Sample solutions for the examination of Models of Computation (DIT310/DIT311/TDA184) from 2019-01-16

Nils Anders Danielsson

1. (a) $A = \emptyset$, $B = \mathbb{N} \to \mathbb{N}$.

   (b) It is not countable.

   Note first that $\{ 0 \} \to \mathbb{N}$ is in bijective correspondence with $\mathbb{N}$: the functions $\lambda f.\, f\, 0 \in (\{ 0 \} \to \mathbb{N}) \to \mathbb{N}$ and $\lambda\, n.\, \lambda\, \_.\, n \in \mathbb{N} \to (\{ 0 \} \to \mathbb{N})$ are inverses of each other. Thus $(\{ 0 \} \to \mathbb{N}) \to (\{ 0 \} \to \mathbb{N})$ is in bijective correspondence with $\mathbb{N} \to \mathbb{N}$, which is not countable. We can conclude that $(\{ 0 \} \to \mathbb{N}) \to (\{ 0 \} \to \mathbb{N})$ is not countable, because if an uncountable set A is in bijective correspondence with a set B, then B is also uncountable.

2. $\mathbf{case}\ x\ \mathbf{of}\ \{\mathsf{True}(x) \to x\}$.

3. No. This can be proved by reducing $\textit{is-total}_4$ (which is not χ-decidable, see the following exercise) to $\textit{is-total}_3$.

   If $\textit{is-total}_3$ is χ-decidable, then there is a closed χ expression $\underline{\textit{is-total}_3}$ witnessing the computability of $\textit{is-total}_3$. This expression is also a witness of the χ-decidability of $\textit{is-total}_4$, because for any $e \in \textit{Fun}$ we have

$$
\begin{array}{ll}
[\![\underline{\textit{is-total}_3}\ \ulcorner e \urcorner]\!] & = \\
\ulcorner \textit{is-total}_3\ e \urcorner & = \\
\ulcorner\ \mathbf{if}\ \forall\, b_1 \in \textit{Bool}.\ \exists\, b_2 \in \textit{Bool}.\ [\![e\ \ulcorner b_1 \urcorner]\!] = \ulcorner b_2 \urcorner & \\
\quad \mathbf{then}\ \mathsf{true}\ \mathbf{else}\ \mathsf{false}\ \urcorner & = \\
\ulcorner \textit{is-total}_4\ e \urcorner. &
\end{array}
$$

   (Note that $\textit{Fun} \subseteq \textit{CExp}$, and that the encoding function for $\textit{CExp}$ is used also for $\textit{Fun}$.)

4. No. We can prove this by reducing the halting problem (which is not χ-decidable) to $\textit{is-total}_4$.

   If $\textit{is-total}_4$ is χ-decidable, then there is a closed χ expression $\underline{\textit{is-total}_4}$ witnessing the computability of $\textit{is-total}_4$. We can use this expression to construct a closed χ expression $\underline{\mathit{halts}}$ (written using a mixture of concrete syntax and meta-level notation):

$$\underline{\mathit{halts}} = \lambda\, e.\ \underline{\mathit{is\text{-}total}_4}\ \ulcorner\, \lambda\,\_.\ (\lambda\,\_.\ulcorner\, \mathsf{true}\,\urcorner)\ \llcorner\, e\,\lrcorner\,\urcorner.$$

Now note that, for any $e \in \mathit{CExp}$, the closed expression $\lambda\,\_.\ (\lambda\,\_.\ulcorner\, \mathsf{true}\,\urcorner)\ e$ is an element of Fun, i.e. there is some $f \in \mathit{Bool} \rightharpoonup \mathit{Bool}$ such that

$$\forall\ b \in \mathit{Bool}.\ \llbracket(\lambda\,\_.\ (\lambda\,\_.\ulcorner\, \mathsf{true}\,\urcorner)\ e)\ \ulcorner\, b\,\urcorner\rrbracket = \ulcorner\, f\, b\,\urcorner.$$

