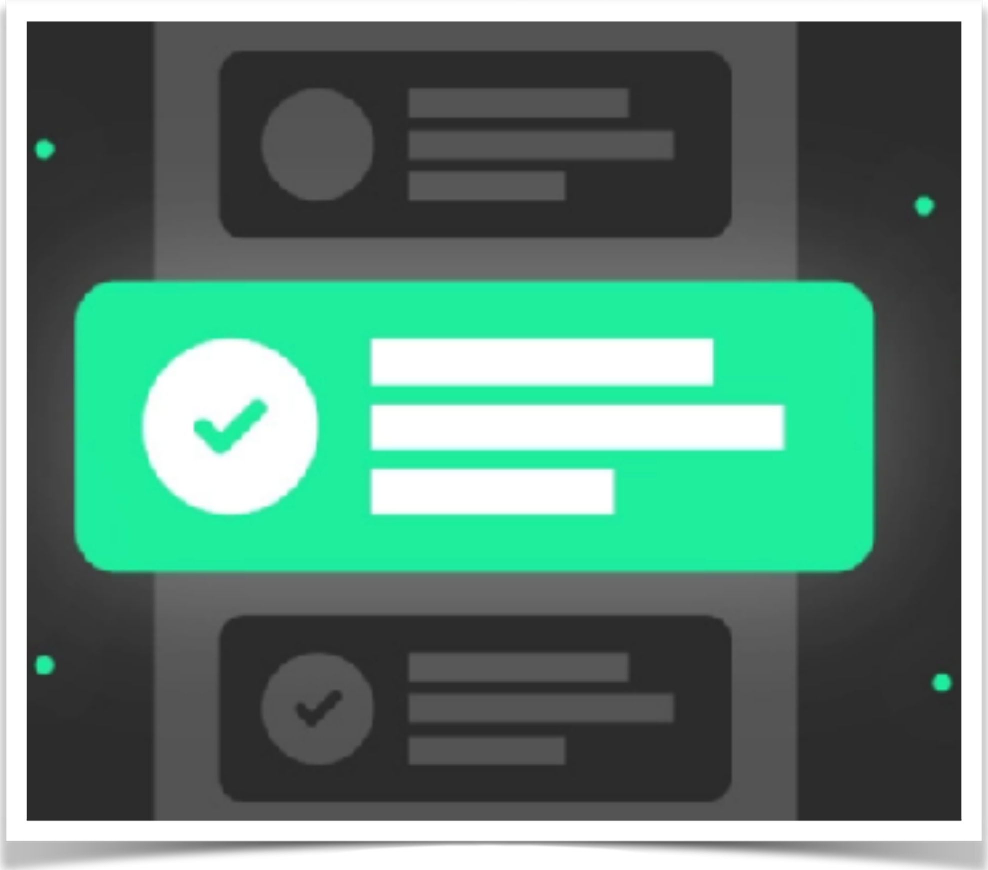


http://didawiki.di.unipi.it/doku.php/magistraleinformatica/psc/start

PSC 2022/23 (375AA, 9CFU)

Principles for Software Composition

Roberto Bruni
http://www.di.unipi.it/~bruni/

19 - Hennessy-Milner Logic

CCS syntax

p, q	$::=$	$\mathbf{nil}$	inactive process
	$\mid$	x	process variable (for recursion)
	$\mid$	$\mu.p$	action prefix
	$\mid$	$p\backslash\alpha$	restricted channel
	$\mid$	$p[\phi]$	channel relabelling
	$\mid$	$p + q$	nondeterministic choice (sum)
	$\mid$	$p \mid q$	parallel composition
	$\mid$	$\mathbf{rec}\ x.\ p$	recursion

(operators are listed in order of precedence)

CCS op. semantics

Act) $$\frac{}{\mu.p \xrightarrow{\mu} p}$$

Res) $$\frac{p \xrightarrow{\mu} q \quad \mu \notin \{\alpha, \overline{\alpha}\}}{p\backslash\alpha \xrightarrow{\mu} q\backslash\alpha}$$

