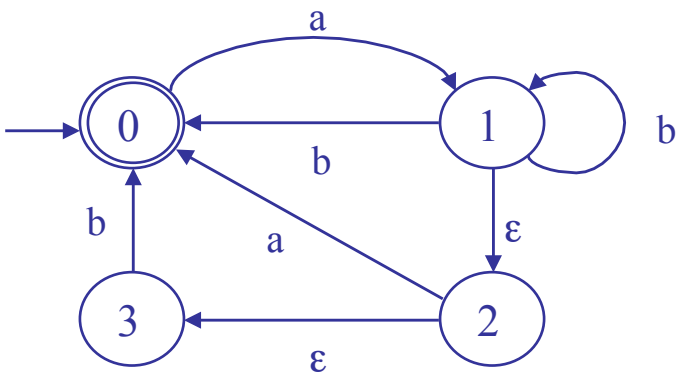
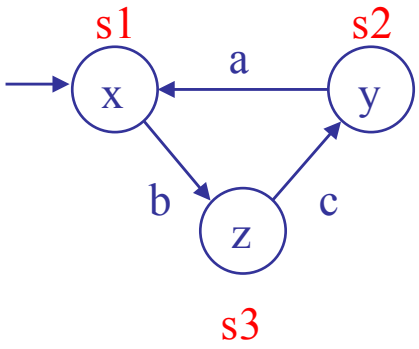
Modélisation des SED

Théorie des langages
Automates à états finis





Théorie des Langages

- Un **alphabet** est un ensemble **fini** de symboles. Dans le cas d'un SED, un alphabet pourra représenter l 'ensemble des événements d'entrée possibles d'un système

  Exemple: $\Sigma = \{a,b\}$

- Un **mot** (ou chaine ou séquence) est une séquence **finie** d'éléments d'un alphabet Σ

  Exemple: *abba* est un mot défini sur Σ.

  - ε est le mot vide
  - Σ^* représente l'ensemble de tous les mots que l'on peut construire sur Σ (ε inclus)

    exemple, $\Sigma^* = \{\varepsilon, a, b, aa, bb, ab, ba, aaa, aab, abb, \ldots\}$

- Un **langage** est un ensemble L de mots définis sur l'alphabet Σ.
  - C'est un sous-ensemble de Σ^*
  - Exemple $\Sigma = \{a, b, g\}$,
    - L1 = {ε, a, ab}
    - L2 = {ensemble des mots de longueur 3 commençant par a}

      → **langage fini** (9 mots) : {aaa, aab, aag, aba, abb, abg, aga, agb, agg}
    - L3 = {ensemble des mots commençant par a} → **langage infini**

Opérations sur les langages

- Soit deux langages L1, L2 $\subseteq \Sigma^*$

- **UNION** $L_1 \cup L_2 = \{w : w \in L_1 \text{ ou } w \in L_2\}$

- **INTERSECTION** $L_1 \cap L_2 = \{w : w \in L_1 \text{ et } w \in L_2\}$

- **CONCATÉNATION** $L_1.L_2 = \{s \in \Sigma^* : (s=s_1s_2) \text{ et } (s_1 \in L_1) \text{ et } (s_2 \in L_2)\}$

- **FERMETURE ITERATIVE** (*fermeture de Kleene*)
  $L^* = \{\varepsilon\} \cup L \cup LL \cup LLL \cup \ldots$
  $= \{\varepsilon\} \cup L^1 \cup L^2 \cup L^3 \cup \ldots L^n \cup \ldots$

- **FERMETURE PREFIXIELLE**
  - **Préfixe d'un mot**
    u, v $\in \Sigma^*$, u est préfixe de v si $\exists$ w $\in \Sigma^*$: v = uw
    $\forall$ s $\in \Sigma^*$, ε et s sont des préfixes de s
  - **Fermeture préfixielle**
    $L \subseteq \Sigma^*$, $\overline{L} = \{u \in \Sigma^* : \exists\, v \in \Sigma^* \ (uv \in L)\}$
    Si $\overline{L} = L$ alors L est dit préfixe-clos ou fermé

