University of Twente
Department of Computer Science,
Faculty EEMCS
November 3, 2011, 8:45 – 12:15h.

Year 2011-2012
Period 1A
Interim examination

Computability and computational complexity
(192111700)

The use of course material during the examination is permitted. To hand in work produced with the help of others (by cribbing, by passing notes or other messages, by wireless consultation of persons or (internet) sources during the examination), is fraudulent.

Your answers to questions must be supported by a clear argument. To give an answer without explanation will have a negative influence on the number of points you will earn. This applies also to incomplete and unclear explanations.

You can earn 90 points in total. To determine your mark, the number of points earned is incremented by 10, and the total is divided by 10. The result is rounded to the nearest natural number.

There are 6 exercises, on 3 pages. The exercises are **not** ordered in increasing difficulty (nor conversely). Moreover, the number of points to be earned for an exercise is not necessarily a measure for the difficulty of that exercise.

Please put your name, your student number and the course name on every sheet of paper you hand in.

Veel succes, good luck, bonne chance!!

Exercise 1 (12 points).

The Subset Sum problem is the well-known problem to decide whether a given set of positive natural numbers has a subset of which the members add up exactly to a given total.

Time complexity is a property of solutions, not of problems. The problem representation influences the possible solutions, and the complexity of those solutions.

We speak of a proper representation of a problem if every change in the representation of an instance, makes it impossible to reconstruct that problem instance. In other words: in the representation is no redundancy.

(a) Is there a proper representation of the Subset Sum problem which allows a polynomial time solution? Please give a convincing argument for your answer.

Exercise 2 (20 points).

Discuss each of the following questions. Do not forget to explain your answers.

(a) L is a recursive language. The languages L_1 and L_2 are both recursively enumerable. Moreover, we know that $L_1 \cup L_2 = L$, and that $L_1 \cap L_2 = \emptyset$. Can we conclude that L_1 is recursive?

(b) Is it decidable whether the intersection $L_1 \cap L_2$ of two recursive languages L_1 and L_2 is empty?

(c) Is the language $L =_{def} \{w \in \{0,1\}^* \mid$ the substring 01 occurs in w as often as 10$\}$ context sensitive? (E.g. 010 and 101, 1001 belong to L, but 0101 does not)

(d) Give a characteristic which all NP complete problems share, but which not necessarily applies to an NP hard problem.

Exercise 3 (12 points).

(a) Construct a nondeterministic TM which accepts the language
$L =_{def} \{w \in \{0,1\}^* \mid$ the substrings 00, 01,10 and 11 all occur in $w\}$.
So 11001 $\in$ L, 01100 $\in$ L, 1010011 $\in$ L, 0001011 $\in$ L.
But 01001 $\notin$ L (since 11 is missing), 000011 $\notin$ L (since 10 is missing),
01010 $\notin$ L (since 00 and 11 are both missing)

(b) What is the order of magnitude of the time complexity of your algorithm?

Exercise 4 (18 points).

In this exercise we consider Turing machines with input alphabet $\Sigma = \{0,1\}$.
$R(M) \in \Sigma^*$ is the representation of M by a sequence of 0's and 1's.
$L(M) \subseteq \Sigma^*$ is the language accepted by M.

We consider the following 3 languages (subsets of Σ^*):
HALT $=_{def}$ {R(M)w | M halts with input w};
INHABITED $=_{def}$ {R(M) | L(M) is not empty};
EMPTY $=_{def}$ {R(M) | L(M) is empty}.

(a) Show that INHABITED is a recursively enumerable language.

(b) Give a reduction of EMPTY to the complement of HALT.

(c) Is EMPTY recursively enumerable?

Do not forget to explain your answers!

Exercise 5 (16 points).

(a) Give an example of a language L such that L $\in$ NP, but L $\notin$ NPC (the set of NP-complete problems)

(b) Let L $\in$ P. What can you say about L* $\in$ P?
By L* we mean the language with the following two properties:
L $\subseteq$ L*, and
For all $w \in$ L*, $v \in$ L the concatenation $wv \in$ L*

Do not forget to explain your answers!

Exercise 6 (12 points).

We consider the "erase-only" variant of the two tape linear bounded Turing Machine concept.

An "erase-only" TM is like an ordinary TM in every respect but one. The "write" options of a erase-only machine are (very) limited.

The erase only machine has a distinguished symbol $ in its tape alphabet which stands for "hasBeenErased". The $ is not in the input alphabet.

At a blank square , the erase only machine can write every symbol of its tape alphabet, except the $.

At all other squares (non-blanks) the erase only machine either leaves the contents unchanged (it replaces the current symbol by itself), or it erases the current contents, by writing $, the "hasBeenErased" symbol.

Show that computations on a two-tape linear bounded Turing Machine M which is of the "erase-only" type, have time complexity $tc_M(n) \in O(n^3)$.

(Of course computations of this M halt for every input.)