Rel) $$\frac{p \xrightarrow{\mu} q}{p[\phi] \xrightarrow{\phi(\mu)} q[\phi]}$$

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

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

ParL) $$\frac{p_1 \xrightarrow{\mu} q_1}{p_1|p_2 \xrightarrow{\mu} q_1|p_2}$$

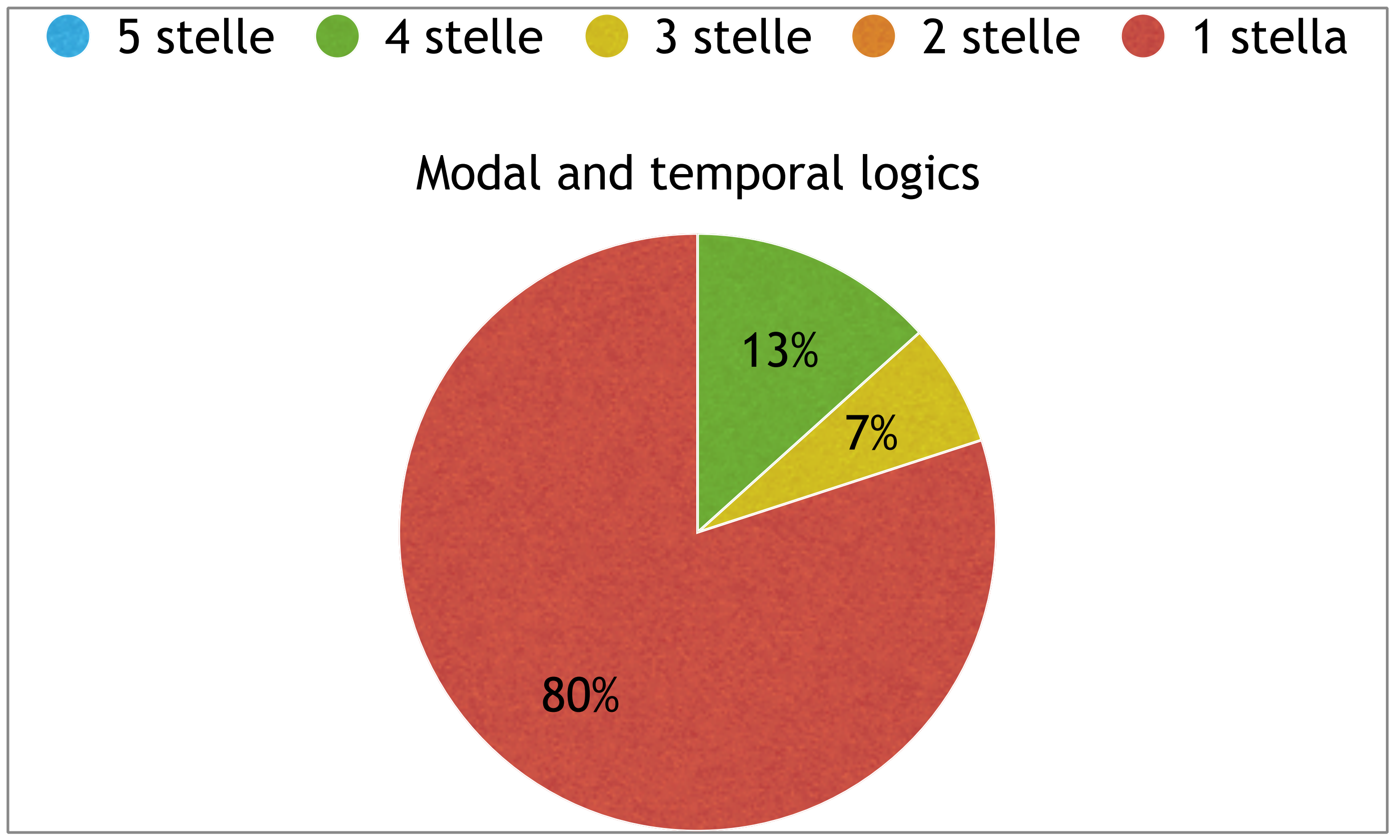
Com) $$\frac{p_1 \xrightarrow{\lambda} q_1 \quad p_2 \xrightarrow{\overline{\lambda}} q_2}{p_1|p_2 \xrightarrow{\tau} q_1|q_2}$$

