# 5 AUTOMATES

## RAPPELS DE COURS

## 5.1 HISTORIQUE

Alan Turing a décrit à 1936 un modèle de « machine idéale » auquel il a laissé son nom. D'autres modèles furent proposés, qui sont tous équivalents à la **machine de Turing**. Church a démontré dans sa thèse que ces modèles sont les plus généraux possibles.

Ces travaux amenèrent à distinguer entre les fonctions qui sont calculables en théorie de celles qui ne le seraient même pas. (Toute fonction calculable peut être calculée par la machine de Turing en temps (nombre d'opérations) fini).

Un modèle plus simple mais fort utile est celui d'un **automate fini**. Un automate fini est une machine abstraite constituée d'**états** et de **transitions**. Cette machine est destinée à traiter des **mots** fournis en entrée : l'automate passe d'état en état, suivant les transitions, à la lecture de chaque lettre de l'entrée. L'automate est dit « fini » car il possède un nombre fini d'états distincts : il ne dispose donc que d'une mémoire bornée, indépendamment de la taille de la donnée sur laquelle on effectue les calculs.

Un automate fini forme un graphe orienté étiqueté, dont les états sont les sommets et les transitions les arcs étiquetés.

Les automates finis fournissent un outil de construction d'algorithmes particulièrement simple.

Il existe plusieurs types de machines à états finis. Les « **accepteurs** » produisent en sortie une réponse « oui » ou « non », c'est-à-dire qu'ils acceptent (oui) ou rejettent (non) l'entrée. Les systèmes de reconnaissance classent l'entrée par catégorie. Les capteurs produisent un certain résultat en fonction de l'entrée.

Les automates finis peuvent caractériser des langages (c'est-à-dire des ensembles de mots) finis (le cas standard), des langages de mots infinis (automates de Rabin, automates de Büchi), ou encore divers types d'arbres (automates d'arbres).

Les machines à états finis se rencontrent également dans les circuits digitaux, où l'entrée, l'état et le résultat sont des vecteurs de bits de taille fixe (machines de Moore et machines de Mealy). Dans les machines de Mealy, les actions (sorties) sont liées aux transitions, tandis que dans les machines de Moore, les actions sont liées aux états.

## 5.2 QUELQUES DÉFINITIONS

Pour définir un automate fini, on a besoin des éléments suivants :

- **Un alphabet** $A$ qui est un ensemble fini d'objets qu'on appelle « lettres » ou « caractères » mais qui peuvent, selon le cas, être de n'importe quel genre (instructions machine, état civil d'une personne, type d'objet géométrique etc.)
- Des **mots** sont des séquences ordonnées de caractères de l'alphabet. Dans nos applications, on aura affaire qu'à des mots de longueur finie, mais il n'est pas interdit d'utiliser des mots de longueur infinie (automates de Rabin, automates de Büchi). La longueur d'un mot est le nombre de lettres le constituant.
  *Exemple* : $w = abacda$ – mot de longueur six sur l'alphabet latin (ou bien sur l'alphabet $\{a, b, c, d\}$).
- **Un mot vide** est noté par $\varepsilon$ ou par 1. C'est le seul mot de longueur nulle.
- On note $A^*$ **l'ensemble** de tous les mots sur l'alphabet $A$, y compris le mot vide.
- Un automate fini sur l'alphabet $A$ est donné par un quadruplet $(Q, I, T, E)$ où :
  - $Q$ est un ensemble **fini** d'états de l'automate ;
  - $I \subset Q$ est un ensemble d'états initiaux ;
  - $T \subset Q$est un ensemble d'états terminaux ;
  - $E \subset Q \times \left(A \cup \{\varepsilon\}\right) \times Q$ est un ensemble de triplets $(p.a.q)$ appelés les flèches ou les transitions de l'automate.

### *Remarque*

Nous allons temporairement oublier la notion du mot vide, jusqu'à l'introduction explicite d'automates asynchrones. Dans tous les exemples que nous allons traiter jusque ce point-là, on pourra donc définir l'ensemble $E$ comme faisant partie de $Q \times A \times Q$.

## 5.3 L'INTERPRÉTATION INTUITIVE

À chaque instant l'automate se trouve dans l'un des états $p$ de l'ensemble d'états $Q$. Il lit alors l'une des lettres $a$ de l'alphabet $A$ (la lettre suivant du mot qui vient à l'entrée) et passe dans un état $q$ tel que $(p.a.q)$ soit une flèche.

