

Problem Set 2

Randomized Algorithms

Due Thursday, September 12

1. Consider an optimization problem $\text{Opt}(x) : \mathcal{X} \rightarrow \{0, 1, \dots, n-1\}$ that associates inputs from some domain with outputs (e.g., min cut is such a problem).

Suppose that we have a BPP algorithm \mathcal{A} that takes T time to answer binary queries of whether $\text{Opt}(x)$ is less than some threshold. That is, it answers queries of the form $\text{Less}(x, k) := (\text{Opt}(x) < k)$, with error probability at most $1/4$.

We saw in class that we can then solve $\text{Opt}(x)$ in $O(T \log n \log \log n)$ time with $3/4$ probability, by amplifying the probability and doing binary search. The goal of this problem is to improve this.

- (a) Consider the function $\text{RobustBisect}(x, L, H)$ that has three outputs:

- LOW if $L \leq \text{Opt}(x) < \frac{L+H}{2}$.
- HIGH if $\frac{L+H}{2} \leq \text{Opt}(x) < H$.
- OUTOFRANGE if $\text{Opt}(x) \notin [L, H)$.

Use \mathcal{A} to construct a randomized algorithm \mathcal{B} to solve RobustBisect with $3/4$ probability and $O(T)$ time.

- (b) Now consider how to use \mathcal{B} as a subroutine in binary search, to determine $\text{Opt}(x)$. Construct a strategy such that, once \mathcal{B} has been correct at least $\log n$ more times than it has been incorrect, you can output $\text{Opt}(x)$ exactly and correctly.
 - (c) Conclude that one can solve $\text{Opt}(x)$ in $O(T \log n)$ time with high probability (which means $1 - 1/n^c$ probability for an arbitrarily large constant c).
2. [MR 2.3]. Consider a uniform rooted tree of height h (every leaf is at distance h from the root). The root, as well as any internal node, has 3 children. Each leaf has a boolean value associated with it. Each internal node returns the value returned by the majority of its children. The evaluation problem consists of determining the value of the root; at each step, an algorithm can choose one leaf whose value it wishes to read.
 - (a) Show that for any deterministic algorithm, there is an instance (a set of boolean values for the leaves) that forces it to read all $n = 3^h$ leaves.
 - (b) Show that there is a nondeterministic algorithm can determine the value of the tree by reading at most $n^{\log_3 2}$ leaves. In other words, prove that one can present a set of this many leaves from which the tree value can be determined.

- (c) Consider the recursive randomized algorithm that evaluates two subtrees of the root chosen at random. If the values returned disagree, it proceeds to evaluate the third sub-tree. Show the expected number of leaves read by the algorithm on any instance is at most $n^{0.9}$.