Homework 2

SNU 4541.664A

Due: 4/29(Wed), 09:30

Kwangkeun Yi, Jinyung Kim, Jeehoon Kang

Exercise 1 다음을 증명하라.

$\gamma : \hat{D} \to D$가 연속(*continuous*) 함수이고 $\hat{D}$가 임의의 부분 집합의 최대 밑뚜껑(*greatest lower bound*)이 있으면($\sqcap$-*complete*), 갈로아 짝 α는

$$\alpha\, x = \sqcap\{\hat{x} \mid x \sqsubseteq \gamma\, \hat{x}\}$$

이다.

Exercise 2 다음의 조립식 요약함수들이 갈로아연결을 만든다는 것을 증명하라.

A와 $\hat{A}$이 갈로아 연결되있고(A를 요약한 것이 $\hat{A}$이고), B와 $\hat{B}$이 갈로아 연결되있다.

- $A \times B$를 $\hat{A} \times \hat{B}$로 요약하는 함수

$$\alpha_{A \times B} = \lambda\langle a, b\rangle.\langle \alpha_A a, \alpha_B b\rangle$$

- $A + B$를 $\hat{A} + \hat{B}$로 요약하는 함수

$$\alpha_{A+B} = \lambda x.\alpha_A x \ \textit{if}\ x \in A,\ \alpha_B x\ o.w.$$

- $A \to B$를 $\hat{A} \to \hat{B}$로 요약하는 함수

$$\alpha_{A \to B} = \lambda f.\alpha_B \circ f \circ \gamma_{\hat{A}}$$

Exercise 3 $2^A \underset{\alpha_1}{\overset{\gamma_1}{\leftrightarrows}} \hat{X}$ 이고 $2^B \underset{\alpha_2}{\overset{\gamma_2}{\leftrightarrows}} \hat{Y}$ 이면, 아래 요약함수들이 갈로아 연결을 만드는 지를 증명하라.

- $2^{A \times B} \leftrightarrows \hat{X} \times \hat{Y}$ 가능

$$\alpha = \lambda X.\langle \alpha_1\{a | \langle a, b\rangle \in X\}, \alpha_2\{b | \langle a, b\rangle \in X\}\rangle$$

- $2^{A \times B} \leftrightarrows A' \to \hat{Y}$ 가능 $(A' \subseteq A)$

$$\alpha = \lambda X.\{a \mapsto \alpha_2 S | \langle a, b\rangle \in X, S = \{b | \langle a, b\rangle \in X\}\}$$

- $2^{A+B} \leftrightarrows \hat{X} \times \hat{Y}$ 가능

$$\alpha = \lambda X.\langle \alpha_1\{a | a \in X, a \in A\}, \alpha_2\{b | b \in X, b \in B\}\rangle$$

- $2^A \to 2^B \leftrightarrows \hat{X} \to \hat{Y}$ 가능

$$\alpha = \lambda f.\alpha_2 \circ f \circ \gamma_1$$

Exercise 4 수업 자료에서 다룬 다룬 `C----` 언어의 abstract machine semantics를 다음과 같이 정의하였다. 빈 칸을 완성하라.

Abstract Machine

$$\langle S, M, C\rangle \in \mathit{Stack} \times \mathit{Memory} \times \mathit{Command}$$

$$\begin{array}{rcrcl} M & \in & \mathit{Memory} & = & \mathit{Var} \overset{\text{fin}}{\to} \mathbb{Z} \\ x & \in & \mathit{Var} & & \\ n & \in & \mathbb{Z} & & \end{array}$$

$$\begin{array}{rcl} S & \to & \epsilon \\ & | & n.S \\ C & \to & \epsilon \\ & | & n.C \\ & | & \texttt{+}.C \\ & | & \texttt{-}.C \\ & | & \texttt{jmpz}(C, C).C \\ & | & \texttt{loop}(C, C).C \\ & | & \texttt{store}(x).C \\ & | & \texttt{load}(x).C \end{array}$$

State Transition

$$
\begin{array}{rcll}
\langle S, M, n.C\rangle & \to & \langle n.S, M, C\rangle & \\
\langle n_2.n_1.S, M, \texttt{+}.C\rangle & \to & \langle (n_1+n_2).S, M, C\rangle & \\
\langle n.S, M, \texttt{-}.C\rangle & \to & \langle (-n).S, M, C\rangle & \\
\langle 0.S, M, \texttt{jmpz}(C_1, C_2).C\rangle & \to & \langle S, M, C_1.C\rangle & \\
\langle n.S, M, \texttt{jmpz}(C_1, C_2).C\rangle & \to & \langle S, M, C_2.C\rangle & (n \neq 0) \\
\langle 0.S, M, \texttt{loop}(C_1, C_2).C\rangle & \to & \boxed{\texttt{(a)}} & \\
\langle n.S, M, \texttt{loop}(C_1, C_2).C\rangle & \to & \boxed{\texttt{(b)}} & (n \neq 0) \\
\langle n.S, M, \texttt{store}(x).C\rangle & \to & \langle S, M\{x \mapsto n\}, C\rangle & \\
\langle S, M, \texttt{load}(x).C\rangle & \to & \langle M(x).S, M, C\rangle &
\end{array}
$$

