Cálculo de Correção parcial $\mathcal{H}$

$[skip_p\ ]$

$$\{\varphi\}\,\texttt{skip}\,\{\varphi\}$$

$[ass_p\ ]$

$$\{\varphi[E/x]\}\,x \leftarrow E\,\{\varphi\}$$

$[comp_p\ ]$

$$\frac{\{\varphi\}\,C_1\,\{\eta\} \qquad \{\eta\}\,C_2\,\{\psi\}}{\{\varphi\}\,C_1; C_2\,\{\psi\}}$$

$[if_p\ ]$

$$\frac{\{\varphi\ \wedge\ B\}\,C_1\,\{\psi\} \qquad \{\varphi\ \wedge\ \neg B\}\,C_2\,\{\psi\}}{\{\varphi\}\,\texttt{if}\,B\,\texttt{then}\,C_1\,\texttt{else}\,C_2\,\{\psi\}}$$

$[if'_p\ ]$

$$\frac{\{\varphi_1\}\,C_1\,\{\psi\} \qquad \{\varphi_2\}\,C_2\,\{\psi\}}{\{(B \to \varphi_1)\ \wedge\ (\neg B \to \varphi_2)\}\,\texttt{if}\,B\,\texttt{then}\,C_1\,\texttt{else}\,C_2\,\{\psi\}}$$

$[while_p\ ]$

$$\frac{\{\psi\ \wedge\ B\}\,C\,\{\psi\}}{\{\psi\}\,\texttt{while}\,B\,\texttt{do}\,C\,\{\psi\ \wedge\ \neg B\}}$$

$[cons_p\ ]$

$$\frac{\vdash \varphi' \to \varphi \qquad \{\varphi\}\,C\,\{\psi\} \qquad \vdash \psi \to \psi'}{\{\varphi'\}\,C\,\{\psi'\}}$$

Exemplos

Exercício 21.1. *Mostrar que¡*

$$\begin{aligned}
&\vdash_p \{\mathsf{true}\}\\
&r \leftarrow x; q \leftarrow 0;\\
&\texttt{while}\,y \leq r\,\texttt{do}\\
&r \leftarrow r - y;\\
&q \leftarrow q + 1\\
&\{r < y\ \wedge\ x = r + (y \times q)\}
\end{aligned}$$

◇

A expressão $x = r + (y \times q)$ é um invariante de ciclo.

Integridade e Completude

Para o sistema dedutivo de Hoare, vamos considerar duas propriedades usuais em sistemas lógicos:

- **Integridade:** Cada regra deve preservar validade. O que implica (por indução nas derivações) que os teoremas obtidos correspondem a asserções válidas de correcção parcial.

$$\vdash_p \{\varphi\}C\{\psi\} \qquad \Rightarrow \qquad \models_p \{\varphi\}C\{\psi\}.$$

- **Completude:** Gostaríamos que o sistema fosse suficientemente forte para inferir todas as asserções de correção parcial válidas.

$$\models_p \{\varphi\}C\{\psi\} \qquad \Rightarrow \qquad \vdash_p \{\varphi\}C\{\psi\}.$$

Vamos começar por formalizar a noção de execução/avaliação.

Estado de execução

Para a avaliação duma expressão é necessário saber o valor das variáveis.

Um **estado** s é uma função que associa a cada variável um valor.

Representamos o conjunto de estados por

$$\mathbf{State} = \mathbf{Var} \to \mathbb{Z}$$

e $s \in \mathbf{State}$ tal que $s : \mathbf{Var} \to \mathbb{Z}$.

Seja $s\ x$ ou $s(x)$ o valor da variável x no estado s. Se $v \in \mathbb{Z}$,

$$s[v/x](y) = \begin{cases} s(y) & \text{se } y \neq x \\ v & \text{se } y = x \end{cases}$$

Semântica das expressões

Aexp - Expressões aritméticas

$\mathcal{A} : \textbf{Aexp} \rightarrow (\textbf{State} \rightarrow Z)$

$$\begin{aligned}
\mathcal{A}[\![n]\!]s &= n \\
\mathcal{A}[\![x]\!]s &= s(x) \\
\mathcal{A}[\![E_1 + E_2]\!]s &= \mathcal{A}[\![E_1]\!]s + \mathcal{A}[\![E_2]\!]s \\
\mathcal{A}[\![E_1 - E_2]\!]s &= \mathcal{A}[\![E_1]\!]s - \mathcal{A}[\![E_2]\!]s \\
\mathcal{A}[\![E_1 \times E_2]\!]s &= \mathcal{A}[\![E_1]\!]s . \mathcal{A}[\![E_2]\!]s
\end{aligned}$$