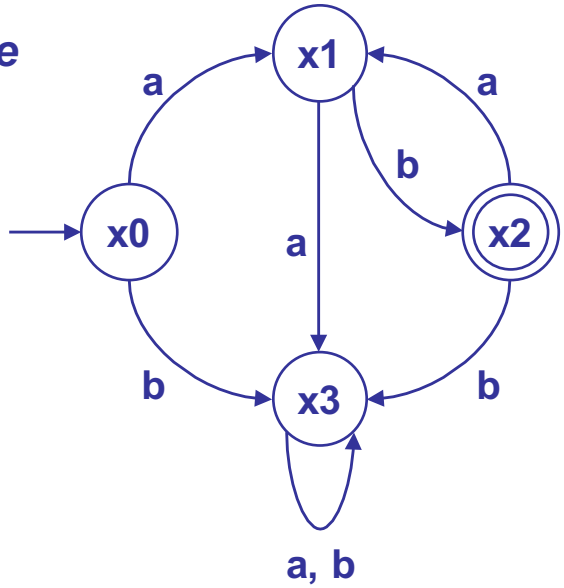
Opérations sur les langages : exemples

- $\Sigma = \{a,d\}$, $L1 = \{a, ad\}$, $L2 = \{aa\}$
  - **UNION** $L_1 \cup L_2 = \{a, aa, ad\}$
  - **INTERSECTION** $L_1 \cap L_2 = \emptyset$
  - **CONCATÉNATION** $L_1.L_2 = \{aaa, adaa\}$
  - **FERMETURE ITERATIVE** $L1^* = \{\varepsilon\} \cup L^1 \cup L^2 \cup L^3 \cup \ldots L^n \cup \ldots$
    $L1^* = \{\varepsilon\} \cup \{a, ad\} \cup \{aa, aad, ada, adad\} \cup \ldots$
  - **FERMETURE PREFIXIELLE** $\overline{L2} = \{\varepsilon , a, aa\}$
- $\Sigma = \{a,b,g\}$, $L1 = \{\varepsilon, a, abb\}$, $L2 = \{\varepsilon, g\}$
  - **CONCATÉNATION** $L1.L2 = \{\varepsilon, g, a, ag, abb, abbg\}$
  - **FERMETURE ITERATIVE** $L1^* = \{\varepsilon, a, abb, aa, aabb, abba, abbabb, \ldots\}$,
    $L2^* = \{\varepsilon, g, gg, ggg, gggg, \ldots\}$
  - **FERMETURE PREFIXIELLE** $\overline{L1} = \{\varepsilon, a, ab, abb\}$, $\overline{L2} = \{\varepsilon, g\}$

Automates

- Un automate est un quintuplet $G = (X, \Sigma, f, x_0, X_m)$ où :
  - X est l'ensemble des états
  - Σ est un alphabet d 'entrée
  - f est la fonction de transition d'états définie de $X \times \Sigma \rightarrow X$ qui associe un état de départ et un symbole d'entrée à un état d'arrivée
  - x_0 est l'état initial
  - X_m est l'ensemble des états finaux ou états marqués ($X_m \subseteq X$)

Exemple



- $X = \{x_0, x_1, x_2, x_3\}$
- $\Sigma = \{a,b\}$
- f est représentée par les arcs associés à des symboles de l'alphabet $f(x_0, a) = x_1$
-x_0 état initial
-$X_m = \{x_2\}$

Langages générés

- Le domaine de f est souvent étendu de X x Σ à X x Σ*
  - f(x, ε) = x et f(x, se) = f(f(x,s), e) pour s ∈ Σ* et e ∈ Σ

- ***Langage généré***

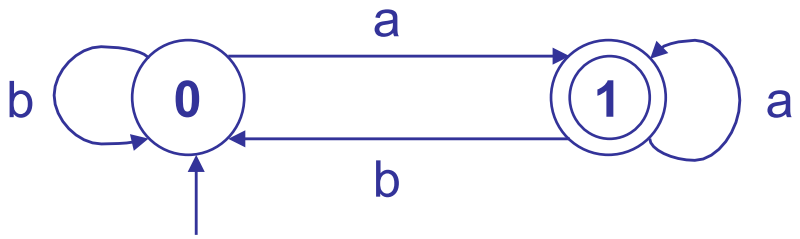
$$\mathcal{L}(G) = \{s \in \Sigma^* : f(x_0, s) \text{ est défini}\}$$

  - $\mathcal{L}(G)$ est préfixe-clos par définition: un chemin depuis x0 est possible si tous ses préfixes sont aussi possibles
  - Si f est une fonction totale, $\mathcal{L}(G) = \Sigma^*$

- ***Langage marqué***

$$\mathcal{L}_m(G) = \{s \in \mathcal{L}(G) : f(x_0, s) \in X_m\}$$

Exemple



- $\mathcal{L}(G) = \Sigma^*$
- $\mathcal{L}_m(G) =$ {a, aa, ba, aaa, aba, bba, …}
  c'est-à-dire tous les mots finissant par a

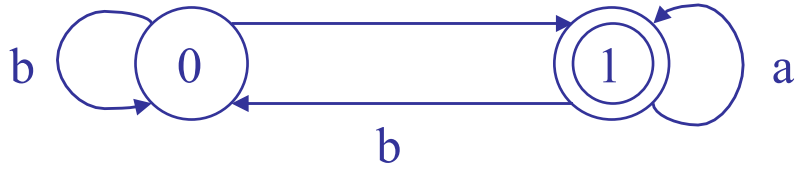
Langages réguliers

- Les **expressions régulières** sont définies à partir d'expressions élémentaires et des opérations d'union, de concaténation et de fermeture itérative.
  - Tout événement est une expression régulière.
  - Ø et ε sont des expressions régulières.
  - Si s et r sont des expressions régulières, alors s+r, s.r, s* et r* (union, concaténation, fermeture de Kleene) sont des expressions régulières.