This holds for f defined by

$$f\, b = \textbf{if}\ \llbracket e\rrbracket \text{ is defined } \textbf{then}\ \mathsf{true}\ \textbf{else}\ \text{undefined},$$

because for any $b \in \mathit{Bool}$ we have

$$\begin{array}{ll}
\llbracket(\lambda\,_.\ (\lambda\,_.\ulcorner\, \mathsf{true}\,\urcorner)\ e)\ \ulcorner\, b\,\urcorner\rrbracket & = \\
\llbracket(\lambda\,_.\ulcorner\, \mathsf{true}\,\urcorner)\ e\rrbracket & = \\
\textbf{if}\ \llbracket e\rrbracket \text{ is defined } \textbf{then}\ \ulcorner\, \mathsf{true}\,\urcorner\ \textbf{else}\ \text{undefined} & = \\
\ulcorner\ \textbf{if}\ \llbracket e\rrbracket \text{ is defined } \textbf{then}\ \mathsf{true}\ \textbf{else}\ \text{undefined}\,\urcorner & = \\
\ulcorner\, f\, b\,\urcorner. &
\end{array}$$

Thus, by the assumption that $\underline{\mathit{is\text{-}total}_4}$ witnesses the computability of $\mathit{is\text{-}total}_4$, we get that

$$\llbracket\underline{\mathit{is\text{-}total}_4}\ \ulcorner\, \lambda\,\_.\ (\lambda\,\_.\ulcorner\, \mathsf{true}\,\urcorner)\ e\,\urcorner\rrbracket = \ulcorner\, \mathit{is\text{-}total}_4\ (\lambda\,\_.\ (\lambda\,\_.\ulcorner\, \mathsf{true}\,\urcorner)\ e)\,\urcorner.$$

Let us now verify that $\underline{\mathit{halts}}$ witnesses the decidability of the halting problem. For any $e \in \mathit{CExp}$ we have

$$\begin{array}{ll}
\llbracket\underline{\mathit{halts}}\ \ulcorner\, e\,\urcorner\rrbracket & = \\
\llbracket\underline{\mathit{is\text{-}total}_4}\ \ulcorner\, \lambda\,_.\ (\lambda\,_.\ulcorner\, \mathsf{true}\,\urcorner)\ e\,\urcorner\rrbracket & = \\
\ulcorner\, \mathit{is\text{-}total}_4\ (\lambda\,_.\ (\lambda\,_.\ulcorner\, \mathsf{true}\,\urcorner)\ e)\,\urcorner & = \\
\textbf{if}\ \forall\ b_1 \in \mathit{Bool}.\ \exists\ b_2 \in \mathit{Bool}.\ \llbracket(\lambda\,_.\ (\lambda\,_.\ulcorner\, \mathsf{true}\,\urcorner)\ e)\ \ulcorner\, b_1\,\urcorner\rrbracket = \ulcorner\, b_2\,\urcorner & \\
\quad \textbf{then}\ \ulcorner\, \mathsf{true}\,\urcorner\ \textbf{else}\ \ulcorner\, \mathsf{false}\,\urcorner & = \\
\textbf{if}\ \exists\ b_2 \in \mathit{Bool}.\ \llbracket(\lambda\,_.\ulcorner\, \mathsf{true}\,\urcorner)\ e\rrbracket = \ulcorner\, b_2\,\urcorner & \\
\quad \textbf{then}\ \ulcorner\, \mathsf{true}\,\urcorner\ \textbf{else}\ \ulcorner\, \mathsf{false}\,\urcorner. &
\end{array}$$

