Partie 1 – Modèle comportemental comparaisons entre systèmes

- Partie 1 – modèle comportemental de système
  - Système de Transitions Etiquetées (STE / *LTS*)
  - sémantique STE de la concurrence synch./asynch.
  - STE communicants
  - autour de la comparaison de STE
- Partie 2 – ”langages” de processus
- Partie 3 – propriétés

Le point et les objectifs

- le comportement d'un système simple (séquentiel) peut être modélisé par un STE
- le comportement d'un système composite (concurrent) peut être modélisé par une architecture et le produit de STE
- comment savoir si:
  - un système correspond à un besoin ?
  - un système peut en remplacer un autre ?

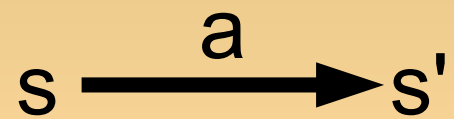
[Intra]-STE vs. [Inter]-STE

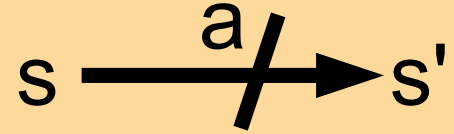
Dans ce qui suit, nous allons nous interesser :

- [INTRA] soit aux propriétés d'un STE
  note: on reverra ça (mais autrement) dans la partie 2
- [INTER] soit à des relations entre plusieurs STE (préordre, equivalence, ...)
- les deux sont liés car on a besoin de réfléchir à [INTRA] pour travailler sur [INTER]
- des exemples seront faits au fur et à mesure au tableau

Notations et définitions

- soit un STE $L = (A,S,s_0,F,T)$,
  quelques notations et définitions :
  - $s\text{–}a\text{–}>s' : (s,a,s') \in T$
  - $s\text{–}a\text{–}> : \exists\ s' \in S\ .\ s\text{–}a\text{–}>s'$
  - $s\text{–}>s' : \exists\ a \in A\ .\ s\text{–}a\text{–}>s'$
  - $s\text{–}/a\text{–}> : \neg\ s\text{–}a\text{–}>$
  - $s\text{–}/\text{–}> : \neg\ \exists\ a \in A\ .\ s\text{–}a\text{–}>$
  - $t=a_1a_2...a_n,\ n \in \mathbb{N}\ .\ \forall\ a_i\ .\ a_i \in A$
    mot sur A^n (on a aussi des mots sur A^*)





Chemin

- soit un STE $L = (A,S,s_0,F,T)$,
  un chemin fini de L est une suite
  $s_0 a_1 s_1 ... s_{i-1} a_i s_i ... s_{n-1} a_n s_n$
  tel que:
  - $\forall i \in 0..n . s_i \in S$
  - $\forall i \in 0..n-1 . s_i –a_{i+1}–> s_{i+1}$

Trace

- idée : possibilité de faire une suite d'actions
- soit un STE $L = (A,S,s_0,F,T)$,

  une trace finie de L est un mot sur A^n

  $t=a_1...a_i...a_n$ (si n=0, la trace est notée ε)

  tel qu'il existe un chemin de L

  $s_0 a_1 s_1 ... s_{i-1} a_i s_i ... s_{n-1} a_n s_n$
- $s –t--> s'$ ssi $s\, a_1 s_1 ... s_{i-1} a_i s_i ... s_{n-1} a_n s'$

Trace observable

- idée : comportement externe visible d'un système
  ne garder que la partie ”visible” d'une trace
- une trace observable est un mot sur (A\{τ})*
- obs: A* ---> (A\{τ})* t.q
  - obs(ε)=ε
    obs(τ)=ε
    obs(a)=a (sinon)
    obs(a.t)=obs(a).obs(t)
- s=t=>s' ssi s–t'-->s' obs(t')=t

Trace (compléments)

- on parle aussi de chemins ou de traces infinies (A)
- l'ensemble des chemins d'un STE L est noté chemins(L) (paths(L) en anglais)
- l'ensemble des traces d'un STE L est noté traces(L)
- on parle aussi des traces d'un état s
  traces(s) = {t ∈ A* | s'∈ S : s–t–>s'}
  rem: traces(L) = traces(s_0)

Trace complète et dérivé

- $t=a_1a_2...a_n \in (A^* \quad A)$ est une trace complète ssi
  - soit $t \in A^n$ $s–t–>s'$ $s'–/–>$
  - soit $t \in A$
- dérivés d'un état s par une trace t ($\in A^*$)
  $deriv_t(s) = \{s' \in S \mid s–t–>s'\}$

Refus

- idée : actions qui ne peuvent pas être faites
- $B \subseteq (A)$ fini
  s refuse B ssi $\forall a \in B$. s--/a-->
- définition pour un STE $L=(A,S,s_0,F,T)$
  L refuse B ssi s_0 refuse B
- on peut prendre en compte que les actions observables : $B \subseteq (A \backslash \{\tau\})$
  s refuse B ssi $\forall a \in B$. s=/a=>

Convergence et divergence

- idée : détecter les suites infinies d'actions internes
- s diverge ($s\uparrow$) ssi $\exists\ s_0,s_1,\ldots\ .\ s=s_0 \wedge \forall\ i\geq 0\ .\ s_i \text{–}\tau\text{–>} s_{i+1}$
  s peut faire une suite infinie de τ
- s diverge sur t ($s\uparrow t$) ssi $\exists\ t_1,t_2\ .\ t=t_1t_2 \wedge s\text{–}t_1\text{–>}s' \wedge s'\uparrow$
  au cours de l'exécution de t depuis s, on peut diverger
- s converge sur t ($s\downarrow t$) ssi $\neg s\uparrow t$
- s convergent ($s\downarrow$) ssi $\forall\ t \in traces(s)\ .\ s\downarrow t$

Echec

- idée : possibilité de ne pas pouvoir faire certaines actions après une trace
- $B \subseteq A$ fini
  $t \in (A \setminus \{\tau\})^*$ séquence d'actions visibles
  <t, B> échec pour s ssi:
  - soit $s\uparrow t$ (on diverge avant d'y être ...)
  - soit $s \xrightarrow{t} s'$ et s' refuse B (refus)
- échecs(L) : l'ensemble des échecs de s_0
- échecs(s) : l'ensemble des échecs de s
- rem: on a aussi une déf. avec => au lieu de –>

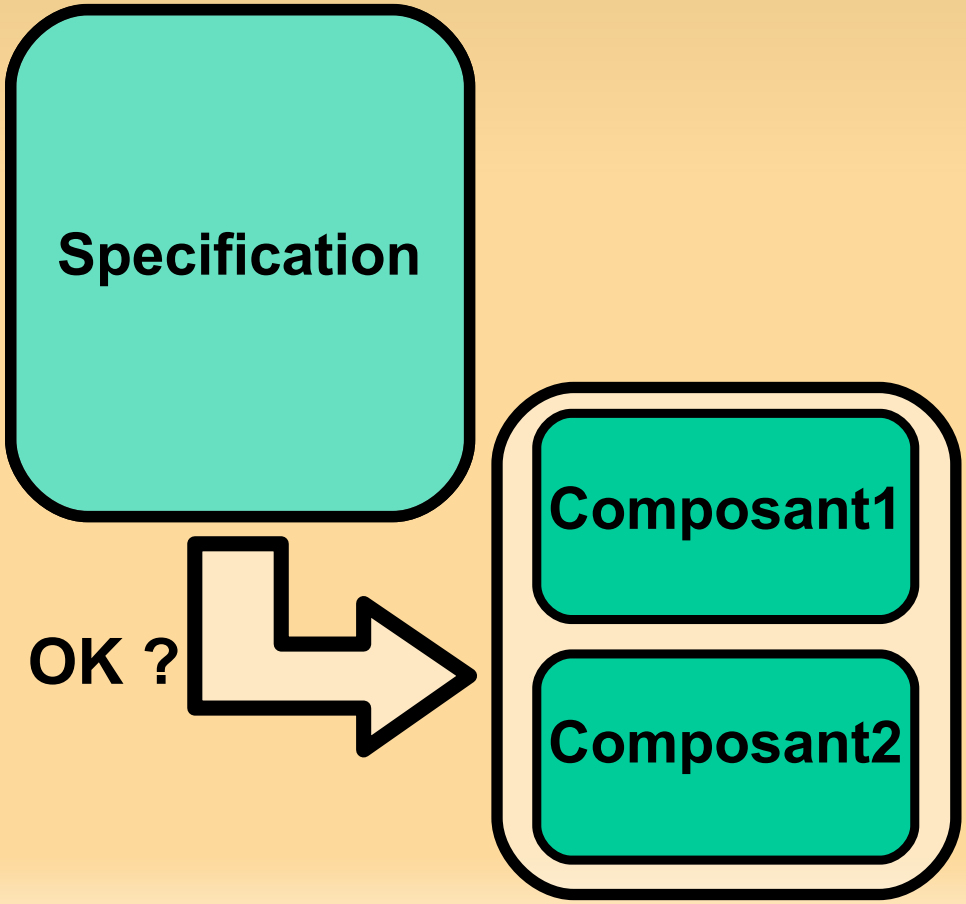
Le point et les objectifs

- le comportement d'un système simple (séquentiel) peut être modélisé par un STE
- le comportement d'un système composite (concurrent) peut être modélisé par une architecture et le produit de STE
- [INTRA] : traces, refus, échecs, ... d'un STE
- comment savoir si:
  - un système correspond à un besoin ?
  - un système peut en remplacer un autre ?

