

Ichou  
Aymane  
L2Y  
N°21007668

# Rendu Algo 2 – F.Balmas

## Sommaire :

---

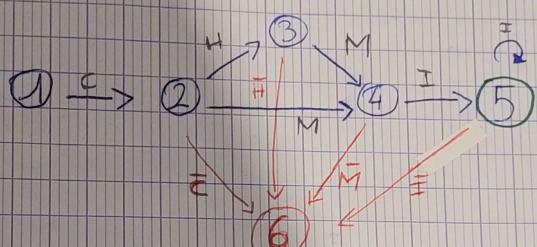
- 1-Schémas des automates/tables de transitions.
  - 2-Traces d'exécutions des étapes 2 à 5 ainsi que 7&9.
  - 3-Traces d'exécutions des programmes de vérifications.
  - 4-Explications des programmes de vérifications.
  - 5-Tests des limites.
  - 6-Choses à savoir.
-

Avant tout, mes motifs sont :

- CH\*MI+
- CH?MI\*

## 1.Schémas des automates correspondant à chacun des motifs, ainsi que leurs tables de transitions :

1<sup>er</sup> motif : CH\*MI+



- Etat succès : 5
- Etat rejett : 6
- Alphabet : {CHMI}
- Etats : 1 à 6

1<sup>er</sup> motif : CH\*MI+

$\rightarrow$	C	H	M	I	$AL^*$	$\rightarrow$
1	2	6	6	6	6	1
2	6	3	4	6	6	2
3	6	6	4	6	6	3
4	6	6	6	5	6	4

Etat rejet : 6

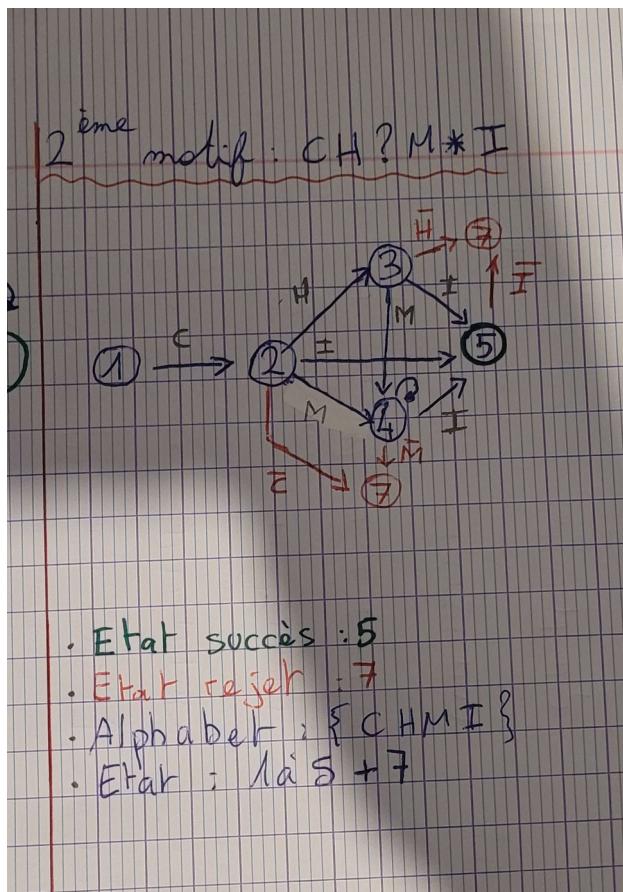
Etat succès : 5

Alphabet : {CHMIS}

Etats : 1 à 6

\* : Autre Lettre, lettre  $\notin$  Alphabet

## 2nd motif : CH?MI\*



2<sup>nd</sup> motif : CH?MI\*

$\rightarrow$	C	H	M	I	AL*
1	2	7	7	7	7
2	7	3	4	5	7
3	7	7	4	5	7
4	7	7	4	5	7

Etat rejet : 7  
 Etat succès : 5  
 Alphabet : {CHMI}  
 Etats : 1 à 5 + 7  
 AL\*: Autre lettre, lettre & Alphabet

