Candidate Number

G5029

THE UNIVERSITY OF SUSSEX

BSc FINAL YEAR EXAMINATION
MComp THIRD YEAR EXAMINATION
May/June 2017 (A2)

LIMITS OF COMPUTATION

Assessment Period: May/June 2017 (A2)

DO NOT TURN OVER UNTIL INSTRUCTED TO BY THE LEAD INVIGILATOR

Candidates should answer TWO questions out of THREE. If all three questions are attempted only the first two answers will be marked.

The time allowed is TWO hours.

Each question is worth 50 marks.

At the end of the examination the question paper and any answer books/answer sheets, used or unused, will be collected from you before you leave the examination room.

1. This question is about various notions of effective computability.

   (a) We used the WHILE-language to describe effective computability. What kind of commands (statements) does this language provide in its pure form without extensions? [3 marks]

   (b) Explain the concept of indirect addressing used by the RAM computation model. Why is this needed at all? [6 marks]

   (c) Let the following WHILE-program myprog be as follows:

```
myprog read X {
       while X {
          Z := cons nil Z;
          X := tl X
          };
       Z := tl Z
 }
 write Z
```


   i. Assuming that we start counting variables from 0, give the program-as-data representation of myprog. [13 marks]

   ii. What is $\mathcal{E}\,[\![\texttt{tl X}]\!]\,\{\texttt{X}:\ulcorner 4\urcorner, \texttt{Y}:\ulcorner 0\urcorner\}$? [3 marks]

   iii. What is $[\![\texttt{myprog}]\!]^{\texttt{WHILE}}(\ulcorner n\urcorner)$ for any $n \in \mathbb{N}$? Explain your result briefly, but you do *not* need to give a formal derivation. [5 marks]

   (d) Assume c_1 is a compiler from S to T written in L . Assume further c_2 is a compiler from L to M written in C. What do we get if we compile c_1 using c_2? Be as precise as possible. [4 marks]

   (e) Assume that languages S and T have the same datatype. What is the correctness condition for a compiling function (i.e. the semantics of a compiler) c from language S to T? Provide either the formal definition or an equivalently *precise* explanation. [6 marks]

   (f) Can we add a new instruction to the instruction set of a standard Turing Machine, such that the resulting new Turing Machines, let's call them Turing Machines Plus, can decide more problems than the standard type of Turing Machines? Explain your answer. [5 marks]

   (g) Can we remove an instruction from the definition of standard Turing Machines, such that the resulting new Turing Machines, let's call them Turing Machines Minus, can decide fewer problems than the standard type of Turing Machines? Explain your answer. [5 marks]

2. This question is about decidability and semi-decidability.

(a) Define precisely what it means for a set $A \subseteq$ `WHILE`-data to be `WHILE`-semi-decidable. [7 marks]

(b) Which of the following problems are `WHILE`-decidable? (No explanation needed.)

i. Halting problem
ii. Complement of the Halting problem
iii. Travelling Salesman problem
iv. Postman problem
v. Tiling problem
vi. The problem whether a natural number is a prime number

[6 marks]

(c) Give a problem (set) that is semi-decidable but *not* decidable. [3 marks]

(d) What proof technique (besides proof-by-contradiction) is used in the proof that the Halting Problem is undecidable? No explanation is required. [2 marks]

(e) What proof technique (besides proof-by-contradiction) is used in the proof of Rice's Theorem? Explain what this technique is applied to. You don't have to explain the technique itself. [4 marks]

(f) Explain for the following sets $A \subseteq$ `WHILE`-data whether they are `WHILE`-decidable. For each case, explain your answer. In cases where A is decidable this explanation should consist of a description of the decision procedure. Here $\ulcorner p \urcorner$ denotes the encoding of a `WHILE`-program as `WHILE`-data.

i. $A = \{ \ulcorner p \urcorner \mid$ `WHILE`-program p returns `nil` if its input encodes a `WHILE`-program that contains one or more `while` loops $\}$ [6 marks]

ii. $A = \{ \ulcorner p \urcorner \mid$ `WHILE`-program p contains no `while` loops $\}$ [6 marks]

iii. $A = \{ \ulcorner p \urcorner \mid$ `WHILE`-program p returns `nil` for finitely many inputs $\}$ [6 marks]

(g) Explain why an arbitrarily large and complicated but finite set $A \subseteq$ `WHILE`-data is always `WHILE`-decidable. [4 marks]

(h) Recall that the set $A \cup B$ contains the elements of A as well as the elements of B and nothing else. Let $A \subseteq$ `WHILE`-data and $B \subseteq$ `WHILE`-data. If A and B are both semi-decidable, is their union, $A \cup B$, also semi-decidable? Give a reasonable argument (no full proof required) for your answer. [6 marks]

3. This question is about complexity.

   (a) Consider $\texttt{WHILE}^{time(f)}$.

   i. Give a (precise) definition of the class $\texttt{WHILE}^{time(f)}$. [5 marks]

   ii. Assume you have two WHILE-programs $\texttt{p} \in \texttt{WHILE}^{time(n^2+13)}$ and $\texttt{q} \in \texttt{WHILE}^{time(34n+12)}$ that have the same semantics. Which one would you use and why? [6 marks]

   (b) What does the **N** in **NP** stand for? What does the **P** stand for? [4 marks]

   (c) The Travelling Salesman problem (TSP) is **NP**-complete.

   i. What does the statement above mean exactly? [5 marks]

   ii. Referring to TSP as example, explain to what extent approximation algorithms are a useful means to solve **NP**-complete problems (viewed as optimisation problems). [8 marks]

   (d) Which of the following problems are *known* to be **NP**-complete?

   i. 0-1 Knapsack problem

   ii. Complement of the Halting problem

   iii. Graph Colouring problem

   iv. Tiling problem

   v. Postman problem

   vi. Factorisation Problem

   [6 marks]

   (e) In the following give a rough description of *how* you would prove the statement in question. You do not have to give a proof, but you are supposed to sketch the required plan, i.e. which activity you would need to carry out and/or which theorems you would use in which way. Be as precise as possible.

   i. There exists a problem that can be decided by a WHILE-program in exponential time but not in polynomial time. [5 marks]

   ii. A certain (fixed) problem A, the details shall not be important, is in $\mathbf{LIN}^{\texttt{WHILE}}$. [5 marks]

   iii. A certain (fixed) problem A, the details shall not be important, can't possibly be in $\mathbf{P}^{\texttt{WHILE}}$ unless $\mathbf{P} = \mathbf{NP}$. [6 marks]

End of Paper