CS 430: Formal Semantics Assignment 3 Sample Solution

Prepared by: Shu-Chun Weng

Reynolds, exercise 5.1 (a, b)

$$[\![c]\!]^{\text{direct}}[\mathtt{x} : m] = \begin{cases} [\mathtt{x} : 0] & \text{if } m \geq 0 \text{ and } m \text{ is even} \\ \langle \mathbf{abort}, [\mathtt{x} : 1] \rangle & \text{if } m > 0 \text{ and } m \text{ is odd} \\ \bot & \text{otherwise} \end{cases}$$

$$[\![c]\!]^{\text{cont}}\kappa_t\kappa_f[\mathtt{x} : m] = \begin{cases} \kappa_t[\mathtt{x} : 0] & \text{if } m \geq 0 \text{ and } m \text{ is even} \\ \kappa_f[\mathtt{x} : 1] & \text{if } m > 0 \text{ and } m \text{ is odd} \\ \bot & \text{otherwise} \end{cases}$$

Reynolds, exercise 5.3

$$[\![\mathbf{catchin}\ c_0\ \mathbf{with}\ c_1]\!]^{\text{direct}}\sigma = \begin{cases} \bot & \text{if } [\![c_0]\!]\ \sigma = \bot \\ \sigma' & \text{if } \sigma' = [\![c_0]\!]\ \sigma \neq \bot \\ [\![c_1]\!]\ \sigma' & \text{if } \langle \mathbf{abort}, \sigma' \rangle = [\![c_0]\!]\ \sigma \end{cases}$$

$$[\![\mathbf{catchin}\ c_0\ \mathbf{with}\ c_1]\!]^{\text{cont}}\kappa_t\kappa_f\sigma = [\![c_0]\!]\ \kappa_t(\lambda\sigma' \in \Sigma.\ [\![c_1]\!]\ \kappa_t\kappa_f\sigma')\sigma$$

Reynolds, exercise 5.4(b)

$$\begin{aligned} [\![\mathbf{newvar}\ v := e\ \mathbf{in}\ c]\!]\ \kappa_t\kappa_f\sigma &= [\![c]\!]\ (\lambda\sigma' \in \Sigma.\ \kappa_t[\sigma' \mid v : \sigma v]) \\ &\quad (\lambda l \in \langle \text{label} \rangle.\ \lambda\sigma' \in \Sigma.\ \kappa_f\ l\ [\sigma' \mid v : \sigma v]) \\ &\quad [\sigma \mid v : [\![e]\!]_{\text{intexp}}\sigma] \\ [\![\mathbf{fail}\ l]\!]\ \kappa_t\kappa_f\sigma &= \kappa_f\ l\ \sigma \\ [\![\mathbf{catch}\ l\ \mathbf{in}\ c_0\ \mathbf{with}\ c_1]\!]\ \kappa_t\kappa_f\sigma &= [\![c_0]\!]\ \kappa_t \\ &\quad (\lambda l' \in \langle \text{label} \rangle.\ \lambda\sigma' \in \Sigma. \\ &\quad\quad \text{if } l = l' \text{ then } [\![c_1]\!]\ \kappa_t\kappa_f\sigma' \text{ else } \kappa_f\ l'\ \sigma') \\ &\quad \sigma \end{aligned}$$

And the rest remains the same.

Problem 4

Problem statement. Show that either the direct or the continuation denotational semantics (of a language without fail) does not distinguish the following commands:

$$\mathbf{if}\ \mathtt{x} = 0\ \mathbf{then}\ ?\mathtt{y}; !\mathtt{y} + 1\ \mathbf{else}\ ?\mathtt{y}; !\mathtt{y} + 2$$

and

$$?\mathtt{y}; \mathbf{if}\ \mathtt{x} = 0\ \mathbf{then}\ !\mathtt{y} + 1\ \mathbf{else}\ !\mathtt{y} + 2$$