Semântica das expressões

Bexp - Expressões booleanas

$\mathbf{T} = \{\mathbf{V}, \mathbf{F}\}$

$\mathcal{B} : \textbf{Bexp} \rightarrow (\textbf{State} \rightarrow \mathbf{T})$

$$\begin{aligned}
\mathcal{B}[\![\mathsf{true}]\!]s &= \mathbf{V} \\
\mathcal{B}[\![\mathsf{false}]\!]s &= \mathrm{F} \\
\mathcal{B}[\![E_1 = E_2]\!]s &= \begin{cases} \mathbf{V} & \text{se } \mathcal{A}[\![E_1]\!]s = \mathcal{A}[\![E_2]\!]s \\ \mathbf{F} & \text{se } \mathcal{A}[\![E_1]\!]s \neq \mathcal{A}[\![E_2]\!]s \end{cases} \\
\mathcal{B}[\![E_1 \leq E_2]\!]s &= \begin{cases} \mathbf{V} & \text{se } \mathcal{A}[\![E_1]\!]s \leq \mathcal{A}[\![E_2]\!]s \\ \mathbf{F} & \text{se } \mathcal{A}[\![E_1]\!]s > \mathcal{A}[\![E_2]\!]s \end{cases} \\
\mathcal{B}[\![\neg b]\!]s &= \begin{cases} \mathbf{V} & \text{se } \mathcal{B}[\![b]\!]s = \mathbf{F} \\ \mathbf{F} & \text{se } \mathcal{B}[\![b]\!]s = \mathbf{V} \end{cases} \\
\mathcal{B}[\![b_1 \ \wedge \ b_2]\!]s &= \begin{cases} \mathbf{V} & \text{se } \mathcal{B}[\![b_1]\!]s = \mathbf{V} \text{ e } \mathcal{B}[\![b_2]\!]s = \mathbf{V} \\ \mathbf{F} & \text{se } \mathcal{B}[\![b_1]\!]s = \mathbf{F} \text{ ou } \mathcal{B}[\![b_2]\!]s = \mathrm{F} \end{cases}
\end{aligned}$$

Semântica operacional natural (*big-step*)

Descreve a execução completa de cada comando.

Configurações: $\langle C, s\rangle$ ou s, onde C é um comando e s um estado $\Gamma = (\textbf{Com} \times \textbf{State}) \cup \textbf{State}$

Configurações Finais: $s \in State$

Transições: $\langle C, s\rangle \ \longrightarrow \ s'$

Regras:

$$\frac{\langle C_1, s_1\rangle \ \longrightarrow \ s'_1 \ ... \ \langle C_n, s_n\rangle \ \longrightarrow \ s'_n}{\langle C, s\rangle \ \longrightarrow \ s'}$$

Hipóteses: $\langle C_i, s_i\rangle \longrightarrow s'_i$
Conclusão: $\langle C, s\rangle \longrightarrow s'$
Se $n = 0$ diz-se um **Axioma**.

Semântica operacional natural (*big-step*)

Semântica operacional para comandos do While

$$
\begin{array}{ll}
\texttt{att}_{sn} & \langle x \leftarrow E, s\rangle \longrightarrow s[\mathcal{A}[\![E]\!]s/x] \\
\texttt{comp}_{sn} & \dfrac{\langle C_1, s\rangle \longrightarrow s' \;,\; \langle C_2, s'\rangle \longrightarrow s''}{\langle C_1; C_2, s\rangle \longrightarrow s''} \\
\texttt{if}^{v}{}_{sn} & \dfrac{\langle C_1, s\rangle \longrightarrow s'}{\langle \texttt{if } B \texttt{ then } C_1 \texttt{ else } C_2, s\rangle \longrightarrow s'} \text{ se } \mathcal{B}[\![B]\!]s = \mathbf{V} \\
\texttt{if}^{f}{}_{sn} & \dfrac{\langle C_2, s\rangle \longrightarrow s'}{\langle \texttt{if } B \texttt{ then } C_1 \texttt{ else } C_2, s\rangle \longrightarrow s'} \text{ se } \mathcal{B}[\![B]\!]s = \mathbf{F} \\
\texttt{while}^{v}{}_{sn} & \dfrac{\langle C, s\rangle \longrightarrow s',\; \langle \texttt{while } B \texttt{ do } C, s'\rangle \longrightarrow s''}{\langle \texttt{while } B \texttt{ do } C, s\rangle \longrightarrow s''} \text{ se } \mathcal{B}[\![B]\!]s = \mathbf{V} \\
\texttt{while}^{f}{}_{sn} & \langle \texttt{while } B \texttt{ do } C, s\rangle \longrightarrow s \text{ se } \mathcal{B}[\![B]\!]s = \mathbf{F}
\end{array}
$$

