MATHÉMATIQUES : Q2

Table des matières

1. LES RELATIONS BINAIRES

I. Introduction

❖ Les couples

1 couple (a,b) est 1 paire ordonnée d'objets

a = ensemble de départ

b = ensemble d'arrivée

Propriétés :

- Si (a ≠ b) => ((a,b) ≠ (b,a))
- Si ((a,b) = (c,d)) ⟺ ((a = c) ⋀ (b =d))
- 1 couple identique : (a,a)

❖ Produit Cartésien

⇨ 2 ensembles A et B (AxB) = l'ensemble de tous les couple (a,b) tels que a est un élément de B ET b est un élément de A

Propriétés :

- |AxB| =|A| x |B|
- |AxA| = |A²|

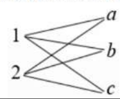
Exemple :
Si

- $A = \{1, 2\}$
- $B = \{a, b, c\}$

Alors

- $A \times B = \{(1, a), (1, b), (1, c), (2, a), (2, b), (2, c)\}$

→ on a associé chaque élément de A avec chaque élément B :



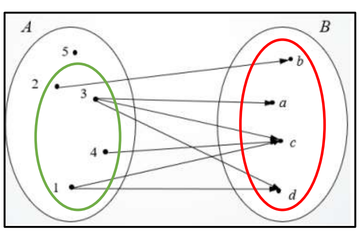
II. Les relations Binaires

❖ Définitions

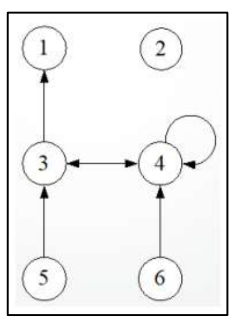
Prenons comme exemple 2 ensembles : A et B

R est une relation binaire de A vers B ssi R est un sous-ensemble de AxB

❖ Représentation sagittale



❖ Digraphe



❖ Domaine & Image

- Domaine dom(R) = {} : éléments sur l'ensemble de départ ayant AU MOINS 1 relation avec l'autre ensemble.
- Image Im(R) = {} : éléments sur l'ensemble d'arrivée ayant AU MOINS 1 relation avec l'autre ensemble.

III. Propriétés des relations sur 1 ensemble

- **La réflexivité** *(La relation R* ***est multiple*** *sur 1 ensemble)*
  - ✓ Digraphe où TOUS les éléments ont 1 flèche vers eux-mêmes (boucle).

- **L'antiréflexivité** $\neg\ (nRn)$ *(La relation R* ***est strictement inférieur*** *sur 1 ensemble)*
  - ✓ Digraphe où AUCUN élément n'a 1 flèche vers lui-même (boucle).

- **La symétrie** *(La relation R* ***est un frère ou sœur d'*** *1 ensemble)*
  - ✓ Digraphe ne comportant AUCUNE flèche simple. Y aura que des doubles flèche ET/OU boucles.

- **L'antisymétrie**
  - ✓ Digraphe ne comportant AUCUNE flèche double. Y aura que des boucles ET/OU flèches simples.

- **La transitivité** *(La relation R* ***est l'ancêtre*** *sur 1 ensemble)*
  - ✓ Si je peux aller d'1 élément **e1** à un autre élément **e2** en 2 étapes (en passant par une autre élément **e3**) , alr il existe une flèche de **e1** vers **e2**.
  - ✓ Si je ne trouve pas → PAS transitive

- **Check-up d'une relation sur base d'une définition mathématique**

Soit la relation sur l'ensemble $\mathbb{Z}$ définie par $x\,R\,y \Leftrightarrow |x-y| \leq 4$.

Quelle(s) propriété(s) a cette relation ?

1. Cette relation est-elle **réflexive ? Oui !**
   Considérons un couple (x,x), autrement dit $y=x$. Alors $|x-y| = |x-x| = |0| = 0 \leq 4$.
   Donc pour tout entier x on a que $(x,x) \in R$ et donc que $x\,R\,x$ est vraie.
   Donc R **est réflexive**.

2. Cette relation est-elle **antiréflexive ? Non !**
   En effet, si on prend le couple $(4,4)$ alors $|4-4| = |0| = 0 \leq 4$.
   On a donc au moins un élément en relation avec lui même.
   Donc R **n'est pas antiréflexive**.

3. Cette relation est-elle **symétrique ? Oui !**
   Considérons un couple (x,y). Si $x\,R\,y$ est vraie alors $|x-y| \leq 4$. Et $|y-x| = |x-y| \leq 4$.
   Donc $y\,R\,x$ est vraie.
   Donc R **est symétrique**.

4. Cette relation est-elle **antisymétrique ? Non !**
   En effet, si on prend le couple $(5,3)$ et $(3,5)$ on a $|5-3| = |2| = 2 \leq 4$ et $|3-5| = |-2| = 2 \leq 4$.
   Donc $5\,R\,3$ et $3\,R\,5$ sont vraies et $5 \neq 3$.
   Donc R **n'est pas antisymétrique**.

5. Cette relation est-elle **transitive ? Non !**
   En effet, si on prend les couples $(1,3)$ et $(3,6)$ alors $|1-3| = |-2| = 2 \leq 4$ et $|3-6| = |-3| = 3 < 4$.
   Donc $1\,R\,3$ et $3\,R\,6$ sont vraies. Mais $1\,R\,6$ est fausse car $|1-6| = |-5| = 5 > 4$.
   Donc R **n'est pas transitive**.

<u>Explication :</u>

On sait que pour une relation réflexive, tous les éléments ont 1 flèches vers eux-mêmes, donc une boucle. Donc on prend un couple avec les mêmes éléments et on regarde si c'est vrai ou pas. Si je trouve un couple où je peux justifier que l'équation est fausse, alr bah la relation est fausse.

IV. Clôtures

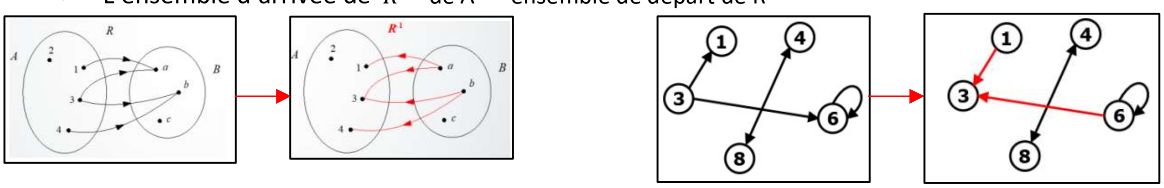
❖ **Clôturer** la relation R == ajouter à R les flèches indispensables pour que R acquière la propriété P.
== ajouter le moins de flèches possibles /!\ Ne PAS retirer des flèches

- ✓ Si 1 clôture est possible ➔ elle est UNIQUE.
- ✓ Les clôtures possibles : - Clôture réflexive / Symétrique / Transitive
- ✓ /!\ L'ordre dans lequel on fait plusieurs clôtures successivement est important. D'abord clôture symétrique **avant** la clôture transitive.

V. Opérations sur les relations binaires

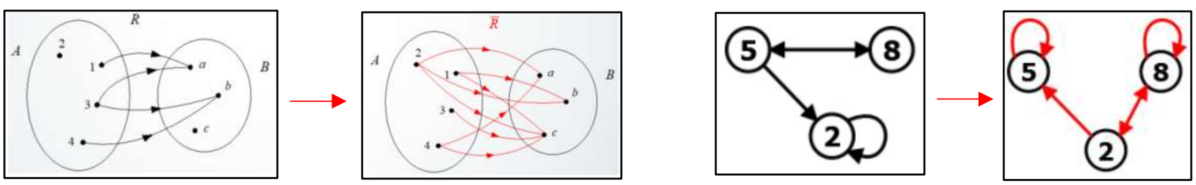
❖ **La réciproque d'1 relation** ~ Inverser le sens ~ R^{-1}

- ✓ L'ensemble de départ de R^{-1} de B == ensemble d'arrivée de R
- ✓ L'ensemble d'arrivée de R^{-1} de A == ensemble de départ de R



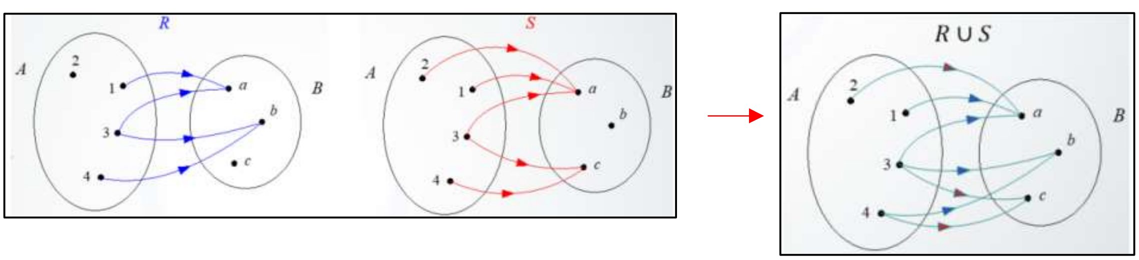
❖ **La complémentaire d'1 relation** ~ $\bar{R}$

- ✓ Tout les couples qui n'appartiennent pas à la relation R



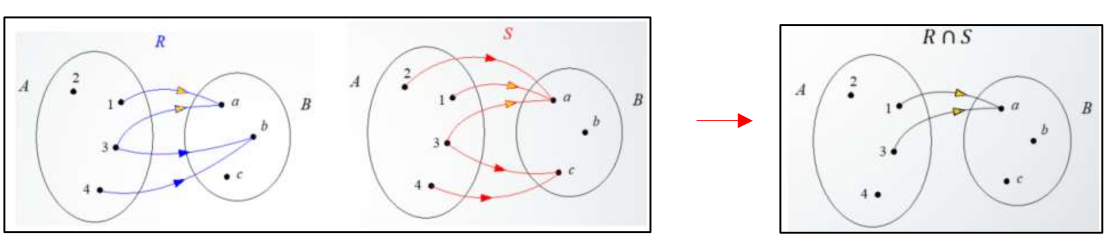
❖ **L'union de 2 relations** ~ **OU** ~ R U S

- ✓ Toutes les relations



❖ **L'intersection de 2 relations** ~ **ET** ~ R ∩ S

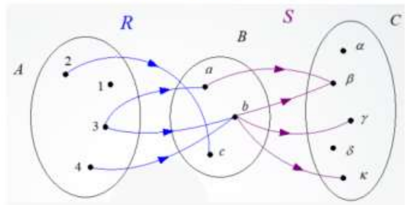
- ✓ Les relations en commun



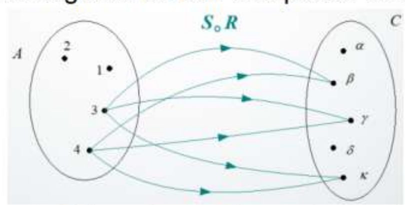
❖ La composée de 2 relations

- ✓ S∘R (S après R → D'abord la relation R puis S)

Soient les représentation sagittales de R de A ver B et S de B vers C



Alors une représentation sagittale de leur composée $S \circ R$ est



<u>Explication :</u>

Je pars du point 3, je vois qu'en passant par le point b, je peux atterrir au point β, γ, κ

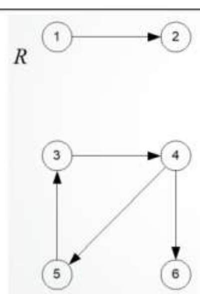
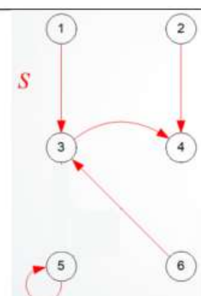
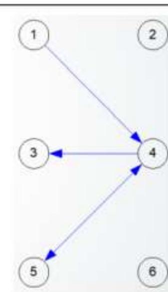
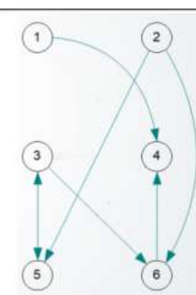
- ✓ R∘S (R après S)

Attention, la composé ∘ de deux relations n'est pas commutative !

En effet, en général $S \circ R \neq R \circ S$.

<u>Exemple :</u>

Voici les relations R, S, $S \circ R$ et $R \circ S$ sur l'ensemble $\{1, 2, 3, 4, 5, 6\}$.

R	S	$S \circ R$	$R \circ S$
			

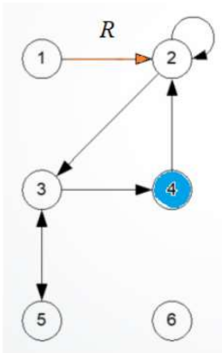
<u>Explication :</u>

Prenons l'exemple pour R∘S

Dans la Relation S, mon élément 1 va chez l'élément 3, et dans la Relation R, mon élément 3 va chez l'élément 4. Donc J'ai une new relation de 1 à 4.

VI. Chemin dans un digraphe

❖ **Définition ==** Représentat° graphique d'une relation sur un ensemble fini E.



- Les disques étiquetés sont appelés **sommets**.
- Les flèches sont appelées **arêtes**.

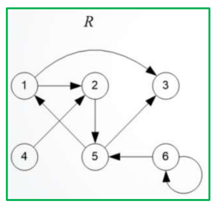
✓ **Chemin de x à y ==** séquence d'arêtes consécutives. L'origine de le 1ère = x. L'extrémité de la dernière = y.

✓ **Cycle ==** chemin dont l'origine de la 1ère arête == l'extrémité de la dernière arête. → cycle = chemin de x à x.

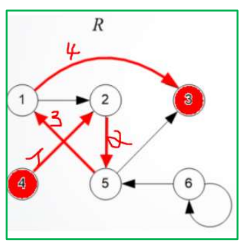
✓ **Longueur ==** le nbre d'arêtes.

Exemple :

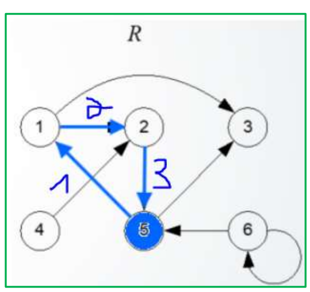
Soit la relation RR sur l'ensemble E={1,2,3,4,5,6}E={1,2,3,4,5,6} dont voici le digraphe



- Le chemin (4 , 2 , 5 , 1 , 3)(4 , 2 , 5 , 1 , 3) est un RR-chemin de **longueur 4** de 44 à 33 :

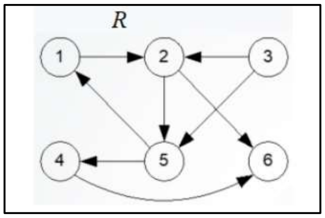


- Le chemin (5,1,2,5)(5,1,2,5) est un RR-**cycle** de **longueur 3** de 55 à 55 :

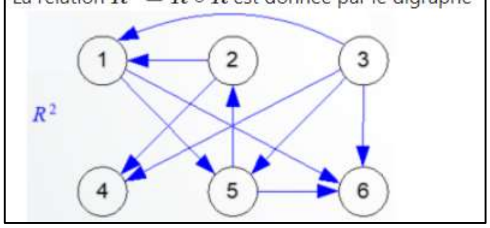


-

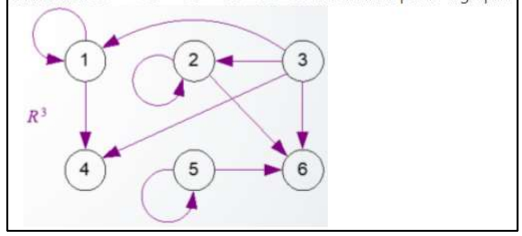
- ❖ **Chemin Hamiltonien ==** S'il ne contient AUCUN sous-chemin qui soit un cycle.
- ❖ **Puissance d'une relation**



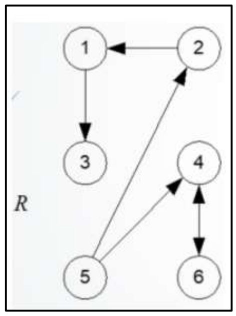
La relation $R^2 = R \circ R$ est donnée par le digraphe



La relation $R^3 = R^2 \circ R = R \circ R \circ R$ est donnée par le digraphe



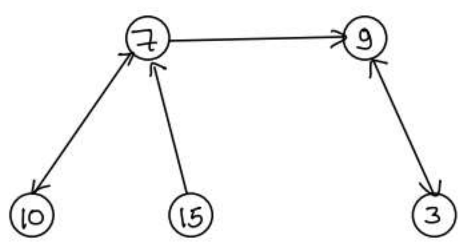
- ❖ **Puissance + ==** la clôture transitive de R → Algorithmes de Warshall



VII. Algorithme de Warshall

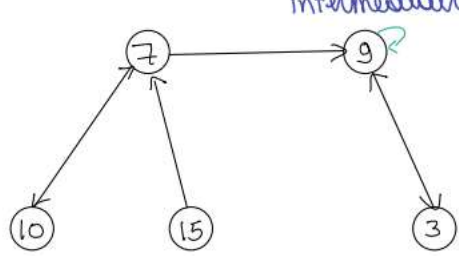
Théorie de Warshall : construire R^+ (= clôture transitive)

$R_0 = R$



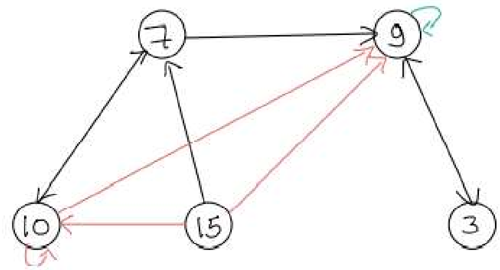
Supposons que l'ordre des commandes soit 3, 7, 9, 10, 15

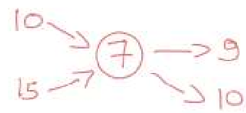
R_1 (sommet 3) : recopier R_0 + ajouter les "raccourcis" de tt les chemins de longueur 2 dans R_0 ayant 3 co sommet intermédiaire



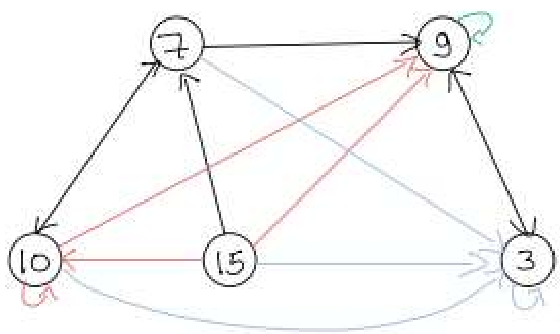
9 → 3 → 9

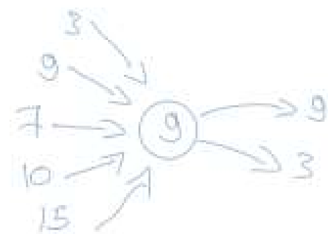
R_2 (sommet 7) : recopier R_1 + ajouter les "raccourcis" de H les chemins de longueur 2 dans R_1 ayant 7 c̄o sommet intermédiaire



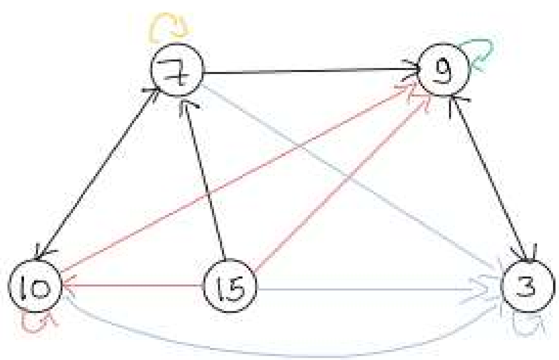


R_3 (sommet 9)





R_4 (sommet 10)



R_5 (sommet 15) $R_5 = R_4$ car aucune flèche arrive à 15.