If $\llbracket e\rrbracket$ is defined, then

$$\begin{array}{ll}
\textbf{if}\ \exists\ b_2 \in \mathit{Bool}.\ \llbracket(\lambda\,_.\ulcorner\, \mathsf{true}\,\urcorner)\ e\rrbracket = \ulcorner\, b_2\,\urcorner\ \textbf{then}\ \ulcorner\, \mathsf{true}\,\urcorner\ \textbf{else}\ \ulcorner\, \mathsf{false}\,\urcorner & = \\
\textbf{if}\ \exists\ b_2 \in \mathit{Bool}.\ \llbracket\ulcorner\, \mathsf{true}\,\urcorner\rrbracket = \ulcorner\, b_2\,\urcorner\ \textbf{then}\ \ulcorner\, \mathsf{true}\,\urcorner\ \textbf{else}\ \ulcorner\, \mathsf{false}\,\urcorner & = \\
\textbf{if}\ \exists\ b_2 \in \mathit{Bool}.\ulcorner\, \mathsf{true}\,\urcorner = \ulcorner\, b_2\,\urcorner\ \textbf{then}\ \ulcorner\, \mathsf{true}\,\urcorner\ \textbf{else}\ \ulcorner\, \mathsf{false}\,\urcorner & = \\
\ulcorner\, \mathsf{true}\,\urcorner, &
\end{array}$$

and if $\llbracket e\rrbracket$ is undefined, then

$$\begin{array}{ll}
\textbf{if}\ \exists\ b_2 \in \mathit{Bool}.\ \llbracket(\lambda\,_.\ulcorner\, \mathsf{true}\,\urcorner)\ e\rrbracket = \ulcorner\, b_2\,\urcorner\ \textbf{then}\ \ulcorner\, \mathsf{true}\,\urcorner\ \textbf{else}\ \ulcorner\, \mathsf{false}\,\urcorner & = \\
\textbf{if}\ \exists\ b_2 \in \mathit{Bool}.\ulcorner\, b_2\,\urcorner \text{ is undefined } \textbf{then}\ \ulcorner\, \mathsf{true}\,\urcorner\ \textbf{else}\ \ulcorner\, \mathsf{false}\,\urcorner & = \\
\ulcorner\, \mathsf{false}\,\urcorner. &
\end{array}$$

Thus we get

$$\llbracket \underline{\mathit{halts}} \ulcorner e \urcorner \rrbracket = \ulcorner \textbf{ if } \llbracket e \rrbracket \text{ is defined } \textbf{then } \mathsf{true} \textbf{ else } \mathsf{false} \urcorner,$$

i.e. $\underline{\mathit{halts}}$ witnesses the decidability of the halting problem.

5. (a) The value is 3:

$$\begin{array}{ll}
\llbracket p \rrbracket \ (\mathsf{nil}, 2) & = \\
\llbracket \mathsf{comp\ suc}\ (\mathsf{nil}, \mathsf{proj}\ 1) \rrbracket \ (\mathsf{nil}, \llbracket p \rrbracket \ (\mathsf{nil}, 1), 1) & = \\
\llbracket \mathsf{suc} \rrbracket \ (\llbracket \mathsf{nil}, \mathsf{proj}\ 1 \rrbracket \star \ (\mathsf{nil}, \llbracket p \rrbracket \ (\mathsf{nil}, 1), 1)) & = \\
\llbracket \mathsf{suc} \rrbracket \ (\mathsf{nil}, \llbracket \mathsf{proj}\ 1 \rrbracket \ (\mathsf{nil}, \llbracket p \rrbracket \ (\mathsf{nil}, 1), 1)) & = \\
\llbracket \mathsf{suc} \rrbracket \ (\mathsf{nil}, \llbracket p \rrbracket \ (\mathsf{nil}, 1)) & = \\
1 + \llbracket p \rrbracket \ (\mathsf{nil}, 1) & = \\
1 + \llbracket \mathsf{comp\ suc}\ (\mathsf{nil}, \mathsf{proj}\ 1) \rrbracket \ (\mathsf{nil}, \llbracket p \rrbracket \ (\mathsf{nil}, 0), 0) & = \\
1 + \llbracket \mathsf{suc} \rrbracket \ (\llbracket \mathsf{nil}, \mathsf{proj}\ 1 \rrbracket \star \ (\mathsf{nil}, \llbracket p \rrbracket \ (\mathsf{nil}, 0), 0)) & = \\
1 + \llbracket \mathsf{suc} \rrbracket \ (\mathsf{nil}, \llbracket \mathsf{proj}\ 1 \rrbracket \ (\mathsf{nil}, \llbracket p \rrbracket \ (\mathsf{nil}, 0), 0)) & = \\
1 + \llbracket \mathsf{suc} \rrbracket \ (\mathsf{nil}, \llbracket p \rrbracket \ (\mathsf{nil}, 0)) & = \\
2 + \llbracket p \rrbracket \ (\mathsf{nil}, 0) & = \\
2 + \llbracket \mathsf{comp\ suc}\ (\mathsf{nil}, \mathsf{zero}) \rrbracket \ \mathsf{nil} & = \\
2 + \llbracket \mathsf{suc} \rrbracket \ (\llbracket \mathsf{nil}, \mathsf{zero} \rrbracket \star \ \mathsf{nil}) & = \\
2 + \llbracket \mathsf{suc} \rrbracket \ (\mathsf{nil}, \llbracket \mathsf{zero} \rrbracket \ \mathsf{nil}) & = \\
3 + \llbracket \mathsf{zero} \rrbracket \ \mathsf{nil} & = \\
3 + 0 & = \\
3. &
\end{array}$$