Translation

$$
\begin{array}{rcl}
[\![skip]\!] & = & \epsilon \\
[\![x \texttt{ := } E]\!] & = & [\![E]\!].\texttt{store}(x) \\
[\![C_1 \texttt{ ; } C_2]\!] & = & [\![C_1]\!].[\![C_2]\!] \\
[\![\texttt{if } E\ C_1\ C_2]\!] & = & [\![E]\!].\texttt{jmpz}([\![C_2]\!], [\![C_1]\!]) \\
[\![\texttt{while } E\ C]\!] & = & \boxed{\texttt{(c)}}
\end{array}
$$

$$
\begin{array}{rcl}
[\![n]\!] & = & n \\
[\![E_1 \texttt{ + } E_2]\!] & = & [\![E_1]\!].[\![E_2]\!].\texttt{+} \\
[\![\texttt{-}E]\!] & = & [\![E]\!].\texttt{-} \\
[\![x]\!] & = & \texttt{load}(x)
\end{array}
$$

Exercise 5 수업자료에서 다룬 `C---` 언어의 abstract machine semantics를 다음과 같이 정의하였다. 빈 칸을 완성하라.

Abstract Machine

$$\langle S, M, E, C\rangle \in \mathit{Stack} \times \mathit{Memory} \times \mathit{Environment} \times \mathit{Command}$$

$$
\begin{array}{rcrcl}
M & \in & \mathit{Memory} & = & \mathit{Loc} \xrightarrow{\text{fin}} \mathbb{Z} \\
E & \in & \mathit{Environment} & = & (\mathit{Var} \times \mathit{Loc})\ \mathit{list} \\
x & \in & \mathit{Var} & & \\
l & \in & \mathit{Loc} & & \\
n & \in & \mathbb{Z} & &
\end{array}
$$

$$
\begin{array}{rcl}
S & \to & \epsilon \\
 & | & n.S \\
C & \to & \epsilon \\
 & | & n.C \\
 & | & \texttt{+}.C \\
 & | & \texttt{-}.C \\
 & | & \texttt{jmpz}(C, C).C \\
 & | & \texttt{loop}(C, C).C \\
 & | & \texttt{store}(x).C \\
 & | & \texttt{load}(x).C \\
 & | & \texttt{bind}(x).C \\
 & | & \texttt{unbind}.C
\end{array}
$$

State Transition

$$
\begin{array}{rcll}
\langle S, M, E, n.C\rangle & \to & \langle n.S, M, E, C\rangle & \\
\langle n_2.n_1.S, M, E, \texttt{+}.C\rangle & \to & \langle (n_1 + n_2).S, M, E, C\rangle & \\
\langle n.S, M, E, \texttt{-}.C\rangle & \to & \langle (-n).S, M, E, C\rangle & \\
\langle 0.S, M, E, \texttt{jmpz}(C_1, C_2).C\rangle & \to & \langle S, M, E, C_1.C\rangle & \\
\langle n.S, M, E, \texttt{jmpz}(C_1, C_2).C\rangle & \to & \langle S, M, E, C_2.C\rangle & (n \neq 0) \\
\langle 0.S, M, E, \texttt{loop}(C_1, C_2).C\rangle & \to & \boxed{\texttt{(d)}} & \\
\langle n.S, M, E, \texttt{loop}(C_1, C_2).C\rangle & \to & \boxed{\texttt{(e)}} & (n \neq 0) \\
\langle n.S, M, E, \texttt{store}(x).C\rangle & \to & \langle S, M\{l \mapsto n\}, E, C\rangle & l = lookup(x, E) \\
\langle S, M, E, \texttt{load}(x).C\rangle & \to & \langle M(l).S, M, E, C\rangle & l = lookup(x, E) \\
\langle n.S, M, E, \texttt{bind}(x).C\rangle & \to & \boxed{\texttt{(f)}} & \\
\langle S, M, (x, l).E, \texttt{unbind}.C\rangle & \to & \boxed{\texttt{(g)}} &
\end{array}
$$

$lookup(x, E) = l$ if (x, l) is the first such entry in E; otherwise undefined.

