

# Lezione 16 30/11/2023

## Correttezza totale

Parliamo della terminazione

$$\{p\} \text{ while } B \text{ do } C \text{ endwhile } \{q\}$$

$W$

c. parziale "se si esegue  $W$  a partire da uno stato in cui vale  $p$  e l'esecuzione termina, nello stato finale vale  $q$ "

c. totale "se si esegue  $W$  a partire da uno stato in cui vale  $p$ , l'esecuzione termina e nello stato finale vale  $q$ "

$$\vdash^{\text{parz}} \{p\} C_1 \{q\}$$

$$\vdash^{\text{TOT}} \{p\} C_1 \{q\}$$

$$\vdash \{p\} C_1 \{q\}$$

Ci concentriamo su una tripla dove c'è dentro una condizione di ciclo  $B$  e un corpo.

Usiamo due modi diversi per mostrare l'interpretazione della tripla tramite la definizione di correttezza parziale o tramite quella di correttezza totale.

$C_1$  è un programma

Come possiamo dimostrare che l'esecuzione terminerà? Ci aspettiamo che, a mano a mano che si procede con le iterazioni, qualcosa cambi. Se tutto rimane invariato allora l'esecuzione non finirà mai. Abbiamo bisogno, oltre all'invariante, di un oggetto che chiameremo **variante**.

### Tecnica di dimostrazione

Supponiamo che  $E$  sia un'espressione aritmetica nella quale compaiono variabili del programma, costanti numeriche e operazioni aritmetiche, e che  $\text{inv}$  sia un invariante di ciclo per  $W$ , scelti in modo che:

1.  $\text{inv} \rightarrow E \geq 0$
2.  $\vdash_{\text{TOT}}^{\text{inv}} \{ \text{inv} \wedge B \wedge E = k > 0 \} C \{ \text{inv} \wedge E < k \}$

Allora:  $\vdash_{\text{TOT}}^{\text{inv}} \{ \text{inv} \} \xrightarrow{W} \{ \text{inv} \wedge \neg B \}$

- $E$  non è una formula logica  $E \geq 0$  è una formula logica
- Lo  $\emptyset$  in  $E \geq 0$  può essere sostituito da qualsiasi numero

$E$ , interpretata in un certo stato del programma, avrà un valore definito.

Se abbiamo questo  $E$  e l'invariante  $\text{inv}$ , che soddisfino le due condizioni:

1. L'invariante (formula logica) deve implicare che il valore dell'espressione  $E$  sia  $\geq 0$ .
2. Dopo l'esecuzione di  $C$  il valore di  $E$  deve essere diminuito.

Se sono entrambe vere, allora possiamo derivare la tripla.

Le due premesse ci dicono che in un certo stato, se vale l'invariante il valore di  $E \geq 0$ , e che ogni volta che eseguiamo il corpo dell'iterazione il valore di  $E$  diminuisce. Questo significa che possiamo eseguire il corpo un numero finito di volte perché prima o poi  $E$  raggiungerà lo 0.

$E$  è la variante (perché supponiamo che il suo valore cambi), è un'espressione aritmetica.  $E \geq 0$  è una formula logica. Possiamo anche cambiare lo 0 con un altro numero.

### Esempio

Cominciamo da un esempio elementare

```
while x > 5 do
    x := x - 1
endwhile
```

$\{x > 5\} \quad P \quad \{x = 5\}$

Dobbiamo cercare un **variante**  $E$ , cioè un'espressione il cui valore decresce a ogni esecuzione del corpo dell'iterazione, e un **invariante**  $i$  che garantisca che l'espressione ha sempre valore maggiore o uguale a 0.

Il valore di  $x$  decresce ad ogni iterazione, quindi il primo candidato per  $E$  sarebbe  $x$ . Però dobbiamo trovare anche un'invariante che garantisca che il valore di  $E$  sia sempre  $\geq 0$ .  $x \geq 5$  è un'invariante e implica che  $E \geq 0$  quindi va bene per la prima premessa.

$$i \rightarrow E \geq 0 \quad \{i \wedge B \wedge E = E_0\} C \{i \wedge E < E_0\}$$

$E_0$  è un valore simbolico che assume  $E$ . Dobbiamo dimostrare questa tripla, applichiamo la regola dell'assegnamento (per il contenuto del ciclo). (ha preso  $x - 5$  come espressione  $E$ , funziona anche questo)

**Soluzione:**  $E : x - 5 \quad i : x \geq 5$

Poniamo

$$i : x \geq 5 \quad B : x > 5 \quad E : x - 5$$

Osserviamo che  $i \rightarrow E \geq 0$

Applicando la regola dell'assegnamento otteniamo l'ultima formula in fondo.

