

# Linguaggi di Programmazione

Roberta Gori

Logica di Hennessy-Milner 11.6

# CCS: sintassi

processo inattivo

variabile di processo (per la ricorsione)

prefisso azione

canale ristretto

rietichettatura del canale

scelta nondeterministica (somma)

composizione parallela

ricorsione

(gli operatori sono elencati in ordine di precedenza)

# CCS semantica

$$\text{Act) } \frac{}{\mu.p \xrightarrow{\mu} p} \quad \text{Res) } \frac{p \xrightarrow{\mu} q \quad \mu \notin \{\alpha, \bar{\alpha}\}}{p \setminus \alpha \xrightarrow{\mu} q \setminus \alpha} \quad \text{Rel) } \frac{p \xrightarrow{\mu} q}{p[\phi] \xrightarrow{\phi(\mu)} q[\phi]}$$

$$\text{SumL) } \frac{p_1 \xrightarrow{\mu} q}{p_1 + p_2 \xrightarrow{\mu} q} \quad \text{SumR) } \frac{p_2 \xrightarrow{\mu} q}{p_1 + p_2 \xrightarrow{\mu} q}$$

$$\text{ParL) } \frac{p_1 \xrightarrow{\mu} q_1}{p_1 | p_2 \xrightarrow{\mu} q_1 | p_2} \quad \text{Com) } \frac{p_1 \xrightarrow{\lambda} q_1 \quad p_2 \xrightarrow{\bar{\lambda}} q_2}{p_1 | p_2 \xrightarrow{\tau} q_1 | q_2} \quad \text{ParR) } \frac{p_2 \xrightarrow{\mu} q_2}{p_1 | p_2 \xrightarrow{\mu} p_1 | q_2}$$

$$\text{Rec) } \frac{p[\mathbf{rec} \ x. \ p / x] \xrightarrow{\mu} q}{\mathbf{rec} \ x. \ p \xrightarrow{\mu} q}$$

# HML

## Logica di Hennessy-Milner

# Equivalenza logica

Prendiamo un altro approccio all'equivalenza

definiamo una logica (insieme di formule)

un processo può soddisfare o meno una formula

due processi sono (logicamente) equivalenti

quando soddisfano esattamente le stesse formule

le formule devono descrivere le proprietà comportamentali dei processi

la capacità/incapacità di eseguire transizioni

(logica modale: possibilmente, necessariamente)

allora, possiamo comporre le formule con i soliti operatori

# Logica di Hennessy-Milner

Presentiamo gli operatori di base

multi-modale:

gli operatori modali sono parametrizzati da azioni

nessuna negazione:

anche la controparte di una formula può essere scritta come una formula

nessuna ricorsione:

ogni formula esprime proprietà su un numero finito di passi in avanti

# HML: sintassi

$F, G$	$::=$	<b>tt</b>	true
		<b>ff</b>	false
		$\bigwedge_{i \in I} F_i$	conjunction
		$\bigvee_{i \in I} F_i$	disjunction
		$\diamond_{\mu} F$	diamond operator $\langle \mu \rangle F$
		$\square_{\mu} F$	box operator $[\mu] F$

$\mathcal{L}$  insieme di tutte le formule

# HML: semantica

$p \models F$  si legge “ $p$  soddisfa  $F$ ”

definito induttivamente sulla struttura della formula

$p \models \mathbf{tt}$  tutti i processi soddisfano true  
(nessun processo soddisfa false)

$p \models \bigwedge_{i \in I} F_i$  iff  $\forall i \in I. p \models F_i$   $p$  soddisfa tutte le  $F_i$

$p \models \bigvee_{i \in I} F_i$  iff  $\exists i \in I. p \models F_i$   $p$  almeno una delle  $F_i$

