

Logica per la Programmazione

Lezione 15

- ▶ Sistema di Dimostrazioni per le **Triple di Hoare**
- ▶ **Comando Vuoto, Assegnamento, Sequenza, Condizionale**

Tripla di Hoare Soddisfatta: richiamo

Data la **tripla di Hoare** $\{Q\} C \{R\}$

- ▶ Q è detta **precondizione**
- ▶ R è detta **postcondizione**
- ▶ La **tripla è soddisfatta** se:
 - ▶ *per ogni* stato σ che soddisfa la **precondizione** Q (ovvero $\sigma \models Q$)
 - ▶ l'esecuzione del comando C a partire dallo stato σ **termina**,
 - ▶ e lo stato in cui termina σ' soddisfa la **postcondizione** R (ovvero $\sigma' \models R$)

Verificare una Tripla di Hoare

- ▶ Obiettivo principale: **verificare che una tripla sia soddisfatta** (come *dimostrare che una proposizione è una tautologia* nel Calcolo Proposizionale, o *dimostrare che una formula è valida* nella Logica del Primo Ordine)
- ▶ Diremo anche, più semplicemente: **verificare una tripla**
- ▶ Per questo introduciamo un **proof system** per verificare le triple
 - ▶ **Assiomi** - alcune triple che sono sempre soddisfatte
 - ▶ **Regole di inferenza** - per induzione strutturale sui comandi
- ▶ **Correttezza degli assiomi e delle regole** basata su semantica informale dei comandi

Regola di Inferenza per la preconditione

$$(PRE) \frac{P \Rightarrow P' \quad \{P'\} C \{R\}}{\{P\} C \{R\}}$$

- ▶ Ricordiamo che $\{P\}$ è l'insieme di stati che soddisfano P .
- ▶ La **correttezza** segue dalla definizione di “tripla soddisfatta” e dal fatto che $P \Rightarrow P'$ se e solo se $\{P\} \subseteq \{P'\}$

Regola di Inferenza “Pre-Post”

$$(POST) \frac{\{P\} C \{R'\} \quad R' \Rightarrow R}{\{P\} C \{R\}}$$

$$(PRE - POST) \frac{P \Rightarrow P' \quad \{P'\} C \{R'\} \quad R' \Rightarrow R}{\{P\} C \{R\}}$$

- La **correttezza** segue dalla definizione di “triplo soddisfatta” in modo analogo al caso della regola (*PRE*)

Assioma per Skip

- ▶ **Assioma** per il comando vuoto:

$$\boxed{(\text{SKIP}) \quad \{P\} \text{ skip } \{P\}}$$

- ▶ Correttezza? Ovvio...
- ▶ Infatti ricordiamo il significato informale:
 - ▶ L'esecuzione di **skip** a partire dallo stato σ porta nello stato σ

Sulla Definizione delle Espressioni

- ▶ Valutando un comando che contiene un'espressione (come $x := E$ o **if E' then C_1 else C_2 fi**), l'espressione potrebbe non essere definita:
 - (1) potrebbe **non essere tipizzata correttamente**, come in $x := 5 + true$, o in **if $7 * 4$ then C_1 else C_2 fi**
 - (2) l'espressione potrebbe essere tipizzata correttamente, ma il suo valore non è definito, come in $x := 0; y := k \text{ div } x$
- ▶ Noi assumiamo che tutte le espressioni che compaiono nei nostri programmi siano sempre **ben tipizzate**, quindi il caso (1) non si può mai verificare.
 - ▶ Si assume che **il controllo dei tipi** sia realizzato in una fase di **analisi statica** preliminare.
- ▶ Per il caso (2), introduciamo una funzione *def* che applicata a un'espressione E restituisce un'asserzione $def(E)$
- ▶ L'asserzione $def(E)$ vale **true** in uno stato se la valutazione di E è **definita**

Definizione di Espressioni

- La funzione $def(_)$, applicata a un'espressione E , restituisce una **asserzione** tale che $\sigma \models def(E)$ se esiste un v tale che $\mathcal{E}(E, \sigma) = v$.

