

# Validation d'algorithmes

Chapitre V

Tests et abstraction : Ma-  
chines de Mealy



Chapitre V

## Machines de Mealy

# Abstraire

```
...  
this.$target = $(this.$target || $(this.$target[0]).is('div'))[0];  
this.$target.removeClass('active');  
this.$target.removeClass('highlight');  
this.$target.removeClass('highlighted');  
this.$target.removeClass('highlighted');
```

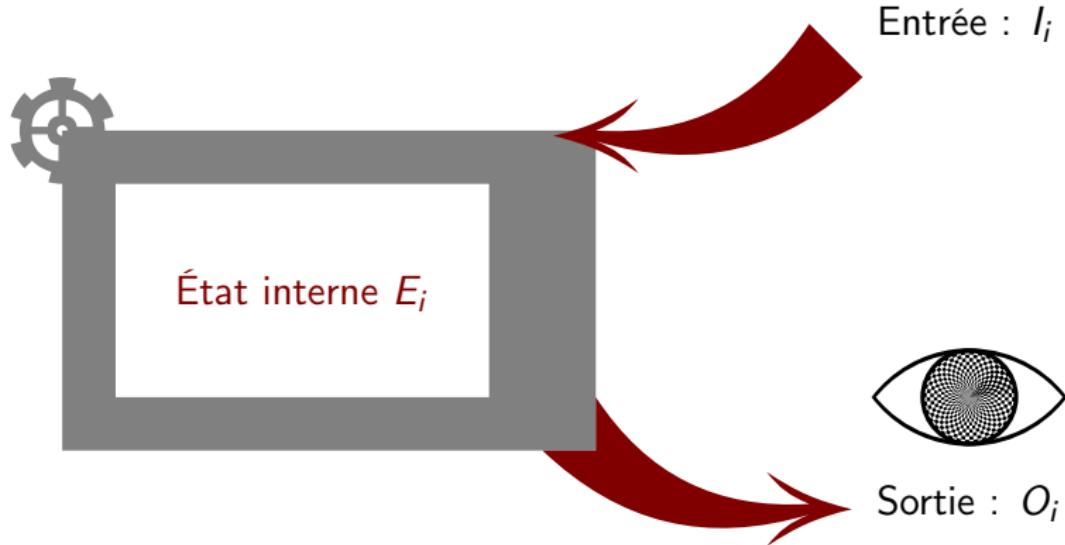
Code réel  
Complexe à traiter

Automate  
de Mealy

Abstraction  
Plus facile à étudier

➡ On teste le code réel en considérant un modèle plus haut niveau.

## Définition



Informellement :



## Concept

Une machine de Mealy est un automate auquel on rajoute des sorties.

Ici, on restera en boîte noire. C'est-à-dire que l'on ne connaît pas l'implémentation a priori de l'automate.

De plus, l'état initial n'est pas connu, et c'est tout le problème.



Une machine de Mealy est un quadruplet  $(\mathcal{S}, \mathcal{I}, \mathcal{O}, \delta)$

- ⊗  $\mathcal{S}$  est un ensemble fini représentant les états possibles du système.
- ⊗  $\mathcal{I}$  est un ensemble fini d'actions d'entrées.
- ⊗  $\mathcal{O}$  est un ensemble fini d'actions de sorties.

Une machine de Mealy est un quintuplet  $(\mathcal{S}, \mathcal{I}, \mathcal{O}, \delta, \lambda)$

- ⊗  $\mathcal{S}$  est un ensemble fini représentant les états possibles du système.
- ⊗  $\mathcal{I}$  est un ensemble fini d'actions d'entrées.
- ⊗  $\mathcal{O}$  est un ensemble fini d'actions de sorties.
- ⊗  $\delta : (\mathcal{S} \times \mathcal{I}) \longrightarrow \mathcal{S}$  est la fonction de transition, et  
 $\lambda : (\mathcal{S} \times \mathcal{I}) \longrightarrow \mathcal{O}$  est la fonction de sortie.



