

# MS Examination

Department of Computer Science, University of New Mexico

January 2007

## 1 INSTRUCTIONS

The exam has 5 sections, corresponding to CS 451, 481, 500, 530, and 561.

**Answer exactly one question from each section**, and submit a front page indicating which problems you are solving.

Each problem solution should be on a separate sheet (or sheets) of paper, clearly marked with problem number, sheet number, and your name. Do not write on both sides of a sheet. Write legibly.

Track 2 students only: If your program of studies does not include one of CS 500 or 530, you may omit the question from that one course. Please indicate this on your front page.

The exam is closed-book, closed-notes, closed-computer. The proctor cannot provide clarification on any questions. You have approximately 36 minutes for each question: you have  $2\frac{1}{2}$  hours to complete the exam if you are answering 4 questions, and 3 hours if you are answering 5 questions.

Exam questions are confidential and must not be divulged to third parties.

## 2 CS 451

### 2.1 ML type system

Use the following examples to discuss the type system of ML:

```
1. let
    fun f x = x div 0
  in
    f 3
  end
```

```
2. let
    fun f x = x
  in
    f 3
  end
```

```
3. let
    fun f x = x
  in
    (f 3, f "a")
  end
```

```
4. let
    fun g (n: int) = n * n
    fun f (h: int -> int) = List.map (List.map h)
  in
    f g [[1],[2,3]]
  end
```

```
5. let
    fun g (n: int) = n * n
    fun twice f a = f (f a)
    fun f (h: int -> int) = twice List.map h
  in
    f g [[1],[2,3]]
  end
```

Where appropriate, provide the type and value of the example expressions and explain how you determined them. If an expression does not have a type, explain why.



## **3 CS 481**

### **3.1 Deadlock**

- Carefully define what deadlock is and what conditions are necessary for deadlock to occur among concurrent threads/processes.
- Describe one strategy for avoiding deadlock in concurrent systems

### **3.2 I/O Handling**

Carefully describe the steps an operating system must perform to start and complete a direct memory access (DMA) data transfer from a user-specified virtual memory buffer to disk.

## 4 CS 500

### 4.1 Automata

I call a string of *as* and *bs* *legal* if its length is a power of 2. A friend of yours claims to have a finite-state automaton which can read a string from left to right and tell if it is legal. Using any method you like, show that no such automaton exists.

### 4.2 NP-completeness

Consider the following variant of 3-SAT. I have a set of clauses, where each clause contains three literals, and each literal is either a variable or its negation. In 3-SAT, we demand that at least one literal in each clause be true. In MAJORITY SAT, we demand that a majority of the literals in each clause, i.e., two or three, are true. Either prove that MAJORITY SAT is NP-complete by reducing 3-SAT to it, or prove that it is in P by reducing it to 2-SAT.

## 5 CS 530

### 5.1 Information

Let  $X$  and  $Y$  be discrete random variables. I now define a new discrete random variable  $Z$  in the following way: first I flip a coin. If it is heads, then  $Z$  is a sample of  $X$ . If it is tails, then  $Z$  is a sample of  $Y$ . Show that the entropy  $H_Z$  is bounded by

$$H_Z \leq \log 2 + \frac{H_X + H_Y}{2}$$

where  $H_X$  and  $H_Y$  are the entropies of  $X$  and  $Y$ . Explain under what circumstances this is an equality instead of an inequality.

### 5.2 Probability

In a Markov process, let  $p^n(i|j)$  be the probability that, if I start in state  $j$ , I will be in state  $i$  after  $n$  steps. Recall that a Markov process is *irreducible* if for all  $i, j$  there is an  $n$  such that  $p^n(i|j) > 0$ , and *aperiodic* if for all  $i$  we have  $p^n(i|i) > 0$  for all sufficiently large  $n$ . Prove that if a Markov process is irreducible, and its transition matrix has at least one non-zero value on its diagonal, then it is also aperiodic.

## 6 CS 561

### 6.1 Locating the hospital

I have a set of  $n$  houses in a rural area, and I want to find a location for a hospital that minimizes the maximum distance from it to any of the houses. Formally, I have a set of  $n$  points in the plane, and I want to find the smallest circle that contains all  $n$  points. I will then locate the hospital at the center of this circle.

Give a algorithm for this problem. It does not have to be the most efficient algorithm possible, but it must run in polynomial time. Give a  $k$  such that its running time is  $O(n^k)$ , and explain why.

You may use the following as a subroutine without having to explain it: given any three points in the plane, in constant time we can find the center of the unique circle that passes through them. (We ignore pathological cases, such as when the three points lie on a straight line.)

### 6.2 Shortest paths

Consider the following pseudocode for ALL-PAIRS SHORTEST PATHS. We start with a matrix  $A$  where  $A(i, j)$  is the length of the edge connecting two vertices  $i$  and  $j$  (where  $A(i, j) = \infty$  if there is no edge between them) and we output a matrix  $B$  where  $B(i, j)$  is the length of the shortest path from  $i$  to  $j$ .

```
All-Pairs Shortest Paths(A)
B := A;
For k = 1 to n
  For i = 1 to n
    For j = 1 to n
      B(i, j) = min(B(i, j), B(i, k) + B(k, j));
Return B;
```

Prove that this algorithm works. Hint: what inductive guarantee can we make after running the outer loop  $k$  times?