is an edge *e* such that *r = m'_k/m_k* is at least *1/n*.
- Clearly *m_k = m'_k + m''_k*. We estimate *m_k* by using the basic Monte Carlo sampling of M_*k* to estimate the number of perfect matching in M_*k* that do not contain edge *e*. That is, to estimate *m'_k/m_k*. We recursively estimate *m'_k* by considering the graph without edge *e*. The ratio of these two estimates is our estimate of *m_k*. We have to try each edge to get the best edge for the estimate.
- Without formally defining it, it should be clear that a nearly uniform sampling of M_*k* will suffice. But that is the main technical issue to resolve.

Sketch of near uniform sampling of M_*k*

- The approach is to be able to sample from the union of M_*k* and M_{*k-1*}. This will give estimates of *r_k = m_k/m_{k-1}* and noting that *m_1* = the number of edges in the graph, the desired estimate is the product *m_1* (*m_2/m_1*) …(*m_n/m_{n-1}*).
- The main idea is to create an aperiodic doubly stochastic transition matrix for a rapidly mixing Markov chain on the union of M_*k* and M_{*k-1*}. As in the correctness amplification, the Markov chain stays in the same state with probability 1/2.

Completing the Markov chain definition

- Here are the allowable moves from matching *m* in M_*k* + M_{*k-1*}: choose a random edge *e* =(*u,v*) in G
  - If *m* in M_*k*, and *e* in *m*, then next state *m' = m*-{*e*}
  - If *m* in M_{*k-1*} and *u,v* both unmatched then *m' = m*+{*e*}
  - If *m* in M_{*k-1*} and exactly one (say *u*) of *u,v* is matched to say *w*, then change to *m' = m*-{*(u,w)*}+{*e*}
  - Otherwise stay in current state.
- The transition matrix P then satisfies *p_ij* = 1/(2|E|) for every possible transition *i -> j*. The stationary distribution for this Markov chain is the uniform distribution on M_*k* + M_{*k-1*} and the underlying graph of this Markov chain is an expander.