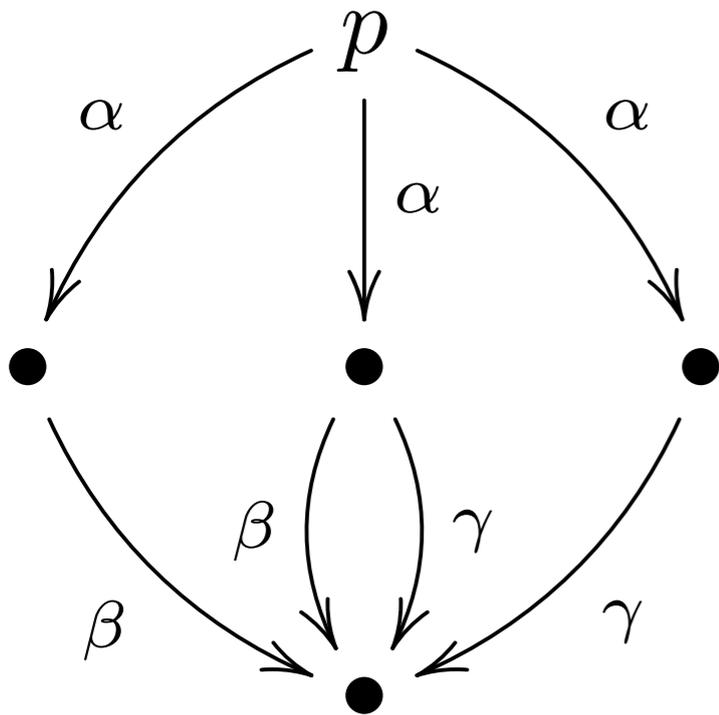
$p \models \diamond_{\mu} F$  iff  $\exists p'. p \xrightarrow{\mu} p' \wedge p' \models F$   $p$  fa un  $\mu$ -passo  
e poi soddisfa  $F$

$p \models \square_{\mu} F$  iff  $\forall p'. p \xrightarrow{\mu} p' \Rightarrow p' \models F$   $F$  e' soddisfatta dopo ogni  
 $\mu$ -passo di  $p$

# Esempi

- $\diamond_{\alpha} \mathbf{tt}$  soddisfatta da qualsiasi processo che può fare un  $\alpha$ -passo
- $\square_{\beta} \mathbf{ff}$  soddisfatta da qualsiasi processo che non può fare nessun  $\beta$ -passo
- $\diamond_{\alpha} \mathbf{ff}$  stesso che  $\mathbf{ff}$   
se un processo non può fare  $\alpha$ , la modalità viene persa  
continuation cannot satisfy  $\mathbf{ff}$
- $\square_{\beta} \mathbf{tt}$  stesso che  $\mathbf{tt}$   
se un processo non può fare  $\beta$  la modalità è banalmente valida  
se un processo fa  $\beta$  la sua continuazione soddisfa  $\mathbf{tt}$
- $\diamond_{\alpha} (\diamond_{\beta} \mathbf{tt} \wedge \square_{\gamma} \mathbf{ff})$  soddisfatto da qualsiasi processo che può fare  $\alpha$   
e raggiungere un processo che può fare  $\beta$  ma non  $\gamma$

# Esempi



$$p \stackrel{?}{\models} \diamond_{\alpha} \mathbf{tt}$$



$$p \stackrel{?}{\models} \square_{\alpha} \diamond_{\beta} \mathbf{tt}$$



$$p \stackrel{?}{\models} \diamond_{\alpha} \square_{\beta} \mathbf{ff} \wedge \diamond_{\alpha} \square_{\gamma} \mathbf{ff}$$



$$p \stackrel{?}{\models} \square_{\alpha} (\diamond_{\beta} \mathbf{tt} \vee \diamond_{\gamma} \mathbf{tt})$$



$$p \stackrel{?}{\models} \square_{\alpha} (\diamond_{\beta} \mathbf{tt} \wedge \diamond_{\gamma} \mathbf{tt})$$



$$p \stackrel{?}{\models} \diamond_{\alpha} (\diamond_{\beta} \mathbf{tt} \wedge \diamond_{\gamma} \mathbf{tt})$$



