TD : Langages et Grammaires

Exercice 1 – Langages décrits par des expressions régulières

Déterminer le langage décrit par chacune des expressions régulières suivantes :

1. $a(a|b)^{*}b$

2. $(a|b)^{*}ab(a|b)^{*}$

3. $(aa)^{*}a$

4. $(a|b)^{*}(c|d)^{*}$

5. $aab(a|b)^{*}(bb|aa)^{+}$

6. $(a|ab)(c|bc)$

Correction détaillée (Exercice 1)

1. Expression : a(a|b)*b

Description formelle : L = { a w b | w ∈ {a,b}* }.

Raisonnement détaillé :

- L'opérateur (a|b)* génère toute chaîne (possiblement vide) sur l'alphabet {a,b}.

- La concaténation a (a|b)* b exige qu'une chaîne commence par 'a', suive d'une partie quelconque sur {a,b} puis se termine par 'b'.

Exemples : 'ab' (cas w = ε), 'aab' (w = 'a'), 'abb' (w = 'b'), 'aabab' (w = 'aab').

Propriétés : longueur minimale = 2, langage non vide, régulier.

2. Expression : (a|b)*ab(a|b)*

Description formelle : L = { x ab y | x,y ∈ {a,b}* }.

Raisonnement détaillé :

- Les (a|b)* de gauche et de droite permettent de placer le motif 'ab' n'importe où dans la chaîne (au début, milieu ou fin).

- Ainsi L est l'ensemble de toutes les chaînes sur {a,b} contenant 'ab' comme sous-chaîne.
Exemples : 'ab', 'aab', 'aba', 'babab', 'aaabbb' (si contient 'ab' à un endroit).
Remarque : Le langage est régulier et peut être reconnu par un automate fini qui détecte la présence du motif 'ab'.

3. Expression : (aa)*a

Description formelle : $L = \{ a^{(2k+1)} \mid k \geq 0 \}$ (chaînes de 'a' de longueur impaire >=1).
Raisonnement détaillé :
- (aa)* génère $(aa)^k = a^{\{2k\}}$ pour $k \geq 0$.
- En concaténant 'a' à droite, on obtient $a^{\{2k\}}\ a = a^{\{2k+1\}}$ (longueur impaire).
Exemples : 'a' (k=0), 'aaa' (k=1), 'aaaaa' (k=2).
Remarque : langage = a (aa)* équivalent.

4. Expression : (a|b)*(c|d)*

Description formelle : L = { x y | x ∈ {a,b}*, y ∈ {c,d}* }.
Raisonnement détaillé :
- Toute chaîne est une concaténation d'une partie (possiblement vide) contenant seulement a et b suivie d'une partie (possiblement vide) contenant seulement c et d.
- Ceci impose que dès qu'un symbole de {c,d} apparaît, aucun symbole de {a,b} ne peut apparaître après.
Exemples : ε (x=ε,y=ε), 'aaa', 'ccd', 'aabccd', 'bbd'.
Contre-exemples : 'cac' (car un 'a' apparaît après un 'c'), 'cab' (car 'a','b' après 'c').
Remarque : langage régulier; équivalent à l'union de langages où on sépare la frontière entre blocs {a,b} et {c,d}.

5. Expression : $aab(a|b)^*(bb|aa)^+$

Description formelle : L = { aab x z | x ∈ {a,b}*, z ∈ (bb|aa)+ }.
Raisonnement détaillé :
- Préfixe fixe 'aab'. Ensuite (a|b)* génère une zone quelconque.
- Le suffixe (bb|aa)+ exige une ou plusieurs occurrences consécutives de 'bb' ou 'aa' (par exemple 'bb', 'aa', 'bbbb', 'bbaa', 'aabb', ... selon concaténation des blocs).

- Remarques importantes sur la forme de z : z est une concaténation de $k \geq 1$ blocs, chaque bloc étant 'bb' ou 'aa'. Donc $|z|$ est pair et z ne contient jamais une seule lettre isolée.
Exemples : 'aabbb' ($x=\varepsilon$, z=bb), 'aabaa' ($x=\varepsilon$, z=aa), 'aabbbaa' (x=b, z=b aa?) vérifier découpage: 'aab' + 'b' + 'b aa' → accept.

6. Expression : (a|ab)(c|bc)

Description formelle : choisir un élément $U \in \{a,ab\}$ et $V \in \{c,bc\}$ puis concaténer UV.
Ensemble des mots possibles : { a c, a bc, ab c, ab bc } = { ac, abc, abbc } (remarquer que 'a'+'bc' = 'ab'+'c' = 'abc' conduisent à duplication).
Raisons : opération cartésienne entre alternatives.
Exemples : ac, abc, abbc.

