Examination, Computability (DAT415/DIT311/DIT312/TDA184)

- Date and time: 2021-01-13, 8:30–12:30.
- Examiner: Nils Anders Danielsson. Telephone number: +46-31 772 1680. Email address: nad@cse.gu.se.
- The GU grades Pass (G) and Pass with Distinction (VG) correspond to the Chalmers grades 3 and 5, respectively.
- To get grade n on the exam you have to be awarded grade n or higher on at least n exercises.
- A completely correct solution of one exercise is awarded the grade 5. Solutions with minor mistakes *might* get the grade 5, and solutions with larger mistakes might get lower grades.
- Exercises can contain parts and/or requirements that are only required for a certain grade. To get grade n on such an exercise you have to get grade n or higher on every part marked with grade n or lower (and every unmarked part), and you have to fulfil every requirement marked with grade n or lower (as well as every unmarked requirement).
- Answers have to be saved to files in one of the following formats: PDF, JPEG or TXT. Use one file per question. Submit your solutions to Canvas before the deadline. Note that there is a separate Canvas assignment for each of the six questions. If Canvas is not working properly, send the solutions to the examiner using email, and include the course code in the subject header.
- Solutions can be rejected if they are hard to read (for instance if a picture is out of focus), unstructured, or poorly motivated.
- **No collaboration is permitted, you have to work on your own.**
- If you want to discuss the grading of the exam, contact the examiner no later than three weeks after the result has been reported.

1. (a) *For grade 3:* Give examples of sets A and B for which $A \to B$ is countable, whereas $B \to A$ is not. You do not need to provide proofs.

   (b) *For grade 4:* Either prove that the set

   $$(\mathbb{N} \to \mathbb{N}) \to \{\, 0, 1 \,\}$$

   is countable, or that it is not countable. You can use theorems from the lecture slides without providing proofs for them.

2. Give concrete syntax for the χ expression e for which the standard χ encoding (as presented in the lectures), given using concrete syntax, is

   $$\begin{array}{l}\ulcorner\, e \,\urcorner = \mathsf{Case}(\mathsf{Const}(\mathsf{Zero}(), \mathsf{Nil}()),\\ \qquad\qquad \mathsf{Cons}(\mathsf{Branch}(\mathsf{Zero}(),\\ \qquad\qquad\qquad\qquad \mathsf{Cons}(\mathsf{Suc}(\mathsf{Zero}()), \mathsf{Nil}()),\\ \qquad\qquad\qquad\qquad \mathsf{Var}(\mathsf{Suc}(\mathsf{Zero}()))),\\ \qquad\qquad\qquad \mathsf{Nil}())).\end{array}$$

   Assume that the number 0 corresponds to the constructor True, and that the number 1 corresponds to the variable x.

3. Is the following partial function χ-decidable?

   $$\begin{array}{l} f \in \mathit{CExp} \times \mathit{CExp} \rightharpoonup \mathit{Bool}\\ f\,(e_1, e_2) = \textbf{if } [\![\mathsf{apply}\ e_1\ e_2]\!] = \ulcorner\, \mathsf{false} \,\urcorner \textbf{ then } \mathsf{true} \textbf{ else } \mathsf{false}\end{array}$$

   For grade 3: Motivate your answer.

   For grade 4: Provide a proof. You are allowed to make use of Rice's theorem, the fact that the halting problem is undecidable, the fact that the *eval* function from the lectures (the χ self-interpreter) is computable, and the fact that equality of closed χ expressions is computable, but not other results stating that some function is or is not computable (unless you provide proofs).

   For grade 5: You may not use Rice's theorem (unless you provide a proof).

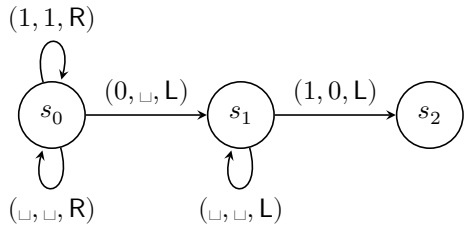
4. Is the following partial function χ-decidable?

   $$\begin{array}{l} f \in \mathit{CExp} \times \mathit{CExp} \rightharpoonup \mathit{Bool}\\ f\,(e_1, e_2) = \textbf{if } [\![\mathsf{apply}\ e_1\ e_2]\!] = \ulcorner\, \mathsf{false} \,\urcorner \textbf{ then } \mathsf{true} \textbf{ else } \text{undefined}\end{array}$$

   The grade criteria of the previous exercise apply to this one as well.

