Verifica di programmi

Informalmente, un programma è **corretto** se l'output prodotto è quello atteso rispetto all'input.

La correttezza dei programmi può essere espressa mediante **formule per la correttezza** della forma

$$\{p\}\ S\ \{q\}$$

dove S è un programma e p, q sono **asserzioni**. p è detta **precondizione** e q **postcondizione**.

La precondizione descrive gli insiemi di stati iniziali di S; la postcondizione descrive l'insieme di stati finali di S.

Studieremo due interpretazioni delle formule per la correttezza:

- **Correttezza parziale**: una formula $\{p\}\ S\ \{q\}$ è vera se ogni computazione convergente di S a partire da un stato che soddisfa p termina in uno stato che soddisfa q.

- **Correttezza totale**: una formula $\{p\}\ S\ \{q\}$ è vera se ogni computazione di S a partire da un stato che soddisfa p termina in uno stato che soddisfa q.

Asserzioni

Consideriamo il seguente insieme di **asserzioni** per descrivere **proprietà degli stati**:

- un'espressione booleana è un'asserzione;
- se p, q sono asserzioni, allora $\neg p$, $p \vee q$, $p \wedge q$, $p \rightarrow q$, $p \leftrightarrow q$ sono asserzioni;
- se x è una variabile semplice e p un'asserzione, allora $\exists x : p$ e $\forall x : p$ sono asserzioni.

Un'occorrenza **legata** di una variabile semplice x in p è un'occorrenza di x che compare nello scope di un quantificatore in p.

Un'occorrenza di una variabile semplice x in p è **libera** se non è legata. Tutte le occorrrenze di variabili array sono libere.

$var(p)$ denota l'insieme di variabili (semplici e array) in p.
$free(p)$ denota l'insieme di variabili semplici libere in p e variabili array in p.

Semantica delle asserzioni

La semantica $\mathcal{S}[\![p]\!] : \Sigma \to \{\textbf{true,false}\}$ è definita come segue (dove $\mathcal{S}[\![p]\!](\sigma) = \textbf{true}$ è abbreviato con $\sigma \models p$):

- $\sigma \models B$ sse $\sigma(B) = \textbf{true}$
- $\sigma \models \neg p$ sse non $\sigma \models p$ ($\sigma \not\models p$)
- $\sigma \models p \wedge q$ sse $\sigma \models p$ e $\sigma \models q$
- $\sigma \models p \vee q$ sse $\sigma \models p$ oppure $\sigma \models q$
- $\sigma \models p \to q$ sse $\sigma \models p$ implica $\sigma \models q$
- $\sigma \models p \leftrightarrow q$ sse ($\sigma \models p$ se e solo se $\sigma \models q$)
- per x variabile semplice di tipo T,
  $\sigma \models \forall x : p$ sse $\sigma[x := d] \models p$ per ogni $d \in \mathcal{D}_T$
- per x variabile semplice di tipo T,
  $\sigma \models \exists x : p$ sse $\sigma[x := d] \models p$ per qualche $d \in \mathcal{D}_T$

Significato delle asserzioni

Definiamo il **significato** di un'asserzione p, $[\![p]\!]$, come:

$$[\![p]\!] = \{\sigma \mid \sigma \in \Sigma \text{ e } \sigma \models p\}$$

p è **vera** se $[\![p]\!] = \Sigma$.

p e q sono **equivalenti** se $p \leftrightarrow q$ è vera.

Per ogni asserzione p, $\bot \not\models p$, $\Delta \not\models p$, **fail** $\not\models p$.

Lemma.
(i) $[\![\neg p]\!] = \Sigma \setminus [\![p]\!]$
(ii) $[\![p \vee q]\!] = [\![p]\!] \cup [\![q]\!]$
(iii) $[\![p \wedge q]\!] = [\![p]\!] \cap [\![q]\!]$
(iv) $p \rightarrow q$ è vera sse $[\![p]\!] \subseteq [\![q]\!]$
(v) $p \leftrightarrow q$ è vera sse $[\![p]\!] = [\![q]\!]$

Interpretazione delle formule per la correttezza

- **Correttezza parziale**: la formula $\{p\}\ S\ \{q\}$ è **vera**, $\models \{p\}\ S\ \{q\}$, se

$$\mathcal{M}[\![S]\!]([\![p]\!]) \subseteq [\![q]\!] \ .$$

- **Correttezza totale**: la formula $\{p\}\ S\ \{q\}$ è **vera**, $\models_{tot} \{p\}\ S\ \{q\}$, se

$$\mathcal{M}_{tot}[\![S]\!]([\![p]\!]) \subseteq [\![q]\!] \ .$$

In particolare, $\models_{tot} \{p\}\ S\ \{q\}$ implica $\models \{p\}\ S\ \{q\}$

Dimostrare che una formula per la correttezza è vera utilizzando direttamente la semantica operazionale può essere molto complicato.

