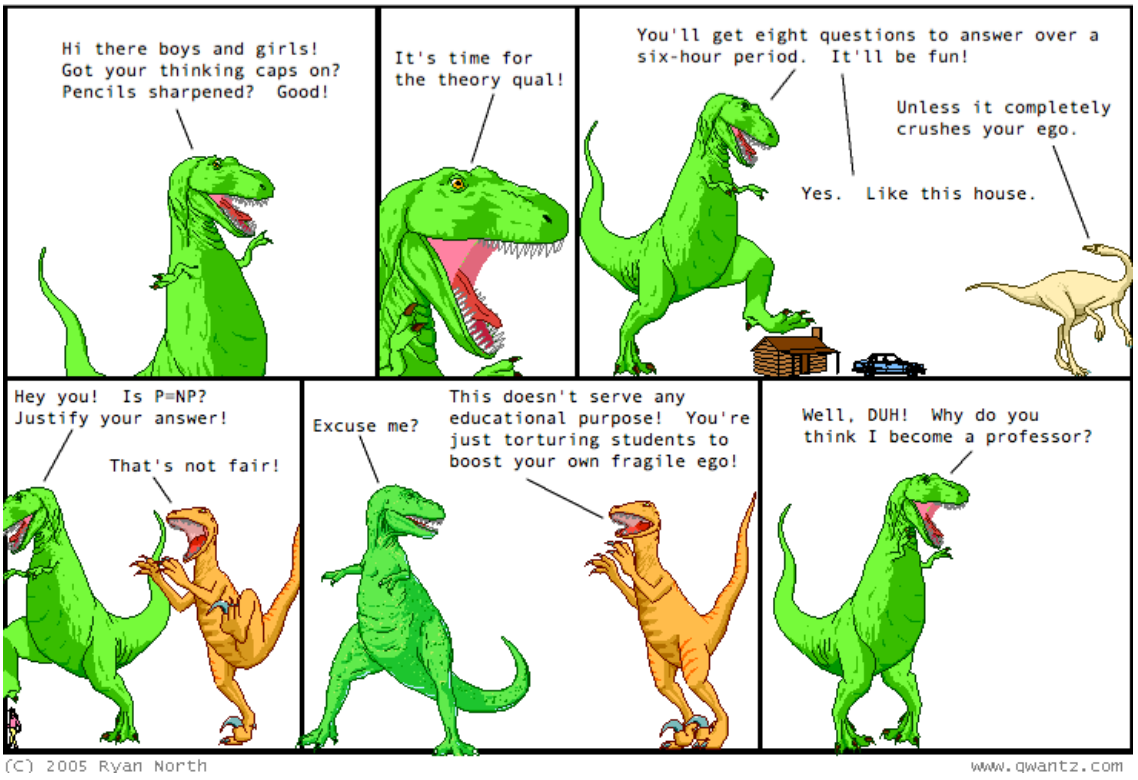
Algorithms and Theoretical Computer Science
Ph.D. Qualifying Examination
Fall 2005

- The exam consists of eight written questions, four in the morning and four in the afternoon. You will have three hours for each group of four questions. Please start your answers to each numbered question on a new sheet of paper.
- All else being equal, it is better to solve some of the problems completely than to get partial credit on every problem.



#	1	2	3	4	5	6	7	8	$\sum$
Score									
Grader									

1. (a) Prove that sorting n numbers requires $\Omega(n \log n)$ time in the worst case, if only comparisons are allowed.

   (b) Prove that given n numbers, deciding whether any two of them are equal requires $\Omega(n \log n)$ time in the worst case, if only comparisons are allowed.

2. A *prefix code* is a set of m bit strings $\{b_1, \ldots, b_m\} \in \{0,1\}^*$ where no string b_i is a prefix of another string b_j. Each string b_i is called the *code word* for symbol i.

   (a) Let ℓ_i denote the number of bits in the ith code word b_i. Prove that every prefix code satisfies the *Kraft inequality*:
   $$\sum_i 2^{-\ell_i} \leq 1.$$

   (b) A *probability distribution* is a vector $p = (p_1, p_2, \ldots, p_m) \in [0,1]^m$ such that $\sum_i p_i = 1$. What is the expected length of a random code word if each symbol i is chosen with probability p_i?

   (c) The *entropy* $H(p)$ of a probability distribution p is the quantity $\sum_i p_i \lg(1/p_i)$. Given the probability distribution p, show how to compute a prefix code such that the expected length of a random code word is at most $H(p) + 1$. How fast is your algorithm?

   (In fact, the entropy is maximized for the uniform distribution, and entropy is in fact a tight bound on the best prefix code one can generate.)