5. Consider the following Turing machine:

   - Input alphabet: $\{ 0, 1 \}$.
   - Tape alphabet: $\{ 0, 1, \sqcup \}$.
   - States: $\{ s_0, s_1, s_2 \}$.
   - Initial state: s_0.
   - Transition function:



   (a) *For grade 3:* What is the result of running this Turing machine with 111 as the input string? Does it halt? In that case, what is the resulting string?

   (b) *For grade 4:* Let us represent natural numbers (0, 1, 2…) in the following way: the number $n \in \mathbb{N}$ is represented by a string with n ones followed by one zero (1^n0). Does this Turing machine witness the Turing-computability of some total function from $\mathbb{N}$ to $\mathbb{N}$? In either case you should provide a proof. If the answer is yes, then you should additionally give a simple description of the function that is witnessed, without any reference to Turing machines (no proof is needed for this part).

6. Can one ensure that every PRF-computable function $f \in \mathbb{N} \to \mathbb{N}$ is increasing by removing exactly one of the constructors of the abstract syntax of PRF (zero, suc, proj, comp or rec)? Either prove that this is possible, or that it is not possible. Note that the natural number constructors zero and suc should not be removed, even if the PRF constructors with the same names are removed.

   A function $f \in \mathbb{N} \to \mathbb{N}$ is increasing if $f\,n \geq n$ for every input $n \in \mathbb{N}$, and a function $g \in \mathbb{N} \to \mathbb{N}$ is PRF-computable if there is a term $\underline{g} \in PRF_1$ such that $\forall n \in \mathbb{N}.\, [\![\underline{g}]\!]\ (\mathsf{nil}, n) = g\ n$.

   For reference, here is the abstract syntax of PRF:

$$\frac{}{\mathsf{zero} \in PRF_0} \qquad \frac{}{\mathsf{suc} \in PRF_1} \qquad \frac{i \in \mathbb{N} \qquad 0 \leq i < n}{\mathsf{proj}\ i \in PRF_n}$$

$$\frac{f \in PRF_m \qquad gs \in (PRF_n)^m}{\mathsf{comp}\ f\ gs \in PRF_n} \qquad \frac{f \in PRF_n \qquad g \in PRF_{2+n}}{\mathsf{rec}\ f\ g \in PRF_{1+n}}$$

   The denotational semantics is defined in the following way (for any $m, n \in \mathbb{N}$):

$$\begin{array}{l}
[\![_]\!] \in PRF_n \to (\mathbb{N}^n \to \mathbb{N}) \\
[\![\mathsf{zero}]\!]\ \mathsf{nil} = 0 \\
[\![\mathsf{suc}]\!]\ (\mathsf{nil}, n) = 1 + n \\
[\![\mathsf{proj}\ i]\!]\ \rho = \mathit{index}\ \rho\ i \\
[\![\mathsf{comp}\ f\ gs]\!]\ \rho = [\![f]\!]\ ([\![gs]\!]\star\ \rho) \\
[\![\mathsf{rec}\ f\ g]\!]\ (\rho, \mathsf{zero}) = [\![f]\!]\ \rho \\
[\![\mathsf{rec}\ f\ g]\!]\ (\rho, \mathsf{suc}\ n) = [\![g]\!]\ (\rho, n, [\![\mathsf{rec}\ f\ g]\!]\ (\rho, n)) \\
[\![_]\!]\star \in (PRF_m)^n \to (\mathbb{N}^m \to \mathbb{N}^n) \\
[\![\mathsf{nil}]\!]\star\ \rho = \mathsf{nil} \\
[\![fs, f]\!]\star\ \rho = [\![fs]\!]\star\ \rho, [\![f]\!]\ \rho
\end{array}$$

   The *index* function is defined as follows (for any set A and $n \in \mathbb{N}$):

$$\begin{array}{l}
\mathit{index} \in A^n \to \{\, i \in \mathbb{N} \mid 0 \leq i < n \,\} \to A \\
\mathit{index}\ (xs, x)\ \mathsf{zero} = x \\
\mathit{index}\ (xs, x)\ (\mathsf{suc}\ i) = \mathit{index}\ xs\ i
\end{array}$$