PhD Qualifier Examination

Department of Computer Science and Engineering

Date: 03-Nov-2015 **Maximum Marks:** 100

Answer any five questions from Group A, and any five questions from Groups B and C.

Group A

A.1 Let f, g, h be positive real-valued functions of non-negative integers.

(a) Let $f(n) = (\log_2 n)^{\log_2 n}$, and $g(n) = 2^{(\log_2 n)^2}$. Prove that $f(n) = O(g(n))$. **(3)**

(b) Prove or disprove: $f(n) = \Theta(g(n))$ if and only if $\log f(n) = \Theta(\log g(n))$. **(3)**

(c) Let $f(n) = 1 + a + a^2 + \cdots + a^{n-1}$, where a is a positive real constant. Find all values of a, for which $f(n) = \Theta(a^n)$? **(4)**

A.2 Let T be a binary tree with root r. For every vertex v in T, there exists a unique r-v path—call this path P_v. The length of P_v is called the *level* of v. Any vertex lying on the path P_v (including r and v) is called an *ancestor* of v. Let n denote the number of nodes in T.

(a) Suppose that each node in T contains a field to store the level of the node. During the creation of T, these fields are kept uninitialized. After the creation of T, you want to populate the level fields properly at all the nodes of T. Propose an $O(n)$-time algorithm to do this task. **(5)**

(b) Given two vertices u, v in T, your task is to find out in $O(1)$ time whether u is an ancestor of v. Propose how this can be done. You may do an $O(n)$-time preprocessing of the tree, in which you are allowed to use extra space in each node of the tree. Clearly explain your preprocessing algorithm. **(5)**

A.3 You buy a black box which, upon the input of an integer array A of size n, outputs an integer array B of the same size. The manufacturer of the black box claims that B is the sorted version of A. You need to verify that the manufacturer's claim is valid.

(a) Design an $O(n)$-time algorithm that, given an array B of size n, tests whether B is sorted or not. **(3)**

(b) The black box passes the test of Part (a). But you realize that running only this test is not sufficient to establish that the black box is working correctly. Why? **(2)**

(c) You communicate your misgivings to the manufacturer, and the manufacturer agrees to modify the black box so that a second array C of size n will be output by the black box (in addition to B). It is your choice what C will consist of, but the creation of C must not take more time than sorting A (in the $O()$ notation). Explain what you want the array C to contain, and how you can use this extra information to test in $O(n)$ time the correctness of the working of the black box. **(5)**

A.4 During the end of the semester, you realize that you have n assignments to solve. The marks for these assignments are $m_1, m_2, \ldots, m_n$, and the times needed to solve these assignments are $t_1, t_2, \ldots, t_n$. You have a total time T in order to solve your pending assignments. There is no part marking, that is, you have to solve an assignment fully or leave it altogether. You want to choose a subset of the assignments, that you can finish in time T, such that the total marks attempted is as large as possible. Assume that all m_i, all t_j, and T are positive integers. Propose an $O(nT)$-time algorithm for this problem. (**Hint:** Prepare a two-dimensional array of size $(n+1) \times (T+1)$.) **(10)**

A.5 You have three glasses with capacities c_1, c_2, c_3 liters. Initially, the glasses contain water of volume a_1, a_2, a_3 liters. Eventually, you want the glasses to contain water of volume b_1, b_2, b_3 liters. Assume that a_i, b_i, c_i are non-negative integers and $a_i, b_i \leqslant c_i$ for all $i = 1, 2, 3$, and $a_1 + a_2 + a_3 = b_1 + b_2 + b_3$. You have no measuring device nor any source or sink of water. All you can do is to transfer water from one glass to another until either the former becomes empty or the latter becomes full (whichever happens earlier). Your task is to find out whether there exists a sequence of moves (water transfers) such that the initial configuration (a_1, a_2, a_3) can be changed to the desired configuration (b_1, b_2, b_3). Pose this problem as a graph-theoretic problem. How do you solve the problem using an efficient algorithm? What is the time complexity of your algorithm? **(5+3+2)**