3. Let $M = (m_{ij})$ be an $n \times n$ matrix in which every row and every column is sorted. Such an array is called *totally monotone*. Assume further that no two elements of M are equal.

   (a) Describe an algorithm to solve the following problem in $O(n)$ time: Given indices i, j, k, l as input, compute the number of elements of M smaller than m_{ij} and larger than m_{kl}.

   (b) Describe an algorithm to solve the following problem in $O(n)$ time: Given indices i, j, k, l as input, return an element of M chosen uniformly at random from the elements smaller than m_{ij} and larger than m_{kl}. Assume the required range is non-empty.

   (c) Describe a randomized algorithm to compute the median element of M in $O(n \log n)$ expected time.

4. For any languages $A, B \subseteq \Sigma^*$, define
   $$A/B = \{x \mid xy \in A \text{ for some } y \in B\}$$
   Show that every recursively enumerable language is equal to A/B for some context-free languages A and B.

5. For any language $L \subseteq \{0,1\}^*$ and any integer n, let L_n denote the subset of strings in L of length at most n. We say that L is *self-reducible* if there exists a polynomial time oracle Turing machine M such that for any string $x \in \{0,1\}^n$, $x \in L$ if and only if $M^{L_{n-1}}$ accepts x. (Here M^A denotes the oracle Turing machine M with oracle A.)

   (a) Give an example of a language that is self-reducible.

   (b) Prove that if L is self-reducible, then L is in PSPACE.

6. This problem is concerned with probability distributions over $\{0,1\}^n$, the set of n-bit strings. A probability distribution is a function $p : \{0,1\}^n \to [0,1]$ such that $\sum_{x\in\{0,1\}^n} p(x) = 1$.

   A distribution p is *δ-balanced at position i* if and only if

$$-\delta \;\le\; \frac{p(y_1 \ldots y_{i-1}0y_i \ldots y_{n-1}) - p(y_1 \ldots y_{i-1}1y_i \ldots y_{n-1})}{p(y_1 \ldots y_{i-1}0y_i \ldots y_{n-1}) + p(y_1 \ldots y_{i-1}1y_i \ldots y_{n-1})} \;\le\; \delta,$$

   or equivalently,

$$\frac{1-\delta}{1+\delta} \le \frac{p(y_1 \ldots y_{i-1}0y_i \ldots y_{n-1})}{p(y_1 \ldots y_{i-1}1y_i \ldots y_{n-1})} \le \frac{1+\delta}{1-\delta},$$

   for every string $y \in \{0,1\}^{n-1}$. (You should verify that these two inequalities are equivalent.) A distribution is *δ-balanced* if it is δ-balanced at every bit position.

   Consider an arbitrary boolean function $f : \{0,1\}^n \to \{0,1\}$. Let p_0^f (resp. p_1^f) denote the probability that $f(x) = 0$ (resp. $f(x) = 1$) when in the input string x is randomly generated according to the distribution p:

$$p_0^f = \sum_{x|f(x)=0} p(x), \qquad p_1^f = \sum_{x|f(x)=1} p(x).$$

   Prove that for every boolean function $f : \{0,1\}^n \to \{0,1\}$ and every $\delta \in [0,1]$, there is a δ-balanced distribution p over $\{0,1\}^n$ such that $|p_0^f - p_1^f| \ge \delta$.

   [Hint: Consider separately the functions f for which $|u_0^f - u_1^f| \ge \delta$ and those for which $|u_0^f - u_1^f| < \delta$, where u is the uniform distribution.]

7. For any strings $x, y \in \Sigma^*$, write $x \sqsubseteq y$ if x is a (not necessarily contiguous) subsequence of y. For example, radar $\sqsubseteq$ abracadabra. A subset $L \subseteq \Sigma^*$ is said to be *downward closed* if $x \in L$ and $w \sqsubseteq x$ implies that $w \in L$. Prove that every downward closed set L is regular.

   [Hint: You may use the following observation due to Higman without proof: The set of $\sqsubseteq$-minimal elements of any set $L \subseteq \Sigma^*$ is finite.]

8. Let $x = x_1x_2 \ldots x_n$ be an n-character string over some finite alphabet Σ, and let A be a deterministic finite-state machine with m states.

   (a) Describe and analyze an algorithm to compute the longest (not necessarily contiguous) subsequence of x that is accepted by A. (For example, if A accepts the language (ar)* and $x =$ abracadabra, your algorithm should output arar.)

   (b) Describe and analyze an algorithm to compute the shortest (not necessarily contiguous) supersequence of x that is accepted by A. (For example, if A accepts the language (abcdr)* and $x =$ abracadabra, your algorithm should output abcdrabcdrabcdrabcdrabcdr.)