Une machine de Mealy est un quintuplet  $(\mathcal{S}, \mathcal{I}, \mathcal{O}, \delta, \lambda)$

- ⊗  $\mathcal{S}$  est un ensemble fini représentant les états possibles du système.
- ⊗  $\mathcal{I}$  est un ensemble fini d'actions d'entrées.
- ⊗  $\mathcal{O}$  est un ensemble fini d'actions de sorties.
- ⊗  $\delta : (\mathcal{S} \times \mathcal{I}) \longrightarrow \mathcal{S}$  est la fonction de transition, et  
 $\lambda : (\mathcal{S} \times \mathcal{I}) \longrightarrow \mathcal{O}$  est la fonction de sortie.
- ⊗  $\delta$  et  $\lambda$  sont des fonctions totales

## Notations

Une machine de Mealy  $(\mathcal{S}, \mathcal{I}, \mathcal{O}, \delta, \lambda)$

On note  $\epsilon$  le mot vide.

On étend les fonctions de transition et de sortie aux mots (séquences) sur les états.

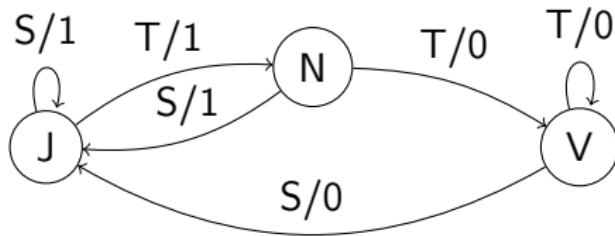
- ⊗  $\forall s \in \mathcal{S}, \delta(s, \epsilon) = s$
- ⊗  $\forall s \in \mathcal{S}, \lambda(s, \epsilon) = \epsilon$
- ⊗  $\forall s \in \mathcal{S}, \forall w \in \mathcal{I}^*, \forall i \in \mathcal{I}, \delta(s, wi) = \delta(\delta(s, w), i)$
- ⊗  $\forall s \in \mathcal{S}, \forall w \in \mathcal{I}^*, \forall i \in \mathcal{I}, \lambda(s, wi) = \lambda(s, w)\lambda(\delta(s, w), i)$



## Exemple

Abstraction de la machine à café intelligente et très ennuyeuse.

- ⊗  $S = \{J=\text{Joyeuse}, N=\text{Neutre}, V=\text{Vexée}\}$
- ⊗  $I = \{S=\text{Sourire}, T=\text{Faire la tête}\}$
- ⊗  $O = \{0=\text{Ne rien faire}, 1=\text{Servir un café}\}$



$$\lambda(J, TT) = 10$$

$$\lambda(N, STSS) = 1111$$

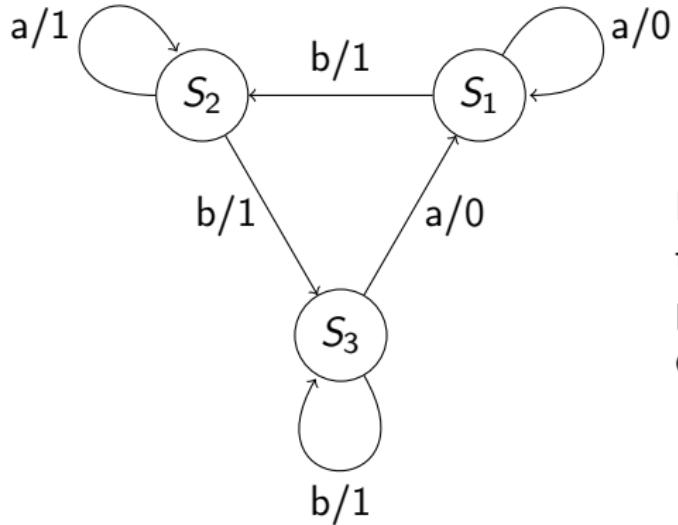
$$\lambda(V, TST) = 001$$

## Test de conformité



➡ Le modèle est-il conforme à l'implémentation ?

