Exam in Models of Computation TDA 183

Date: Dec 15, 2008, 14.00 - 17.00

Permitted aids: English-Swedish or English-other language dictionary.

Teacher: Bengt Nordström, phone 0730 79 42 89, Computer Science, University of Gothenburg and Chalmers

All solutions must be explained! It is not enough to just give a program without an explanation of why it works. The examination of the course consists of three parts: homework assignments count up to 40 points, weekly exercises count up to 20 points and this written exam count up to 140 points (20 points for each of the seven parts). You have to have 100 points in total in order to pass the course.

Solutions to the exam will be available from the homepage of the course.

1. Two problems about computability:
   (a) Explain what it means for a function `f : N -> N` to be Turing-computable. In order to do this, you have to first explain how a natural number is represented.
   (b) The function `f : N -> N` which returns 1 for even numbers and is undefined for odd numbers is computable. Prove or disprove!
2. Give a definition of a primitive recursive function in **PRF**(1) which always returns 0. Don't forget to give an explanation of why it is correct!
3. Two problems about iteration:
   (a) Define (in a functional language) the function `iter` which has the following properties:

   ```
   iter :: Int -> (a -> a) -> a -> a
   iter n f a = f (f ... (f a)...)
   ```

   where there are `n` occurrences of `f`, `n` is non-negative.
   (b) Define (still in a functional language) the function `iter` without using recursion. You can assume the existence of the function `rec` which has the following definition in Haskell:

   ```
   rec 0 d e = d
   rec (n+1) d e = e n (rec n d e)
   rec _ d e = error "negative argument to rec"
   ```

4. Two questions about the operator for primitive recursion:
   (a) In the language **PRF** of primitive recursive functions there is an operator which is similar to the function `rec` above. Describe it by giving its abstract syntax (using the set **PRF**(n), the set of primitive recursive functions of arity n) and give its operational semantics (big step).

(b) Implement `rec` in the language **X**! Show first the standard representation of a natural number.

5. There is a computation model which exactly captures the set of all total computable functions from N to N. Prove or disprove!

6. This problem is about decidability and semidecidability of the halting problem:

   (a) There is a function in Haskell (or some other functional language)

   ```
   halt :: (N -> N) -> N -> Bool
   ```

   with the property that `(halt f i)` evaluates to `true` if `(f i)` terminates. Prove or disprove!

   (b) There is such a function with the property that `(halt f i)` evaluates to `true` if `(f i)` terminates and otherwise evaluates to `false`. Prove or disprove!

7. The set `P(N)` of all subsets of `N` is enumerable. Prove or disprove!