Problem Set II

6.897 Algorithms for Massive Data Sets

Due: Tuesday, December 3

Grading policy:

- All problems are worth 10 points. This does *not* mean that all of them are equally easy or hard. If it helps, I believe that P1 is the easiest one, followed by P0, P3, P4 and P2 (in the increasing order of difficulty).
- In your solutions, focus on main ideas, instead of writing extremely detailed answers. This is a *graduate* class, and I am not going to subtract points because of some small omissions/errors. Therefore, try to avoid writing long solutions. Each time you start another page, think about the rainforest.

Problem 0: Speeding-up histogram recovery. In Lecture 14, we have seen how to recover (approximately) the best B-bucket histogram h for $x \in \Re^n$, from a k-dimensional sketch Ax, for $k = O(B \log n)$. The recovery algorithm proceeded by starting with $h = 0$, and then iteratively choosing the interval I and value c which minimizes

$$\|Ax - A(h \oplus c\chi_I)\|^2$$

and replacing h by $h \oplus c\chi_I$ (recall that the histogram $h' = h \oplus c\chi_I$ is such that $h'(i) = c$ if $i \in I$ and $h'(i) = h(i)$ if $i \notin I$).

Show how to find I and c minimizing the above function in $O(n^2 k)$ time (as opposed to $O(n^3 k)$ time algorithm seen in the class). Moreover, show that if the choice of I is restricted to dyadic intervals, then your algorithm has running time $O(nk \log n)$.

Problem 1: Efficient computation of BW transform. Show that, given a text string T of length n, one can compute BW transform of T in time $O(n \log n)$.

Hint: Use reduction.

Problem 2: BW on disk. Since we are already at it, show how to perform BW transform when the text resides on disk (with block size B) in time $O(n/B \cdot \log^{O(1)} n)$.

Problem 3: Error-correcting codes. Consider the following code. For each $x \in \{0,1\}^k$, the codeword $C(x)$ has length $n = 2^k$. The i-th coordinate $C(x)_i$ of $C(x)$ is equal to

$$\sum_{j=1}^{k} x_j \cdot \mathrm{bin}(i)_j \bmod 2$$

where $bin(i)$ denotes the binary vector of length k containing the binary representation of i.

Show that the minimum distance of C is equal to $n/2$.

Problem 4: Error-correcting codes ctd. Consider the code C defined above. Assume that you are given a message $y \in \{0,1\}^n$ which is within Hamming distance (say) $n/20$ from $C(x)$ for some x. Show that for any $i = 1 \ldots k$, one can retrieve x_i with constant probability $> 1/2$, by reading only *two* positions of $C(x)$. Note: this is not a mistake, your algorithm should *not* decode the whole vector x, but only its i-th coordinate.

Hint: The first probed position j_1 should be random, and the second one should depend on j_1 and i.