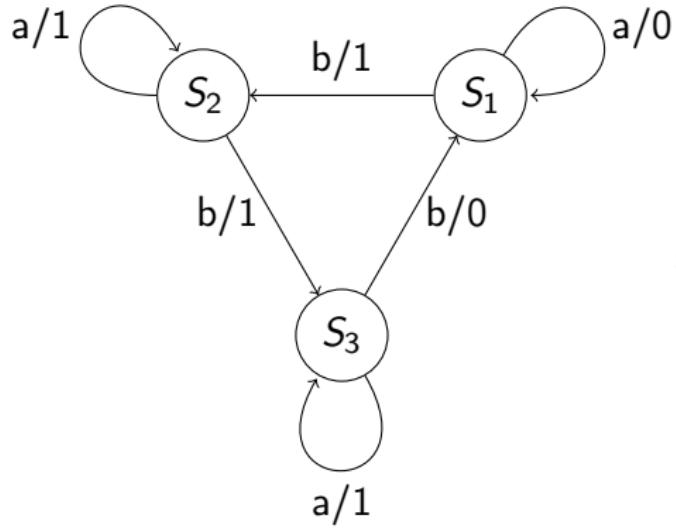
Dans nos abstractions, nous utiliserons les lettres pour les entrées et les chiffres pour la sortie



Ne connaissant pas l'état initial, quelle séquence donner pour être sûr de se retrouver dans l'état  $S_1$  ?



Dans nos abstractions, nous utiliserons les lettres pour les entrées et les chiffres pour la sortie

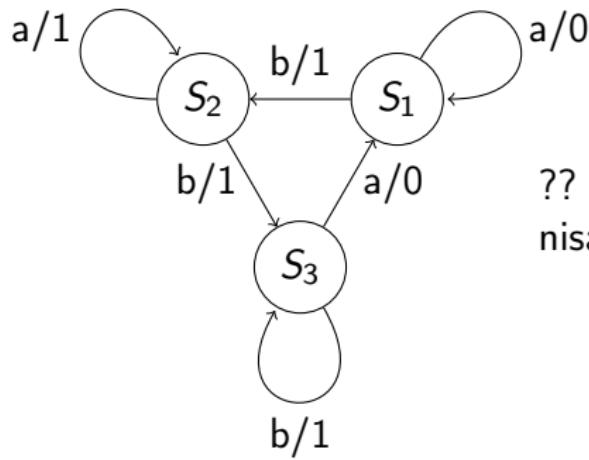


Et maintenant, comment être sûr de se retrouver dans l'état  $S_1$  ?



## Synchronizing sequence

$x \in \mathcal{I}^*$  est une séquence de synchronisation ssi  
 $|\delta(\mathcal{S}, x)| = 1$

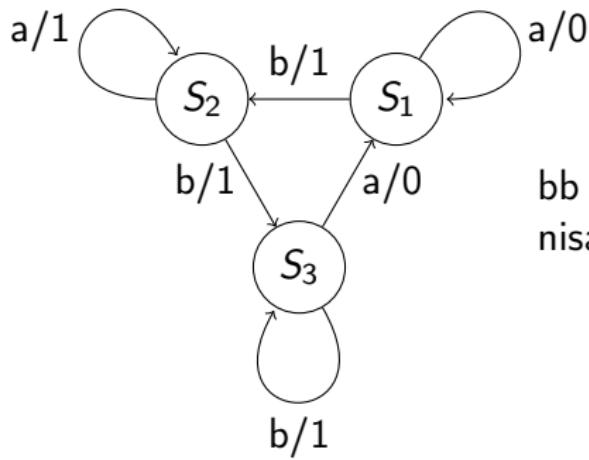


?? est une séquence synchronisante car  $\delta(\mathcal{S}, bb) = S_3$



## Synchronizing sequence

$x \in \mathcal{I}^*$  est une séquence de synchronisation ssi  
 $|\delta(\mathcal{S}, x)| = 1$



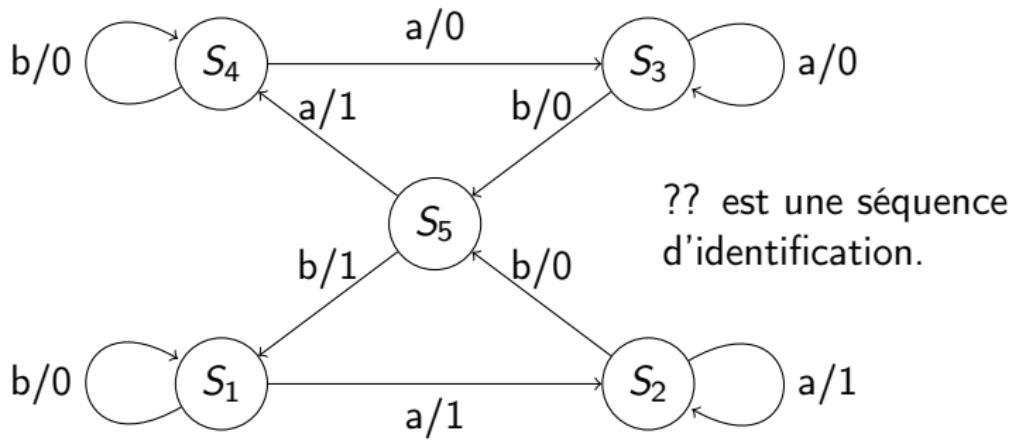
bb est une séquence synchronisante car  $\delta(\mathcal{S}, bb) = S_3$