2. LES RELATIONS D'ORDRE

I. Définitions & représentation

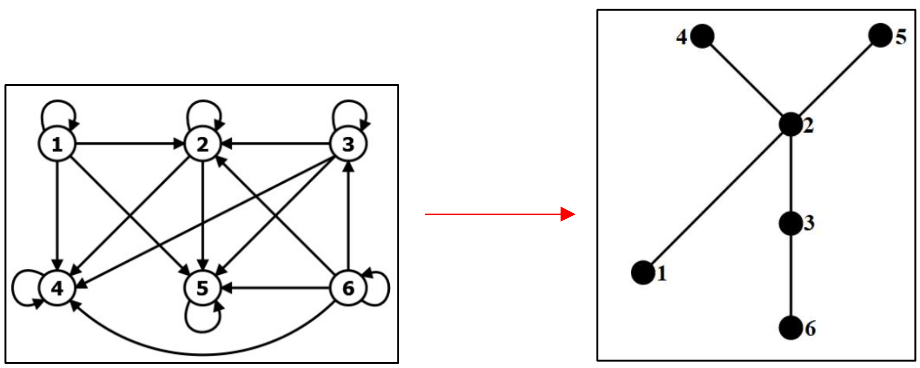
- **Définition ==** Une relation d'ordre DOIT être à la fois **Réflexive – Antisymétrique – Transitive**
- **L'ordre Partiel ==** Il existe 2 éléments qui ne sont pas comparables.
- **L'ordre Total ==** Tout élément est comparable avec tout autre élément.

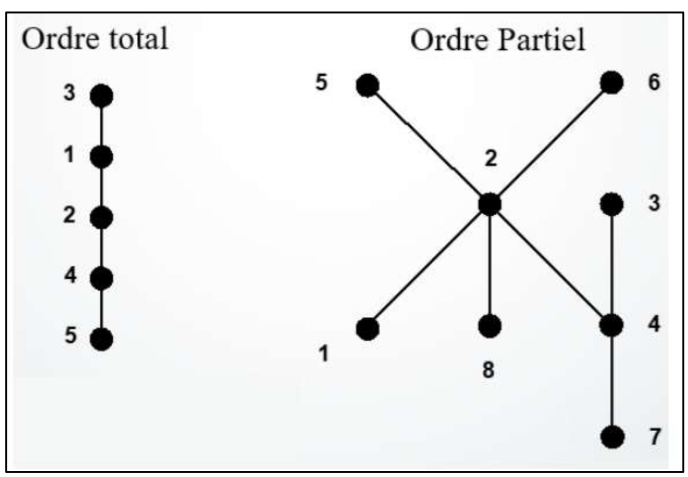
Exemples :

1. La relation "est multiple de" sur $\mathbb{N}$ est un ordre partiel. Par exemple, 2 n'est pas multiple de 3 et 3 n'est pas multiple de 2. Il existe donc deux éléments non comparables.
2. L'ordre alphabétique est un ordre total sur les lettres $\{\text{'a'}, \text{'b'}, \cdots, \text{'z'}\}$. En effet, si on prend deux lettres différentes, il y en aura toujours une qui sera avant l'autre dans l'ordre alphabétique.

- **Diagramme de Hasse - Pour les relations d'ordre**

1. Supprimer les boucles
2. Supprimer les flèches récupérables par une clôture transitive
3. On oriente les flèches vers le haut. /!\ Y aura que des flèches simples
4. On représente les sommets par des points (==convention de Hasse) & supprimer la direction des flèches.

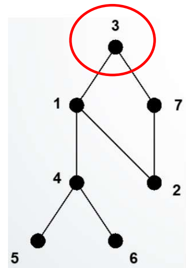




II. Vocabulaire dans un ensemble ordonné

❖ **Maximum *max(E)*** : Un SEUL élément supérieur à tous les autres.

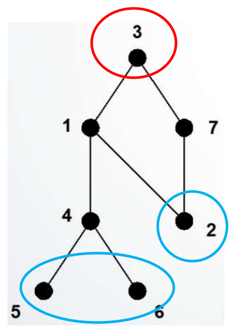
❖ **Minimum *min(E)*** : Un SEUL élément inférieur à tous les autres.



Pas de minimum ici car il n'existe pas 1 SEUL et UNIQUE élément inférieur aux autres.

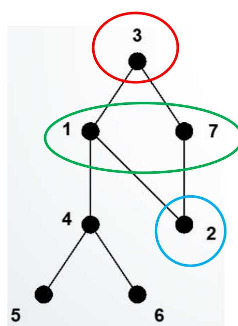
❖ **Maximal/Maximaux** : S'il n'existe pas d'éléments qui lui est supérieur
→ Le/Les éléments supérieurs aux autres.

❖ **Minimal/Minimaux** : S'il n'existe pas d'éléments qui lui est inférieur → Le ou Les éléments inférieur aux autres.



❖ **Majorant** *Major(B)*: Tout élément de E supérieur ou égal à chacun des éléments de B.

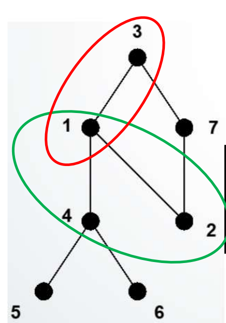
❖ **Minorant** *Minor(B)*: Tout élément de E supérieur ou égal à chacun des éléments de B.



Prenons comme exemple B={1,7}

Major(B) = {1,3} car ce sont les seuls éléments qui sont >= à 1

Minor(B) = {}. En effet, 66 est inférieur à 4 et 1 mais n'est pas comparable à 2. Il en est de même pour 5.



Prenons un autre exemple B={1,4,2}

❖ **Supremum *sup(B)*** **:** S'il existe, **LE** minimum de l'ensemble des majorants de B.
❖ **Infimum *inf(B)*** **:** S'il existe, **LE** maximum de l'ensemble des majorants de B.

III. Treillis

❖ **Définitions :** Un ensemble est appelé treillis SI Toute paire d'éléments a 1 supremum et 1 infimum.

- ✓ **Supremum** de 2 naturels == le PPCM de n et m strictement positifs. *(Le + petit naturel divisible par n et m).*
- ✓ **Infimum** de 2 naturels == le PGCD de n et m strictement positifs. *(Le + grand entier qui divise n et m).*

❖ **3 astuces pour déterminer si un ensemble ordonné est un trellis**

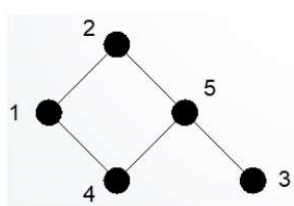
1. Un ensemble totalement ordonné *(type ordre total)* → Un trellis.
2. Si l'ensemble à plusieurs maximaux/minimaux → PAS un trellis
3. Les seuls paires d'éléments à vérifier sont les paires d'éléments non comparables. (Ceux qui ne sont pas liés par des flèches).

IV. Tri topologique

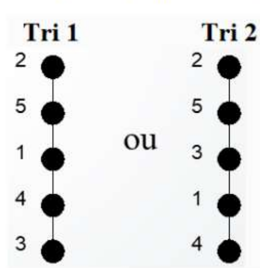
❖ **Construction d'un tri topologique**

- ✓ De bas en haut en sélectionnant, à chaque étape, un élément minimal parmi ceux-restants.
- ✓ Il existe généralement plusieurs tri topologiques possibles pour un ordre partiel.
- ✓ Pas de formules permettant de calculer le nbre de tris topologiques possibles.

Soit l'ensemble $E = \{1, 2, 3, 4, 5\}$ et l'ordre $\leq$ sur cet ensemble dont le diagramme de Hasse est le suivant

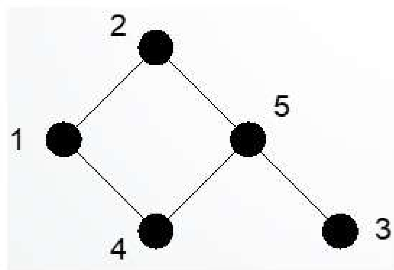


Alors les ordres totaux ci-dessous sont des tris topologiques de $\leq$

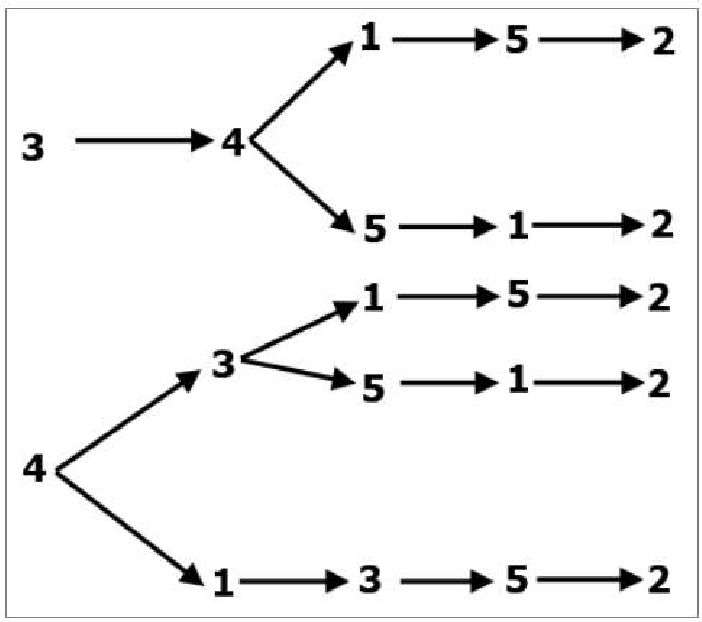


- **Nombre de tris topologiques :** Technique permettant de déterminer le nbre de tris topologiques d'un ordre ➔ Faire l'arbre des tris possibles.

Soit l'ordre de la section 4.2 dont le diagramme de Hasse est le suivant :



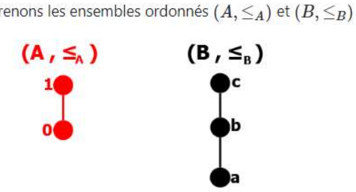
Alors l'arbre des tris topologiques possibles de cet ordre est

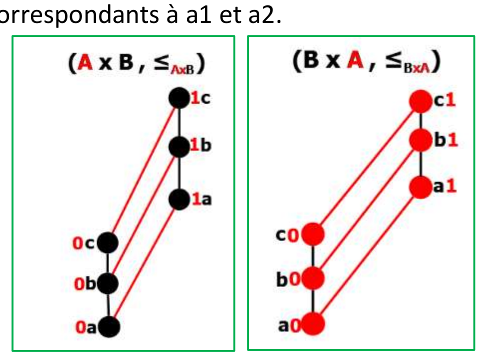


V. Produit de relations d'ordre

- **Produit Classique**
  - ✓ Dessiner autant de copies de B que d'éléments dans A.
  - ✓ Pour chaque arête d'un élément a1 vers un élément a2 sur le diagramme de Hasse de A, on relie les éléments correspondants des copies de B correspondants à a1 et a2.

Prenons les ensembles ordonnés $(A, \leq_A)$ et $(B, \leq_B)$

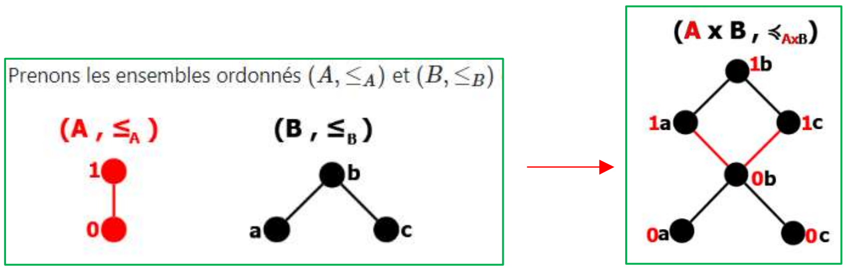


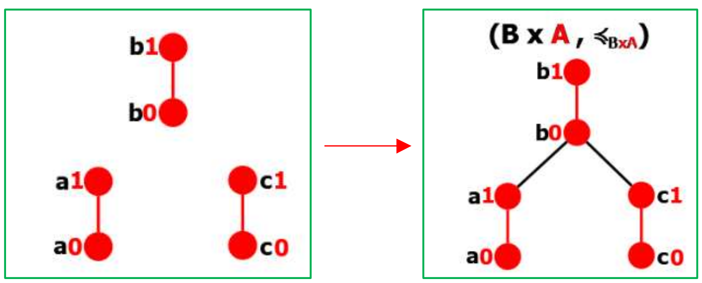


❖ Produit Lexicographique

✓ Dessiner autant de copie de B que d'éléments dans A

✓ Pour chaque arêtes d'un élément a1 vers un élément a2 sur le diagramme de Hasse de A, on relie les minimaux de la copie de B correspondant à a2 avec les maximaux de la copie de B correspondant à a1.





❖ Conclusion pour construire …

1. Produit classique $(A \times B, \leq_{A\times B})$:
   - Au temps de copie de $(B, \leq B)$ que d'éléments dans $(A, \leq_A)$.
   - S'il y une arête de a_1 vers a_2 sur le diagramme de Hasse de $(A, \leq_A)$ (autrement dit si $a_1 \leq_A a_2$), alors on relie les éléments correspondants des copies de $(B, \leq_B)$ correspondant à a_1 et a_2.
2. Produit lexicographique : $(A \times B, \preccurlyeq_{A\times B})$
   - Au temps de copie de $(B, \leq B)$ que d'éléments dans $(A, \leq_A)$.
   - S'il y une arête de a_1 vers a_2 sur le diagramme de Hasse de $(A, \leq_A)$ (autrement dit si $a_1 \leq_A a_2$), alors on relie les minimaux de la copie de $(B, \leq_B)$ correspondant à a_2 avec les maximaux de la copie de $(B, \leq_B)$ correspondant à a_1

Pas trop capté

3. LES RELATIONS D'ÉQUIVALENCE

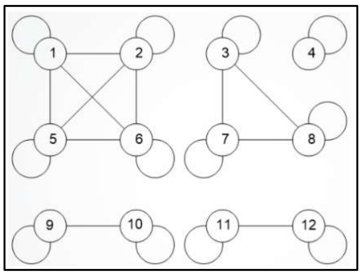
I. Définition, Classes et Quotient

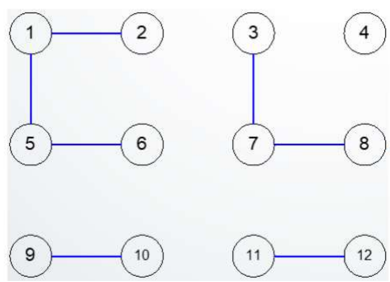
❖ **Définition :** Une relation d'équivalence $(\sim)$ DOIT être à la fois **Réflexive – <u>Symétrique</u> – Transitive.**

❖ **Diagramme de classe – Pour les relations d'équivalence**

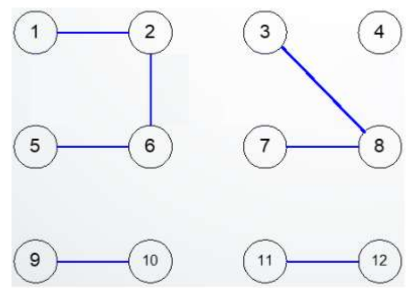
Soit $\sim$ une relation d'équivalence sur l'ensemble $E=\{1,2,3,4,5,6,7,8,9,10,11,12\}$ dont voici le graphe.

- On retire le sens des flèches.
- Connectant avec **un minimum d'arêtes** les éléments d'une même classe. *(On sous-entend qu'une même classe, les éléments sont tous connecter donc pas besoin de toute les flèches).*
- Les sommets de la relation d'équivalence apparaissent groupés en « classes » séparées. Donc ici, y a 5 classes.
- Dans une même classe, tous les éléments ont une connexion avec les autres éléments.





OU



→Un élément <u>par classe</u> sera représenté pour un **représentant de la classe.** (*comme le délégué quoi).*

→**Le quotient** de E par la relation d'équivalence, noté $E/\sim$, **la partition** de E en classe = $E/\sim=\{[x]\,|\,|x\in E\}$. Pour dire résumé == l'ensemble de cette classe.

Ex :

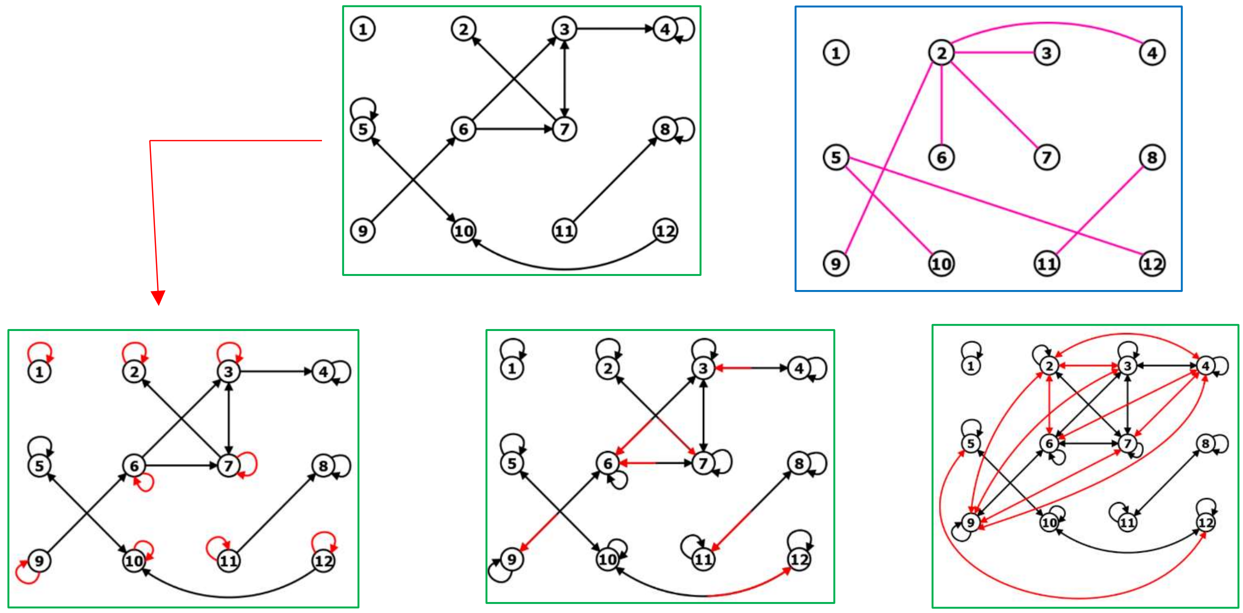
- $[1]=\{1,2,5,6\}$
- $[3]=\{3,7,8\}$
- $[4]=\{4\}$
- $[9]=\{9,10\}$
- $[11]=\{11,12\}$
- On a donc choisi $1,\ 3,\ 4,\ 9$ et 11 comme représentants des différentes classes d'équivalence
- Le quotient de E par la relation d'équivalence $\sim$ est donc

$$E/\sim=\Big\{[1],\ [3],\ [4],\ [9],\ [11]\Big\}=\Big\{\{1,2,5,6\},\{3,7,8\},\{4\},\{9,10\},\{11,12\}\Big\}$$

II. Opérations sur les relations d’équivalence

❖ **Clôture équivalente :** Pour obtenir cette clôture, on va :

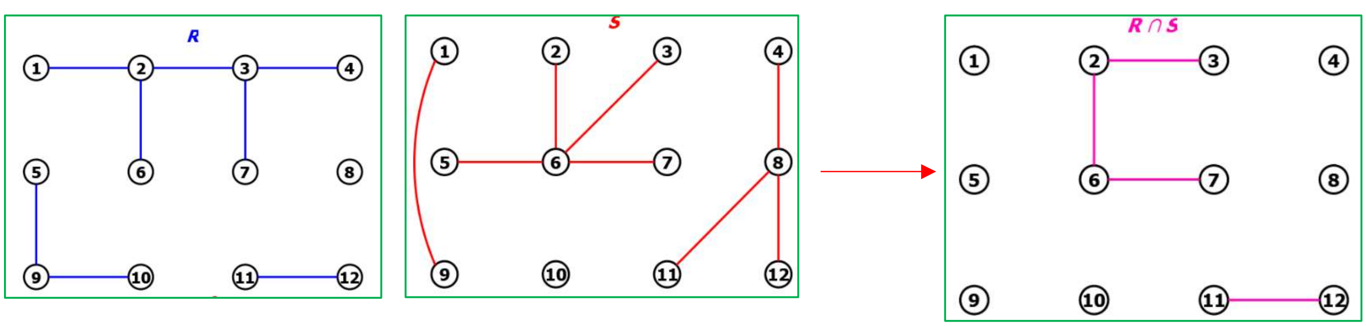
1. Ajouter toutes les boucles, pour avoir une clôture réflexive.
2. Rajouter un sens de flèches aux flèches simple, pour avoir une clôture symétrique.
3. Rajouter des flèches, pour avoir une clôture transitive.
4. Faire le diagramme de classe de la obtenue en 3 étapes.



