Universidade Federal de Santa Maria
Programa de Pós-Graduação em Informática - PPGI

Subtipos e Metateoria dos Subtipos

Linguagens de Programação – ELC921
Profª Drª Juliana Kaiser Vizzotto

Alunos: Alberto Kummer, Daniel Di Domenico, Fernando Campagnolo, Jéssica Lasch de Moura e José Puiati

15 Subtyping

- Também chamado de *subtype polymorphism*;
- Característica presente nas linguagens orientadas a objetos;
- Cálculo Lambda simplesmente tipado com subtipos: $\lambda_{<:}$

15.1 Subsumption

- Sem subtipos:
  - ✓ Regras de tipos bastante rígidas;
  - ✓ Rejeição de expressões que, aos olhos do programador, são bem tipadas.
- Exemplo:

$$\frac{\Gamma \vdash t_1 : T_{11} \to T_{12} \qquad \Gamma \vdash t_2 : T_{11}}{\Gamma \vdash t_1\, t_2 : T_{11}} \qquad (\text{T-App})$$

$(\lambda r : \{x : Nat\}. \quad r.x) \quad \{x = 0, y = 1\}$ Inválido?

15.1 Subsumption

- Sem subtipos:
  - ✓ Regras de tipos bastante rígidas;
  - ✓ Rejeição de expressões que, aos olhos do programador, são bem tipadas.
- Exemplo:

$$\frac{\Gamma \vdash t_1 : T_{11} \to T_{12} \qquad \Gamma \vdash t_2 : T_{11}}{\Gamma \vdash t_1\, t_2 : T_{11}} \qquad (\text{T-App})$$

$$(\lambda r : \{x : Nat\}. \quad r.x) \quad \{x = 0, y = 1\} \qquad \text{Válido}$$

15.1 Subsumption

- Objetivo dos subtipos:
  - ✓ Refinamento das regras de tipos;
  - ✓ Se S é subtipo de T ($S <: T$), qualquer termo do tipo S pode ser utilizado no contexto onde T é esperado;
  - ✓ Princípio da substituição segura (*safe substituition*).

Regra *Subsumption*

$$\frac{\Gamma \vdash t : S \qquad S <: T}{\Gamma \vdash t : T} \qquad \text{(T-Sub)}$$

Adaptado de (**?**).

15.1 Subsumption

- Exemplo: $(\lambda r : \{x : Nat\}. \quad r.x) \quad \{x = 0, y = 1\}$
  - ✓ Considerando que: $\{x : Nat, y : Nat\} <: \{x : Nat\}$
  - ✓ A regra T-$\textsc{Sub}$ permite a aplicação pois são tipos válidos.

15.2 A relação de subtipos

- Coleção de regras de inferência para derivar declarações

$$\texttt{S} \texttt{ <: } \texttt{S} \qquad \text{(S-Refl)}$$

$$\frac{\texttt{S} \texttt{ <: } \texttt{U} \qquad \texttt{U} \texttt{ <: } \texttt{T}}{\texttt{S} \texttt{ <: } \texttt{T}} \qquad \text{(S-Trans)}$$

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

15.2 A relação de subtipos

- Exemplos:
  - $\{x : Nat\}$
    - $\{x = 3\},\ \{x = 5\},\ e\ \{x = 3, a = true, b = true\}$
  - $\{x : Nat, y : Nat\}$
    - $\{x = 3, y = 100\}\ e\ \{x = 3, y = 100, z = true\}$

- É seguro permitir que os tipos dos campos variem desde que os tipos correspondentes nos dois registros estejam na relação de subtipos;

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

- A ordem dos campos no registro não faz diferença em como podemos usá-los;

$$\frac{\{k_j : S_j \ ^{j \in 1..n}\} \texttt{ is a permutation of } \{\rceil_i : T_i \ ^{i \in 1..n}\}}{\{k_j : S_j \ ^{j \in 1..n}\} <: \{_i : T_i \ ^{i \in 1..n}\}} \quad \text{(S-RcdPerm)}$$

- Como estamos trabalhando com uma linguagem que não terá apenas números e registros, mas também funções funções podem ser utilizadas como argumentos, precisamos especificar sobre quais circunstâncias é seguro usar uma função de um determinado tipo em um contexto onde é esperada uma função de um tipo diferente;

$$\frac{T_1 <: S_1 \qquad S_2 <: T_2}{S_1 \to S_2 <: T_1 \to T_2} \qquad \text{(S-Arrow)}$$

- É conveniente ter um tipo que seja um "supertipo" de cada tipo, para isso introduzimos a nova constante de tipo "Top";

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

15.4 The Top and Bottom Types

Formas de subtipos - Top e Bot

Formas sintáticas:

$$\begin{aligned} T ::= & \ldots \\ & \mathit{Top} \\ & \mathit{Bot} \end{aligned}$$

Regras de subtipos:

$$\mathit{S} <: \mathit{Top} \qquad (\text{S-Top})$$

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

Adaptado de (**?**).

15.4 The Top and Bottom Types

- Top:
  - ✓ Elemento **máximo** da relação de subtipos;
  - ✓ Equivale ao tipo *Object* das linguagens orientadas a objetos;
  - ✓ Dispositivo técnico sofisticado em sistemas que combinam subtipos com poliformismo.
- Bot:
  - ✓ Elemento **mínimo** da relação de subtipos;
  - ✓ Tipo vazio (não existem valores do tipo Bot);
  - ✓ Muito útil para expressar algumas operações que não visam retorno de valores, como exceções, pois:
    - Permite ao programador definir expressões sem retorno com o tipo Bot;
    - Indica ao *typechecker* que a expressão pode ser utilizada com segurança em qualquer contexto.

15.4 The Top and Bottom Types

- Exemplo:

$\lambda x : T.$
  if $<$ valor apropriado para x $>$ then
    $<$ calcula o resultado $>$
  else
    error

15.4 The Top and Bottom Types

- Tipo `Bot` dificulta a implementação;
- Mudança da regra de tipo da aplicação:
  - ✓ `t1 t2` :
  - ✓ `t1` pode ser tanto do tipo seta $(T1 \to T2)$ ou do tipo `Bot`.

SLIDES FERNANDO

SLIDES ALBERTO

SLIDES KADICO