UNIVERSITY OF EDINBURGH

FACULTY OF SCIENCE AND ENGINEERING

COMPUTER SCIENCE THREE

COMPUTABILITY AND INTRACTABILITY

Thursday 19 September 2002

09:30 to 11:00

Examiners:

Prof. K. Turner (External)

Prof. C. Stirling (Chairman)

INSTRUCTIONS TO CANDIDATES

Answer any TWO questions.

All questions carry equal weight.

1. Parts (a) and (b) of this question are independent and can be attempted in either order.

   (a) This part is concerned with the properties of finite directed graphs that can be expressed by formulae of first order logic.

      i. Let ϕ be a formula of first order logic that expresses a property of finite directed graphs. Prove that the problem ϕ-GRAPHS is in P, i.e., there is a polynomial time algorithm that can decide for any finite directed graph G if it is a model of ϕ. *[8 marks]*

      ii. Consider the formula

      $$\phi = \forall x\, \neg E(x,x) \wedge \exists x\ (\forall y\, \forall z\, (E(y,z) \Rightarrow y = x) \wedge \forall y\, (x = y \vee E(x,y)))\,.$$

      Describe clearly the graphs that are models of ϕ. Give an example of a graph that is a model of ϕ, and one that is not. *[5 marks]*

   (b) For this part suppose that we have a general system of computation expressed textually (i.e., as programs) over a fixed alphabet. Let $P_0, P_1, P_2, \ldots$ be an effective enumeration of all (single input) programs. For simplicity we assume that all programs take as input a single natural number and if they return a result then that is also a natural number.

   Recall that a partial function from $\mathbb{N}$ to $\mathbb{N}$ is a function that takes a natural number as argument and is either undefined for that number or returns a natural number as result.

   Let f and g be partial functions from $\mathbb{N}$ to $\mathbb{N}$. We say that g *extends* f if for all natural numbers n such that $f(n)$ is defined we have that $g(n)$ is also defined and $g(n) = f(n)$. (In other words g might be defined at more arguments than f but whenever f is defined then so is g and it agrees with f.)

   Define the function f by

   $$f(n) = \begin{cases} P_n(n) + 1, & \text{if } P_n \text{ returns a result on input } n, \\ \text{undefined}, & \text{otherwise.} \end{cases}$$

      i. Do you think that f is computable? Give a brief justification of your answer. *[3 marks]*

      ii. Let g be any computable function that extends f. Prove that g is not total, i.e., there is at least one natural number m such that $g(m)$ is not defined. *[5 marks]*

      iii. Define a total function that is not computable and explain why this is the case. *[4 marks]*

2. (a) Explain what it means for a language L to be (i) *recursive*, and (ii) *recursively enumerable* (r.e.). [*4 marks*]

(b) Prove that a language L is recursive if and only if L and its complement $\overline{L}$ are both recursively enumerable. [*8 marks*]

(c) Explain what it means for language $L_1 \subseteq \Sigma_1^*$ to be *reducible to* language $L_2 \subseteq \Sigma_2^*$. [*4 marks*]

(d) Let

$$L_{\mathrm{ne}} = \{\langle M\rangle \mid L(M) \neq \emptyset\}$$

be the language of (encodings of) Turing machines that accept *some* input.

i. Prove that L_{ne} is *not* recursive.
[**Hint:** Use a reduction from L_{halt}; you may assume without proof that L_{halt} is not recursive.] [*6 marks*]

ii. It can be shown that the language L_{ne} *is* recursively enumerable. Is the language

$$L_{\mathrm{e}} = \{\langle M\rangle \mid L(M) = \emptyset\}$$

recurively enumerable? Justify your answer. [*3 marks*]

3. (a) Define what it means for a language to be *NP-complete.* [*4 marks*]

(b) Consider the following problem, DomSet:

> Instance: An undirected graph $G = (V, E)$.
> Question: Does G contain a dominating set of size k ? (A subset U of V is a *dominating set* for G if and only if every non-isolated vertex in $V - U$ is adjacent to some vertex in U. An isolated vertex is one that has no edges incident at it.)

Suggest a possible 'real-life' application of this problem. [*3 marks*]

(c) Give an informal demonstration that DomSet is in NP. [*5 marks*]

(d) Recall the Vertex Cover problem, VC, defined as follows:

> Instance: An undirected graph G and an integer k.
> Question: Does G possess a vertex cover of size k? (A subset U of the vertex set of G is a *vertex cover* for G if and only if every edge of G has at least one endpoint in U.)

Let (G, k) be an instance of VC, and consider the graph $\widetilde{G}$ constructed by adding vertices and edges to G as follows: for each edge $e = \{u, v\}$ of G, add one new vertex w_e plus the edges $\{u, w_e\}$ and $\{v, w_e\}$. Thus, more formally, $\widetilde{G}$ is the graph $(\widetilde{V}, \widetilde{E})$, where

$$\widetilde{V} = V \ \cup \ \{w_e \mid e \in E\};$$
$$\widetilde{E} = E \ \cup \ \{\{u, w_e\}, \{v, w_e\} \mid e \in E\}.$$

Verify that the function f mapping (G, k) to $(\widetilde{G}, k)$ is a polynomial time reduction from VC to DomSet. [*10 marks*]

(e) Recall that VC is NP-complete. Use this to deduce that DomSet is also NP-complete. [*3 marks*]