Indecibilidade da Inferência de Tipos com Recursão Polimórfica

Gabriela Moreira e Luiz Gustavo Eburneo

gabrielamoreira05@gmail.com,
botucacontato@gmail.com

Departamento de Ciência da Computação
Centro de Ciências e Tecnológicas
Universidade do Estado de Santa Catarina

7 de Dezembro de 2017

Sistemas de Tipos

Problemas

Redução

Conclusão

Cálculo Lambda

$$e := v \mid e_1 e_2 \mid \lambda x.e$$

Sistema de Tipos

Sentenças (*judgments*)

$$\Gamma \vdash e : \sigma$$

Regras

$$\frac{\Gamma_1 \vdash e_1 : \sigma_1 \quad \Gamma_2 \vdash e_2 : \sigma_2 \quad \ldots \quad \Gamma_n \vdash e_n : \sigma_n}{\Gamma \vdash e : \sigma}$$

Cálculo Lambda Simplesmente Tipado

Nesse sistema, além de expressões lambda, são definidos tipos simples. Um tipo simples τ tem forma:

$$\tau := \alpha \mid \tau \rightarrow \tau'$$

Onde α é uma variável de tipo e $\tau \rightarrow \tau'$ é o tipo de uma função que recebe um parâmetro do tipo τ e retorna um tipo τ'.

Regras:

$$\Gamma \vdash x : \tau \ (VAR) \quad \{x : \tau\} \in \Gamma$$

$$\frac{\Gamma \vdash e : \tau \rightarrow \tau' \quad \Gamma \vdash e' : \tau}{\Gamma \vdash e\ e' : \tau'} \ (APP) \qquad \frac{\Gamma, x : \tau' \vdash e : \tau}{\Gamma \vdash \lambda x.e : \tau' \rightarrow \tau} \ (ABS)$$

Exemplo para regra APP

```
chr 97 = 'a'
chr ::  Int -> Char
97 ::  Int
```

Exemplo para regra APP

```
chr 97 = 'a'
chr ::  Int -> Char
97 ::  Int

Então
chr 'a' ::  Char
```

Operador de Ponto Fixo

Tipos recursivos podem ser definidos através do operador de ponto fixo, com o uso de expressões de tipo $\mu\alpha.\tau$, que denota o isomorfismo dos tipos que satisfazem a equação

$$\mu\alpha.\tau \cong \{\alpha \mapsto \mu\alpha.\tau\}\tau.$$

As regras para definir tal isomorfismo precisam ser acrescentadas ao sistema:

$$\frac{\Gamma \vdash e : \{\alpha \mapsto \mu\alpha.\tau\}\tau}{\Gamma \vdash \text{fold } e : \mu\alpha.\tau}\ (\textit{FOLD}) \qquad \frac{\Gamma \vdash e : \mu\alpha.\tau}{\Gamma \vdash \text{unfold } e : \{\alpha \mapsto \mu\alpha.\tau\}\tau}\ (\textit{UNFOLD})$$

Tipos Parametricamente Polimórficos

Para a interpretação de tipos parametricamente polimórficos, são definidos tipos quantificados. Assim, tem-se a combinação de tipos simples (τ) e tipos quantificados (σ):

$$\tau := \alpha \mid \tau \rightarrow \tau'$$
$$\sigma := \tau \mid \forall \alpha . \sigma$$

Por exemplo, o tipo da função identidade:

$$\texttt{id} :: \forall \alpha . \alpha \rightarrow \alpha$$
$$\texttt{id 'a'} :: \texttt{Char}$$
$$\texttt{id 1} :: \texttt{Int}$$

Regras Sistema Damas-Milner

Assim, são acrescentadas três novas regras para quantificar, instanciar e utilizar tipos parametricamente polimórficos:

$$\frac{\Gamma \vdash e : \forall\alpha.\sigma \quad (\alpha \notin ftv(\Gamma))}{\Gamma \vdash e : \sigma}\ (GEN) \qquad \frac{\Gamma \vdash e : \sigma \quad (\sigma \geqslant \sigma')}{\Gamma \vdash e : \sigma'}\ (INST)$$

$$\frac{\Gamma \vdash e : \sigma \quad \Gamma, x : \sigma \vdash e' : \tau}{\Gamma \vdash \text{let } x = e \text{ in } e' : \tau}\ (LET)$$