Exemplos

Sendo $s_0 = [x = 5, y = 7]$ determinar o estado após a execução de:

$$(z \leftarrow x; x \leftarrow y); y \leftarrow z.$$

Para tal constrói-se uma **Árvore de derivação** com esse comando como raiz:

$$\dfrac{\dfrac{\langle z \leftarrow x, s_0\rangle \longrightarrow s_1 \quad \langle x \leftarrow y, s_1\rangle \longrightarrow s_2}{\langle z \leftarrow x; x \leftarrow y, s_0\rangle \longrightarrow s_2} \quad \langle y \leftarrow z, s_2\rangle \longrightarrow s_3}{\langle (z \leftarrow x; x \leftarrow y); y \leftarrow z, s_0\rangle \longrightarrow s_3}$$

onde,

$$
\begin{aligned}
s_1 &= s_0[5/z] \\
s_2 &= s_1[7/x] \\
s_3 &= s_2[5/y]
\end{aligned}
$$

Temos $s_3 = [z = 5, x = 7, y = 5]$.

Integridade da semântica axiomática

Teorema 21.1 (Integridade)**.** *Para todas as asserções de correcção parcial* $\{\varphi\}C\{\psi\}$,

$$\vdash_p \{\varphi\}C\{\psi\} \textit{ implica } \models_p \{\varphi\}C\{\psi\}$$

Isto é se $\vdash_p \{\varphi\}C\{\psi\}$ então

- se para um estado s, $s \models \varphi$
- e se $\langle C, s\rangle \longrightarrow s'$
- então $s' \models \psi$

Integridade da semântica axiomática

A demonstração é por indução na árvore de inferência de $\vdash_p \{\varphi\}C\{\psi\}$:

- Mostrar que a propriedade se verifica para as árvores simples, i.e os **axiomas** do sistema de inferência.
- Mostrar que a propriedade se verifica para as Árvores de inferência compostas: para cada regra, supor que a propriedade se verifica para as premissas (e as condições se verificam) e mostrar que a propriedade também se verifica para a conclusão da regra.

Integridade da semântica axiomática

Caso ass_p**.** Suponhamos que $\vdash_p \{\varphi[E/x]\}x \leftarrow E\{\varphi\}$.

Seja

$$\langle x \leftarrow E, s\rangle \longrightarrow s'$$

e $s \models \varphi[E/x]$. Então ma $s[\mathcal{A}[\![E]\!]s/x] \models \varphi$. (Exercício)

Temos que provar que $s' \models \varphi$.

Por $[ass_{sn}]$ temos que $s' = s[\mathcal{A}[\![E]\!]s/x]$, e portanto

$$s' \models \varphi \text{ sse } s[\mathcal{A}[\![E]\!]s/x] \models \varphi$$

Integridade da semântica axiomática

Caso $comp_p$**.** Por hip. de indução $\models_p \{\varphi\}C_1\{\eta\}$ e $\models_p \{\eta\}C_2\{\psi\}$.

Queremos mostrar que $\models_p \{\varphi\}C_1; C_2\{\psi\}$. Sejam s e s'' estados, tal que $s \models \varphi$ e $\langle C_1; C_2, s\rangle \longrightarrow s''$. Pela regra $[comp_{sn}]$ existe s' tal que

$$\langle C_1, s\rangle \longrightarrow s' \text{ e } \langle C_2, s'\rangle \longrightarrow s''$$

De $\langle C_1, s\rangle \longrightarrow s'$, $s \models \varphi$ e $\models_p \{\varphi\}C_1\{\eta\}$, temos que $s' \models \eta$. De $\langle C_2, s'\rangle \longrightarrow s''$, $s' \models \eta$ e $\models_p \{\eta\}C_2\{\psi\}$, temos que $s'' \models \psi$. Que é o que queríamos.

Integridade da semântica axiomática

Caso if_p**.** Por hip. de indução $\models_p \{B \wedge \varphi\}C_1\{\psi\}$ e $\models_p \{\neg B \wedge \varphi\}C_2\{\psi\}$.

Para provar que

$$\models_p \{\varphi\}\,\texttt{if}\,B\,\texttt{then}\,C_1\,\texttt{else}\,C_2\,\{\psi\}$$

sejam s e s' estados tais que $s \models \varphi$ e $\langle \texttt{if}\,B\,\texttt{then}\,C_1\,\texttt{else}\,C_2, s\rangle \longrightarrow s'$.