Exercice 2

On considère la grammaire G = (N, T, P, S) avec :

N = {S}

T = {b, c}

P = { S → bS|cc }

Questions :

1. Quel est le type de G ?

2. Déterminer L(G).

Correction détaillée (Exercice 2)

1. Type de la grammaire

La grammaire contient les deux productions suivantes :

- **S → bS** : un non-terminal produit un **terminal suivi d'un non-terminal**, ce qui correspond à une règle de la forme **A → aB**.
- **S → cc** : un non-terminal produit une **suite de terminaux** uniquement.

Ces deux formes correspondent aux règles autorisées dans une **grammaire régulière droite** (type 3 dans la hiérarchie de Chomsky), qui admet :

- A → aB
- A → a
- ainsi que des suites de terminaux uniquement.

Conclusion :

G est une grammaire régulière (type 3).

2. Langage généré par la grammaire

La règle **S → bS** peut être appliquée plusieurs fois, ajoutant un symbole **b** à chaque itération.
La règle **S → cc** met fin à la dérivation.

Ainsi, les chaînes générées par la grammaire suivent toutes le schéma :

- un certain nombre de **b** (éventuellement aucun),
- suivi obligatoirement de **cc**.

Exemples de dérivations :

- S ⇒ cc → chaîne : **cc**
- S ⇒ bS ⇒ bcc → chaîne : **bcc**
- S ⇒ bS ⇒ bbS ⇒ bbcc → chaîne : **bbcc**
- S ⇒ bS ⇒ bbS ⇒ bbbS ⇒ bbbcc → chaîne : **bbbcc**

Forme générale :

$L(G)=\{b^n cc \mid n \geq 0\}$

Cela signifie que le langage contient :
cc, bcc, bbcc, bbbcc, …

Résumé

- La grammaire est **régulière droite (type 3)**.
- Elle génère toutes les chaînes composées de **n symboles b**, suivis de **cc**, avec **n ≥ 0**.

Exercice 3

Construire une grammaire pour les langages suivants :

1. $L = \{ ab^n a \ / \ n \in N \}$

2. $L = \{ 0^n 2^n \ / \ n \geq 0 \}$

3. $L = \{ 0^n 1^n \ / \ n \geq 0 \}$

4. $L = \{ a^n b^{\{2n\}} \ / \ n \geq 0 \}$

Correction détaillée (Exercice 3)

1) Langage $L = \{ ab^n a \ / \ n \in N \}$

- Soit la grammaire G=(N,T,P,S) avec :

  - ✓ **N** = { S, B }
  - ✓ **T** = { a, b }
  - ✓ **S** = S
  - ✓ **P** : $\{S \to a\ B\ a,\ B \to b\ B \mid \varepsilon\}$

Justification pas à pas :

- B génère b^n par récurrence : $B \Rightarrow \varepsilon$ (n=0) ; $B \Rightarrow bB \Rightarrow b\varepsilon$ (n=1) ; etc.

- S encadre B par deux 'a'. Donc $S \Rightarrow aBa \Rightarrow a\ b^n\ a$.

Exemple détaillé (n=2) : $S \Rightarrow aBa \Rightarrow a\ bBa \Rightarrow abbBa \Rightarrow abb\varepsilon a = abba$.

2) Langage $L = \{ 0^n 2^n \ / \ n \geq 0 \}$

- Soit la grammaire G=(N,T,P,S) avec :

  - ✓ **N** = { S }
  - ✓ **T** = { 0, 2 }
  - ✓ **S** = S
  - ✓ **P** : $\{S \to \varepsilon \mid 0\ S\ 2\}$

Justification pas à pas :

- Chaque application de 0S2 ajoute un 0 à gauche et un 2 à droite ; ε arrête la construction.

Preuve par induction sur n :

- Base n=0 : S ⇒ ε.
- Induction : si $S \Rightarrow 0^n 2^n$, alors $0\ S\ 2 \Rightarrow 0\ 0^n\ 2^n\ 2 = 0^{\{n+1\}}\ 2^{\{n+1\}}$.

Exemples : ε (n=0), 02 (n=1), 0022 (n=2).

Conclusion de l'induction

Par le principe de récurrence mathématique, toutes les chaînes 0^n2^n pour $n \geq 0$ sont générées par S.

3) Langage $L = \{ 0^n1^n / n \geq 0 \}$

Soit la grammaire G=(N,T,P,S) avec :

- ✓ N={S}
- ✓ T={0,1}
- ✓ P={S → ε | 0 S 1}

4) Langage $L = \{ a^nb^{\{2n\}} / n \geq 0 \}$

Grammaire proposée (construction simple) :

S → ε | a S bb

Justification pas à pas :

- L'application de aSbb ajoute un 'a' et deux 'b'. Après n applications, on obtient $a^n\ b^{\{2n\}}$.

Exemple (n=2) : $S \Rightarrow aSbb \Rightarrow a\ (aSbb)\ bb \Rightarrow aa\ \varepsilon\ bbbb = a^2\ b^4$.

Remarque : cette grammaire est context-free (type 2).