CS520, Handout 24, 21 November 2006

Subtyping (continued)

Reading: [Pierce, Chapter 15]

The simply-typed lambda-calculus with subtyping

• The simply-typed lambda-calculus with subtyping (denoted $\boldsymbol{\lambda}_{<:}$) extends $\boldsymbol{\lambda}_{\rightarrow}$ as presented in [Pierce, Figure 9-1, page 103].

• The full definition of $\boldsymbol{\lambda}_{<:}$ is in [Pierce, Figure 15-1, page 186].

• The only piece missing from [Pierce, Figure 15-1, page 186] to make it self-contained is a base type in the syntax of types.

• In Handout 07, a base type was denoted by $\bullet$. In Handout 09, base types were ranged over by metavariable A. (Reminder: "metavariables" ranging over base types are different from "type variables" – the latter are used in Handouts 18 to 22.)

Subtyping rules and new typing rule for $\lambda_{<:}$

[Pierce, Figure 15-1, right column]

The new type constant `Top` is like `Object` in Java.

$$T <: T \qquad \text{(S-REFL)}$$

$$\frac{T <: T' \qquad T' <: T''}{T <: T''} \qquad \text{(S-TRANS)}$$

$$T <: \texttt{Top} \qquad \text{(S-TOP)}$$

$$\frac{T_1 <: T_1' \qquad T_2 <: T_2'}{T_1' \to T_2 <: T_1 \to T_2'} \qquad \text{(S-ARROW)}$$

$$\frac{\Gamma \vdash t : T \qquad T <: T'}{\Gamma \vdash t : T'} \qquad \text{(T-SUB)}$$

If base types (e.g., `Nat`, `Bool`, `Int`, `Real`, `String`, etc.) are included in the syntax of types, then several subtyping relationships will hold between them. For example,

$$\texttt{Nat} <: \texttt{Int}$$
$$\texttt{Int} <: \texttt{Real}$$
$$\texttt{Real} <: \texttt{Top}$$
$$\ldots$$

None of the preceding is included in [Pierce, Figure 15-1].

If base types (e.g., `Nat`, `Bool`, `Int`, `Real`, `String`, etc.) are included in the syntax of types, then several subtyping relationships will hold between them. For example,

$$
\begin{aligned}
&\texttt{Nat} <: \texttt{Int}\\
&\texttt{Int} <: \texttt{Real}\\
&\texttt{Real} <: \texttt{Top}\\
&\ldots
\end{aligned}
$$

With the preceding subtyping relationships among base types, we can derive further subtyping relationships, using the subtyping rules. For example,

$$
\begin{aligned}
&\texttt{Nat} <: \texttt{Real}\\
&\texttt{Real} \rightarrow \texttt{Int} <: \texttt{Real} \rightarrow \texttt{Real}\\
&\texttt{Real} \rightarrow \texttt{Real} <: \texttt{Int} \rightarrow \texttt{Real}\\
&\ldots
\end{aligned}
$$

Two different typing derivations for $(\texttt{lambda x : Real. sqRoot x})\ 5$, the first from Handout 23, where $\Gamma = \{\texttt{x} : \texttt{Real}\}$.

The first promotes the type of the argument 5:

$$
\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma \vdash \texttt{sqRoot} : \texttt{Real} \rightarrow \texttt{Real}} \qquad \dfrac{(\texttt{x} : \texttt{Real}) \in \Gamma}{\Gamma \vdash \texttt{x} : \texttt{Real}}}{\Gamma \vdash \texttt{sqRoot x} : \texttt{Real}}}{\vdash (\lambda \texttt{x} : \texttt{Real}.\, \texttt{sqRoot x}) : \texttt{Real} \rightarrow \texttt{Real}} \qquad \dfrac{\dfrac{}{\vdash 5 : \texttt{Int}} \qquad \dfrac{}{\texttt{Int} <: \texttt{Real}}}{\vdash 5 : \texttt{Real}}}{\vdash (\lambda \texttt{x} : \texttt{Real}.\, \texttt{sqRoot x})\ 5 : \texttt{Real}}
$$

Two different typing derivations for $(\mathtt{lambda\ x :\ Real.\ sqRoot\ x})\ 5$, the first from Handout 23, where $\Gamma = \{\mathtt{x} : \mathtt{Real}\}$.