ParR) $$\frac{p_2 \xrightarrow{\mu} q_2}{p_1|p_2 \xrightarrow{\mu} p_1|q_2}$$

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

HML
Hennessy-Milner Logic

From your forms



(over 15 answers)

Logical equivalence

Let us take another approach to equivalence

we define some logic (set of formulas)

a process may or may not satisfy a formula

two processes are (logically) equivalent
when they satisfy exactly the same formulas

formulas must describe behavioural properties of processes

the ability / inability to perform transitions
(modal logic: possibly, necessarily)

then, we can compose formulas with usual operators

Hennessy-Milner Logic

We present the core operators

multi-modal:
modal operators are parameterised by actions

no negation:
the converse of a formula can also be written as a formula

no recursion:
each formula express properties about finite steps ahead

denotational semantics of a formula (postponed):
set of processes that satisfy the formula

HML: syntax

$$
\begin{array}{rcll}
F, G & ::= & \mathbf{tt} & \text{true} \\
& | & \mathbf{ff} & \text{false} \\
& | & \bigwedge_{i \in I} F_i & \text{conjunction} \\
& | & \bigvee_{i \in I} F_i & \text{disjunction} \\
& | & \Diamond_\mu F & \text{diamond operator} \quad \langle \mu \rangle F \\
& | & \Box_\mu F & \text{box operator} \quad [\mu] F
\end{array}
$$

$\mathcal{L}$ set of all formulas

HML: semantics

$p \models F$ reads " *p* satisfies *F* "

defined inductively on the structure of the formula

$p \models \mathbf{tt}$ any process satisfies true (no process satisfies false)

$p \models \bigwedge_{i \in I} F_i \quad \text{iff} \quad \forall i \in I.\ p \models F_i$ *p* satisfies all F_i

$p \models \bigvee_{i \in I} F_i \quad \text{iff} \quad \exists i \in I.\ p \models F_i$ *p* satisfies one of the F_i

$p \models \Diamond_\mu F \quad \text{iff} \quad \exists p'.\ p \xrightarrow{\mu} p' \wedge p' \models F$ *p* can make one μ-step and then satisfy *F*

$p \models \Box_\mu F \quad \text{iff} \quad \forall p'.\ p \xrightarrow{\mu} p' \Rightarrow p' \models F$ *F* is satisfied after any μ-step of *p*

Examples

$\Diamond_\alpha \mathbf{tt}$ satisfied by any process that can make an *α*-step

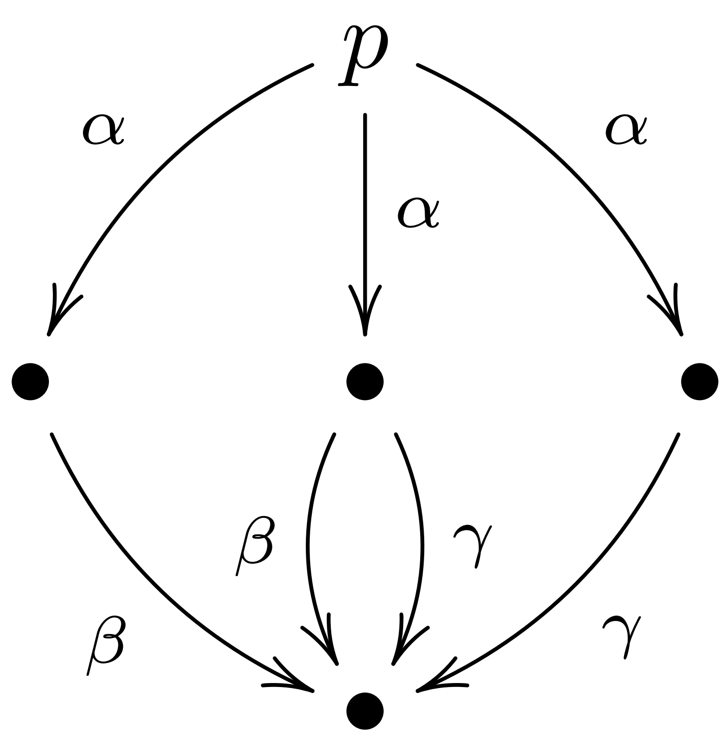
$\Box_\beta \mathbf{ff}$ satisfied by any process that cannot make a *β*-step

