**Exercice1**

Soit l'automate fini déterministe suivant :

- Q = {q0, q1, q2} (ensemble des états),

- Σ= {0, 1} alphabet,

-δ(fonction de transition) définie par :

* δ(q0, 0) = q1,
* δ (q0, 1) = q0,
* δ (q1, 0) = q1,
* δ (q1, 1) = q2,
* δ (q2, 0) = q2,
* δ (q2, 1) = q2,

- q0  est l'état initial, et

- F ={q2} est l'ensemble des états acceptants.

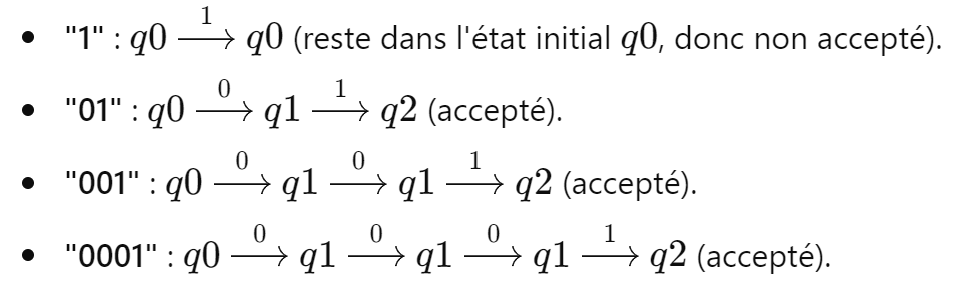
1. Donne des exemples de chaînes acceptées par cet automate.

2. Quelle est la propriété que satisfait une chaîne pour être acceptée ?

3. Tracer cette automate

**1. Exemples de chaînes acceptées par l'automate**

Pour que l'automate accepte une chaîne, il doit se retrouver dans l'état acceptant q2 après avoir lu toute la chaîne. Voici quelques exemples de chaînes qui atteignent cet état acceptant :



Ainsi, les chaînes "01", "001", "0001", etc., sont acceptées par l'automate.

**2. Propriété pour qu'une chaîne soit acceptée**

Pour qu'une chaîne soit acceptée par cet automate, elle doit contenir **au moins un "1" après être passée par l'état q1**. En d'autres termes, pour qu'une chaîne soit acceptée, elle doit contenir **au moins un "0" suivi d'un "1"**.

* La chaîne doit amener l'automate de l'état initial q0 à l'état q1 en lisant un "0".
* Ensuite, il doit lire un "1" pour faire une transition de q1 à l'état acceptant q2.
* Une fois dans l'état q2, toute autre séquence de "0" ou "1" maintient l'automate dans l'état q2.

0

q0

q1

1

q2

1

0

**Exercice2 :**

**Problème** : Créer un automate fini qui reconnaît toutes les chaînes binaires contenant le sous-mot 101 au moins une fois.

* **Alphabet** : Σ={0,1}
* **Objectif** : Concevoir un NFA pour ce langage, puis comparer avec un DFA.
* **Exemple de Fonctionnement du NFA**
  + Pour la chaîne 1101010 :

Expression régulière : (1|0)\*101(1|0)\*

0

A

B

1

1

C

D

0

1

1

0

0

1

A

B

1

C

D

0

1

1

0

0

Dans la AFN on peut aller à 2 états différentes avec un même terminal pour ce cas la fonction de transition dans A : A 1 A | 0 A et A 1 B | 0 A

Par contre dans la AFD on peut aller seulement à 1 état avec un même terminal

A 1 B | 0 A

**Exercice3 :**

Conçois un automate fini déterministe (DFA) qui reconnaît les chaînes de l’ensemble L = {a, b} contenant un nombre pair de `a`.

(b\*ab\*ab\*)+

b

b

q0

q1

a

a

q3

b

a

**Exercice4 :**

Concevons un automate fini déterministe (DFA) pour vérifier si une chaîne est un nom de variable valide en langage C.

**Règles de Nommage des Variables en C**

En C, un nom de variable valide doit respecter les règles suivantes :

1. Il doit commencer par une lettre (majuscule ou minuscule) ou un underscore (\_).
2. Les caractères suivants peuvent être des lettres, des chiffres ou des underscores (\_).
3. Un nom de variable ne peut pas être un mot réservé du langage (ici, nous ne traiterons pas cette contrainte pour simplifier l’exercice).

**Exemples**

* Chaînes valides : var1, \_temp, data\_123
* Chaînes invalides : 1var, #temp, var-2

**^[A-Za-z\_][A-Za-z0-9\_]\*$**

**[A-Za-z0-9\_]**

q0

q1

**[A-Za-z0-9\_]**

**[A-Za-z\_]**

q3

Autres rejet

Autres rejet

Autres rejet

**Exercice 5:**

Soit l'ensemble des entiers multiples de '10'

1- Donner une expression régulière qui valide ces mots.

2- Donner l'automate à états finis déterministe correspondant.

3- Donner une grammaire régulière équivalente.

**1. Donner une expression régulière qui valide les multiples de 10**

Les multiples de 10 sont des entiers qui se terminent toujours par le chiffre `0`. En notation décimale, les multiples de 10 peuvent être représentés par une suite de chiffres suivie d'un `0`. L'expression régulière qui correspond à cela est :

(0|[1-9][0-9]\*)0

**2. Donner l'automate à états finis déterministe (AFD) correspondant**

L'automate correspondant à cette expression régulière reconnaîtra toute chaîne de chiffres décimaux qui se termine par `0`. Voici un schéma textuel de l'automate :

- Etat : q0 , q1, qf

- Alphabet : {0, 1, 2, 3, ..., 9}

- Transitions :

**0**

- *S* *B* : Si on lit un `0` en état initial, on accepte immédiatement (cela correspond au nombre `0` qui est un multiple de 10.