A.6 Catalan numbers C_n are recursively defined as: $C_0 = 1$, and $C_n = C_0C_{n-1} + C_1C_{n-2} + \cdots + C_{n-2}C_1 + C_{n-1}C_0$ for $n \geqslant 1$. Write an efficient C function that, upon the input of n, returns the value of C_n. **(10)**

A.7 You are given a black-box implementation of a stack. You do not know how the stack data in this implementation is organized. All you can do is to make the following calls to perform the basic utility functions on stacks.

```
stack initStack()
int isEmpty(stack S)
stack push(stack S, int x)
stack pop(stack S)
int top(stack S)
```

Write a recursive function `stack sortStack(stack S)` to sort a stack. The function modifies the original stack to a sorted one. The stack elements in the output stack follow a non-decreasing sequence from bottom to top. You can define additional functions if necessary. **(10)**

A.8 Let $G = (V, E, w)$ be a weighted directed acyclic graph with $V = \{0, 1, 2, \ldots, n-1\}$, $|E| = m$, and $w : E \to \mathbb{Z}^+$ (where $\mathbb{Z}^+$ is the set of positive integers). Any edge $(i, j) \in E$ satisfies $i < j$ (however, not every pair (i, j) with $i < j$ is required to be present in E). The edge weight $w(e)$ stands for the profit that you encounter when you follow the edge e. Your task is to enter the graph at Vertex 0, collect as much profit as possible by following the directed edges in E, and leave the graph at Vertex $n-1$.

(a) Define a C data type to store G in the adjacency-list representation. **(3)**

(b) Write an $O(n+m)$-time C function to find the maximum profit that you can have in a path in G from Vertex 0 to Vertex $n-1$. If no such path exists, your function should return 0. **(7)**

Group B

B.1 (a) Let $f(n) = n^3 + n + 1$, where n is a positive integer-valued variable. Prove that for infinitely many n, the integer value $f(n)$ is divisible by three. **(5)**

(b) Let R be a binary relation on a set S, satisfying the property that for all $a, b, c \in S$, whenever $(a, b) \in R$ and $(c, b) \in R$, we have $(a, c) \in R$. Prove or disprove: R must be a transitive relation on S. **(5)**

B.2 (a) For integers $m, n \geqslant 0$, let $T(m, n)$ denote the count of functions $f : \{0, 1, 2, \ldots, m\} \to \{0, 1, 2, \ldots, n\}$ such that $f(0) = 0$, $f(m) = n$, and $f(i-1) \leqslant f(i)$ for all $i = 1, 2, \ldots, m$. Prove that for all $m, n \geqslant 1$, we have $T(m, n) = T(m, n-1) + T(m-1, n)$. **(5)**

(b) Let A be a set of n 4-digit integers (that is, a subset of size n of $\{1000, 1001, \ldots, 9999\}$). What is the smallest n for which A is guaranteed to contain two different integers with the same sum of digits? **(5)**

B.3 There are two bags: the first bag contains three red and three blue balls, and the second bag contains three red and two green balls. One of the two bags is chosen at random (with equal probability), and a ball is drawn at random from the chosen bag. The ball drawn is not returned to any of the bags. For a second time, a bag is chosen uniformly at random, and a ball is drawn randomly from the chosen bag.

(a) What is the probability that both the balls drawn are red? **(5)**

(b) Given that both the balls drawn are red, what is the probability that both the balls are drawn from the first bag? **(5)**

B.4 Let A be a non-empty language over the alphabet $\{0, 1\}$. Define

$$A^+ = \{\alpha 0 \beta \in \{0, 1\}^* \mid \alpha\beta \in A\},$$

that is, A^+ is obtained from A by adding one 0 to each string in A (at any position).

(a) Give an example where $A^+ \subseteq A$, and another example where $A^+ \not\subseteq A$. **(2 + 2)**

(b) Prove that for any non-empty language A, we have $A^+ \neq A$. **(6)**

B.5 One of the following languages is regular, the other not. Which one is what? Give proper justifications. **(5 + 5)**

(a) $L_{5a} = \{a^i b^j \mid i, j \geqslant 0 \text{ and } i + j \geqslant 10\}$.

