Programmi paralleli disgiunti

Definizione. S_1, S_2 programmi deterministici sono **disgiunti** se

$$change(S_1) \cap var(S_2) = \emptyset \ \wedge \ change(S_2) \cap var(S_1) = \emptyset \ .$$

I **programmi paralleli disgiunti** sono i programmi della forma

$$[S_1 | \ldots | S_n]$$

dove $n > 1$ e $S_1, \ldots, S_n$ sono programmi deterministici a due a due disgiunti.

Regola di transizione per programmi paralleli

$$\frac{\langle S_i, \sigma\rangle \to \langle T_i, \tau\rangle}{\langle [S_1 | \ldots | S_i | \ldots | S_n], \sigma\rangle \to \langle [S_1 | \ldots | T_i | \ldots | S_n], \tau\rangle}$$

dove $i \in \{1, \ldots, n\}$.

Sequenzializzazione

Definizione. Due computazioni sono **input/output (i/o) equivalenti** se partono dallo stesso stato inziale e sono entrambe infinite o entrambe terminano nello stesso stato finale.

Lemma.[**Sequenzializzazione**] Siano $S_1, \ldots, S_n$ programmi deterministici disgiunti a due a due. Allora
i) $\mathcal{M}[\![S_1|\ldots|S_n]\!] = \mathcal{M}[\![S_1;\ldots;S_n]\!]$
ii) $\mathcal{M}_{tot}[\![S_1|\ldots|S_n]\!] = \mathcal{M}_{tot}[\![S_1;\ldots;S_n]\!]$.

Verifica di programmi paralleli disgiunti

Regola di Sequenzializzazione

$$\frac{\{p\}\ S_1; \ldots ; S_n\ \{q\}}{\{p\}\ [S_1 | \ldots | S_n]\ \{q\}}$$

Problema: La Regola di Sequenzializzazione non riflette l'idea del parallelismo disgiunto, dove le componenti sequenziali sono programmi indipendenti.

Regola per il Parallelismo Disgiunto, [Hoare72]

$$\frac{\{p_i\}\ S_i\ \{q_i\}\ , i \in \{1, \ldots, n\}}{\{\bigwedge_{i=1}^{n} p_i\}\ [S_1 | \ldots | S_n]\ \{\bigwedge_{i=1}^{n} q_i\}}$$

dove $free(p_i, q_i) \cap change(S_j) = \emptyset$, per $i \neq j$.

Problema: La Regola per il Parallelismo Disgiunto è più debole della Regola di Sequenzializzazione, e.g. la formula

$$\{x = y\}\ [x := x + 1 | y := y + 1]\ \{x = y\}$$

non è derivabile con la Regola per il Parallelismo Disgiunto.

Variabili Ausiliarie

Definizione. Sia A un insieme di variabili in S. A è un insieme di **variabili ausiliarie** se ogni variabile in A occorre in S solo in assegnamenti del tipo $z := t$, con $z \in A$.

Regola delle Variabili Ausiliarie:

$$\frac{\{p\}\ S\ \{q\}}{\{p\}\ S_0\ \{q\}}$$

dove S_0 è ottenuto da S cancellando tutti gli assegnamenti a variabili in A, per A insieme di variabili ausiliarie di S tali che $\mathit{free}(q) \cap A = \emptyset$.

Il sistema PP consiste delle regole in PD più le regole per il parallelismo disgiunto e per le variabili ausiliarie.

Il sistema TP consiste delle regole in TD più le regole per il parallelismo disgiunto e per le variabili ausiliarie.

Correttezza

Lemma. Le regole per il parallelismo disgiunto e le variabili ausiliarie sono corrette sia rispetto alla correttezza paraziale che rispetto alla correttezza totale.

Corollario. I sistemi PP e TP sono corretti rispetto alla semantica operazionale per la correttezza parziale e per la correttezza totale, rispettivamente.

Problema. Sia a un array di interi, $N \geq 1$.
Scrivere un programma *FIND* che trova il più piccolo $k \in \{1, \ldots, N\}$ tale che $a[k] > 0$, se esiste un elemento positivo, altrimenti restituisce $N + 1$.
Il programma deve lasciare invariato l'array a, cioè $a \notin change(FIND)$.

Quindi il programma deve soddisfare la seguente formula per la correttezza totale

$$\{true\}\ FIND\ \{1 \leq k \leq N+1\ \wedge\ \forall(1 \leq l \leq k).\ a[l] \leq 0\ \wedge\ (k \leq N\ \longrightarrow\ a[k] > 0)\}\ .$$

Il programmma FIND

FIND è composto da 2 componenti eseguite in parallelo: S_1 cerca un indice k dispari, S_2 un indice k pari.

$S_1 \equiv$ **while** $i < oddtop$ **do**
if $a[i] > 0$ **then** $oddtop := i$
else $i := i + 2$ **fi**
od

$S_2 \equiv$ **while** $j < eventop$ **do**
if $a[j] > 0$ **then** $eventop := j$
else $j := j + 2$ **fi**
od

$FIND \equiv i := 1;\ j := 2;\ oddtop := N + 1 :\ eventop := N + 1;$
$[S_1 | S_2];$
$k := min(eventop, oddtop).$

Verifica di correttezza di FIND

Dimostriamo che:

- S_1 soddisfa alla formula

$$\{i = 1 \ \wedge \ oddtop = N + 1\} \ S_1 \ \{q_1\} \tag{1}$$

dove

$$q_1 \equiv 1 \leq oddtop \leq N + 1 \ \wedge \ \forall l(odd(l) \ \wedge \\ 1 \leq l < oddtop \ \longrightarrow \ a[l] \leq 0) \ \wedge \\ (oddtop \leq N \ \longrightarrow \ a[oddtop > 0).$$

- S_2 soddisfa alla formula

$$\{j = 2 \ \wedge \ eventop = N + 1\} \ S_2 \ \{q_2\} \tag{2}$$

dove

$$q_2 \equiv 2 \leq eventop \leq N + 1 \ \wedge \ \forall l(even(l) \ \wedge \\ 1 \leq l < eventop \ \longrightarrow \ a[l] \leq 0) \ \wedge \\ (eventop \leq N \ \longrightarrow \ a[eventop > 0).$$