Un approccio più conveniente consiste nell'utilizzare **sistemi di regole alla Hoare** per derivare formule $\{p\}\ S\ \{q\}$.

Prima di introdurre sistemi alla Hoare per la **correttezza parziale** e **totale**, serve definire la nozione di **sostituzione**.

Sostituzione di espressioni in espressioni

La **sostituzione** di espressioni in espressioni è una funzione da espressioni in espressioni.

Date espressioni s, t e una variabile u semplice o sottoscritta, $s[u := t]$ denota l'espressione ottenuta sostituendo in s l'espressione t al posto di u.

Formalmente, $s[u := t]$ si definisce per **induzione** sulla struttura di s:

- se s è una variabile semplice

$$s[u := t] \equiv \begin{cases} t & \text{if } s \equiv u \\ s & \text{otherwise .} \end{cases}$$

- se s è una costante di tipo base, $s[u := t] \equiv s$

- se $s \equiv op(s_1, \ldots, s_n)$

$$s[u := t] \equiv op(s_1[u := t], \ldots, s_n[u := t])$$

. . . sostituzione di espressioni in espressioni

- se $s \equiv a[s_1, \ldots, s_n]$ e u è una variabile semplice o una variabile sottoscritta $b[t_1, \ldots, t_m]$, con $a \not\equiv b$,

$$s[u := t] \equiv a[s_1[u := t], \ldots, s_n[u := t]]$$

- se $s \equiv a[s_1, \ldots, s_n]$ e $u \equiv a[t_1, \ldots, t_n]$,

$$s[u := t] \equiv \textbf{if } \bigwedge_{i=1}^{n} s_i[u := t] = t_i \textbf{ then } t \textbf{ else } a[s_1[u := t], \ldots, s_n[u := t]] \textbf{ fi}$$

- se $s \equiv \textbf{if } B \textbf{ then } s_1 \textbf{ else } s_2 \textbf{ fi}$

$$s[u := t] \equiv \textbf{if } B[u := t] \textbf{ then } s_1[u := t] \textbf{ else } s_2[u := t] \textbf{ fi}$$

Sostituzione di espressioni in asserzioni

La sostituzione di una variabile u semplice o sottoscritta con un'espressione t in un'asserzione p è definita per induzione su p:

- se $p \equiv s$, s espressione booleana,

$$p[u := t] \equiv s[u := t]$$

- se $p \equiv \neg q$,

$$p[u := t] \equiv \neg(q[u := t])$$

- se $p \equiv q \vee r$,

$$p[u := t] \equiv q[u := t] \vee r[u := t]$$

- se $p \equiv \exists x : q$,

$$p[u := t] \equiv \exists y : q[x := y][u := t]$$

  dove y non compare in p, t, u e ha lo stesso tipo di x

- se $p \equiv \forall x : q$,

$$p[u := t] \equiv \forall y : q[x := y][u := t]$$

  dove y non compare in p, t, u e ha lo stesso tipo di x.

Sostituzione identica

Lemma.
Siano s, t espressioni, x variabile semplice, $a[t_1, \ldots, t_n]$ variabile sottoscritta. Allora:

(i) $s[x := t] \equiv s$, se s non contiene x.

(ii) $s[a[t_1, \ldots, t_n] := t] \equiv s$, se s non contiene a.

Lemma di Coincidenza

Lemma.
Per ogni espressione s, asserzione p, stati propri σ, τ,

(i) se $\sigma[var(s)] = \tau[var(s)]$ allora $\sigma(s) = \tau(s)$;

(ii) se $\sigma[free(p)] = \tau[free(p)]$ allora $\sigma \models p$ sse $\tau \models p$.

Lemma di sostituzione

Lemma.
Per tutte le espressioni s, t, per tutte le asserzioni p, e tutte le variabili u semplici o sottoscritte dello stesso tipo di t e per tutti gli stati propri σ,

(i) $\sigma(s[u := t]) = \sigma[u := \sigma(t)](s)$

(ii) $\sigma \models p[u := t]$ sse $\sigma[u := \sigma(t)] \models p$.

Regole per la correttezza parziale di programmi deterministici

Sistema formale PD:

$$\frac{}{\{p\}\ skip\ \{q\}}\ \text{skip} \qquad \frac{}{\{p[u := t]\}\ u := t\ \{p\}}\ :=$$

$$\frac{\{p \wedge B\}\ S_1\ \{q\}\quad \{p \wedge \neg B\}\ S_2\ \{q\}}{\{p\}\ \textbf{if}\ B\ \textbf{then}\ S_1\ \textbf{else}\ S_2\textbf{fi}\ \{q\}}\ \textbf{if}$$

$$\frac{\{p \wedge B\}\ S\ \{p\}}{\{p\}\ \textbf{while}\ B\ \textbf{do}\ S\ \textbf{od}\ \{p \wedge \neg B\}}\ \textbf{while}$$

$$\frac{\{p\}\ S_1\ \{r\}\quad \{r\}\ S_2\ \{q\}}{\{p\}\ S_1; S_2\ \{q\}}\ ; \qquad \frac{p \rightarrow p_1\quad \{p_1\}\ S\ \{q_1\}\quad q_1 \rightarrow q}{\{p\}\ S\ \{q\}}\ \rightarrow$$

L'asserzione p nella regola **while** è chiamata **invariante di ciclo**.