$\Diamond_\alpha \mathbf{ff}$ same as **ff**
if a process cannot do *α* the modality is missed
if a process can do *α* its continuation cannot satisfy **ff**

$\Box_\beta \mathbf{tt}$ same as **tt**
if a process cannot do *β* the modality holds trivially
if a process does *β* its continuation will satisfy **tt**

$\Diamond_\alpha(\Diamond_\beta \mathbf{tt} \wedge \Box_\gamma \mathbf{ff})$ satisfied by any process the can do *α* and reach a process that can do *β* but not *γ*

Examples



$$p \overset{?}{\models} \Diamond_{\alpha}\mathbf{tt}$$ ✓

$$p \overset{?}{\models} \Box_{\alpha}\Diamond_{\beta}\mathbf{tt}$$ ✗

$$p \overset{?}{\models} \Diamond_{\alpha}\Box_{\beta}\mathbf{ff} \ \wedge \ \Diamond_{\alpha}\Box_{\gamma}\mathbf{ff}$$ ✓

$$p \overset{?}{\models} \Box_{\alpha}(\Diamond_{\beta}\mathbf{tt} \vee \Diamond_{\gamma}\mathbf{tt})$$ ✓

$$p \overset{?}{\models} \Box_{\alpha}(\Diamond_{\beta}\mathbf{tt} \wedge \Diamond_{\gamma}\mathbf{tt})$$ ✗

$$p \overset{?}{\models} \Diamond_{\alpha}(\Diamond_{\beta}\mathbf{tt} \wedge \Diamond_{\gamma}\mathbf{tt})$$ ✓

Negation

not present in the syntax, but not needed

any formula *F* has a *converse* formula F^c such that

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

F^c can be defined by structural induction

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

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

$$(\Diamond_\mu F)^c \triangleq \Box_\mu F^c \qquad (\Box_\mu F)^c \triangleq \Diamond_\mu F^c$$

example

$(\Diamond_\alpha \mathbf{tt})^c = \Box_\alpha \mathbf{tt}^c = \Box_\alpha \mathbf{ff}$ (can do *a*)^c = cannot do *a*

Extended syntax

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

$$\Diamond_A F \triangleq \Diamond_{\mu_1} F \vee \cdots \vee \Diamond_{\mu_n} F = \bigvee_{i \in [1,n]} \Diamond_{\mu_i} F$$

$$\Box_A F \triangleq \Box_{\mu_1} F \wedge \cdots \wedge \Box_{\mu_n} F = \bigwedge_{i \in [1,n]} \Box_{\mu_i} F$$

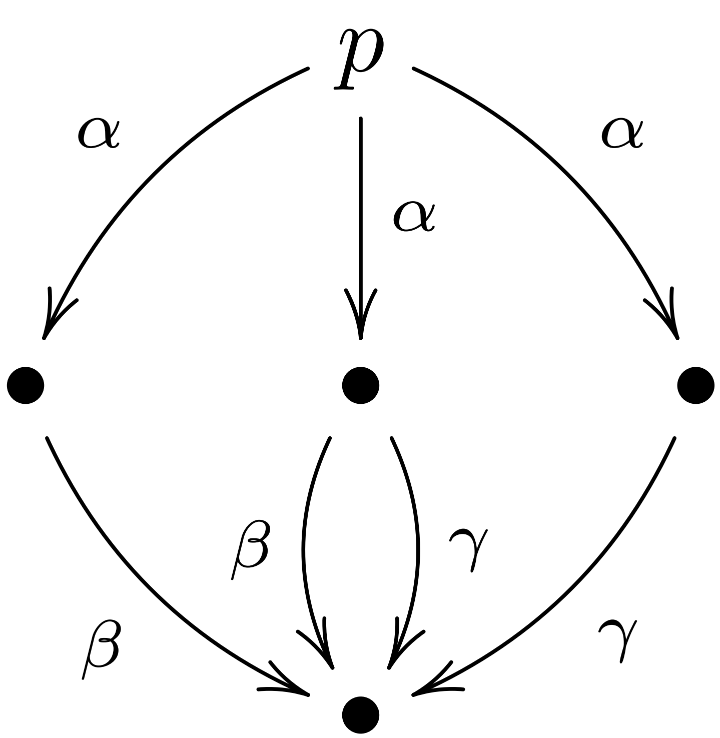
$$\Diamond_\emptyset F \triangleq \mathbf{ff}$$