1 2 3

/!\ Faire d’abord la clôture symétrique avant la clôture transitive.

❖ **Intersection d’équivalence.** R∩S



Explication :

- On voit que dans la Relation R, les éléments 1-2-3-4-6-7 font partie de la même classe. Et que dans la Relation S, les éléments 2-3-5-6-7 font partie de la même classe. Donc les éléments en commun sont 2-3-6-7.

❖ **L’union d’équivalence.**

- Piège. L’union de deux équivalences n’est pas une équivalence.

4. LES MATRICES

I. Introduction

❖ **Définition :** Une matrice m x n est un tableau $m \cdot n$ éléments disposées en m lignes et n colonnes.

- ✓ D'abord on lit la ligne puis la colonne
- ✓ L'élément se trouvant au croisement de la **ième ligne** et de la **jème colonne** se notera a_{ij}.

❖ **Matrice carrée :** Matrice avec le même nbre de lignes que de colonne.

II. Addition / Multiplication Matricielle

❖ **Addition :** Additionner des éléments de mêmes indices.

- ✓ Condition : Avoir la même dimension pour les 2 matrices.

$$\begin{pmatrix} 5 & 2 & 4 & 1 \\ 3 & 0 & 2 & 3 \\ 2 & 1 & 5 & 0 \end{pmatrix} + \begin{pmatrix} 1 & 3 & 0 & 2 \\ 5 & 2 & 2 & 4 \\ 0 & 1 & 2 & 0 \end{pmatrix} = \begin{pmatrix} 5+1 & 2+3 & 4+0 & 1+2 \\ 3+5 & 0+2 & 2+2 & 3+4 \\ 2+0 & 1+1 & 5+2 & 0+0 \end{pmatrix} = \begin{pmatrix} 6 & 5 & 4 & 3 \\ 8 & 2 & 4 & 7 \\ 2 & 2 & 7 & 0 \end{pmatrix}$$

❖ **Multiplication :** C'est dur… faut trop réfléchir.. cmt c'est casse couilles

- ✓ /!\ Multiplier une matrice AxB ≠ que multiplier une matrice BxA
- ✓ Condition : Le nbre de colonne de la 1ière matrice == le nbre de ligne de la seconde matrice.
- ✓ La dimension de la matrice résultante = Le nbre de ligne de la 1ère x le nbre de colonne de la seconde.

$$\begin{pmatrix} 5 & 2 & 4 & 1 \\ 3 & 0 & 2 & 3 \\ 2 & 1 & 5 & 0 \end{pmatrix} \cdot \begin{pmatrix} 2 \\ 1 \\ 3 \\ 4 \end{pmatrix} = \begin{pmatrix} 5\cdot 2+2\cdot 1+4\cdot 3+1\cdot 4 \\ 3\cdot 2+0\cdot 1+2\cdot 3+3\cdot 4 \\ 2\cdot 2+1\cdot 1+5\cdot 3+0\cdot 4 \end{pmatrix} = \begin{pmatrix} 28 \\ 24 \\ 20 \end{pmatrix}$$

Soit les matrices $A = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$ et $B = \begin{pmatrix} 2 & 2 \\ -2 & -2 \end{pmatrix}$ alors

$$A \cdot B = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 2 & 2 \\ -2 & -2 \end{pmatrix} = \begin{pmatrix} 1\cdot 2+1\cdot(-2) & 1\cdot 2+1\cdot(-2) \\ 1\cdot 2+1\cdot(-2) & 1\cdot 2+1\cdot(-2) \end{pmatrix}$$
$$= \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} (= O_2 \text{ Matrice nulle d'ordre 2})$$

$$B \cdot A = \begin{pmatrix} 2 & 2 \\ -2 & -2 \end{pmatrix} \cdot \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} 2\cdot 1+2\cdot 1 & 2\cdot 1+2\cdot 1 \\ (-2)\cdot 1+(-2)\cdot 1 & (-2)\cdot 1+(-2)\cdot 1 \end{pmatrix}$$
$$= \begin{pmatrix} 4 & 4 \\ -4 & -4 \end{pmatrix}$$

Quelle sera le résultat de la multiplication matricielle ci-dessous ?

$$\begin{pmatrix} -2 & 3 & 0 \\ 2 & -1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 0 & 2 \\ -3 & 1 \\ 1 & 3 \end{pmatrix}$$

Votre réponse :

$$\begin{pmatrix} -9 & -1 \\ 4 & 6 \end{pmatrix}$$

❖ **Multiplication d'une matrice par un réel.** *C'est + facile*

Matriciellement

$$2 \cdot \begin{pmatrix} 2 \\ 1 \\ 3 \\ 4 \end{pmatrix} = \begin{pmatrix} 2\cdot 2 \\ 2\cdot 1 \\ 2\cdot 3 \\ 2\cdot 4 \end{pmatrix} = \begin{pmatrix} 4 \\ 2 \\ 6 \\ 8 \end{pmatrix}$$

III. Transposition d'une matrice

- ✓ La transposition d'une matrice *m x n* == matrice *n x m*
- ✓ Les lignes → Colonnes
- ✓ Les colonnes → Lignes
- ✓ La transposée d'une matrice A se note A^T

Matriciellement

$$C = \begin{pmatrix} 5 & 2 & 4 & 1 \\ 3 & 0 & 2 & 3 \\ 2 & 1 & 5 & 0 \end{pmatrix} \Longrightarrow C^T = \begin{pmatrix} 5 & 3 & 2 \\ 2 & 0 & 1 \\ 4 & 2 & 5 \\ 1 & 3 & 0 \end{pmatrix}$$

IV. Modélisation matricielle de systèmes linéaires

❖ **En gros :** Un système de m équations linéaires à n inconnues s'écrit matriciellement : ***A . X = B***

- ✓ *A* = la matrice *m x n* des coefficients
- ✓ *X* = la matrice colonne *n x 1* des inconnues
- ✓ *B* = la matrice colonne *m x 1* des termes indépendants
- ✓ *Vasy un exemple ce sera + clair*

Etape 1 : Mise en équation

Posons $\begin{cases} x \text{ le nombre d'articles de type } A \\ y \text{ le nombre d'articles de type } B \end{cases}$

Alors

- achat de 10 articles de 2 types différents *A* et B $\Longrightarrow$ $x + y = 10$
- Les articles de type *A* coûtent 5€, ceux de type *B* coûtent 6€ et le **client a payé un total de 56€** $\Longrightarrow$ $5x + 6y = \mathbf{56}$

On a donc le système linéaire de 2 équations à 2 inconnues suivant : $\begin{cases} x & + & y & = & 10 \\ 5x & + & 6y & = & 56 \end{cases}$

Etape 2 : Modélisation à l'aide de matrice

Le système peut être réécrit de la manière suivante

$$\begin{cases} 1 \cdot x & + & 1 \cdot y & = & 10 \\ 5 \cdot x & + & 6 \cdot y & = & 56 \end{cases}$$

On remarque que les inconnues x et y interviennent dans les deux équations.

Si on les regroupe dans une matrice 2×1, alors on peut regrouper les coefficients multiplicateur dans une matrice 2×2

et le membre de gauche des équations peut s'écrire

$$\begin{pmatrix} 1 & 1 \\ 5 & 6 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix}$$

En effet, si on effectue la multiplication matricielle on obtient

$$\begin{pmatrix} 1 & 1 \\ 5 & 6 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 1 \cdot x & + & 1 \cdot y \\ 5 \cdot x & + & 6 \cdot y \end{pmatrix}$$

Enfin, si on regroupe les termes indépendant se trouvant dans les membres de droite des équations, le système s'écrit matriciellement

$$\begin{pmatrix} 1 & 1 \\ 5 & 6 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 10 \\ 56 \end{pmatrix}$$

V. L'évolution de populations

❖ **En gros :**

- ✓ A chaque période, le nbre d'individus dans chaque population change en focntion des populations de la période précédente.
- ✓ Matriciellement, ça se traduit par

$$P^{(n+1)} = M \cdot P^{(n)}$$
$$\Downarrow$$
$$\begin{pmatrix} p_1^{(n+1)} \\ p_2^{(n+1)} \\ \vdots \\ p_i^{(n+1)} \end{pmatrix} = \begin{pmatrix} m_{11} & m_{12} & \cdots & m_{1i} \\ m_{21} & m_{22} & \cdots & m_{2i} \\ \vdots & \vdots & \vdots & \vdots \\ m_{i1} & m_{i2} & \cdots & m_{ii} \end{pmatrix} \cdot \begin{pmatrix} p_1^{(n)} \\ p_2^{(n)} \\ \vdots \\ p_i^{(n)} \end{pmatrix}$$

- ✓ Une distribution stable == matrice des populations ***P*** telle que ***M . P = P***
- ✓ *On va copier-coller un exemple ce sera + facile pour capter*

Deux comptes financiers *a* et *b* sont manipulés de la façon suivante : à la fin de chaque année

- 10% du montant du compte *a* est versé sur le compte *b* ;
- 40% du montant du compte *b* est versé sur le compte *a*.

Au départ il y a 11000€ : 1000€ sur le compte *a* et 10000€ sur le compte *b*.

a) Exprimez par un système d'équations les mouvements d'argent sur ces 2 comptes en 1 an.

b) Exprimez ces mouvements sous forme matricielle.

c) Combien d'argent y aura-t-il sur ces comptes après 1 an et après 2 ans ?

d) Sur le long terme, sur quel compte y aura-t-il le plus d'argent ?

e) Y a-t-il une répartition stable ? (qui ne change pas)

<u>Résolution :</u>

a) Soient $a^{(n)}$ et $b^{(n)}$ les montants sur les comptes *a* et *b* au début de l'an *n*. Si à la fin de l'année,

- 10% du montant du compte *a* est versé sur le compte *b* ;
- 40% du montant du compte *b* est versé sur le compte *a*.

alors les montants sur les comptes *a* et *b* au début de l'année $n+1$ sont

$$\begin{cases} a^{(n+1)} = 0.90 \cdot a^{(n)} + 0.40 \cdot b^{(n)} \\ b^{(n+1)} = 0.10 \cdot a^{(n)} + 0.60 \cdot b^{(n)} \end{cases}$$

b) Matriciellement on a alors,

$$\begin{pmatrix} a^{(n+1)} \\ b^{(n+1)} \end{pmatrix} = \begin{pmatrix} 0.90 & 0.40 \\ 0.10 & 0.60 \end{pmatrix} \cdot \begin{pmatrix} a^{(n)} \\ b^{(n)} \end{pmatrix}.$$

/!\ Les parenthèses pour $a^{(n)}$ et $b^{(n)}$ est HYPER IMORTANTE.

Ça signifie *a* au temps n. Si on met pas les parenthèses ça voudrait dire *a* exposant n et c'est TRES DIFERENTS.

c) On a $\begin{pmatrix} a^{(0)} \\ b^{(0)} \end{pmatrix} = \begin{pmatrix} 1000 \\ 10000 \end{pmatrix}$.

Après 1 et 2 ans on a

$$\begin{aligned}
\begin{pmatrix} a^{(1)} \\ b^{(1)} \end{pmatrix} &= \begin{pmatrix} 0.90 & 0.40 \\ 0.10 & 0.60 \end{pmatrix} \cdot \begin{pmatrix} a^{(0)} \\ b^{(0)} \end{pmatrix} = \begin{pmatrix} 0.90 & 0.40 \\ 0.10 & 0.60 \end{pmatrix} \cdot \begin{pmatrix} 1000 \\ 10000 \end{pmatrix} \\
&= \begin{pmatrix} 0.90 \cdot 1000 + 0.40 \cdot 10000 \\ 0.10 \cdot 1000 + 0.60 \cdot 10000 \end{pmatrix} \Longrightarrow \begin{pmatrix} a^{(1)} \\ b^{(1)} \end{pmatrix} = \begin{pmatrix} 4900 \\ 6100 \end{pmatrix}. \\
\begin{pmatrix} a^{(2)} \\ b^{(2)} \end{pmatrix} &= \begin{pmatrix} 0.90 & 0.40 \\ 0.10 & 0.60 \end{pmatrix} \cdot \begin{pmatrix} a^{(1)} \\ b^{(1)} \end{pmatrix} = \begin{pmatrix} 0.90 & 0.40 \\ 0.10 & 0.60 \end{pmatrix} \cdot \begin{pmatrix} 4900 \\ 6100 \end{pmatrix} \\
&= \begin{pmatrix} 0.90 \cdot 4900 + 0.40 \cdot 6100 \\ 0.10 \cdot 4900 + 0.60 \cdot 6100 \end{pmatrix} \Longrightarrow \begin{pmatrix} a^{(2)} \\ b^{(2)} \end{pmatrix} = \begin{pmatrix} 6850 \\ 4150 \end{pmatrix}.
\end{aligned}$$

Remarque :

$$\begin{aligned}
\begin{pmatrix} a^{(2)} \\ b^{(2)} \end{pmatrix} &= \begin{pmatrix} 0.90 & 0.40 \\ 0.10 & 0.60 \end{pmatrix} \cdot \begin{pmatrix} a^{(1)} \\ b^{(1)} \end{pmatrix} \\
&= \begin{pmatrix} 0.90 & 0.40 \\ 0.10 & 0.60 \end{pmatrix} \cdot \begin{pmatrix} 0.90 & 0.40 \\ 0.10 & 0.60 \end{pmatrix} \cdot \begin{pmatrix} a^{(0)} \\ b^{(0)} \end{pmatrix} \\
&= \begin{pmatrix} 0.90 & 0.40 \\ 0.10 & 0.60 \end{pmatrix}^2 \cdot \begin{pmatrix} a^{(0)} \\ b^{(0)} \end{pmatrix}
\end{aligned}$$

d) Il y a de plus en plus d'argent sur le compte *a* et de moins en moins sur le compte *b*.
Cela se stabilise avec 8800€ sur le compte *a* et 2200€ sur le compte *b*.

On remarque que si on veut calculer à l'année 2 par exemple, il suffit de multiplier ce que j'avais à la fin de la 1ère année. Et pour la 1ère année, on peut constater que j'ai multiplié par ce que j'avais au début de l'année 0.

Pour le d, le prof a utiliser un logiciel MatLab pour calculer.

5. Systèmes d'équations linéaires

I. Méthode de Gauss

❖ **Étape :**

1. Écriture de la matrice du système : (*A* | *B*)
2. Echelonnage de la matrice du système
3. Écriture du système échelonné à partir de la matrice échelonnée
4. Dans chaque équat° en commençant par la dernière :
   - Remplacer la variable de tête par sa valeur dans les équations précédentes.
   - Réisoler les variables de tête des équations précédentes afin de trouver leur valeur.
5. Réécrire l'ensemble des solutions si bien évidemment toutes les valeurs sont connues.
   Résumé d'un exemple p.22

II. Matrice échelonnée

❖ **Définition :** Une matrice échelonnée == matrice dont le nbre de 0 en début de ligne ↗ à chaque ligne qui n'est pas entièrement composée. *En gros ça doit former un escalier, et en dessous des escaliers à chaque fois qu'on passe d'une ligne y a que des 0. Sauf si les dernière lignes sont que des zéros, y a po moyen d'augmenter.*

Ex : Ceux-ci sont des matrices échelonnées

$$\begin{pmatrix} 1 & 3 & 3 & 5 & 6 \\ 0 & -1 & 2 & 5 & -3 \\ 0 & 0 & 0 & 4 & 5 \\ 0 & 0 & 0 & 0 & 5 \end{pmatrix} \qquad \begin{pmatrix} 4 & -3 & 2 & 5 & 7 \\ 0 & -1 & 2 & 5 & -3 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Ex : Matrices NON échelonnées

$$\begin{pmatrix} -2 & 3 & 2 & 5 & 7 \\ 0 & -1 & 2 & 5 & -3 \\ 0 & 0 & 4 & 5 & 6 \\ 0 & 0 & 7 & -5 & 3 \end{pmatrix} \qquad \begin{pmatrix} -2 & 3 & 2 & 5 & 7 \\ 0 & -1 & 2 & 5 & -3 \\ 0 & 4 & 5 & 6 & 7 \\ 0 & 0 & 7 & -5 & 3 \end{pmatrix}$$

III. Échelonnage

❖ **Cmt échelonner une matrice ?**

Prenons un exemple :

Trouvez toutes les solutions du système

$$\begin{cases} x_1 + 2x_2 + 3x_3 - x_4 - x_5 = 2 \\ 2x_1 + x_2 + x_4 + x_5 = 1 \\ 3x_1 - 2x_3 + 5x_4 + 4x_5 = 3 \end{cases}$$

Matriciellement :

$$\left(\begin{array}{ccccc|c} 1 & 2 & 3 & -1 & -1 & 2 \\ 2 & 1 & 0 & 1 & 1 & 1 \\ 3 & 0 & -2 & 5 & 4 & 3 \end{array}\right)$$

Échelonnons cette matrice.

Jusqu'ici, rien de compliquer

1. On va essayer de trouver une ligne dont le premier élément vaut 1. Ça facilitera bcppp les choses, ça évite les fractions. Ici c'est parfait, le 1er élément de la 1ère ligne vaut 1.

- Si par exemple, dans une matrice, le 1er élément de la 3ième ligne seulement, c'est pas grave, on inverse les lignes oklm.