## Homing sequence

$x \in \mathcal{I}^*$  est une séquence d'identificationssi

$\forall s, t \in \mathcal{S}, \delta(s, x) \neq \delta(t, x) \implies \lambda(s, x) \neq \lambda(t, x)$

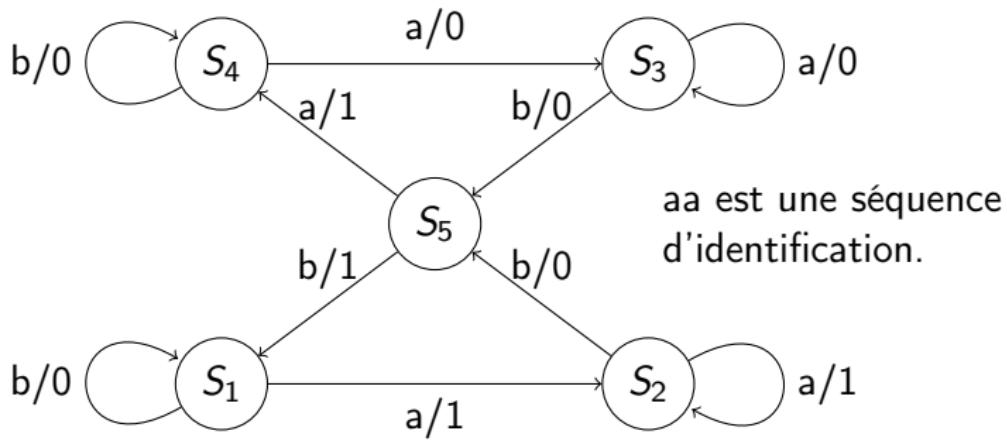




## Homing sequence

$x \in \mathcal{I}^*$  est une séquence d'identificationssi

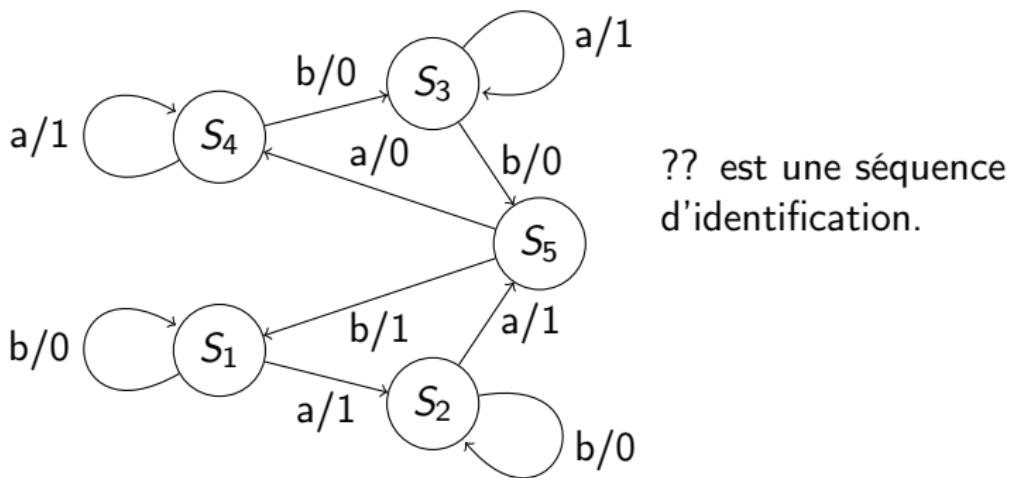
$\forall s, t \in \mathcal{S}, \delta(s, x) \neq \delta(t, x) \implies \lambda(s, x) \neq \lambda(t, x)$





