INF05516 - Semântica formal N
Ciência da Computação - UFRGS
2006-2

Marcus Ritt
mrpritt@inf.ufrgs.br

09/05/2006

Introdução

2 / 20

Agenda

Última aula:

- Revisão prova 1.

Hoje:

- Semântica denotational do IMP.

Semântica denotational

4 / 20

Introdução

- A semântica denotational mapa categorias sintáticas para categorias semânticas.
- A categoria semântica (a denotação) é um objeto matemático.
- A semântica denotational também foi chamada *semântica matemática*.
- Uma característica importante é que a denotação de uma entidade sintática e dado usando as denotações das componentes: A semântica denotational é *compositional*.

Dana Stewart Scott (*1932)

Christopher Strachey (*1916, +1975)

Porque semântica denotational

- Um outro ponto de vista: o comportamento de um programa é uma função parcial (porque parcial?)
- Se conseguimos definir a função de um programa, livramos-nos completamente das detalhes do execução!
- Nesse sentido a denotação descreve o "significado puro" de uma linguagem.
- Logo, é uma abordagem prometida para um entendimento mais profundo da linguagem em consideração.

Notação

- O mapeamento entre a sintaxe é a semântica usa a notação $[\![\cdot]\!]$.
- Os *chaves empáticas* $[\![\cdot]\!]$ separam a sintaxe da semântica.
- Por exemplo, que seria a semântica de um numeral $\mathbf{n}$?
- Um número inteiro $\in \mathbb{Z}$. Logo escrevemos

$$[\![\mathbf{n}]\!] = n.$$

- Observe que $\mathbf{n}$ denota a sintaxe e n a semântica (um objeto matemático)!

Exemplo: Expressões aritméticas

- Em geral, uma semântica adequada para expressões aritméticas é um número inteiro $\in \mathbb{Z}$.
- Logo, definimos

$$
\begin{aligned}
\mathcal{A}[\![\mathbf{n}]\!] &= n \\
\mathcal{A}[\![\mathbf{a} + \mathbf{a}']\!] &= A[\![\mathbf{a}]\!] + A[\![\mathbf{a}']\!] \\
\mathcal{A}[\![\mathbf{a} - \mathbf{a}']\!] &= A[\![\mathbf{a}]\!] - A[\![\mathbf{a}']\!] \\
\mathcal{A}[\![\mathbf{a} \times \mathbf{a}']\!] &= A[\![\mathbf{a}]\!] \times A[\![\mathbf{a}']\!]
\end{aligned}
$$

- Observações:
  - Não definimos a semântica de variáveis ainda.
  - Nesse nível a semântica ainda é bem paracido com a semântica operacional.

Plano

- Seguindo essas idéias a denotação de frases de IMP seriam objetos matemáticos.
- Para completar a semântica, temos que considerar o estado também (por exemplo $x + 3$ depende do valor atual de x).
- O que seria uma denotação adequada?
- O plano então é o seguinte: Definir toda a semântica de IMP usando um modelo matemático, com a denotação de
  - expressões aritméticas sendo elementos de $[\Sigma \to \mathbb{Z}]$;
  - de expressões booleanas sendo elementos de $[\Sigma \to \mathrm{Bool}]$;
  - de comandos sendo funções parcias de $[\Sigma \rightharpoonup \Sigma]$.

Funções semânticas

- Temos três categorias sintaticas é semânticas diferentes.
- Logo, a definição da semântica consiste em três mapeamentos (funções) diferentes da sintaxe à semântica:

$$\begin{array}{rcl} A[\![\cdot]\!] & \in & \text{Aexp} \to (\Sigma \to \mathbb{Z}) \\ B[\![\cdot]\!] & \in & \text{Bexp} \to (\Sigma \to \text{Bool}) \\ C[\![\cdot]\!] & \in & \text{Com} \to (\Sigma \rightharpoonup \Sigma) \end{array}$$

Definições

- *Equações semânticas* definam essas funções

$$\begin{array}{rcl} A[\![a]\!]\sigma & = & \ldots \\ B[\![b]\!]\sigma & = & \ldots \\ C[\![c]\!]\sigma & = & \ldots \end{array}$$