2. Après on regarde le 1er élément de la 2ème ligne. On va faire en sorte que ce premier élément vaut 0, en faisant la différent de la multiplication de la première ligne.

$$\left(\begin{array}{ccccc|c} 1 & 2 & 3 & -1 & -1 & 2 \\ \boxed{2} & 1 & 0 & 1 & 1 & 1 \\ 3 & 0 & -2 & 5 & 4 & 3 \end{array}\right) \xrightarrow[\ell_2 \leftarrow \ell_2 - 2\cdot\ell_1]{} \left(\begin{array}{ccccc|c} 1 & 2 & 3 & -1 & -1 & 2 \\ 0 & -3 & -6 & 3 & 3 & -3 \\ \boxed{3} & 0 & -2 & 5 & 4 & 3 \end{array}\right)$$

$$\xrightarrow[\ell_3 \leftarrow \ell_3 - 3\cdot\ell_1]{} \left(\begin{array}{ccccc|c} 1 & 2 & 3 & -1 & -1 & 2 \\ 0 & -3 & -6 & 3 & 3 & -3 \\ 0 & -6 & -11 & 8 & 7 & -3 \end{array}\right)$$

On va essayer de faire en sorte que le 2 deviennent 0 avec des maths.

2 = 2.1 OK ?

Donc on va multiplier la 1ère ligne par 2. Et après on soustrait le résultat à la 2ème ligne.

Donc on multiplie toute la 1ère ligne par 2, ce qui donnera :

2 4 6 -2 -2 | 4

Mnt on prend la 2ème ligne et on soustrait par ce qu'on a obtenu :

2-2 = 0 **1-4 =** -3 **0-6=** -6 **1-(-2) =** 3 etc, bref t'as capté le principe

3. Après on fait pareil pour la ligne suivante, en multipliant avec la 1ère ligne.
4. En gros : faut permuter les lignes pour avoir un pivot qui vaut 1. Et si tt les éléments d'une ligne ont 1 diviseur commun, on divise toute la ligne par ce diviseur. Et on EVITE les fractions.

1) Le 1er élément de la 1er ligne vaut 1 → ⇒ 1er pivot.
2) Mettons des 0 sous le pivot dans la première colonne :

$$\left(\begin{array}{ccccc|c} 1 & 2 & 3 & -1 & -1 & 2 \\ \boxed{2} & 1 & 0 & 1 & 1 & 1 \\ 3 & 0 & -2 & 5 & 4 & 3 \end{array}\right) \xrightarrow[\ell_2 \leftarrow \ell_2 - 2\cdot\ell_1]{} \left(\begin{array}{ccccc|c} 1 & 2 & 3 & -1 & -1 & 2 \\ 0 & -3 & -6 & 3 & 3 & -3 \\ \boxed{3} & 0 & -2 & 5 & 4 & 3 \end{array}\right)$$

$$\xrightarrow[\ell_3 \leftarrow \ell_3 - 3\cdot\ell_1]{} \left(\begin{array}{ccccc|c} 1 & 2 & 3 & -1 & -1 & 2 \\ 0 & -3 & -6 & 3 & 3 & -3 \\ 0 & -6 & -11 & 8 & 7 & -3 \end{array}\right)$$

3) Tous les éléments de la ligne 2 sont divisibles par -3 ⇒ divisons la ligne par -3 :

$$\left(\begin{array}{ccccc|c} 1 & 2 & 3 & -1 & -1 & 2 \\ 0 & -3 & -6 & 3 & 3 & -3 \\ 0 & -6 & -11 & 8 & 7 & -3 \end{array}\right) \xrightarrow[\ell_2 \leftarrow -\frac{1}{3}\cdot\ell_2]{} \left(\begin{array}{ccccc|c} 1 & 2 & 3 & -1 & -1 & 2 \\ 0 & 1 & 2 & -1 & -1 & 1 \\ 0 & -6 & -11 & 8 & 7 & -3 \end{array}\right)$$

4) Le 1er élément non nul de la ligne 2 est le second qui vaut 1 ⇒ pivot suivant.
5) Mettons un 0 sous le pivot dans la deuxième colonne :

$$\left(\begin{array}{ccccc|c} 1 & 2 & 3 & -1 & -1 & 2 \\ 0 & 1 & 2 & -1 & -1 & 1 \\ 0 & \boxed{-6} & -11 & 8 & 7 & -3 \end{array}\right) \xrightarrow[\ell_3 \leftarrow \ell_3 + 6\cdot\ell_2]{} \left(\begin{array}{ccccc|c} 1 & 2 & 3 & -1 & -1 & 2 \\ 0 & 1 & 2 & -1 & -1 & 1 \\ 0 & 0 & 1 & 2 & 1 & 3 \end{array}\right)$$

La matrice ainsi obtenue est échelonnée :

$$\left(\begin{array}{ccccc|c} 1 & 2 & 3 & -1 & -1 & 2 \\ 0 & 1 & 2 & -1 & -1 & 1 \\ 0 & 0 & 1 & 2 & 1 & 3 \end{array}\right)$$

- **Opération permises d'un échelonnage :**
  - Multiplier 1 ligne par un nbre ≠ de 0.
  - Permuter 2 lignes
  - Diviser toute une ligne par un même !!!! diviseur.

IV. Résolution du système

❖ **Résumé d'un exemple :**

Dans l'exemple précédent, nous avons obtenu la matrice échelonnée

$$\left(\begin{array}{ccccc|c} 1 & 2 & 3 & -1 & -1 & 2 \\ 0 & 1 & 2 & -1 & -1 & 1 \\ 0 & 0 & 1 & 2 & 1 & 3 \end{array}\right)$$

Résolution du système :

1) Repassons à l'écriture en système pour terminer la résolution :

$$\begin{cases} x_1 + 2x_2 + 3x_3 - x_4 - x_5 = 2 \\ \quad x_2 + 2x_3 - x_4 - x_5 = 1 \\ \quad\quad x_3 + 2x_4 + x_5 = 3 \end{cases}$$

2) Dans chaque équation, isolons la variable de tête (la première de l'équation) :

$$\begin{cases} x_1 = -2x_2 - 3x_3 + x_4 + x_5 + 2 \\ x_2 = -2x_3 + x_4 + x_5 + 1 \\ x_3 = -2x_4 - x_5 + 3 \end{cases}$$

3) Remplaçons la dernière variable, x_3, par son expression, dans les 2 équations précédentes :

$$\begin{cases} x_1 = -2x_2 - 3\cdot(-2x_4 - x_5 + 3) + x_4 + x_5 + 2 \\ x_2 = -2\cdot(-2x_4 - x_5 + 3) + x_4 + x_5 + 1 \\ x_3 = -2x_4 - x_5 + 3 \end{cases}$$

$$\Longrightarrow \begin{cases} x_1 = -2x_2 + 7x_4 + 4x_5 - 7 \\ x_2 = 5x_4 + 3x_5 - 5 \\ x_3 = -2x_4 - x_5 + 3 \end{cases}$$

4) Remplaçons la 2ème variable, x_2, par son expression, dans la 1ère équation :

$$\begin{cases} x_1 = -2\cdot(5x_4 + 3x_5 - 5) + 7x_4 + 4x_5 - 7 \\ x_2 = 5x_4 + 3x_5 - 5 \\ x_3 = -2x_4 - x_5 + 3 \end{cases}$$

$$\Longrightarrow \begin{cases} x_1 = -3x_4 - 2x_5 + 3 \\ x_2 = 5x_4 + 3x_5 - 5 \\ x_3 = -2x_4 - x_5 + 3 \end{cases}$$

/!\ à ne pas oublier les ()

5) Nous avons utilisé toutes les équations.
$\Rightarrow$ écriture de l'ensemble des solutions :

Solution :

$$S = \left\{ \left(-3x_4 - 2x_5 + 3,\ 5x_4 + 3x_5 - 5,\ -2x_4 - x_5 + 3,\ x_4,\ x_5 \right) \middle|\ x_4, x_5 \in \mathbb{R} \right\}.$$

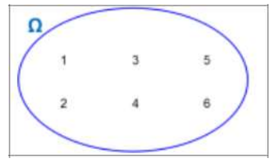
❖ **Un système peut être :**

$$\begin{cases} \text{déterminé} & \Longrightarrow & \text{1 seule solution} & \text{Ex : } S = \{(-2,3,7)\} \\ \text{indéterminé} & \Longrightarrow & \text{infinité de solutions} & \text{Ex : } S = \left\{ (x_3 - 1, 3 - x_3, x_3) \middle|\ x_3 \in \mathbb{R} \right\} \\ \text{impossible} & \Longrightarrow & \text{pas de solution} & \text{Ex : } S = \varnothing \end{cases}$$

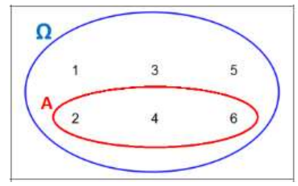
6. PROCESSUS DE MARKOV

I. Probabilité

- **Expérience aléatoire :** Expérience dont les résultats dépendent du hasard. *Ex : Jet d'un dé*
- **Éventualité :** Résultat possible de l'expérience. *Ex : 1,2,3,4,5,6*
- **Univers, *noté Ω* :** Ensemble des éventualités. *Ex :*



- **Évènement :** Sous-ensemble de l'univers. *Ex : A = « Le dé tombe sur un chiffre pair »*



- **Probabilité d'un évènement :** mesure de sa vraisemblance. *Ex : p(A) =* $\frac{1}{2}$
- **Distribution de probabilité :** fonction f(x) qui a 1 éventualité x associe sa probabilité

x	1	2	3	4	5	6
$f(x)$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$

Propriétés :

- 1 probabilité est TJS comprise entre 0 et 1.
- La somme de *la distribution de probabilité* tjs égale à 1.
- Équiprobabilité == ttes les éventualités ont la même probabilité.
  - ☞ $p(E) = \frac{Nbre\ de\ cas\ favorable\ à\ E}{Nbre\ de\ cas\ possible}$ *nbre de cas favorable== évènement ; nbre de cas possible == Univers*

II. Matrices colonne-stochastique

- **Caractéristiques :**

$$\begin{pmatrix} 0.50 & 0.2 & 0.7 \\ 0.25 & 0 & 0.2 \\ 0.25 & 0.8 & 0.1 \end{pmatrix}$$
$$\begin{array}{ccc} 1 & 1 & 1 \end{array}$$

  - ✓ Tous ses éléments sont positifs ou nuls.
  - ✓ Chacune des colonnes a la somme qui équivaut à 1.
  - ✓ Chacune de ses éléments représente une distribution de probabilité.

III. Processus de Markov homogènes à temps direct et à états finis

- **Définition ==** Modélisat° d'1 système évoluant entre plusieurs états au cours du temps permettant la prévision de l'état de celui-ci dans le futur.

Ex : On pense qu'un individu non endetté a une possibilité sur 3 de devenir endetté le jour suivant. Alors qu'un individu endetté a une possibilité sur 6 de régler ses dettes. **Comment peut-on modéliser cela ?**

1) Etats :

L'individu ne peut être que dans 2 états : Non endetté ⇒ Etat 1 ; Endetté ⇒ Etat 2

2) Probabilités de changement d'état :

Jours		Etats	
	t	*Etat 1*	*Etat 2*
	$t+1$	p(rester dans l'Etat 1) $= \frac{2}{3}$	p(passer dans l'Etat 1) $= \frac{1}{6}$
		p(passer dans l'Etat 2) $= \frac{1}{3}$	p(rester dans l'Etat 2) $= \frac{5}{6}$

⇒ A chaque état on associe la distribution de probabilités sur l'état au temps suivant

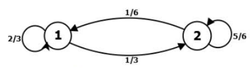
L'état 1

L'état 2

- L'état 1(un non endetté) a deux chances sur 3 de rester à l'état 1 (donc de rester non endetté) le jour suivant.
- Et il a une chance sur 3 de passer à l'état 2 (d'être endetté) le jour suivant.
- L'état 2 (un endetté) a une chance sur six de passer à l'état 1 (donc ne plus être endetté) le jour suivant.
- Et il a cinq chances sur six de rester à l'état 2 (donc de rester endetté) le jour suivant.

❖ **Représentation d'un processus de Markov**

3) Graphe :



⇒ Processus de Markov

/!\ La somme des probabilité des flèches sortant D'un même état doivent TJS valoir 1

❖ **Matrice de transition**

4) Matrice de transition :

$$P = \begin{pmatrix} \frac{2}{3} & \frac{1}{6} \\ \frac{1}{3} & \frac{5}{6} \end{pmatrix}$$

L'état 1 au t +1

L'état 2 au t + 1

⇒ Matrice carrée colonne-stochastique

❖ **Probabilité au jour suivant :** ½ car la probabilité d'être endetté est de ½ et la probabilité d'être endetté est de ½ aussi.

5) Supposons qu'au temps t, il y autant de chance que l'individu soit endetté que non endetté. Qu'en sera-t-il au temps suivant ?

Au temps t, on a $d^{(t)} = \begin{pmatrix} \frac{1}{2} \\ \frac{1}{2} \end{pmatrix}$. Alors au temps $t+1$ on a

$$\begin{aligned} d^{(t+1)} &= P \cdot d^{(t)} \\ &= \begin{pmatrix} \frac{2}{3} & \frac{1}{6} \\ \frac{1}{3} & \frac{5}{6} \end{pmatrix} \cdot \begin{pmatrix} \frac{1}{2} \\ \frac{1}{2} \end{pmatrix} = \begin{pmatrix} \frac{1}{3} + \frac{1}{12} \\ \frac{1}{6} + \frac{5}{12} \end{pmatrix} = \begin{pmatrix} \frac{5}{12} \\ \frac{7}{12} \end{pmatrix} \end{aligned}$$

⇒ Au temps $t+1$ l'individu a 7 chances sur 12 d'être endetté.

- 2/3 * ½ du haut = 1/3   1/6 * ½ du bas = 1/12
- 1/3 * ½ du haut = 1/6   5/6 * ½ du bas = 5/12

IV. Type d'états et périodicité

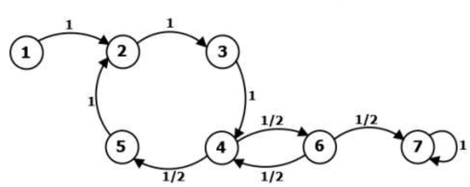
❖ **2 types d'états :**

✓ **Transient :** état dans lequel le système n'est pas certain de pouvoir revenir après l'avoir quitté.
Il existe un chemin qui ne le ramène pas vers lui-même.

✓ **Récurrent :** état dans lequel le système a tjs la possibilité de revenir après l'avoir quitté.
N'importe quel chemin le ramène vers lui.

Ex :

Exemple : Processus de Markov de l'exercice précédent :



transient : 1, 2, 3, 4, 5, 6
récurrent : 7

❖ **La période ==** PGCD des nbres possibles d'unités de temps, pour revenir dans l'état après l'avoir quitté.

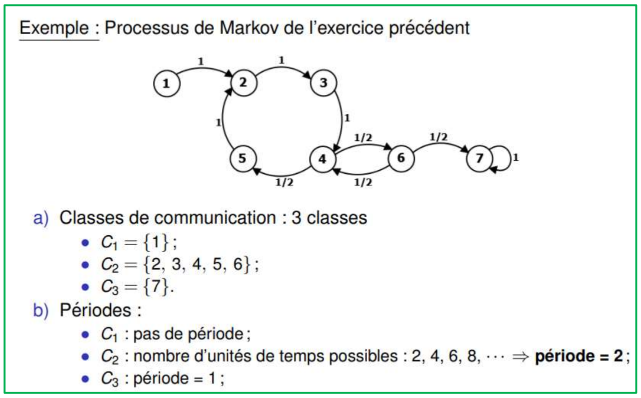
❖ **Classe de communication :** un ensemble d'états communicants entre eux.

✓ Un état e_1 communique avec un état e_2 s'il existe un chemin de e_1 vers e_2 **ET** un chemin de e_2 vers e_1.

Propriétés :

- Un processus de Markov qui n'a qu'1 classe de communicat° est dit **irréductible** ou **ergodique.**
- Ts les états d'1 même classe de communicat° ont le même type **et** la même période.
- 1 classe dont la période est égale à 1 est dite **apériodique.**

Ex :



V. Stabilité

❖ **Définition :** La distribut° stable donne la proport° de temps passé dans chaque état sur le long terme.

❖ **État transient & distribut° stable :** Si un état est de type transient alr dans la D.S. elle sera égale à 0.

❖ **Distribution stable ==** prévision d qui n'évolue pas avec le temps → P.d = d

❖ **Théorème d'existence :** Un processus de Markov possède tjs une distribut° stable.

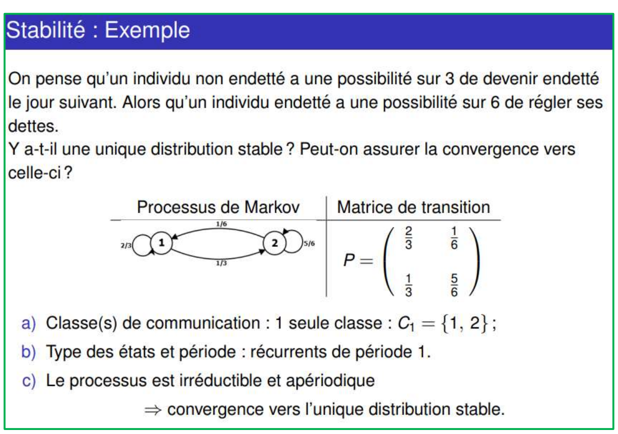
❖ **Théorème d'unicité :** Si un processus de Markov est irréductible alr il possède 1 et 1 seule D.S.

❖ **Théorème de convergence :** Si un processus de Markov est irréductible et apériodique.

Irréductible == quand le système n'a qu'1 classe.

Apériodique == qd un état possède 1 boucle vers lui-même.

Ex :



VI. Temps moyen de parcours t_i

❖ **Définition :** le nbre d'unités de temps qu'il faut en moyenne au système pour passer de l'état e_i à l'état e_j.

❖ **Méthode :** Si i=j → t_i = 0 ; Si i≠j → Faire le processus suivant :

❖ /!\ Il faut que le processus admette une seule classe d'états récurrents à laquelle appartient l'état d'arrivée.

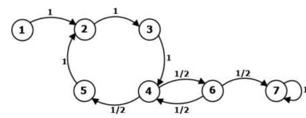
Temps de parcours

Exemple : le chien

Quel est, en moyenne, le nombre de déplacements nécessaires pour que le chien partant du sas correspondant à l'état 1 arrive au sas correspondant à l'état 7 ?

Résolution :

1) **Graphe :**



2) **Système :** soit t_i le temps moyen pour passer de l'état i à l'état 7. Alors

- Quand on est à l'état 1 on ne peut passer qu'à l'état 2
  Donc pour aller de l'état 1 à l'état 7 il faut
  1. Une unité de temps pour passer à l'état 2
  2. Le nombre d'unité de temps pour passer de l'état 2 à l'état 7 → t_2

  Donc $t_1 = 1 + t_2$
- $t_2 = 1 + t_3$
- $t_3 = 1 + t_4$.

- De l'état 4, on ne peut passer qu'à l'état 5 ou à l'état 6.
  Donc le temps pour passer de l'état 4 à l'état 7 peut être
  Soit $1 + t_5$
  Soit $1 + t_6$
  Or la probabilité de passer de l'état de 4 à 5 est de $\frac{1}{2}$
  et la probabilité de passer de l'état de 4 à 6 est de $\frac{1}{2}$
  On dira alors que

$$\begin{aligned} t_4 &= \frac{1}{2}\cdot(1+t_5)+\frac{1}{2}(1+t_6) = \frac{1}{2}+\frac{1}{2}\cdot t_5+\frac{1}{2}+\frac{1}{2}\cdot t_6 \\ \to t_4 &= 1+\frac{1}{2}\cdot t_5+\frac{1}{2}\cdot t_6 \end{aligned}$$

  Donc t_4 = une unité de temps (pour changer d'état) + la somme des temps de parcours depuis 5 et 6 pondérés par la probabilité de passer de l'état 4 à ces états.
- $t_5 = 1 + t_2$
- $t_6 = 1 + \frac{1}{2}\cdot t_4 + \frac{1}{2}\cdot t_7$
- $t_7 = 0$.