## 2.Traces d'exécutions des étapes 2 à 5 ainsi que 7 et 9

Étape 2 : Recherche d'occurrences sur le 1<sup>er</sup> motif dans un tableau de 200 lettres aléatoires.

```
Tableau des occurrences:  
CHHMI [ 7      11 ]  
CMI [ 170     172 ]  
2 motifs trouvés.  
-----
```

Étape 3: Recherche d'occurrences sur le 2nd motif dans le même tableau de lettre.

```
Tableau des occurrences:  
CI [ 19      20 ]  
CI [ 21      22 ]  
CI [ 37      38 ]  
CI [ 47      48 ]  
CI [ 62      63 ]  
CI [ 70      71 ]  
CI [ 77      78 ]  
CI [ 85      86 ]  
CI [ 93      94 ]  
CMMMI [ 95      99 ]  
CI [ 118     119 ]  
CHI [ 120     122 ]  
CI [ 140     141 ]  
CHI [ 142     144 ]  
CI [ 148     149 ]  
CI [ 152     153 ]  
CMI [ 170     172 ]  
CI [ 173     174 ]  
CI [ 180     181 ]  
CI [ 183     184 ]  
CI [ 195     196 ]  
21 motifs trouvés.  
-----
```

Étape 4 : Union des occurrences des 2 motifs trouvées, sans doublons.

Tableau des occurrences:  
CHHMI avec 1 occurrences trouvée(s)  
CMI avec 2 occurrences trouvée(s)  
CI avec 17 occurrences trouvée(s)  
CMMMI avec 1 occurrences trouvée(s)  
CHI avec 2 occurrences trouvée(s)  
5 motifs unique trouvés.

Étape 5 : Classement de l'union selon le nombre d'occurrence

Tableau des occurences:  
CI avec 17 occurences trouvée(s)  
CMI avec 2 occurences trouvée(s)  
CHI avec 2 occurences trouvée(s)  
CMMMI avec 1 occurrences trouvée(s)  
CHHMI avec 1 occurrences trouvée(s)  
5 motifs unique trouvés.

Étape 9 : Intégration du code donnée, avec un tableau de 200 lettres aléatoires.

7 11  
170 172  
19 20  
21 22  
37 38  
47 48  
62 63  
70 71  
77 78  
85 86  
93 94  
95 99  
118 119  
120 122  
140 141  
142 144  
148 149  
152 153  
170 172  
173 174  
180 181  
183 184  
195 196

CHHMI CMI CI CI CI CI CI CI CMMMI CI CHI CI CHI CI CI CMI CI CI CI CI

### 3.Traces d'exécutions des programmes de vérifications des étapes 6, 7, 8

Étape 6 : Vérification de la fonction effectuant la recherche de motifs par le biais de résultat connu à l'avance.

```
-----Résultat attendu de la fonction search_motif :-----
wz[ 1  2]
wxyz[ 4  7]
wyyz[ 48  51]
wyz[ 65  67]
wz[ 69  70]
wz[ 79  80]
wz[ 96  97]
wxz[ 118 120]
wz[ 132 133]
wxyyyz[ 145 150]
wz[ 180 181]
wxz[ 189 191]
wxz[ 195 197]
Soit un total de 13 motifs
-----Résultat obtenu avec la fonction search_motif :-----
Tableau des occurrences:
wz [ 1      2 ]
wxyz [ 4      7 ]
wyyz [ 48     51 ]
wyz [ 65     67 ]
wz [ 69     70 ]
wz [ 79     80 ]
wz [ 96     97 ]
wxz [ 118    120 ]
wz [ 132    133 ]
wxyyyz [ 145    150 ]
wz [ 180    181 ]
wxz [ 189    191 ]
wxz [ 195    197 ]
13 motifs trouvés.
-----
```