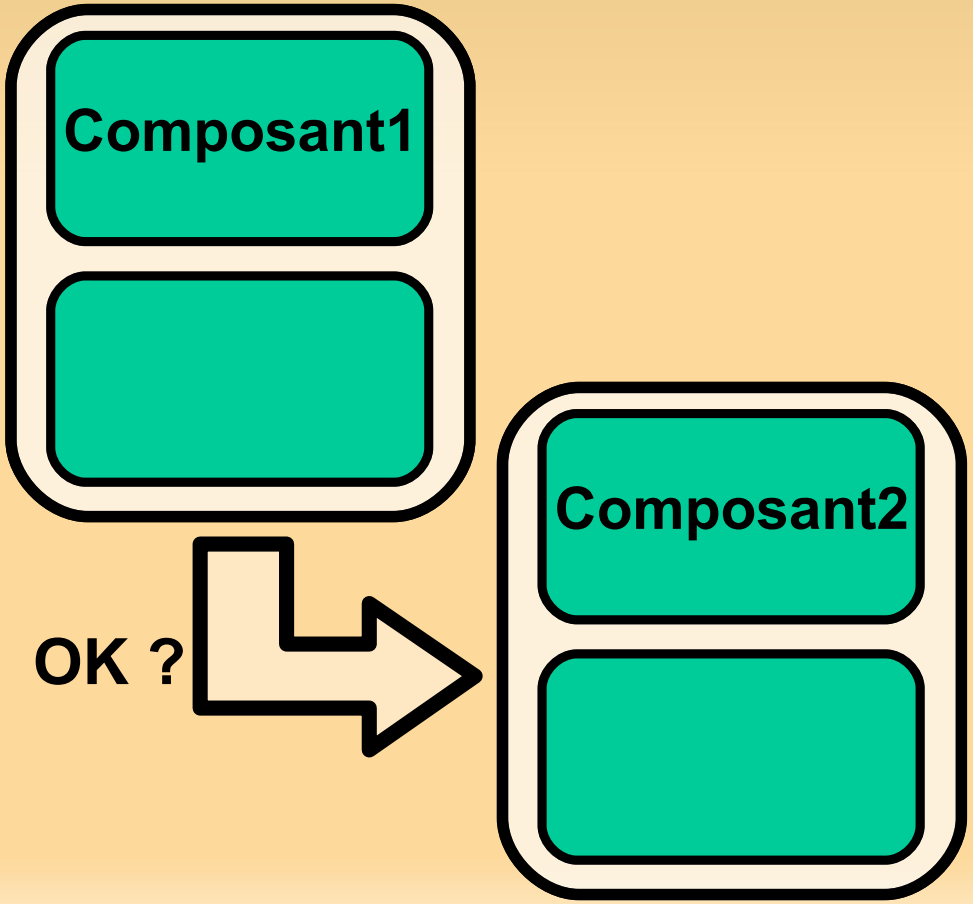
-> [INTER]

Idée générale

- conformité besoin/implantation



- substituabilité



Préordre

- soient E ensemble qque. et $R \subseteq E \times E$
  rem: on dit que R est une relation binaire sur E
- R est un préordre ssi :
  - reflexivité : $\forall e \in E \,.\, eRe$
  - transitivité : $\forall e \in E \, \forall e' \in E \, \forall e'' \in E \,.\, (eRe' \wedge e'Re'') \Rightarrow eRe''$
  - pour R préordre, $(e_1,e_2) \in R$ est aussi noté $e_1 \leq_R e_2$
- exemples :
  E:$\mathbb{N}$, R:$\leq$
  E:Nom$\times$Taille, R:{((n,t),(n',t')) | $t \leq t'$}

Equivalence

- soient E ensemble qque. et $R \subseteq E \times E$
- R est une relation d'équivalence ssi :
  - reflexivité : $\forall e \in E \,.\, eRe$
  - transitivité : $\forall e \in E \, \forall e' \in E \, \forall e'' \in E \,.\, (eRe' \wedge e'Re'') \Rightarrow eRe''$
  - symétrie : $\forall e \in E \, \forall e' \in E \,.\, (eRe' \Rightarrow e'Re)$
  - pour R équivalence, $(e_1,e_2) \in R$ est aussi noté $e_1 \simeq_R e_2$
- exemples :
  E:$\mathbb{N}$, R:=
  E:N°Etudiant$\times$Filière, R: {((n,f),(n',f')) | f=f'}

Finesse

- soient E ensemble qque., $R_1 \subseteq E \times E$ et $R_2 \subseteq E \times E$
- on dit que R_1 est plus fine que R_2 ssi $R_1 \subseteq R_2$
- on dit aussi que R_1 est plus restrictive que R_2
- exemple:
  E:$\mathbb{N}$, R1:=, R2:$\{(n_1,n_2) | n_1 \bmod 2 = n_2 \bmod 2\}$

Congruence

- soit E un ensemble quelconque E, R une relation binaire sur E et f une fonction d'arité n :
  - $f : E^n \rightarrow E$
- R est une congruence par rapport à f ssi :
  - $\forall e \in E \, \forall e' \in E \, . \, eRe' \Rightarrow f(...,e,...) \, R \, f(...,e',...)$
- exemple :
  E: $\mathbb{N}$, f: $+ : \mathbb{N}^2 \rightarrow \mathbb{N}$, R: $\leq$
  $\forall x \in \mathbb{N} \, \forall x' \in \mathbb{N} \, \forall y \in \mathbb{N} \, . \, (x \leq x') \Rightarrow (f(x,y) \leq f(x',y))$

Préordres et équivalences de STE

- dans la suite on va appliquer ces définitions à des systèmes décrits en termes de STE
- l'ensemble ”E” qque. sera celui des états, S
- on peut appliquer les définitions suivantes à deux STE $L_1=(A_1,S_1,s_{01},F_1,T_1)$ et $L_2=(A_2,S_2,s_{02},F_2,T_2)$
  - on suppose S tel que $(S_1 \cup S_2) \subseteq S$
  - pour un préordre $\preceq \subseteq S \times S$, $L_1 \preceq L_2$ ssi $s_{01} \preceq s_{02}$
    (idem pour les autres préordres et équivalences)

Préordre et équivalence de traces

- soit S un ensemble d'états
- préordre de traces $\leq_T \subseteq S \times S$:
  $\leq_T = \{(s,s') \mid s \in S \wedge s' \in S \wedge traces(s) \subseteq traces(s')\}$
- équivalence de traces $\simeq_T \subseteq S \times S$:
  $\simeq_T = \{(s,s') \mid s \in S \wedge s' \in S \wedge traces(s') = traces(s)\}$
- on a des variantes observables $\leq^{\Rightarrow}_T$ et $\simeq^{\Rightarrow}_T$
  en utilisant les traces observables

Préordre et équivalence d'échecs

- soit S un ensemble d'états
- préordre d'échecs $\leq_F \subseteq S \times S$: (échecs=Failures)

  $\leq_F = \{(s,s') \mid s \in S \wedge s' \in S \wedge traces(s) \subseteq traces(s') \wedge échecs(s') \subseteq échecs(s)\}$
- échecs(s') $\subseteq$ échecs(s) (comment le tester?)
  - $\forall$ (t,B') $\in$ échecs(s') : t $\in$ traces(s) implique $\exists$ (t,B) $\in$ échecs(s) : B' $\subseteq$ B
- équivalence d'échecs $\simeq_F \subseteq S \times S$:

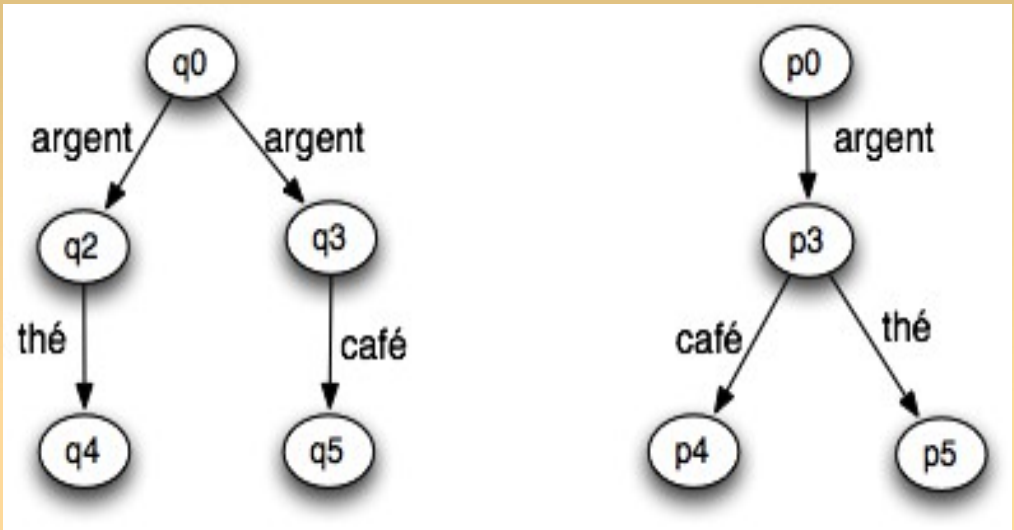
  $\simeq_F = \{(s,s') \mid (s,s') \in \leq_F \wedge (s',s) \in \leq_F\}$
- on a des variantes observables $\leq^{\Rightarrow}_F$ et $\simeq^{\Rightarrow}_F$

Préordres de traces vs. D'échecs et substituabilité

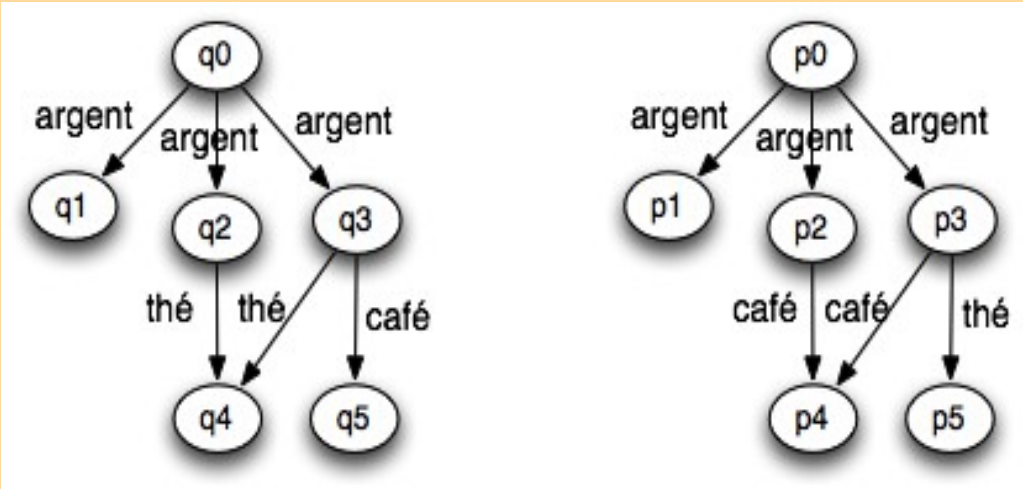
- $s \leq_T s'$
  s a moins d'exécutions possibles que s'
- Exemple au tableau
- $s \leq_F s'$
  pour les traces de s,
  s a plus de possibilités d'échecs que s'

Vers de nouvelles équivalences ...

- soient les 2 machines suivantes :



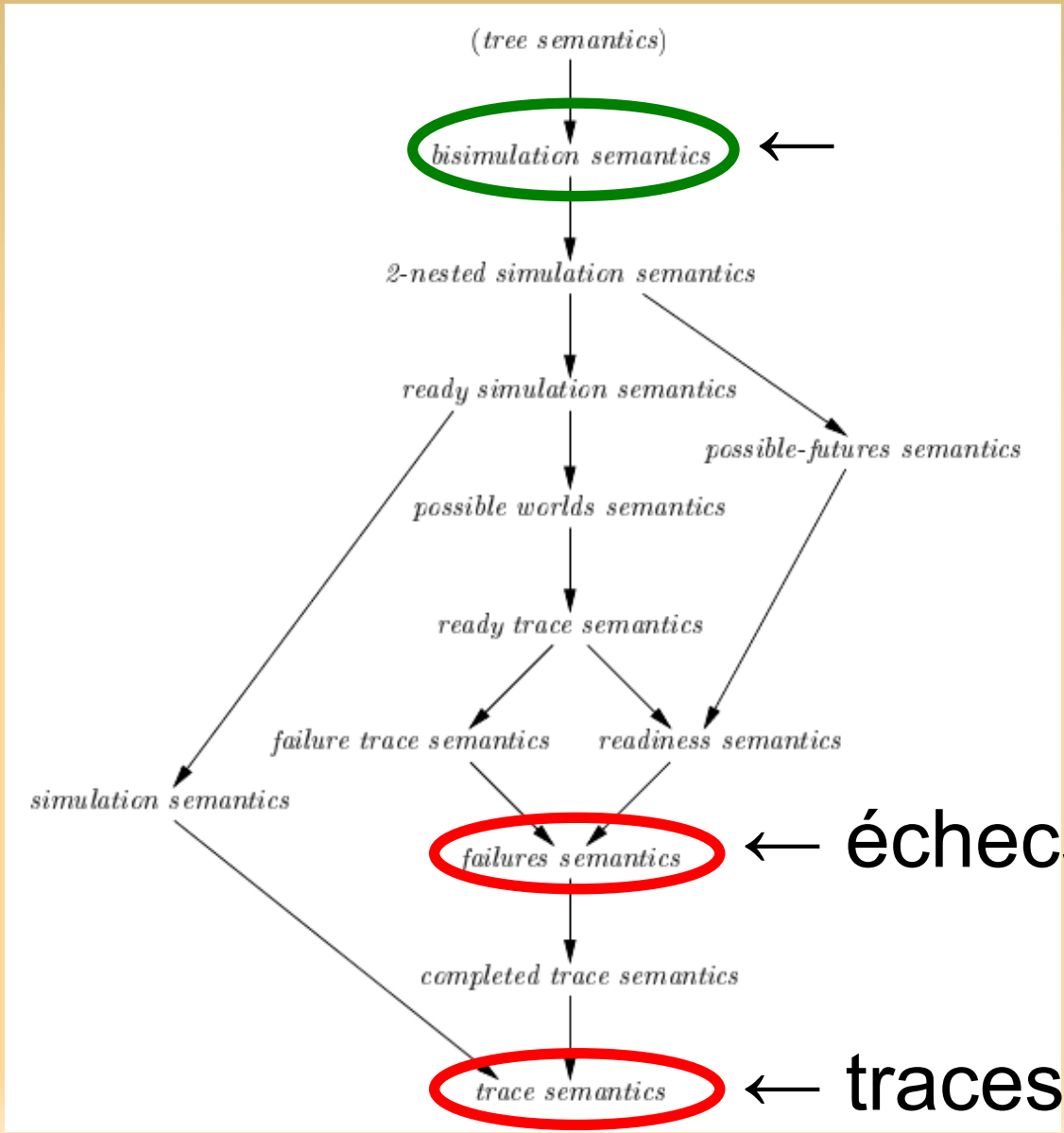
- mêmes traces : $p_0 \simeq_T q_0$
- $q_0 \leq_F p_0$



- mêmes traces : $p_0 \simeq_T q_0$
- mêmes échecs : $p_0 \simeq_F q_0$
- => Et pourtant ils sont différents

Vers de nouvelles équivalences ...