- ou, usando a notação lambda

$$\begin{array}{rcl} A[\![a]\!] & = & \lambda\sigma \in \Sigma. \quad \ldots \\ B[\![b]\!] & = & \lambda\sigma \in \Sigma. \quad \ldots \\ C[\![c]\!] & = & \lambda\sigma \in \Sigma. \quad \ldots \end{array}$$

Expressões aritméticas

O significado de uma expressão aritmética é uma função em $[\Sigma \to \mathbb{Z}]$:

$$
\begin{aligned}
A[\![\mathbf{n}]\!] &= \lambda\sigma \in \Sigma.n && \text{(eqa1)}\\
A[\![\mathbf{x}]\!] &= \lambda\sigma \in \Sigma.\sigma(x) && \text{(eqa2)}\\
A[\![\mathbf{a_1+a_2}]\!] &= \lambda\sigma \in \Sigma.A[\![\mathbf{a_1}]\!]\sigma + A[\![\mathbf{a_2}]\!]\sigma && \text{(eqa3)}\\
A[\![\mathbf{a_1-a_2}]\!] &= \lambda\sigma \in \Sigma.A[\![\mathbf{a_1}]\!]\sigma - A[\![\mathbf{a_2}]\!]\sigma && \text{(eqa4)}\\
A[\![\mathbf{a_1*a_2}]\!] &= \lambda\sigma \in \Sigma.A[\![\mathbf{a_1}]\!]\sigma \times A[\![\mathbf{a_2}]\!]\sigma && \text{(eqa5)}
\end{aligned}
$$

Expressões booleanas

O significado de uma expressão booleana é uma função em $[\Sigma \to \mathrm{Bool}]$:

$$
\begin{aligned}
B[\![\mathbf{t}]\!] &= \lambda\sigma \in \Sigma.t && \text{(eqb1)}\\
B[\![\mathbf{a_1=a_2}]\!] &= \lambda\sigma \in \Sigma.A[\![\mathbf{a_1}]\!]\sigma = A[\![\mathbf{a_2}]\!]\sigma && \text{(eqb2)}\\
B[\![\mathbf{a_1<a_2}]\!] &= \lambda\sigma \in \Sigma.A[\![\mathbf{a_1}]\!]\sigma < A[\![\mathbf{a_2}]\!]\sigma && \text{(eqb3)}\\
B[\![\neg\mathbf{b}]\!] &= \lambda\sigma \in \Sigma.\neg B[\![\mathbf{b}]\!]\sigma && \text{(eqb4)}\\
B[\![\mathbf{b_1}\wedge\mathbf{b_2}]\!] &= \lambda\sigma \in \Sigma.B[\![\mathbf{b_1}]\!]\sigma \wedge B[\![\mathbf{b_2}]\!]\sigma && \text{(eqb5)}\\
B[\![\mathbf{b_1}\vee\mathbf{b_2}]\!] &= \lambda\sigma \in \Sigma.B[\![\mathbf{b_1}]\!]\sigma \vee B[\![\mathbf{b_2}]\!]\sigma && \text{(eqb6)}
\end{aligned}
$$

Comandos simples

O significado de uma comando é uma função (parcial!) em $[\Sigma \rightharpoonup \Sigma]$:

$$\begin{aligned} C[\![\texttt{skip}]\!] &= \mathrm{id} && \text{(eqc1)} \\ C[\![\texttt{x:=a}]\!] &= \lambda\sigma \in \Sigma.\sigma[x \mapsto A[\![\mathbf{a}]\!]\sigma] && \text{(eqc2)} \\ C[\![\mathbf{c_1}\,\texttt{;}\,\mathbf{c_2}]\!] &= C[\![\mathbf{c_2}]\!] \circ C[\![\mathbf{c_1}]\!] && \text{(eqc3)} \end{aligned}$$

com a abreviação

$$\mathrm{id} \equiv \lambda\sigma \in \Sigma.\sigma.$$

Exemplo

$$C[\![\texttt{x:=1;skip;y:=2}]\!] = ?$$

$$\begin{aligned} C[\![\texttt{x:=1;skip;y:=2}]\!] &= C[\![\texttt{skip;y:=2}]\!] \circ C[\![\texttt{x:=1}]\!] \\ &= (C[\![\texttt{y:=2}]\!] \circ C[\![\texttt{skip}]\!]) \circ C[\![x := 1]\!] \\ &= (C[\![\texttt{y:=2}]\!] \circ \mathrm{id}) \circ C[\![\texttt{x:=1}]\!] \\ &= C[\![\texttt{y:=2}]\!] \circ C[\![\texttt{x:=1}]\!] \\ &= \lambda\sigma \in \Sigma.\sigma[y \mapsto 2] \circ \lambda\sigma.\sigma[x \mapsto 1] \\ &= \lambda\sigma \in \Sigma.\sigma[x \mapsto 1][y \mapsto 2] \end{aligned}$$

