OLLSCOIL NA hÉIREANN, MÁ NUAD

NATIONAL UNIVERSITY OF IRELAND, MAYNOOTH

THIRD COMPUTER SCIENCE EXAMINATION

FOURTH COMPUTER SCIENCE EXAMINATION

THIRD COMPUTER SCIENCE AND SOFTWARE ENGINEERING EXAMINATION

FOURTH COMPUTER SCIENCE AND SOFTWARE ENGINEERING EXAMINATION

MASTERS OF COMPUTER SCIENCE (YEAR 1) EXAMINATION

MASTERS OF COMPUTER SCIENCE (YEAR 2) EXAMINATION

JANUARY 2004

PAPER CS355/SE307/CS403

COMPUTATION AND COMPLEXITY

Dr. Philip Morrow, Prof. R. Reilly, Mr. T. Naughton.

Attempt any THREE questions. Time Allowed: 2 hours.

1. Let the language $\text{VARNOCHANGE}_\text{J}$ be defined as $\text{VARNOCHANGE}_\text{J} = \{\langle J, v\rangle :$ J is a Java program, v is a variable declared in J, and when J is run the value in v never changes after v is initialised in its declaration statement$\}$. You are given that HALT_J is undecidable. HALT_J is defined $\text{HALT}_\text{J} = \{\langle J, y\rangle : y \in \mathbb{Z}, J$ is a Java function that takes an integer argument and makes no function calls other than `System.out.Print()`, and J halts on input $y\}$.

   (a) Prove that $\text{VARNOCHANGE}_\text{J}$ is undecidable. You may answer this question by assigning a name, mathematical construct, or piece of pseudocode to each of the numbered blanks in the proof template in Figure 1. Where blanks have the same number, this denotes their contents will be the same. Alternatively, you can choose to ignore the template and construct your own proof from scratch (such as a subroutine reduction). [15 marks]

   (b) Prove that $\text{VARNOCHANGE}_\text{J}$ is Turing recognisable or prove that it is not Turing recognisable. [5 marks]

   (c) Give a definition of the language $\overline{\text{VARNOCHANGE}_\text{J}}$ (the complement of $\text{VARNOCHANGE}_\text{J}$). Prove that $\overline{\text{VARNOCHANGE}_\text{J}}$ is Turing recognisable or prove that it is not Turing recognisable. [5 marks]

Proof. We will use a mapping reduction to prove the reduction __1__. Assume that __2__ is decidable. The function f that maps instances of __3__ to instances of __4__ is performed by TM F given by the following pseudocode.

$F =$ "On input $\langle$__5__$\rangle$:
1. Construct the following M' given by the following pseudocode.
   $M' =$ "__6__"
2. Output $\langle$__7__$\rangle$"

Now, $\langle$__7__$\rangle$ is an element of __8__ iff $\langle$__5__$\rangle$ is an element of __9__. So using f and the assumption that __2__ is decidable, we can decide __10__. A contradiction. Therefore, __2__ is undecidable. (This also means that the complement of __2__ is undecidable; the complement of any undecidable language is itself undecidable.)

Figure 1: Proof template for questions 1a and 2b.

2. (a) What is the distinction between the complexity of an algorithm and the complexity of a problem? [5 marks]

   (b) Let the language $\text{INFINITE}_{\text{TM}}$ be defined as $\text{INFINITE}_{\text{TM}} = \{\langle M \rangle : M$ is a TM and $L(M)$ is an infinite language$\}$. Prove that $\text{INFINITE}_{\text{TM}}$ is undecidable. You are given that AT_{TM} is undecidable. AT_{TM} is defined as $\text{AT}_{\text{TM}} = \{\langle M, w \rangle : M$ is a TM and w is a word and M accepts $w\}$. [20 marks]

   You may answer this question by assigning a name, mathematical construct, or piece of pseudocode to each of the numbered blanks in the proof template in Figure 1. Where blanks have the same number, this denotes their contents will be the same. Alternatively, you can choose to ignore the template and construct your own proof from scratch (such as a subroutine reduction).

