15-859(M): Randomized Algorithms **Homework 2**
Date: Oct 4, 2004 **Due:** Oct 18, 2004

Ground Rules

The homework consists of a few exercises followed by some questions. The exercises will not be graded, and are given to help you better understand the course material. You are allowed to work in small groups, but must turn in solutions individually. Please let us know, for each question, if you worked in a group, or if you have seen the question before.

Exercises

1. **Near-MinCuts** Suppose the MinCut in a graph G is of size k. Given an integer α, modify the randomized MinCut algorithm discussed in Lecture 5 to produce a fixed cut of size αk with probability roughly $n^{-\alpha}$.

2. **Universal Hash Families.** Prove that if $p \geq m$ is a prime, then

$$\mathcal{H} = \{h_{a,b}(x) = ((ax+b) \bmod p) \bmod n \mid a, b \in \mathbb{Z}_p, a \neq 0\} \tag{1}$$

   is a 2-universal hash family from $M \to N$, with $|M| = m$ and $|N| = n$. If we remove the restriction $a \neq 0$, does the family still remain 2-universal?

3. Prove using Yao's principle that any Las Vegas algorithm for sorting n numbers must have an expected running time of $\Omega(n \log n)$.

4. ***s*-*t* MinCuts.** Consider adapting the MinCut algorithm of Lecture 5 to finding an s-t MinCut in a graph — at every stage, we have an s-supernode and a t-supernode. Of all the edges that *do not* go between the s-supernode and the t-supernode, we pick a random edge and contract it.

   (a) Show that there are graphs for which this algorithm produces an s-t MinCut with exponentially low probability.

   (b) What is the largest number of distinct s-t MinCuts that a graph can have? Give a graph that achieves this number.

5. **Pattern Matching.** In this question you will extend the communication complexity example of Lecture 2 to the following pattern matching problem. Given an n-bit string X and an m-bit pattern Y with $m \ll n$, you have to determine whether Y is a substring of X; that is, if $Y = X[j, \cdots, j+m-1]$ for some j.

   (a) In the communication complexity example from Lecture 2, Alice picks a random prime p and sends a hash $H(x) = x \pmod p$ of her string x to Bob, who compares it to the hash $H(y) = y \pmod p$ of his string y. Extend this scheme to obtain an $O(m+n)$ time algorithm for the pattern matching problem.

   (b) What is the probability that the above algorithm fails? How big should p be for this algorithm to work with probability at least $\frac{1}{2}$?

Questions

1. **Lower Bounds.** Using Yao's principle, prove that for any 2-universal hash family $\mathcal{H}$ from M to N, there exists $x \neq y \in M$ such that

$$\Pr_{h \in_R \mathcal{H}}[h(x) = h(y)] \geq \frac{1}{|N|} - \frac{1}{|M|}. \tag{2}$$

2. **Randomization and Derandomization.** Recall the randomized Max-Cut algorithm given in Lecture 1. For a vertex v in the graph, let X_v be the indicator random variable denoting whether v is in the set S or not. If X is the r.v. denoting the number of edges cut, then we showed that $E[X] = m/2$.

   (a) **(Markov)** Use Markov's inequality to get a lower bound on the probability $\Pr[X \geq (1-\epsilon)m/2]$. How many times would you repeat the algorithm to get a cut with at least $(1-\epsilon)m/2$ edges with probability $\frac{1}{2}$?

   (b) **(Pairwise Independence)** Consider a *pairwise independent marking scheme* by taking $\mathcal{H}$, a strongly 2-universal hash family from $|V| \to \{0,1\}$, picking a function $h \in \mathcal{H}$ randomly, and adding vertex v to S iff $h(v) = 1$.

      i. What is the expected size μ of the cut $(S, V \setminus S)$ using this marking scheme?

      ii. Infer that there must be a *good* $h \in \mathcal{H}$ such that the cut $(h^{-1}(0), h^{-1}(1))$ in G has at least μ edges.

      iii. How small can you make $|\mathcal{H}|$? Show that if this size is small, you can enumerate over all $h \in \mathcal{H}$ to find a cut with at least μ edges.

   (c) **(Conditional Expectations)** The fact that $E[X] = m/2$ implies that there must be at least one setting of the X_v's for which $X \geq m/2$. We can convert this idea into a *deterministic* algorithm to find such a cut. The technique is known as the *Method of Conditional Expectations.*

      i. Write down an expression for $E[X]$ in terms of $E[X \mid X_{v_1} = 0]$ and $E[X \mid X_{v_1} = 1]$. Can you compute these two expectations?

      ii. Suppose you fix X_{v_1} to be the value in $\{0,1\}$ for which the conditional expectation is at least $m/2$. Can you now make a similar argument with the r.v. X_{v_2}? Use this idea to develop an algorithm that outputs a cut of size at least $m/2$?

3. **Isolation Lemma and Perfect Matchings.**

   (a) Consider an undirected graph G. An assignment of lengths to edges in the graph determines shortest paths between all vertices. An assignment is *good* if

      - there is a unique shortest path between each pair of vertices, and
      - all pairwise distances in the graph are distinct.

      Find the smallest W you can for which there exists a good assignment of weights to edges, such that all weights are drawn from the range $\{0, \cdots, W\}$.

   (b) How can we alter the parallel perfect matching algorithm given in Lecture 3 to find to find a maximum cardinality matching (in case the graph does not have a perfect matching)?

4. **Balls and Bins.** We will analyse an occupancy problem which involves multiple rounds. We start off by throwing n balls into n bins in the first round. After the round $i \geq 1$, we remove every bin that was populated by some ball thrown in round i, and in the following round $i+1$, we throw n balls into the remaining bins. The process ends when there are no more bins left.

   Show that the expected number of rounds for which this process runs is at most $c \log^* n$.