## Distinguishing sequence

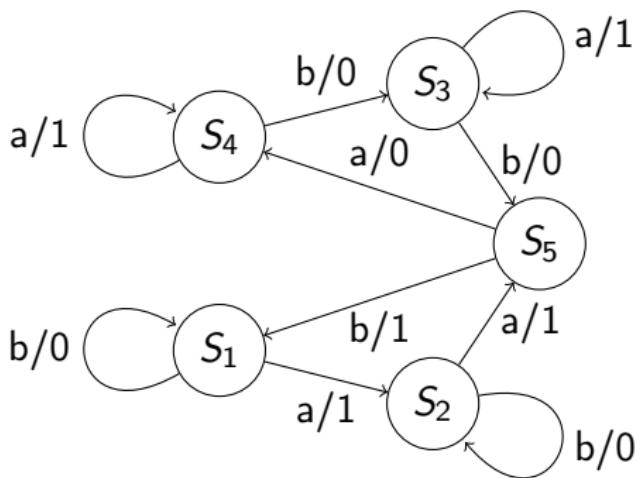
$x \in \mathcal{I}^*$  est une séquence discriminante ssi  
 $\forall s, t \in \mathcal{S}, s \neq t \implies \lambda(s, x) \neq \lambda(t, x)$





## Distinguishing sequence

$x \in \mathcal{I}^*$  est une séquence discriminante ssi  
 $\forall s, t \in \mathcal{S}, s \neq t \implies \lambda(s, x) \neq \lambda(t, x)$



ababa est une séquence d'identification.



Soit  $x \in \mathcal{I}^*$  une séquence, quelles sont les implications ?

$x$  est une séquence discriminante

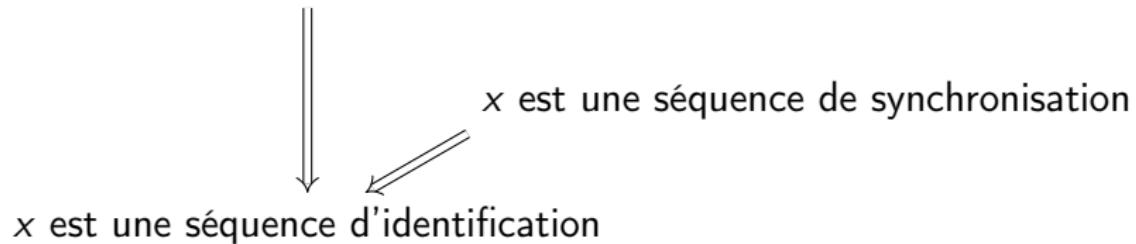
$x$  est une séquence de synchronisation

$x$  est une séquence d'identification



Soit  $x \in \mathcal{I}^*$  une séquence, quelles sont les implications ?

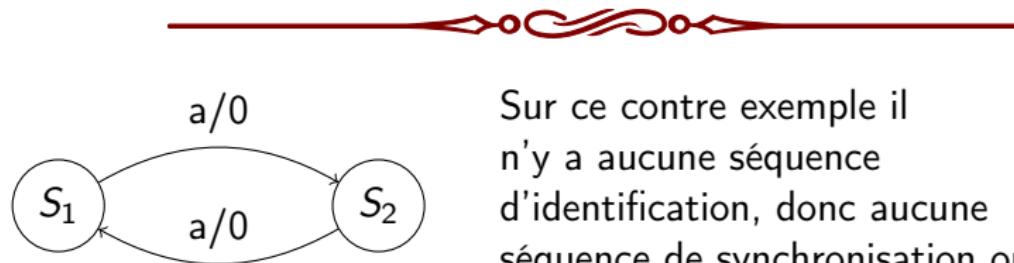
$x$  est une séquence discriminante



➡ Peut-on toujours trouver une séquence de synchronisation, une séquence d'identification et/ou une séquence discriminante ?



- ☞ Peut-on toujours trouver une séquence de synchronisation, une séquence d'identification et/ou une séquence discriminante ?