On a donc le système suivant :

$$\begin{cases} t_1 = 1+t_2 \\ t_2 = 1+t_3 \\ t_3 = 1+t_4 \\ t_4 = 1+\frac{1}{2}\cdot t_5+\frac{1}{2}\cdot t_6 \\ t_5 = 1+t_2 \\ t_6 = 1+\frac{1}{2}\cdot t_4 (\text{car } t_7=0) \end{cases} \Rightarrow \begin{cases} t_1-t_2 = 1 \\ t_2-t_3 = 1 \\ t_3-t_4 = 1 \\ t_4-\frac{1}{2}\cdot t_5-\frac{1}{2}\cdot t_6 = 1 \\ t_5-t_2 = 1 \\ t_6-\frac{1}{2}\cdot t_4 = 1 \end{cases}$$

On a donc un système que l'on va résoudre par la méthode de Gauss :

$$\begin{cases} t_1-t_2 = 1 \\ t_2-t_3 = 1 \\ t_3-t_4 = 1 \\ t_4-\frac{1}{2}\cdot t_5-\frac{1}{2}\cdot t_6 = 1 \\ t_5-t_2 = 1 \\ t_6-\frac{1}{2}\cdot t_4 = 1 \end{cases} \Rightarrow \left(\begin{array}{cccccc|c} 1 & -1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & -1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & -1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & -\frac{1}{2} & -\frac{1}{2} & 1 \\ 0 & -1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & -\frac{1}{2} & 0 & 1 & 1 \end{array}\right)$$

$$\xrightarrow[\substack{\ell_4 \leftarrow 2\cdot\ell_4 \\ \ell_6 \leftarrow 2\cdot\ell_6 \\ \ell_6 \leftrightarrow \ell_4 \\ \ell_5 \leftarrow \ell_5+\ell_2+\ell_3-\ell_4 \\ \ell_6 \leftarrow \ell_6+\ell_5+2\cdot\ell_4}]{} \left(\begin{array}{cccccc|c} 1 & -1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & -1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & -1 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1 & 0 & 2 & 2 \\ 0 & 0 & 0 & 0 & 1 & -2 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 7 \end{array}\right)$$

On a donc le système échelonné :

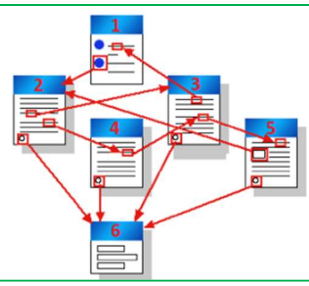
$$\begin{cases} t_1-t_2 = 1 \\ t_2-t_3 = 1 \\ t_3-t_4 = 1 \\ -t_4+2t_6 = 2 \\ t_5-2t_6 = 1 \\ t_6 = 7 \end{cases} \Rightarrow \begin{cases} t_1 = t_2+1 \\ t_2 = t_3+1 \\ t_3 = t_4+1 \\ -t_4 = -2t_6+2 \\ t_5 = 2t_6+1 \\ t_6 = 7 \end{cases} \Rightarrow \begin{cases} t_1 = 15 \\ t_2 = 14 \\ t_3 = 13 \\ t_4 = 12 \\ t_5 = 15 \\ t_6 = 7 \end{cases}$$

7. ALGORITHMES PAGERANK DE GOOGLE

❖ **But ?** Classer les sites par pertinence lors d'une recherche Google.

I. Algorithme

❖ **Faire une matrice adjacente :**



$$\begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 \end{pmatrix}$$

- ✓ En gros, mettre des 1, là où l'élément à une relation avec un autre élément.
- ✓ Ex : *mij* ,dont i = la ligne j = la colonne
- ✓ m_{32} , on va mettre un 1, car on voit dans le dessin, que la page 2 à une relation qui va vers la page 3-4-6.
- ✓ Et ainsi de suite…

❖ **Normalisation de la matrice adjacente :**

Equiprobabilité :

$$\begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 \end{pmatrix} \Rightarrow \begin{pmatrix} 0 & 0 & \frac{1}{3} & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & \frac{1}{2} & 0 \\ 0 & \frac{1}{3} & 0 & \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{3} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{3} & 0 & 0 & 0 \\ 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{2} & \frac{1}{2} & 0 \end{pmatrix}$$

- ✓ Notre but, c'est d'avoir une matrice colonne-stochastique (== la somme d'une colonne doit valoir 1).
- ✓ Donc on a qu'à faire $\frac{1}{le\ nbre\ d'élément=1}$
- ✓ Par exemple dans la 3^{ième} colonne, on a 3x1 donc 1/3 = $\frac{1}{3}$

❖ **Gestion des trous noirs :**

$$\begin{pmatrix} 0 & 0 & \frac{1}{3} & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & \frac{1}{2} & 0 \\ 0 & \frac{1}{3} & 0 & \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{3} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{3} & 0 & 0 & 0 \\ 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{2} & \frac{1}{2} & 0 \end{pmatrix} \Rightarrow \begin{pmatrix} 0 & 0 & \frac{1}{3} & 0 & 0 & \frac{1}{6} \\ 1 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{6} \\ 0 & \frac{1}{3} & 0 & \frac{1}{2} & 0 & \frac{1}{6} \\ 0 & \frac{1}{3} & 0 & 0 & 0 & \frac{1}{6} \\ 0 & 0 & \frac{1}{3} & 0 & 0 & \frac{1}{6} \\ 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{2} & \frac{1}{2} & \frac{1}{6} \end{pmatrix}$$

- ✓ Comme j'ai dit, notre but c'est d'avoir une matrice colonne-stochastique.
- ✓ Mais ici, on voit bien que notre dernière colonne est nul de chez nul.
- ✓ No panique, on a qu'à faire $\frac{1}{le\ nbre\ de\ pages/de\ lignes}$

❖ **Introduction de l'aléatoire :**

$$P = 0{,}85 \cdot \begin{pmatrix} 0 & 0 & \frac{1}{3} & 0 & 0 & \frac{1}{6} \\ 1 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{6} \\ 0 & \frac{1}{3} & 0 & \frac{1}{2} & 0 & \frac{1}{6} \\ 0 & \frac{1}{3} & 0 & 0 & 0 & \frac{1}{6} \\ 0 & 0 & \frac{1}{3} & 0 & 0 & \frac{1}{6} \\ 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{2} & \frac{1}{2} & \frac{1}{6} \end{pmatrix} + 0{,}15 \cdot \begin{pmatrix} \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \end{pmatrix}$$

0.85 et 0.15, pcq c'est comme ça, c'est les maths, c'est la règle.

La 1ère matrice colonne-stochastique = celle qu'on a résolu.

La 2ième matrice = tu remplace tout par $\frac{1}{le\ nbre\ de\ pages/de\ lignes}$.

❖ **Recherche de la distribution**

5) Recherche de la distribution :

a) Choix d'une distribution de départ $d^{(0)}$.

b) Calcul de $d^{(k+1)} = P \cdot d^{(k)}$ jusqu'à convergence.

⚠ Il faut normaliser la distribution $d^{(k)}$ à chaque itération !

Normalisation : la somme des éléments de $d^{(k)}$ doit être égale à 1 :

i) Calcul de la somme des éléments de $d^{(k)}$: $s = d_{11}^{(k)} + d_{21}^{(k)} + \cdots + d_{n1}^{(k)}$.

ii) - Normalisation : faire $d_{i1}^{(k)} = d_{i1}^{(k)}/s$ pour tous les éléments de $d^{(k)}$.

$$d^{(0)} = \begin{pmatrix} \frac{1}{6} \\ \frac{1}{6} \\ \frac{1}{6} \\ \frac{1}{6} \\ \frac{1}{6} \\ \frac{1}{6} \end{pmatrix} \xrightarrow{\text{85 itérations}} d^{(85)} = \begin{pmatrix} 0.112731386620343 \\ 0.207157903921133 \\ 0.174021178729935 \\ 0.122120125424516 \\ 0.112731386620343 \\ 0.271238018683729 \end{pmatrix}$$

- ✓ J'ai pas trop capé cette partie, mais alz on devras pas calculer nous-même. Et avec ce résultat, on peut classer les pages par ordre de pertinence : 6-2-3-4-5-1
- ✓ En gros :Tu vas créer ton d0 qui sera de type (nbreLignePx1) et qui sera equi probable et le multiplier a chaque itération avec la matrice P.

Etant donné que tu dois normaliser à chaque itération, il faudra faire la somme de ta ligne et ensuite diviser tous les éléments de la ligne par cette somme.
Le tout 85 fois !!!

8. LANGAGES FORMELS

I. Alphabet – Mots – Langages

❖ **Alphabet Σ ==** Ensemble fini non vide. Les éléments peuvent être de n'importe quelle nature.

- ✓ Ex : des caractères, des lettres, des mots-clés Java, des chiffres, …
- ✓ On appellera les éléments de l'alphabet **lettres** ou **symboles terminaux.**

❖ **Mots ==** Suite finie(éventuellement vide) de lettres.

- ✓ **La longueur *m*** d'un mot == le nbre de lettres qu'il contient (== Valeur Absolue).
- ✓ **Le mot vide *ϵ*** Seul mot de longueur 0. /!\ N'appartient pas à l'alphabet.
- ✓ Ex :

Si on a les deux mots $m = acba$ et $p = abc$ définit sur l'alphabet $\Sigma = \{a, b, c\}$, alors on aura :

1. $mp = acbaabc$
2. $p^2 = abcabc$
3. $p^0 = \varepsilon$
4. $p\varepsilon = abc = \varepsilon p$
5. $pm^2p = abcacbaacbaabc$

❖ **Langages ==** Tout ensemble de mots sur Σ.

Sur l'alphabet $\Sigma = \{a, b, c\}$, on peut définir les langages

1. $L_1 = \{abc,\ bca,\ a,\ aaa\}$
2. $L_2 = \{aa,\ bca,\ abc,\ (bbc)^3,\ \varepsilon\}$
3. $L_3 = \text{ensemble des mots contenant au moins deux } a$
4. $\{\varepsilon\}$ est un langage particulier sur Σ $\rightarrow$ il contient 1 mot.
5. $\varnothing$ est le plus petit langage sur Σ $\rightarrow$ il ne contient aucun mot.
6. $\cdots$

Il existe une infinité de langages sur un alphabet Σ.

❖ **Opération sur les langages : ∪ , ∩ , −**

❖ **Opérateur de concaténation : ●** /!\ *ϵ* Neutre pour la concaténat°.

/!\ Un ensemble de peux pas contenir 2x le même éléments

Soit l'alphabet $\Sigma = \{a,\ c\}$ et les deux langages sur Σ

- $L_1 = \{ac,\ a\}$
- $L_2 = \{a,\ ca\}$

Donnez en extension le langage $L_1 \bullet L_2$.

Ne mettez pas de blanc et n'oubliez pas les accolades !

Votre réponse :

{aca,acca,aa}

Feedback:
Bravo !

Effectivement si on concatène chaque mot de L_1 avec chaque mot de L_2 en commençant par le mot de L_1 on obtient

$L_1 \bullet L_2 = \{\mathbf{ac}a, \mathbf{ac}ca, \mathbf{a}a, \mathbf{a}ca\} = \{aca, acca, aa\}$ car un ensemble ne peut pas contenir deux fois le même élément (aca).

Exemple :

Soit le langage $L = \{ab,\ a,\ abc,\ bc\}$ sur l'alphabet $\Sigma = \{a, b, c\}$, alors

1. Le langage L^2 est

$$L^2 = L \bullet L \quad = \quad \{\mathbf{ab}ab,\ \mathbf{ab}a,\ \mathbf{ab}abc,\ \mathbf{ab}bc,\ \mathbf{a}ab,\ \mathbf{a}a,\ \mathbf{a}abc,\ \mathbf{a}bc,\ \mathbf{abc}ab,\ ,\mathbf{abc}a,\ \mathbf{abc}abc,\ \mathbf{abc}bc,\ \mathbf{bc}ab,\ \mathbf{bc}a,\ \mathbf{bc}abc,\ \mathbf{bc}bc\}$$

$\rightarrow$ **chaque mot** de L concaténé avec chaque mot de L.

❖ **Opérateur + et ***

- ✓ L^+ == le langage constitué de tous les mots obtenus en concaténant un <u>nbre fini non nul</u> de mots de L.
  ➔On doit prendre min. 1 mot en L $\boxed{L^+ = L \cup L^2 \cup L^3 \cup L^4 \cup \cdots}$
- ✓ L^* == le langage constitué de tous les mots obtenus en concaténant un <u>nbre fini quelconque</u> de mots de L.
  ➔On peut aussi prendre 0 mot de L $\boxed{L^* = L^0 \cup L \cup L^2 \cup L^3 \cup L^4 \cup \cdots = \{\varepsilon\} \cup L^+}$

<u>Propriétés :</u>

- Les langages L^+ et L^* sont égaux ssi le mot ϵ appartient à L.
- Le langage Σ^* est constitué de **tous** les mots sur Σ.

/!\ Pour déterminer si un langage est régulier/compilable, nous avions besoin de 3 outils.

- **Les expressions régulières**
- **Les grammaires régulières/Systèmes générateurs**
- **Les reconnaisseurs/automates**

II. Expressions Régulières

❖ **Définition ==** Outils permettant de décrire de manière compacte, certains types de langages. Pour ça, nous avions besoin :

- D'un ensemble delta Δ contenant 6 symboles : $\emptyset$; λ ; (;) ; $*$; |
- D'un alphabet Σ et donc aucun symbole n'est une lettre de l'alphabet.

- ✓ Les expressions **primitives :** $\emptyset, \lambda, x$ (*x représente n'importe quelle lettre de l'alphabet*).
- ✓ Les expressions **composées :** construit à l'aide des symbole de Δ et à partir des express° primitives ou composées.

Si α et β sont des expressions régulières quelconques alors
- $\alpha\beta$ est une expression régulière.
- $(\alpha|\beta)$ est une expression régulière.
- $(\alpha)^*$ est une expression régulière

Si α est une expression régulière primitive ou parenthésée (entre parenthèses) alors α^* est une expression régulière.

→La concaténat° de 2 expressions régulières == 1 expression régulière.

Expressions Régulières : Exemples

Soit l'alphabet $\Sigma = \{a, b, c\}$

1) <u>Expressions régulières primitives sur Σ :</u>
   - Les symboles $\varnothing$ et λ
   - Les lettres de l'alphabet Σ : $\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{c}$

2) <u>Expressions régulières composées sur Σ</u>
   - $(a|b) \longrightarrow a$ et b sont des expressions régulières primitives donc $(a|b)$ aussi
   - $(a|b)\,c \longrightarrow (a|b)$ et c sont des expressions régulières donc $(a|b)\,c$ aussi
   - $(a|b)\,c^*$
     $\longrightarrow$ $(a|b)$ et c^* sont des expressions régulières (c : expression primitive)
     $\longrightarrow$ $(a|b)\,c^*$ est une expression régulière composée
   - $(a|b)^*(ccba)^*\lambda$
     $\longrightarrow$ $(a|b)$, a, b, c et λ sont des expressions primitives
     $\longrightarrow$ $(a|b)^*$ et $(ccba)^*$ sont des expressions régulières composées
     $\longrightarrow$ $(a|b)^*(ccba)^*\lambda$ est une expression régulière
   - $a^*\left((a|b)^*ccba^*\middle|\lambda\right)bc^*b(bc^*)^*$
   - $\cdots$

Expressions Régulières : Contre-Exemples

- **(aa)** :
  - ⟶ Les parenthèses ne sont utilisées que dans deux cas : $(\alpha|\beta)$ ou $(\alpha)^*$.
  - ⟶ Cette expression ne peut être apparentée à aucun de ces 2 cas.
- **$(aa|*)$** :
  - ⟶ $(\alpha|\beta)$ est une expression régulière ssi α et β sont des expressions régulières
  - ⟶ le symbole $*$ n'est pas une expression régulière correcte.
- **$(aa|\ |aa)$** :
  - ⟶ $(\alpha|\beta)$ expression régulière ssi α et β soit expressions régulières
  - ⟶ l'expression $|aa$ n'est pas une expression régulière correcte.
- **$^*(aa|ab^*a)$** :
  - ⟶ $*$ doit être précédé soit d'une expression régulière soit primitive soit parenthésée.
  - ⟶ le $*$ au début de l'expression n'est précédé par rien.

❖ **Langage associé :** Si est une expression régulière, alr le langage associé à α sera noté L_α

Expression	Langage associé	
$\varnothing$	$\{\}$	(Le langage vide)
λ	$\{\varepsilon\}$	(Le langage ne contenant que le mot vide)
x $(x \in \Sigma)$	$\{x\}$	(Le langage ne contenant que le mot x)
$\alpha\beta$	$L_\alpha \bullet L_\beta$	(Le langage des mots construit en concaténant un mot défini par α avec un mot défini par β)
$(\alpha\|\beta)$	$L_\alpha \cup L_\beta$	(Tous les mots définis soit par α soit par β)
$(\alpha)^*$ α^*	L_α^*	(Le langage des mots obtenus par concaténation d'un nombre quelconque de mots définis par α)

❖ **Langages Equivalents :** Si elles ont le **même langage associé**

Exemple :

Sur l'alphabet Σ les expressions régulières $\alpha = a(ba|a)$ et $\beta = (ab|a)a$ sont équivalentes.
En effet les langages associés à ces deux langages sont

- $L_\alpha = \{\mathbf{a}ba, \mathbf{a}a\} = \{aba, aa\}$
- $L_\beta = \{\mathbf{ab}a, \mathbf{a}a\} = \{aba, aa\} = L_\alpha$

Ces deux langages sont les mêmes ⟶ α et β sont équivalentes.

Abus fréquents (et commodes) de notations :

1. Les expressions régulières $\left(\alpha \middle| (\beta|\gamma)\right)$ et $\left((\alpha|\beta) \middle| \gamma\right)$ sont équivalentes quelles que soient α, β, γ
   ⟶ on va les écrire de manière raccourcie par $\left(\alpha \middle| \beta \middle| \gamma\right)$
2. On écrira $(\alpha)^+$ pour raccourcir les expressions $\alpha(\alpha)^*$ et $(\alpha)^*\alpha$
3. On écrira $(\alpha)^k$ pour raccourcir $\underbrace{\alpha\cdots\alpha}_{k \text{ fois}}$

- **Langages Réguliers :** s'il est le langage associé à une expression régulière sur Σ

Exemples :

1. Sur l'alphabet $\Sigma = \{a,\ b,\ c\}$, le langage constitué de tous les mots qui commencent et finissent par '*a*' est régulier :

   Un mot sur Σ commencera et finira par '*a*'

   soit si c'est le mot *a*

   soit si c'est un mot qui commence par *a* suivit par un nombre quelconque de lettres de Σ et qui se termine par *a* autrement dit un mot définit par l'expression régulière $a(a|b|c)^*a$

   $\longrightarrow$ langage régulier car associé à l'expression régulière $\left(a \middle| a(a|b|c)^*a\right)$

Expressions Régulières : Examen de Juin 2017

Soit les langages L_1, L_2 et L_3 construits sur l'alphabet $\Sigma = \{a, b, c, d\}$.

