

LES AUTOMATES

Les automates permettent de modéliser de nombreuses situations.

Il s'agit :

- d'une notion mathématique abstraite ;
- d'une notion très concrète utilisée dans de nombreux appareils (rendu de monnaie, ascenseur, chaîne d'assemblage, ...)

1 Découverte des automates

Exercice n° 1

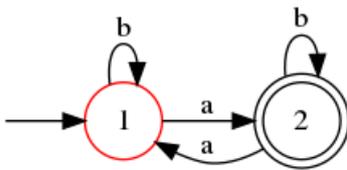
Ci-dessous on a construit un automate à deux états. "1" est l'état initial et "2" est un état final.

- en partant de "1" :

- le symbole "a" permet de passer à "2",
- le symbole "b" laisse l'état inchangé ;

- en partant de "2" :

- le symbole "a" permet de passer à l'état "1" ,
- le symbole "b" laisse l'état inchangé.



L'automate est actuellement dans l'état "1", indiqué en rouge.

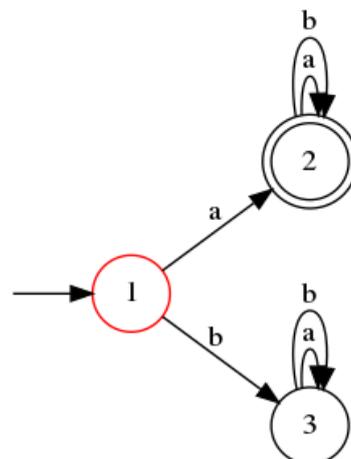
Définition : On dit qu'un mot est **reconnu** par l'automate, ou que l'automate **accepte** un mot si, en partant de l'état initial, et en fournissant à l'automate chaque lettre du mot, alors on arrive dans un état final.

1. Vérifier que l'automate reconnaît le mot "ababba".
2. Vérifier que l'automate ne reconnaît pas le mot babaaa".
3. L'automate reconnaît-il le mot "aaaa" ?
4. L'automate reconnaît-il le mot aaaba" ?
5. L'automate reconnaît-il le mot "aaaab" ?
6. L'automate reconnaît-il le mot "aaaaa" ?
7. Trouver l'objectif de cet automate.

Exercice n° 2

On considère l'automate ci-contre.

1. Vérifier que l'automate ne reconnais pas le mot "baba".
2. Vérifier que l'automate reconnais le mot "abba".
3. Trouver l'objectif de cet automate.



Exercice n° 3

Créer un automate sur l'alphabet $\{a, b\}$ permettant de reconnaître le mot "ba"

Exercice n° 4

Créer un automate sur l'alphabet $\{a, b, c\}$ permettant de reconnaître le mot "bac"

Exercice n° 5

Créer un automate sur l'alphabet $\{a, b\}$, permettant de reconnaître le mot "baba"

2 Python et les automates

La classe Automate

La classe Automate fournie permet de modéliser un automate. Par exemple, celui de l'exercice 1.

```
from Automata import Automata
```

```
automat = Automata(1, {2}, { 1 : {'a' : 2, 'b' : 1}, 2 : {'a' : 1, 'b' : 2 } } )
```

On peut obtenir une représentation de "automat" : `automat.show()`

L'automate est actuellement dans l'état 1, indiqué en rouge.

On modifie l'état de l'automate grâce à la méthode "next" : `automat.next('b')`

Ici l'automate est resté dans l'état "1", non final : `automat.is_final()`

Si on fournit à l'automate de caractère "a" : `automat.next('a')`

Alors l'automate passe dans l'état "1", **état final** : `automat.is_final()`

On peut alors réinitialiser l'automate dans son état initial : `automat.reset()`

Exercice n° 6

Avec le code ci-dessous, construire et visualiser l'automate des exercices 1 et 2.

Exercice n° 7

Créer et tester les automates des exercices 3,4 et 5.

3 Utiliser un automate pour chercher un mot dans un texte

Puisque les automates permettent d'adapter leur état en fonction des entrées lues précédemment, il est possible de les utiliser pour rechercher des mots dans un texte.

Étant donné un automate A, l'algorithme suivant permet de déterminer si le motif reconnu par A est présent dans le texte :

Fonction `automata_search(A, texte)`

Entrées : un automate A, un texte `texte`

Sortie : Vrai si le motif reconnu par A est présent dans `texte` , Faux sinon

1. $i \leftarrow 0$

2. $n \leftarrow \text{longueur}(\text{texte})$

3. **Tant que** $i < n$ et A n'est pas dans un état final **faire**

4. fournit à A le caractère `texte[i]`

5. $i \leftarrow i + 1$

6. **Fin tant que**

7. **Renvoyer** $i < n$

Exercice n° 8

1. Créer un automate qui reconnais le motif "chercher".

Aide : `A[3]['!=a']=0` permet de passer de l'état 3 à l'état 0 si le caractère est différent de "a"

2. Écrire une fonction "automata_search" implantant l'algorithme précédent en python.

3. Utiliser votre fonction pour rechercher le motif "chercher" dans les misérables.

4. Comparer le temps de cette recherche avec les algorithmes vus dans le chapitre.