Sur ce contre exemple il n'y a aucune séquence d'identification, donc aucune séquence de synchronisation ou discrimination





## Automates équivalents

Deux automates  $(\mathcal{S}_1, \mathcal{I}, \mathcal{O}, \delta_1, \lambda_1)$  et  $(\mathcal{S}_2, \mathcal{I}, \mathcal{O}, \delta_2, \lambda_2)$  sont équivalents ssi

$\forall s_1 \in \mathcal{S}_1, \exists s_2 \in \mathcal{S}_2, \forall x \in \mathcal{I}^* \lambda_1(s_1, x) = \lambda_2(s_2, x)$  et  
 $\forall s_2 \in \mathcal{S}_2, \exists s_1 \in \mathcal{S}_1, \forall x \in \mathcal{I}^* \lambda_2(s_2, x) = \lambda_1(s_1, x).$

« Deux automates  $\mathcal{A}_1$  et  $\mathcal{A}_2$  sont équivalents si pour tout état  $e_1$  de  $\mathcal{A}_1$  il existe un état  $e_2$  dans  $\mathcal{A}_2$  tel que pour toute séquence la sortie en partant de  $e_1$  dans  $\mathcal{A}_1$  est la même que celle en partant de  $e_2$  dans  $\mathcal{A}_2$ , et vice-versa. »





## Automate minimal

Un automate est minimal ssi il n'existe pas d'automate équivalent ayant strictement moins d'états.

➡ Comment minimiser un automate ?



### États inséparables

Soit un automate  $(\mathcal{S}, \mathcal{I}, \mathcal{O}, \delta, \lambda)$ , les états  $s_1, s_2 \in \mathcal{S}$  sont inséparables ssi  $\forall x \in \mathcal{I}^*, \lambda(s_1, x) = \lambda(s_2, x)$

Des états inséparables peuvent être fusionnés, c'est-à-dire toutes les transitions sont redirigées vers l'un et l'autre est supprimer.

➡ Comment détecter les états inséparables ?



## Trouver les états séparables

- Créons un ensemble des états séparables en  $i$  lettres, c'est-à-dire  $\mathcal{T}_i = \{(s_1, s_2) \in \mathcal{S}^2, \exists x \in \mathcal{I}^i \lambda(s_1, x) \neq \lambda(s_2, x)\}$   
« On peut trouver une séquence de  $i$  lettres telle que la sortie en partant de  $s_1$  soit différente de celle en partant de  $s_2$  ».



## Trouver les états séparables

- ⊗ Créons un ensemble des états séparables en  $i$  lettres, c'est-à-dire  $\mathcal{T}_i = \{(s_1, s_2) \in \mathcal{S}^2, \exists x \in \mathcal{I}^i \lambda(s_1, x) \neq \lambda(s_2, x)\}$   
« On peut trouver une séquence de  $i$  lettres telle que la sortie en partant de  $s_1$  soit différente de celle en partant de  $s_2$  ».
- ⊗  $\mathcal{T}_1$  se construit à partir de la définition de l'automate.



- ⊗ Créons un ensemble des états séparables en  $i$  lettres, c'est-à-dire  $\mathcal{T}_i = \{(s_1, s_2) \in \mathcal{S}^2, \exists x \in \mathcal{I}^i \lambda(s_1, x) \neq \lambda(s_2, x)\}$   
« On peut trouver une séquence de  $i$  lettres telle que la sortie en partant de  $s_1$  soit différente de celle en partant de  $s_2$  ».
- ⊗  $\mathcal{T}_1$  se construit à partir de la définition de l'automate.
- ⊗ Puis par récurrence on obtient

$$\begin{aligned}\mathcal{T}_{i+1} = & \{(s_1, s_2) \in \mathcal{S}^2, (s_1, s_2) \in \mathcal{T}_1 \\ & \quad \vee \exists a \in \mathcal{I}, (\delta(s_1, a), \delta(s_2, a)) \in \mathcal{T}_i\}\end{aligned}$$



## Trouver les états séparables

- ⊗ Créons un ensemble des états séparables en  $i$  lettres, c'est-à-dire  $\mathcal{T}_i = \{(s_1, s_2) \in \mathcal{S}^2, \exists x \in \mathcal{I}^i \lambda(s_1, x) \neq \lambda(s_2, x)\}$   
« On peut trouver une séquence de  $i$  lettres telle que la sortie en partant de  $s_1$  soit différente de celle en partant de  $s_2$  ».
- ⊗  $\mathcal{T}_1$  se construit à partir de la définition de l'automate.
- ⊗ Puis par récurrence on obtient

$$\begin{aligned}\mathcal{T}_{i+1} = & \{(s_1, s_2) \in \mathcal{S}^2, (s_1, s_2) \in \mathcal{T}_1 \\ & \quad \vee \exists a \in \mathcal{I}, (\delta(s_1, a), \delta(s_2, a)) \in \mathcal{T}_i\}\end{aligned}$$

- ⊗ On remarque que  $\mathcal{T}_i \subseteq \mathcal{T}_{i+1}$



## Trouver les états inséparables (suite)

➡ Quand s'arrêter ?