- $L_1 = aa^*(c|d)^*(cd)^*$
- $L_2 = (aa)^*(c|d)^*(cd)^*$
- $L_3 = \left\{(aa)^n(c|d)^p(cd)^n \middle| n, p \in \mathbb{N}\right\}$

Montrez que les langages L_1, L_2 et L_3 sont deux à deux différents.

Solution :

On a $L_1 \neq L_2$ si L_1 contient un mot qui n'est pas dans L_2. Or

1. Le mot *a* est dans L_1 car $a = aa^0(c|d)^0(cd)^0$
2. Le mot *a* n'est pas dans L_2 ni dans L_3 car les mots de ces langages ne peuvent contenir que des groupes *aa* et auront donc un nombre pair de *a*.
3. Le mot *aa* est dans L_2 car $aa = (aa)^1(c|d)^0(cd)^0$
4. Le mot *aa* n'est pas dans L_3 car pour obtenir seulement *aa*, il faudrait prendre $n = 1$ et $p = 0$ mais dans ce cas le mot obtenu est $aacd \neq aa$.

Donc par 1. et 2., on a $L_1 \neq L_2$ et $L_1 \neq L_3$.Et par 3. et 4. on a $L_2 \neq L_3$

Conclusion les langages L_1, L_2 et L_3 sont deux à deux différents.

III. Grammaires Régulières

- **Définition ==** Ensemble de règles formelles décrivant la/les manière de construire les mots du langage.

  Est composée de :

  - Un alphabet Σ (== **symboles terminaux**).
  - Un ensemble fini N disjoint de Σ dont les éléments sont des symboles non terminaux.
    (Noté < lettre majuscule >)
  - < *S* > un élément de N appelé **axiome** ou **point de départ** des « dérivat° ».
  - Un ensemble fini P dont les éléments sont des **règles de dérivation**.

  ➔**Une grammaire régulière est donc un quadruplet *G*** = (Σ, N, < *S* >, P)

- **Langage engendré :** par grammaire ***G*** est celui des mots auxquels on peut arriver en appliquant un nbre fini de dérivation à partir de l'axiome.

Ex : Soit la grammaire dont les règles de dérivations sont les suivantes :

$$
\begin{array}{lcl}
\langle S \rangle & \rightarrow & a\langle A \rangle \mid b\langle B \rangle \\
\langle A \rangle & \rightarrow & cc\langle A \rangle \mid b\langle B \rangle \mid \varepsilon \\
\langle B \rangle & \rightarrow & ab\langle B \rangle \mid b\langle S \rangle \mid c
\end{array}
$$

- ✓ On peur dire que j'ai 3 symboles « non terminaux » : N = {<S> , <A> , <B>}
- ✓ <S> == l'axiome (le point de départ des dérivations) comme on l'a dit dans la théorie.
- ✓ → Signifie : peut être remplacé par
- ✓ On voit que l'ensemble P des règles de dérivations possède 8 éléments.
- ✓ Donc par exemple <S> peut être remplacé soit par *a<A>* soit par *b<B>.*
- ✓ <B> peut être remplacé soit par *ab* <B> soit par *b* <S> soit par *c*
- ✓ Voici qlq exemples de dérivations possibles

1) Dérivation 1 :

Grammaire

$$
\begin{array}{lcl}
\langle S \rangle & \rightarrow & a\langle A \rangle \mid b\langle B \rangle \\
\langle A \rangle & \rightarrow & cc\langle A \rangle \mid b\langle B \rangle \mid \varepsilon \\
\langle B \rangle & \rightarrow & ab\langle B \rangle \mid b\langle S \rangle \mid c
\end{array}
$$

Dérivation

$$
\begin{array}{lcl}
\langle S \rangle & \rightarrow & a\langle A \rangle \\
 & \rightarrow & acc\langle A \rangle \\
 & \rightarrow & acc\varepsilon
\end{array}
$$

✓ → **le mot *acc* appartient au langage engendré**

2) Dérivation 2 :

Grammaire

$$
\begin{array}{lcl}
\langle S \rangle & \rightarrow & a\langle A \rangle \mid b\langle B \rangle \\
\langle A \rangle & \rightarrow & cc\langle A \rangle \mid b\langle B \rangle \mid \varepsilon \\
\langle B \rangle & \rightarrow & ab\langle B \rangle \mid b\langle S \rangle \mid c
\end{array}
$$

Dérivation

$$
\begin{array}{lcl}
\langle S \rangle & \rightarrow & a\langle A \rangle \\
 & \rightarrow & acc\langle A \rangle \\
 & \rightarrow & accb\langle B \rangle \\
 & \rightarrow & accbc
\end{array}
$$

✓ → **le mot *accbc* appartient au langage engendré**

3) Dérivation 3 :

Grammaire

$$
\begin{array}{lcl}
\langle S \rangle & \rightarrow & a\langle A \rangle \mid b\langle B \rangle \\
\langle A \rangle & \rightarrow & cc\langle A \rangle \mid b\langle B \rangle \mid \varepsilon \\
\langle B \rangle & \rightarrow & ab\langle B \rangle \mid b\langle S \rangle \mid c
\end{array}
$$

Dérivation

$$
\begin{array}{lcl}
\langle S \rangle & \rightarrow & b\langle B \rangle \\
 & \rightarrow & bb\langle S \rangle \\
 & \rightarrow & bba\langle A \rangle \\
 & \rightarrow & bbacc\langle A \rangle \\
 & \rightarrow & bbaccb\langle B \rangle \\
 & \rightarrow & bbaccbab\langle B \rangle \\
 & \rightarrow & bbaccbabc
\end{array}
$$

✓ → **le mot *bbaccbabc* appartient au langage engendré**

✓ **REMARQUE : Une dérivation se termine seulement si on a pris le dernier élément qui est 1 symbole/lettre (ex ε, c)**

❖ **Grammaire normalisée :** si tous les membres droits de ses règles de dérivations sont d'une des formes suivantes :

- ✓ ***ε***
- ✓ x où x est une lettre de l'alphabet Σ
- ✓ *x <Z>* où x est une lettre de l'alphabet Σ ET <Z> est un symbole non terminal.

Toute grammaire régulière peut être normalisée

Une règle non normale de la forme $\langle X \rangle \rightarrow ab\langle Z \rangle$ peut être remplacée par les règles (normales) :

$\langle X \rangle \rightarrow a\langle Y \rangle$
$\langle Y \rangle \rightarrow b\langle Z \rangle$

→ Il a fallu ajouter le symbole non terminal $\langle Y \rangle$ à l'ensemble P des symboles non terminaux

Conclusion :
Les grammaires régulières normalisées engendrent donc les mêmes langages que les grammaires régulières.

En gros, les étapes pour en arriver

❖ **Grammaire & expressions** … r une grammaire régulière.

- ✓ Tout langage est régulier si est associé à une expression régulière.
- ✓ Un langage régulier est tjs associé à une expression régulière.

1) Pour l'expression régulière primitive $\varnothing$: $< S > \to \ x < S >$ où x est une lettre quelconque de Σ.
   - $\longrightarrow$ règle non terminale
   - $\longrightarrow$ une grammaire avec cette unique règle ne génèrera aucun mot.
   - $\longrightarrow$ génère le langage vide comme $\varnothing$.

2) Pour l'expression régulière primitive λ : $< S > \to \ \varepsilon$
   - $\longrightarrow$ une grammaire avec cette unique règle ne peut générer que le mot vide ε
   - $\longrightarrow$ génère le langage ne contenant que le mot vide comme λ

3) Pour l'expression régulière primitive x $(x \in \Sigma)$: $< S > \to x$
   - $\longrightarrow$ une grammaire avec cette unique règle ne peut générer que le mot x
   - $\longrightarrow$ génère le langage $\{x\}$ comme x

4) Pour l'expression composée $(\alpha|\beta)$: à partir des grammaires G_α et G_β on va
   1. Ajouter un nouveau symbole non terminal $< S >$ comme axiome
   2. Ajouter la règle de dérivation $< S > \to < S_\alpha > \Big| < S_\beta >$
   3. Garder toutes les règles de dérivation associées aux grammaires G_α et G_β.

   Donc
   - $\longrightarrow$ à partir de l'axiome on pourra soit partir vers la grammaire générée par α soit vers la grammaire générée par β.
   - $\longrightarrow$ on pourra générer soit un mot associé à α soit un mot associé à β.
   - $\longrightarrow$ on générera le même langage que $(\alpha|\beta)$

5) Pour l'expression composée $\alpha\beta$: à partir des grammaires G_α et G_β on va
   1. Garder $< S_\alpha >$ comme axiome
   2. Remplacer toute règle terminale $< X > \to u$ de G_α par la règle $< X > \to u < S_\beta >$
   3. Garder toutes les autres règles de dérivation de G_α et toutes les règles de dérivation de G_β.

   Donc
   - $\longrightarrow$ on partira de l'axiome de G_α
   - $\longrightarrow$ on génèrera un mot associé à α
   - $\longrightarrow$ on repartira de l'axiome de G_β et on génèrera un mot associé à β.
   - $\longrightarrow$ on pourra généré un mot qui est la concaténation d'un mot associé à α avec un mot associé à β.
   - $\longrightarrow$ on générera le même langage que $\alpha\beta$

6) Pour l'expression composée $(\alpha)^*$: à partir de la grammaires G_α on va
   1. Ajouter un nouveau symbole non terminal $< S >$ comme axiome
   2. Ajouter la règle de dérivation $< S > \to \varepsilon \Big| < S_\alpha >$
   3. Remplacer toute règle terminale $< X > \to u$ de G_α par la règle $< X > \to u < S >$
   4. Garder toutes les autres règles de dérivation de G_α.

   Donc
   - $\longrightarrow$ on partira du nouvel axiome
   - $\longrightarrow$ soit on génèrera soit le mot vide
   - $\longrightarrow$ soit on ira à l'axiome de G_α on génèrera un mot associé à α on retournera en 1.
   - $\longrightarrow$ on pourra générer un mot qui est la concaténation d'un nombre quelconque de mots associés à α.
   - $\longrightarrow$ on générera le même langage que $(\alpha)^*$.

Grammaire et expression régulière : Conclusion

Regardons les variations d'états en fonction de l'ajout d'une nouvelle lettre :

1. Si le mot a un nombre pair de a alors
   - un mot acceptable car mot du langage.
     $\longrightarrow$ règle de dérivation : $< S > \rightarrow \varepsilon$
   - si on ajoute un a alors le nombre de a devient impair.
     $\longrightarrow$ le mot change d'état
     $\longrightarrow$ la règle de dérivation $< S > \rightarrow a < A >$
   - si on ajoute un b alors le nombre de a ne change pas et reste pair.
     $\longrightarrow$ le mot reste dans le même état
     $\longrightarrow$ règle de dérivation $< S > \rightarrow b < S >$
   - si on ajoute un c alors le nombre de a ne change pas et reste pair.
     $\longrightarrow$ le mot reste dans le même état
     $\longrightarrow$ règle de dérivation $< S > \rightarrow c < S >$

   Ces quatres règles peuvent se réécrire

$$< S > \rightarrow \varepsilon \,\Big|\, a < A > \,\Big|\, b < S > \,\Big|\, c < S >$$

2. Si le mot a un nombre impair de a alors
   - si on ajoute un a alors le nombre de a devient pair.
     $\longrightarrow$ le mot change d'état
     $\longrightarrow$ la règle de dérivation $< A > \rightarrow a < S >$
   - si on ajoute un b alors le nombre de a ne change pas et reste impair.
     $\longrightarrow$ le mot reste dans le même état
     $\longrightarrow$ règle de dérivation $< A > \rightarrow b < A >$
   - si on ajoute un c alors le nombre de a ne change pas et reste impair.
     $\longrightarrow$ le mot reste dans le même état
     $\longrightarrow$ règle de dérivation $< A > \rightarrow c < A >$

   Ces trois règles peuvent se réécrire

$$< A > \rightarrow a < S > \,\Big|\, b < A > \,\Big|\, c < A >$$

On a donc trouvé la grammaire suivante qui engendre L :

$$\boxed{\begin{array}{lcl} < S > & \rightarrow & \varepsilon \,|\, a < A > \,|\, b < S > \,|\, c < S > \\ < A > & \rightarrow & a < S > \,|\, b < A > \,|\, c < A > \end{array}}$$

Cette grammaire étant régulière, le langage L est donc régulier.

Expression régulière dont L serait le langage associé ?
Ajout de 'a' toujours par 2 pour garder un nombre pair.
Pour avoir toujour un mot du langage L après ajout de lettre(s) il faut

soit ajouter un 'b' $\rightarrow$ expression régulière : b
soit ajouter un 'c' $\rightarrow$ expression régulière : c
soit ajouter deux 'a' avec un nombre quelques de 'b' et/ou de 'c' entre les 2
$\rightarrow$ expression régulière : $a(b|c)^*a$

Donc une expression régulière dont L est le langage associé est la suivante

$$\boxed{\left(b \,\Big|\, c \,\Big|\, a(b|c)^*a\right)^*}$$

IV. Automates de Moore et NDFA

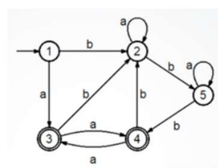
❖ **Définition ==** Schémas d'algorithmes permettant de décider si oui ou non, 1 mot proposé appartient au langage.

Est composé de :

- Un alphabet Σ (== **symboles terminaux**).
- Un ensemble *E* dont les éléments sont appelées états
- Un élément e_0 de *E* == **état initial**
- Un sous ensemble de *A* de *E* == **états acceptants**
- Une fonction ***t*** : $\Sigma \times E \rightarrow E$

➔**Une grammaire régulière est donc un quintuplet *M*** = (Σ, *E* , e_0, *A, t)*

Soit l'alphabet $\Sigma = \{a, b\}$ et l'automate suivant sur Σ



Alors,

1. L'ensemble des états est $E = \{1, 2, 3, 4, 5\}$
2. L'état initial est $e_0 = 1$ (marqué par une $\rightarrow$ ne venant d'aucun état)
3. L'ensemble des états acceptants est $A = \{3, 4\}$ (les doubles cercles)
4. La fonction de transition est indiquée par les flèches :
   Une flèche $(i) \xrightarrow{x} (j)$ signifie que, si on est à l'état *i* et que la lettre suivante du mot est *x*, alors on va passer à l'état *j*.

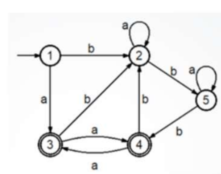
❖ **Obtenir un langage reconnu par l'automate :**

- Commencer à l'état entrant
- Passer d'un tat à un autre en suivant les flèches
- Terminer sur un état acceptant
- Concaténer dans l'ordre du parcours des flèches, les lettres indexant ces flèches.

➔**Un automate est un automate de Moore ssi :**

- Il possède 1 et 1 seul état entrant
- Pour tout état *i* et pour tte lettre de l'alphabet *x* , il existe 1 et 1 seule flèche partant de *i* étiquetée avec *x* .

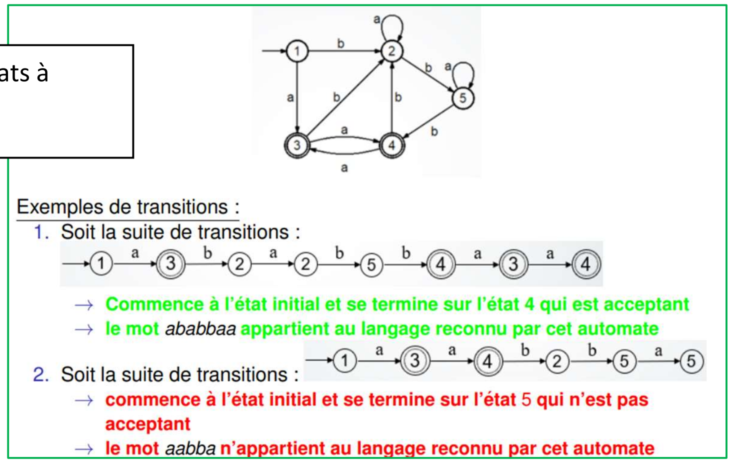
Notre exemple :



- a un seul état entrant : 1
- de chaque état partent deux flèches : une étiquetée *a* et une étiquetée *b*

Comme l'alphabet $\Sigma = \{a, b\} \rightarrow$ automate de Moore !

Un état acceptant sont les états à double ronds.



❖ **Table de transitions :** un automate de Moore peut être décrit par une table de transition

- ✓ Les états en en-tête de colonne
- ✓ Les lettre de l'alphabet Σ en en-tête de ligne
- ✓ L'état auquel on arrive en partant de l'état *i* en suivant la flèche indexée de la lettre *a* dans la case à l'intersection de la ligne correspondant à l'état *i* et de la colonne correspondant à la lettre *a.*
- ✓ Les états acceptant écrit dans une autre couleur
- ✓ Une flèche au-dessus de l'état initial.

Automate de Moore

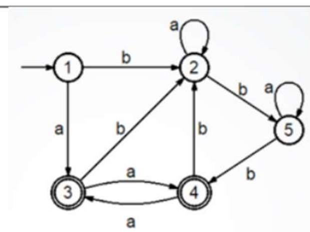


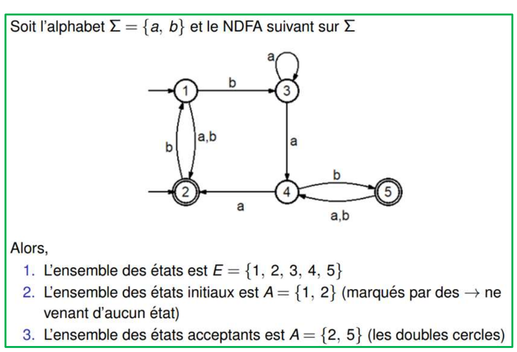
Table de transitions

	↓ **1**	**2**	**3**	**4**	**5**	***E***
a	3	2	4	3	5	
b	2	5	2	2	4	
Σ						

❖ **NDFA** Est composé de :

- Un alphabet Σ (== **symboles terminaux**).
- Un ensemble *E* dont les éléments sont appelées états
- Un sous-ensemble *I* de *E* == **états initiaux**
- Un sous ensemble de *A* de *E* == **états acceptants**
- Une fonction ***t*** : Σ *x E* → *E*

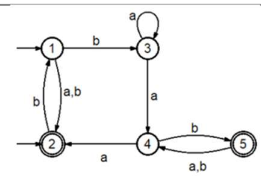
➔**Un NDFA est donc un quintuplet** ***M*** = (Σ, *E* , *I, A, t)*



❖ **Table de transition d'un NDFA**

Dans notre exemple, on obtient la table de transition :

NDFA	Table de transitions



	↓	↓				
	1	**2**	**3**	**4**	**5**	***E***
a	2	—	3,4	2	4	
b	2,3	1	—	5	4	
Σ						

La table de droite décrit entièrement le NDFA de gauche.
En effet,

- Les états sont en-tête de colonne
- Les lettres de l'alphabet sont en en-tête de ligne
- Les états acceptant sont en rouge
- Les flèches ↓ indiquent les états initiaux
- Les — signifient qu'il n'y pas de flèche indexée par la lettre correspondant à la colonne sortant de l'état correspondant à la ligne

⇨ **Un NDFA** est un automate dans lequel on peut trouver :

- Plusieurs états initiaux

/!\ NDFA est un Automate de Moore ≠ NDFA qui n'est pas un automate de Moore

Soit l'alphabet Σ = {a,b}

→NDFA qui est un automate de Moore :

- **Possède un et un seul état initial**
- **De chaque état part 1 et une seule flèche indexée par a ET 1 et une seule flèche indexée par b.**

→NDFA qui N'EST PAS automate de Moore :

- **Possède plusieurs état initial**
- **De chaque état, il n'y a pas les deux flèches pour a ET pour b (soit y a 2x b, soit juste a, soit juste , soit 2x b)** (*go regarder l'exemple dans le pdf exo Parcours Pédagogique p. 9-10)*

V. Opérations sur les Automates

❖ **Subset Construction ==** algorithme permettant d'obtenir un automate de Moore engendrant le même langage qu'un NDFA.