Se $\mathcal{B}[\![B]\!]s = \mathbf{V}$ então por $[if_{sn}]$, temos que $\langle C_1, s\rangle \longrightarrow s'$. Então dado que $\models_p \{B \wedge \varphi\}C_1\{\psi\}$, concluímos que $s' \models \psi$.

Analogamente se conclui, caso $\mathcal{B}[\![B]\!]s = \mathrm{F}$.

Integridade da semântica axiomática

Caso $while_p$**.** Por hip. de indução

$$\models_p \{B \wedge \varphi\}C\{\varphi\}. \tag{1}$$

Para provar que

$$\models_p \{\varphi\}\,\texttt{while}\,B\,\texttt{do}\,C\,\{\neg B \wedge \varphi\},$$

sejam s e s'' estados tais que $s \models \varphi$ e

$$\langle \texttt{while}\,B\,\texttt{do}\,C, s\rangle \longrightarrow s''.$$

Temos que mostrar que $s'' \models \neg B \wedge \varphi$. Usamos indução na árvore de derivação da semântica natural.

Integridade da semântica axiomática

Caso $while_p$**.** Há dois casos a considerar, consoante $[while_{sn}]$.

Se $\mathcal{B}[\![B]\!]s = \mathrm{F}$ então $s'' = s$ e $s'' \models (\neg B \wedge \varphi)$.

Senão, $\mathcal{B}[\![b]\!]s = \mathbf{V}$ e existe s' tal que $\langle C, s\rangle \longrightarrow s'$ e $\langle \texttt{while}\,B\,\texttt{do}\,C, s'\rangle \longrightarrow s''$.

Temos que $s \models (B \wedge \varphi)$ e pela hipótese (1) temos que $s' \models \varphi$. Aplicando a hipótese de indução a $\langle \texttt{while}\,B\,\texttt{do}\,C, s'\rangle \longrightarrow s''$, temos que $s'' \models (\neg B \wedge \varphi)$, como queríamos.

Integridade da semântica axiomática

Caso $cons_p$**.** Por hip. de indução

$$\models_p \{\varphi'\}C\{\psi'\},\ \varphi \to \varphi',\ \text{e}\ \psi' \to \psi. \tag{2}$$

Para provar que $\models_p \{\varphi\}C\{\psi\}$, sejam s e s' tal que $s \models \varphi$ e $\langle C, s\rangle \longrightarrow s'$.

Como $s \models \varphi$ e $\varphi \to \varphi'$ então $s \models \varphi'$ e pela hipótese (2), $s' \models \psi'$. Mas como $\psi' \to \psi$, temos que $s' \models \psi$, como queríamos.

Completude da semântica axiomática

Teorema 21.2 (Incompletude de Gödel (1931))**.** *Não existe um sistema de demonstração para lógica para a aritmética (formulas sobre inteiros e operações aritméticas (**PA**) , de tal forma que os teoremas coincidam com as asserções válidas de **PA**.*

Teorema 21.3 (Completude)**.** *Para todas as asserções de correcção parcial* $\{\varphi\}C\{\psi\}$*,*

$$\models_p \{\varphi\}C\{\psi\} \textit{ implica } \vdash_p \{\varphi\}C\{\psi\}$$

Note-se que $\models \psi$, se e só se $\models \{\mathsf{true}\}\texttt{skip}\{\psi\}$. O que significa que a completude de $\vdash_p$ contrariaria o teorema de incompletude de Gödel.

Completude da semântica axiomática

Proposição 21.1. *Não existe um sistema de demonstração para asserções de correcção parcial, de tal forma que os teoremas coincidam com as asserções de correcção parcial válidas.*

Prova: Note-se que

$$\models \{\mathsf{true}\}C\{\mathsf{false}\}$$

se e só se o comando C diverge (não para) em todos os estados.

Um sistema de demonstração para asserções de correcção parcial, poderia ser usado para confirmar que o comando diverge em todos os estados. O que é impossível (pela indecidibilidade do *Halting Problem*).

Completude relativa

Teorema 21.4. *O sistema de prova para correcção parcial é relativamente completo, i.e. para qualquer asserção de correcção parcial* $\{\varphi\}C\{\psi\}$*:*

$$\vdash_p \{\varphi\}C\{\psi\} \textit{ se } \models_p \{\varphi\}C\{\psi\}$$

O resultado de correcção parcial relativa foi estabelecido por S. Cook (1978).

O facto de $\vdash_p \{\varphi\}C\{\psi\}$ ser uma prova depende do facto de certas asserções em **PA** serem válidas.

