

# Compte Rendu Projet THL

Kichou, Nadir (G7)  
nadirkichou@hotmail.fr

Mansoura, Wael (G8)  
wael@gmail.com

Ouldali, Najib (G9)  
najib@gmail.com

Seddiki, Yacine (G9)  
yacine@gmail.com

May 28, 2023

## Contents

# 1 Automate Choisi

Nous avons choisi un automate pour reconnaître tout les nombres dans  $\mathbb{R}$

1, 2.0, -4.5, 10, +10, 0.45, -0.78, 85, 10.55...

## 2 Descripton de l'Automate Choisi

### 2.1 Definition

$$A = (\Sigma, E, e_0, F, \Delta)$$

Alphabet:  $\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, ., -, +\}$

Etats:  $E = \{e_0, e_1, e_2, e_3, e_4, e_5\}$

Etats Finaux:  $F = \{e_2, e_3, e_5\}$

Etat Initial:  $e_0 = e_0$

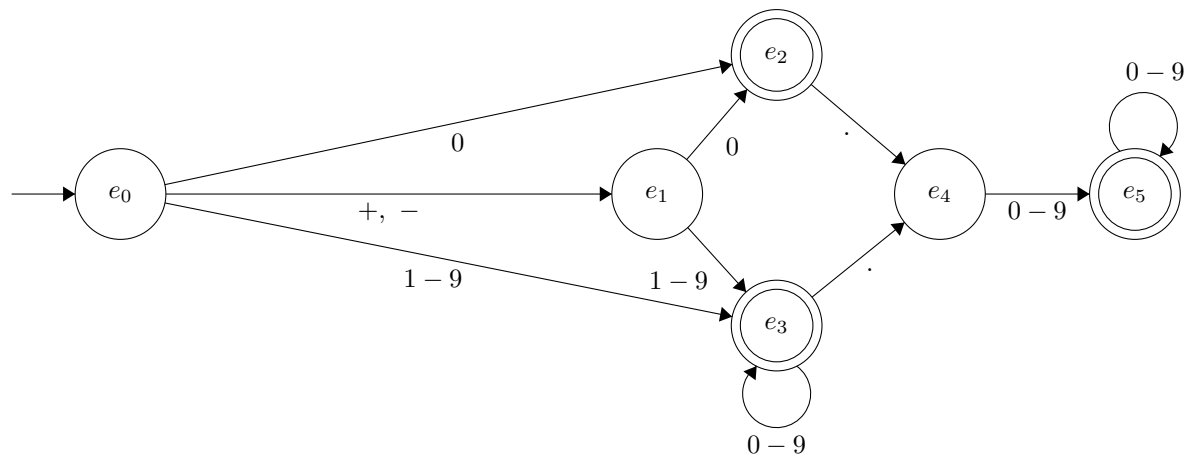
Transitions:

- $\delta(e_0, +) = e_1$
- $\delta(e_0, -) = e_1$
- $\delta(e_0, 0) = e_2$
- $\delta(e_0, 1) = e_3$
- $\delta(e_0, 2) = e_3$
- ...

### 2.2 Representation Matricielle

State	0	1	2	3	4	5	6	7	8	9	.	-	+
$e_0$	$e_2$	$e_3$	$e_3$	$e_3$	$e_3$	$e_3$	$e_3$	$e_3$	$e_3$	$e_3$	$e_p$	$e_1$	$e_1$
$e_1$	$e_2$	$e_3$	$e_3$	$e_3$	$e_3$	$e_3$	$e_3$	$e_3$	$e_3$	$e_3$	$e_p$	$e_p$	$e_p$
$e_2$	$e_p$	$e_p$	$e_p$	$e_p$	$e_p$	$e_p$	$e_p$	$e_p$	$e_p$	$e_p$	$e_4$	$e_p$	$e_p$
$e_3$	$e_3$	$e_3$	$e_3$	$e_3$	$e_3$	$e_3$	$e_3$	$e_3$	$e_3$	$e_3$	$e_4$	$e_p$	$e_p$
$e_4$	$e_5$	$e_5$	$e_5$	$e_5$	$e_5$	$e_5$	$e_5$	$e_5$	$e_5$	$e_5$	$e_p$	$e_p$	$e_p$
$e_5$	$e_5$	$e_5$	$e_5$	$e_5$	$e_5$	$e_5$	$e_5$	$e_5$	$e_5$	$e_5$	$e_p$	$e_p$	$e_p$

