Homework Final Set

프로그래밍언어 특강

이 광근

기한: 12/16(화) 14:00
(302동428호 IN박스)

1. 슬라이드 2-3:p.16 의 언어와 고전논리$_{\text{classical propositional logic}}$는 서로 거울이다.

   - 슬라이드 2-3:p.15 의 증명에 해당하는 프로그램을 작성하라.
   - $(A \Rightarrow B) \wedge ((A \vee B) \Rightarrow C) \Rightarrow (A \Rightarrow (C \wedge C))$의 증명에 해당하는 프로그램을 작성하라.
   - 가정) 논리식에서 이름 A, B, C 에 대응하는 세가지 기본타입이 위 언어에 있다고 가정하자.

2. 슬라이드 2-3:p.28 의 CoC 언어로 다음 명제들의 증명에 해당하는 프로그램들을 작성하라. (∀-명제 증명규칙과 CoC 식의 대응은 슬라이드 2-3:pp.37-38, 나머지 증명규칙은 고전논리 규칙과 동일)

   - $(\forall x{:}X.(P \Rightarrow Q(x))) \;\Rightarrow\; (P \;\Rightarrow\; \forall x{:}X.Q(x))$
   - $(\forall x{:}X.(P(x) \Rightarrow Q(x)))$
     $\Rightarrow\; (\forall x{:}X.(Q(x) \Rightarrow R(x)) \Rightarrow\; \forall x{:}X.(P(x) \Rightarrow R(x)))$
   - 가정) 논리식에서 집합 X에 해당하는 것이 CoC 언어에서는 타입 X라고 가정하고, 논리식에서 성질$_{\text{predicate}}$ P, Q, R에 해당하는 것이 CoC 언어에서는 같은 이름으로 타입 $X \rightarrow \text{Type}$를 가지고 있다고 가정하자.

3. 작은타입$_{\text{subtype}}$(슬라이드 2-0:p.19)으로 아래 프로그램의 타입유추 과정(증명나무)을 보이라.

```
(rec f x (x.a + f(a=1, b=false, c=2)))(a=1, b=true)
```

4. 재귀타입$_{\text{recursive type}}$(슬라이드 2-0:pp.20-21)으로 아래 프로그램의 타입유추 과정(증명나무)을 보이라.

   갈래타입$_{\text{variant type}}$ 방정식 $a = \mathtt{N}(unit) + \mathtt{C}(int \times a)$의 답을 재귀타입 $\mu a.\mathtt{N}(unit) + \mathtt{C}(int \times a)$로 한다.

   ```
   C(2, N())
   ```

5. System F(슬라이드 2-1:pp.8-11)로 아래 프로그램의 타입유추 과정(증명나무)을 보이라.

   $$\lambda n : \text{nat}.\Lambda a.\lambda s : a \to a.\lambda z : a.s((n\ a)\ s\ z)$$

   여기서,

   $$\text{nat} \stackrel{\text{def}}{=} \forall a.(a \to a) \to (a \to a).$$

6. CoC(슬라이드 2-3:pp.32-35)로 아래 프로그램의 타입유추 과정(증명나무)을 완성하라.

   $$\begin{array}{l} X{:}\text{Type}, P{:}X \to X \to \text{Type} \\ \vdash \\ E : (\Pi x{:}X.\Pi y{:}X.P\ x\ y) \to (\Pi z{:}X.P\ z\ z) \end{array}$$

   여기서,

   $$E \stackrel{\text{def}}{=} (\lambda k{:}(\Pi x{:}X.\Pi y{:}X.P\ x\ y).\lambda z{:}X.k\ z\ z).$$

7. 코드가 값으로 자유롭게 다뤄지는$_{\text{first-class object}}$ 다단계$_{\text{multi-staged}}$ 프로그래밍언어를 생각하자 (슬라이드 1-2:pp.23-33).

   - 아래 프로그램의 타입유추(슬라이드 2-0:p.18) 과정(증명나무)을 보이라.

     ```
     run box(fn x (unbox (box (x+1))))
     ```

   - 다단계$_{\text{multi-staged}}$ 프로그래밍 부품들$_{\text{language constructs}}$은 당연히 설탕이다. 논문 `https://kwangkeunyi.snu.ac.kr/paper/11-popl-chakyita.pdf` 을 참고해서, 위의 다단계 프로그램의 설탕을 녹이면 어떤 프로그램이 되는지 보이라.

