COMP 6902 Spring, 2018

Homework Assignment #3
Due: July 11, 2018, by 11:55pm

---

[25] 1. **Random SAT**

Recall that in the Random SAT framework formulas are generated as follows. Given the number of variables n, a number of clauses m, and a number of literals per clause k, repeatedly select k out of n literals uniformly at random, and for each selected literal, choose whether it is negated or not with probability 1/2. Repeat this process until there are m distinct clauses (provided m is small enough).

For each of the following subquestions, give both the number for the numeric part of the subquestion, and a formula for the general part.

(a) Suppose there are $n = 5$ variables x_1, x_2, x_3, x_4, x_5, and you select one clause with $k = 5$ literals in it. How many assignments to these variables would satisfy this clause? Now give a formula for a number of assignments satisfying an arbitrary clause with $n = k$ literals.

(b) Now suppose that $n = 8$, $k = 5$. How many assignments to all $n = 8$ variables satisfy one clause with $k = 5$ literals? Give a formula for a number of assignments satisfying an arbitrary clause with $n > k$ literals.

(c) Now suppose that $n = 8$, $k = 5$ and $m = 3$. What is the average (that is, expected) number of assignments satisfying a randomly generated formula with these parameters? Give a formula for arbitrary n, k, m, where $n > k$ and $m \leq n$ for some constant $c \geq 0$

(d) Now, consider the following randomized algorithm: on a given formula ϕ generated as above with parameters n, k, m, pick a random assignment and check if it satisfies ϕ. If so, accept, otherwise fail. What is the probability that this algorithm will succeed, on average, for a formula ϕ generated with $n = 8, k = 5$ and $m = 3$? Give a bound on m such that the probability of this algorithm succeeding on formulas generated with parameters n, k, m is at least 1/2.

(e) How large should m be for $n = 8, k = 5$ to guarantee that the resulting formula is always unsatisfiable? Give a general formula for m as a function of arbitrary n and k.

[25] 2. **Search-to-decision reduction**

The optimization search problem MinTSP is defined as follows. Given a complete undirected graph G and a cost function $c\colon E \to \mathbb{R}_{\geq 0}$ on its edges (in binary), find a tour (that is, a Hamiltonian cycle in G) such that its cost (sum of the costs of all edges in the cycle) is minimal.

(a) Describe how to find the *cost* of an optimal tour in G with cost function c by using as a black box the decision problem TSP=$\{\langle G, c, B\rangle \mid \exists$ a tour in G of cost at most $B\}$. Your program should run in time polynomial in the length of the input, counting each call to TSP as one unit of time.

(b) Now describe how to find an optimal tour, again by using TSP as a black box. Use your answer from the previous subquestion.