CS 473g: Algorithms, Fall 2007  
Midterm — October 4, 2007

Name: 
Net ID: Alias: 

- This is a closed-book, closed-notes, open-brain exam. If you brought anything with you besides writing instruments and your handwritten $8\frac{1}{2}'' \times 11''$ cheat sheet, please leave it at the front of the classroom.

- Print your name, netid, and alias in the boxes above. Print your name at the top of every page (in case the staple falls out!).

- You should answer all the questions on the exam.

- The last few pages of this booklet are blank. Use that for a scratch paper. Please let us know if you need more paper.

- If your cheat sheet is not hand written by yourself, or it is photocopied, please do not use it and leave it in front of the classroom.

- Submit your cheat sheet together with your exam. An exam without your cheat sheet attached to it will not be checked.

- If you are NOT using a cheat sheet you should indicate it in large friendly letters on this page.

- There are 4 questions on the exam. Each question is worth 25 points.

- Answers containing only the expression: “I don’t know”, will get 20% of the points of the question. If you write anything else, it would be ignored. Overall, points given for “I don’t know” will not exceed 10 points.

- Write your exam using a pen not a pencil.

- Time limit: 75 minutes.

- Relax. Breathe. This is just an easy, silly and stupid midterm.

<table>
<thead>
<tr>
<th>#</th>
<th>Score</th>
<th>IDK</th>
<th>Grader</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
1. **NP Completeness.**  
   **[25 Points]**

Prove that the following problem is **NP-Complete**.

**Problem:** Monotone Satisfiability

**Instance:** A CNF formula $F$ over $n$ variables $x_1, \ldots, x_n$, where NO variable appears in negation. And a parameter $k$.

**Question:** Is there a satisfying assignment to $F$, such that at most $k$ variables are assigned value 1, and all other variables are assigned value 0.

As a concrete example, $F = (x_1 \lor x_2 \lor x_3)(x_1 \lor x_4)$. The formula $F = (\pi \lor x_3)$ is of course NOT a legal input for Monotone Satisfiability since it contains negation. Observe that it easy to satisfy a monotone formula, as you can assign all variables the value 1. Here, however, we look for the assignment with minimum number of 1s (or with at most $k$ trues).

2. **Maximum matching**  
   **[25 Points]**

Given a graph $G$ with $n$ vertices and $m$ edges, a matching $M \subseteq E(G)$ is a set of edges such that no two edges of $M$ share an endpoint. A natural question is to find a matching of $M$ of maximum size. Provide a simple algorithm that outputs a matching set $X$, such that $|X| \geq \text{opt}/2$, where $\text{opt}$ is the size of the maximum size matching in $G$. How fast is your algorithm?

Prove that your algorithm provides the required approximation.

3. **Majority tree**  
   **[25 Points]**

Consider a uniform rooted tree of height $h$ (every leaf is at distance $h$ from the root). The root, as well as any internal node, has 3 children. Each leaf has a boolean value associated with it. Each internal node returns the value returned by the majority of its children. The evaluation problem consists of determining the value of the root; at each step, an algorithm can choose one leaf whose value it wishes to read.

Consider the recursive randomized algorithm $\text{EvalTree}$ that evaluates two subtrees of the root chosen at random. If the values returned disagree, it proceeds to evaluate the third sub-tree. If they agree, it returns the value they agree on.

(A) **[10 Points]** For a node $v$ in this tree, $\text{EvalTree}$ performs either two or three recursive calls on its children. In the worst case, what is the probability that $\text{EvalTree}$ performs only two recursive calls?

Let denote this probability by $\alpha$.

(B) **[10 Points]** Let $T(h)$ denote the expected time to evaluate a tree of height $h$ by $\text{EvalTree}$ (this is just the expected number of leafs evaluated by $\text{EvalTree}$). Give a recurrence that bounds $T(h)$ exactly (as a function of $h$ and $\alpha$ and $T(h-1)$).

(C) **[5 Points]** What is the expected number of leafs that would be read by $\text{EvalTree}$ if executed on a tree that has $n$ leaves (that is $n = 3^h$)?

[25 Points]

You are given a tree $T$ defined over $n$ nodes. Every node $v \in V(T)$ has an associated weight $c(v) \geq 0$ with it. Provide an algorithm, as fast as possible, that computes the independent set of maximum weight in $T$. A weight of a set of vertices is just the total weight of its vertices.

What is the running time of your algorithm? Provide a pseudo-code for your algorithm.