     ```
     run box(fn x (unbox (box (x+1))))
     ```

8. 실용적으로 널리 사용되는 let-여러모양 타입유추$_{\text{let-polymorphic type inference}}$는 타입의 크기가 프로그램크기($N$)에 기하급수($2^N$)로 커지는 경우가 사실

있다. 슬라이드 2-2:p.33의 예제 프로그램의 타입유추 과정(슬라이드 2-2:p.27 규칙들로 만든 증명나무)를 보이라.

9. 예외상황이 발생할때 예외상황 이름과 함께 값을 전달하는 언어를 생각하자. 슬라이드 1-2:p.19의 언어에서 `raise`/`handle`의 문법과 의미를 살짝 확장하자.

   - `raise L` E : E를 먼저 계산한다. 정상적으로 끝나면 그 값을 예외상황 L을 발생시킬 때 L과 함께 전달한다.
   - E `handle L` x E': E를 실행중에 예외상황 `L`이 발생하면 E'을 수행한 값이 이 식의 결과가 된다. 이 때 x는 `L`과 함께 온 값에 붙는 이름이고 그 이름의 유효범위는 E'이다. E 실행중에 예외상황이 발생하지 않으면 그 값이 이 식의 결과가 된다.

   이런 의미가 되도록 슬라이드 1-2:p.19를 변경해서 정의하라. 그리고, 이런 의미의 예외상황을 녹이는 정의를 보이라. 슬라이드 1-2:p.20을 변경하도록 한다.

10. 다음의 언어를 생각하자:

$$
\begin{array}{rcll}
E & \to & n \mid x & \text{정수} \mid \text{변수} \\
 & \mid & \texttt{let}\, x\, E\, E & \text{이름 정하고 쓰기} \\
 & \mid & E \,\texttt{|}\, E & \text{무작위 선택} \\
 & \mid & \texttt{stretch}\, E\, E & \text{늘이기} \\
 & \mid & E\texttt{+} & \text{1 증가} \\
 & \mid & E\texttt{-} & \text{1 감소}
\end{array}
$$

   $\texttt{let}\, x\, E_1\, E_2$는 E_1의 값을 x로 이름짓고 E_2를 계산한 값이다. x의 유효범위는 E_2이다. $E_1 \,\texttt{|}\, E_2$는 E_1이거나 E_2 중 하나의 값이다. $\texttt{stretch}\, E_1\, E_2$의 값은 E_1 크기만큼 E_2기준으로 무작위 선택지를 늘린다:

$$
\texttt{stretch}\, E_1\, E_2 \quad \equiv \quad E_2(\texttt{+}\texttt{|}\texttt{-})^{0..|E_1|}
$$

   즉, `stretch 2 100`은 `100|100+|100-|100++|100+-|100-+|100--`를 뜻한다.

   - 큰보폭 스타일로 위 언어의 의미구조를 정의하라. 스타일은, $\sigma \vdash E \Downarrow v$를 증명하는 규칙들로 정의하고, 실행환경 σ와 값 v의 공간을 비롯해서 모든 것을 정의하도록 한다.

- 실행전 의미구조$_{\text{static semantics}}$를 정의하라.
  - 스타일은, $\Sigma \vdash E : \rho$를 증명하는 규칙들로 정의하고, Σ와 ρ의 공간을 비롯해서 모든 것을 정의하도록 한다.
  - 정의한 실행전 의미구조는 유한한 계산으로 실현가능해야 한다.
  - 정의한 실행전 의미구조는 다음을 만족해야한다:

$$\epsilon \vdash E : \rho \quad \text{이면} \quad [\![E]\!] = [\![\rho]\!].$$

    위에서, ϵ은 빈 Σ를 뜻하고, $[\![E]\!] \in 2^{\mathbb{Z}}$는 식 E안의 자유변수(입력값을 받는 변수)를 커버하는 모든 실행환경 아래서 계산되는 모든 가능한 값을 모은 것이다:

$$[\![E]\!] \stackrel{\text{def}}{=} \{v \mid \sigma \vdash E \Downarrow v\}.$$

    위에서, $[\![\rho]\!]$는 ρ가 뜻하는 집합이다.