Inferência de typedef's para C

Rodrigo Ribeiro

24 de agosto de 2015

Sumário

Visão Geral do Algoritmo

- Algoritmo baseado em inferência de tipos para ML
- Dividido em duas etapas
  - Gerar restrições
  - Solucionar restrições
- Solução das restrições produz, como efeito colateral, definições de typedefs.

Metavariáveis

Símbolo	Significado
l	literal
x	variável
f	função
τ	tipo
$\circ$	operador binário qualquer
$\rho : l \to \tau$	função para atribuir tipos a literais

Expressões

$$
\begin{array}{lcll}
e & ::= & l & \{\text{literal}\} \\
 & \mid & x & \{\text{variável}\} \\
 & \mid & e.x & \{\text{acesso a campo}\} \\
 & \mid & e\,[e_1] & \{\text{accesso arranjo}\} \\
 & \mid & (\tau)\;e & \{\text{casting}\} \\
 & \mid & e \circ e' & \{\text{bin op}\} \\
 & \mid & \&\,e & \{\text{address}\} \\
 & \mid & \star\,e & \{\text{deref. ponteiro}\} \\
 & \mid & e \rightarrow x & \{\text{ref. campo ponteiro}\} \\
 & \mid & f\,(e_i)^{i=0..n} & \{\text{chamada função}\}
\end{array}
$$

Comandos

$$
\begin{array}{rclcll}
s & ::= & \tau\, x & = & e & \{\text{var. def.}\} \\
 & | & x & = & e & \{\text{atrib. var.}\} \\
 & | & \star x & = & e & \{\text{atrib. pont.}\} \\
 & | & x[e] & = & e & \{\text{atrib. arranjo}\}
\end{array}
$$

Declarações

- typedefs são apenas um par formado por um tipo e um nome.
- Sintaxe de funções
  - Formada por: tipo de retorno, nome, parâmetros, comandos
  - Último comando deve ser um retorno (restrição para facilitar geração de restrições).

Tipos

$$
\begin{array}{lcll}
\tau & ::= & \mathbf{B} & \{\text{tipos básicos: void, int, etc.}\} \\
 & \mid & \star\,\tau & \{\text{ponteiros}\} \\
 & \mid & \{x_i : \tau_i\}^{i=1..n} & \{\text{registros}\} \\
 & \mid & \tau[n] & \{\text{arranjo}\} \\
 & \mid & \tau^{0..n} \rightarrow \tau & \{\text{tipos de funções}\}
\end{array}
$$

Sintaxe de restrições

- Existência de declaração: $def\ x : \tau$.
- Igualdade de tipos: $\tau \equiv \tau'$.
- Existencia de campos: $has(\tau, x : \tau')$.
- Arranjos: $\tau \equiv \tau'[]$.
- Pointeiros: $\tau \equiv \star\tau'$.

Geração de restrições para expressões

$$
\begin{array}{lcl}
\langle\langle\, l : \tau \,\rangle\rangle & = & \rho(l) \equiv \tau \\
\langle\langle\, x : \tau \,\rangle\rangle & = & x \equiv \tau \\
\langle\langle\, \circ : \tau \,\rangle\rangle & = & \circ \equiv \tau \\
\langle\langle\, f : \tau^{0..n} \rightarrow \tau' \,\rangle\rangle & = & f \equiv \tau^{0..n} \rightarrow \tau' \\
\langle\langle\, e.x : \tau \,\rangle\rangle & = & \exists \alpha_1\, \alpha_2 . \langle\langle\, e : \alpha_1 \,\rangle\rangle \wedge \langle\langle\, x : \alpha_2 \,\rangle\rangle \wedge \\
 & & has(\alpha_1, x : \alpha_2) \wedge \tau \equiv \alpha_2 \\
\langle\langle\, e[e_1] : \tau \,\rangle\rangle & = & \exists \alpha_1 \alpha_2 \alpha_3 . \langle\langle\, e : \alpha_1 \,\rangle\rangle \wedge \langle\langle\, e_1 : \alpha_2 \,\rangle\rangle \wedge \\
 & & \alpha_1 \equiv \alpha_3[] \wedge \tau \equiv \alpha_3 \\
\langle\langle\, (\tau')\, e : \tau \,\rangle\rangle & = & \langle\langle\, e : \tau \,\rangle\rangle \wedge \tau \equiv \tau' \\
\langle\langle\, f(e^{i=0..n}) : \tau \,\rangle\rangle & = & \exists \alpha^{i=0..n} . \bigwedge_{i=0..n} \langle\langle\, e_i : \alpha_i \,\rangle\rangle \wedge \langle\langle\, f : \alpha^{i=0..n} \rightarrow \tau \,\rangle\rangle
\end{array}
$$