Let

$$\begin{array}{rcl} c_1 & = & \textbf{if } \mathtt{x} = 0 \textbf{ then } ?\mathtt{y}; !\mathtt{y}+1 \textbf{ else } ?\mathtt{y}; !\mathtt{y}+2 \\ c_2 & = & ?\mathtt{y}; \textbf{if } \mathtt{x} = 0 \textbf{ then } !\mathtt{y}+1 \textbf{ else } !\mathtt{y}+2 \end{array}$$

For direct semantics

$$\begin{array}{ll} & [\![c_1]\!]^{\mathrm{direct}}\sigma \\ = & \begin{cases} [\![?\mathtt{y}; !\mathtt{y}+1]\!]^{\mathrm{direct}}\sigma & \text{if } \sigma\mathtt{x} = 0 \\ [\![?\mathtt{y}; !\mathtt{y}+2]\!]^{\mathrm{direct}}\sigma & \text{if } \sigma\mathtt{x} \neq 0 \end{cases} \\ = & \begin{cases} [\![!\mathtt{y}+1]\!]^{\mathrm{direct}}_*([\![?\mathtt{y}]\!]^{\mathrm{direct}}\sigma) & \text{if } \sigma\mathtt{x} = 0 \\ [\![!\mathtt{y}+2]\!]^{\mathrm{direct}}_*([\![?\mathtt{y}]\!]^{\mathrm{direct}}\sigma) & \text{if } \sigma\mathtt{x} \neq 0 \end{cases} \\ = & \begin{cases} [\![!\mathtt{y}+1]\!]^{\mathrm{direct}}_*(\imath_{\mathrm{in}}(\lambda k \in \mathbf{Z}.\ \imath_{\mathrm{term}}[\sigma \mid \mathtt{y} : k])) & \text{if } \sigma\mathtt{x} = 0 \\ [\![!\mathtt{y}+2]\!]^{\mathrm{direct}}_*(\imath_{\mathrm{in}}(\lambda k \in \mathbf{Z}.\ \imath_{\mathrm{term}}[\sigma \mid \mathtt{y} : k])) & \text{if } \sigma\mathtt{x} \neq 0 \end{cases} \\ = & \begin{cases} \imath_{\mathrm{in}}(\lambda k \in \mathbf{Z}.\ [\![!\mathtt{y}+1]\!]^{\mathrm{direct}}[\sigma \mid \mathtt{y} : k]) & \text{if } \sigma\mathtt{x} = 0 \\ \imath_{\mathrm{in}}(\lambda k \in \mathbf{Z}.\ [\![!\mathtt{y}+2]\!]^{\mathrm{direct}}[\sigma \mid \mathtt{y} : k]) & \text{if } \sigma\mathtt{x} \neq 0 \end{cases} \\ = & \begin{cases} \imath_{\mathrm{in}}(\lambda k \in \mathbf{Z}.\ \imath_{\mathrm{out}}(k+1, \imath_{\mathrm{term}}[\sigma \mid \mathtt{y} : k])) & \text{if } \sigma\mathtt{x} = 0 \\ \imath_{\mathrm{in}}(\lambda k \in \mathbf{Z}.\ \imath_{\mathrm{out}}(k+2, \imath_{\mathrm{term}}[\sigma \mid \mathtt{y} : k])) & \text{if } \sigma\mathtt{x} \neq 0 \end{cases} \end{array}$$