Dobbiamo dimostrare la seguente tripla

$$\{i \wedge B \wedge x - 5 = E_0\} \quad x := x - 1 \quad \{i \wedge x - 5 < E_0\}$$

Applichiamo la regola dell'assegnamento e otteniamo la precondizione

$$x - 1 \geq 5 \wedge x - 1 - 5 < E_0$$

che diventa

$$x \geq 6 \wedge x - 6 < E_0$$

Poiché  $(i \wedge B \wedge x - 5 < E_0) \rightarrow (x \geq 6 \wedge x - 6 < E_0)$ , possiamo applicare la regola di derivazione della conseguenza e derivare la tripla richiesta.

Cose extra su cui ragionare.

Come cambia la dimostrazione se  $B$  diventa  $x \neq 5$ ?

Quali sono i passi della dimostrazione di correttezza parziale?

Se il programma fosse `while x != 5 do x := x-2 endwhile` ?

Che cosa fare se nessuna variabile viene decrementata?

```
while x < 5 do
    x := x + 1
endwhile
```

$\{x < 5\} \quad P \quad \{x = 5\}$

$i : x \leq 5 \quad E : 5 - x$

Cambiamo il programma.

In questo caso nel corpo dell'iterazione non abbiamo una quantità il cui valore diminuisce durante l'iterazione.

Useremo quindi questa invariante e variante.

## Schema di dimostrazione di correttezza

Schema generale di dimostrazione

$\{p\} \quad V; W; Z \quad \{q\}$

↗ istruzione iterativa

supponiamo che  $V$  e  $Z$  non contengano while

Da  $Z \{q\}$  ricaviamo  $wp(Z, q) \equiv s \quad \{s\} Z \{q\}$

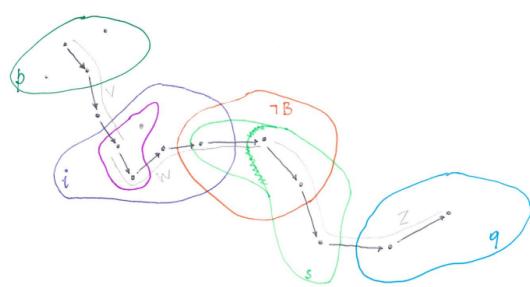
Cerchiamo un invariante  $i$  per  $W$  tale che  $(i \wedge \neg B) \rightarrow s \quad \{i\} W; Z \{q\}$

Cerchiamo una formula  $u$  tale che  $\{p\} \nV \{u\}$  e  $u \rightarrow i \quad \{p\} V; W; Z \{q\}$

La strategia tipica è questa: concentriamoci sulla fine del programma, il blocco  $Z$  e la postcondizione. Se  $Z$  non contiene funzioni iterative è abbastanza semplice ricavare una pre condizione  $s$  per cui questa tripla è valida.

Dobbiamo cercare l'invariante di modo che l'invariante insieme alla condizione di ciclo negata implichi  $s$  (caso migliore sarebbe che la condizione di ciclo negata sia già  $s$  ma difficilmente lo è).

In questo modo avremo precondizione  $i$ , corpo  $W; Z$  e post condizione  $q$ . Cercheremo una formula  $u$  che implichi  $i$ .



L'insieme verde è l'insieme di stati dove è vera p.