(b) $L_{5b} = \{a^i b^j \mid i, j \geqslant 0 \text{ and } i - j \geqslant 10\}$.

B.6 Let L_6 be the context-free language over the alphabet $\{a, b, c\}$, generated by the following grammar with the start symbol S:

$$S \rightarrow c \mid cS \mid aS \mid aSbS$$

(a) Show a derivation of the string *accbac* following this grammar. **(3)**

(b) Find a string α starting with a and ending with c, such that $\alpha \notin L_6$. **(2)**

(c) Find a string $\alpha \in L_6$ such that α has (at least) two different parse trees under the given grammar. **(5)**

Group C

C.1 (a) Simplify the switching expression: $AB + \overline{AC} + A\overline{B}C(AB + C)$. **(4)**

(b) Optimize the switching function $f(a, b, c, d, e) = \sum m(0, 2, 8, 10, 16, 17, 18, 19, 24, 26)$ using Quine–McCluskey algorithm. **(4)**

(c) How are don't care terms handled by the Quine–McCluskey algorithm? **(2)**

C.2 A bit-wise shift-and-add multiplier has external inputs as follows:

- L: load operand (active high)
- M: multiplicand
- X: multiplier

It has an internal status line Z (active high) to indicate that multiplication is over. It uses internal control signals $W/\overline{M}$ to load operands ($W/\overline{M} = 1$) or to multiply ($W/\overline{M} = 0$). The output status line R (active high) indicates that the results are available for transferring out.

Design a finite state machine with minimum number of states to control the multiplier and give correct indication of availability of results. The FSM will have as inputs, the signals L and Z, and will have as outputs the signals $W/\overline{M}$ and R. Its states should be labeled $S_0, S_1, \ldots$. **(10)**

C.3 Non-negative integers x and y are stored in registers $\$s_0$ and $\$s_1$, respectively.

(a) Write a sequence of micro-MIPS instructions to compute x mod y and store the result in $\$t_0$. Use only the instructions from Table 1. Your answer should reasonably handle the special case $y = 0$. **(3)**

(b) Augment the instruction sequence in Part (a) so that it also yields $\lfloor x/y \rfloor$ in $\$t_1$. **(2)**

(c) Let x and y be k-bit 2's-complement numbers. An adder computes $s = x + y$ but does not produce a carry-out signal c_k. Prove that c_k can be derived externally as: $c_k = x_{k-1}y_{k-1} \vee s'_{k-1}(x_{k-1} \vee y_{k-1})$. **(5)**

C.4 A computer system has 4 GB of byte-addressable main memory and a unified 256 KB unified cache memory with 32-byte blocks.

(a) Draw a diagram showing each of the components of a main memory address (that is, how many bits for tag, set index, and byte offset) for a four-way set-associative cache. **(6)**

(b) The performance of the computer system with four-way set-associative cache architecture proves unsatisfactory. Two redesign options are being considered, implying roughly the same additional design and production costs. Option A is to increase the size of the cache to 512 KB. Option B is to increase the associativity of the 256 KB cache to 16-way. In your judgment, which option is more likely to result in better overall performance, and why? **(4)**

C.5 (a) A process-management module uses a prioritized round-robin scheduling policy (that is, the ready queue is implemented as a priority queue). New processes are assigned an initial quantum of length q. Whenever a process uses its entire quantum without blocking, its new quantum is set to twice its current quantum (for its next turn). If a process blocks before its quantum expires, its new quantum is reset