Geração de restrições para expressões

$$
\begin{array}{lcl}
\langle\langle\, e \circ e' : \tau \,\rangle\rangle & = & \exists \alpha_1 \alpha_2 . \langle\langle\, e : \alpha_1 \,\rangle\rangle \wedge \langle\langle\, e' : \alpha_2 \,\rangle\rangle \wedge \langle\langle\, \circ : \alpha_1 \rightarrow \alpha_2 \rightarrow \tau \,\rangle\rangle \\
\langle\langle\, \& e : \tau \,\rangle\rangle & = & \exists \alpha \alpha' . \langle\langle\, e : \alpha \,\rangle\rangle \wedge \alpha \equiv \star\alpha' \wedge \tau = \mathit{int} \\
\langle\langle\, \star e : \tau \,\rangle\rangle & = & \exists \alpha . \langle\langle\, e : \alpha \,\rangle\rangle \wedge \alpha = \star\tau \\
\langle\langle\, e \rightarrow x : \tau \,\rangle\rangle & = & \exists \alpha_1 \alpha_2 \alpha_3 . \langle\langle\, e : \alpha_1 \,\rangle\rangle \wedge \langle\langle\, x : \alpha_3 \,\rangle\rangle \wedge \alpha_1 = \star\alpha_2 \wedge \\
 & & \mathit{has}(\alpha, x : \alpha_3) \wedge \tau \equiv \alpha_3
\end{array}
$$

Geração de restrições para comandos

$$
\begin{array}{lcl}
\langle\langle\, \emptyset \,\rangle\rangle & = & \mathit{true} \\
\langle\langle\, \tau\; x := e; S \,\rangle\rangle & = & \exists\alpha.\langle\langle\, e : \alpha \,\rangle\rangle \wedge \mathit{def}\; \tau\; \mathit{in}\; \mathit{def}\; x : \tau\; \mathit{in}\, \langle\langle\, S \,\rangle\rangle \wedge \tau \equiv \alpha \\
\langle\langle\, x := e; S \,\rangle\rangle & = & \exists\alpha.\langle\langle\, x : \alpha \,\rangle\rangle \wedge \langle\langle\, e : \alpha \,\rangle\rangle \wedge \langle\langle\, S \,\rangle\rangle \\
\langle\langle\, \star x := e; S \,\rangle\rangle & = & \exists\alpha\alpha'.\langle\langle\, x : \alpha' \,\rangle\rangle \wedge \langle\langle\, e : \alpha \,\rangle\rangle \wedge \alpha' \equiv \star\alpha) \wedge \langle\langle\, S \,\rangle\rangle \\
\langle\langle\, x[e'] := e; S \,\rangle\rangle & = & \exists\alpha_1\alpha_2\alpha_3.\langle\langle\, e' : \alpha_1 \,\rangle\rangle \wedge \alpha_1 = \mathit{int} \wedge \langle\langle\, e : \alpha_2 \,\rangle\rangle \wedge \\
 & & \langle\langle\, x : \alpha_3 \,\rangle\rangle \wedge \alpha_3 \equiv \alpha_2[]
\end{array}
$$

Geração de restrições para funções e definições

$$
\begin{array}{lcl}
\langle\langle\, \tau\ f(\tau_i\ x_i)^{i=0..n}\ \ S\ \ e\,\rangle\rangle & = & \exists\alpha.def\ f : \tau^{i=0..n} \rightarrow \tau\ \ in\ \ def\ \ x_i : \tau_i^{i=0..n}\ in \\
 & & \langle\langle\, S\,\rangle\rangle \wedge \langle\langle\, e : \alpha\,\rangle\rangle \wedge \alpha \equiv \tau \\
\langle\langle\, typedef\ \ \tau\ x\,\rangle\rangle & = & def\ x\ =\ \tau
\end{array}
$$

Solucionando restrições — Visão geral

- Restrições são basicamente restrições de igualdade e, portanto, podem ser resolvidas por unificação.
  - Quantificadores existenciais correspondem a variáveis "fresh", i.e., novas variáveis.
  - Restrições de definição substituem o nome de um identificador pelo tipo associado a este.
  - Restrições de igualdade: resolvidas por unificação.

Solucionando restrições — Visão geral

- Restrições de campos
  - Consideradas após a solução de todas as restrições de igualdade.
  - Substituições geradas por unificação devem ser aplicadas a restrições ainda não processadas.

Solucionando restrições — Unificação

Limitações conhecidas

- Dimensão de arranjos não são tratadas.
- Ponteiros para funções.
- Algo que ainda não percebi? :)