Étape 7 : Vérification de la fonction effectuant le calcul de l’union

```
-----Résultat attendu de la fonction union_motif :-----
Union des motifs trouvés sans doublons
wxxyz 1
wxyz 2
wyyz 1
wxz 6
wyz 1
wz 6
wxyyyz 1
--> 7 motifs uniques
-----Résultat obtenu avec la fonction union_motif:-----
Tableau des occurrences:
wz avec 6 occurrences trouvée(s)
wxyz avec 2 occurrences trouvée(s)
wyyz avec 1 occurrences trouvée(s)
wyz avec 1 occurrences trouvée(s)
wxz avec 6 occurrences trouvée(s)
wxyyyz avec 1 occurrences trouvée(s)
wxxyz avec 1 occurrences trouvée(s)
7 motifs unique trouvés.
-----
```

Étape 8 : Vérification de la fonction effectuant le tri décroissant

```
-----  
Résultat du test de tri décroissant : RESULTAT CORRECT
```

## 4.Explciations des programmes de vérifications

Pour le programme de vérification de l’étape 6, vérification de la recherche de motifs, il s’agit d’une vérification sur le fonctionnement de la fonction en elle même.

Il s’agit d’une vérification semi automatique.

Le programme affiche le résultat attendu de la fonction ainsi que le résultat qu’elle produit grâce à la fonction elle même.

C’est-à-dire qu’un résultat correct et vérifié lui est fourni, et est affiché afin de voir les attentes. Ensuite le résultat obtenu avec la fonction est affiché.

Nous pouvons ensuite faire une comparaison manuelle pour vérifier le bon fonctionnement de la fonction.

Pour le programme de vérification de l’étape 7, vérification de la fonction de l’union, possède un fonctionnement calqué sur la fonction de vérification des motifs.

Cette fonction est aussi semi automatique.

Le programme affiche le résultat attendu de la fonction ainsi que le résultat qu'elle produit grâce à la fonction d'union.

C'est-à-dire qu'un résultat correct et vérifié lui est fourni, et est affiché afin de voir les attentes.

Ensuite le résultat obtenu avec la fonction est affiché.

Nous pouvons ensuite faire une comparaison manuelle pour vérifier le bon fonctionnement de la fonction.

Pour le programme de vérification de l'étape 8, vérification de la fonction de tri décroissant.

Celle-ci va vérifier le résultat produit par la fonction de tri.

Elle est complètement automatique.

Soit  $t$  : la taille du tableau, et  $n$  : l'indice actuelle.

La fonction va vérifier la condition :  $n \geq n+1$  jusqu'à atteindre  $t-1$ .

Et va afficher « RESULTAT CORRECT » si le tri a bien été effectué, sinon « RESULTAT INCORRECT » .

## **5. Tests des limites du programme avec et sans les fonctions à intégrer.**

Pour ma part, mon code n'affiche pas d'erreur, cependant cela devient très long pour une recherche dans un tableau de 200 000 000 lettres, cela prend plus que 10 minutes, j'ai donc arrêté le programme manuellement.

Et une trentaine de secondes pour 20 000 000 lettres aléatoires.

La limite où le programme indique une erreur avec le code à intégrer est 2 000 000 de lettre.

Une erreur est produite indiquant un Stack Overflow, c'est à dire une surcharge de la stack.

Cela est causé par un dépassement de l'utilisation de la stack(pile).

Ce dépassement de l'utilisation de la stack est dû à un trop gros nombre d'appel récursif une même fonction.

La limite de l'utilisation de la stack peut être repoussé grâce à des options de compilation.

## **6. Choses à savoirs**

-Le programme « algo\_rendu\_final.c », est à compiler avec l'option de compilation « -fsanitize=address » .

-Il y a deux programmes :

-algo\_rendu\_final.c, qui correspond à mon code.

-main\_code\_prof.c, qui correspond à mon code avec les modifications apportés pour répondre à la question 10.

-La question n°11 n'a pas été traité, cependant mon programme peut gérer des entrées plus grandes sans afficher de stack overflow.

-La librairie Regex a été utilisé pour faire la recherche de motif.

-Le code de toutes les étapes se trouvent dans le même fichier sauf l'étape d'intégration du code.

-Dans le cas où vous souhaiteriez me contacter, veuillez passer par ce mail svp :

[aichou.pro@gmail.com](mailto:aichou.pro@gmail.com) ou bien par Mattermost.