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

Marcus Ritt
mrpritt@inf.ufrgs.br

09/10/2006

Agenda

Última aula:

- Semântica axiomática: Laços. Exemplos.

Hoje:

- Semântica axiomática: Consistência e completude. Corretude total.

Exemplo

Entrada x
Pre-condição $x \geq 0$
Saida r
Pos-condição ?

```
s := 0
r := 0
while s<x do (
  s := s + 2*r + 1;
  r := r + 1
)
```

Exemplo...

Invariante? $(r^2 = s) \wedge (0 \leq r < \sqrt{x} + 1)$

$$\{x \geq 0\}$$
$$\{(0 = 0) \wedge (0 < \sqrt{x} + 1)\}$$
```
s:=0
```
$$\{(0 = s) \wedge 0 \leq 0 < \sqrt{x} + 1\}$$
```
r:=0;
```
$$\{(r^2 = s) \wedge (0 \leq r < \sqrt{x} + 1)\}$$
```
while s<x
```
$$\{(r^2 = s) \wedge (0 \leq r < \sqrt{x} + 1) \wedge (s < x)\}$$
$$\{(r^2 + 2r + 1 = s + 2r + 1) \wedge (0 \leq r < \sqrt{x})\}$$
$$\{((r + 1)^2 = s + 2 * r + 1) \wedge (0 \leq r + 1 < \sqrt{x} + 1)\}$$
```
s:=s+2r+1
```
$$\{((r + 1)^2 = s) \wedge (0 \leq r + 1 < \sqrt{x} + 1)\}$$
```
r:=r+1
```
$$\{(r^2 = s) \wedge (0 \leq r < \sqrt{x} + 1)\}$$
```
)
```
$$\{(r^2 = s) \wedge (0 \leq r < \sqrt{x} + 1) \wedge (s \geq x)\}$$
$$\{\sqrt{x} \leq r < \sqrt{x} + 1\}$$
$$\{r = \lceil \sqrt{x} \rceil\}$$

Consistência e completude

Os noções

- Lembre-se da lógica: Um conjunto de regras é
  - *consistente*, se podemos provar somente características semânticamente corretas.
  - *completo*, se podemos provar qualquer característica semânticamente correta.
- Essas características também são desejaveis na semântica axiomática.

Relações

- Considerando a semântica operacional temos a relação

$$\models \{\Phi\}\mathbf{c}\{\Psi\} \leftrightarrow \forall\sigma \in \Sigma : \forall I \in [V \to \mathbb{Z}] : \sigma \models^I \{\Phi\}\mathbf{c}\{\Psi\}$$

- Considerando nosso sistema de provas usamos

$$\vdash \{\Phi\}\mathbf{c}\{\Psi\}$$

se é possível de *provar* $\{\Phi\}\mathbf{c}\{\Psi\}$ usando as regras da semântica axiomática.
- Logo, a perguntas em aberta são
  - O semântica axiomática é consistente?

$$\vdash \{\Phi\}\mathbf{c}\{\Psi\} \Rightarrow \models \{\Phi\}\mathbf{c}\{\Psi\}?$$

  - O semântica axiomática é completa?

$$\models \{\Phi\}\mathbf{c}\{\Psi\} \Rightarrow \vdash \{\Phi\}\mathbf{c}\{\Psi\}?$$

Completude

- Da lógica sabemos que a lógica de predicados com expressões aritméticas não é decidível (Gödel).
- Logo, a linguagem de asserções não é completo.
- Conseqüentemente, nosso sistema de regras não pode ser completo também.
- Mas é possível de provar a completude relativa:
- Supondo a decibilidade da linguagem de asserções a semântica axiomática é completo.

Regras para corretude total

$$\frac{}{\{\Phi\}\mathtt{skip}\{\Downarrow\Phi\}}\ \text{skip}$$

$$\frac{}{\{\Phi[a/l]\}\mathbf{l}\mathtt{:=}\mathbf{a}\{\Downarrow\Phi\}}\ \text{assign}$$

$$\frac{\{\Phi\}\mathbf{c_1}\{\Downarrow\chi\} \quad \{\chi\}\mathbf{c_2}\{\Downarrow\Psi\}}{\{\Phi\}\mathbf{c_1}\mathtt{;}\mathbf{c_2}\{\Downarrow\Psi\}}\ \text{seq}$$

$$\frac{\{\Phi\wedge b\}\mathbf{c_1}\{\Downarrow\Psi\} \quad \{\Phi\wedge\neg b\}\mathbf{c_2}\{\Downarrow\Psi\}}{\{\Phi\}\mathtt{if}\ \mathbf{b}\ \mathtt{then}\ \mathbf{c_1}\ \mathtt{else}\ \mathbf{c_2}\{\Downarrow\Psi\}}\ \text{if}$$

$$\frac{\{\Phi\wedge b\wedge 0\leq t=z\}\mathbf{c}\{\Phi\wedge 0\leq t<z\}}{\{\Phi\wedge 0\leq t\}\mathtt{while}\ \mathbf{b}\ \mathtt{do}\ \mathbf{c}\{\Downarrow\Phi\wedge\neg b\}}\ \text{while}$$

$$\frac{\vdash\Phi\rightarrow\Phi' \quad \{\Phi'\}\mathbf{c}\{\Downarrow\Psi'\} \quad \vdash\Psi'\rightarrow\Psi}{\{\Phi\}\mathbf{c}\{\Downarrow\Psi\}}\ \text{impl}$$

Corretude total

Corretude parcial + Terminação = Corretude total

- Analisando as regras é obviou que só o laço é um fonte de não-terminação.
- Por isso, temos que modificar só a regra de while.

$$\frac{\{\Phi\wedge b\wedge 0\leq t=z\}\mathbf{c}\{\Phi\wedge 0\leq t<z\}}{\{\Phi\wedge 0\leq t\}\mathtt{while}\ \mathbf{b}\ \mathtt{do}\ \mathbf{c}\{\Downarrow\Phi\wedge\neg b\}}\ \text{while}$$

- A expressão t é a *variante* do laço.

