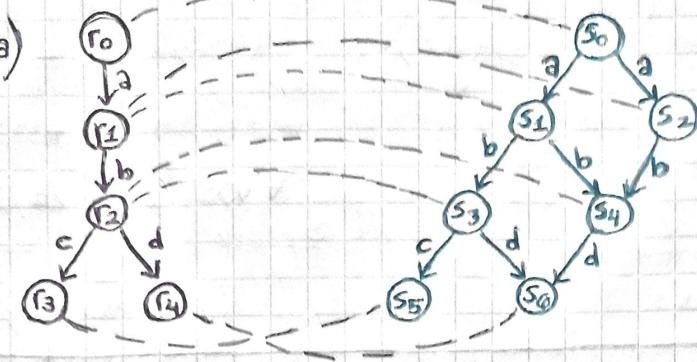


## Ingeniería de Software II

### Parcial 1

① a)

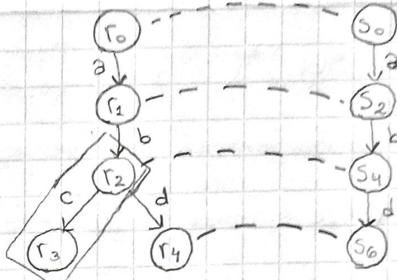


Notemos que  $r_2 \xrightarrow{c} r_3$  pero de  $s_4$  no sale ninguna transición c a algún  $s_i$ .

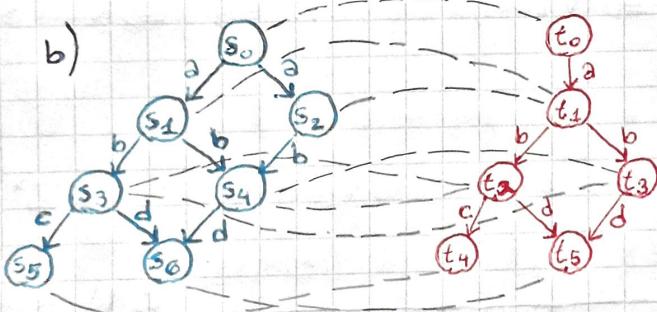
$\therefore$  No son ready-similares.

Se puede ver que son similares porque hay simulación de  $r_0$  a  $s_0$  y también de  $s_0$  a  $r_0$ .

Pero podemos observar que cuando  $(s_0)$  está simulando a  $(r_0)$  tenemos que:



b)



Se puede ver que son similares porque hay simulación de  $s_0$  a  $t_0$  y de  $t_0$  a  $s_0$ .

Notar que se cumple que:

$$(\exists t' \in S : t \xrightarrow{a} t') \Rightarrow (\exists s' \in S : s \xrightarrow{a} s')$$

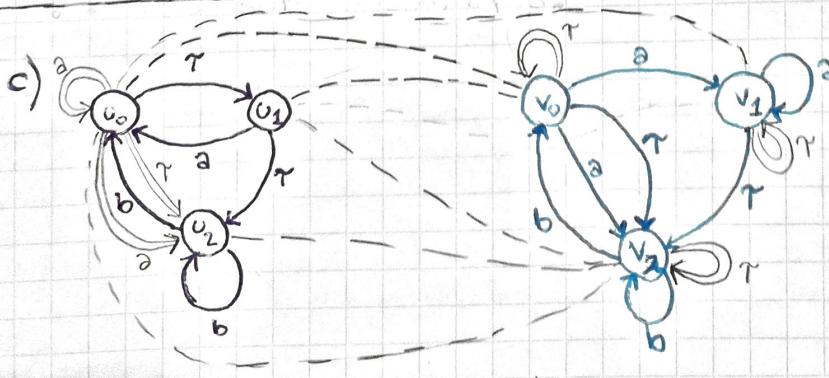
Supongamos que  $(s_0)$  está siendo simulado por  $(t_0)$

$\Rightarrow (s_0)$  propone hacer  $s_0 \xrightarrow{a} s_2$ , entonces  $t_0 \xrightarrow{a} t_1$

Ahora se hace un cambio de rol  $\Rightarrow (t_1)$  propone hacer  $t_1 \xrightarrow{b} t_2$ , entonces  $s_2 \xrightarrow{b} s_4$ .

$\Rightarrow (t_2)$  propone hacer  $t_2 \xrightarrow{c} t_4$  pero  $(s_4)$  no puede imitarlo.

$\therefore$  No son bisimilares.



$$\begin{aligned} U &\text{ es simulado por } V \\ U_0 &\xrightarrow{\gamma} U_1 \equiv V_0 \xrightarrow{\gamma} V_0 \\ U_0 &\xrightarrow{\gamma} U_1 \equiv V_1 \xrightarrow{\gamma} V_1 \end{aligned}$$

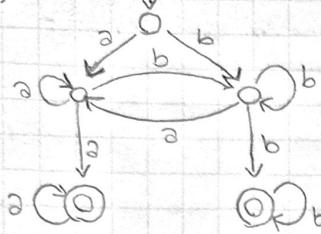
V es simulado por U

$$\begin{aligned} V &\xrightarrow{a} V_1 \equiv U_0 \xrightarrow{a} U_0 \\ V_1 &\xrightarrow{a} V_1 \equiv U_0 \xrightarrow{a} U_0 \\ V_1 &\xrightarrow{\gamma} V_2 \equiv U_0 \xrightarrow{\gamma} U_2 \\ V_0 &\xrightarrow{a} V_2 \equiv U_0 \xrightarrow{a} U_2 \end{aligned}$$

$\therefore$  Existe una bisimulación débil entre  $U_0$  y  $V_0$ .

$$\textcircled{2} \quad \Sigma = \{a, b\}$$

a)



