Randomized Algorithms

CME309/CS365, Winter 2012-2013, Stanford University
Instructor: Ashish Goel
Handout 5: Homework 3. Due 3/1/2013 @ 5:00pm.

Collaboration policy: Limited collaboration is allowed you can discuss the problem with other students in this class but cannot collaborate on writing the actual final answer. Please do not look at someone elses solution and do not share your solution with anyone else.

Late HW policy: You are allowed one late day for any one HW (i.e. submit the HW on Friday as opposed to Thursday).

Problem numbers refer to the text-book (Motwani and Raghavan).

Non-letter grade students: please do any two problems. If you do more, we will grade any two.

1. More practice on Chernoff bounds
   (a) (Exercise 4.11 from Mitzenmacher and Upfal) Consider a collection $X_1, \ldots, X_n$ of n independent integers chosen uniformly from the set $\{0, 1, 2\}$. Let $X = \sum_{i=1}^{n} X_i$ and $0 < \delta < 1$. Derive a Chernoff bound for $\mathbf{Pr}[X \geq (1+\delta)n]$ and $\mathbf{Pr}[X \leq (1-\delta)n]$.
   (b) (Exercise 4.17 from Mitzenmacher and Upfal) Suppose that we have n jobs to distribute among m processors. For simplicity, we assume that m divides n. A job takes 1 step with probability p and $k > 1$ steps with probability $1 - p$. Use Chernoff bounds to determine upper and lower bounds (that hold with high probability) on when all jobs will be completed if we randomly assign exactly n/m jobs to each processor.
2. Given a stream of values $a_0, a_1, a_2, \ldots$, your goal is to estimate the median of the unique elements in the stream at any time t. Design an algorithm that always returns a value inside the $[\frac{1}{2} - \delta, \frac{1}{2} + \delta]$-quantile with probability at least $1 - \epsilon$. The amount of memory your algorithm uses should only depend on the accuracy desired (i.e. δ and ϵ), and not on the number of unique elements seen.
   Hint: Use consistent random number generation to generate uniform samples from the unique elements. Then apply Chernoff bounds.
3. Modify the distribution of trees returned by the Fakcharoenphol-Rao-Talwar algorithm so that the set of nodes in each tree of the support is identical to the set of nodes in the graph. The new distribution of modified trees should still satisfy $d_T(u, v) \geq d_G(u, v)$ and $\mathbb{E}[d_T(u, v)] \leq O(\log n)\mathbb{E}[d_G(u, v)]$.
   Hint: Just modify each tree in the support of the distribution returned by "pushing up" nodes to replace the intermediate ones.
4. Embeddings into L_1
   (a) Show that any tree can be embedded into a hypercube.
   Hint: Use each edge of the tree to define one dimension of the hypercube embedding.
   (b) Show that any metric space can be embedded into L_1 (*not* a distribution) such that for any two points u, v, $\|u - v\|_1 \geq d_T(u, v)$ and $\|u - v\|_1 \leq O(\log n) d_T(u, v)$.
   Hint: Do something similar to the above hypercube embedding of trees with the distribution returned from the Fakcharoenphol-Rao-Talwar algorithm.