L'insieme viola è quello dove è vero l'invariante i. Noi sappiamo

## Estensione del linguaggio

- ▶ do C while B endwhile  $\equiv$  C; while B do C endwhile
- ▶ repeat C until B endrepeat  $\equiv$  C; while not B do C endwhile
- ▶ for ( D; B; F ) C endfor  $\equiv$  Esercizio
  
- ▶ Procedure (metodi, funzioni)
- ▶ Array (solo lettura)

l'endwhile della seconda dovrebbe essere a destra dopo "do C"

---

## Logica di Hoare

# Proprietà generali

CORRETTEZZA  $\vdash \Rightarrow \models$

COMPLETEZZA  $\models \Rightarrow \vdash$   
(relativa)



incompletezza dell'aritmetica

**Problema** Dati un comando  $C$  e una formula  $q$ , trovare una formula  $p$  tale che

$$\vdash \{p\} C \{q\}$$

**Esempio**  $x := k; y := 2*x \quad \{y > 0\}$

**Soluzioni**  $k = 5 \quad k = 12 \quad k > 3 \quad k > 0$   
 ~~$k \neq 0$~~

Esiste una precondizione "migliore"?

Come definirla? Come calcolarla?

Il programma deve produrre il risultato  $q$ .

Siamo in grado di determinare un'altra formula  $p$  tale che valga la tripla.

$V$  insieme delle variabili di  $C$

$\Sigma = \{ \sigma \mid \sigma: V \rightarrow \mathbb{Z} \}$  insieme degli stati

$\Pi$  insieme delle formule su  $V$

$\sigma \models p$  "p è vera in  $\sigma$ "

$\models \subseteq \Sigma \times \Pi$

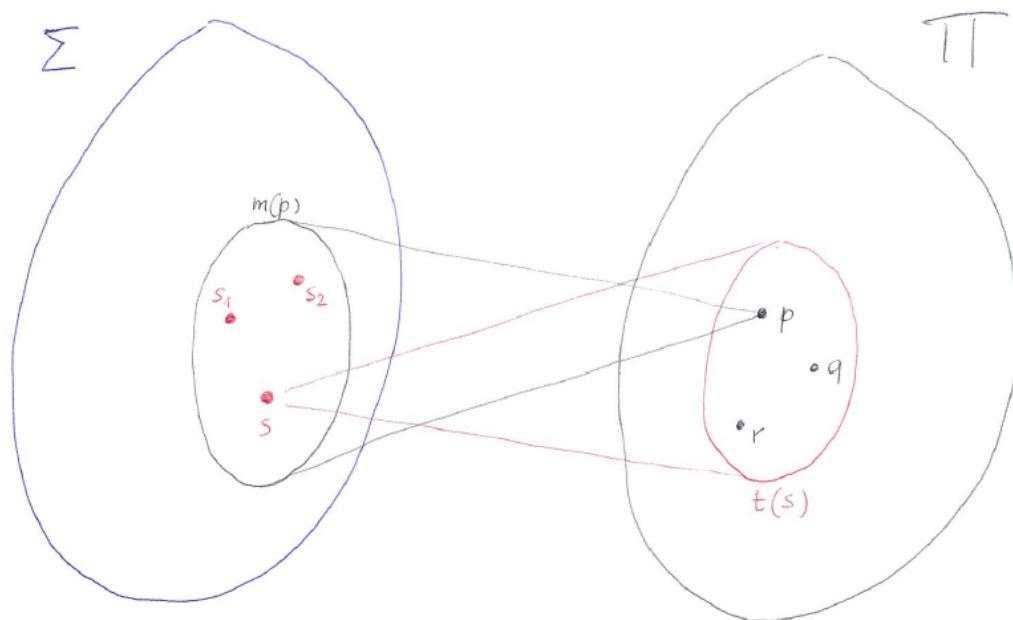
$t(\sigma) = \{ p \in \Pi \mid \sigma \models p \}$  formule vere in  $\sigma$

$m(p) = \{ \sigma \in \Sigma \mid \sigma \models p \}$  stati che soddisfano  $p$

$p$  è vera in  $\sigma$  se le espressioni di  $p$  sono vere nello stato  $\sigma$ .

$t(\sigma)$  è un insieme di formule, quelle vere in  $\sigma$ .