$$def(c) = tt \quad \text{se } c \in Num \text{ o } c \in Bool$$

$$def(x) = tt \quad \text{se } x \in Ide$$

$$def(E \text{ op } E') = def(E) \wedge def(E') \quad \text{se } op \in \left\{ +, -, =, \neq, <, >, \leq, \geq, and, or \right\}$$

$$def(E \text{ op } E') = def(E) \wedge def(E') \wedge E' \neq 0 \quad \text{se } op \in \{div, mod\}$$

$$def(not E) = def(E)$$

$$def((E)) = def(E)$$

Assioma per Assegnamento Semplice

- ▶ **Assioma** per l'assegnamento semplice

$$(ASS) \quad \{ \text{def}(E) \wedge P[E/x] \} x := E \{ P \}$$

dove $P[E/x]$ denota l'asserzione P in cui E viene sostituito ad x

- ▶ **Intuizione:** affinché la postcondizione P sia vera dopo l'assegnamento, la stessa P deve essere vera nello stato di partenza quando alla variabile x è sostituita l'espressione E
- ▶ Inoltre l'espressione E deve essere definita nello stato di partenza (cioè $\text{def}(E)$ deve essere vera)
- ▶ Esempio:

$$\begin{aligned} & \{ \text{????} \} x := x + 2 \{ x > y \} \\ & \{ \text{def}(x + 2) \wedge (x > y)[x+2/x] \} x := x + 2 \{ x > y \} \\ & \{ (x + 2 > y) \} x := x + 2 \{ x > y \} \end{aligned}$$

Correttezza dell'Assioma per Assegnamento Semplice

$$(ASS) \quad \{ \text{def}(E) \wedge P[E/x] \} x := E \{ P \}$$

- ▶ La **correttezza** dell'assioma si vede confrontando l'assioma con la semantica informale:
 - ▶ L'esecuzione dell'assegnamento $x := E$ a partire dallo stato σ porta nello stato $\sigma[\mathcal{E}(E, \sigma)/x]$
- ▶ e ricordando che
 - ▶ per ogni variabile x

$$\sigma[\mathcal{E}(E, \sigma)/x] \models P \quad \text{se e solo se} \quad \sigma \models P[E/x]$$

- ▶ Esempio: $\sigma[\mathcal{E}(5, \sigma)/x] \models (x > y)$ se e solo se $\sigma \models (x > y)[5/x]$
 cioè: $\sigma[5/x] \models (x > y)$ se e solo se $\sigma \models (5 > y)$

Esempi di Assegnamento Semplice

Verificare le triple:

$$(ASS) \quad \{def(E) \wedge P[E/x]\} x := E \{P\}$$

▶ $\{\mathbf{T}\} x := 5 \{x = 5\}$

- ▶ Dall'assioma (ASS) otteniamo la **precondizione**

$$def(5) \wedge (x = 5)[5/x] \equiv (5 = 5) \equiv \mathbf{T}$$

▶ $\{x > 2\} x := 5 \{x = 5\}$

- ▶ La stessa postcondizione di prima. Vale??? Come la possiamo dimostrare????
- ▶ Dalla regola (PRE) notando che $x > 2 \Rightarrow \mathbf{T}$

▶ $\{x = 5\} x := x + 1 \{x > 2\}$

- ▶ Analogamente da (ASS) e (PRE) osservando che

$$(x > 2)[x+1/x] = x + 1 > 2 \equiv x > 1 \quad \text{e} \quad x = 5 \Rightarrow x > 1$$

Regola di Inferenza per l'Assegnamento

- Combinando la regola (PRE) con l'assioma per l'assegnamento semplice, si ottiene la seguente **regola**, utile per verificare che una tripla data sia soddisfatta:

$$(ASS) \quad \frac{R \Rightarrow \text{def}(E) \wedge P[E/x]}{\{R\} x := E \{P\}}$$

- Infatti abbiamo la seguente istanza di (PRE), in cui la seconda premessa è vera (e quindi scompare) perché è un **assioma**:

$$\frac{R \Rightarrow \text{def}(E) \wedge P[E/x] \quad \{\text{def}(E) \wedge P[E/x]\} x := E \{P\}}{\{R\} x := E \{P\}}$$