The first promotes the type of the argument 5:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma \vdash \mathtt{sqRoot} : \mathtt{Real} \to \mathtt{Real}} \quad \dfrac{(\mathtt{x} : \mathtt{Real}) \in \Gamma}{\Gamma \vdash \mathtt{x} : \mathtt{Real}}}{\Gamma \vdash \mathtt{sqRoot\ x} : \mathtt{Real}}}{\vdash (\lambda \mathtt{x} : \mathtt{Real}.\, \mathtt{sqRoot\ x}) : \mathtt{Real} \to \mathtt{Real}} \quad \dfrac{\dfrac{}{\vdash 5 : \mathtt{Int}} \quad \dfrac{}{\mathtt{Int} <: \mathtt{Real}}}{\vdash 5 : \mathtt{Real}}}{\vdash (\lambda \mathtt{x} : \mathtt{Real}.\, \mathtt{sqRoot\ x})\ 5 : \mathtt{Real}}$$

The second promotes the type of $(\mathtt{lambda\ x :\ Real.\ sqRoot\ x})$:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma \vdash \mathtt{sqRoot} : \mathtt{Real} \to \mathtt{Real}} \quad \dfrac{(\mathtt{x} : \mathtt{Real}) \in \Gamma}{\Gamma \vdash \mathtt{x} : \mathtt{Real}}}{\Gamma \vdash \mathtt{sqRoot\ x} : \mathtt{Real}}}{\vdash (\lambda \mathtt{x} : \mathtt{Real}.\, \mathtt{sqRoot\ x}) : \mathtt{Real} \to \mathtt{Real}} \quad \dfrac{}{\mathtt{Real} \to \mathtt{Real} <: \mathtt{Int} \to \mathtt{Real}}}{\vdash (\lambda \mathtt{x} : \mathtt{Real}.\, \mathtt{sqRoot\ x}) : \mathtt{Int} \to \mathtt{Real}} \quad \dfrac{}{\vdash 5 : \mathtt{Int}}}{\vdash (\lambda \mathtt{x} : \mathtt{Real}.\, \mathtt{sqRoot\ x})\ 5 : \mathtt{Real}}$$

Subtyping in the presence of records

[Pierce, Figure 15-3, page 187]

Three additional subtyping rules, the first of which (S-RcdWidth) already mentioned in Handout 23.

$$\{\ell_i : T_i{}^{i \in 1..n+k}\} <: \{\ell_i : T_i{}^{i \in 1..n}\} \qquad \text{(S-RcdWidth)}$$

Subtyping in the presence of records

[Pierce, Figure 15-3, page 187]

Three additional subtyping rules, the first of which (S-RcdWidth) already mentioned in Handout 23.

$$\{\ell_i : T_i{}^{i\in 1..n+k}\} <: \{\ell_i : T_i{}^{i\in 1..n}\} \qquad \text{(S-RcdWidth)}$$

$$\frac{\text{for each } i \quad T_i <: U_i}{\{\ell_i : T_i{}^{i\in 1..n}\} <: \{\ell_i : U_i{}^{i\in 1..n}\}} \qquad \text{(S-RcdDepth)}$$

Subtyping in the presence of records

[Pierce, Figure 15-3, page 187]

Three additional subtyping rules, the first of which (S-RcdWidth) already mentioned in Handout 23.

$$\{\ell_i : T_i{}^{i\in 1..n+k}\} <: \{\ell_i : T_i{}^{i\in 1..n}\} \qquad \text{(S-RcdWidth)}$$

$$\frac{\text{for each } i \quad T_i <: U_i}{\{\ell_i : T_i{}^{i\in 1..n}\} <: \{\ell_i : U_i{}^{i\in 1..n}\}} \qquad \text{(S-RcdDepth)}$$

$$\frac{\{k_i : T_i{}^{i\in 1..n}\} \text{ is a permutation of } \{\ell_j : U_j{}^{j\in 1..n}\}}{\{k_i : T_i{}^{i\in 1..n}\} <: \{\ell_j : U_j{}^{j\in 1..n}\}} \qquad \text{(S-RcdPerm)}$$

Safety = Progress + Preservation

[Pierce, Section 15.3, page 188]

Preservation [Pierce, Theorem 15.3.5]
If $\Gamma \vdash t : T$ is derivable and $t \Rightarrow t'$, then $\Gamma \vdash t' : T$ is derivable.