### 2.3 Representation Graphique



## 3 Fichier D'Entré

Le fichier d'entrée est sous cette forme

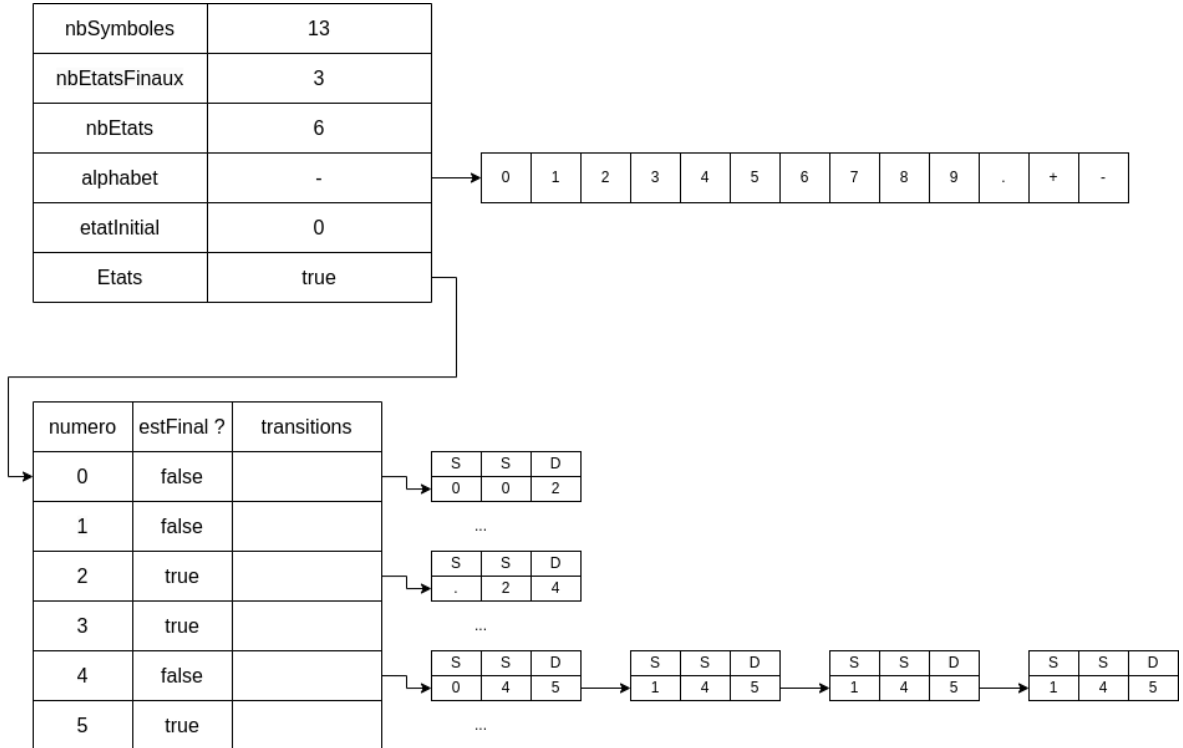
```

1 Nombre Etats = 6
2 Nombre Symboles = 13
3 Nombre Etats Finaux = 3
4 Nombre Transitions = 54
5 Nombre Mots Test = 6
6 Etats = 0, 1, 2, 3, 4, 5,
7 Symboles = 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, ., +, -,
8 Etat Initial = 0
9 Etats Finaux = 2, 3, 5,
10 delta(0, +) = 1
11 delta(0, -) = 1
12 delta(0, 0) = 2
13 delta(0, 1) = 3
14 delta(0, 2) = 3
15 delta(0, 3) = 3
16 delta(0, 4) = 3
17 delta(0, 5) = 3
18 delta(0, 6) = 3
19 delta(0, 7) = 3
20 delta(0, 8) = 3
21 delta(0, 9) = 3
22 delta(1, 0) = 2
23 delta(1, 1) = 3
24 delta(1, 2) = 3
25 delta(1, 3) = 3
26 delta(1, 4) = 3
27 delta(1, 5) = 3
28 delta(1, 6) = 3
29 delta(1, 7) = 3
30 delta(1, 8) = 3
31 delta(1, 9) = 3
32 delta(3, 0) = 3
33 delta(3, 1) = 3
34 delta(3, 2) = 3
35 delta(3, 3) = 3
36 delta(3, 4) = 3
37 delta(3, 5) = 3
38 delta(3, 6) = 3
39 delta(3, 7) = 3
40 delta(3, 8) = 3
41 delta(3, 9) = 3
42 delta(2, .) = 4
43 delta(3, .) = 4
44 delta(4, 0) = 5
45 delta(4, 1) = 5
46 delta(4, 2) = 5
47 delta(4, 3) = 5
48 delta(4, 4) = 5
49 delta(4, 5) = 5
50 delta(4, 6) = 5
51 delta(4, 7) = 5
52 delta(4, 8) = 5
53 delta(4, 9) = 5
54 delta(5, 0) = 5
55 delta(5, 1) = 5
56 delta(5, 2) = 5
57 delta(5, 3) = 5
58 delta(5, 4) = 5
59 delta(5, 5) = 5
60 delta(5, 6) = 5
61 delta(5, 7) = 5
62 delta(5, 8) = 5
63 delta(5, 9) = 5
64 Mots Test Valide = -40.00, 480790, -10, 8, 4, +2,
65 Mots Test Invalide = 007.07, -.1., ++78549, ---4.887, +--.15.+2, -45.45