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

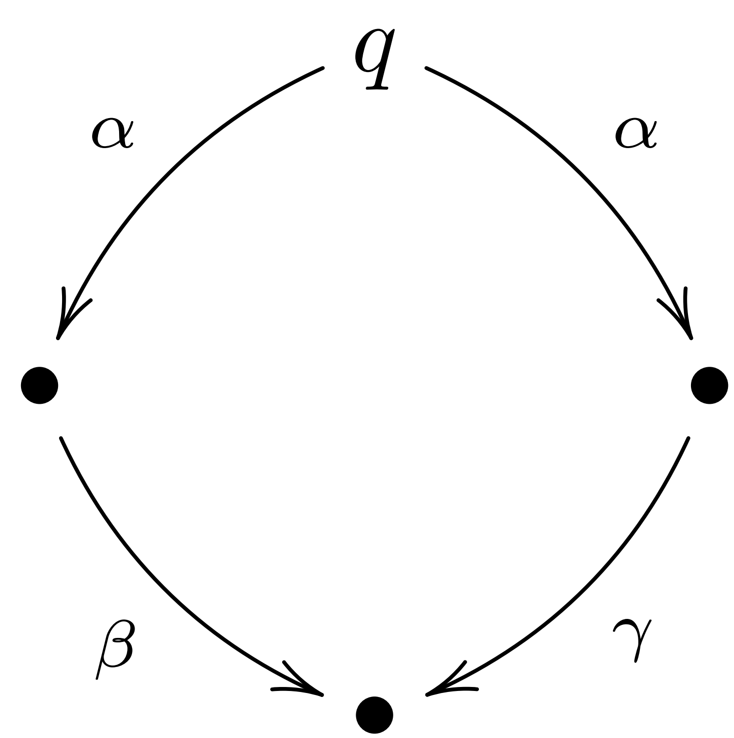
HML: logical equivalence

two processes are equivalent iff they satisfy the same formulas

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



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



$p \models F$ $\qquad F \triangleq \Diamond_\alpha(\Diamond_\beta \mathbf{tt} \wedge \Diamond_\gamma \mathbf{tt})$ $\qquad q \not\models F$

$p \not\models F^c$ $\qquad F^c \triangleq \Box_\alpha(\Box_\beta \mathbf{ff} \vee \Box_\gamma \mathbf{ff})$ $\qquad q \models F^c$

Strong bis as logic equiv

TH. for any finitely branching processes *p,q*

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

(proof omitted)