Sistema Milner-Mycroft

A modificação proposta por Alan Mycroft (1984) trata-se apenas de aceitar o uso de tipos quantificados com operadores de ponto fixo, o que depende de uma alteração nas regras (*FOLD*) e (*UNFOLD*):

$$\frac{\sigma \vdash e : \{\sigma \mapsto \mu\sigma.\tau\}\tau}{\sigma \vdash \text{fold } e : \mu\sigma.\tau}\ (FOLD') \qquad \frac{\Gamma \vdash e : \mu\sigma.\tau}{\sigma \vdash \text{unfold } e : \{\sigma \mapsto \mu\sigma.\tau\}\tau}\ (UNFOLD')$$

Mas a inferência de tipos para esse sistema é um problema indecidível.

Inferência de tipos com recursão polimórfica

Um tipo recursivamente polimórfico é aquele que ocorre em sua própria definição. Uma árvore binária totalmente balanceada pode assumir essa representação:

```
data Arvore a = Cons a (Arvore (a,a)) | Nil
```

Uma árvore de altura igual a três pode ser descrita como:

```
Cons 1 (Cons (2,3) (Cons ((4,5),(6,7)) Nil))
```

Inferência de tipos com recursão polimórfica

Pode-se declarar uma função para calcular a altura de uma árvore:

```
altura ::  Arvore a -> Int
altura Nil = 0
altura (Cons n a) = 1 + altura a
```

```
n = 1,              a =Cons (2,3) (Cons ((4,5),(6,7)) Nil)
n = (2,3),          a = Cons ((4,5),(6,7)) Nil
n = ((4,5),(6,7)), a = Nil
```

O problema em questão é a inferência do tipo de expressões como a função *altura*, onde ocorre recursão polimófica.

Unificação

Dois tipos σ e σ' são ditos unificáveis se e somente se existe uma substituição S da forma $\{\alpha_1 \mapsto \tau_1, \alpha_2 \mapsto \tau_2, \ldots, \alpha_n \mapsto \tau_n\}$ tal que $S(\sigma) = S(\sigma')$.

```
S(a,Char) = S(Int,b)

S = {a ↦ Int,b ↦ Char}
```

Semi Unificação

Dado um conjunto de pares de tipos $\{(M_1, N_1), (M_2, N_2), \ldots, (M_n, N_n)\}$ onde $M_i \geqslant N_i$ (M_i é mais geral que N_i). O conjunto é dito semi unificável se e somente se existe um semi unificador S e um conjunto de substituições $\{S_1, S_2, \ldots, S_n\}$ tal que

$$S_i(S(M_i) = S(N_i)$$

Semi Unificação - SEI's

Esse problema pode ser descrito em sistemas de equações e inequações (SEI) de tipos. Seja um SEI $\{M_{01} = N_{01}, M_{02} = N_{02}, \ldots, M_{0n} = N_{0n}, M_1 \geqslant N_1, M_2 \geqslant N_2, \ldots, M_m \geqslant N_m\}$. Um semi unificador S e um conjunto de m substituições $\{S_1, S_2, \ldots, S_m\}$ para esse SEI são tais que:

$$S(M_{0i}) = S(N_{0i})$$
$$S(M_j) \geqslant S(N_j)$$
$$S_j((S(M_j)) = S(N_j)$$

Para i de 1 a n e j de 1 a m.

Semi Unificação $\leqslant_R$ Inferência de tipos com RP

Kfoury et al. (1993) provou que o problema da semi unificação é indecidível.

Será mostrada uma função computável que transforma uma instância do problema da semi unificação - um SEI - em uma expressão do sistema Milner-Mycroft. Essa função é de tal forma que o SEI será semi unificável se e somente se a expressão for tipável no sistema.