- il existe de nombreuses équivalences



← échecs

← traces

- on va en voir une qui est plus fine
- elle supporte bien les notions:
  - d'indeterminisme
  - d'interaction
  - d'état

R. J. Van Glabeek, The Linear Time – Branching Time Spectrum I, chapitre 1, Hanbook of Process Algebra, Elsevier, 2001.

Bissimulation

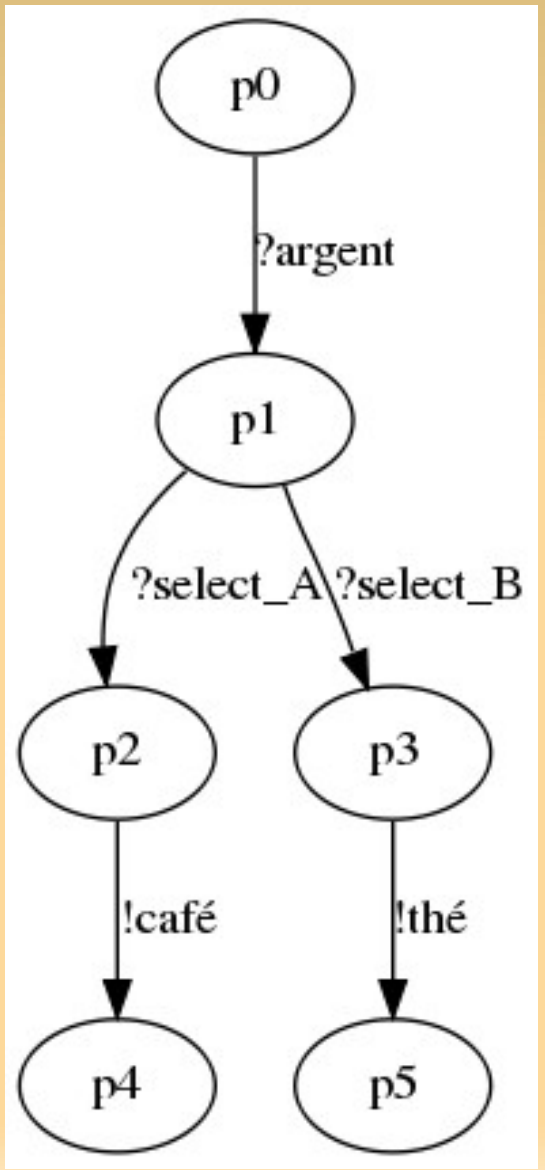
- une relation de bissimulation (forte) est une relation R symétrique telle que, pour tout $s_1 R\ s_2$ on a :
  - $\forall\ a \in A_1, s'_1 \in S_1 : s_1 \text{–a-->} s'_1$ implique $\exists s'_2 \in S_2 : s_2 \text{–a-->} s'_2 \wedge s'_1 R\ s'_2$

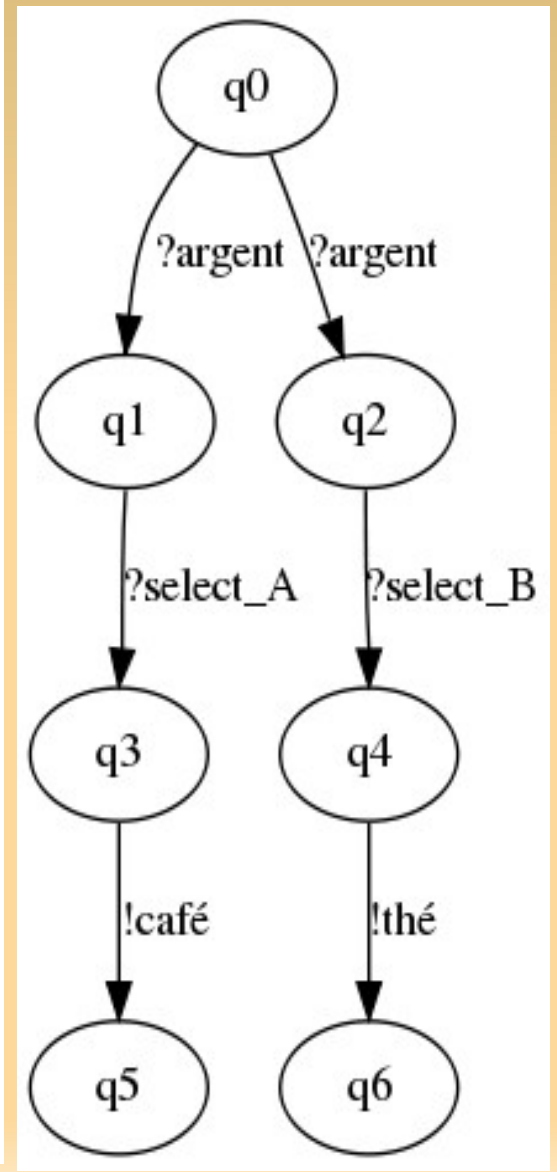
  $\wedge$

  - $\forall\ a \in A_2, s'_2 \in S_2 : s_2 \text{–a-->} s'_2$ implique $\exists s'_1 \in S_1 : s_1 \text{–a-->} s'_1 \wedge s'_2 R\ s'_1$
- ~ est l'union de toutes les relations de bissimulation $s_1 \sim s_2$ se lit s_1 et s_2 sont (fortement) bissimilaires
- $L_1=(A_1,S_1,s_{01},F_1,T_1) \sim L_2=(A_2,S_2,s_{02},F_2,T_2)$ ssi $s_{01} \sim s_{02}$

Retour sur nos 2 machines

- soient les 2 machines suivantes :

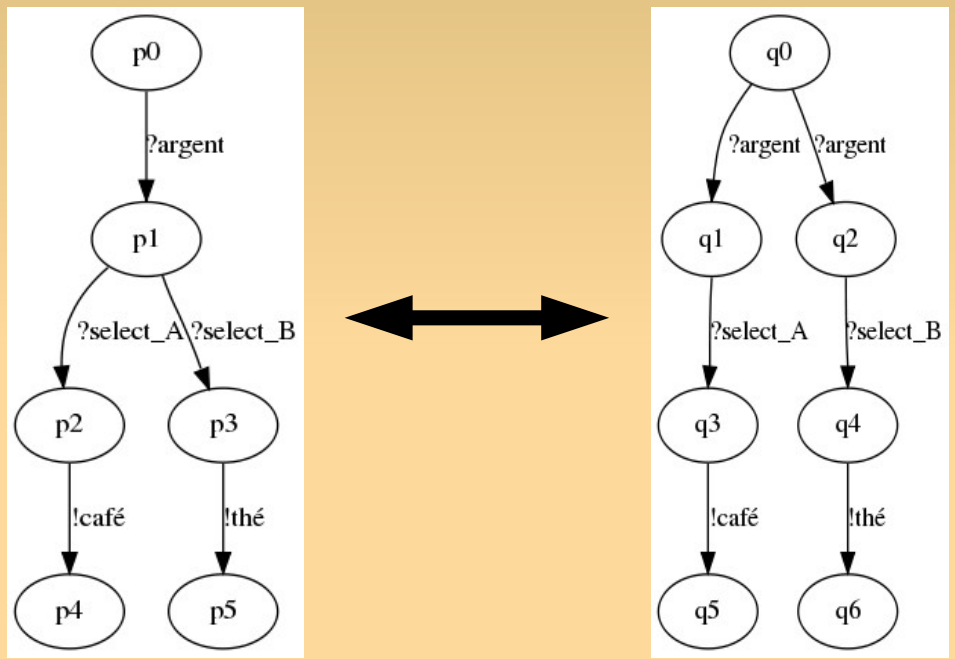




- mêmes traces : $p_0 \simeq_T q_0$
- pourtant différentes, e.g. tester avec test T: !argent.!select_A.?café.✓
  (P|T)\{...} →* ✓ toujours
  (Q|T)\{...} →* ✓ pas toujours
- $q_0 \subseteq_F p_0$
- $q_0 \sim p_0$?