(b) The function takes n to $1 + n$.

(c) Let us prove by induction on $n \in \mathbb{N}$ that $\llbracket p \rrbracket \ (\mathsf{nil}, n) = 1 + n$.

- $n = \mathsf{zero}$: We have

$$\begin{array}{ll}
\llbracket p \rrbracket \ (\mathsf{nil}, n) & = \\
\llbracket p \rrbracket \ (\mathsf{nil}, \mathsf{zero}) & = \\
\llbracket \mathsf{comp\ suc}\ (\mathsf{nil}, \mathsf{zero}) \rrbracket \ \mathsf{nil} & = \\
\llbracket \mathsf{suc} \rrbracket \ (\llbracket \mathsf{nil}, \mathsf{zero} \rrbracket \star \ \mathsf{nil}) & = \\
\llbracket \mathsf{suc} \rrbracket \ (\mathsf{nil}, \llbracket \mathsf{zero} \rrbracket \ \mathsf{nil}) & = \\
1 + \llbracket \mathsf{zero} \rrbracket \ \mathsf{nil} & = \\
1 & = \\
1 + \mathsf{zero} & = \\
1 + n. &
\end{array}$$

- $n = \mathsf{suc}\ n'$ for some $n' \in \mathbb{N}$: The inductive hypothesis tells us that $\llbracket p \rrbracket \ (\mathsf{nil}, n') = 1 + n'$. We get

$$\begin{array}{ll}
\llbracket p \rrbracket \ (\mathsf{nil}, n) & = \\
\llbracket p \rrbracket \ (\mathsf{nil}, \mathsf{suc}\ n') & = \\
\llbracket \mathsf{comp\ suc}\ (\mathsf{nil}, \mathsf{proj}\ 1) \rrbracket \ (\mathsf{nil}, \llbracket p \rrbracket \ (\mathsf{nil}, n'), n') & =
\end{array}$$

$$
\begin{array}{ll}
[\![\mathsf{suc}]\!]\ ([\![\mathsf{nil}, \mathsf{proj}\ 1]\!]\star\ (\mathsf{nil}, [\![p]\!]\ (\mathsf{nil}, n'), n')) & = \\
[\![\mathsf{suc}]\!]\ (\mathsf{nil}, [\![\mathsf{proj}\ 1]\!]\ (\mathsf{nil}, [\![p]\!]\ (\mathsf{nil}, n'), n')) & = \\
[\![\mathsf{suc}]\!]\ (\mathsf{nil}, [\![p]\!]\ (\mathsf{nil}, n')) & = \\
1 + [\![p]\!]\ (\mathsf{nil}, n') & = \\
1 + (1 + n') & = \\
1 + n. &
\end{array}
$$

6. No. The total function that maps every natural number to zero cannot be implemented. In particular, if the input is $\ulcorner 1 \urcorner = 10$, then it is impossible to produce the output $\ulcorner 0 \urcorner = 0$, because the head cannot move to the second square and write a blank.