$m(p)$  da l'insieme di stati in cui  $p$  è vera (stati che soddisfano  $p$ ).



$$S \subseteq \Sigma \text{ sottoinsieme di stati} \quad F \subseteq T \text{ sottoinsieme di formule}$$

$$t(S) = \{ p \in T \mid \forall s \in S : s \models p \} = \bigcap_{s \in S} t(s)$$

$$m(F) = \{ s \in \Sigma \mid \forall p \in F : s \models p \} = \bigcap_{p \in F} m(p)$$

Esercizio: dimostrare che  $S \subseteq m(t(S))$      $F \subseteq t(m(F))$   
dimostrare che se  $A \subseteq B$ , allora  $m(B) \subseteq m(A)$

Prendiamo un sottoinsieme di stati e di formule.  
Definiamo  $t(S)$  come tutte le formule che sono vere in tutti gli stati dell'insieme  $S$  (cioè che gli stati hanno in comune dal punto di vista logico).

### Logica proposizionale e insieme di stati

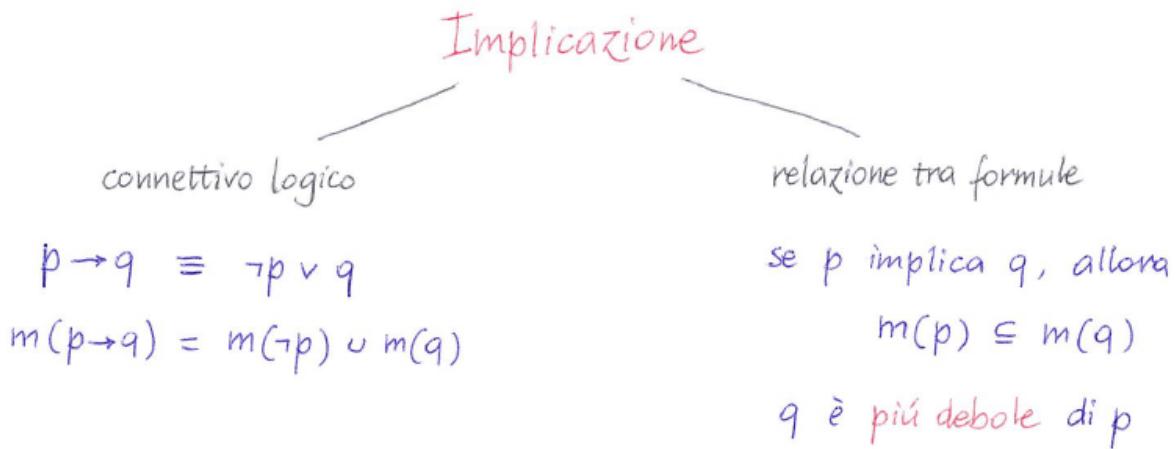
$$\neg p \quad p \vee q \quad p \wedge q \quad p \rightarrow q$$

- $m(\neg q) = \Sigma \setminus m(q)$
- $m(p \vee q) = m(p) \cup m(q)$
- $m(p \wedge q) = ?$
- $m(p \rightarrow q) = ?$

Con le proposizioni  $p$  e  $q$  possiamo costruire nuove formule.

$m(\text{non } q)$  (insieme di stati dove vale non  $q$ ) si calcola con l'insieme degli stati sigma e togliendo  $m(q)$  (qui errore sulla slide dovrebbe essere  $q$ )

Penultimo caso: intersezione



L'implicazione è anche una relazione tra formule. Possiamo osservare che se  $p$  implica  $q$ , allora ogni stato in cui è vera  $p$ , soddisfa anche  $q$ .

La formula  $p$  ci da un'informazione più precisa sugli stati.  $q$  è più debole di  $p$  (informazione meno precisa).

Vogliamo determinare una precondizione  $p$  che formi una tripla valida, ma vogliamo anche cercare tra le soluzioni quella migliore. Un criterio ragionevole è quello che

viene tradotto nella precondizione più debole.

Criterio di scelta della precondizione "migliore"

$$C \{ q \}$$

Cerchiamo la <sup>WEAKEST PRECONDITION</sup> precondizione più debole  $p$  tale che

$$\vdash \{ p \} C \{ q \}$$

$p$  corrisponde al più grande insieme di stati a partire dai quali l'esecuzione di  $C$  porta a uno stato in  $m(q)$

Notazione:  $\text{wp}(C, q)$  precondizione più debole per  $C\{q\}$

Teorema  $\vdash \{ p \} C \{ q \}$  se e solo se  $p \rightarrow \text{wp}(C, q)$

Indicheremo la precondizione più debole, indicati  $C$  e  $q$ , con  $\text{wp}(C, q)$ .

