Randomized Algorithms

Readings: *Randomized algorithms*, Motwani and Raghavan, page 83 in the readings. Also, Appendix C (page 90 in the readings) for the background in probability.

Correction to Theorem 7.1 page 84 in the readings: $AB = C$ should be $AB \neq C$. (This is correct in the original book, but did not copy properly.)

Randomized Quicksort

Page 59 in the readings.

As a first example of randomized algorithms, consider the quicksort sorting algorithm. The expected execution time of this algorithm is the same as the runtime of some other sorting algorithms, however this algorithm has the advantage of being very simple. (For some other problems, randomized algorithms run much faster than the best known deterministic algorithms.)

Input: Array $A[1 \ldots n]$ of unique numbers.

Problem: Sort A in increasing order.

1. Choose $y \in A$ uniformly at random.
2. By comparing each element with y, determine array A_1 of elements $< y$, and array A_2 of elements $> y$.
3. Output the recursively sorted version A_1, followed by y, followed by the recursively sorted version of A_2.

Worst case runtime is $\Theta(n^2)$.

Average runtime is $\Theta(n \log n)$.

Analysis is complicated (but very nice) - see page 59 in the readings (you are *not* responsible for it).

[1]These notes were taken originally by Antonina Kolokolova in Steven Cook's csc364 lecture, Spring 99, and they have been edited for this term.

Checking matrix multiplication (Freivald's technique)

Freivald's technique is used for the verification of identities involving matrices, polynomials, and integers. We will consider an application of this technique to the verification of matrix multiplication.

Suppose that we are given three $n \times n$ matrices, A, B, C, and we have to check whether $C = AB$. The total number of scalar multiplications to compute C is n^3, assuming we use the straightforward method for computing $c_{ik} = \sum_{j=1}^{n} a_{ij} b_{jk}$. In fact, we can use only $n^{2.376}$ multiplications (but with a complicated algorithm).

We can use a randomized algorithm based on Freivald's technique. We describe it below.

Input: $n \times n$ matrices A, B, C.

Question: Is $AB = C$?

Randomized Algorithm **Freivald:**

1. Choose uniformly at random a 0-1 matrix $R = \begin{pmatrix} r_1 \\ r_2 \\ \vdots \\ r_n \end{pmatrix}$
2. Compute $T = A(BR) - CR$.

   Note that the parentheses determine the order of computation. This way we need only $O(n^2)$ multiplications because the product BR is a column matrix.
3. Output "YES" if $T = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix}$, and "NO" otherwise.

Given inputs A, B, C, let *error* be the event that **Freivald** gives an incorrect answer. If $AB = C$, then $\Pr(\mathit{error})$ is the probability that **Freivald** outputs "NO". In this case $\Pr(\mathit{error}) = 0$. This is because if $AB = C$, then $D = AB - C$ is a zero matrix and whatever R is chosen, T is a zero matrix and this gives the correct answer "YES".

If $AB \neq C$, then $\Pr(\mathit{error})$ is the probability that **Freivald** outputs "YES". We claim that in this case $\Pr(\mathit{error}) \leq \frac{1}{2}$. We will prove this claim below (lemma 1).

It seems that the probability of an error in case $AB \neq C$ might be quite large. However, as we repeatedly execute the algorithm, the probability of an error decreases. To see this, suppose we have another algorithm built up on top of **Freivald** that executes **Freivald** k times where each time a new R is chosen independently, and outputs "NO" when at least one answer of **Freivald** was "NO". By the claim above, if $AB \neq C$, the probability for each call on **Freivald** that it will (incorrectly) return "YES" is at most $\frac{1}{2}$. These probabilities multiply because the choices of R are independent. Thus, the probability that this compound algorithm makes a mistake (i.e., outputs "YES" even though $AB \neq C$) is at most $\frac{1}{2^k}$. Repeating 20 times brings the error probability below one in a million.

Now we prove our claim. Let A, B, C be $n \times n$ matrices. Let $D = AB - C$. Recall $T = A(BR) - CR = (AB - C)R = DR$.