# Negazione

non presente nella sintassi, ma non necessaria

ogni formula  $F$  ha una formula inversa  $F^c$  tale che

$$\forall p. p \models F \quad \text{iff} \quad p \not\models F^c$$

$F^c$  può essere definito per induzione strutturale

$$\mathbf{tt}^c \triangleq \mathbf{ff}$$

$$\mathbf{ff}^c \triangleq \mathbf{tt}$$

$$\left( \bigwedge_{i \in I} F_i \right)^c \triangleq \bigvee_{i \in I} F_i^c$$

$$\left( \bigvee_{i \in I} F_i \right)^c \triangleq \bigwedge_{i \in I} F_i^c$$

$$\left( \diamond_{\mu} F \right)^c \triangleq \square_{\mu} F^c$$

$$\left( \square_{\mu} F \right)^c \triangleq \diamond_{\mu} F^c$$

$$\left( \diamond_{\alpha} \mathbf{tt} \right)^c = \square_{\alpha} \mathbf{tt}^c = \square_{\alpha} \mathbf{ff} \quad \begin{array}{l} \text{esempio} \\ \text{(posso fare } \alpha)^c = \\ \text{non posso fare } \alpha \end{array}$$

# Sintassi estesa

$$A = \{\mu_1, \dots, \mu_n\}$$

$$\begin{aligned} \diamond_A F &\triangleq \diamond_{\mu_1} F \vee \dots \vee \diamond_{\mu_n} F & \square_A F &\triangleq \square_{\mu_1} F \wedge \dots \wedge \square_{\mu_n} F \\ &= \bigvee_{i \in [1, n]} \diamond_{\mu_i} F & &= \bigwedge_{i \in [1, n]} \square_{\mu_i} F \end{aligned}$$

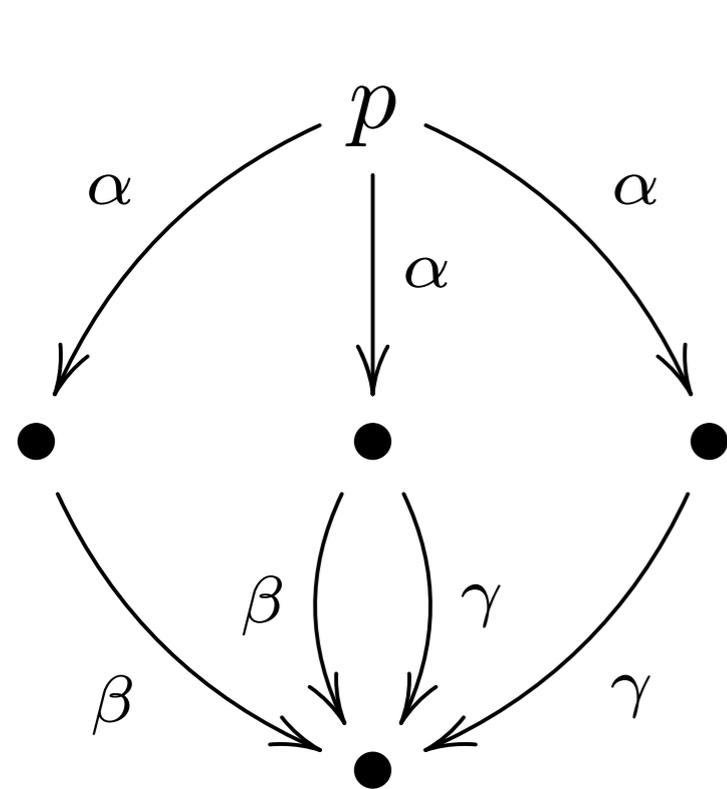
$$\diamond_{\emptyset} F \triangleq \mathbf{ff}$$