- Prendre l'ensemble des états initiaux du NDFA comme état initial de l'automate de Moore correspondant
- Sil reste un état *e* non traité : Pour chaque lettre de l'alphabet, faire une flèche indexée par cette lettre de cet état vers l'état *f* qui sera l'ensemble de tous les états du NDFA auxquels on peut arriver en suivant une flèche indexée par cette lettre partant d'un état du NDFA appartenant à l'état *e*
- Les langages reconnus par les automates ***M*** et ***M'*** sont égaux.
- Le nbre maximum d'états de *M'* est de $2^{|E|}$

En gros ce qu'il faut faire c'est à partir de la table de transition, faire un digraphe.

On va tjs commencer par prendre les états initials, et à partir de ces états, quel autre état on peut aller.

Par exemple, donc ici, on prend l’état 1 ET 2, qu’on encercle par UN DOUBLE ROND (car 2 est un état acceptant). On voit donc que l’état 1 ET 2, peuvent aller en

- A → juste à l’état 2
- B → à l’état 123 → Les mettre dans 1 MÊME CERCLE

Soit l’alphabet $\{a,\ b\}$ et le NDFA suivant sur cet alphabet avec sa table de transition

NDFA

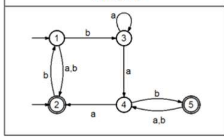


Table de transitions

	↓ 1	↓ 2	3	4	5	E
a	2	—	3, 4	2	4	
b	2, 3	1	—	5	4	
Σ						

Construisons un automate de Moore générant le même langage :

1) L’état initial est $\{1,\ 2\}$, l’ensemble des états initiaux du NDFA.
→ état acceptant car 2 est acceptant :

NDFA

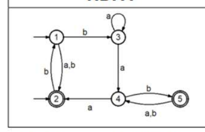


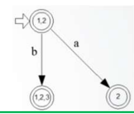
Table de transitions

	↓ 1	↓ 2	3	4	5	E
a	2	—	3, 4	2	4	
b	2, 3	1	—	5	4	
Σ						

2) Etat $\{1,\ 2\}$: avec le NDFA, en partant des états 1 et 2 et :

- en choisissant *a* comme caractère suivant, on ne peut aller qu’à l’état 2
  → flèche indexée par *a* vers le nouvel l’état $\{2\}$ acceptant car 2 l’est ;
- en choisissant *b* comme caractère suivant, on peut aller aux états 1, 2 et 3
  → flèche indexée par *b* vers le nouvel état $\{1,\ 2,\ 3\}$ acceptant car 2 l’est.

On obtient :



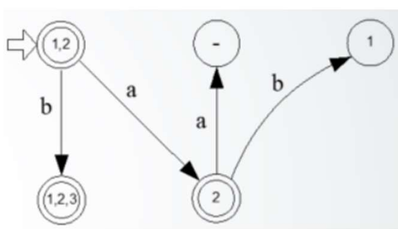
Puis on refais la même chose pour l’état {1,2,3} et {2}

Ainsi de suite jusque quand on a touuuut fini.

3. Regardons l'état $\{2\}$: avec le NDFA, en partant de l'état 2 et :

- en choisissant a comme caractère suivant, on ne peut aller nulle part
  → flèche indexée par a vers le nouvel état —, appelé parfois état poubelle, qui n'est pas acceptant ;
- en choisissant b comme caractère suivant, on peut aller qu'à l'état 1
  → flèche indexée par b vers le nouvel état $\{1\}$ qui n'est pas acceptant car 1 ne l'est pas.

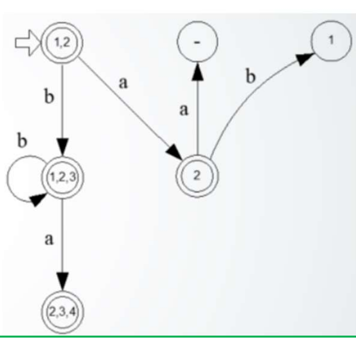
On obtient :



4. Regardons l'état $\{1,\ 2,\ 3\}$: avec le NDFA, en partant des états $1,\ 2$ et 3 et :

- en choisissant a comme caractère suivant, on peut aller aux états $2,\ 3,\ 4$
  → flèche indexée par a vers le nouvel état $\{2,\ 3,\ 4\}$ qui est acceptant car 2 l'est ;
- en choisissant b comme caractère suivant, on peut aller aux états $1,\ 2$ et 3
  → flèche indexée par b de l'état $\{1,\ 2,\ 3\}$ vers lui-même.

On obtient :

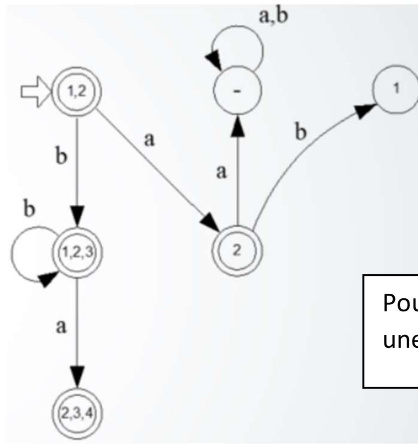


5. Regardons l'état poubelle — :

- C'est un état "sans issue" :

  $\longrightarrow$ flèche indexée par a et b de cet état vers lui-même.

On obtient :



Pour un état poubelle, on va refaire une boucle sur lui-même

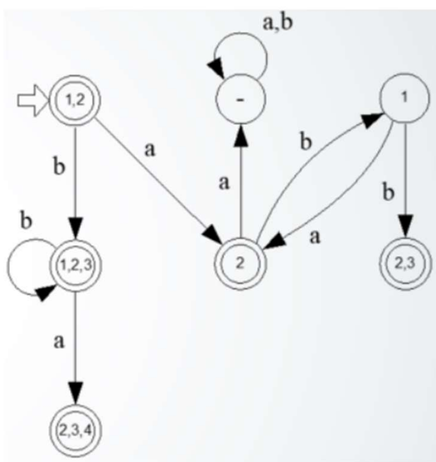
6. Regardons l'état $\{1\}$: avec le NDFA, en partant de l'état 1 et :

- en choisissant a comme caractère suivant, on ne peut aller qu'à l'état 2

  $\longrightarrow$ flèche indexée par a vers l'état $\{2\}$;
- en choisissant b comme caractère suivant, on peut aller aux états 2 et 3

  $\longrightarrow$ flèche indexée par b de l'état $\{1\}$ vers le nouvel état $\{2,\ 3\}$ qui est acceptant car 2 l'est.

On obtient :

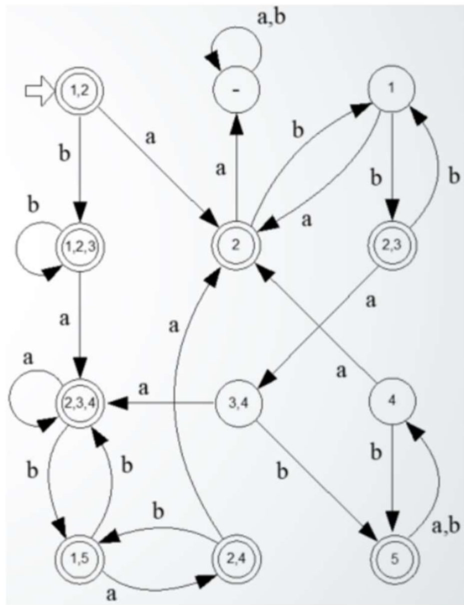


Bref ainsi de suite c'est lonnng. Je saute qlq étapes, puis voilà la dernière étape :

13. Regardons l'état $\{4\}$: avec le NDFA, en partant de l'état 4 et :

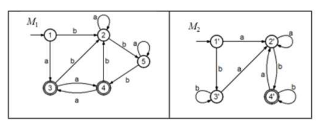
- en choisissant a comme caractère suivant, on ne peut aller qu'à l'état 2 $\rightarrow$ flèche indexée par a de l'état $\{4\}$ vers l'état $\{2\}$;
- en choisissant b comme caractère suivant, on ne peut aller qu'à l'état 5 $\rightarrow$ flèche indexée par b de l'état $\{4\}$ vers l'état $\{5\}$.

On obtient :



❖ Union de 2 langages :

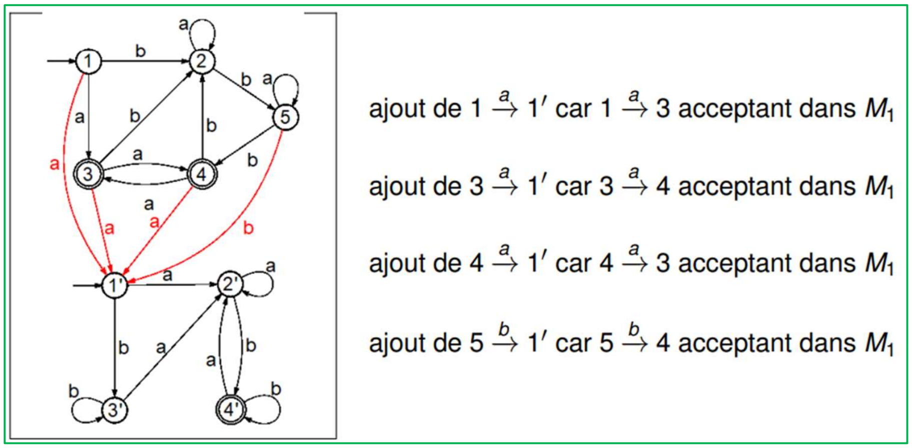
Soient L_1, L_2 2 langages reconnus par les automates de Moore M_1 et M_2 :



- ✓ Rassembler les 2 automates M_1 et M_2 en 1 seul NDFA
- ✓ *En gros je prends les 2 automates et je copie-colle en 1*
- ✓ Appliquer le subset construction si je veux obtenir un automate de Moore.

❖ Concaténation de 2 langages L_1 • LL_2:

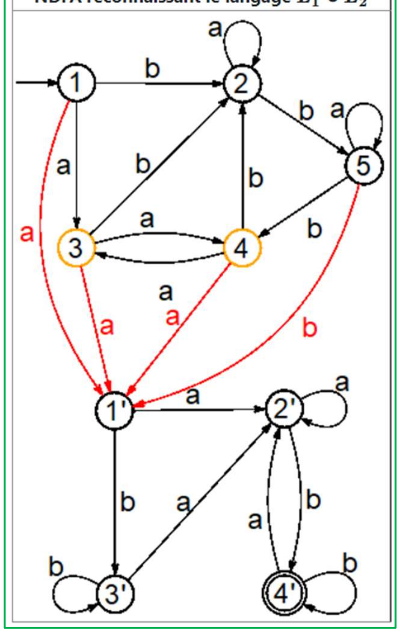
- ✓ Pour toute flèche étiquetée *x* d'un état *i* vers un état *j* où *j* est un **état acceptant de** M_1, ajoutons une flèche étiquetée par *x* de l'état vers l'état initial de M_2



En gros

- Déjà toute les flèches arriveront à l'état initial de M' (donc ici état 1)
- Tous les états qui ont une flèche qui va vers un état acceptant (ici états 3-4), bah on va rajouter une flèche de ces état à l'état initial M'.
- Par exemple, mon état 5, par la flèche étiquée b, va vers l'état 4, alr je rajoute juste flèche étiquée b de 5 à l'état 1 de M'.
- Et puis supprimer les état acceptant de M car l'état initial de M2 n'est pas acceptant.

NDFA reconnaissant le langage $L_1 \bullet L_2$

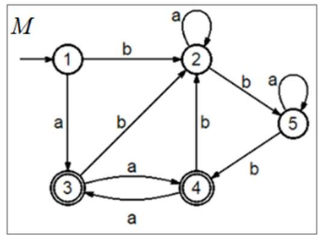


- Et go appliquer la subset construction au NDFA si je veux obtenir un automate de Moore.

❖ **Étoile d'un langage :** Pour obtenir un automate de Moore engendrant le Langage L^* :

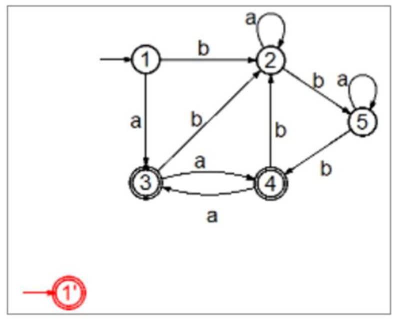
✓ Ajout d'1 état acceptant qui sera le nouvel état initial

Soient L le langage reconnu par l'automate de Moore M ci-dessous

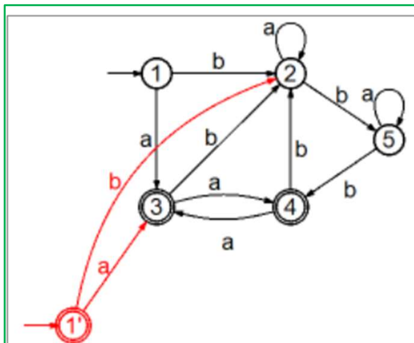


Construisons un automate de Moore engendrant le langage L^* :

1. Ajoutons le nouvel état initial **1'** :

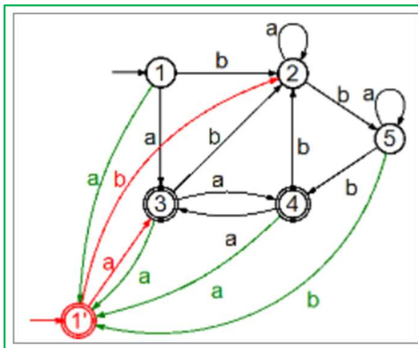


- ✓ Pour toute flèche étiquetée <u>x de l'état initial de **M** vers un état j</u>, on ajoute une flèche par x du nouvel état initial vers l'état j.



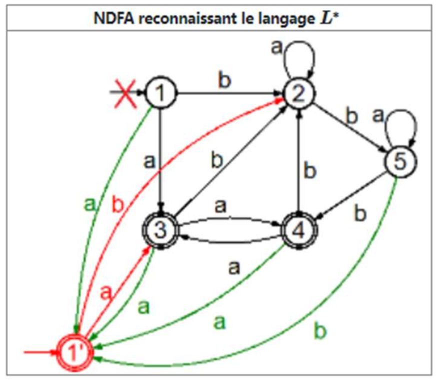
a. On a ajouté une flèche indexée par b du nouvel état initial $1'$ vers l'état 2 car il y en a une de l'état 1 (l'état initial de M) vers 2.
b. On a ajouté une flèche indexée par a du nouvel état initial $1'$ vers l'état 3 car il y en a une de l'état 1 (l'état initial de M) vers 3.

- ✓ Pour toute flèche <u>étiquetée x d'un état i vers un état j, où j est un état acceptant de **M**</u>
→Ajouter une flèche étiquetée par x de l'état i vers le new état initial.



a. On a ajouté une flèche indexée par a de l'état 1 vers le nouvel état initial $1'$ car il y en a une de 1 vers 3 qui est acceptant dans M
b. On a ajouté une flèche indexée par a de l'état 3 vers le nouvel état initial $1'$ car il y en a une de 3 vers 4 qui est acceptant dans M
c. On a ajouté une flèche indexée par a de l'état 4 vers le nouvel état initial $1'$ car il y en a une de 4 vers 3 qui est acceptant dans M
d. On a ajouté une flèche indexée par b de l'état 5 vers le nouvel état initial $1'$ car il y en a une de 5 vers 4 qui est acceptant dans M

- ✓ Retirer la flèche de l'état initial de M



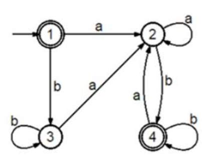
- ✓ Et go appliquer la subset construction au NDFA si je veux obtenir un automate de Moore.

VI. Automates & Grammaire

❖ **Grammaire associé à un Automate de Moore :** il faut

1. Associer à chaque état *i* de l'automate, un symbole non terminal *<i>*.
2. Associer l'axiome avec l'état initial de l'automate
3. Pour toute flèche indexée par une lettre *x* d'un état *i* vers un état *j*
   → Ajouter une règle de dérivation *<i>* → *x <j>*
4. Pour tout état acceptant *i*, ajouter la règle de dérivation *<i>* → ε

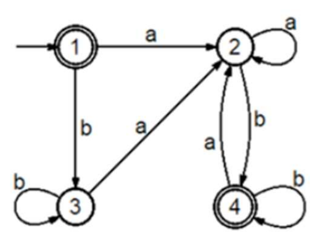
Soit le langage reconnu par l'automate Moore suivant



Construisons la grammaire engendrant ce langage.

1) Il y a 4 états : 1, 2, 3 et 4
   $\rightarrow$ les symboles non terminaux sont donc $<1>$, $<2>$, $<3>$ et $<4>$

2) Le symbole non terminal $<1>$ est l'axiome



3) Ajoutons une règle de dérivation par flèche
   a) $1 \xrightarrow{a} 2 \Longrightarrow$ règle de dérivation $<1> \rightarrow a<2>$
   b) $1 \xrightarrow{b} 3 \Longrightarrow$ règle de dérivation $<1> \rightarrow b<3>$
   c) $2 \xrightarrow{a} 2 \Longrightarrow$ règle de dérivation $<2> \rightarrow a<2>$
   d) $2 \xrightarrow{b} 4 \Longrightarrow$ règle de dérivation $<2> \rightarrow b<4>$
   e) $3 \xrightarrow{a} 2 \Longrightarrow$ règle de dérivation $<3> \rightarrow a<2>$
   f) $3 \xrightarrow{b} 3 \Longrightarrow$ règle de dérivation $<3> \rightarrow b<3>$
   g) $4 \xrightarrow{a} 2 \Longrightarrow$ règle de dérivation $<4> \rightarrow a<2>$
   h) $4 \xrightarrow{b} 4 \Longrightarrow$ règle de dérivation $<4> \rightarrow b<4>$

4) Ajoutons une règle par état acceptant

a) l'état 1 est acceptant $\Longrightarrow$ règle de dérivation $<1> \rightarrow \varepsilon$

b) l'état 4 est acceptant $\Longrightarrow$ règle de dérivation $<4> \rightarrow \varepsilon$

5) Toutes ces règles mises ensemble donne la grammaire recherchée :

$<1>$	$\rightarrow$	$a<2>$	$b<3>$	ε
$<2>$	$\rightarrow$	$a<2>$	$b<4>$	
$<3>$	$\rightarrow$	$a<2>$	$b<3>$	
$<4>$	$\rightarrow$	$a<2>$	$b<4>$	ε

❖ Automate de Moore associé à une grammaire régulière normalisée

1. Construire un NDFA comprenant le langage généré par la grammaire comme suit :

- Pour chaque symbole non terminal < I > → Ajouter un état I à l'automate
- Prendre l'état associé à l'axiome comme état initial
- Pour toute règle de la forme $<I> \rightarrow x<J>$ Ajouter $\boldsymbol{I} \xrightarrow{x} \boldsymbol{J}$
- Pour toute regle de la forme $<I> \rightarrow \varepsilon$, Rendre l'état I acceptant.
- Pour toute règle de dérivation de la forme $<I> \rightarrow x$, avec $x \in \Sigma$, ajouter une flèche indexée par x vers l'état acceptant « universel » Ω (en ajoutant cet état si celui-ci n'est pas déjà présent)

2. Appliquer la subset construction au NDFA construit en 1.

Soit l'alphabet $\Sigma = \{a, b, c\}$ et la grammaire régulière normalisée suivante

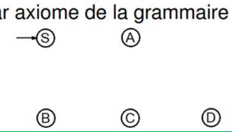
$<S>$	$\rightarrow$	$a<A>$	$b<B>$
$<A>$	$\rightarrow$	b	$c<S>$
$<B>$	$\rightarrow$	a	$b<C>$
$<C>$	$\rightarrow$	$c<D>$	ε
$<D>$	$\rightarrow$	$b<B>$	

Construisons le NDFA générant le langage reconnu par cette grammaire :

1) 5 symboles non terminaux : $<S>$, $<A>$, $<B>$, $<C>$ et $<D>$

$\Longrightarrow$ l'automate aura comme états S, A, B, C et D.

