Automates et Langages

1. Notions de base

- **Alphabet (Σ)** : Ensemble fini de symboles (ex: {a, b, c}).
- **Mot** : Suite finie de symboles d'un alphabet (ex: "abc").
- **Mot vide (ε)** : Chaîne de longueur 0.
- **Langage (L)** : Ensemble de mots sur un alphabet. Peut être fini ou infini.
- **Langages particuliers** :
  - Langage vide : ne contient aucun mot (∅).
  - Langage ne contenant que le mot vide : {ε}.
  - Langage universel : ensemble de tous les mots possibles (Σ*).

2. Types de langages

- **Langage régulier** : Un langage est dit régulier s'il peut être construit à partir de trois éléments de base :
  - Le langage vide (ne contenant aucun mot): **∅**.
  - Le langage contenant uniquement le mot vide **{ε}**.
  - Tout langage constitué d'un seul symbole de l'alphabet L: {a}, {b}, etc
  - **Puis, on peut combiner ces éléments en appliquant un nombre fini de fois les opérations sur les langages**
- **Langage algébrique (ou hors-contexte)** : Peut être décrit par une grammaire algébrique et reconnu par un automate à pile.

3. Opérations sur les Langages

- **Union** : L1 ∪ L2 = Ensemble des mots appartenant à L1 ou L2.
  - Ex : Si L1 = {a, b} et L2 = {b, c}, alors L1 ∪ L2 = {a, b, c}.
- **Intersection** : L1 ∩ L2 = Ensemble des mots présents à la fois dans L1 et L2.
  - Ex : Si L1 = {a, b} et L2 = {b, c}, alors L1 ∩ L2 = {b}.
- **Différence** : L1 - L2 = Ensemble des mots qui sont dans L1 mais pas dans L2.
  - Ex : Si L1 = {a, b} et L2 = {b, c}, alors L1 \ L2 = {a}.

- **Complémentaire** : L^c : Ensemble des mots qui ne sont pas dans L (relativement à un univers donné).
  - Ex : Si Σ* = {a, b, c}* et L = {a, b}, alors L^c contient tous les mots possibles sauf "a" et "b".
- **Concaténation** : L1L2 = Ensemble des mots obtenus en accolant un mot de L1 avec un mot de L2.
  - Ex : Si L1 = {a, b} et L2 = {c, d}, alors L1L2 = {ac, ad, bc, bd}.
- **Étoile de Kleene** : L* = ensemble des mots obtenus par concaténation de 0, 1 ou plusieurs mots de L.
  - Ex : Si L = {ab}, alors L* = {ε, ab, abab, ababab, ...}.

> **Ordre de priorité des opérations :**
>
> 1. Étoile de Kleene (*)
> 2. Concaténation.
> 3. Union, intersection, différence.

4. Automates Finis (AF)

- **Définition** : Un automate fini est défini par (Σ, E, E0, F, δ) avec :
  - Σ : Alphabet.
  - E : Ensemble des états.
  - E0 : Ensemble des états initiaux.
  - F : Ensemble des états finaux.
  - δ : Fonction de transition (définit le passage d'un état à un autre en lisant un symbole).
    - Exemple: $\delta(A, a) = B$ passage de A à B avec l'arrète a
- **Automate Déterministe (AFD)** :
  - Un seul état initial.
  - Une seule transition possible par symbole et par état.
- **Automate Non Déterministe (AFN)** :
  - Plusieurs transitions possibles pour un même symbole.
  - Peut être transformé en AFD.

5. Expressions Régulières

- Permettent de décrire des langages réguliers.
- Opérateurs :
  - Union : $a|b$ ("a" ou "b").
  - Concaténation : ab ("a" suivi de "b").

- Étoile : a^* (répétition de "a" 0 ou plusieurs fois).
- Plus : a^+ (répétition de "a" 1 ou plusieurs fois).