La regola $\rightarrow$ è chiamata regola di **conseguenza logica**.

Composizionalità di PD

Il sistema formale *PD* è **composizionale**, in quanto il comportamento di un programma complesso è spiegato in termini del comportamento dei suoi sottoprogrammi.

Esempio 1

Sia $S \equiv x := x + 1; y := y + 1$. Dimostriamo la correttezza in *PD* della formula

$$\{x = y\}\ S\ \{x = y\}\ , \tag{1}$$

cioè dimostriamo che la formula (1) è derivabile in *PD*.

Applichiamo l'assioma := due volte:

- per sostituzione all'indietro, $(x = y)[y := y + 1] \equiv x = y + 1$,
  per l'assioma := $\{x = y + 1\}\ y := y + 1\ \{x = y\}$

- per sostituzione all'indietro, $(x = y + 1)[x := x + 1] \equiv x + 1 = y + 1$,
  per l'assioma := $\{x + 1 = y + 1\}\ x := x + 1\ \{x = y + 1\}$

- per la regola ; $\{x + 1 = y + 1\}\ x := x + 1; y := y + 1\ \{x = y\}$

- dalla regola di conseguenza $\rightarrow$, poichè $(x = y)\ \rightarrow\ (x+1 = y+1)$, deriviamo la (1).

Esempio 2

Dimostrare che la seguente formula è derivabile in *PD*:

$$\{\textbf{true}\}\ (x := 1; a[1] := 2; a[x] := x)\ \{a[1] = 1\}$$

Esempio 3: iterazione

Sia *DIV* il seguente programma per calcolare quoziente e resto della divisione di due naturali x, y:

$$DIV \equiv quo := 0; rem := x; S_0$$

dove

$$S_0 \equiv \textbf{while } rem \geq y \textbf{ do } rem := rem - y; quo := quo + 1 \textbf{ od}$$

L'obiettivo è dimostrare che: "se x, y sono interi positivi e *DIV* termina, allora *quo* è il quoziente intero e *rem* è il resto della divisione di x per y".

Dimostriamo che è derivabile in *PD*:

$$\{x \geq 0 \ \wedge \ y \geq 0\} \ DIV \ \{quo \cdot y + rem = x \ \wedge \ 0 \leq rem < y\} \qquad (2)$$

Ma abbiamo dimostrato la **correttezza** (parziale) del programma?

Ossia: dal fatto che la formula (2) è derivabile in *PD* possiamo dedurre

$$\models \{x \geq 0 \ \wedge \ y \geq 0\} \ DIV \ \{quo \cdot y + rem = x \ \wedge \ 0 \leq rem < y\} \ \ ?$$

Cioè

$$\mathcal{M}[\![DIV]\!]([\![x \geq 0 \ \wedge \ y \geq 0]\!]) \subseteq [\![quo \cdot y + rem = x \ \wedge \ 0 \leq rem < y]\!] \ \ ?$$

Un sistema formale G si dice **corretto** rispetto alla semantica operazionale, quando la derivabilità G di una formula implica la sua verità.

Correttezza di un sistema formale

Sia G un sistema formale per derivare formule $\{p\}\ S\ \{q\}$ per una classe C di programmi.

- G è **corretto** rispetto alla semantica operazionale per la **correttezza parziale** se, per ogni formula $\{p\}\ S\ \{q\}$, con S in C,

$$\vdash_G \{p\}\ S\ \{q\} \quad \text{implica} \quad \models \{p\}\ S\ \{q\}\ .$$

- G è **corretto** rispetto alla semantica operazionale per la **correttezza totale** se, per ogni formula $\{p\}\ S\ \{q\}$, con S in C,

$$\vdash_G \{p\}\ S\ \{q\} \quad \text{implica} \quad \models_{tot} \{p\}\ S\ \{q\}\ .$$

Correttezza di PD

Teorema.
Il sistema formale *PD* è **corretto** rispetto alla semantica operazionale per la **correttezza parziale** di **programmi deterministici**.

dim. Per **induzione** sull'altezza della derivazione in *PD*.
Caso base: gli assiomi sono **veri**.
Passo induttivo: le regole sono **corrette**, cioè se le premesse sono vere, allora anche la conclusione è vera.