Supponiamo che la tripla sia valida, allora siamo sicuri che  $p$  implica la precondizione più debole di  $C$  e  $q$ .

Regole di calcolo di wp  
ASSEGNAZIONTO  $\text{wp}(x := E, q) = \cancel{q[E/x]} \quad (\text{vedi regola di derivazione})$

SEQUENZA  $\text{wp}(C_1; C_2, q) = \text{wp}(C_1, \text{wp}(C_2, q))$

SCELTA  $C: \text{if } B \text{ then } C_1 \text{ else } C_2 \text{ endif}$

$$\text{wp}(C, q) = (B \wedge \text{wp}(C_1, q)) \vee (\neg B \wedge \text{wp}(C_2, q))$$

Scelta: per ricavare la precondizione più debole dobbiamo separare i due casi, insieme alla condizione  $B$  della scelta. La precondizione per la scelta sarà la disunione tra i due risultati. I due risultati sono disgiunti perché da una parte  $B$  è vera e dall'altra  $B$  è falsa, non ci sono sovrapposizioni.

Calcolo tutte le precondizioni che formano una tripla valida, e scelgo quella più debole.

Dati  $C$  e  $q$  esiste sempre una precondizione più debole.

L'algoritmo di calcolo per la precondizione più debole coincide con la regola di derivazione dell'assegnamento.

Sequenza: supponiamo di voler calcolare la precondizione più debole avendo la post condizione  $q$  e  $C_1 C_2$ . Per calcolarla, procediamo partendo dal fondo. Calcoliamo la wp per il secondo comando e la post

condizione e poi  
camcoliamo la wp tra il  
primo comando e la wp  
appena calcolata.

## Esercizio

P      [ if  $\underline{y == 0}$  then  
            $x := 0$                   C  
       else  
            $x := x * y$               D  
       endif ]

$wp(P, \underline{x = y}) ?$

$$wp(C, q) : \boxed{0 = y}$$

$$wp(D, q) : xy = y \quad \boxed{y = 0 \vee x = 1}$$

$$\begin{aligned} wp(P, q) &= (B \wedge wp(C, q)) \vee (\neg B \wedge wp(D, q)) = \\ &= (y = 0 \wedge y = 0) \vee \\ &\quad (y \neq 0 \wedge (y = 0 \vee x = 1)) \equiv \end{aligned}$$

$$y = 0 \vee (y \neq 0 \wedge x = 1)$$

Il primo pezzo è l'assegnamento, abbiamo sostituito 0 a tutte le x.

Facciamo la stessa cosa con il secondo caso della scelta.

Ora dobbiamo unire questo alla B (copiando la definizione)

Poi sostituiamo B ( $y = 0$ )

## Istruzione iterativa

$W \left[ \begin{array}{l} \text{while } \overbrace{x > 0}^B \text{ do} \\ \quad x := x - 1 \quad C \\ \text{endwhile} \end{array} \right]$   
 $\wp(W, \overbrace{x=0}^q) \quad ?$   
 $\overbrace{\quad\quad\quad}^r$   
 $\wp(W, \overbrace{x=0}^q) = (\neg B \wedge q) \vee$   
 $\quad (\overbrace{B \wedge \wp(C; W, q)}^r)$   
 $\quad \quad \quad \overbrace{\quad\quad\quad}^r$   
 $\quad \quad \quad \rightarrow \wp(C, \overbrace{\wp(W, q)}^r)$

$$\begin{aligned}
 \neg B \wedge q &\equiv x \leq 0 \wedge x = 0 \equiv \boxed{x = 0} \\
 B \wedge \wp(C, r) &\equiv x > 0 \wedge r[x-1/x] \equiv \\
 &x > 0 \wedge (x-1 = 0 \vee (x-1 > 0 \wedge r[x-2/x])) \equiv \\
 &x > 0 \wedge (x = 1 \vee (x > 1 \wedge \dots))
 \end{aligned}$$

$$x = 0 \vee x = 1 \vee x = 2 \vee \dots$$

$$x > 0$$