$$\square_{\emptyset} F \triangleq \mathbf{tt}$$

# HML: Equivalenza logica

due processi sono equivalenti sse soddisfano le stesse formule

$$p \equiv_{\text{HM}} q \quad \text{sse} \quad \forall F \in \mathcal{L}. (p \models F \Leftrightarrow q \models F)$$



$$p \models F$$

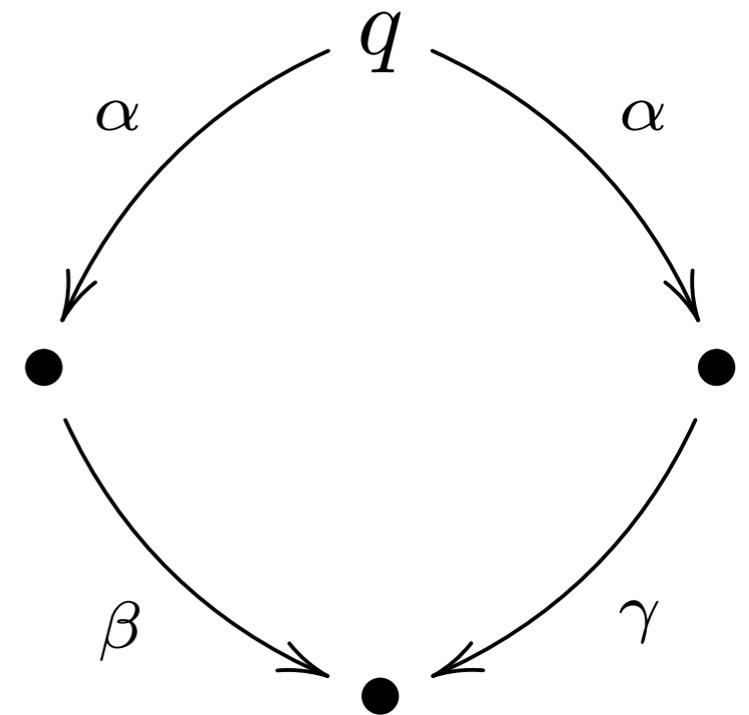
$$p \not\models F^c$$

$$p \stackrel{?}{\equiv}_{\text{HM}} q$$



$$F \triangleq \diamond_{\alpha}(\diamond_{\beta} \mathbf{tt} \wedge \diamond_{\gamma} \mathbf{tt})$$

$$F^c \triangleq \square_{\alpha}(\square_{\beta} \mathbf{ff} \vee \square_{\gamma} \mathbf{ff})$$



$$q \not\models F$$

$$q \models F^c$$

# Bis. forte come equiv. logica

**TH.** per qualsiasi processo finitamente ramificato  $p, q$

$$p \simeq q \quad \text{sse} \quad p \equiv_{\text{HM}} q$$

(prova omessa)

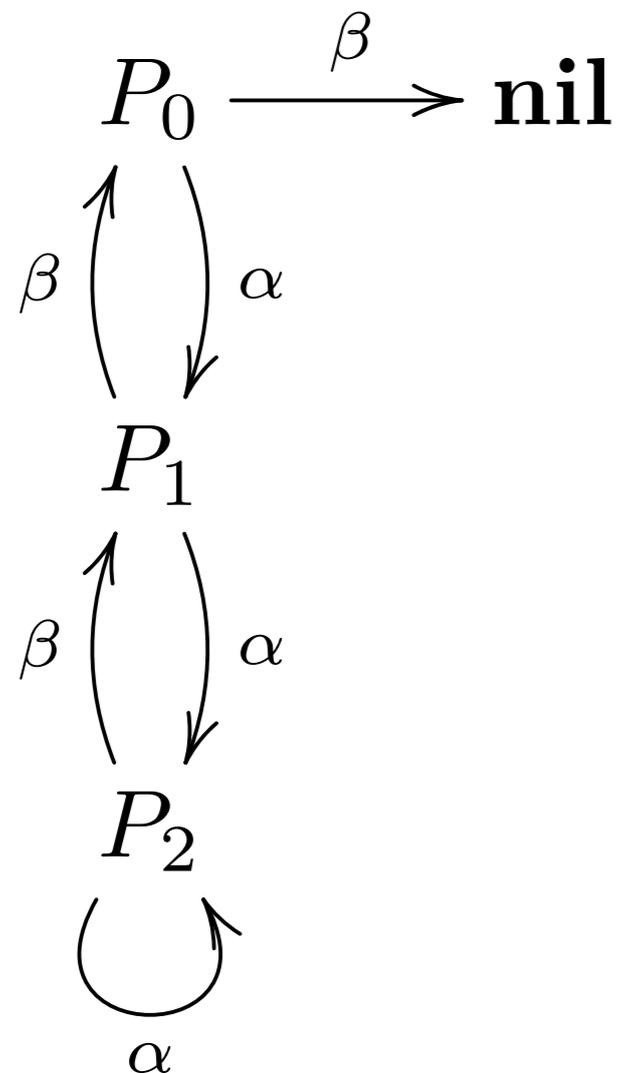
conseguenze:

per dimostrare che due processi sono fortemente bisimili:  
esibire una relazione di bisimulazione forte che li metta in relazione

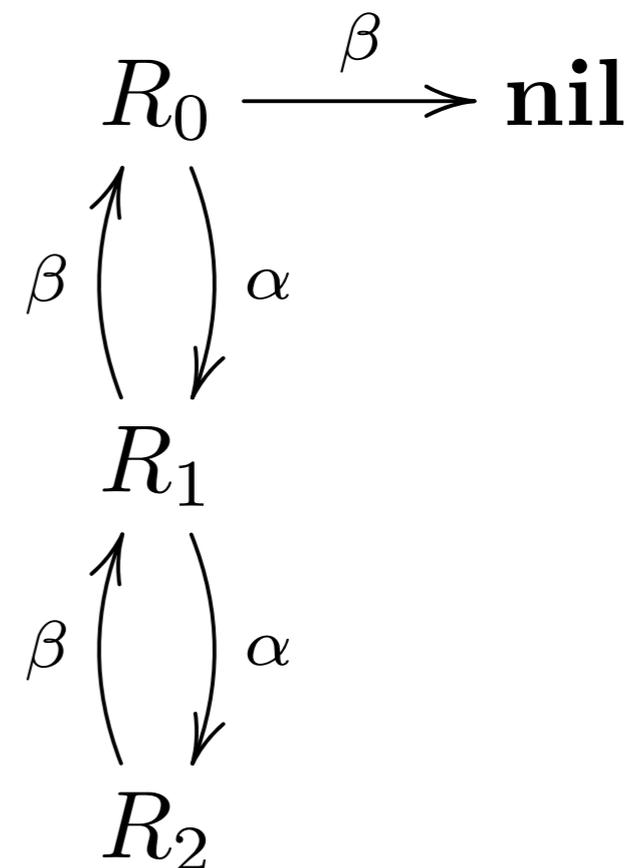
per dimostrare che due processi non sono fortemente bisimili: esibire una formula HML che li distingua