3. (a) Gödel's Incompleteness Theorem begins with the concept that statements about numbers can be encoded as numbers. Define a computable function that assigns a unique natural number to each statement in arithmetic (statements consisting of symbols from the set $\{0, 1, \ldots, 9, (,), +, -, \times, \div, =\}$). [5 marks]

   (b) Figure 2 illustrates the space of languages 2^{Σ^*} for some finite Σ, where $|\Sigma| > 1$. Place each of the following languages, <u>and</u> its complement, in its appropriate place in this space. [12 marks]

   i. $\text{AT}_{\text{TM}} = \{\langle M, w \rangle : M$ is a TM and w is a word and M accepts $w\}$
   ii. $\text{NEM}_{\text{TM}} = \{\langle M \rangle : M$ is a TM and $L(M) \neq \emptyset\}$
   iii. $\text{SON}_{\text{TM}} = \{S : S$ is a binary string (strings are finite by default) and S contains at least three '1's$\}$
   iv. $\text{SE}_{\text{J}} = \{\langle J_1, J_2 \rangle : J_1$ and J_2 are Java functions that contain syntax errors$\}$
   v. $\text{NEQ}_{\text{J}} = \{\langle J_1, J_2 \rangle : J_1$ and J_2 are Java functions that recognise different languages$\}$

   (c) The cross product of two sets A and B, denoted $A \times B$, is the set of all possible 2-tuples, or pairs, (a, b) where $a \in A$ and $b \in B$. Prove that the cross product $\mathbb{N} \times \mathbb{Z}$ is countable. [8 marks]

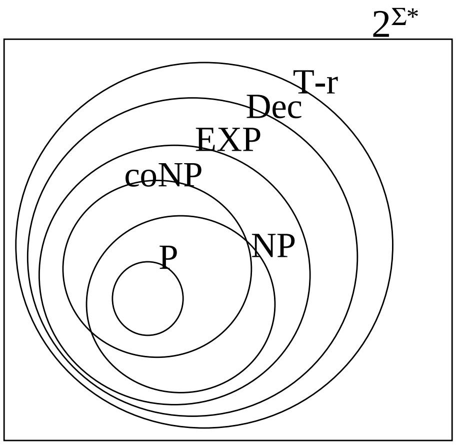

Figure 2: Illustration of the space of languages for question 3b: EXP denotes EXPTIME, Dec denotes the decidable languages (sometimes called the recursive languages), and T-r denotes the Turing-recognisable languages (sometimes called the recursively enumerable languages).

4. (a) The following TM M recognises the language $L = \{w1w : w \in \{0\}^*\}$. The start state is 00. To accept a word M goes into state 99. The symbol '–' denotes a blank. [5 marks]

S_i	R	S_f	W	M
00	0	01	–	R
01	0	01	0	R
01	1	01	1	R
01	–	02	–	L
02	0	03	–	L
03	0	03	0	L
03	1	03	1	L
03	–	00	–	R
00	1	04	–	R
04	–	99	–	R

Construct a TM M' that decides L. You do not need to write out the full table of behaviour for M', just specify how M needs to be modified. If you create new states, explain their purpose.

(b) Analyse the worst-case time and space complexity of M'. State the lower bounds on the complexity of a k-tape TM to decide L. [5 marks]

(c) Let PRINTERPROBLEM be defined as follows. You are required to print out a pile of n documents within a particular deadline. You have two printers at your disposal, each of which prints one page per second. Given a set $P = \{p_1, p_2, \ldots, p_n\}$ of nonnegative integers representing the number of pages in each of n documents to be printed, and a nonnegative integer t, can you print out all n documents in t seconds or less? You must print out whole documents at a time, but the order that the documents are printed is not important. Each instance of PRINTERPROBLEM has the form (n, P, t). Prove that PRINTERPROBLEM is $\mathcal{NP}$-complete. You are given that PARTITION is $\mathcal{NP}$-complete. [15 marks]