Topics in Algorithms- Random SAT (0510-7410) - Spring 2014
Benny Applebaum

Please submit a printed copy of your solution by April 1st.

Problem Set #1

Question 1: 2-SAT in linear time

Describe an algorithm that solves any 2-SAT formula in linear time $O(n+m)$ where n is the umber of variables and m is the number of clauses. **Hint:** Carefully modify the unit-clause algorithm shown in class.

Question 2: Unit Clause

In class we saw that for $r < 8/3$ Unit-Clause($F_3(n, rn)$) outputs, with constant probability, a satisfying assignment. Extend the analysis of the algorithm to the case of random 4-CNF formulas. Find a threshold r^* such that for every $r < r^*$ Unit-Clause($F_4(n, rn)$) outputs, with constant probability, a satisfying assignment. Sketch the argument by stating the main (modified) claims – there is no need to re-prove these claims.

Question 3: Implementation

Implement the following programs in MATLAB. Submit the code and the resulting graphs.

1. Generate $F_k(n, m)$. Input: n, m and k. Output: a random k-CNF formula chosen from $F_k(n, m)$ encoded as an $m \times k$ matrix A whose entries are integers in $\{\pm 1, \ldots, \pm n\}$, and the row (i, j, k) corresponds to the clause $(\ell_i \vee \ell_j \vee \ell_k)$, where ℓ_i is the literal x_i if $i > 0$ and $\bar{x}_i$ if $i < 0$.

2. Evaluate k-CNF formulas. Input: a k-CNF formula A encoded as in the previous question, and an assignment $\alpha \in \{\pm 1\}^n$. Output: 1 if and only if α satisfies A.

3. Implement Unit Clause and test its performance as follows.

   (a) For $r_0 = 2.5$ generate 20 instances of $F_3(n, rn)$ for each of the following values $n = 2000, 2500, 3000, 3500, 4000, 4500, 5000, 5500, 6000$. Plot a grid whose x-axis is the number of variables and the y-axis is the number of steps performed by the algorithm, and put a blue point in the grid for every trial.

   (b) Conduct the same experiment with $r_1 = 3$. Add the points that represent this experiment to the previous grid as red points.