Condicional

Para definir um condicional usamos $\mathrm{cond} \in [[\Sigma \to \mathrm{Bool}] \times [\Sigma \rightharpoonup \Sigma] \times [\Sigma \rightharpoonup \Sigma] \to [\Sigma \rightharpoonup \Sigma]]$ tal que

$$\mathrm{cond}(c,t,e)\sigma = \begin{cases} t\sigma & c\,\sigma = \mathrm{true} \\ e\sigma & c\,\sigma = \mathrm{false} \end{cases}$$

$$\mathcal{C}[\![\texttt{if b then c else c'}]\!] = \mathrm{cond}(B[\![b]\!], C[\![c]\!], C[\![c']\!])$$

Simplificações

- Qual é a definição da composição para funções parcias $f \circ g$?

$$(f \circ g)(x) = \begin{cases} f(g(x)) & \text{se } g(x) \text{ e } f(g(x)) \text{ são definidos} \\ \text{não definido} & \text{se } g(x) \text{ é definido, mas } f(g(x)) \text{ não} \\ \text{não definido} & \text{se } g(x) \text{ não é definido} \end{cases}$$

- Obtemos um jeito mais simples de tratar a composição, usando um valor especial $\perp$ para "não definido"
- Usando $\Sigma_\perp = \Sigma \cup \{\perp\}$, podemos definir o significado de um comando como elemento de

$$[\Sigma \to \Sigma_\perp]$$

- A função agora é total!

Simplificações

- Obtemos a nova definição

$$(f \circ g)(x) = \begin{cases} f(g(x)) & \text{se } g(x) \text{ é definidos} \\ \text{não definido} & \text{se } g(x) \text{ não é definido} \end{cases}$$

- Se f aceitasse $\perp$ como argumento tal que $f(\perp) = \perp$ a definição ficaria ainda mais simples

$$(f \circ g)(x) = f(g(x)).$$

- Logo, modificamos outra vez o significado de um comando: Ele seja um elemento

$$f \in [\Sigma_\perp \to \Sigma_\perp]$$

  tal que f é *estrita*: $f(\perp) = \perp$.
- Para a função totalmente indefinido, escrevemos $\perp_{[\Sigma_\perp \to \Sigma_\perp]}$.

O laço while

- Qual seria uma definição boa para o laço while?
- Uma primeira tentativa de definir a semântica do laço `while` pode usar

$$\texttt{while} \sim \texttt{if b then (c;while b do c) else skip}.$$

- Com isso, chegamos em

$$\begin{aligned} [\![\texttt{while b do c}]\!] &= [\![\texttt{if b then (c;while b do c) else skip}]\!] \\ &= \mathrm{cond}([\![\mathrm{b}]\!], [\![\texttt{c;while b do c}]\!], [\![\texttt{skip}]\!]) \\ &= \mathrm{cond}([\![\mathrm{b}]\!], [\![\texttt{while b do c}]\!] \circ [\![\mathrm{c}]\!], \mathrm{id}) \end{aligned}$$

- Ou: "Se $[\![\mathrm{b}]\!]\sigma$ então $([\![\texttt{while b do c}]\!] \circ [\![\mathrm{c}]\!])\sigma$ senão σ".

O laço while

Problemas com isso:

- O significado do `while` é definido usando si mesmo: a definição não é compositional.
- Queremos definir uma função, usando a mesma função na definição!
- Na semântica operacional, uma definição similar não tem problemas.
  - Na SON: não obtemos provas para laçõs infinitos.
  - Na SOE: temos derivações infinitas para laços infinitos.
- Para definir um objeto matemático, precisamos mais cautela.
- Temos uma equação para uma função não conhecida:
  - A equação tem uma solução (senão: o que significa o laço)?
  - A equação tem uma solução única (senão: qual solução é o significado do laço)?