## Trouver les états inséparables (suite)

➡ Quand s'arrêter ?

Quand  $\mathcal{T}$  ne grossit plus, au pire,  $\mathcal{T}_\infty = \mathcal{T}_{|\mathcal{S}|-1}$ .



## Trouver les états inséparables (suite)

➡ Quand s'arrêter ?

Quand  $\mathcal{T}$  ne grossit plus, au pire,  $\mathcal{T}_\infty = \mathcal{T}_{|\mathcal{S}|-1}$ .

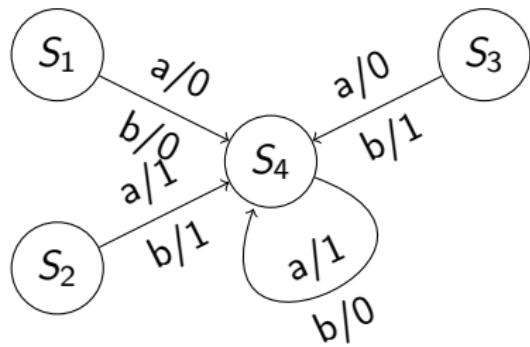
On a alors trouvé les états séparables. Toutes les paires qui ne sont pas dans  $\mathcal{T}_\infty$  peuvent être fusionnées.



- Si un automate n'est pas minimal, il n'existe pas de séquence discriminante (si l'automate n'est pas minimal, alors, il y a des états inséparables).



- Si un automate n'est pas minimal, il n'existe pas de séquence discriminante (si l'automate n'est pas minimal, alors, il y a des états inséparables).
- L'inverse n'est pas vrai.



- ✖ Si un automate n'est pas minimal, il n'existe pas de séquence discriminante (si l'automate n'est pas minimal, alors, il y a des états inséparables).
- ✖ L'inverse n'est pas vrai.
- ✖ Un automate minimal a toujours une séquence d'identification.



## Trouver une séquence d'identification

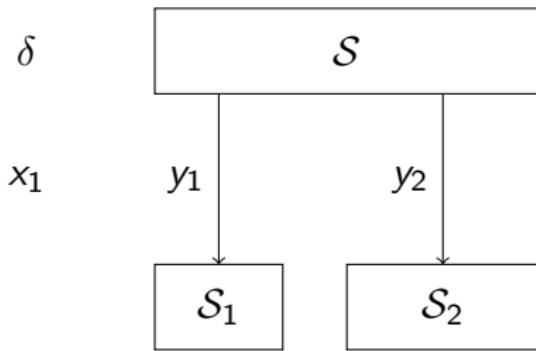
$\delta$

$S$

☞ On trouve une courte séquence pour discriminer l'ensemble des états



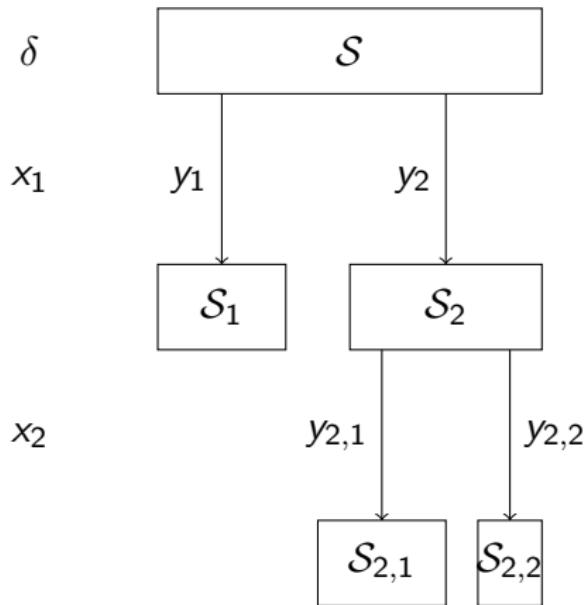
## Trouver une séquence d'identification



