SOLUTION DÉTAILLÉE DE L’EXO 2-1 DE LA FEUILLE 2

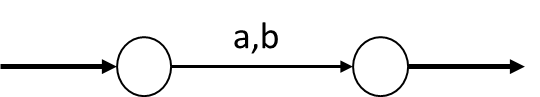
Expression rationnelle : E=((0+1) (0+1))\*+( (0+1) (0+1) (0+1))\*

L’automate asynchrone obtenu selon les règles :



Tant qu’il est possible de déterminiser cet automate asynchrone tel quel, il est plus simple de commencer par une simplification graphique, utilisant les règles de simplification suivantes :

1. La somme des caractères seuls peut être représentée comme une seule flèche :

a+b :

1. L’étoile d’un seul caractère a\* peut être représentée comme

a\* :

X\* :

**mais il faut introduire un ε additionnel** entre toutes deux étoiles qui risquent de s’entremêler. Par exemple, a\*X\* devient

1. (non programmable car prendrait trop de temps machine) : élimination de tous les **ε** manifestement inutiles.

Avec cela, on a les éléments suivants :

0+1 devient



(0+1)(0+1) devient



((0+1) (0+1) )\* devient

et donc l’expression initiale, ((0+1) (0+1))\*+( (0+1) (0+1) (0+1))\*, devient



Cet automate est facile à déterminiser. Il est suffisamment petit, et l’utilisation des **ε**-clôtures apporte assez peu. Voici la déterminisation avec et sans l’utilisation des **ε**-clôtures, et où on a profité du fait que les cibles de transitions en 0 et en 1 sont les mêmes :

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | déterminisation | | | | |
|  | en termes des  -clôtures | |  | sans les utiliser | |
|  | état | 0 ou 1 |  | état | 0 ou 1 |
| E/S | 0' | 2'5' | E/S | 014589 | 25 |
|  | 2'5' | 3'6' |  | 25 | 1368 |
| S | 3'6' | 2'7' | S | 1368 | 2478 |
| S | 2'7' | 3'5' | S | 2478 | 1358 |
| S | 3'5' | 2'6' | S | 1358 | 26 |
|  | 2'6' | 3'7' |  | 26 | 13478 |
| S | 3'7' | 2'5' | S | 13478 | 25 |

Nous avons obtenu un ADC à 7 états.

Minimisation :

Initialisation : 0 ={T,NT} où T=(0',3'6',2'7',3'5',3'7'), NT=(2'5',2'6').

Itération 1 :

|  |  |  |  |
| --- | --- | --- | --- |
|  | état | 0 ou 1 | sous 0 |
| T | 0' | 2'5' | NT |
| 3'6' | 2'7' | T |
| 2'7' | 3'5' | T |
| 3'5' | 2'6' | NT |
| 3'7' | 2'5' | NT |
|  |  |  |  |
| NT | 2'5' | 3'6' | T |
| 2'6' | 3'7' | T |

En résultat, 1 ={NT,A,B} où A=(0',3'5',3'7'), B=(3'6',2'7).

Itération 2 :

|  |  |  |  |
| --- | --- | --- | --- |
|  | état | 0 ou 1 | sous 1 |
| A | 0' | 2'5' | NT |
| 3'5' | 2'6' | NT |
| 3'7' | 2'5' | NT |
|  |  |  |  |
| B | 3'6' | 2'7' | B |
| 2'7' | 3'5' | A |
|  |  |  |  |
| NT | 2'5' | 3'6' | B |
| 2'6' | 3'7' | A |

En résultat, 2={A,(3'6'),(2'7'),(2'5'),(2'6')}.