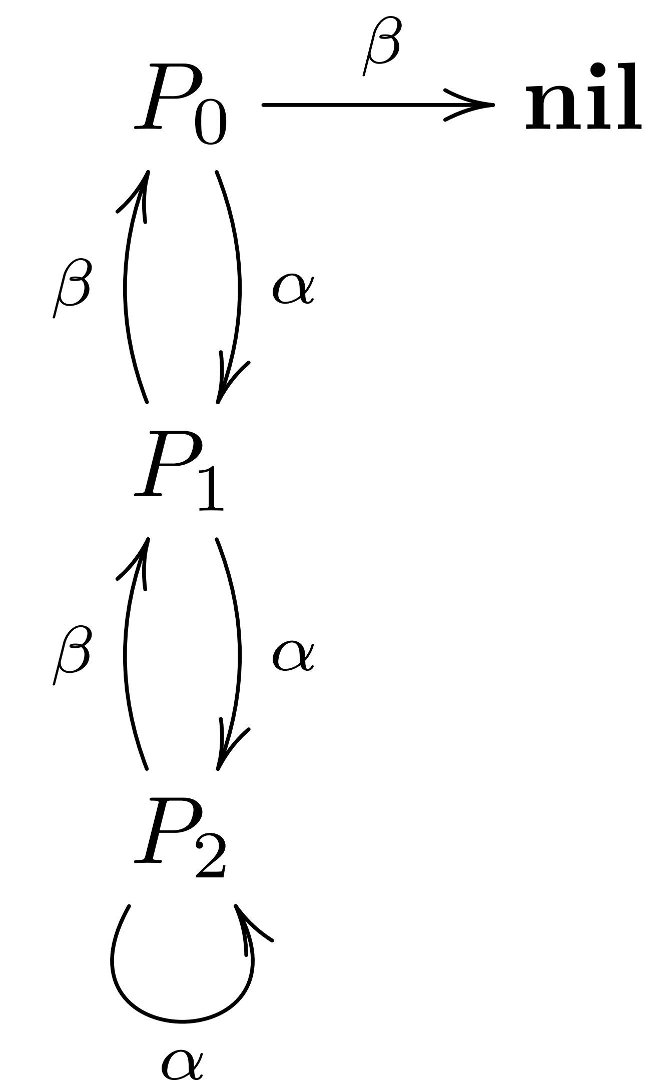
consequences:

to show that two processes are strong bisimilar:
exhibit a strong bisimulation relation that relates them

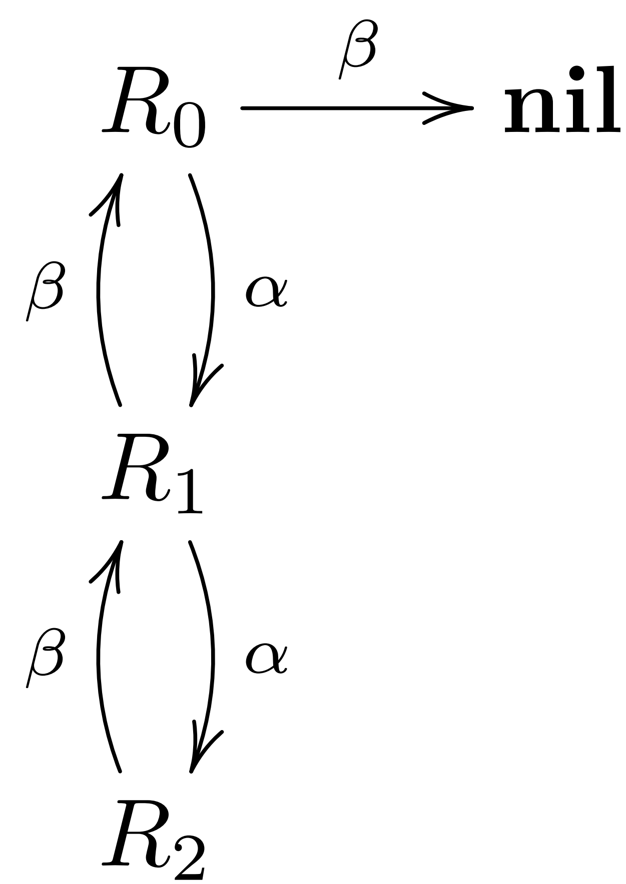
to show that two processes are not strong bisimilar:
exhibit a HML formula that distinguishes between them