Lemma 1 Assume $D \neq 0$. If R is a chosen uniformly at random as described above, then $\Pr(T = 0) \leq \frac{1}{2}$.

Proof of Lemma 1. We give a different proof from the one that was given in the lecture. For the proof that was given in the lecture, see page 85 in the readings.

$T = DR$, and let $D = AB - C = (d_{ij})$. Assume some $d_{ij} \neq 0$.

$$t_i = d_{i1}r_1 + \cdots + d_{ij}r_j + \cdots + d_{in}r_n = d_{ij}r_j + y$$

Suppose that $\Pr(y = 0) = q$. Then, assuming $q \neq 0$ and $q \neq 1$,

$$\Pr(t_i = 0) = \Pr(t_i = 0|y = 0)q + \Pr(t_i = 0|y \neq 0)(1 - q) \tag{$*$}$$

In general, if E_1, E_2 are any events, and $\Pr(E_2) = q$, then

$$\Pr(E_1) = \Pr(E_1|E_2)q + \Pr(E_1|\bar{E_2})(1 - q)$$

where $\Pr(E_1|E_2) =$ "probability of E_1 given E_2" $= \dfrac{\Pr(E_1 \cap E_2)}{\Pr(E_2)}$.

Using this:

$$\begin{aligned} \Pr(t_i = 0|y = 0) = \Pr(r_j = 0) &= \frac{1}{2} \\ \Pr(t_i = 0|y \neq 0) \leq \Pr(r_j = 1) &= \frac{1}{2} \\ \text{From } (*),\ \Pr(t_i = 0) \leq \frac{1}{2}q + \frac{1}{2}(1 - q) &= \frac{1}{2} \end{aligned}$$

Thus, $\Pr(T = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix}) \leq \Pr(t_i = 0) \leq \frac{1}{2}$.

Note: In fact, q is never 0. If $q = 1$ then another, simpler argument is needed.

Exercise: Why can't $q = 0$? What is the simpler argument if $q = 1$?

Bipartite matching

Definition 1 A *bipartite graph* is a graph $G = (U, V, E)$, where U, V are finite sets of nodes, and $E \subseteq U \times V$.

Definition 2 A *perfect matching* is a set $M \subseteq E$ of edges such that every node touches exactly one edge in M. For a perfect matching to exist, it is necessary that $|U| = |V| = |M|$.

E can be thought of as a compatibility relation between U and V.

We present a randomized algorithm to check whether a given bipartite graph has a perfect matching.

Input: Bipartite graph $G = (U, V, E)$, where $U = \{u_1, \ldots, u_n\}$, $V = \{v_1, \ldots, v_n\}$.

Question: Does G have a perfect matching?

Before considering the algorithm, observe that each perfect matching M in G can be viewed as a permutation π from U to V. For example, if $(u_1, v_3) \in M$, we can view it as $\pi(1) = 3$. The algorithm below first constructs a matrix R, and then uses the algebraic property that each term of the determinant of the matrix corresponds to a permutation, thus, to a perfect matching. The determinant is not identically zero if and only if there is a permutation π (thus, a perfect matching) for which the corresponding term in a summation is non-zero.

Perfect Match:

1. For each pair (i, j), $1 \le i, j \le n$, choose uniformly and randomly $r_{ij} \in \{1, \ldots, n^2\}$, except if $(u_i, v_j) \notin E$, then set $r_{ij} = 0$.
2. Let $R = (r_{ij})$ be the resulting $n \times n$ matrix. Compute $D = \det(R)$. If $D = 0$, then output "NO". Otherwise, output "YES".

Recall the Lagrange formula for the determinant of a matrix R is

$$\det(r_{ij}) = \sum_{\pi \in S_n} \operatorname{sgn}(\pi) r_{1\pi(1)} r_{2\pi(2)} \cdots r_{n\pi(n)}$$

where the sum is over all permutations π of the set $\{1, ..., n\}$.