- **Expressions régulières** et **langages**

L'expression régulière b + a.(b + c) correspond au langage L = {b, ab, ac}

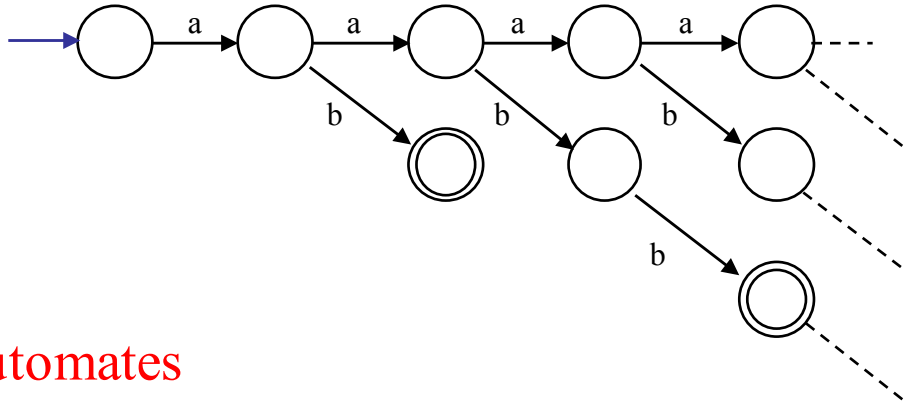
- Un **langage régulier** est un langage marqué par un automate fini (**théorème de Kleene**)
  - L'automate est fini mais le langage peut être fini ou infini
  - L'expression régulière est une notation plus concise
  - L'automate permet une manipulation plus aisée



- $\mathcal{L}(G) = \Sigma^*$
- $\mathcal{L}_m(G) = \{a, aa, ba, aaa, aba, bba, \ldots\}$
  c'est-à-dire tous les mots finissant par a

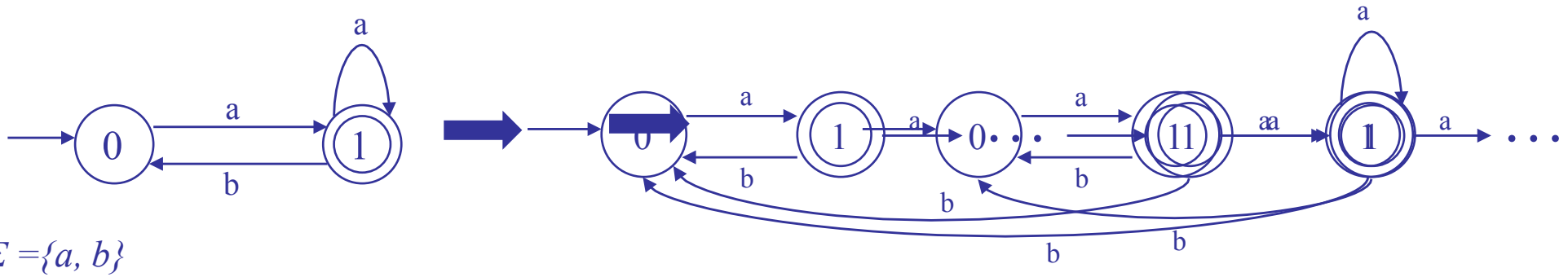
Propriétés

- Tous les langages ne peuvent pas être représentés par des automates à états finis
  - L = {ε, ab, aabb, aaabbb, …} = {$a^n b^n$: n>0} sur l'ensemble Σ = {a,b}



- Équivalence entre automates
  - Plusieurs solutions pour construire un automate générant et marquant un même langage
  - Deux automates sont équivalents s'ils génèrent et marquent les mêmes langages :

    G1 et G2 sont équivalents si L (G1) = L (G2) et Lm (G1) = Lm (G2)



- *Σ ={a, b}*
- *ℒ (G) est l'ensemble des mots de Σ* commençant par l'événement a et ne comportant plusieurs occurrences successives de b*
- *$ℒ_m$ (G) est le sous-ensemble de ℒ (G) composé de mots terminant par a*

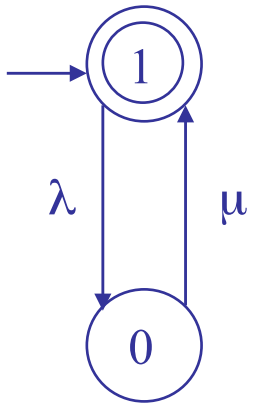
- **Lemmes d'Arden**

  Soit Σ un alphabet et A, B deux langages sur Σ, l'équation X = XA + B où X est un langage sur Σ :

  – si $\varepsilon \notin A$, admet une solution unique $X = B.A^*$

  – si $\varepsilon \in A$, admet un ensemble de solutions $X = \{L.A^* , B \subset L\}$

- **Exemple d'une machine soumise à des pannes**



– 1: état de marche (état initial et état marqué)
– 0: état de panne
– λ: panne de machine (événement)
– μ : réparation de machine (événement)
– f(1, λ) = 0, f(1, μ) non définie
– f(0, μ) = 1, f(0, λ) non définie
– $x_0 = 1$ et $x_m = 1$