$$\begin{array}{ll} & [\![c_2]\!]^{\mathrm{direct}}\sigma \\ = & [\![\textbf{if } \mathtt{x} = 0 \textbf{ then } !\mathtt{y}+1 \textbf{ else } !\mathtt{y}+2]\!]^{\mathrm{direct}}_*([\![?\mathtt{y}]\!]^{\mathrm{direct}}\sigma) \\ = & [\![\textbf{if } \mathtt{x} = 0 \textbf{ then } !\mathtt{y}+1 \textbf{ else } !\mathtt{y}+2]\!]^{\mathrm{direct}}_*(\imath_{\mathrm{in}}(\lambda k \in \mathbf{Z}.\ \imath_{\mathrm{term}}[\sigma \mid \mathtt{y} : k])) \\ = & \imath_{\mathrm{in}}(\lambda k \in \mathbf{Z}.\ \begin{cases} [\![!\mathtt{y}+1]\!]^{\mathrm{direct}}_*(\imath_{\mathrm{term}}[\sigma \mid \mathtt{y} : k]) & \text{if } [\sigma \mid \mathtt{y} : k]\mathtt{x} = 0 \\ [\![!\mathtt{y}+2]\!]^{\mathrm{direct}}_*(\imath_{\mathrm{term}}[\sigma \mid \mathtt{y} : k]) & \text{if } [\sigma \mid \mathtt{y} : k]\mathtt{x} \neq 0 \end{cases}) \\ = & \imath_{\mathrm{in}}(\lambda k \in \mathbf{Z}.\ \begin{cases} \imath_{\mathrm{out}}(k+1, \imath_{\mathrm{term}}[\sigma \mid \mathtt{y} : k]) & \text{if } [\sigma \mid \mathtt{y} : k]\mathtt{x} = 0 \\ \imath_{\mathrm{out}}(k+2, \imath_{\mathrm{term}}[\sigma \mid \mathtt{y} : k]) & \text{if } [\sigma \mid \mathtt{y} : k]\mathtt{x} \neq 0 \end{cases}) \end{array}$$

But since the extension of σ on $\mathtt{y}$ does not effects the value $\mathtt{x}$ maps to, the predicates are equivalent to testing if $\sigma\mathtt{x} = 0$ or not.

$$\begin{array}{ll} = & \imath_{\mathrm{in}}(\lambda k \in \mathbf{Z}.\ \begin{cases} \imath_{\mathrm{out}}(k+1, \imath_{\mathrm{term}}[\sigma \mid \mathtt{y} : k]) & \text{if } \sigma\mathtt{x} = 0 \\ \imath_{\mathrm{out}}(k+2, \imath_{\mathrm{term}}[\sigma \mid \mathtt{y} : k]) & \text{if } \sigma\mathtt{x} \neq 0 \end{cases}) \\ = & \begin{cases} \imath_{\mathrm{in}}(\lambda k \in \mathbf{Z}.\ \imath_{\mathrm{out}}(k+1, \imath_{\mathrm{term}}[\sigma \mid \mathtt{y} : k])) & \text{if } \sigma\mathtt{x} = 0 \\ \imath_{\mathrm{in}}(\lambda k \in \mathbf{Z}.\ \imath_{\mathrm{out}}(k+2, \imath_{\mathrm{term}}[\sigma \mid \mathtt{y} : k])) & \text{if } \sigma\mathtt{x} \neq 0 \end{cases} \\ = & [\![c_1]\!]^{\mathrm{direct}} \end{array}$$

For continuation semantics:

$$
\begin{array}{ll}
& [\![c_1]\!]^{\mathrm{cont}}\kappa\sigma \\
= & \begin{cases} [\![?\mathtt{y}; !\mathtt{y}+1]\!]^{\mathrm{cont}}\kappa\sigma & \text{if } \sigma\mathtt{x}=0 \\ [\![?\mathtt{y}; !\mathtt{y}+2]\!]^{\mathrm{cont}}\kappa\sigma & \text{if } \sigma\mathtt{x}\neq 0 \end{cases} \\
= & \begin{cases} [\![?\mathtt{y}]\!]^{\mathrm{cont}}([\![!\mathtt{y}+1]\!]^{\mathrm{cont}}\kappa)\sigma & \text{if } \sigma\mathtt{x}=0 \\ [\![?\mathtt{y}]\!]^{\mathrm{cont}}([\![!\mathtt{y}+2]\!]^{\mathrm{cont}}\kappa)\sigma & \text{if } \sigma\mathtt{x}\neq 0 \end{cases} \\
= & \begin{cases} \imath_{\mathrm{in}}(\lambda k \in \mathbf{Z}.\ [\![!\mathtt{y}+1]\!]^{\mathrm{cont}}\kappa[\sigma \mid \mathtt{y} : k]) & \text{if } \sigma\mathtt{x}=0 \\ \imath_{\mathrm{in}}(\lambda k \in \mathbf{Z}.\ [\![!\mathtt{y}+2]\!]^{\mathrm{cont}}\kappa[\sigma \mid \mathtt{y} : k]) & \text{if } \sigma\mathtt{x}\neq 0 \end{cases} \\
= & \begin{cases} \imath_{\mathrm{in}}(\lambda k \in \mathbf{Z}.\ \imath_{\mathrm{out}}(k+1, \kappa[\sigma \mid \mathtt{y} : k])) & \text{if } \sigma\mathtt{x}=0 \\ \imath_{\mathrm{in}}(\lambda k \in \mathbf{Z}.\ \imath_{\mathrm{out}}(k+2, \kappa[\sigma \mid \mathtt{y} : k])) & \text{if } \sigma\mathtt{x}\neq 0 \end{cases}
\end{array}
$$