Given an input graph $G = (U, V, E)$, let *error* be the event that the above algorithm gives an incorrect output.

Claim 1: If G has a perfect matching, then $\Pr(\mathit{error}) \le \frac{1}{n}$. If G does not have a perfect matching, $\Pr(\mathit{error}) = 0$.

Proof of Claim 1. Every non-zero term in the expression for the determinant corresponds to a perfect matching, and conversely every perfect matching corresponds to a non-zero term in the expression for the determinant. If there is no perfect matching, then the determinant is certainly zero, so $\Pr(\mathit{error}) = 0$. On the other hand, if there is a perfect matching, then it is possible that the positive and negative terms cancel, so $\det(R) = 0$ and the algorithm makes an error. However the next Claim shows that this event is unlikely, and completes the proof of Claim 1.

Claim 2: If there is a perfect matching, then $\Pr(\det(R) = 0) \leq \frac{1}{n}$.

Proof of Claim 2. Define the "Edmonds matrix of G" as $A = (a_{ij})$, where

$$a_{ij} = \begin{cases} x_{ij} & \text{(a variable)} \quad \text{if } (u_i, v_j) \in E \\ 0 & \text{otherwise} \end{cases}$$

Then, $\det(A) = Q(x_{11}, x_{12}, \ldots, x_{nn})$ is a polynomial in n^2 variables of degree n.

Theorem 1 (Edmonds) $\det(A) \neq 0$ iff G has a perfect matching.

The proof of Edmonds' Theorem is to observe that every perfect matching corresponds to a term (i.e. a monomial) in the polynomial $\det(A)$, and conversely. Unlike the case of a numerical determinant, every two monomials are distinct, so that it is impossible for two of them to cancel.

Despite Edmonds' Theorem, our randomized algorithm is not foolproof, because it is possible that $\det(A)$ is a non-zero polynomial $Q(x_{11}, \ldots, x_{nn}) \neq 0$, but when we plug in the numbers $r_{11}, ..., r_{nn}$ for the variables $x_{11}, ..., x_{nn}$, the resulting expression $Q(r_{11}, \ldots, r_{nn})$ evaluates to 0. However the Schwartz-Zippel Theorem below says that this is not too likely.

Before stating that theorem we should consider the question: Why not compute $\det(A)$ directly? The answer is that it is not feasible to do so, since there are up to $n!$ distinct terms in this polynomial (no two of which cancel). On the other hand, R an integer matrix, and $\det(R)$ can be computed feasibly even for large n, using Gaussian elimination and exact rational arithmetic.

Theorem 2 (Schwartz-Zippel) Let $P(y_1, \ldots, y_m)$ be a polynomial in m variables of degree d, such that $P \not\equiv 0$. Let S be any finite set of numbers. Choose $r_1, \ldots, r_m$ uniformly and randomly from S. Then $\Pr(P(r_1, \ldots, r_m) = 0) \leq \frac{d}{|S|}$

The proof is by induction on m, the number of variables in the polynomial P. Here we will only prove the base case, $m = 1$. Then the polynomial $P(y)$ has only one variable.

How many 0's does $P(y)$ have? We know from high school that a polynomial $P(y)$ of degree d has at most d zeroes (where a "zero" of $P(y)$ is a number c such that $P(c) = 0$). Therefore the set S of numbers contains at most d zeroes of $P(y)$. Thus if we choose $r \in S$ uniformly at random, then $\Pr(P(r) = 0) \leq \frac{d}{|S|}$.

To see how Claim 2 follows from the Schwartz-Zippel Theorem, let $S = \{1, 2, \ldots, n^2\}$, $P(x_{11}, \ldots, x_{nn}) = \det(A)$, and $d = n$. Then, the event $\det(R) = 0$ is the same as the event $P(r_{11}, \ldots, r_{nn}) = 0$. Hence by Schwartz-Zippel, $\Pr(\det(R) = 0) \le \frac{n}{n^2} = \frac{1}{n}$.