**[1-9]**

- *S* A: Si on lit un chiffre entre `1` et `9` en état initial, on passe à l’état A.

- A A : En état A, si on lit un chiffre quelconque, on reste dans cet état.

**0**

**[0-9]**

- A B : En état q1, si on lit `0` à la fin, on passe à l'état final B, signifiant un multiple de 10.

- État final : B

**0**

**[0-9]**

**-9]**

**[1-9]**

**B**

**3. Donner une grammaire régulière équivalente**

Nous définissons les productions suivantes en utilisant des règles pour chaque état :

- Non-terminaux : S, A et B

- Terminaux: `{0, 1, 2, ..., 9}`

- Symbole de départ : S

- Règles de production :

- S 0 B | [1-9] A

- A 0 B | [0-9] A

**Exercice 6 :**

Soit l'ensemble des entiers multiples de '5'

1- Donner une expression régulière qui valide ces mots.

2- Donner l'automate à états finis déterministe correspondant.

3- Donner une grammaire régulière équivalente.

Pour valider les entiers multiples de 5 en notation décimale, nous devons traiter les entiers qui se terminent par `0` ou `5`. Voici comment aborder les différentes questions :

Donner une expression régulière qui valide les multiples de 5

Les multiples de 5 se terminent toujours par `0` ou `5`. En notation décimale, cela signifie que l’expression régulière qui valide ces entiers est :

(0|[1-9][0-9]\*)[05]

2. Donner l'automate à états finis déterministe (AFD) correspondant

**L'automate correspondant à cette expression régulière reconnaîtra toute chaîne de chiffres décimaux qui se termine par `0`. Voici un schéma textuel de l'automate :**

**- Etat : q0 , q1, qf**

**- Alphabet : {0, 1, 2, 3, ..., 9}**

**- Transitions :**

**0**

**- *S* *B* : Si on lit un `0` en état initial, on accepte immédiatement (cela correspond au nombre `0` qui est un multiple de 10.**

**[1-9]**

**- *S* A: Si on lit un chiffre entre `1` et `9` en état initial, on passe à l’état A.**

**- A A : En état A, si on lit un chiffre quelconque, on reste dans cet état.**

**0**

**[0-9]**

**- A B : En état A, si on lit `0` ou ‘5’ à la fin, on passe à l'état final B, signifiant un multiple de 5.**

**- État final : B**

**0 ou 5**

**[0-9]**

**-9]**

**[1-9]**

**B**

**3. Donner une grammaire régulière équivalente**

**Nous définissons les productions suivantes en utilisant des règles pour chaque état :**

**- Non-terminaux : S, A et B**

**- Terminaux: `{0, 1, 2, ..., 9}`**

**- Symbole de départ : S**

**- Règles de production :**

**- S (0 | 5) B | [1-9] A**

**- A (0 | 5) B | [0-9] A**

**Exercice7 :**

* + Donner un AFD qui accepte les mots spécifiés par l'expression régulière : (01|10)+

1. **Transitions :**
   * Depuis q0​ :
     + 0 → q1​ (commence un bloc "01").
     + 1 → q2​ (commence un bloc "10").
   * Depuis q1​ :
     + 1 → qf ​ (complet "01").
   * Depuis q2​ :
     + 0 → qf​ (complet "10").
   * Depuis qf​ :
     + 0 → q1​ (nouveau bloc "01" après un bloc précédent).
     + 1 → q2​ (nouveau bloc "10" après un bloc précédent).
2. **Diagramme d'états** :

Voici la représentation des transitions de cet AFN :

**0**

**C**

**1**

**0**

**0**

**1**

**1**

**D**

**0**

**1**

**Exercice8 :**

* + Donner un AFD qui accepte les mots spécifiés par l'expression régulière :

(0|1)+ 10 (1|0)+

**0**

**1**

**-9]**

**1**

**B**

**0**

**-9]**

**0|1**

**-9]**

**S 0 S | 1A**

**A 1A | 0B**

**B 0B | 1B | ε**

**Exercice 9 :**

* + Donner un AFD qui accepte les mots sur l'alphabet {0,1} qui ne contiennent pas la chaîne : "011"

**Construction de l’AFD**

1. **États** :
   * q0: état initial, aucun symbole de "011" n'a été rencontré.
   * q1​ : un 0 a été rencontré.
   * q2​ : la séquence "01" a été rencontrée.
   * qrej​ : état de rejet, atteint si "011" est détecté. Toute transition à partir de cet état reste dans cet état, ce qui signifie que la chaîne est rejetée.
2. **Transitions** :
   * Depuis q0:
     + 0 → q1 (le premier 0 est rencontré).
     + 1 → q1​ (on reste dans l'état initial, car 1 seul ne peut pas former "011").
   * Depuis q1​ :
     + 0 → q1​ (on reste dans cet état si un autre 0 est rencontré, sans risque d'atteindre "011").
     + 1 → q2​ (on a rencontré "01").
   * Depuis q2​ :
     + 0 → q1​ (on retourne à la recherche de "011" si un 0 est rencontré).
     + 1 → qrejq​ (on rencontre "011", donc on passe à l'état de rejet).
   * Depuis qrej ​ :
     + 0 → qrej​ (on reste dans l'état de rejet).
     + 1 → qrej​ (on reste dans l'état de rejet).
3. **État d'acceptation** : L'état q0​, ainsi que les états q1​ et q2​ sont des états d'acceptation. Une chaîne est acceptée si l’automate termine dans un de ces états sans jamais atteindre qrej​.

**Diagramme de transition**

0|1

0

q0

1

0

1

q2

q1

qrej

0

1