$$
\begin{array}{ll}
& [\![c_2]\!]^{\mathrm{cont}}\kappa\sigma \\
= & [\![?\mathtt{y}]\!]^{\mathrm{cont}}([\![\textbf{if } \mathtt{x}=0 \textbf{ then } !\mathtt{y}+1 \textbf{ else } !\mathtt{y}+2]\!]^{\mathrm{cont}}\kappa)\sigma \\
= & \imath_{\mathrm{in}}(\lambda k \in \mathbf{Z}.\ [\![\textbf{if } \mathtt{x}=0 \textbf{ then } !\mathtt{y}+1 \textbf{ else } !\mathtt{y}+2]\!]^{\mathrm{cont}}\kappa[\sigma \mid \mathtt{y} : k]) \\
= & \imath_{\mathrm{in}}(\lambda k \in \mathbf{Z}.\ \begin{cases} [\![!\mathtt{y}+1]\!]^{\mathrm{cont}}\kappa[\sigma \mid \mathtt{y} : k] \\ [\![!\mathtt{y}+2]\!]^{\mathrm{cont}}\kappa[\sigma \mid \mathtt{y} : k] \end{cases}) \quad \begin{array}{l} \text{if } [\sigma \mid \mathtt{y} : k]\mathtt{x}=0 \\ \text{if } [\sigma \mid \mathtt{y} : k]\mathtt{x}\neq 0 \end{array} \\
= & \imath_{\mathrm{in}}(\lambda k \in \mathbf{Z}.\ \begin{cases} \imath_{\mathrm{out}}(k+1, \kappa[\sigma \mid \mathtt{y} : k]) \\ \imath_{\mathrm{out}}(k+2, \kappa[\sigma \mid \mathtt{y} : k]) \end{cases}) \quad \begin{array}{l} \text{if } [\sigma \mid \mathtt{y} : k]\mathtt{x}=0 \\ \text{if } [\sigma \mid \mathtt{y} : k]\mathtt{x}\neq 0 \end{array}
\end{array}
$$

Similarly since the extension of σ on $\mathtt{y}$ does not effects the value $\mathtt{x}$ maps to, the predicates are equivalent to testing if $\sigma\mathtt{x}=0$ or not.

$$
\begin{array}{ll}
= & \imath_{\mathrm{in}}(\lambda k \in \mathbf{Z}.\ \begin{cases} \imath_{\mathrm{out}}(k+1, \kappa[\sigma \mid \mathtt{y} : k]) \\ \imath_{\mathrm{out}}(k+2, \kappa[\sigma \mid \mathtt{y} : k]) \end{cases}) \quad \begin{array}{l} \text{if } \sigma\mathtt{x}=0 \\ \text{if } \sigma\mathtt{x}\neq 0 \end{array} \\
= & \begin{cases} \imath_{\mathrm{in}}(\lambda k \in \mathbf{Z}.\ \imath_{\mathrm{out}}(k+1, \kappa[\sigma \mid \mathtt{y} : k])) & \text{if } \sigma\mathtt{x}=0 \\ \imath_{\mathrm{in}}(\lambda k \in \mathbf{Z}.\ \imath_{\mathrm{out}}(k+2, \kappa[\sigma \mid \mathtt{y} : k])) & \text{if } \sigma\mathtt{x}\neq 0 \end{cases} \\
= & [\![c_1]\!]^{\mathrm{cont}}
\end{array}
$$