$L_1 = L_0 \mu + \varepsilon$

$L_0 = L_1 \lambda$

Par substitution $L_1 = L_1\lambda\mu + \varepsilon$

*Règles d'Arden : X = XA + B a pour solution X= BA**

$L_1 = (\lambda\mu)^*$ et $L_0 = (\lambda\mu)^*\lambda$ d'où

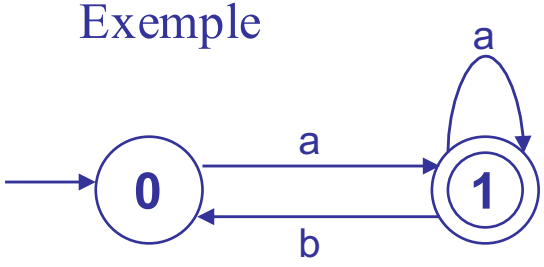
$L(G) = L1 + L0 = (\lambda\mu)^*(\varepsilon + \lambda)$

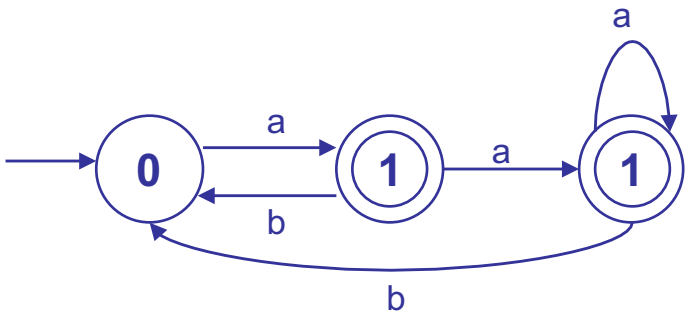
$L_m(G) = L1 = (\lambda\mu)^*$

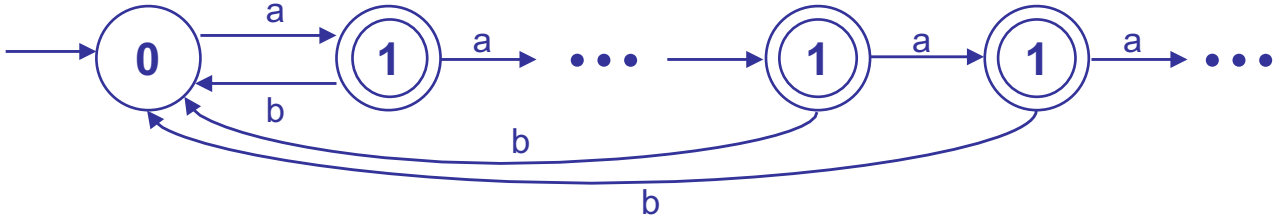
- Équivalence entre automates
  - Plusieurs solutions possibles pour construire un automate générant ou marquant un même langage
  - Deux automates sont équivalents s'ils génèrent et marquent les mêmes langages:

G1 et G2 sont équivalents si $\mathcal{L}(G1) = \mathcal{L}(G2)$ *et* $\mathcal{L}_m(G1) = \mathcal{L}_m(G2)$

Exemple







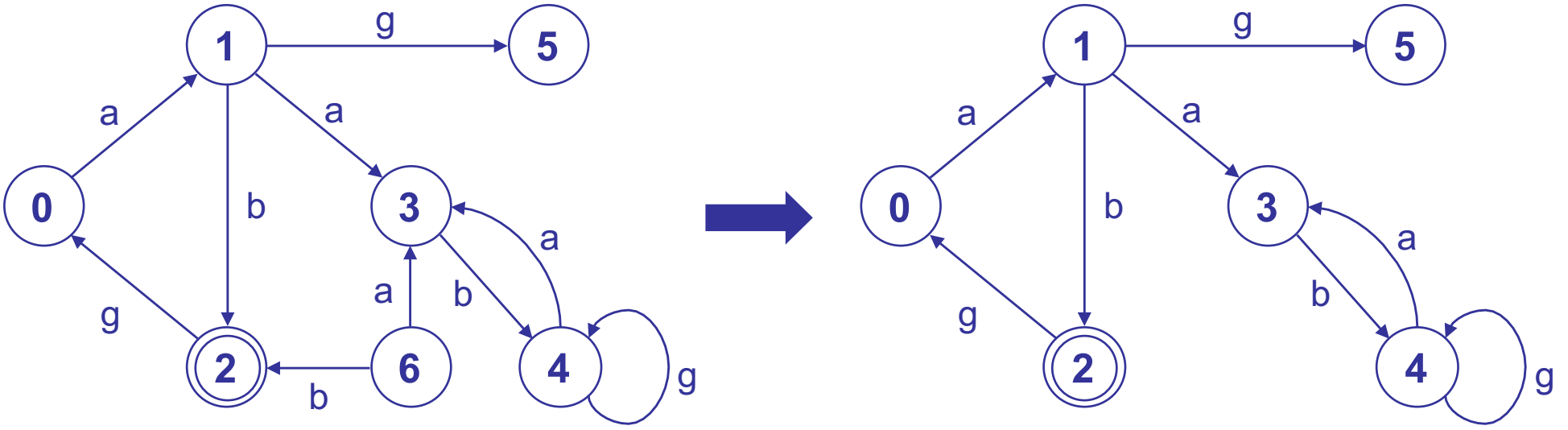
- $\Sigma = \{a, b\}$
- $\mathcal{L}(G)$ est l'ensemble des mots de Σ^* commençant par l'événement a et ne comportant plusieurs occurrences successives de b
- $\mathcal{L}_m(G)$ est le sous-ensemble de $\mathcal{L}(G)$ composé de mots terminant par a

