CS 573: Graduate Algorithms, Fall 2010
Homework 4

Due Monday, November 1, 2010 at 5pm
(in the homework drop boxes in the basement of Siebel)

---

1. Consider an n-node treap T. As in the lecture notes, we identify nodes in T by the ranks of their search keys. Thus, 'node 5' means the node with the 5th smallest search key. Let i, j, k be integers such that $1 \le i \le j \le k \le n$.

   (a) What is the *exact* probability that node j is a common ancestor of node i and node k?

   (b) What is the *exact* expected length of the unique path from node i to node k in T?

2. Let $M[1..n, 1..n]$ be an $n \times n$ matrix in which every row and every column is sorted. Such an array is called *totally monotone*. No two elements of M are equal.

   (a) Describe and analyze an algorithm to solve the following problem in $O(n)$ time: Given indices i, j, i', j' as input, compute the number of elements of M smaller than $M[i, j]$ and larger than $M[i', j']$.

   (b) Describe and analyze an algorithm to solve the following problem in $O(n)$ time: Given indices i, j, i', j' as input, return an element of M chosen uniformly at random from the elements smaller than $M[i, j]$ and larger than $M[i', j']$. Assume the requested range is always non-empty.

   (c) Describe and analyze a randomized algorithm to compute the median element of M in $O(n \log n)$ expected time.

3. Suppose we are given a complete undirected graph G, in which each edge is assigned a weight chosen independently and uniformly at random from the real interval $[0, 1]$. Consider the following greedy algorithm to construct a Hamiltonian cycle in G. We start at an arbitrary vertex. While there is at least one unvisited vertex, we traverse the minimum-weight edge from the current vertex to an unvisited neighbor. After $n-1$ iterations, we have traversed a Hamiltonian path; to complete the Hamiltonian cycle, we traverse the edge from the last vertex back to the first vertex. What is the expected weight of the resulting Hamiltonian cycle? *[Hint: What is the expected weight of the first edge? Consider the case $n = 3$.]*

4. (a) Consider the following deterministic algorithm to construct a vertex cover C of a graph G.

```
VERTEXCOVER(G):
  C ← ∅
  while C is not a vertex cover
      pick an arbitrary edge uv that is not covered by C
      add either u or v to C
  return C
```

Prove that VERTEXCOVER can return a vertex cover that is $\Omega(n)$ times larger than the smallest vertex cover. You need to describe both an input graph with n vertices, for any integer n, and the sequence of edges and endpoints chosen by the algorithm.

(b) Now consider the following randomized variant of the previous algorithm.

```
RANDOMVERTEXCOVER(G):
  C ← ∅
  while C is not a vertex cover
      pick an arbitrary edge uv that is not covered by C
      with probability 1/2
          add u to C
      else
          add v to C
  return C
```

Prove that the expected size of the vertex cover returned by RANDOMVERTEXCOVER is at most $2 \cdot \text{OPT}$, where OPT is the size of the smallest vertex cover.

(c) Let G be a graph in which each vertex v has a weight $w(v)$. Now consider the following randomized algorithm that constructs a vertex cover.

```
RANDOMWEIGHTEDVERTEXCOVER(G):
  C ← ∅
  while C is not a vertex cover
      pick an arbitrary edge uv that is not covered by C
      with probability w(v)/(w(u) + w(v))
          add u to C
      else
          add v to C
  return C
```

Prove that the expected weight of the vertex cover returned by RANDOMWEIGHTEDVERTEXCOVER is at most $2 \cdot \text{OPT}$, where OPT is the weight of the minimum-weight vertex cover. A correct answer to this part automatically earns full credit for part (b).

5. (a) Suppose n balls are thrown uniformly and independently at random into m bins. For any integer k, what is the *exact* expected number of bins that contain exactly k balls?

   (b) Consider the following balls and bins experiment, where we repeatedly throw a fixed number of balls randomly into a shrinking set of bins. The experiment starts with n balls and n bins. In each round i, we throw n balls into the remaining bins, and then discard any non-empty bins; thus, only bins that are empty at the end of round i survive to round $i+1$.

```
BALLSDESTROYBINS(n):
  start with n empty bins
  while any bins remain
      throw n balls randomly into the remaining bins
      discard all bins that contain at least one ball
```

   Suppose that in every round, *precisely* the expected number of bins are empty. Prove that under these conditions, the experiment ends after $O(\log^* n)$ rounds.[1]

   ⋆(c) **[Extra credit]** Now assume that the balls are really thrown randomly into the bins in each round. Prove that with high probability, BALLSDESTROYBINS(n) ends after $O(\log^* n)$ rounds.

   (d) Now consider a variant of the previous experiment in which we discard balls instead of bins. Again, the experiment n balls and n bins. In each round i, we throw the remaining balls into n bins, and then discard any ball that lies in a bin by itself; thus, only balls that collide in round i survive to round $i+1$.

```
BINSDESTROYSINGLEBALLS(n):
  start with n balls
  while any balls remain
      throw the remaining balls randomly into n bins
      discard every ball that lies in a bin by itself
      retrieve the remaining balls from the bins
```

   Suppose that in every round, *precisely* the expected number of bins contain exactly one ball. Prove that under these conditions, the experiment ends after $O(\log\log n)$ rounds.

   ⋆(e) **[Extra credit]** Now assume that the balls are really thrown randomly into the bins in each round. Prove that with high probability, BINSDESTROYSINGLEBALLS(n) ends after $O(\log\log n)$ rounds.

[1] Recall that the iterated logarithm is defined as follows: $\log^* n = 0$ if $n \le 1$, and $\log^* n = 1 + \log^*(\lg n)$ otherwise.