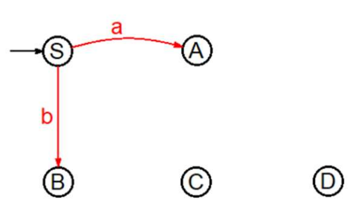
$\Longrightarrow$ S est l'état initial car axiome de la grammaire

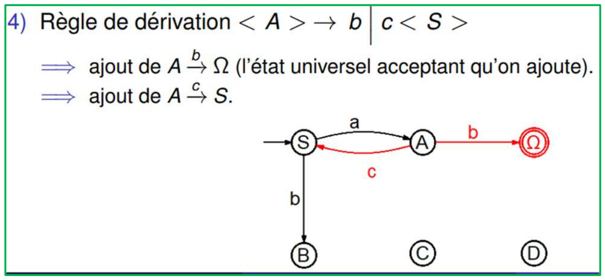


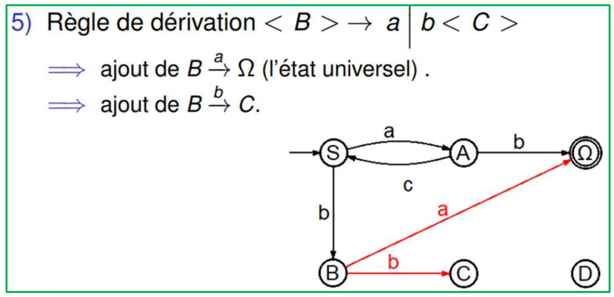
3) Règle de dérivation $<S> \rightarrow a<A> \mid b<B>$

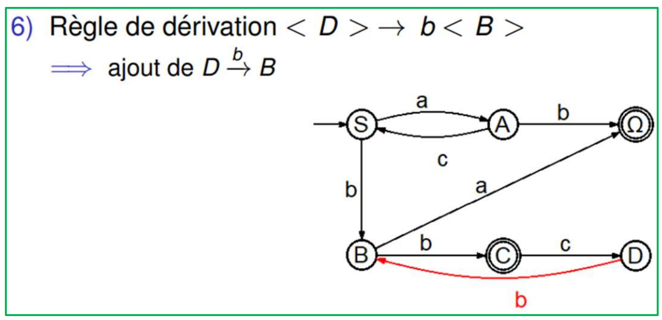
$\Longrightarrow$ ajout de $S \xrightarrow{a} A$.

$\Longrightarrow$ ajout de $S \xrightarrow{b} B$.









❖ **Conclusion :** Un langage *L* sur un alphabet Σ est **régulier**

- **SSI** *L* est associé à une expression régulière
- **SSI** *L* peut être engendré par une grammaire régulière || une grammaire régulière normalisée
- **SSI** *L* peut être reconnu par un automate de Moore || un NDFA

VII. Automate de Moore minimal : Algorithme de Moore-Nerode

❖ **Unicité de l'automate de Moore minimal** == Parmi ts les automates de Moore reconnaissant un même langage celui possédant le minimum d'état.

2 automates de Moore sont distincts :

- Soit s'ils n'ont pas le même nbre d'états
- Soit s'ils ont le même nbre d'états mais qu'on ne peut pas passer de l'un à l'autre par un simple renommage d'état.

❖ **Algorithme de Moore-Nerode : Optimisation d'un automale de Moore**

✓ En gros, on va fusionner plusieurs état possible dans un même cercle.

Exemple

Soit l'automate de Moore suivant et sa table de transitions

Automate de Moore

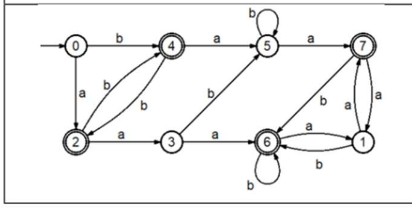


Table de transitions

	↓ 0	1	2	3	4	5	6	7	*E*
a	2	7	3	6	5	7	1	1	
b	4	6	4	5	2	5	6	6	
Σ									

Utilisons l'algorithme de Moore-Nerode pour optimiser cet automate de Moore :

Etape 0 :

On considère $\sim_0$ qui répartit les états en 2 ensembles : les acceptants et les autres.

$\Longrightarrow$ partition $E/\sim_0 = \Big\{ \boxed{[0] = \{0,\ 1,\ 3,\ 5\}} \ , \boxed{[2] = \{2,\ 4,\ 6,\ 7\}} \Big\}$

✓ Étape 0 : faire 2 ensembles : 1 pour les états acceptants, 1 pour les états non acceptants.

Etape 1 : Construction de $\sim_1$ à partir de $\sim_0$:

Regardons chaque classe d'équivalence séparément :

a) Regardons les états de la classe [0] :

- Etat 0 : $\begin{cases} 0 \overset{a}{\rightarrow} 2 \in \boxed{[2]} \\ 0 \overset{b}{\rightarrow} 4 \in \boxed{[2]} \end{cases}$ Etat 1 : $\begin{cases} 1 \overset{a}{\rightarrow} 7 \in \boxed{[2]} \\ 1 \overset{b}{\rightarrow} 6 \in \boxed{[2]} \end{cases}$
- Etat 3 : $\begin{cases} 3 \overset{a}{\rightarrow} 6 \in \boxed{[2]} \\ 3 \overset{b}{\rightarrow} 5 \in \boxed{[0]} \end{cases}$ Etat 5 : $\begin{cases} 5 \overset{a}{\rightarrow} 7 \in \boxed{[2]} \\ 5 \overset{b}{\rightarrow} 5 \in \boxed{[0]} \end{cases}$

$\Longrightarrow$ partition de cette classe en 2 sous-classes : $\{0,\ 1\}$ et $\{3,\ 5\}$.

b) Regardons les états de la classe [2] :

- Etat 2 : $\begin{cases} 2 \overset{a}{\rightarrow} 3 \in [0] \\ 2 \overset{b}{\rightarrow} 4 \in [2] \end{cases}$ Etat 4 : $\begin{cases} 4 \overset{a}{\rightarrow} 5 \in [0] \\ 4 \overset{b}{\rightarrow} 2 \in [2] \end{cases}$
- Etat 6 : $\begin{cases} 6 \overset{a}{\rightarrow} 1 \in [0] \\ 6 \overset{b}{\rightarrow} 6 \in [2] \end{cases}$ Etat 7 : $\begin{cases} 7 \overset{a}{\rightarrow} 1 \in [0] \\ 7 \overset{b}{\rightarrow} 6 \in [2] \end{cases}$

$\Longrightarrow$ pas de partition de cette classe.

Nouvelle partition $E/\sim_1 = \Big\{ [0] = \{0,\ 1\}\ ,\ [3] = \{3,\ 5\}\ ,[2] = \{2,\ 4,\ 6,\ 7\} \Big\}$

✓ Pour la partie a), on voit que l'état 3 et 5, on 2 sous classes différents par rapport à l'état 0 et 1.

✓ Donc ça veut dire qu'on va créer 2 new sous-classes pour cette classe [0]

✓ Par contre pour la b), on voit que les 4 états ont la même partition (genre chaque état à un [0] et un [2]

✓ Donc pas de new sous-classe

✓ Entre crochet, on va mettre l'état le + petit

Etape 2 : Construction de $\sim_2$

A partir de $E/\sim_1 = \Big\{[0] = \{0, 1\}\ , [3] = \{3, 5\}\ , [2] = \{2, 4, 6, 7\}\Big\}$

Regardons chaque classe d'équivalence séparément :

a) Regardons les états de la classe [0] :

- Etat 0 : $\begin{cases} 0 \xrightarrow{a} 2 \in [2] \\ 0 \xrightarrow{b} 4 \in [2] \end{cases}$ Etat 1 : $\begin{cases} 1 \xrightarrow{a} 7 \in [2] \\ 1 \xrightarrow{b} 6 \in [2] \end{cases}$

$\Longrightarrow$ Pas de partition de cette classe

	0	1	2	3	4	5	6	7	E
a	2	7	3	6	5	7	1	1	
b	4	6	4	5	2	5	6	6	
Σ									

$E/\sim_1 = \Big\{[0] = \{0, 1\}\ , [3] = \{3, 5\}\ , [2] = \{2, 4, 6, 7\}\Big\}$

b) Regardons les états de la classe [3] :

- Etat 3 : $\begin{cases} 3 \xrightarrow{a} 6 \in [2] \\ 3 \xrightarrow{b} 5 \in [3] \end{cases}$ Etat 5 : $\begin{cases} 5 \xrightarrow{a} 7 \in [2] \\ 5 \xrightarrow{b} 5 \in [3] \end{cases}$

$\Longrightarrow$ Pas de partition de cette classe

	0	1	2	3	4	5	6	7	E
a	2	7	3	6	5	7	1	1	
b	4	6	4	5	2	5	6	6	
Σ									

$E/\sim_1 = \Big\{[0] = \{0, 1\}\ , \boxed{[3] = \{3, 5\}}\ , [2] = \{2, 4, 6, 7\}\Big\}$

c) Regardons les états de la classe [2] :

- Etat 2 : $\begin{cases} \boxed{2 \xrightarrow{a} 3 \in [3]} \\ 2 \xrightarrow{b} 4 \in [2] \end{cases}$ Etat 4 : $\begin{cases} \boxed{4 \xrightarrow{a} 5 \in [3]} \\ 4 \xrightarrow{b} 2 \in [2] \end{cases}$
- Etat 6 : $\begin{cases} 6 \xrightarrow{a} 1 \in [0] \\ 6 \xrightarrow{b} 6 \in [2] \end{cases}$ Etat 7 : $\begin{cases} 7 \xrightarrow{a} 1 \in [0] \\ 7 \xrightarrow{b} 6 \in [2] \end{cases}$

$\Longrightarrow$ partition de cette classe en 2 sous-classes : $\{2, 4\}$ et $\{6, 7\}$.

Nouvelle partition $E/\sim_2 = \Big\{[0] = \{0, 1\}\ , [3] = \{3, 5\}\ , [2] = \{2, 4\}\ , [6] = \{6, 7\}\Big\}$

- ✓ Idem, à chaque fois on va regarder si y a des nouvel sous-classe à créer.
- ✓ /!\ Faire référence au nouvelle classe

Etape 3 : Construction de $\sim_3$

A partir de $E/\sim_2 = \Big\{[0] = \{0,\ 1\}\ ,\ [3] = \{3,\ 5\}\ ,[2] = \{2,\ 4\}\ ,\ [6] = \{6,\ 7\}\Big\}$

Regardons chaque classe d'équivalence séparément :

a) Regardons les états de la classe [0] :

- Etat 0 : $\begin{cases} 0 \xrightarrow{a} 2 \in [2] \\ 0 \xrightarrow{b} 4 \in [2] \end{cases}$ Etat 1 : $\begin{cases} 1 \xrightarrow{a} 7 \in [6] \\ 1 \xrightarrow{b} 6 \in [6] \end{cases}$

$\Longrightarrow$ partition de cette classe en 2 sous-classes : $\{0\}$ et $\{1\}$.

$\downarrow$

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	***E***
a	2	7	3	6	5	7	1	1	
b	4	6	4	5	2	5	6	6	

Σ

$E/\sim_2 = \Big\{[0] = \{0,\ 1\}\ ,\ [3] = \{3,\ 5\}\ ,[2] = \{2,\ 4\}\ ,\ [6] = \{6,\ 7\}\Big\}$

b) Regardons les états de la classe [3] :

- Etat 3 : $\begin{cases} 3 \xrightarrow{a} 6 \in [6] \\ 3 \xrightarrow{b} 5 \in [3] \end{cases}$ Etat 5 : $\begin{cases} 5 \xrightarrow{a} 7 \in [6] \\ 5 \xrightarrow{b} 5 \in [3] \end{cases}$

$\Longrightarrow$ Pas de partition de cette classe

$\downarrow$

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	***E***
a	2	7	3	6	5	7	1	1	
b	4	6	4	5	2	5	6	6	

Σ

$E/\sim_2 = \Big\{[0] = \{0,\ 1\}\ ,\ [3] = \{3,\ 5\}\ ,[2] = \{2,\ 4\}\ ,\ [6] = \{6,\ 7\}\Big\}$

c) Regardons les états de la classe [2] :

- Etat 2 : $\begin{cases} 2 \xrightarrow{a} 3 \in [3] \\ 2 \xrightarrow{b} 4 \in [2] \end{cases}$ Etat 4 : $\begin{cases} 4 \xrightarrow{a} 5 \in [3] \\ 4 \xrightarrow{b} 2 \in [2] \end{cases}$

$\Longrightarrow$ Pas de partition de cette classe

$\downarrow$

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	***E***
a	2	7	3	6	5	7	1	1	
b	4	6	4	5	2	5	6	6	

Σ

$E/\sim_2 = \Big\{[0] = \{0,\ 1\}\ ,\ [3] = \{3,\ 5\}\ ,[2] = \{2,\ 4\}\ ,\ [6] = \{6,\ 7\}\Big\}$

d) Regardons les états de la classe [6] :

- Etat 6 : $\begin{cases} 6 \xrightarrow{a} 1 \in [0] \\ 6 \xrightarrow{b} 6 \in [6] \end{cases}$ Etat 7 : $\begin{cases} 7 \xrightarrow{a} 1 \in [0] \\ 7 \xrightarrow{b} 6 \in [6] \end{cases}$

$\Longrightarrow$ Pas de partition de cette classe

Nouvelle partition

$E/\sim_3 = \Big\{[0] = \{0\}\ ,\ [1] = \{1\}\ ,\ [3] = \{3,\ 5\}\ ,[2] = \{2,\ 4\}\ ,\ [6] = \{6,\ 7\}\Big\}$

	0	1	2	3	4	5	6	7	E
a	2	7	3	6	5	7	1	1	
b	4	6	4	5	2	5	6	6	
Σ									

Etape 4 : Construction de $\sim_4$

A partir de

$$E/\sim_3 = \Big\{[0] = \{0\}\,,\ [1] = \{1\}\,,\ [3] = \{3,5\}\,,[2] = \{2,4\}\,,\ [6] = \{6,7\}\Big\}$$

Regardons chaque classe d'équivalence séparément :

a) Les classes [0] et[1] n'ont qu'un élément et ne peuvent donc plus être partitionnées.

	0	1	2	3	4	5	6	7	E
a	2	7	3	6	5	7	1	1	
b	4	6	4	5	2	5	6	6	
Σ									

$$E/\sim_3 = \Big\{[0] = \{0\}\,,\ [1] = \{1\}\,,\ [3] = \{3,5\}\,,[2] = \{2,4\}\,,\ [6] = \{6,7\}\Big\}$$

b) Regardons les états de la classe [3] :

- Etat 3 : $\begin{cases} 3 \xrightarrow{a} 6 \in [6] \\ 3 \xrightarrow{b} 5 \in [3] \end{cases}$ Etat 5 : $\begin{cases} 5 \xrightarrow{a} 7 \in [6] \\ 5 \xrightarrow{b} 5 \in [3] \end{cases}$

$\Longrightarrow$ Pas de partition de cette classe

	0	1	2	3	4	5	6	7	E
a	2	7	3	6	5	7	1	1	
b	4	6	4	5	2	5	6	6	
Σ									

$$E/\sim_3 = \Big\{[0] = \{0\}\,,\ [1] = \{1\}\,,\ [3] = \{3,5\}\,,[2] = \{2,4\}\,,\ [6] = \{6,7\}\Big\}$$

c) Regardons les états de la classe [2] :

- Etat 2 : $\begin{cases} 2 \xrightarrow{a} 3 \in [3] \\ 2 \xrightarrow{b} 4 \in [2] \end{cases}$ Etat 4 : $\begin{cases} 4 \xrightarrow{a} 5 \in [3] \\ 4 \xrightarrow{b} 2 \in [2] \end{cases}$

$\Longrightarrow$ Pas de partition de cette classe

	0	1	2	3	4	5	6	7	E
a	2	7	3	6	5	7	1	1	
b	4	6	4	5	2	5	6	6	
Σ									

$$E/\sim_3 = \Big\{[0] = \{0\}\,,\ [1] = \{1\}\,,\ [3] = \{3,5\}\,,[2] = \{2,4\}\,,\ [6] = \{6,7\}\Big\}$$

d) Regardons les états de la classe [6] :

- Etat 6 : $\begin{cases} 6 \xrightarrow{a} 1 \in [1] \\ 6 \xrightarrow{b} 6 \in [6] \end{cases}$ Etat 7 : $\begin{cases} 7 \xrightarrow{a} 1 \in [1] \\ 7 \xrightarrow{b} 6 \in [6] \end{cases}$

$\Longrightarrow$ Pas de partition de cette classe

$$E/\sim_4 = E/\sim_3 = \Big\{[0] = \{0\}\,,\ [1] = \{1\}\,,\ [3] = \{3,5\}\,,[2] = \{2,4\}\,,\ [6] = \{6,7\}\Big\}$$

$$E/\sim_4 = E/\sim_3 = \left\{[0] = \{0\}\,,\, [1] = \{1\}\,,\, [3] = \{3, 5\}\,, [2] = \{2, 4\}\,,\, [6] = \{6, 7\}\right\}$$

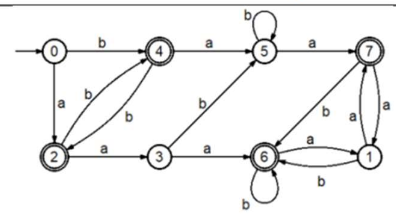
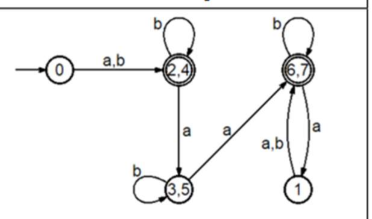
$\Longrightarrow$ $\sim_4 = \sim_3$

$\Longrightarrow$ on s'arrête

<u>États à fusionner :</u>

- état fusionné $\{3, 5\}$ non acceptant comme 1 et 5
- état fusionné $\{2, 4\}$ acceptant car 2 et 4 le sont
- état fusionné $\{6, 7\}$ acceptant car 6 et 7 le sont

On obtient l'automate moore optimisé ci-dessous

Automate de départ	Automate optimisé
	

✓ -On stop quand y a plus de nouvelle classe et on fusionne. Donc parfois on peut avoir 4 étapes comme parfois on peut seulement avoir 2 étapes puis on fusionne.