***Exemple***

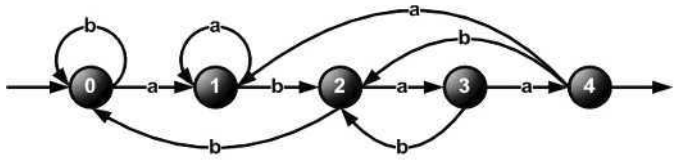


**Figure 5.1**

$$A = \{a, b\};\ Q = \{0,1,2,3,4\};\ I = \{0\};\ T = \{4\}$$

La lecture du mot $w$ se termine dans l'état 4 ssi $w$ se termine par $abaa$.
Le même automate peut être représenté par une **table de transitions** :

Tableau 5.1

| | État | État résultant | |
|---|---|---|---|
| | | en lisant a | en lisant b |
| (entrée) | 0 | 1 | 0 |
| | 1 | 1 | 2 |
| | 2 | 3 | 0 |
| | 3 | 4 | 2 |
| (sortie) | 4 | 1 | 2 |

### *Remarque*

Si vous regardez cet automate de près, vous verrez que nous avons libellé les états de telle façon que l'automate se trouve dans l'état numéro $i$ ssi le plus long **suffixe du mot déjà lu** qui est en même temps un **préfixe de** $abaa$**,** est de **longueur** $i$.

On peut montrer que cet automate réalise de façon optimale l'algorithme de recherche de la séquence $abaa$ dans un texte : il résout pour ce mot l'algorithme du « *string matching* » utilisé par exemple dans les éditeurs de textes.

## La lecture

L'automate prend un mot (constitué de symboles de son alphabet) en entrée et démarre dans son ou ses état(s) initial(-aux). Il parcourt ensuite le mot de gauche à droite : si l'automate se trouve dans un état $p$ et le symbole à lire est $a$**,** alors s'il existe(nt) un ou des état(s) $q_i$ tels que la transition $(p.a.q_i)$ existe, il passe aux états $q_i$, et de suite. (S'il y a plus d'un état $q_i$, on considère chacun d'eux séparément). La lecture se termine soit si la lettre suivante ne peut pas être lue (il n'y a pas de transition y correspondant), soit si on a atteint la fin du mot. Si, à la fin de la lecture, l'automate est dans un état final (accepteur), on dit qu'il accepte l'entrée ou qu'il la reconnaît. Autrement, on dit qu'il la rejette.

## Notation

Pour un mot $w \in A^*$ on note $p \xrightarrow{w} q$ s'il existe un chemin de l'état $p$ à l'état $q$ obtenu en lisant $w$.

## Définitions formelles

$\forall a \in A$, on a $p \xrightarrow{a} q$ ssi il existe une flèche $(p.a.q) \in E$.

Ensuite, pour $u \in A^*$ et $a \in A$ on a $p \xrightarrow{ua} q$ ssi $\exists r \in Q$ tq $p \xrightarrow{u} r$et $r \xrightarrow{a} q$.

Quand on a $p \xrightarrow{w} q$ on va dire qu'il existe un chemin de $p$ à $q$ d'étiquette $w$.

© Dunod – La photocopie non autorisée est un délit

On peut facilement voir (par récurrence) que pour $u, v \in A^*$ on a :

$$p \xrightarrow{uv} q \text{ ssi } \exists r \in Q \text{ tq } p \xrightarrow{u} r \text{ et } r \xrightarrow{v} q.$$

Un mot $w \in A^*$ est donc reconnu par l'automate fini s'il existe un chemin $i \xrightarrow{uv} t$ où $i \in I$ et $t \in T$ – c.-à-d. qu'en partant d'un état initial et qu'en lisant le mot $w$, on atteint un état terminal à la fin de la lecture.

Le **langage** $L$ **reconnu** par l'automate fini est l'ensemble de tous les mots reconnus.

***Exemple d'exercice type***

Construire un automate fini reconnaissant les entiers écrits en base deux et divisibles par trois.