Bissimulation HOWTO

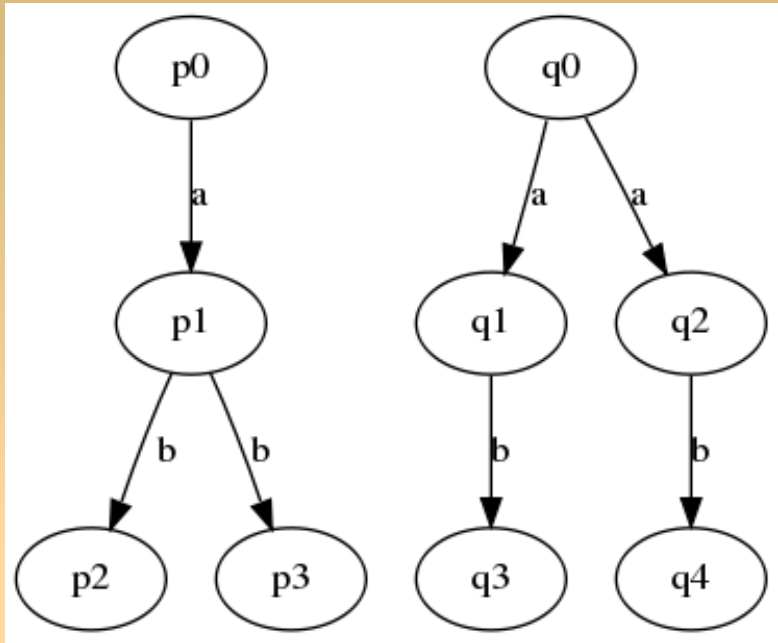
- pour montrer que 2 états sont bissimilaires:
  → on construit une relation de bissimulation les contenant



- pour montrer que 2 états ne sont pas bissimilaires:
  → on trouve un contre-exemple / on prouve (par l'absurde) qu'aucune relation de bissimulation ne peut les contenir

- ltscompare bisim s s' (*mcrl2*)

Bissimulation / exemple



questions/faits:

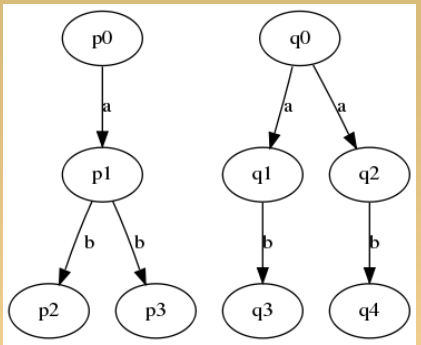
- $p0 \sim^{?} q0$

hypothèses:

- $p1 \sim q1 \vee p1 \sim q2$
- $q1 \sim p1 \wedge q2 \sim p1$

Ne pas en oublier, rappel: une bissimulation est symétrique !

Bissimulation / exemple



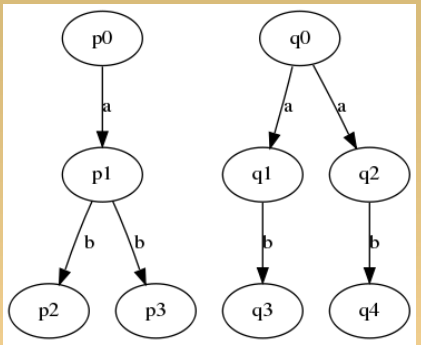
questions/faits:

- $p0 \sim^{?} q0$
- $q1 \sim^{?} p1$

hypothèses:

- $p1 \sim q1 \vee p1 \sim q2$
- $q1 \sim p1 \wedge q2 \sim p1$
- $p2 \sim q3$
- $p3 \sim q3$
- $q3 \sim p2 \vee q3 \sim p3$

Bissimulation / exemple



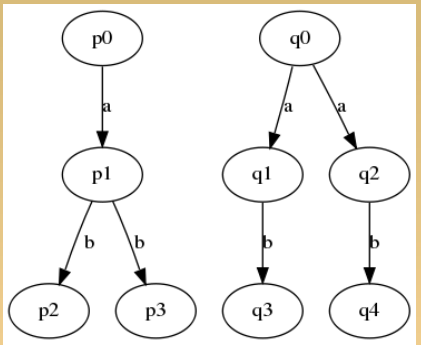
questions/faits:

- p0~?q0
- q1~p1
- p2~q3 ✓
- p3~q3 ✓
- q3~p2 /\ q3~p3 ✓

hypothèses:

- p1~q1 \/ p1~q2
- q1~p1 /\ q2~p1
- p2~q3
- p3~q3
- q3~p2 \/ q3~p3

Bissimulation / exemple



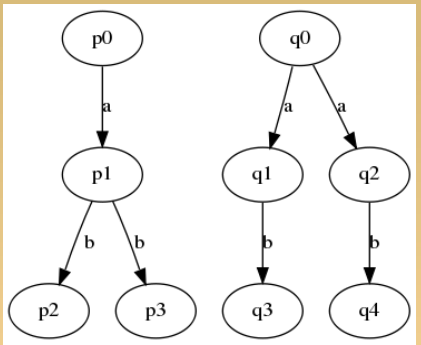
questions/faits:

- p0~?q0
- q1~p1 ✓
- p2~q3
- p3~q3
- q3~p2 /\ q3~p3

hypothèses:

- p1~q1 \/ p1~q2
- q1~p1 /\ q2~p1
- p2~q3 ✓
- p3~q3
- q3~p2 \/ q3~p3

Bissimulation / exemple



questions/faits:

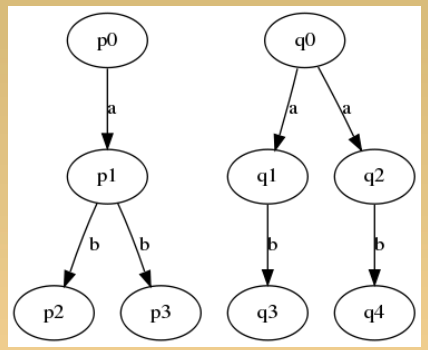
- $p0 \sim^{?} q0$
- $q1 \sim p1 \wedge q2 \sim p1$ ✓
- $p2 \sim q3 \wedge p3 \sim q3 \wedge q3 \sim p2 \wedge q3 \sim p3$
- $q4 \sim p2 \wedge q4 \sim p3 \wedge p2 \sim q4 \wedge p3 \sim q4$

hypothèses:

- $p1 \sim q1 \vee p1 \sim q2$
- $q1 \sim p1 \wedge q2 \sim p1$
- $q4 \sim p2 \vee q4 \sim p3$ ✓
- $p2 \sim q4$
- $p3 \sim q4$

de même, on montre $q2 \sim p1$

Bissimulation / exemple



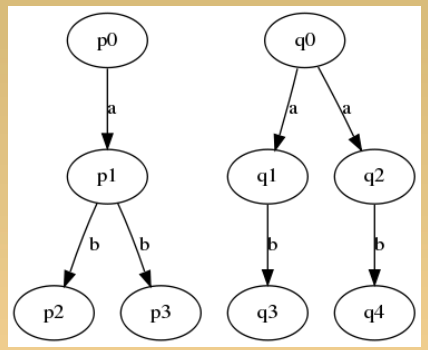
questions/faits:

- $p0 \sim^{?} q0$
- $q1 \sim p1 \wedge q2 \sim p1$
- $p2 \sim q3 \wedge p3 \sim q3 \wedge q3 \sim p2 \wedge q3 \sim p3$
- $q4 \sim p2 \wedge q4 \sim p3 \wedge p2 \sim q4 \wedge p3 \sim q4$

hypothèses:

- $p1 \sim q1 \vee p1 \sim q2$
- $q1 \sim p1 \wedge q2 \sim p1$ ✓

Bissimulation / exemple



questions/faits:

- $p0 \sim^{?} q0$
- $q1 \sim p1 \wedge q2 \sim p1$
- $p2 \sim q3 \wedge p3 \sim q3 \wedge q3 \sim p2 \wedge q3 \sim p3$
- $q4 \sim p2 \wedge q4 \sim p3 \wedge p2 \sim q4 \wedge p3 \sim q4$

si on montre

$p1 \sim q1 \vee p1 \sim q2$

de la même façon

ou via des faits prouvés

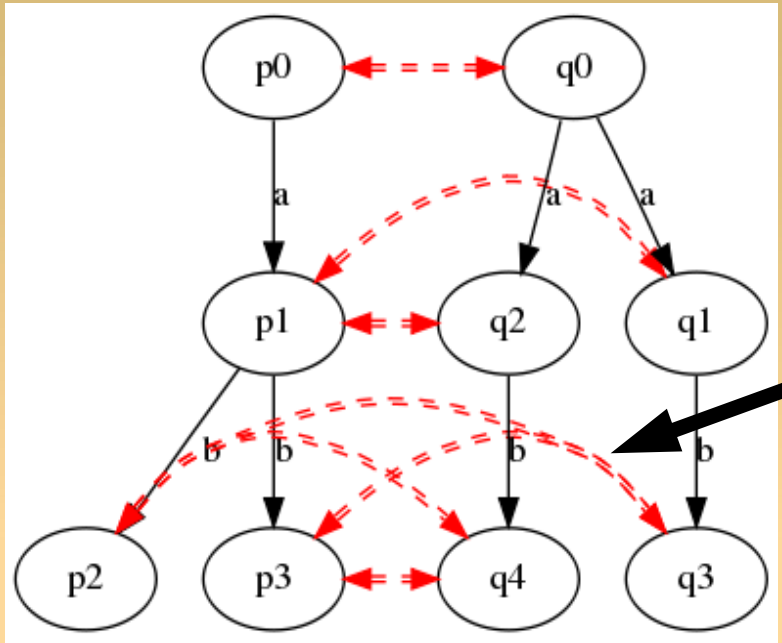
correctement + symétrie ~

(ici $q1 \sim p1 \wedge q2 \sim p1$)

alors, hypothèses:

- vide et pas eu de contradictions
  $\rightarrow p0 \sim q0$

Bissimulation / exemple



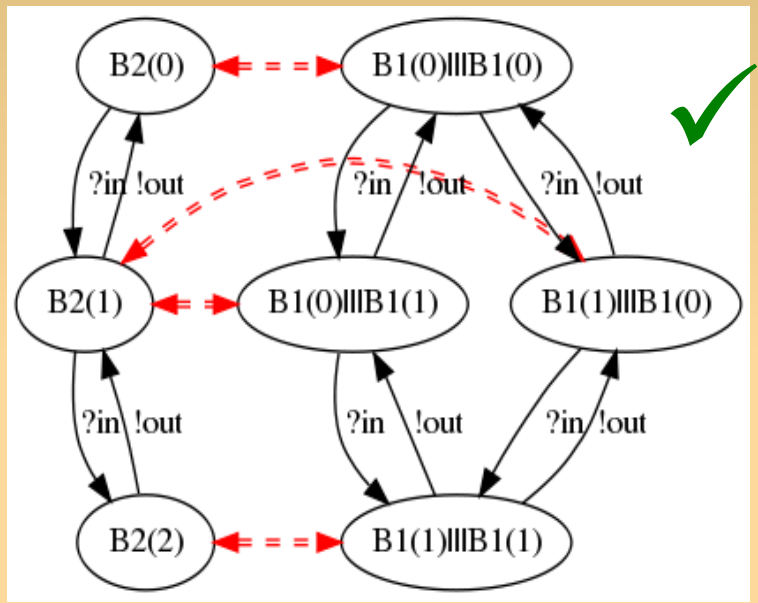
on a:
q1~p1/\ q2~p1/\ p2~q3 /\ p3~q3 /\ q3~p2 /\ q3~p3 /\ q4~p2 /\ q4~p3 /\ p2~q4 /\ p3~q4 /\ p1~q1 /\ p1~q2 /\ p0~q0 /\ q0~p0

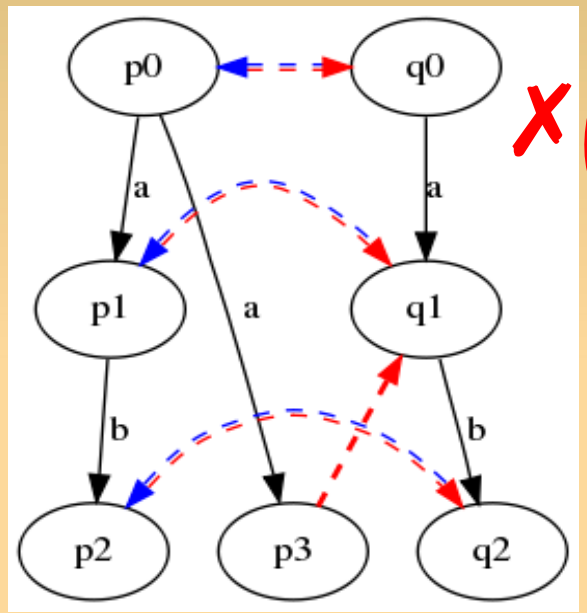
soit la relation symétrique: $R = R_{pq} \cup R_{pq}^{-1}$

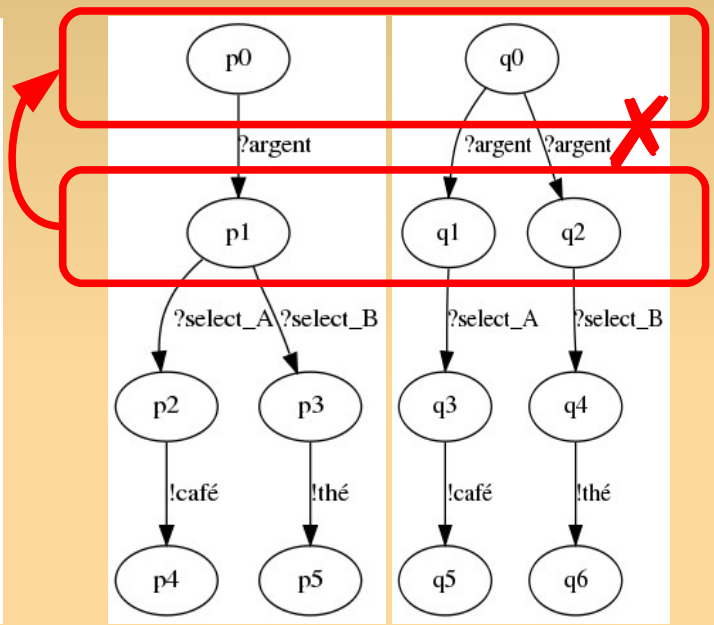
avec R_{pq} = {(p0,q0),(p1,q1),(p1,q2),(p2,q3),(p2,q4),(p3,q3),(p3,q4)}

Bissimulation HOWTO + exemples

- pour montrer que 2 états sont bissimilaires:
  ⟶ on construit une relation de bissimulation les contenant







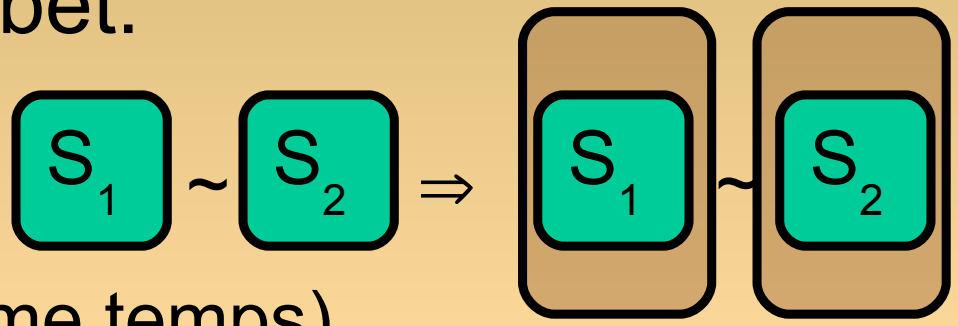
- pour montrer que 2 états ne sont pas bissimilaires:
  ⟶ on trouve un contre-exemple / on prouve (par l'absurde) qu'aucune relation de bissimulation ne peut les contenir

- ltscompare -c -e bisim s s'

Où l'on voit que la bissimulation forte est ... ... trop forte

- la bissimulation permet de comparer deux systèmes de même alphabet:

✓ pour la substituabilité



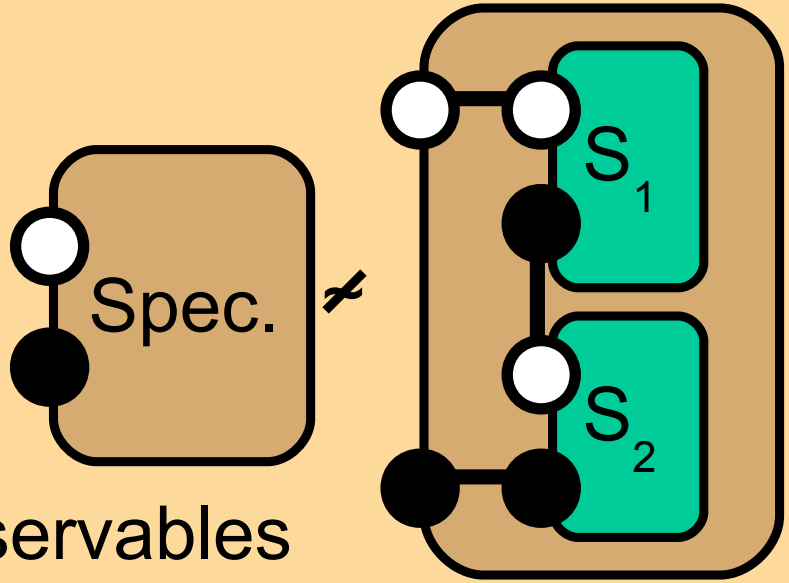
(si aucun raffinement en même temps)

