General CS II (320102) Final Exam
May 23. 2007

NAME:

MATRICULATION NUMBER:

You have two hours (sharp) for the test;
Write the solutions to the sheet.

You can reach 80 points if you solve all problems. You will only need 72 points for a perfect score, i. e. 8 points are bonus points.

Different problems test different skills and knowledge, so do not get stuck on one problem.

To be used for grading, do not write into this box												
prob.	1.1	2.1	2.2	3.1	3.2	4.1	5.1	5.2	6.1	7.1	Sum	grade
total	4	4	6	12	8	10	10	8	12	6	80	
reached												

1 An Old GenCS Favourite ;-)

Problem 1.1 (Function Definition) 4pt 3min

Let A and B be sets. State the definition of the concept of a partial function with domain A and codomain B. Also state the definition of a total function with domain A and codomain B.

2 Graphs

Problem 2.1 (Alternative Definition of a Tree) 4pt 5min

Prof. Simplovsky approaches Prof. Kohlhase at a conference in Saskatchewan and suggests a different definition of trees which he claims is simpler than Prof. Kohlhase's (in the slides) and yet it fully captures the notion of trees too:

"A tree is a directed, connected acyclic graph with exactly one node with indegree zero, which we call the root node."

Is Prof. Simplovsky right? Explain your answer by proving the equivalence of the respective definitions or giving an example that differentiates them.

Note: We call a directed graph connected, iff for any two nodes n_1 and n_2 there is a path in the underlying *undirected* graph (that we get by disregarding all directions of the edges) starting at n_1 and ending at n_2. (If this property holds for the directed graph itself, it is called *strongly* connected instead.)

Problem 2.2 (Another Alternative Definition of a Tree) 6pt 8min

Prof. Trivialczyk approaches Prof. Kohlhase at a conference on Hawaii and notes that a fully balanced binary tree can be defined without introducing balanced trees before, but simply as *"a binary tree with the numbers of nodes at each depth equal to powers of two, and with a total number of nodes equal to* $2^n - 1$ *for some* $n \geq 1$*."*

Is Prof. Trivialczyk right? Explain your answer by proving the equivalence of the respective definitions or giving an example that differentiates them.

3 Combinatorial Circuits and Memory

Problem 3.1 (A Vending Machine Circuit) 12pt 20min

Given is a vending machine for candies that accepts coins of 5 and 10 cents only and stores the current credit c. One candy costs 15 cents. If the credit reaches 15 cents or more ($c \geq 15$), the machine dispenses one candy (you don't need to implement this! ;-) and reduces the credit by 15 cents, i. e. the new credit c' is $c - 15$.

Assume that no two coins can be thrown into the machine at the same time, and that a clock signal is only generated when a coin is thrown into the machine, i. e. there are no transitions like $c' = c + 0$.

Note: Take the clock signal as a "magic" input from outside. You do not need to generate or manipulate it yourself.

We shall model this machine as a black box that accepts an input for the coins (appropriately encoded) and gives two output bits that represent the current credit.

1. Devise an appropriate encoding for storing the credit and representing coins as input bits.
2. Draw a state diagram, i. e. a graph whose nodes are the possible states of the machine (here, a state is the value of c) and whose edges are possible transitions between the states that are taken on a certain input.
3. Now draw the sequential logic circuit. You may use D-flipflops and any combinatorial circuits or gates that have been introduced in this course. Make sure you remove any ambiguity in your drawing by properly labeling wires and/or circuit elements and sufficiently explaining the layout.

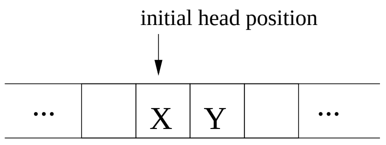
Problem 3.2 (Conditional Sum Adder) 8pt 10min
Draw the circuit of a Conditional Sum Adder (CSA) that adds two four-bit numbers. Go down to the level of elementary gates.

4 Virtual Machines

Problem 4.1 (While Loop in $\mathcal{L}(\texttt{VM})$) 10pt 15min
Write a program in the Simple While language that takes two numbers A and B, given at the memory addresses 1 and 2, and returns $(A+B)^{42}$. Show how the compiled version of it looks like in the Virtual Machine Language $\mathcal{L}(\texttt{VM})$ (concrete, not abstract syntax).

5 Turing Machines

Problem 5.1 (Boolean Equivalence) 10pt 15min
Consider a tape arbitrarily filled with ones and zeros and the head initially positioned over some cell "X" as depicted below



Define a transition table for an always terminating Turing machine TM that computes the boolean equivalence of "X" and "Y": Upon halting, your TM should return the value 1 in cell "X" if the values of the cells "X" and "Y" were initially equal and otherwise 0.

Try to use as few states as possible. The number of points you can obtain for this exercise is $\max(0, 14-x)$, where x is the number of states of your working TM.

Note:

1. Admissible moves are *left*, *right*, and *none* with the obvious meaning.
2. You are free to overwrite the initial value of "Y" and to introduce additional symbols in the alphabet, if you need it for your solution.

Problem 5.2 (Halting Problem) 8pt 10min

Define the Halting Problem:

- What does it state about the computational power of Turing machines?
- Outline (informally) how one would prove this statement.
- Why is the halting problem interesting; what is its practical relevance?

6 Problem Solving and Search

Problem 6.1 (Interpreting Search Results) 12pt 20min

The state of Ingushetia has only four cities (A, B, C, and D) and a few two-way roads between them, so that it can be modeled as an undirected graph with four nodes. The task is to go from city A to city D. The UCS algorithm finds a solution to this task that is 10km shorter than the one BFS finds. The solution of BFS in turn is 10km shorter than the one of the DFS algorithm.

Draw a map of Ingushetia with roads and their distances that satisfies both conditions. What paths between A and D in your map will be found as solutions by each of those algorithms?

Note: All algorithms had repetition checking implemented, so that when a node is expanded, all its children that belong to a list of previously expanded nodes during the execution of that algorihtm are ignored. In addition, when no order of choosing a node for expansion is specified by an algorithm, expansion in alphabetical order takes place.

---

7 Prolog

Problem 7.1 (Paths in a Graph) 6pt 10min

Given a directed graph, represented by `edge(from, to)` facts, write a predicate `trip(A, B, L)` that succeeds if the node `B` is accessible from `A` via the intermediate nodes `L` (an ordered list of nodes).

Note: It is not required to avoid cyclic trips.
