CS 577

Assignment 13 – Randomization

Spring 2022

Answer the questions in the boxes provided on the question sheets. If you run out of room for an answer, add a page to the end of the document.

Name: ____________________ Wisc id: ____________________

Randomization

1. *Kleinberg, Jon. Algorithm Design (p.782, q.1).*

   3-Coloring is a yes/no question, but we can phrase it as an optimization problem as follows.

   Suppose we are given a graph $G = (V, E)$, and we want to color each node with one of three colors, even if we aren't necessarily able to give different colors to every pair of adjacent nodes. Rather, we say that an edge (u, v) is *satisfied* if the colors assigned to u and v are different. Consider a 3-coloring that maximizes the number of satisfied edges, and let c^* denote this number. Give a polynomial-time algorithm that produces a 3-coloring that satisfies at least $\frac{2}{3}c^*$ edges. If you want, your algorithm can be randomized; in this case, the expected number of edges it satisfies should be at least $\frac{2}{3}c^*$.

2. *Kleinberg, Jon. Algorithm Design (p.787, q.7).*

   In lecture, we designed an approximation algorithm to within a factor of 7/8 for the MAX 3-SAT Problem, where we assumed that each clause has terms associated with three different variables. In this problem, we will consider the analogous MAX SAT Problem: Given a set of clauses $C_1, \ldots, C_k$ over a set of variables $X = \{x_1, \ldots, x_n\}$, find a truth assignment satisfying as many of the clauses as possible. Each clause has at least one term in it, and all the variables in a single clause are distinct, but otherwise we do not make any assumptions on the length of the clauses: There may be clauses that have a lot of variables, and others may have just a single variable.

   (a) First consider the randomized approximation algorithm we used for MAX 3-SAT, setting each variable independently to true or false with probability 1/2 each. Show that the expected number of clauses satisfied by this random assignment is at least $k/2$, that is, at least half of the clauses are satisfied in expectation. Give an example to show that there are MAX SAT instances such that no assignment satisfies more than half of the clauses.

(b) If we have a clause that consists only of a single term (e.g., a clause consisting just of x_1, or just of $\overline{x_2}$), then there is only a single way to satisfy it: We need to set the corresponding variable in the appropriate way. If we have two clauses such that one consists of just the term x_i, and the other consists of just the negated term $\overline{x_i}$, then this is a pretty direct contradiction. Assume that our instance has no such pair of "conflicting clauses"; that is, for no variable x_i do we have both a clause $C = \{x_i\}$ and a clause $C' = \{\overline{x_i}\}$. Modify the randomized procedure above to improve the approximation factor from 1/2 to at least .6. That is, change the algorithm so that the expected number of clauses satisfied by the process is at least $.6k$.

(c) Give a randomized polynomial-time algorithm for the general MAX SAT Problem, so that the expected number of clauses satisfied by the algorithm is at least a .6 fraction of the maximum possible. (Note that, by the example in part (a), there are instances where one cannot satisfy more than $k/2$ clauses; the point here is that we'd still like an efficient algorithm that, in expectation, can satisfy a .6 fraction of the maximum that can be satisfied by an optimal assignment.)

3. *Kleinberg, Jon. Algorithm Design (p.789, q.10).*

   Consider a very simple online auction system that works as follows. There are n *bidding agents*; agent i has a bid b_i, which is a positive natural number. We will assume that all bids b_i are distinct from one another. The bidding agents appear in an order chosen uniformly at random, each proposes its bid b_i in turn, and at all times the system maintains a variable b^* equal to the highest bid seen so far. (Initially b^* is set to 0.) What is the expected number of times that b^* is updated when this process is executed, as a function of the parameters in the problem?

   Example. Suppose $b_1 = 20, b_2 = 25$, and $b_3 = 10$, and the bidders arrive in the order 1, 3, 2. Then b^* is updated for 1 and 2, but not for 3.

4. Recall that in an undirected and unweighted graph $G = (V, E)$, a *cut* is a partition of the vertices $(S, V \setminus S)$ (where $S \subseteq V$). The *size* of a cut is the number of edges which cross the cut (the number of edges (u, v) such that $u \in S$ and $v \in V \setminus S$). In the *MAXCUT* problem, we try to find the cut which has the largest value. (The decision version of MAXCUT is NP-complete, but we will not prove that here.) Give a randomized algorithm to find a cut which, in expectation, has value at least 1/2 of the maximum value cut.

5. Implement an algorithm which, given a MAX 3-SAT instance, produces an assignment which satisfies at least 7/8 of the clauses, in either C, C++, C#, Java, or Python.

   The input will start with a positive integer n giving the number of variables, then a positive integer m giving the number of clauses, and then m lines describing each clause. The description of the clause will have three integers x y z, where $|x|$ encodes the variable number appearing in the first literal in the clause, the sign of x will be negative if and only if the literal is negated, and likewise for y and z to describe the two remaining literals in the clause. For example, 3 -1 -4 corresponds to the clause $x_3 \wedge \overline{x_1} \wedge \overline{x_4}$. A sample input is the following:

```
10
5
-1 -2 -5
6 9 4
-9 -7 -8
2 -7 10
-1 3 -6
```

   Your program should output an assignment which satisfies at least $\lfloor 7/8 \rfloor m$ clauses. Return n numbers in a line, using a ± 1 encoding for each variable (the ith number should be 1 if x_i is assigned TRUE, and -1 otherwise). One possible correct output to the sample input would be:

```
-1 1 1 1 1 1 -1 1 1 1
```