✗ pour le raffinement

car il faut abstraire

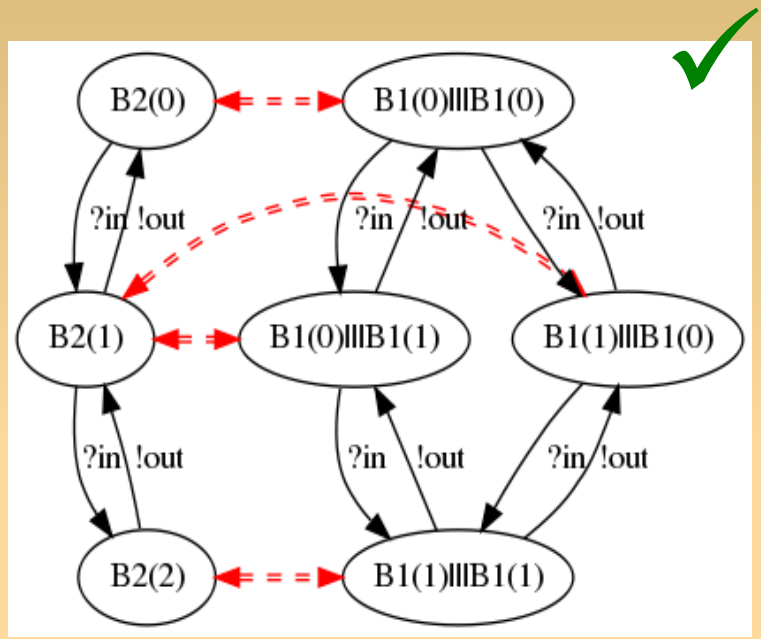
- des choix
- des communications

internes à l'implantation et non observables

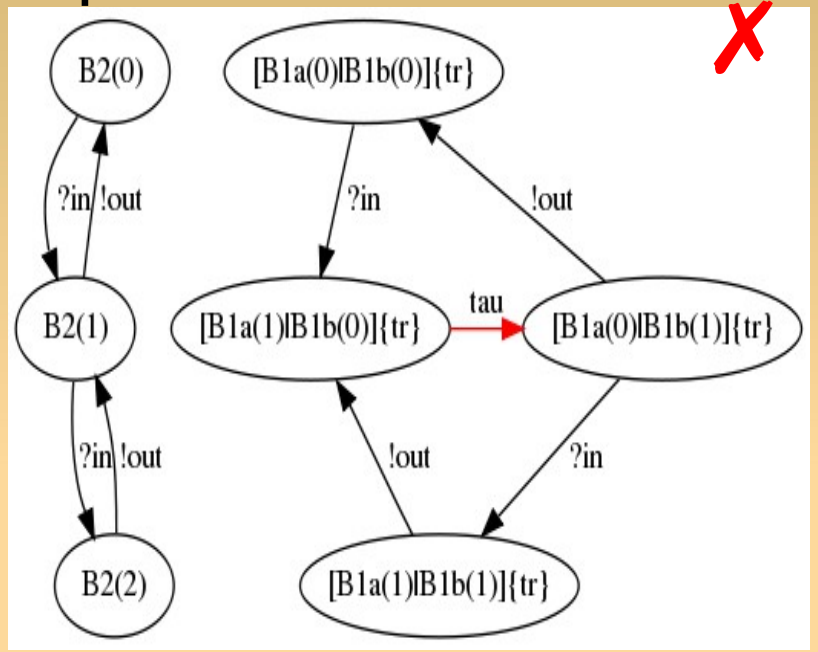


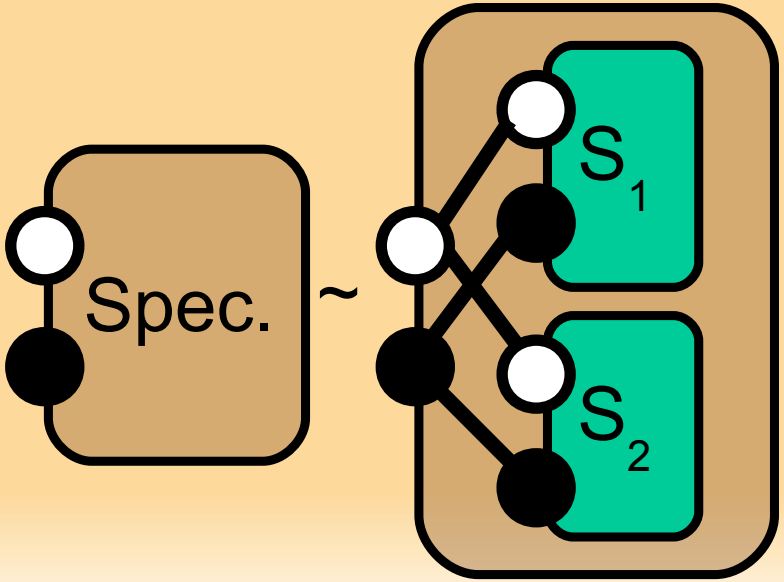
Où l'on voit que la bissimulation forte est ...
... trop forte