Progress [Pierce, Theorem 15.3.7]
Let t be a closed term. If t is typable, i.e., $\vdash t : T$ is derivable for some type T, then either t is a value or else $t \Rightarrow t'$ for some term t'.

Safety = Progress + Preservation

[Pierce, Section 15.3, page 188]

Preservation [Pierce, Theorem 15.3.5]
If $\Gamma \vdash t : T$ is derivable and $t \Rightarrow t'$, then $\Gamma \vdash t' : T$ is derivable.

Progress [Pierce, Theorem 15.3.7]
Let t be a closed term. If t is typable, i.e., $\vdash t : T$ is derivable for some type T, then either t is a value or else $t \Rightarrow t'$ for some term t'.

The two theorems are proved for $\boldsymbol{\lambda}_{<:}$ augmented with records. They still hold when $\boldsymbol{\lambda}_{<:}$ is augmented with **some (not all)** of the features considered in [Pierce, Section 15.5] with appropriate typing rules: variants (pp 196-197), lists (p 197), references (p 198) and arrays (pp 198-199).

Adding a "bottom" type

[Pierce, Section 15.4, page 191]

Add a new type constant `Bot`, with no counterpart in Java.
`Bot` is the opposite of `Top`.

A new subtyping rule:

$$\texttt{Bot} <: T \qquad \text{(S-BOT)}$$

Quick Exercise: Show that the type `Bot` is not inhabited, i.e., there is no value (or no well-typed closed term) which has type `Bot`.

(By contrast, every term can be given type `Top`.)

See [Pierce, page 192] for uses of the type `Bot`.

Ascription and casting

[Pierce, Section 15.5, pp 193-196]

Instead of writing $(T)t$ to denote casting, we write $t \texttt{ as } T$.
In the presence of subtyping, take ascription = up-casting.

Ascription and casting

[Pierce, Section 15.5, pp 193-196]

Instead of writing $(T)t$ to denote casting, we write $t\ \texttt{as}\ T$.
In the presence of subtyping, take ascription = up-casting.

$$\dfrac{\dfrac{\dfrac{\vdots}{\Gamma \vdash t : T} \qquad \dfrac{\vdots}{T <: T'}}{\Gamma \vdash t : T'}\ \textsc{T-Sub}}{\Gamma \vdash t\ \texttt{as}\ T' : T'}\ \textsc{T-Ascribe}$$

Ascription and casting

[Pierce, Section 15.5, pp 193-196]

Instead of writing $(T)t$ to denote casting, we write $t\ \texttt{as}\ T$.
In the presence of subtyping, take ascription = up-casting.

$$\dfrac{\dfrac{\dfrac{\vdots}{\Gamma \vdash t : T} \qquad \dfrac{\vdots}{T <: T'}}{\Gamma \vdash t : T'}\ \textsc{T-Sub}}{\Gamma \vdash t\ \texttt{as}\ T' : T'}\ \textsc{T-Ascribe}$$

The evaluation rules for up-cast are simply those of ascription [Pierce, Section 11.4]:

$$v\ \texttt{as}\ T \Rightarrow v \qquad \text{(E-Ascribe)}$$

$$\dfrac{t \Rightarrow t'}{t\ \texttt{as}\ T \Rightarrow t'\ \texttt{as}\ T} \qquad \text{(E-Ascribe1)}$$

Ascription $\neq$ down-casting

Typing rule for down-casts

$$\frac{\Gamma \vdash t : T}{\Gamma \vdash t \text{ as } T' : T'} \quad \text{(T-DOWNCAST)} \quad \text{this is dangerous!}$$

Ascription $\neq$ down-casting

Typing rule for down-casts

$$\frac{\Gamma \vdash t : T}{\Gamma \vdash t \text{ as } T' : T'} \quad \text{(T-DOWNCAST)} \quad \text{this is dangerous!}$$

Evaluation rules for down-casts

$$\frac{\vdash v : T}{v \text{ as } T \Rightarrow v} \quad \text{(E-DOWNCAST)} \quad \text{a bit strange?}$$

$$\frac{t \Rightarrow t'}{t \text{ as } T \Rightarrow t' \text{ as } T} \quad \text{(E-ASCRIBE1)}$$