Exemplo: Fatorial

Entrada x
Pre-condição $x \geq 0 \wedge x = n$
Saida y
Pos-condição $y = n!$

```
y := 1
while ¬(x=0) do
  y := y∗x;
  x := x − 1
```


Exemplo...

Invariante, Variante? $yx! = n!$, x.

$$\{x = n \wedge x \geq 0\}$$
$$\{x! = n! \wedge 0 \leq x\}$$

```
y := 1;
```

$$\{\Downarrow yx! = n! \wedge 0 \leq x\}$$

```
while ¬(x=0) do (
```

$$\{yx! = n! \wedge 0 \leq x = x_0 \wedge x \neq 0\}$$
$$\{yx! = n! \wedge 0 \leq x - 1 < x_0\}$$
$$\{(yx)(x-1)! = n! \wedge 0 \leq x - 1 < x_0\}$$

```
y := y*x;
```

$$\{y(x-1)! = n! \wedge 0 \leq x - 1 < x_0\}$$

```
x := x-1
```

$$\{yx! = n! \wedge 0 \leq x < x_0\}$$

```
)
```

$$\{\Downarrow yx! = n! \wedge x = 0\}$$
$$\{\Downarrow y = n!\}$$

Exemplo: Fatorial 2

Entrada x
Pre-condição $x \geq 0$
Saida y
Pos-condição $y = x!$

```
y := 1;
z := 0;
while ¬(x=z) do
  z := z+1;
  y := y*z
```


Exemplo...

Invariante, Variante? $y = z!$, $x - z$.

$$\{x \geq 0\}$$
$$\{1 = 0! \wedge 0 \leq x - 0\}$$
`y := 1;`
$$\{\Downarrow y = 0! \wedge 0 \leq x - 0\}$$
`z := 0;`
$$\{\Downarrow y = z! \wedge 0 \leq x - z\}$$
`while ¬(x=z) do (`
$$\{y = z! \wedge 0 \leq x - z = t_0\}$$
$$\{y * (z+1) = (z+1)! \wedge 0 \leq x - (z+1) < t_0\}$$
`z := z+1;`
$$\{y * z = z! \wedge 0 \leq x - z < t_0\}$$
`y := y*z`
$$\{y = z! \wedge 0 \leq x - z < t_0\}$$
`)`
$$\{\Downarrow y! = z! \wedge x = z\}$$
$$\{\Downarrow y = x!\}$$

Busca binária

Entrada Campo $t_1, \ldots, t_n$, com $n \geq 0$, elemento b.
Pre-condição $t_i < t_j$ para $i < j$.
Saida $a \in \{\mathrm{true}, \mathrm{false}\}$.
Pos-condição $a = \mathrm{true} \leftrightarrow \exists i : a = t_i$.

Exemplos

```
i:=1; j:=n;
while i≠j do (
  m := (i+j)/2;
  if t_m <= x then
    i:=m
  else
    j:=m
)
if t_i=x then
  a:=true
else
  a:=false
```

```
i:=0; j:=n;
while i≠j do (
  m := (i+j+1)/2;
  if t_m <= x then
    i:=m+1
  else
    j:=m
)
if i>=1 ∧ i<=n then
  if t_i=x then
    a:=true
  else
    a:=false
else
  a:=false
```

```
i:=1; j:=n;
while j>=i do (
  m := (i+j)/2;
  if x=t_m then
    return true;
  if x<t_m then
    j:=m-1
  else
    i:=m+1
)
```

Projeto por contrato

- O que é software de qualidade?
  - Corretude
  - Robustez
  - Extensibilidade
  - Reuso e compatibilidade
  - Eficiencia
- Uma abordagem no desenvolvimento é *projeto por contrato* (inglês: design by contract).
- As triplas de Hoare servem para a base de um contrato entre o cliente do função e a função.
- Linguagens como Eiffel ou Java (JML) permitem projeto (ou programação) por contrato.

Exemplo: Eiffel

```
fat(x: INTEGER): INTEGER is
  -- fatorial de x
require
  x>=0
local
  y: INTEGER
do
  from
    y:=1
  invariant
    -- y*fat(x) = fat(old x)
  variant
    x
  until
    x=0
  loop
    y := y * x;
    x := x - 1
  end
ensure
  -- y=fat(old x)
end
```

Tempo de execução

- E possível de extender a semântica axiomática para provar asserções sobre o tempo de execução.
- O novos triplas da forma

$$\{\Phi\}\mathbf{c}\{t \Downarrow \Psi\}$$

significam que com pre-condição Φ c termina, satisfaz a pos-condição Ψ e precisa tempo $O(t)$.