Un lenguaje  $\omega$ -regular que representa al automata

$$\text{es: } (a+b)^*(a^w + b^w)$$

Se puede observar que todos los prefijos aparecen en la propiedad  $(a+b)^*$   $\Rightarrow$  es de Liveness

Se nota que se cumple que:  $\forall \tau: \forall i \geq 0: \exists B: \tau[i..]B \in P \Rightarrow \tau \in P \Rightarrow$  es de Safety

$\therefore$  Es de ambos (Safety y Liveness). //

b)  $(a^*bb)^*(a^w + b^w)$  acepta cantidad par de b's en la traza

Un contraejemplo para ver que no es de safety puede ser:

$$\tau = b a \dots \text{ donde } \tau \notin P \text{ pero } \exists i \geq 0: \exists B: \tau[..i]B \in P.$$

$$\text{Si } i=0 \Rightarrow \tau[..0] = b \wedge B = b^w \Rightarrow \tau[..0]B = bb \dots \in P \Rightarrow \text{No es Safety}$$

Para ver si es de liveness podemos chequear por su definición que

$$\forall \alpha \in \Sigma^*: \exists B \in \Sigma^w: \alpha B \in P \Rightarrow \text{supongo entonces } \alpha = ab \in \Sigma^*$$

$$\Rightarrow \text{Notemos que } \forall \beta \in \Sigma^w \quad \alpha \beta \notin P \Rightarrow \text{No es de liveness.}$$

$\therefore$  No es de ninguno

c) Si  $P = \{\tau \mid \forall i \geq 0: \#_a^i(\tau) \geq \#_b^i(\tau)\} \Rightarrow$  el complemento de  $P$

$$\text{es } \overline{P} = \{\tau \mid \forall i \geq 0: \#_a^i(\tau) < \#_b^i(\tau)\}$$

③ a)  $\phi \sqsubset \psi \equiv \text{"Siempre ocurre } \phi \text{ antes que } \psi \text{"}$  donde  $\phi, \psi$  son fórmulas LTL.

$$\text{Se sabemos } \tau \models \phi \sqsubset \psi \iff \forall i: i \geq 0: \tau[i..] \models \psi \implies \exists j: 0 \leq j < i: \tau[j..] \models \phi$$

$$\equiv \forall i: i \geq 0: (\tau[i..] \models \psi \implies \exists j: 0 \leq j < i: \tau[j..] \models \phi)$$

$$\equiv \forall i: i \geq 0: \neg(\tau[i..] \models \psi) \circ (\exists j: 0 \leq j < i: \tau[j..] \models \phi)$$

$$\equiv \neg(\forall i: i \geq 0: \neg(\tau[i..] \models \psi) \circ \exists j: 0 \leq j < i: \tau[j..] \models \phi)$$

$$\equiv \neg(\neg(\forall i: i \geq 0: \neg(\tau[i..] \models \psi) \circ \exists j: 0 \leq j < i: \tau[j..] \models \phi))$$

$$\equiv \neg(\exists i: i \geq 0: \neg(\tau[i..] \models \psi) \wedge \neg(\exists j: 0 \leq j < i: \tau[j..] \models \phi))$$

$$\equiv \neg(\exists i: i \geq 0: \tau[i..] \models \psi \wedge \neg(\exists j: 0 \leq j < i: \tau[j..] \models \phi))$$

$$\tau \models \neg(\neg \phi \cup \psi) \iff \neg(\exists i: i \geq 0: \tau[i..] \models \psi \wedge \neg(\exists j: 0 \leq j < i: \tau[j..] \models \phi))$$

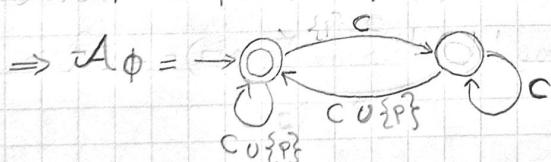
$$\therefore \phi \sqsubset \psi \equiv \neg(\neg \phi \cup \psi) //$$

b) I)  $\square(\text{env\_msj}_1 \wedge \text{env\_msj}_2) \rightarrow \diamond \text{col\_det}_1$

II)  $\square(\text{env\_msj}_1 \wedge \text{env\_msj}_2) \rightarrow (\neg \text{env\_msj}_1 \vee \text{col\_det}_1)$

④ Un modelo finito  $M \models \phi$  si  $L(M) \subseteq L(\phi)$  donde  $\hat{\phi} = \square \diamond p$

como  $\phi = \square \diamond p \Rightarrow \frac{C}{\square} \xrightarrow{C} \frac{C}{\{p\} \cup C} \xrightarrow{C} \dots \xrightarrow{C} \frac{C}{\{p\} \cup C} \xrightarrow{C} \dots$



y como  $\neg \phi = \neg(\square \diamond p) \Rightarrow A_{\neg \phi} = \xrightarrow{C} \begin{array}{c} \bullet \\ \circlearrowright \end{array} \xrightarrow{C} \dots$

Por otro lado como  $M$  es un sistema de transiciones, sabemos que  $L(M)$  es el conjunto de todas las trazas que puede ejecutarse (es decir, su comportamiento).

Ademas  $M$  puede verse como un automata de Büchi  $A_M \Rightarrow L(M) = L(A_M)$

$\Rightarrow$  verificar  $L(M) \subseteq L(\phi)$  es equivalente a chequear  $L(A_M) \subseteq L(A_\phi)$

que a su vez es equivalente a chequear  $L(A_M) \cap L(A_{\neg \phi}) = \emptyset$

$\Rightarrow$  "Desde el estado inicial del grafo subyacente a  $M$  no se alcanza ningún ciclo tal que  $p$  no es válido en ninguno de los nodos de dicho ciclo."

(5) a)