Itération 3 :

|  |  |  |  |
| --- | --- | --- | --- |
|  | état | 0 ou 1 | sous 2 |
| A | 0' | 2'5' | (2'5') |
| 3'5' | 2'6' | (2'6') |
| 3'7' | 2'5' | (2'5') |

En résultat, 3={(0',3'7'),(3'5'),(3'6'),(2'7'),(2'5'),(2'6')}.  
Les états 0’ et 3’7’ ne pourront pas se séparer car ils appartiennent   
au même groupe et ont la même transition. Donc, 3= fin.  
Les états de l’AM : ={(0',3'7'),(3'5'),(3'6'),(2'7'),(2'5'),(2'6').   
Seuls les états 0’ et 3’7’ se fusionnent, et il est très facile   
de dessiner l’automate minimal :

L’automate minimal consiste donc en 6 états, avec les sorties en positions 0,2,3,4.

Maintenant nous pouvons réexaminer l’ER d’origine et établir un lien direct entre sa structure et l’AM obtenu.

Nous avons l’ER :

E=((0+1) (0+1))\*+( (0+1) (0+1) (0+1))\*

L’alphabet est A={0,1}, mais ni 0 ni 1 ne se rencontrent jamais dans une autre combinaisons que 0+1.

Appelons c=0+1. Alors E=(c2)\* + (c3)\* : c’est un ensemble consistant en mots de longueur multiple de 2 ou multiple de 3, dont les caractères sont 0 ou 1 dans n’importe quelle position.

Les longueurs sont donc

0 6 12  
~~1 7 13~~2 8 14  
3 9 15  
4 10 16  
~~5 11 17~~ etc., où j’ai barré les longueurs interdites.

On voit que ces longueurs, permises et interdites, sont toutes définies modulo 6. La longueur k telle que k mod 6=0 ou 2 ou 3 ou 4 est permise, et si k mod 6 = 1 ou 5, non.

C’est pourquoi il nous faut dans l’automate un cycle de longueur 6 avec les sorties en 0, 2, 3 et 4.

On pourrait créer un automate de longueur 12 ou 18 avec des sorties en multiples de 2 et de 3, mais il ne serait pas minimal.

Le minimal en l’occurrence a la longueur 2x3. Est-ce une règle générale ?

Non : si je veux un automate minimal reconnaissant (c4)\* + (c6)\*, je n’ai pas besoin de 24 états. Il suffit en voir 12, car 12 est divisible et par 4 et par 6. Voici l’automate minimal reconnaissant

((0+1) (0+1) (0+1) (0+1))\*+( (0+1) (0+1) (0+1) (0+1) (0+1) (0+1))\*

Le nombre minimal divisible par n et par m s’appelle PPCM(n,m). Nous avons donc obtenu, semble-t-il, le résultat que quels que soient deux nombres n et m, l’automate minimal reconnaissant (cn)\*+(cm)\* consiste en PPCM(n,m) états avec des sorties en multiples de n et multiples de m. Ceci est presque vrai. Il y a une exception :

Considérons (c3)\*+(c6)\*. PPCM(3,6)=6. Faut-il 6 états pour un automate minimal reconnaissant (c3)\*+(c6)\* ? Non, car (c6)\* ⊂ (c3)\* , et donc (c3)\*+(c6)\* =(c3)\* . En effet, (c3)\*={c3, c6, c9, c12, …} et toutes les puissances de c6 sont déjà dedans. Donc si un des deux nombres est le multiple de l’autre, on peut tout simplement barrer ce terme.

De cette façon, par exemple, construisons l’automate minimal reconnaissant

((a+b+c) (a+b+c) (a+b+c))\*+( (a+b+c) (a+b+c) (a+b+c) (a+b+c) (a+b+c) (a+b+c))\*

+( (a+b+c) (a+b+c) (a+b+c) (a+b+c) (a+b+c))\*=(d3)\*+(d6)\*+(d5)\* où d=a+b+c.

On peut barrer (d6)\* car il ne contribue rien par rapport à (d3)\*. On a alors n=3, m=5. L’AM aura 15 états, avec des sorties en positions 0,3,5,6,9,10,12.