Exemples :

- $(a|b)c^*$: Mots commençant par "a" ou "b", suivis de zéro ou plusieurs "c".
- $0(01)^*$: Un "0" suivi de 0 ou plusieurs "01".
- a^+b^* : Un ou plusieurs "a" suivis de zéro ou plusieurs "b".

Regex (regex101)

> Les expressions régulières sont des séquences de caractères qui forment un motif de recherche, principalement utilisées pour les opérations de recherche et de manipulation de texte. Voici quelques-unes des regex les plus couramment utilisées

Ancrages

- `^` : Début d'une ligne.
- `$` : Fin d'une ligne.

Caractères Spéciaux

- `.` : N'importe quel caractère sauf un retour à la ligne.
- `\d` : N'importe quel chiffre (équivalent à `[0-9]`).
- `\D` : N'importe quel caractère non chiffre.
- `\w` : N'importe quel caractère alphanumérique ou underscore (équivalent à `[a-zA-Z0-9_]`).
- `\W` : N'importe quel caractère non alphanumérique.
- `\s` : N'importe quel espace blanc (espace, tabulation, retour à la ligne).
- `\S` : N'importe quel caractère non espace blanc.

Quantificateurs

- `*` : 0 ou plusieurs occurrences du caractère précédent.
- `+` : 1 ou plusieurs occurrences du caractère précédent.
- `?` : 0 ou 1 occurrence du caractère précédent.
- `{n}` : Exactement n occurrences du caractère précédent.
- `{n,}` : n ou plus occurrences du caractère précédent.
- `{n,m}` : Entre n et m occurrences du caractère précédent.

Groupes et Alternances

- `(...)` : Groupe les sous-expressions.
- `|` : Alternance (ou logique).

Flags

Les flags sont des options qui modifient le comportement de la recherche. Ils sont souvent ajoutés à la fin de l'expression régulière, après une barre oblique (`/`).

- `g` : Global. Trouve toutes les correspondances dans la chaîne, pas seulement la première.
- `m` : Multi-line. Traite la chaîne comme multi-lignes, affectant les ancrages `^` et `$` .
- `i` : Insensible à la casse. Ignore la casse lors de la recherche.

Exemples

- `\d{3}-\d{2}-\d{4}` : Correspond à un numéro de sécurité sociale américain (ex. 123-45-6789).
- `[a-zA-Z]+@[a-zA-Z]+\.[a-zA-Z]+` : Correspond à une adresse e-mail simple.
- `\b\w+\b` : Correspond à un mot entier.

6. Grammaires Algébriques

- **Définition** : Une grammaire est définie par (Σ, V, P, S) avec :
  - Σ : Alphabet ou symboles terminaux (en min.)
  - V : Symboles non terminaux. (en maj.)
  - S : Axiome (symbole de départ).
  - R : Règles de production, sous la forme $A \to w$
    - A est un symbol non terminal
    - w est une suite de symboles (terminaux ou non)

Exemple

- $G = (\Sigma, V, P, S)$
- $\Sigma = a, b$
- $V = S, T$
- Les règles:
  - $S \to aS \mid bT \mid \varepsilon$
  - $T \to bT \mid \varepsilon$

Dérivation:

$$S \implies aS \implies aaS \implies aabT \implies aab \quad \text{et donc} \quad S \implies aab$$

7. Grammaire régulière

- **Définition** : Une grammaire est dite régulière si toutes ses règles de production sont de la forme :
  - **Grammaire régulière à droite** : $A \to aB$ ou $A \to a$ ou $A \to \varepsilon$
  - **Grammaire régulière à gauche** : $A \to Ba$ ou $A \to a$ ou $A \to \varepsilon$
- où A et B sont des symboles non terminaux, et a est un symbole terminal.

Correspondance en automate fini

- Chaque symbole non terminal (A) devient un **état de l'automate**
- L'axiome (S) et l'unique éta initial
- Les règles:
  - $A \to aB$ devient la transition $\delta(A, a) = B$
  - $A \to aB$ devient la transition $\delta(A, a) = étatterminal$
  - $A \to \varepsilon$ donne un état terminal$