Reynolds, exercise 6.1(a)

$$
\begin{array}{ll}
& \langle \textbf{while } \mathtt{x}\neq 0 \textbf{ do if } \mathtt{x}=1 \textbf{ then fail else } \mathtt{x} := \mathtt{x}-2, [\mathtt{x} : 3]\rangle \\
\longrightarrow & \langle \textbf{if } \mathtt{x}=1 \textbf{ then fail else } \mathtt{x} := \mathtt{x}-2; \textbf{while } \mathtt{x}\neq 0 \textbf{ do if } \mathtt{x}=1 \textbf{ then fail else } \mathtt{x} := \mathtt{x}-2, [\mathtt{x} : 3]\rangle \\
\longrightarrow & \langle \mathtt{x} := \mathtt{x}-2; \textbf{while } \mathtt{x}\neq 0 \textbf{ do if } \mathtt{x}=1 \textbf{ then fail else } \mathtt{x} := \mathtt{x}-2, [\mathtt{x} : 3]\rangle \\
\longrightarrow & \langle \textbf{while } \mathtt{x}\neq 0 \textbf{ do if } \mathtt{x}=1 \textbf{ then fail else } \mathtt{x} := \mathtt{x}-2, [\mathtt{x} : 1]\rangle \\
\longrightarrow & \langle \textbf{if } \mathtt{x}=1 \textbf{ then fail else } \mathtt{x} := \mathtt{x}-2; \textbf{while } \mathtt{x}\neq 0 \textbf{ do if } \mathtt{x}=1 \textbf{ then fail else } \mathtt{x} := \mathtt{x}-2, [\mathtt{x} : 1]\rangle \\
\longrightarrow & \langle \textbf{fail}; \textbf{while } \mathtt{x}\neq 0 \textbf{ do if } \mathtt{x}=1 \textbf{ then fail else } \mathtt{x} := \mathtt{x}-2, [\mathtt{x} : 1]\rangle \\
\longrightarrow & \langle \textbf{abort}, [\mathtt{x} : 1]\rangle
\end{array}
$$

Reynolds, exercise 6.2

(a)

$$\langle \textbf{repeat } c \textbf{ until } e, \sigma\rangle \longrightarrow \langle c; \textbf{if } e \textbf{ then skip else repeat } c \textbf{ until } e, \sigma\rangle$$

(b)

$$\frac{\langle c_0, \sigma\rangle \longrightarrow \langle c_0', \sigma'\rangle}{\langle \textbf{catchin } c_0 \textbf{ with } c_1, \sigma\rangle \longrightarrow \langle \textbf{catchin } c_0' \textbf{ with } c_1, \sigma'\rangle}$$

$$\frac{\langle c_0, \sigma\rangle \longrightarrow \sigma'}{\langle \textbf{catchin } c_0 \textbf{ with } c_1, \sigma\rangle \longrightarrow \sigma'}$$

$$\frac{\langle c_0, \sigma\rangle \longrightarrow \langle \textbf{abort}, \sigma'\rangle}{\langle \textbf{catchin } c_0 \textbf{ with } c_1, \sigma\rangle \longrightarrow \langle c_1, \sigma'\rangle}$$

(c)

$$\frac{\langle c_0, \sigma\rangle \longrightarrow \langle c_0', \sigma'\rangle}{\langle \textbf{catch } l \textbf{ in } c_0 \textbf{ with } c_1, \sigma\rangle \longrightarrow \langle \textbf{catch } l \textbf{ in } c_0' \textbf{ with } c_1, \sigma'\rangle}$$

$$\frac{\langle c_0, \sigma\rangle \longrightarrow \sigma'}{\langle \textbf{catch } l \textbf{ in } c_0 \textbf{ with } c_1, \sigma\rangle \longrightarrow \sigma'}$$

$$\frac{\langle c_0, \sigma\rangle \longrightarrow \langle l', \sigma'\rangle}{\langle \textbf{catch } l \textbf{ in } c_0 \textbf{ with } c_1, \sigma\rangle \longrightarrow \langle l', \sigma'\rangle} \quad \text{when } l \neq l'$$

$$\frac{\langle c_0, \sigma\rangle \longrightarrow \langle l', \sigma'\rangle}{\langle \textbf{catch } l \textbf{ in } c_0 \textbf{ with } c_1, \sigma\rangle \longrightarrow \langle c_1, \sigma'\rangle} \quad \text{when } l = l'$$