Para a demonstração de completude relativa ver Capítulo 7 [Winskel].

Cálculo para a correcção total

Na linguagem imperativa apresentada, o único comando que pode levar à não terminação é o comando `while`.

O *cálculo* $\vdash_{tot}$ irá ser igual ao $\vdash_p$ excepto na regra $\texttt{while}_{tot}$.

Para demonstrar que um programa termina temos que lhe associar uma expressão estritamente decrescente, denominada variante.

No caso do `while`, podemos associar uma expressão inteira não negativa e mostrar que em cada iteração o valor dessa expressão diminui, mantendo-se não negativa: temos a certeza que `while` termina pois essa expressão só pode tomar um número finito de valores até chegar a zero!!!

No caso do factorial:

$y \leftarrow 1; z \leftarrow 0;$ `while` $z \neq x$ *do* $(z \leftarrow z+1; y \leftarrow y \times z)$

podemos tomar o variante $x - z$.

Cálculo **para a correcção total**

Lógica de Hoare (correcção total)

As regras ass_{tot}, $comp_{tot}$, if_{tot} e $cons_{tot}$ coincidem com as do *cálculo* para a correcção parcial.

$[while_{tot}]$

$$\frac{\{\eta \wedge B \wedge E \geq 0 \wedge E = e_0\}\, C\, \{\eta \wedge E \geq 0 \wedge E < e_0\}}{\{\eta \wedge E \geq 0\}\, \texttt{while}\, B\, \texttt{do}\, C\, \{\eta \wedge \neg B\}}$$

onde e_0 é uma variável lógica cujo valor é o da expressão E antes da execução do comando C.

Pré condição mais fraca **-** $while_{tot}$

$\{\varphi\}$

$\{\eta \wedge E \geq 0\}$

`while` B `do`

$\{\eta \wedge B \wedge E \geq 0 \wedge E = e_0\}$

C

$\{\eta \wedge E \geq 0 \wedge E < e_0\}$

$\{\eta \wedge \neg B\}$ $\quad while_{tot}$

$\{\psi\}$ $\quad cons_{tot}$

Exemplo

$\vdash_{tot} \{x \geq 0\} y \leftarrow 1; z \leftarrow 0; \texttt{while}\ z \neq x\ do\ (z \leftarrow z+1; y \leftarrow y \times z)\{y = x!\}$

$\{x \geq 0\}$

$\{1 = 0!\ \wedge\ x - 0 \geq 0\}$

$\texttt{y} \leftarrow \texttt{1}$

$\{y = 0!\ \wedge\ x - 0 \geq 0\}$

$\texttt{z} \leftarrow \texttt{0}$

$\{y = z!\ \wedge\ x - z \geq 0\}$ $\qquad ass_{tot}$

$\texttt{while}\ z \neq x\ do$

$\{$

$\{y = z!\ \wedge\ z \neq x\ \wedge\ x - z \geq 0\ \wedge\ x - z = e_0\}$ $\qquad cons_{tot}$

$\{y \times (z+1) = (z+1)!\ \wedge\ x - (z+1) \geq 0\ \wedge\ x - (z+1) < e_0\}$ $\qquad ass_{tot}$

$\texttt{z} \leftarrow \texttt{z} + \texttt{1}$

$\{y \times z = z!\ \wedge\ x - z \geq 0\ \wedge\ x - z < e_0\}$ $\qquad ass_{tot}$

$\texttt{y} \leftarrow \texttt{y} \times \texttt{z}$

$\{y = z!\ \wedge\ x - z \geq 0\ \wedge\ x - z < e_0\}$

$\}$

$\{y = z!\ \wedge\ x = z\}$

$\{y = x!\}$

Como determinar um variante ?

Os variantes são mais difíceis de encontrar que os invariantes...porque não é possível saber genericamente se um programa termina

$\vdash_{tot}\{x > 0\}$

```
c = x
while(c! = 1)do
 if(c%2 == 0)c = c/2
 else c = 3 * c + 1
```

$\{\top\}$

Será que este triplo é válido? Neste caso este triplo só estabelecia a terminação do programa...

Mas não se sabe se termina ou não!

Exemplos

Exercício 21.2. *Mostrar que¡*

$$
\begin{aligned}
&\vdash_{tot} \{\neg y = 0\}\\
&r \leftarrow x; q \leftarrow 0;\\
&\texttt{while}\, y \leq r\, \texttt{do}\\
&\quad r \leftarrow r - y;\\
&\quad q \leftarrow q + 1\\
&\{r < y \;\wedge\; x = r + (y \times q)\}
\end{aligned}
$$

$\diamond$