```

## 4 Structures de Données

Pour stocker les noms des etats et les identifier nous avons utilisé des entiers au lieu de chaines de caracteres pour un soucis de simplicité



```

1 {
2   int nbSymboles;
3   int nbEtats;
4   int nbEtatsFinaux;
5   char *alphabet;
6   // TODO: change to string ?
7   int etatInitial;
8   Etat **etats;
9 };

```

Listing 1: Structure Automate

```

1 struct Etat
2 {
3   // TODO: change to string
4   int numero;
5   bool estFinal;
6   List *transitions;
7 };

```

Listing 2: Structure Etat

```

1 struct Transition
2 {
3   char symbole;
4   // TODO: change to string
5   int source;
6   // TODO: change to string
7   int destination;
8 };

```

Listing 3: Structure Transition

## 5 Captures D'Ecran de L'execution

```
• raiden@raiden-pc:~/Desktop/dev/TP-L2/TP-This$ gcc *.c
• raiden@raiden-pc:~/Desktop/dev/TP-L2/TP-This$ ./a.out
[automate]: nom du fichier: automate-decimales.txt

- Nombre D'Etats: 6
- Nombre De Symboles: 13
- Nombre D'Etats Finaux: 3
- Symboles: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, ., +, -]

Etat(0): (12 transitions) non-final
  delta(0, +) = 1
  delta(0, -) = 1
  delta(0, 0) = 2
  delta(0, 1) = 3
  delta(0, 2) = 3
  delta(0, 3) = 3
  delta(0, 4) = 3
  delta(0, 5) = 3
  delta(0, 6) = 3
  delta(0, 7) = 3
  delta(0, 8) = 3
  delta(0, 9) = 3

Etat(1): (10 transitions) non-final
  delta(1, 0) = 2
  delta(1, 1) = 3
  delta(1, 2) = 3
  delta(1, 3) = 3
  delta(1, 4) = 3
  delta(1, 5) = 3
  delta(1, 6) = 3
  delta(1, 7) = 3
  delta(1, 8) = 3
  delta(1, 9) = 3

Etat(2): (1 transitions) final
  delta(2, .) = 4

Etat(3): (11 transitions) final
  delta(3, 0) = 3
  delta(3, 1) = 3
  delta(3, 2) = 3
  delta(3, 3) = 3
  delta(3, 4) = 3
  delta(3, 5) = 3
  delta(3, 6) = 3
  delta(3, 7) = 3
  delta(3, 8) = 3
  delta(3, 9) = 3
  delta(3, .) = 4

Etat(4): (10 transitions) non-final
  delta(4, 0) = 5
  delta(4, 1) = 5
  delta(4, 2) = 5
  delta(4, 3) = 5
  delta(4, 4) = 5
  delta(4, 5) = 5
  delta(4, 6) = 5
  delta(4, 7) = 5
  delta(4, 8) = 5
  delta(4, 9) = 5

Etat(5): (10 transitions) final
  delta(5, 0) = 5
  delta(5, 1) = 5
  delta(5, 2) = 5
  delta(5, 3) = 5
  delta(5, 4) = 5
  delta(5, 5) = 5
  delta(5, 6) = 5
  delta(5, 7) = 5
  delta(5, 8) = 5
  delta(5, 9) = 5

[automate]: deterministe
[automate]: donnez un mot: -45,0
(0, ., no-final) =>(1, 4, no-final) =>(3, 5, final) =>(3, ., final) =>(4, 0, no-final)
[automate]: "-45,0" appartient au langage
```

## 6 Autre

Si vous souhaitez consulter le code du projet vous pourrez le retrouver sur mon [github](#).