Reynolds, exercise 6.4

We write $\hat{\sigma}$ to represent both σ and $\langle \mathbf{abort}, \sigma \rangle$.

$$\frac{}{\langle v := e, \sigma \rangle \Rightarrow [\sigma \mid v : [\![e]\!]_{\mathrm{intexp}}\sigma]} \qquad \frac{}{\langle \mathbf{skip}, \sigma \rangle \Rightarrow \sigma} \qquad \frac{}{\langle \mathbf{fail}, \sigma \rangle \Rightarrow \langle \mathbf{abort}, \sigma \rangle}$$

$$\frac{\langle c_0, \sigma \rangle \Rightarrow \sigma'' \qquad \langle c_1, \sigma'' \rangle \Rightarrow \hat{\sigma}'}{\langle c_0; c_1, \sigma \rangle \Rightarrow \hat{\sigma}'} \qquad \frac{\langle c_0, \sigma \rangle \Rightarrow \langle \mathbf{abort}, \sigma' \rangle}{\langle c_0; c_1, \sigma \rangle \Rightarrow \langle \mathbf{abort}, \sigma' \rangle}$$

$$\frac{\langle c_0, \sigma \rangle \Rightarrow \hat{\sigma}'}{\langle \mathbf{if}\ b\ \mathbf{then}\ c_0\ \mathbf{else}\ c_1, \sigma \rangle \Rightarrow \hat{\sigma}'} \quad \text{when } [\![b]\!]_{\mathrm{boolexp}}\sigma = \mathtt{true}$$

$$\frac{\langle c_1, \sigma \rangle \Rightarrow \hat{\sigma}'}{\langle \mathbf{if}\ b\ \mathbf{then}\ c_0\ \mathbf{else}\ c_1, \sigma \rangle \Rightarrow \hat{\sigma}'} \quad \text{when } [\![b]\!]_{\mathrm{boolexp}}\sigma = \mathtt{false}$$

$$\frac{\langle c, \sigma \rangle \Rightarrow \sigma'' \qquad \langle \mathbf{while}\ b\ \mathbf{do}\ c, \sigma'' \rangle \Rightarrow \hat{\sigma}'}{\langle \mathbf{while}\ b\ \mathbf{do}\ c, \sigma \rangle \Rightarrow \hat{\sigma}'} \quad \text{when } [\![b]\!]_{\mathrm{boolexp}}\sigma = \mathtt{true}$$

$$\frac{\langle c, \sigma \rangle \Rightarrow \langle \mathbf{abort}, \sigma' \rangle}{\langle \mathbf{while}\ b\ \mathbf{do}\ c, \sigma \rangle \Rightarrow \langle \mathbf{abort}, \sigma' \rangle} \quad \text{when } [\![b]\!]_{\mathrm{boolexp}}\sigma = \mathtt{true}$$

$$\frac{}{\langle \mathbf{while}\ b\ \mathbf{do}\ c, \sigma \rangle \Rightarrow \sigma} \quad \text{when } [\![b]\!]_{\mathrm{boolexp}}\sigma = \mathtt{false}$$

$$\frac{\langle c, [\sigma \mid v : [\![e]\!]_{\mathrm{intexp}}\sigma] \rangle \Rightarrow \sigma'}{\langle \mathbf{newvar}\ v := e\ \mathbf{in}\ c, \sigma \rangle \Rightarrow [\sigma' \mid v : \sigma v]} \qquad \frac{\langle c, [\sigma \mid v : [\![e]\!]_{\mathrm{intexp}}\sigma] \rangle \Rightarrow \langle \mathbf{abort}, \sigma' \rangle}{\langle \mathbf{newvar}\ v := e\ \mathbf{in}\ c, \sigma \rangle \Rightarrow \langle \mathbf{abort}, [\sigma' \mid v : \sigma v] \rangle}$$