# Esercizio

trovare una formula HML che distingue i due processi



$$P_0 \neq R_0$$



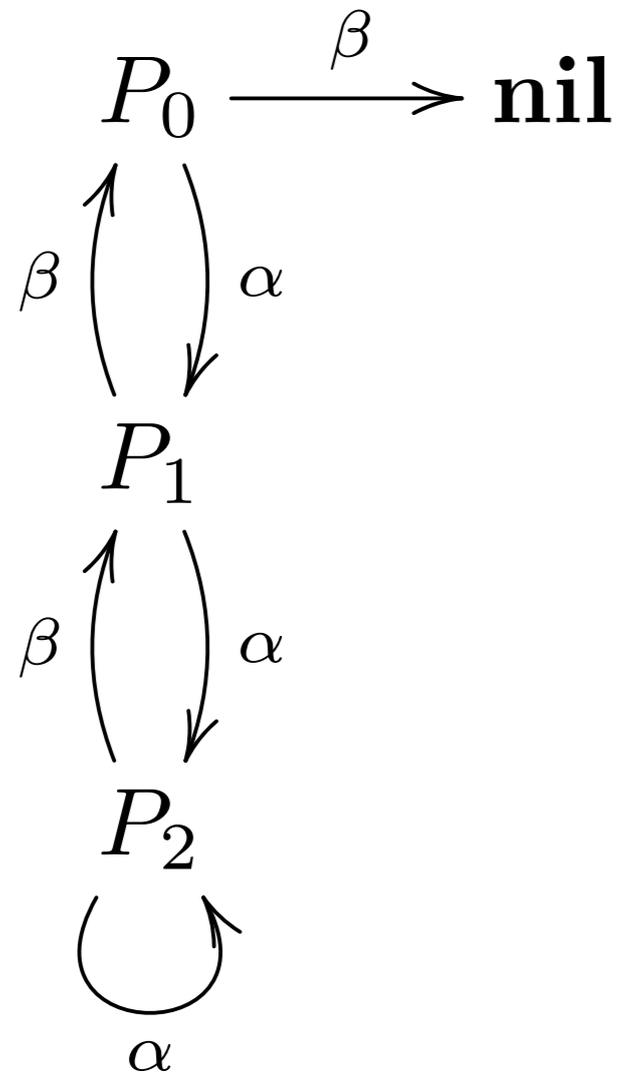
$$P_0 \models F$$

$$F \triangleq \diamond_{\alpha} \diamond_{\alpha} \diamond_{\alpha} \mathbf{tt}$$

$$R_0 \not\models F$$

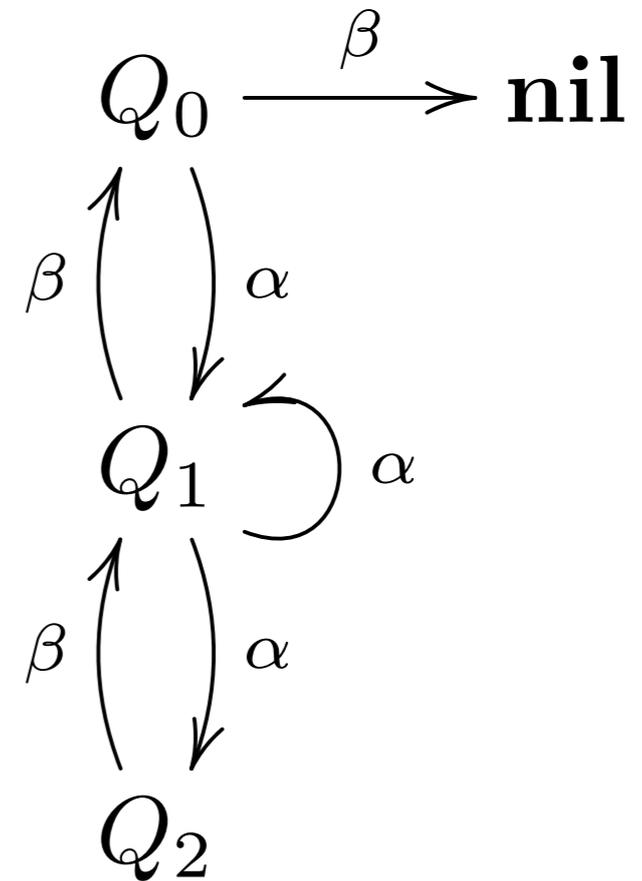
# Esercizio

trovare una formula HML che distingue i due processi



$$P_0 \models F$$

$$P_0 \not\equiv Q_0$$



$$Q_0 \not\models F$$

$$F \triangleq \diamond_{\alpha} \square_{\alpha} \diamond_{\alpha} \mathbf{tt}$$