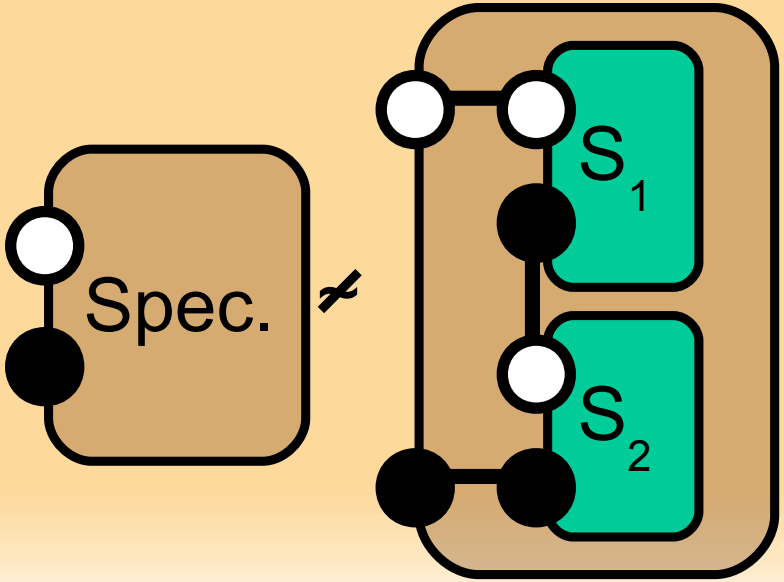
- bi-pile non communicante



- bi-pile communicante

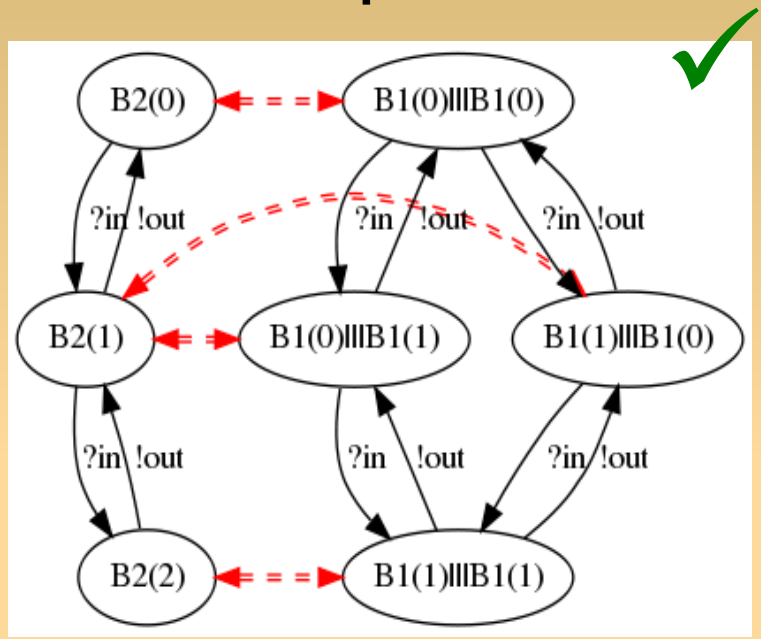


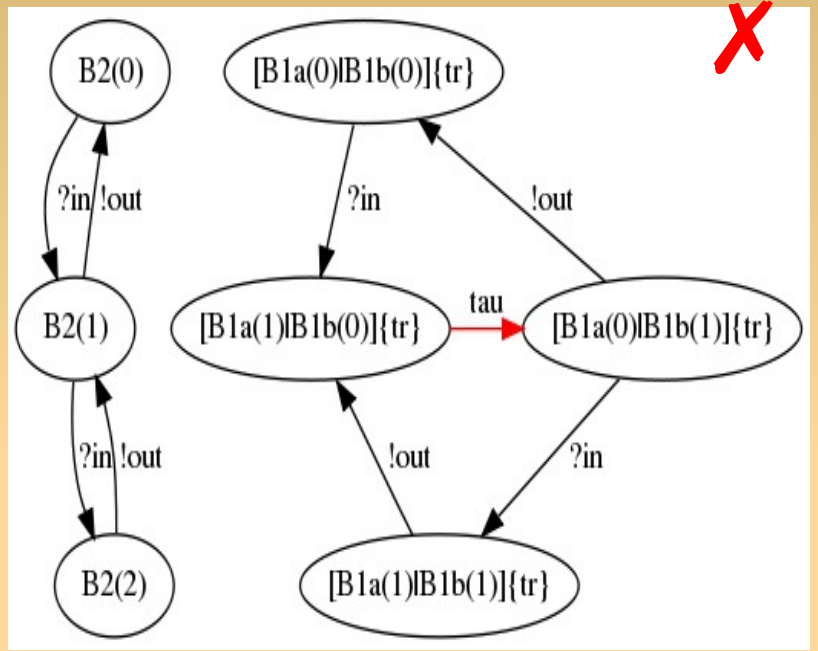




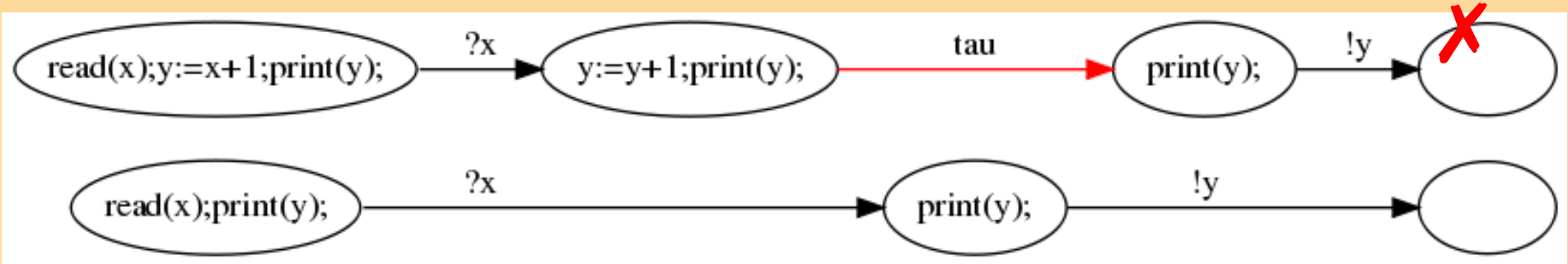
Où l'on voit que la bissimulation forte est ... ... trop forte

- ~ est trop sensible





aux communications internes



aux calculs internes

Bissimulation faible

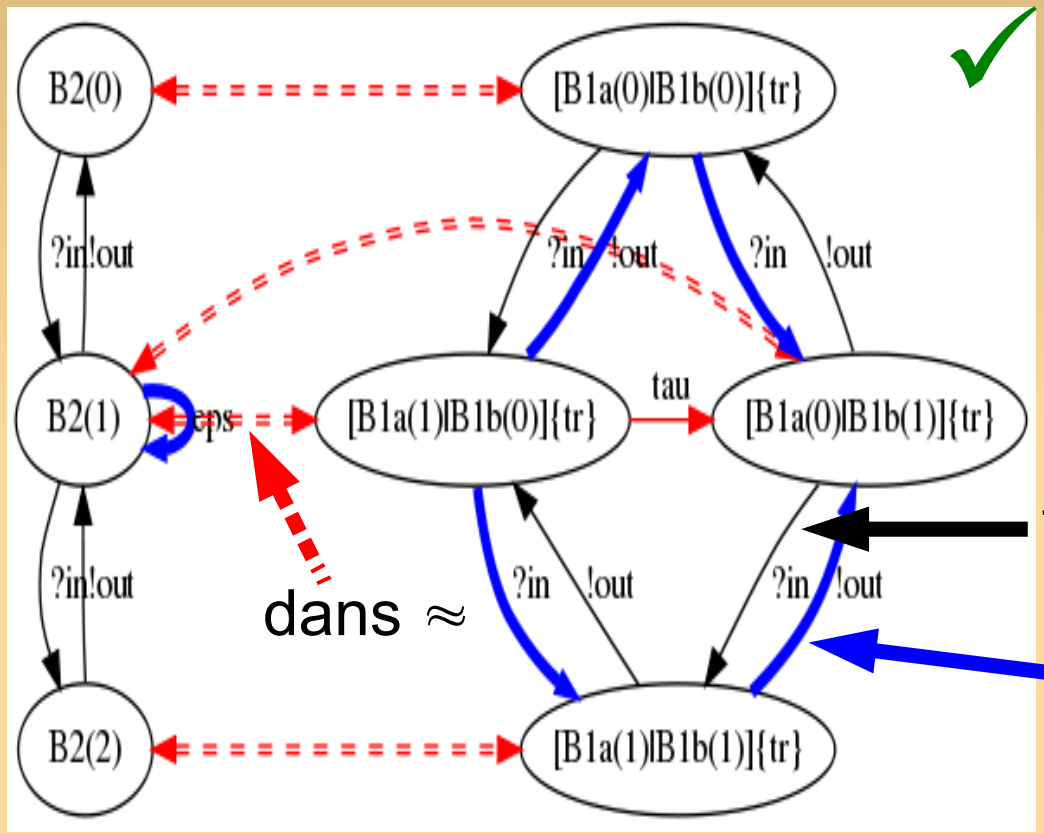
- une relation de bissimulation faible est une relation R symétrique telle que, pour tout $s_1 R s_2$:
  - $\forall a \in A_1, s'_1 \in S_1 : s_1 \text{–}a\text{-->} s'_1$ implique $\exists s'_2 \in S_2$: $s_2 =obs(a)=> s'_2 \wedge s'_1 R s'_2$
  - $\wedge$
  - $\forall a \in A_2, s'_2 \in S_2 : s_2 \text{–}a\text{-->} s'_2$ implique $\exists s'_1 \in S_1$: $s_1 =obs(a)=> s'_1 \wedge s'_2 R s'_1$

$obs(a)=\varepsilon$ si $a=\tau$, $obs(a)=a$ sinon

$p=\varepsilon=>q$ ssi $p\text{-}\tau\text{->}\ldots\text{-}\tau\text{->}q$, $p=a=>q$ ssi $p\text{-}\tau\text{->}\ldots\text{-}a\text{->}\ldots\text{-}\tau\text{->}q$

- $\approx$ est l'union de toutes les relations de biss. faibles $s_1 \approx s_2$ se lit s_1 et s_2 sont faiblement bissimilaires

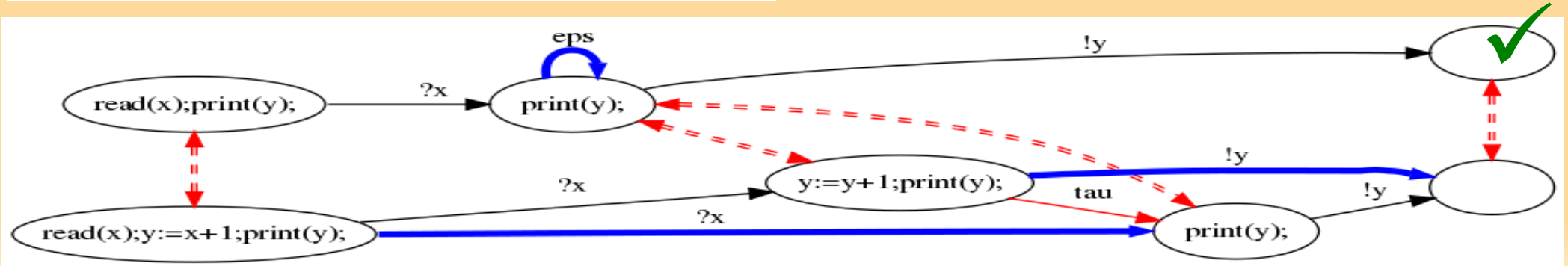
Retour sur nos exemples



- mêmes traces observ.
- τ mais ≈
- HOWTO ≈: similaire à ~

transition -->

transition ==>



Bissimulation branchante

- Alternative à la bissimulation faible
- Nous la verrons en TD avec *mcrl2*

  ltscompare -ebranching-bisim s s' *(mcrl2)*