**Solution**

Ajout d'un 0 à la fin d'un nombre binaire le multiplie par 2.
Ajout d'un 1 à la fin d'un nombre binaire le multiplie par 2 et lui ajoute 1.
Marquant les états par le reste de la division entière par 3 :

Tableau 5.2

| $N$ | $N$ mod3 | ajout d'un 0 à la fin | reste | ajout d'un 1 à la fin | reste |
|---|---|---|---|---|---|
| $3n$ | 0 | $6n$ | 0 | $6n + 1 = 3 \times 2n + 1$ | 1 |
| $3n + 1$ | 1 | $6n + 2 = 3 \times 2n + 2$ | 2 | $6n + 3 = 3 \times (2n + 1)$ | 0 |
| $3n + 2$ | 2 | $6n + 4 = 3 \times (2n + 1) + 1$ | 1 | $6n + 5 = 3 \times (2n + 1) + 2$ | 2 |

Notons les états par le reste de la division entière par 3. On obtient :

Tableau 5.3

| État | État résultant | |
|---|---|---|
| | 0 | 1 |
| 0 | 0 | 1 |
| 1 | 2 | 0 |
| 2 | 1 | 2 |

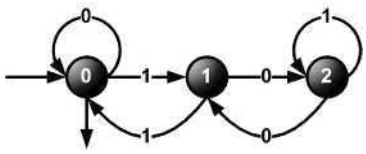


Figure 5.2

En fait, cet automate est bon pour reconnaître des nombres en écriture binaire avec n'importe quel reste de la division entière par 3, au prix de choisir quel état est terminal. L'état terminal 0 correspond au reste 0 (divisibilité par 3) ; l'état terminal 1, au reste 1 ; l'état terminal 2, au reste 2.

### 5.3.1 Automates déterministes

On distingue des automates finis déterministes et non déterministes. L'automate que nous avons montré précédemment est déterministe.

Voici l'exemple d'un automate ***non déterministe*** :

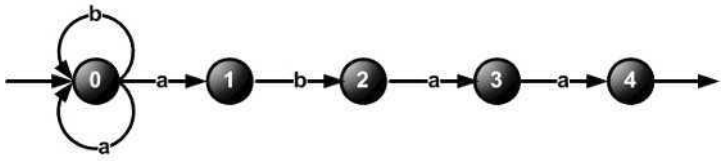


**Figure 5.3** **Cet automate reconnaît tous les mots qui se terminent par abaa. Il est très facile à construire.**

Le fait qu'il n'est pas déterministe se manifeste en ce que, en lisant ***a*** à partir de l'état 0, on peut aller et à l'état 0, et à l'état 1, ce qui ne serait pas possible pour un automate déterministe.

On va montrer dans ce qui suit que l'on peut toujours construire, à partir d'un automate fini quelconque, un autre automate qui est déterministe et qui lui est équivalent (reconnaît le même langage).

#### Automate déterministe, définition

L'automate $C = (Q, I, T, E)$ est déterministe ssi pour $\forall p \in Q$ et $\forall a \in A$ il existe au plus un état $q \in Q$ tq $(p.a.q)$, et qu'il y ait un seul état initial : $I = \{i\}$.

On peut donc caractériser un automate déterministe $C$ par un quadruplet $(Q, i, T, E)$ où $i$ est l'état initial.

Si la transition $(p.a.q)$ existe, on notera $p.a = q$**.** Si elle n'existe pas, on conviendra que $p.a$n'a pas de valeur.

On peut facilement vérifier par récurrence sur la longueur des mots que pour un automate déterministe $C$**,** pour $\forall p \in Q$ et $\forall$ mot $u \in A^*$, il existe au plus un état $q \in Q$ tq $p \xrightarrow{u} q$.

On notera alors $p.u = q$en convenant que $p.u$ n'est pas défini quand il n'existe pas de tel état $q$; et on aura pour deux mots $u, v \in A^*$:

$$(p.u).v = p.(u.v)$$

#### Déterminisation

Soit $C = (Q, I, T, E)$ un automate fini.

Notons $\Pi$ l'ensemble des parties de $Q$. C'est un ensemble fini puisque si $Q$ a $n$ éléments, $\Pi$ en a $2^n$.

