

CS599 (Spring 2009) - Takehome Final

Due in David Kempe's office by 12:00 noon on Wednesday, 05/13

No late submission will be granted. As on the midterm, you cannot discuss problems, ideas, or solutions to this midterm with anyone (in or outside of class) except the instructor. Also, as a reminder, you are not allowed to seek solutions to these problems online or elsewhere. If you are stuck on a problem and need hints, or you need help understanding a problem, you are welcome to e-mail me, or talk to me in person.

Problem 1

In class, we used a coupling argument to prove that the random walk on the d -dimensional hypercube mixes in time $O(d \log d)$. Reprove that the random walk mixes rapidly using the technique of canonical paths. (Probably, the bound you will obtain on the time it takes to be nearly mixed will be worse than the one we obtained using coupling. So long as it is polynomial in d , that is not a problem.)

Problem 2

- Consider the random walk on the line $\{1, 2, \dots, n\}$, with a stalling probability of $\frac{1}{2}$. (I.e., with probability $\frac{1}{2}$, the random walk does not move.) Prove that the random walk mixes within $O(n^2)$ steps, in the sense that the TV distance to the stationary distribution is at most $\frac{1}{4}$ after $O(n^2)$ steps for this walk.
- Generalize the result to the d -dimensional grid $\{1, 2, \dots, n\}^d$. Prove that the random walk with stalling probability $\frac{1}{2}$ on the d -dimensional grid mixes in $O(n^2 \cdot d \log d)$ steps, in the same sense as above. (You can use the first part of this problem even if you didn't solve it.)

Problem 3

The BIN PACKING problem is defined as follows: you are given n items, with the size of item i being $s_i \in (0, 1)$, a fractional number between 0 and 1. You are given a supply of bins, each of size exactly 1. You want to pack all the items in bins, and minimize the number of bins. That is, the objective function is the number of bins used to pack all items. Of course, a bin may be filled only partially, but you cannot overfill any bin. This problem is known to be NP-complete.

- Give and analyze a simple deterministic algorithm using at most $2\text{OPT} + 1$ bins, where OPT is the optimum number of bins to use.
- We now improve the approximation guarantee to $(1 + 2/e)\text{OPT} + 1$, by using LP-rounding. We say that an n -dimensional 0-1 vector a is *feasible* if the items i with $a_i = 1$ can all fit into one bin, i.e., $\sum_i a_i s_i \leq 1$. Let a^1, \dots, a^m be all feasible 0-1 vectors. Notice that m will be exponentially large in n . We can now write the following IP for Bin Packing:

$$\begin{array}{ll} \text{Minimize} & \sum_j x_j \\ \text{subject to} & \sum_j a_i^j x_j \geq 1 \quad \text{for all } i \\ & x_j \in \{0, 1\} \quad \text{for all } j. \end{array}$$

Of course, we can turn this into an LP in the usual way. The meaning of the variable x_j is that if $x_j = 1$, then the set of items with $a_i^j = 1$ is put into a bin together.

While this LP has exponential size, and is thus not obviously solved, you get to assume that you have a fractional optimum solution x_j . In fact, you get to assume that the solution only has polynomially many x_j entries which are non-zero, and you are given this list of non-zero entries explicitly. So you don't have to deal with exponentially many variables any more.

Show how to use this solution in order to obtain a solution using at most $(1 + 2/e)\text{OPT} + 1$ bins. (Hint: you may want to use your solution to part (1).)

Problem 4

Here, we prove that random graphs tend to be good edge expanders. Recall that the definition of the edge expansion of a graph G is

$$\alpha(G) := \min_{S \subseteq V, |S| \leq n/2} \frac{|e(S, \bar{S})|}{|S|},$$

i.e., the smallest ratio of edges leaving S divided by the size of S . Thus, graphs with large expansion don't have bottlenecks separating large parts of the graphs, whereas graphs with small expansion do.

Suppose that we generate a random graph as follows: each node v gets to have $c \log n$ outgoing edges, and the other endpoints of those edges are chosen independently and uniformly at random. For simplicity, we allow parallel edges and self loops. Thus, each node just chooses $c \log n$ edge endpoints, and creates edges to all of them. Prove that for some constant c of your choice (but independent of n), the resulting graph has expansion at least $\frac{1}{2}$, with high probability (say, at least $1 - 1/n$).

(This actually holds not only for degree $c \log n$, but if every node has degree at least 3. For degree 3, it is somewhat tricky to prove, and for degree 6, it is a bit easier, but still a little cumbersome. You only need to prove it for $c \log n$ here.)

Problem 5

Sometimes, randomized algorithms are really good at saving space. Imagine the following scenario: you have a sequence of m items, each from a universe U of size n . Items may repeat, possibly many times, and you want to know how even or uneven the distribution is. Specifically, if item i appears a total of x_i times in the sequence, you want to know the variance $\sum_i (x_i - m/n)^2 = \sum_i x_i^2 - m^2/n$. Since m, n are easy to keep track of, this is equivalent to calculating $\sum_i x_i^2$.

Now, this would be a very easy problem if you could simply count how many times you have seen each element i . Unfortunately, you don't have enough space for n separate counters. You only want to keep track of one number instead. One way to accomplish this — surprisingly — is as follows: for each i , generate independently a number ξ_i , which is ± 1 with probability $\frac{1}{2}$ each. Start out with a counter $Y = 0$. Now, whenever you see another occurrence of element i , add ξ_i to Y , i.e., increment or decrement Y , according to the element i .

- (a) Prove that $E[Y^2] = \sum_i x_i^2$.
- (b) The previous part shows that in expectation, this method lets you calculate $\sum_i x_i^2$. Unfortunately, the variance of Y^2 is quite high. To do better, we can run in parallel k independent copies of this algorithm, with independent random choices $\xi_{i,j}$. Call the resulting counters Y_1, \dots, Y_k . Then, we instead keep track of $Z := \frac{1}{k} \cdot \sum_{j=1}^k Y_j^2$. Bound the probability that Z deviates from $\sum_i x_i^2$ by more than a $(1 \pm \delta)$ factor, in terms of δ and k .