Milner-Mycroft Extendido

$$\frac{(\tau, \vec{\tau}) \geqslant (\tau', \vec{\tau})}{\Gamma \cup \{x : \tau\}, \vec{\tau} \vdash x : \tau'} \; (TAUT')$$

$$\frac{\Gamma \cup \{x : \tau_x\}, (\vec{\tau}, \tau_x) \vdash e : \tau \qquad \tau' = \tau_x \to \tau}{\Gamma, \vec{\tau} \vdash \lambda x.e : \tau'} \; (ABS')$$

$$\frac{\Gamma, \vec{\tau} \vdash e : \tau \qquad \Gamma, \vec{\tau} \vdash e' : \tau' \qquad \tau = \tau' \to \tau''}{\Gamma, \vec{\tau} \vdash (ee') : \tau''} \; (APP')$$

$$\frac{\Gamma, \vec{\tau} \vdash e : \tau \qquad \Gamma \cup \{x : \tau_x\}, \vec{\tau} \vdash e' : \tau' \qquad \tau = \tau', \tau' = \tau''}{\Gamma, \vec{\tau} \vdash \; \textbf{let } x = e \textbf{ in } e' : \tau''} \; (LET')$$

$$\frac{\Gamma \cup \{x : \tau_x\}, \vec{\tau} \vdash e : \tau \qquad \tau_x = \tau, (\tau, \vec{\tau}) \geqslant (\tau', \vec{\tau})}{\Gamma, \vec{\tau} \vdash \textbf{fix } x.e : \tau'} \; (FIX{-}P')$$

Pares dependentes

$$(e, e1) = \lambda x.xee'$$
$$e.1 = e(\lambda x.\lambda y.x)$$
$$e.2 = e(\lambda x.\lambda y.y)$$

com $x \notin ftv(e) \cup ftv(e')$. As regras para derivação de tipos são definidas:

$$\frac{\Gamma \vdash e : \tau \quad \Gamma \vdash e' : \tau'}{\Gamma \vdash (e, e') : (\tau, \tau')_{\tau''}} \ (PAIR')$$

$$\frac{\Gamma \vdash \ e : (\tau, \tau')_{\tau}}{\Gamma \vdash e.1 : \tau} \ (PROJ') \ \frac{\Gamma \vdash \ e : (\tau, \tau')_{\tau'}}{\Gamma \vdash e.2 : \tau'}$$

Escreve-se $e \doteq e'$ para a expressão $\lambda y.(ye, ye')$ com $y \notin ftv(e) \cup ftv(e')$.

Conversão de Equações

Dada uma equação de tipos $M = N$, constroe-se um contexto Γ e uma expressão $\lambda\vec{x}.(m \doteq n)$ onde m e n são expressões tal que $\Gamma \vdash m : M$, $\Gamma \vdash n : N$, e $\vec{x} = ftv(M) \cup ftv(N)$. Ou seja, $M = N$ é semi unificável se e somente se $\lambda\vec{x}.(m \doteq n)$ é tipável. Como $M = N$ é uma equação, seu semi unificador é igual ao seu unificador, formando o teorema a seguir.

Theorem

$M = N$ *é unificável* $\iff$ $\lambda\vec{x}.(m \doteq n)$ *é tipável*

Conversão de Inequações

Sem perder generalidade, conforme Henglein (1993), é possível reduzir a prova e mostrar a conversão de um sistema de apenas duas inequações. Assim, um sistema de inequações $M_1 \geqslant N_1, M_2 \geqslant N_2$ é convertido para uma expressão do sistema Milner-Mycroft da forma.

$$e = \mathbf{fix} f.\lambda\vec{x}.K(m_1 \doteq m_2)(\lambda\vec{y}.((f\,\vec{y}).1 \doteq n_1), \lambda\vec{y}.((f\,\vec{y}).2 \doteq n_2))$$

Onde $\vec{x} = ftv(M_1) \cup ftv(M_2) \cup ftv(N_1) \cup ftv(N_2)$ e $K = \lambda x.\lambda y.x$. Assim como um contexto Γ tal que $\Gamma \vdash m_1 : M_1, \Gamma \vdash m_2 : M_2, \Gamma \vdash n_1 : N_1$ e $\Gamma \vdash n_2 : N_2$.

Theorem

$\{M_1 \geqslant N_1, M_2 \geqslant N_2\}$ *é semi unificável* $\iff$ *e é tipável*

Conclusão

Semi-algoritmos

Não param respondendo corretamente para todas as instâncias do problema.

Fim =)