Opérations sur automates

- Accessibilité
  - L'ensemble des états accessibles est défini par:

$$X_{ac} = \{x \in X : \exists\, s \in \Sigma^* \ (f(x_0, s) = x)\}$$

  - $Ac(G) = \{X_{ac}, \Sigma, f_{ac}, x_0, X_{ac,m}\}$ avec $f_{ac} = f \mid X_{ac} \times \Sigma \rightarrow X_{ac}$ et $X_{ac,m} = X_m \cap X_{ac}$
  - Les langages $\mathcal{L}(G))$ et $\mathcal{L}_m(G)$ ne sont pas affectés par cette opération

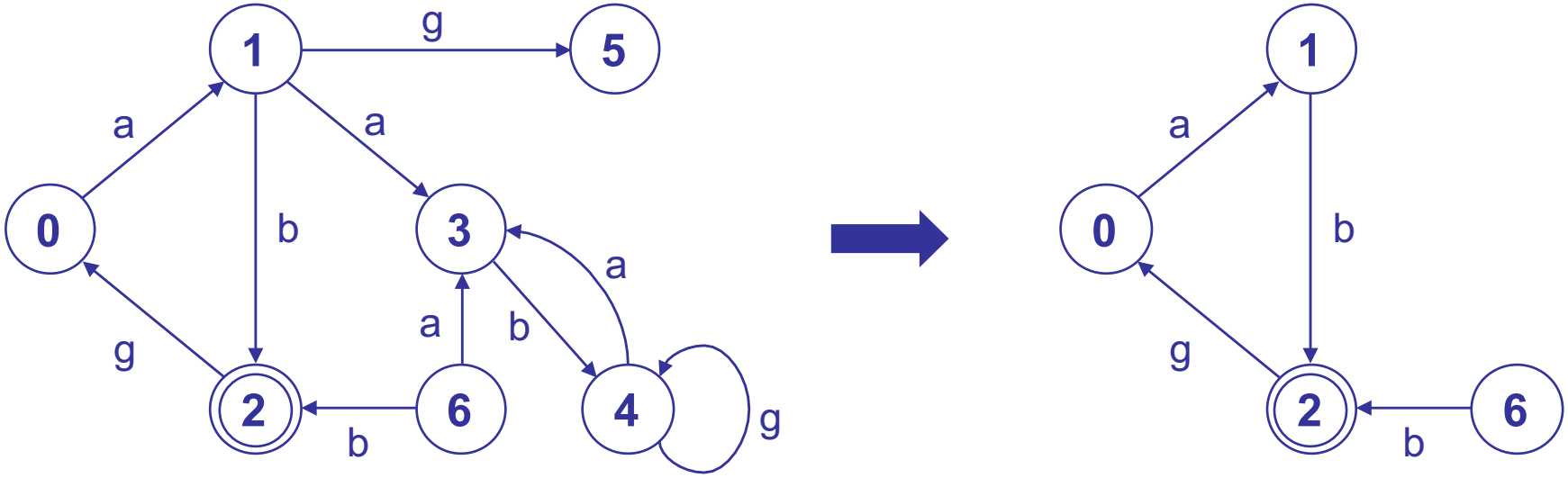


Opérations sur automates

- Co-accessibilité
  - Un état x est dit co-accessible s'il existe une chaîne conduisant à un état marqué qui passe par x: il existe un chemin depuis x vers un état marqué.

$$X_{coac} = \{x \in X : \exists s \in \Sigma^* \ (f(x, s) \in X_m)\}$$

  - $CoAc(G) = \{X_{coac}, \Sigma, f_{coac}, x_0, X_m\}$ avec $f_{coac} = f \mid X_{coac} \times \Sigma \rightarrow X_{coac}$
  - Le langage $\mathcal{L}_m(G)$ n'est pas affecté par cette opération *(les états supprimés ne se trouvent pas sur un chemin de x_0 vers un état marqué)*
  - Le langage $\mathcal{L}(G)$ est restreint à $\mathcal{L}(Coac(G)) = \overline{\mathcal{L}_m(Coac(G))}$

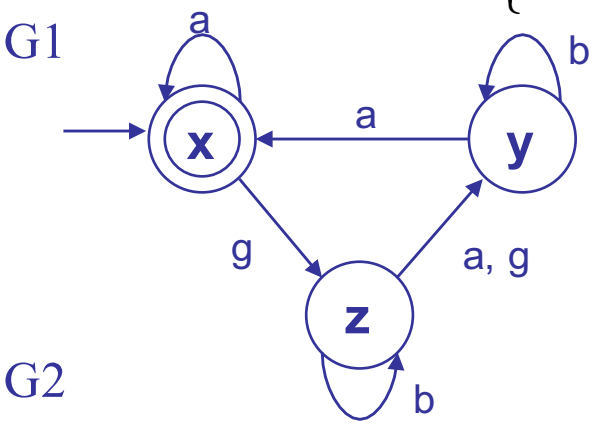


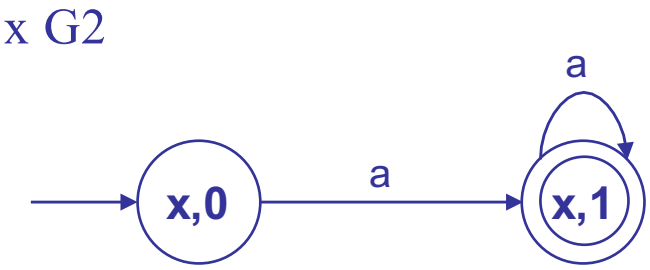
Opérations sur automates

- Produit synchrone
  - Le produit des automates G1 et G2 est l'automate:

$$G1 \times G2 = Ac(X_1 \times X_2,\ \Sigma_1 \cap \Sigma_2,\ f,\ (x_{01},x_{02}),\ X_{m1} \times X_{m2})$$

Où $f((x_1,x_2), e) := \begin{cases} (f_1(x_1,e), f_2(x_2, e)) \text{ si } f_1(x1,e) \text{ et } f_2(x2, e) \text{ sont tous deux définis} \\ \text{indéfini sinon} \end{cases}$





$\Sigma = \Sigma_1 \cap \Sigma_2 = \{a, b\}$
$f((x,0), a) = (f_1(x,a), f_2(0, a)) = (x, 1)$
$f((x,1), a) = (f_1(x,a), f_2(1, a)) = (x, 1)$

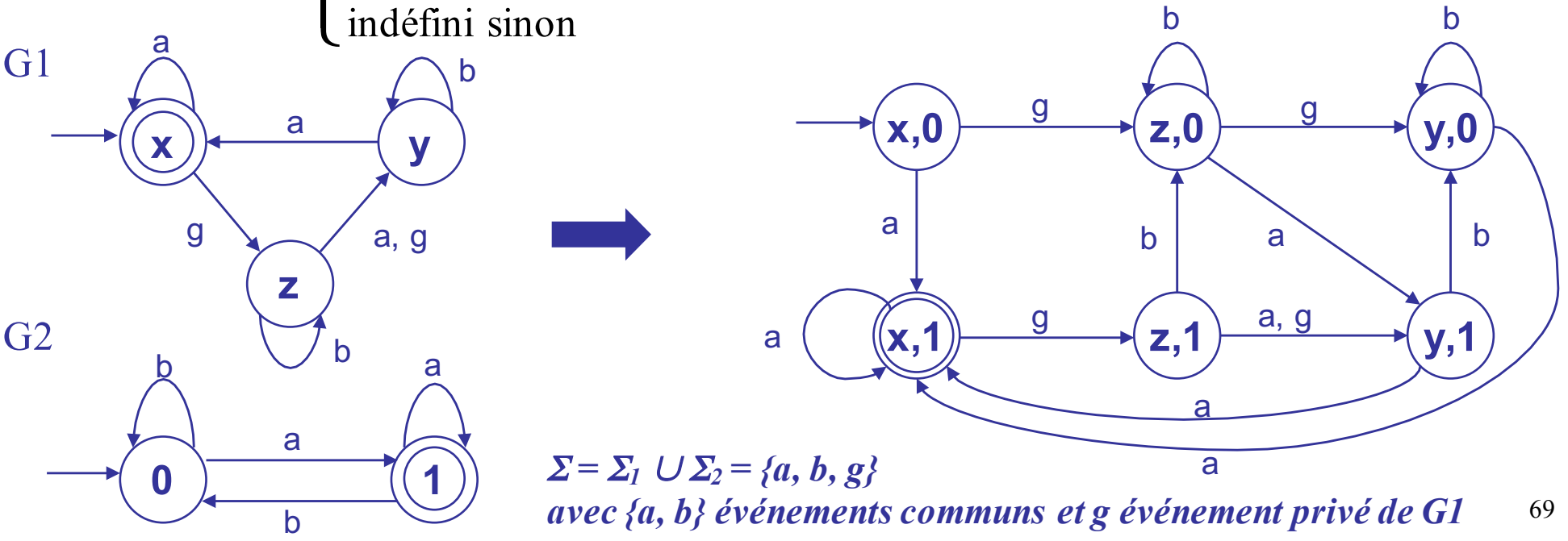
Opérations sur automates

- Composition synchrone
  - La composition synchrone de deux automates G1 et G2 est l'automate:

$$G1 \,||\, G2 = Ac(X_1 \times X_2,\ \Sigma_1 \cup \Sigma_2,\ f,\ (x_{01}, x_{02}),\ X_{m1} \times X_{m2})$$