Table 1: Micro-MIPS ISA

Class	Instruction	Usage	Meaning
Copy	Load upper immediate	lui rt, imm	$rt \leftarrow (imm, 0x0000)$
Arithmetic	Add	add rd,rs,rt	$rd \leftarrow (rs) + (rt)$
	Subtract	sub rd,rs,rt	$rd \leftarrow (rs) - (rt)$
	Set less than	slt rd,rs,rt	$rd \leftarrow$ if $(rs) < (rd)$ then 1 else 0
	Add Immediate	addi rt,rs,imm	$rt \leftarrow (rs) + imm$
	Set Less than immediate	slti rt,rs,imm	$rt \leftarrow if(rs) < imm$then 1 else 0
Logic	AND	and rd,rs,rt	$rd \leftarrow (rs) \wedge (rt)$
	OR	or rd,rs,rt	$rd \leftarrow (rs) \vee (rt)$
	XOR	xor rd,rs,rt	$rd \leftarrow (rs) \oplus (rt)$
	NOR	nor rd,rs,rt	$rd \leftarrow ((rs) \vee (rt))'$
	AND immediate	andi rt,rs,imm	$rt \leftarrow (rs) \wedge imm$
	OR immediate	ori rt,rs,imm	$rt \leftarrow (rs) \vee imm$
	XOR immediate	xori rt,rs,imm	$rt \leftarrow (rs) \oplus imm$
Memory Word	Load Word	lw rt,imm(rs)	$rt \leftarrow mem[(rs) + imm]$
	Store Word	sw rt,imm,(rs)	$mem[(rs) + imm] \leftarrow (rt)$
Control Transfer	Jump	j L	goto L
	Jump register	jr rs	goto (rs)
	Branch on less than 0	bltz rs,L	if$(rs) < 0$ then goto L
	Branch on equal	beq rs,rt,L	if $(rs) = (rt)$ then goto L
	Branch on not equal	bne rs,rt,L	if$(rs) \neq (rt)$ then goto L
	Jump and link	jal L	goto L; 31 $\leftarrow$ (PC)+4
	System call	syscall	Associated with an OS system routine

to q. In this question, assume that every process requires a finite total amount of CPU time. Is starvation possible for each of the following cases? The scheduler gives higher priority to (i) processes that have larger quanta, and (ii) processes that have smaller quanta. Justify your answers in both the cases. **(3)**

(b) A computer system implements 8 KB pages and a 32-bit physical address space. Each page-table entry contains a valid bit, a dirty bit, three permission bits, and the frame number. If the maximum size of the page table of a process is 24 MB, compute the length of the virtual address (in bits) supported by the system. **($2\frac{1}{2}$)**

(c) Explain the utility of a mid-term scheduler. **(2)**

(d) Suppose that a processor does not have a TSL (Test and Set lock) instruction, but does have an instruction "SWAP REG MEM" to swap the content of a register (REG) and a memory word (MEM) in a single indivisible (atomic) action. Show how you can use SWAP to implement mutual exclusion. Propose the entry_section and exit_section code modules. **($2\frac{1}{2}$)**

C.6 (a) Each process P_i, $i = 0, 1, 2, 3, \ldots, 9$ is coded as follows:

```
wait(mutex);
{Critical Section}
signal(mutex);
```

The code for P_{10} is identical except that it mistakenly uses `signal(mutex)` instead of `wait(mutex)`. What is the largest number of processes that can be inside the critical section at any moment (the semaphore mutex being initialized to 1)? **(3)**

(b) Identify the advantage of associating process ID information with each TLB entry. **(2)**

(c) Can a local page-replacement policy applied to the page frames allocated to a process affect the performance of any other process in the system? Justify. **(2)**

(d) Consider a handheld system with 1 MB main memory. Customized operating system of size 130 KB has been loaded in one partition of the memory. The contiguous-memory-allocation (variable partition) scheme is used with first-fit strategy. The processes arrive and terminate in the following sequence: P_1 (250 KB) arrives, P_2 (100 KB) arrives, P_3 (150 KB) arrives, P_2 terminates, P_4 (220 KB) arrives. All the processes (including OS) have been loaded from the lowest address of the hole. Show the state of the memory after each process arrival, termination. Now if process P_5 of size 200 KB arrives in the system, would it be possible to satisfy the request immediately? Would it be possible after compaction? What will be the (minimum) cost of compaction? What kind of address binding is required for compaction? **(3)**