

Questions for the MS Examination

Department of Computer Science, University of New Mexico

January 2006

The exam has 5 sections (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.

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.

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.

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.

1 CS 451

1.1 Elementary functional programming (100pts)

What is the type and value of the following SML expressions? Explain.

1.

```
let
  fun a m n = if m > n then nil else m :: a (m+1) n @ a (m+1) n
in
  (List.length o (List.filter op>)) o (List.map (fn a => (a, 9))) (a 1 10)
end
```
2.

```
let
  fun a m n = if m > n then nil else m :: a (m+1) n @ a (m+1) n
  val b = a 1 10
  val c = ListPair.zip (List.tl b, List.rev (List.tl (List.rev b)))
in
  (List.length o (List.filter op>)) c
end
```

1.2 ML type system (100pts)

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

1.

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

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

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

```
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.

2 CS 481

2.1 I/O (100 pts)

What is the difference between the following I/O strategies/techniques, and when is one preferable to the other?

1. Polling vs. Interrupt-driven I/O
2. DMA vs. Programmed I/O

2.2 Data storage (100pts)

Answer the following three questions.

1. What is the difference between internal and external fragmentation?
2. How can external fragmentation occur in a UNIX-style file system?
3. How can internal fragmentation occur in a memory-allocation (e.g. malloc) system?

2.3 Semaphores (100pts)

Informally define what a (counting) semaphore is and describe the major operations it supports.

3 CS 500

3.1 Rainbow Coloring (100 pts)

A *3-uniform hypergraph* $H = (V, E)$ is like a graph, but where each “hyperedge” $e \in E$ is a set of three vertices rather than 2. (If you like, you can think of each hyperedge as a triangle.) A *rainbow coloring* of H is an assignment of three colors to the vertices such that each hyperedge contains one vertex of each color, as in the following figure:

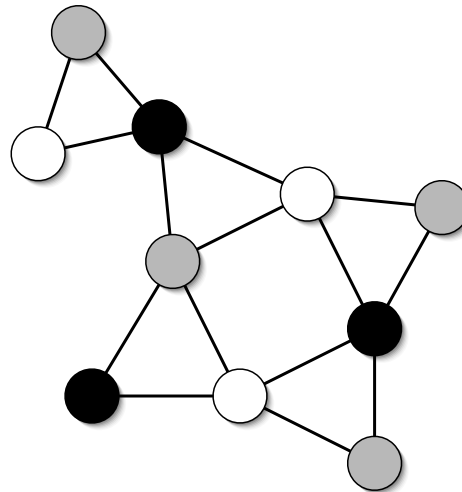


Figure 1: A hypergraph with 10 vertices and 5 hyperedges, and a rainbow coloring of it.

RAINBOW COLORABILITY is the question, given H , of whether H has a rainbow coloring. Prove that RAINBOW COLORABILITY is NP-complete by reducing to it from a classic NP-complete problem.

3.2 DNF-SAT (100 pts)

An instance of SAT consists of a Boolean formula in *conjunctive normal form* (CNF), where the formula is the AND (conjunction) of a set of clauses, each of which is the OR (disjunction) of a set of literals.

Suppose instead that I have a Boolean formula in *disjunctive normal form* (DNF), which is the OR of a set of clauses, each of which is the AND of a set of literals. For instance,

$$(x_1 \wedge \bar{x}_3 \wedge x_5 \wedge \bar{x}_7) \vee (x_{18} \wedge \bar{x}_{37}) \vee \dots$$

Prove that DNF-SAT, the problem of telling whether a DNF formula is satisfiable, is in P.

3.3 Forbidden Symbol (100pts)

Assume you are given a Turing Machine M over the alphabet $\{0, 1, \star\}$. In the “Forbidden Symbol” problem, the goal is to design an algorithm to determine if there is some input string that will cause M to write the symbol \star on its output tape. Can this problem be solved efficiently? If so, give an algorithm to do so. If not, prove that no efficient algorithm exists.

4 CS 530

4.1 Information (100pts)

Suppose I have a source of random strings which outputs symbols a, b, c, d with the following probabilities:

$$P(a) = 1/8$$

$$P(b) = 1/4$$

$$P(c) = 1/2$$

$$P(d) = 1/8$$

If I want to translate these strings into strings of bits, i.e., string of 0s and 1s, how many bits per symbol do I need on average?

4.2 Probability (100pts)

After years of driving through a particular intersection, you know that the probability of being stopped by a red light is $\frac{1}{4}$. You drive through the intersection twice a day, five days a week. What is the probability of being stopped by a red light five or more times in a single week?

4.3 Fourier Transforms (100pts)

Let $f(t) = -f(-t)$. Prove that $F(0) = 0$ where $F(s) = \mathcal{F}\{f\}(s)$ is the Fourier transform of f evaluated at frequency s .

5 CS 561

5.1 Selection (100pts)

Let S be a set of n real numbers a_1, a_2, \dots, a_n , not in sorted order. Given m integers k_1, k_2, \dots, k_m with $1 \leq k_1 < k_2 < \dots < k_m \leq n$, the multiple selection problem is that of finding, for each $i = 1, 2, \dots, m$, the k_i -th smallest element in the set S . Design an $O(n \log m)$ algorithm for solving this problem.

You may use the fact that when $m = 1$, i.e., when we are looking for the k th largest element for a single k , an algorithm exists which takes $O(n)$ time.

5.2 Minimum Spanning Tree (100pts)

Given a weighted graph $G = (V, E)$ in which each edge has a weight $w(e)$, a *minimum spanning tree* is a spanning tree where the total weight of all its edges is as small as possible. The classic greedy algorithm for finding minimum spanning trees relies on the following lemma:

Lemma 1 *Let $G = (V, E)$ be a weighted graph. Let $S \subset V$ be a subset of its vertices, let $U \subseteq E$ be the set of edges with one endpoint in S and the other in \bar{S} , and let $e \in U$ be the lowest-weight edge in U . Then there is a minimum spanning tree T of G which contains e .*

Prove this lemma.

5.3 Walking on a Graph (100pts)

Assume you are given an undirected and connected graph G . A *walk* over this graph is a sequence of vertices, W , with the property that any pair of vertices that are adjacent in W are connected by an edge in G , in which case the walk is said to *traverse* this edge.

Design a polynomial-time algorithm to find a walk in G that traverses every edge exactly twice, once in each direction.