Où $f((x_1,x_2), e) :=$

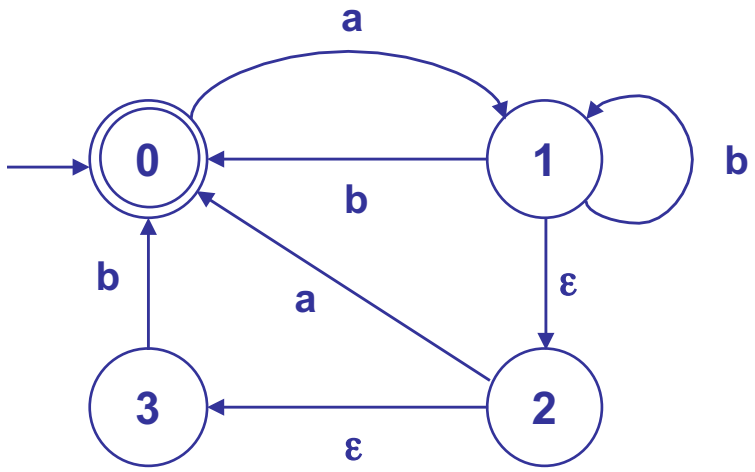
- $(f_1(x_1,e), f_2(x_2, e))$ si $f_1(x1,e)$ et $f_2(x2, e)$ sont tous deux définis
- $(f_1(x_1,e), x_2)$ si $f_1(x_1,e)$ est défini pour $e \in \Sigma_1 \setminus \Sigma_2$ *(événements privés)*
- $(x_1, f_2(x_2,e))$ si $f_2(x_2,e)$ est défini pour $e \in \Sigma_2 \setminus \Sigma_1$ *(événements privés)*
- indéfini sinon



$\Sigma = \Sigma_1 \cup \Sigma_2 = \{a, b, g\}$
avec {a, b} événements communs et g événement privé de G1

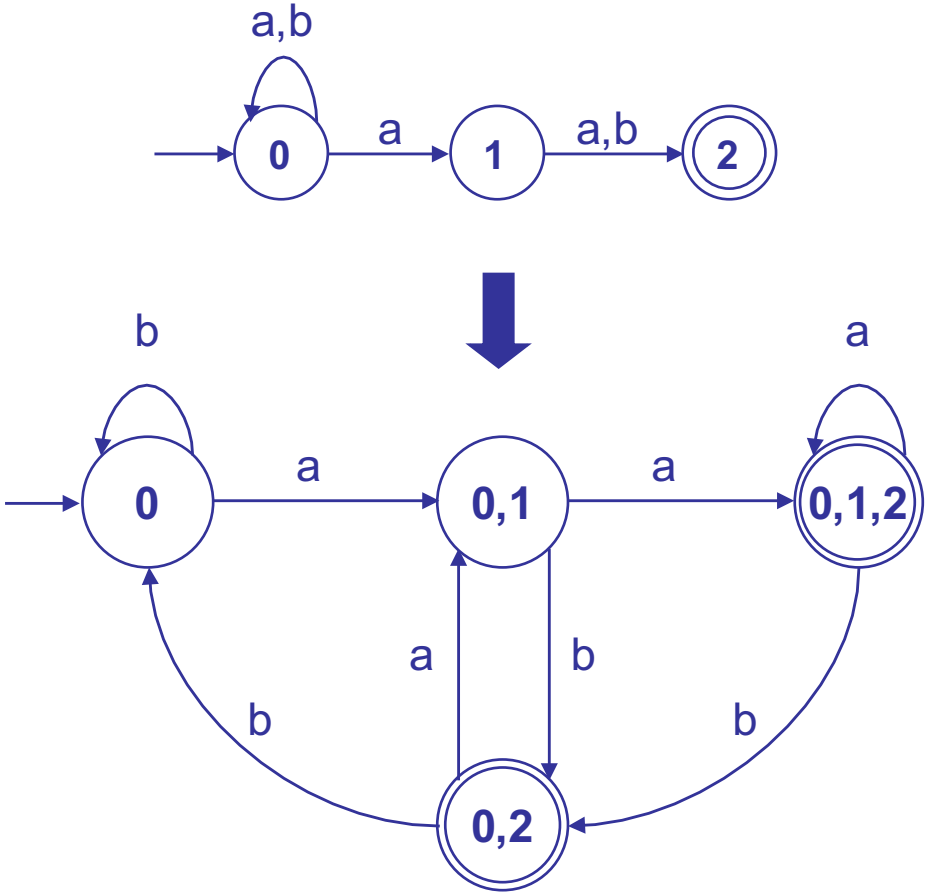
Déterminisme des automates

- Déterministes
  - La fonction f est définie sur $X \times \Sigma \rightarrow X$
  - Un état ne possède qu'au plus une image par f
- Non-déterministes
  - $(X, \Sigma \cup \{\varepsilon\}, f_{nd}, x_0, X_m)$
  - La fonction f_{nd} est définie sur $X \times \Sigma \cup \{\varepsilon\} \rightarrow 2^X$ où 2^X est l'ensemble des sous-ensembles de X
  - Un état peut posséder plusieurs images et une transition entre états peut être associée au mot vide ε



Déterminisme des automates

- Transformation non-déterministe/déterministe
  - *tout automate indéterministe peut être transformé en automate déterministe équivalent (appelé « observateur »)*



Etape 1: on examine l'état 0
- a conduit à l'état 0 ou 1
- b conduit à l'état 0

Etape 2: on examine l'état (0,1)
- a conduit à l'état 0 ou 1 ou 2
- b conduit à l'état 0 ou 2

