6.856 — Randomized Algorithms

David Karger

Mar. 24, 2021 — Problem Set 6, Due 3/31

Problem 1. Consider the following alternative parallel algorithm for maximal independent set. In a phase, a (non maximal) independent set S is output, and all of its neighbors are deleted. To find S, randomly permute the vertices, then mark each vertex that precedes all of its neighbors in the permutation. It may help to use the following equivalent formulation: assign to each vertex a uniformly distributed random weight from the range $\{1, ..., n^4\}$. Mark all vertices, then in parallel unmark the larger-weight endpoint of each edge. The set of vertices that remain marked is S. Continue until the graph is empty.

(a) Argue that the set of vertices output over all phases is a maximal independent set.

(b) Assuming that v has degree d during a phase, what is the probability that vertex v stays marked?

(c) Show that this approach yields an **RNC** algorithm for maximal independent set.

Problem 2—This problem should be done without collaboration. The NP-hard *Max-Cut* problem is to divide the vertices into two groups so as to maximize the number of edges with one endpoint in each group. A common RNC heuristic for this problem is to *randomly partition* the vertices into two groups, which will cut half the edges in expectation. Explain how pairwise independence can be used to derandomize this algorithm, producing a cut of at least half the edges *with certainty* in NC.

Problem 3. Consider the problem of finding a *minimum weight* (total weight of included edges) perfect matching in a bipartite graph whose edges are given integer weights of magnitude bounded by a polynomial in the number of vertices n. Note that it is not possible to apply the Isolating Lemma directly to this case since the random weights chosen there would conflict with the input weights.

(a) Explain how you would devise an **RNC** algorithm for this problem. **Hint:** start by scaling up the input edge weights by a large polynomial factor. Apply random perturbations to the scaled weights and prove a variant of the Isolating Lemma for this situation.

(b) The parallel complexity of the version where the edge weights can have a polynomial number of *bits* has not yet been resolved. Note that arithmetic operations on such weights are still tractable. Explain why the **RNC** algorithm you developed above does not work in this case.

(c) Devise an **RNC** algorithm for finding a *maximum matching* (i.e., most possible edges) in a bipartite graph (without weights) that may not have a perfect matching. **Hint:** use the min-weight perfect matching algorithm above as a "black box" by making nonexistent edges very expensive.

Problem 4. Suppose you are given a graph whose edge lengths are all integers in the range from 0 to B (inclusive). Suppose also that you are given the all-pairs distance matrix for this graph (it can be constructed by a variant of Seidel's deterministic distance algorithm). Prove that you can identify the (successor matrix representation of the) shortest paths in $O(B^2 MM(n) \log^2 n)$ expected time, where $MM(n)$ is the time to multiply $n \times n$ matrices.

Problem 5—Optional. In the *exact matching* problem, a bipartite graph is given with a subset of the edges colored red, along with an integer k. The goal is to find a perfect matching with exactly k red edges. Devise an **RNC** algorithm for this problem using a (non-trivial) application of the Isolating Lemma. Note that this problem is not known to be solvable in **P**.

Problem 6—Optional. One use of maximal independent sets is in wireless networks. A wireless network can be modeled as an n-vertex graph, where an edge between two nodes indicates that the two nodes will collide (interfere) with each other if they transmit at the same time. When this happens, messages being transmitted get garbled.

In this setting, an independent set represents a set of nodes that can all transmit simultaneously without any collisions, which provides useful parallel exploitation of the available communication capacity. We have seen a parallel algorithm for constructing an independent set; unfortunately it requires all nodes to communicate simultaneously with their neighbors, which leads to collisions and garbled communication.

In this problem you will show how to construct an independent set even as messages become garbled through collisions. We take the following approach. Suppose that in each round every node can choose to *talk* or to *listen*. Afterwards, each node will know whether any of its neighbors talked. But if multiple neighbors talk, garbled messages mean the node will know some neigbor talked but will not know *which* neighbor talked or what was said. It turns out that we can still find an independent set quickly.

In a round of the algorithm, each node talks with probability p for some p to be chosen later. If, during a round, any node talked but none of its neighbors talked, that node immediately

joins the independent set and announces this fact. Upon hearing this announcement, its neighbors immediately *go dead* and never talk again.

(a) For this subproblem, suppose the maximum degree of the graph is at most D. Prove that by performing a round of the algorithm with $p = \frac{1}{2D}$, any vertex with degree at least $D/2$ is deleted with constant probability. Conclude that after $O(\log n)$ rounds, the maximum degree of the graph becomes at most $D/2$ with high probability.

(b) Use part (a), devise a randomized algorithm (which works with high probability) that finds a maximal independent set in $O(\log^2 n)$ rounds.

(c) A problem with the above scheme is that in practice, a node that is talking (transmitting) overloads its own receiver so cannot hear if any neighbor talks. Show how to overcome this problem. In particular, show how a network where talkers can't simultaneously listen can simulate (with high probability) a network where talkers can listen, with an $O(\log n)$ factor slowdown in the number of rounds needed. **Hint:** To simulate one talking-while-listening round, run $O(\log n)$ rounds in which each node that wants to talk does so with probability $1/2$.