Translation

$$
\begin{array}{rcl}
[\![skip]\!] & = & \epsilon \\
[\![x \texttt{ := } E]\!] & = & [\![E]\!].\texttt{store}(x) \\
[\![C_1 \texttt{ ; } C_2]\!] & = & [\![C_1]\!].[\![C_2]\!] \\
[\![\texttt{if } E\ C_1\ C_2]\!] & = & [\![E]\!].\texttt{jmpz}([\![C_2]\!], [\![C_1]\!]) \\
[\![\texttt{while } E\ C]\!] & = & \boxed{\quad (\texttt{h}) \quad} \\
[\![\texttt{local } x \texttt{ := } E \texttt{ in } C]\!] & = & [\![E]\!].\texttt{bind}(x).[\![C]\!].\texttt{unbind}
\end{array}
$$

$$
\begin{array}{rcl}
[\![n]\!] & = & n \\
[\![E_1 \texttt{ + } E_2]\!] & = & [\![E_1]\!].[\![E_2]\!].\texttt{+} \\
[\![\texttt{-}E]\!] & = & [\![E]\!].\texttt{-} \\
[\![x]\!] & = & \texttt{load}(x)
\end{array}
$$

Exercise 6 수업자료에서 다룬 `C---+` 언어의 abstract machine semantics를 다음과 같이 정의하였다. 빈 칸을 완성하라.

Abstract Machine

$$\langle S, M, H, C\rangle \in \mathit{Stack} \times \mathit{Memory} \times \mathit{HandlerStack} \times \mathit{Command}$$

$$
\begin{array}{rcrcl}
M & \in & \mathit{Memory} & = & \mathit{Var} \xrightarrow{\text{fin}} \mathbb{Z} \\
H & \in & \mathit{HandlerStack} & = & (\mathit{Stack} \times \mathit{Command})\ \mathit{list} \\
x & \in & \mathit{Var} & & \\
n & \in & \mathbb{Z} & &
\end{array}
$$

$$
\begin{array}{rcl}
S & \to & \epsilon \\
 & | & n.S \\
C & \to & \epsilon \\
 & | & n.C \\
 & | & \texttt{+}.C \\
 & | & \texttt{-}.C \\
 & | & \texttt{jmpz}(C, C).C \\
 & | & \texttt{loop}(C, C).C \\
 & | & \texttt{store}(x).C \\
 & | & \texttt{load}(x).C \\
 & | & \texttt{install}(C).C \\
 & | & \texttt{raise}.C \\
 & | & \star.C
\end{array}
$$

State Transition

$$
\begin{array}{rcll}
\langle S, M, H, n.C\rangle & \to & \langle n.S, M, H, C\rangle & \\
\langle n_2.n_1.S, M, H, \texttt{+}.C\rangle & \to & \langle (n_1+n_2).S, M, H, C\rangle & \\
\langle n.S, M, H, \texttt{-}.C\rangle & \to & \langle (-n).S, M, H, C\rangle & \\
\langle 0.S, M, H, \texttt{jmpz}(C_1, C_2).C\rangle & \to & \langle S, M, H, C_1.C\rangle & \\
\langle n.S, M, H, \texttt{jmpz}(C_1, C_2).C\rangle & \to & \langle S, M, H, C_2.C\rangle & (n \neq 0) \\
\langle 0.S, M, H, \texttt{loop}(C_1, C_2).C\rangle & \to & \boxed{\texttt{(i)}} & \\
\langle n.S, M, H, \texttt{loop}(C_1, C_2).C\rangle & \to & \boxed{\texttt{(j)}} & (n \neq 0) \\
\langle n.S, M, H, \texttt{store}(x).C\rangle & \to & \langle S, M\{x \mapsto n\}, H, C\rangle & \\
\langle S, M, H, \texttt{load}(x).C\rangle & \to & \langle M(x).S, M, H, C\rangle & \\
\langle S, M, H, \texttt{install}(C').C\rangle & \to & \langle S, M, (S, C').H, C\rangle & \\
\langle S, M, (S', C').H, \texttt{raise}.\cdots.\star.C\rangle & \to & \boxed{\texttt{(k)}} & (\text{“}\cdots\text{” does not contain } \star) \\
\langle S, M, (S', C').H, \star.C\rangle & \to & \langle S, M, H, C\rangle &
\end{array}
$$

Translation

$$
\begin{array}{rcl}
[\![skip]\!] & = & \epsilon \\
[\![x \texttt{ := } E]\!] & = & [\![E]\!].\texttt{store}(x) \\
[\![C_1 \texttt{ ; } C_2]\!] & = & [\![C_1]\!].[\![C_2]\!] \\
[\![\texttt{if } E\ C_1\ C_2]\!] & = & [\![E]\!].\texttt{jmpz}([\![C_2]\!], [\![C_1]\!]) \\
[\![\texttt{while } E\ C]\!] & = & \boxed{\texttt{(l)}} \\
[\![\texttt{raise}]\!] & = & \texttt{raise} \\
[\![\texttt{try } C_1 \texttt{ handle } C_2]\!] & = & \texttt{install}([\![C_2]\!]).[\![C_1]\!].\star
\end{array}
$$

$$
\begin{array}{rcl}
[\![n]\!] & = & n \\
[\![E_1 \texttt{ + } E_2]\!] & = & [\![E_1]\!].[\![E_2]\!].\texttt{+} \\
[\![\texttt{-}E]\!] & = & [\![E]\!].\texttt{-} \\
[\![x]\!] & = & \texttt{load}(x)
\end{array}
$$

□