Etape 3: on examine l'état (0,1,2)
- a conduit à l'état 0 ou 1 ou 2
- b conduit à l'état 0 ou 2

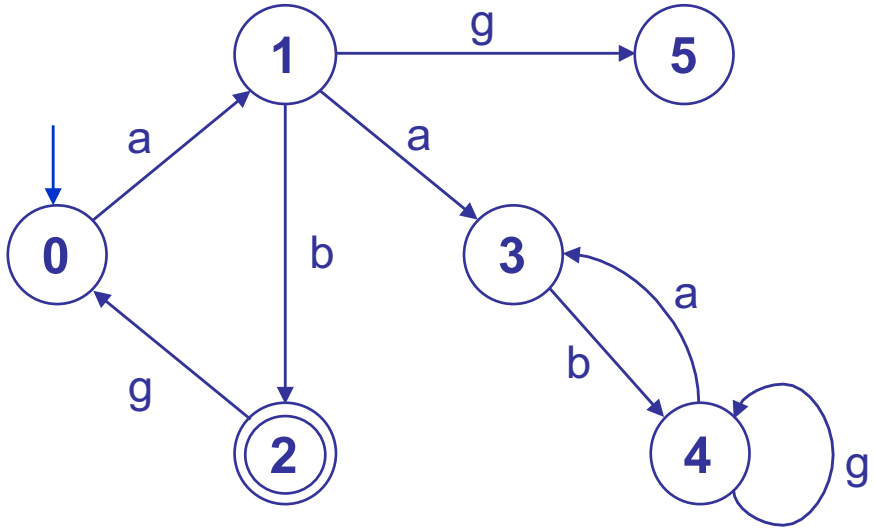
Etape 4: on examine l'état (0,2)
- a conduit à l'état 0 ou 1
- b conduit à l'état 0

Blocage des automates

- Deadlock
  - L'automate arrive dans un état x à partir duquel aucun événement de Σ ne peut être sensibilisé

- Livelock
  - L'automate arrive dans un groupe d'états non marqués dont on ne peut ressortir (groupe absorbant)

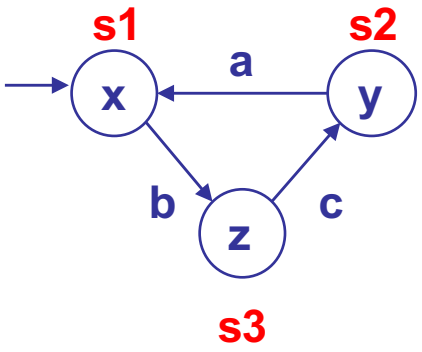
Par définition, $\overline{\mathcal{L}_m(G)} \subseteq \mathcal{L}(G)$,
l'automate est dit:
- non bloquant si $\overline{\mathcal{L}_m(G)} = \mathcal{L}(G)$
- bloquant si $\overline{\mathcal{L}_m(G)} \subset \mathcal{L}(G)$



Automates particuliers

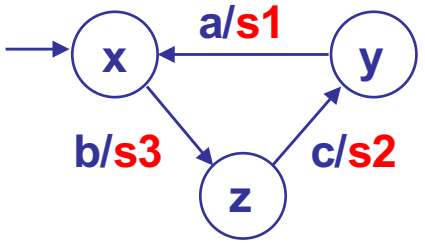
- Notions d'événements d'entrées et de sorties
- Machine de Moore
  - Une fonction de sortie assigne un événement de sortie à chaque état

$$X_{t+1} = f(X_t, E_t)$$
$$Y_t = g(X_t)$$



- Machine de Mealy
  - Transitions associées à des événements de la forme entrée/sortie
  - *Interprétation d'une transition ei/eo depuis l'état x: lorsque l'automate est dans l'état x et qu'il « reçoit » l'événement ei, il évolue vers l'état y et « émet » la sortie eo.*

$$X_{t+1} = f(X_t, E_t)$$
$$Y_t = g(X_t, E_t)$$



SED Pourquoi ?

- **Exemples de Systèmes à Evénements Discrets**
  - **Réseaux de communication** : protocoles
  - **Systèmes manufacturiers** : flux de production, commande des systèmes de production
  - **Systèmes embarqués** : automobile, avionique, ferroviaire
  - **Systèmes hybrides** : continus / discrets


Aéronautique

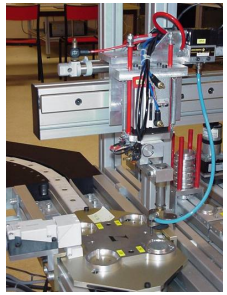

Ferroviaire


Automobile


Militaire


Spatial


Energie


Système de production manufacturier

- **Langages et automates**
  - Opérations de composition/produit sur les automates
    - Systèmes complexes de trop grande taille impossible à modéliser « d'un seul coup »
    - Petits modèles correspondant chacun à un petit sous-système du système complet puis composition pour obtenir le modèle global
  - Identification des langages générés ou marqués : recherche de séquences
    - Trajectoires à suivre pour atteindre un état désiré (conduite de systèmes complexe)
    - Séquences critiques conduisant à un état redouté tel qu'une défaillance