Assioma per Assegnamento Multiplo

- ▶ Generalizza quello per l'assegnamento singolo (ASS):

$$\{ \text{def}(E_1) \wedge \dots \wedge \text{def}(E_k) \wedge P[E_1/x_1, \dots, E_k/x_k] \} \quad x_1, \dots, x_k := E_1, \dots, E_k \quad \{ P \}$$

- ▶ Tutte le espressioni vengono valutate **prima** di tutti gli assegnamenti: confrontiamo con la semantica informale
 - ▶ L'esecuzione dell'assegnamento $x_1, \dots, x_k := E_1, \dots, E_k$ a partire dallo stato σ porta nello stato $\sigma[\mathcal{E}(E_1, \sigma)/x_1, \dots, \mathcal{E}(E_k, \sigma)/x_k]$
- ▶ Nota che gli assiomi per l'assegnamento consentono di **completare** una tripla di cui si conosce solo il comando (l'assegnamento) e la postcondizione, “propagando” all'indietro l'asserzione.

Regola per la Sequenza di Comandi

- ▶ Regola per verifica di una tripla in cui il comando è una **sequenza** (per induzione strutturale):

$$(SEQ) \frac{\{P\} C \{R\} \quad \{R\} C' \{Q\}}{\{P\} C; C' \{Q\}}$$

- ▶ Convinciamoci della correttezza confrontandola con la semantica informale:
 - ▶ L'esecuzione di $C; C'$ a partire dallo stato σ porta nello stato σ' ottenuto eseguendo C' a partire dallo stato σ'' ottenuto dall'esecuzione di C nello stato σ .
- ▶ Come esempio verifichiamo la seguente tripla:

$$\{x \geq y - 1\} x := x + 1; \quad y := y - 1 \{x > y\}$$

Esempio di Sequenza di Comandi

Verificare la tripla:

$$\{x \geq y - 1\} x := x + 1; y := y - 1 \{x > y\}$$

- ▶ Per la **regola** (SEQ), dobbiamo trovare una **asserzione intermedia** R e verificare le seguenti triple:

$$(1) \{x \geq y - 1\} x := x + 1 \{R\}$$

$$(2) \{R\} y := y - 1 \{x > y\}$$

- ▶ Per determinare R , usiamo l'**assioma** (ASS) nella (2). Quindi la seguente è verificata:

$$\{def(y - 1) \wedge (x > y)^{[y-1/y]}\} y := y - 1 \{x > y\}$$

- ▶ Fissando $R = def(y - 1) \wedge x > y - 1$ resta da verificare la (1):

$$\{x \geq y - 1\} x := x + 1 \{R\}$$

- ▶ Usando la **regola** (ASS) basta dimostrare (per esercizio):

$$x \geq y - 1 \Rightarrow def(x + 1) \wedge (def(y - 1) \wedge (x > y - 1))[x + 1/x]$$

Regola per il Comando Condizionale

- ▶ Regola per verifica di una tripla in cui il comando è un **condizionale** (per induzione strutturale):

$$(COND) \frac{P \Rightarrow \text{def}(E) \quad \{P \wedge E\} C_1 \{Q\} \quad \{P \wedge \neg E\} C_2 \{Q\}}{\{P\} \text{ if } E \text{ then } C_1 \text{ else } C_2 \text{ fi } \{Q\}}$$

- ▶ La correttezza della regola segue dal confronto con il significato informale del condizionale:
 - ▶ L'esecuzione di **if** E **then** C_1 **else** C_2 **fi** a partire da uno stato σ porta nello stato σ' :
 - ▶ che si ottiene dall'esecuzione di C_1 in σ , se $\mathcal{E}(E, \sigma) = \mathbf{tt}$
 - ▶ che si ottiene dall'esecuzione di C_2 in σ , se $\mathcal{E}(E, \sigma) = \mathbf{ff}$

Esempio: comando condizionale

Verificare la seguente tripla:

```
{  $m = 0$  }  
  if  $x < y$   
    then  $m := y$   
    else  $m := x$   
  fi  
{  $m = \max(x, y)$  }
```

Soluzione: Verifica della Tripla

Per la **regola** (*COND*) dobbiamo mostrare che:

1. $m = 0 \Rightarrow \text{def}(x < y)$
2. $\{m = 0 \wedge x < y\} m := y \{m = \max(x, y)\}$
3. $\{m = 0 \wedge x \geq y\} m := x \{m = \max(x, y)\}$

1. Banale applicando la definizione di *def*:

$$\text{def}(x < y) = \text{def}(x) \wedge \text{def}(y) \equiv \mathbf{T}$$

2. Usando la **regola** (*ASS*) ci riduciamo a mostrare:

$$m = 0 \wedge x < y \Rightarrow \text{def}(y) \wedge (m = \max(x, y))[y/m]$$

ovvero $m = 0 \wedge x < y \Rightarrow y = \max(x, y)$

3. Analogamente, ci riduciamo a mostrare:

$$m = 0 \wedge x \geq y \Rightarrow x = \max(x, y)$$

Esercizio: completare la dimostrazione

Sequenza con **Variabili di Specifica** (1)

- ▶ Determinare E in modo che la seguente tripla sia verificata:

$$\{x = N \wedge y = M\} t := E; \quad x := y; \quad y := t \quad \{x = M \wedge y = N\}$$

- ▶ Attenzione: M e N sono chiamate **variabili di specifica**: non possono essere usate nei comandi
- ▶ Candidati per E ???

Sequenza con Variabili di Specifica (2)

- ▶ Per la **regola** (SEQ) dobbiamo trovare R_1 e R_2 tali che:

$$(1) \{x = N \wedge y = M\} t := E \{R_1\}$$

$$(2) \{R_1\} x := y \{R_2\}$$

$$(3) \{R_2\} y := t \{x = M \wedge y = N\}$$

- ▶ Per determinare R_2 , usiamo l'**assioma** (ASS) in (3):

$$\{def(t) \wedge (x = M \wedge y = N)[t/y]\} y := t \{x = M \wedge y = N\}$$

- ▶ Quindi fissiamo $R_2 = (x = M \wedge t = N)$

- ▶ Analogamente per R_1 , usiamo l'**assioma** (ASS) in (2):

$$\{def(y) \wedge (x = M \wedge t = N)[y/x]\} x := y \{x = M \wedge t = N\}$$

- ▶ Quindi fissiamo $R_1 = (y = M \wedge t = N)$

- ▶ Resta da verificare la (1):

$$\{x = N \wedge y = M\} t := E \{y = M \wedge t = N\}$$

- ▶ Usando la **regola** (ASS), basta trovare un E tale che:

$$x = N \wedge y = M \Rightarrow def(E) \wedge (y = M \wedge E = N)$$

Esercizi

- ▶ Verificare la seguente tripla:

$$\{s = (\sum i : i \in [0, x].i)\}$$

$$s, x := s + x; x + 1$$

$$\{s = (\sum i : i \in [0, x].i)\}$$

- ▶ La seguente tripla non è soddisfatta. Mostrare formalmente perché.

$$\{z = 5 \wedge y = 7 \wedge x = 3\} x := 0; y := z \text{ div } x \{z > 4\}$$

- ▶ Le seguenti triple sono soddisfatte? Perché?
 - ▶ $\{x = N \wedge y = M\} x := y; y := x \{x = M \wedge y = N\}$
 - ▶ $\{x = N \wedge y = M\} x, y := y, x \{x = M \wedge y = N\}$

Esercizi

Verificare la seguente tripla:

$$\{x = 5\}$$

$$x := 3;$$

$$\mathbf{if} (x = 3) \mathbf{then} y := 7 \mathbf{else} y := 5 \mathbf{fi}$$

$$\{x = 3 \wedge y = 7\}$$

- ▶ Per la **regola** (*SEQ*) dobbiamo trovare una asserzione intermedia R che soddisfi le seguenti triple:
 - (1) $\{x = 5\} x := 3 \{R\}$
 - (2) $\{R\} \mathbf{if} (x = 3) \mathbf{then} y := 7 \mathbf{else} y := 5 \mathbf{fi} \{x = 3 \wedge y = 7\}$
- ▶ A differenza di altri esempi, qui non possiamo usare l'assioma dell'assegnamento per trovare R .
- ▶ Un candidato naturale per R è $x = 3$.
- ▶ **Esercizio:** completare la dimostrazione di (1) e (2)