Exercise

find a HML formula that distinguishes the two processes



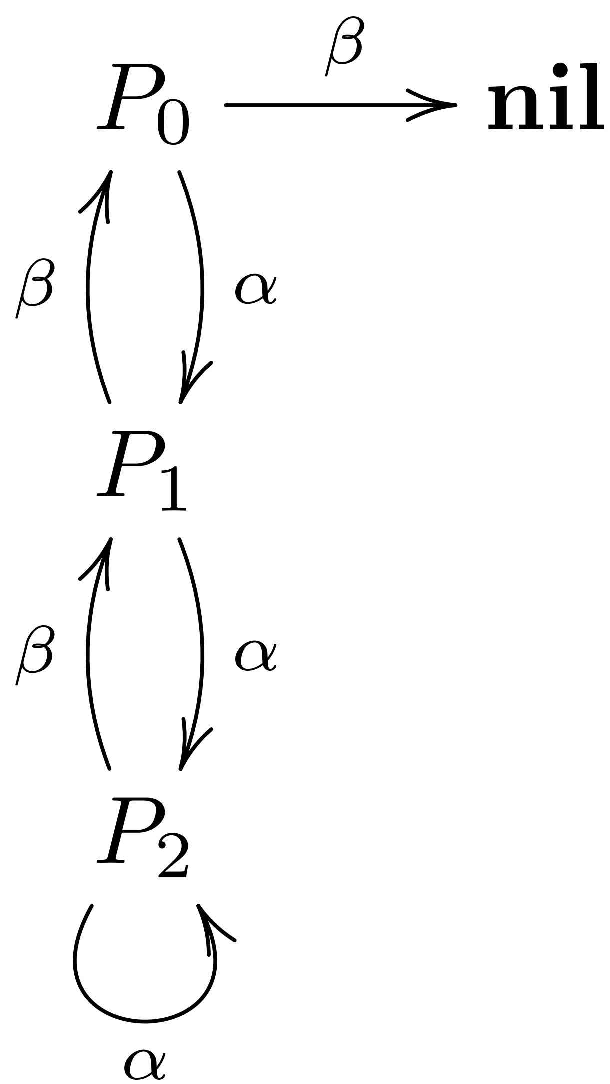
$P_0 \not\simeq R_0$



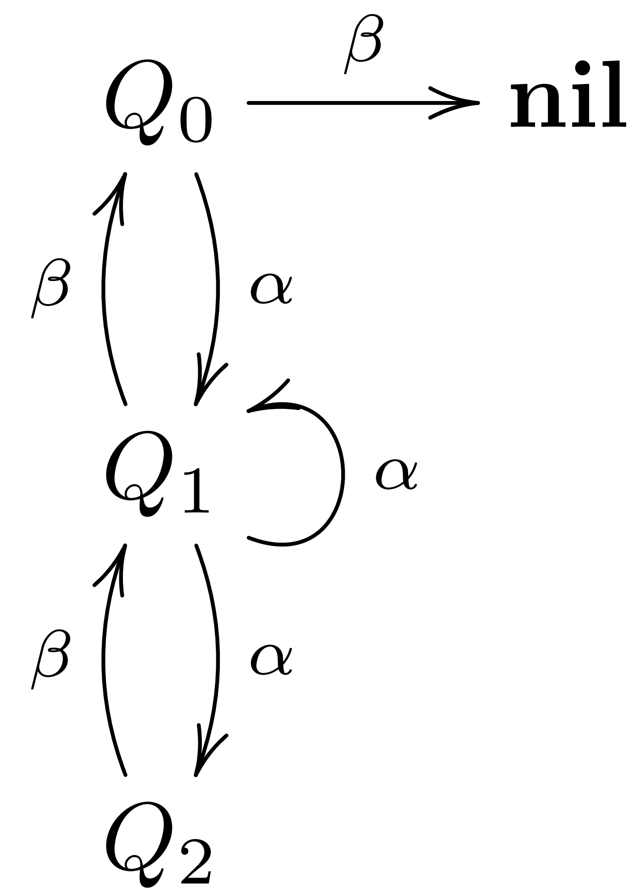
$P_0 \models F$ $\qquad F \triangleq \Diamond_\alpha \Diamond_\alpha \Diamond_\alpha \mathbf{tt}$ $\qquad R_0 \not\models F$

Exercise

find a HML formula that distinguishes the two processes



$$P_0 \not\approx Q_0$$



$P_0 \models F$ $\qquad F \triangleq \Diamond_\alpha \Box_\alpha \Diamond_\alpha \mathbf{tt}$ $\qquad Q_0 \not\models F$