- ➡ On trouve une courte séquence pour discriminer l'ensemble des états
- ➡ On détermine des ensembles  $\mathcal{S}_1$ ,  $\mathcal{S}_2$  selon la sortie  $y_1$  ou  $y_2$  donnée par la séquence  $x_1$ .
- ➡ On recommence avec  $\mathcal{S}_2$



## Trouver une séquence d'identification



➡ On trouve une courte séquence pour discriminer l'ensemble des états

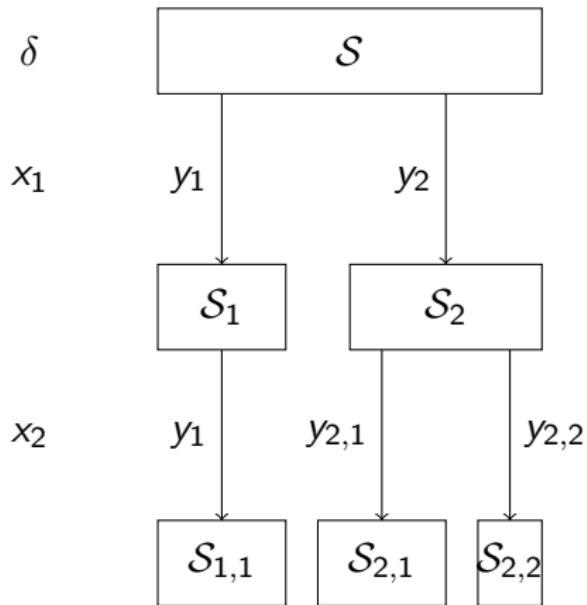
➡ On détermine des ensembles  $S_1$ ,  $S_2$  selon la sortie  $y_1$  ou  $y_2$  donnée par la séquence  $x_1$ .

➡ On recommence avec  $S_2$

➡ Il faut diminuer la taille des sous-ensembles



## Trouver une séquence d'identification



➡ On trouve une courte séquence pour discriminer l'ensemble des états

➡ On détermine des ensembles  $S_1$ ,  $S_2$  selon la sortie  $y_1$  ou  $y_2$  donnée par la séquence  $x_1$ .

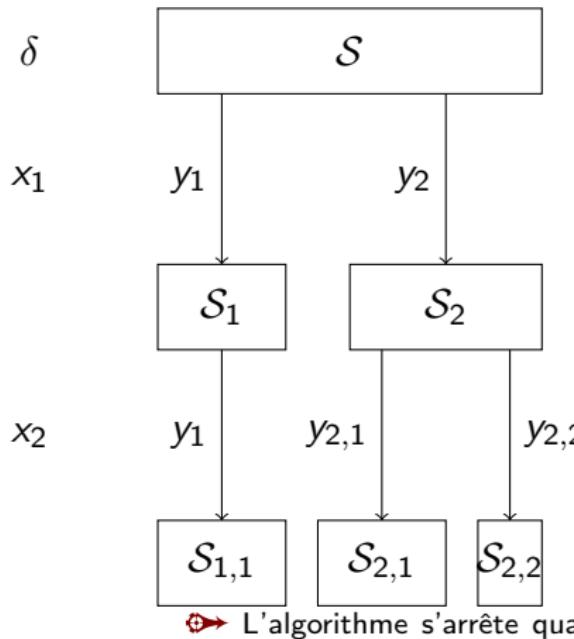
➡ On recommence avec  $S_2$

➡ Il faut diminuer la taille des sous-ensembles

➡ Ici  $S_1$  n'est pas scindé par  $x_2$ .



## Trouver une séquence d'identification



☞ On trouve une courte séquence pour discriminer l'ensemble des états

☞ On détermine des ensembles  $\mathcal{S}_1$ ,  $\mathcal{S}_2$  selon la sortie  $y_1$  ou  $y_2$  donnée par la séquence  $x_1$ .

☞ On recommence avec  $\mathcal{S}_2$

☞ Il faut diminuer la taille des sous-ensembles

☞ Ici  $\mathcal{S}_1$  n'est pas scindé par  $x_2$ .

☞ L'algorithme s'arrête quand tout les ensembles sont de taille 1.



## Trouver une séquence d'identification - exemple