Un automate déterministe $D$ correspondant à $C$ prend :

- comme ensemble d'états un sous-ensemble $P$ de l'ensemble $\Pi$ des parties de $Q$,

© Dunod – La photocopie non autorisée est un délit

- comme unique état initial l'ensemble $I$ des états initiaux de $C$,
- comme ensemble d'états terminaux l'ensemble $\varsigma = \{u \in P | u \cap T = \emptyset\}$ des parties de $Q$ qui contiennent au moins un état terminal de $C$,
- comme flèches l'ensemble des $(u.a.v)$ où $u \in P$ et $v$ est l'ensemble de tous les états $q \in Q$ tels qu'il existe un état $p$ dans $u$ avec $(p.a.q) \in E$.

Cet algorithme formel s'explique le plus facilement sur un exemple.

***Exemple de déterminisation***

Prenons l'automate non déterministe précédent :

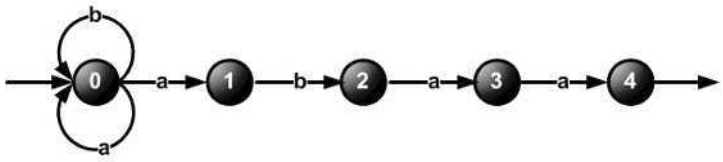


Figure 5.4

- *Déterminisation*

Tableau 5.4

| | État | États résultant | |
|---|---|---|---|
| | | en lisant a | en lisant b |
| (entrée) | (0) | (0,1) | (0) |
| | (0,1) | (0,1) | (0,2) |
| | (0,2) | (0,1,3) | (0) |
| | (0,1,3) | (0,1,4) | (0,2) |
| (sortie) | (0,1,4) | (0,1) | (0,2) |

- *Explications*

On commence par l'état initial qui dans ce cas particulier coïncide avec l'état initial de l'automate non déterministe (0), car notre automate non déterministe n'en a qu'un seul. (S'il en avait plusieurs, on les aurait regroupés pour former l'état initial de l'automate déterministe).

À partir de cet état (0), en $a$ on va vers les états (0), (1) de l'automate initial. Pour l'automate déterministe, on les regroupe dans l'état qu'on appelle (0,1). En $b$, on va vers l'état (0) de l'automate initial. Donc, (0) est lui aussi un état de l'automate déterminisé.

Chaque fois qu'on obtient un état de l'automate déterminisé, on le met dans la colonne de gauche pour agir sur cet état par les lettres de l'alphabet. Ainsi on obtient la deuxième ligne, où figure un nouvel état (0,2). On procède ainsi tant qu'il y reste des états non traités.

Les états finaux sont ceux qui contiennent un état final de l'automate initial. Dans notre cas, il n'y en a qu'un : (0,1,4).

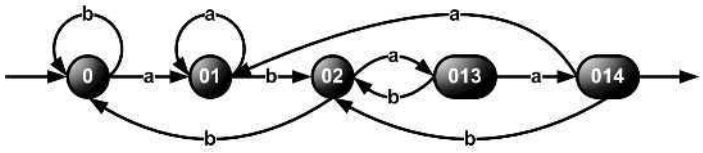


**Figure 5.5** **Automate déterminisé**

Voici l'automate déterminisé :

La démonstration montrant que et $C$ reconnaissent le même langage se fait par récurrence.

Un automate déterministe peut être ou ne pas être **complet.**

### Automate déterministe complet (définition)

Pour $\forall u \in P$ et $\forall a \in A$ , $\exists v \in P$ tel que $u.a = v$.

C'est-à-dire que de chaque état sortent des flèches avec toutes les étiquettes possibles.

L'automate $D$ de l'exemple précédent est un automate complet.

N'importe quel automate non déterministe est équivalent à un automate déterministe et complet : il suffit, dans l'automate déterminisé, d'introduire un état poubelle s'il n'est pas déjà complet.

***Exemple***

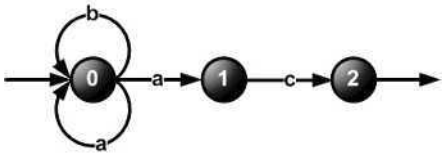


**Figure 5.6**

- *Déterminisation*

**Tableau 5.5**

| | État | États résultant | | |
|---|---|---|---|---|
| | | en lisant a | en lisant b | en lisant c |
| (entrée) | (0) | (0,1) | (0) | – |
| | (0,1) | (0,1) | (0) | (2) |
| (sortie) | (2) | – | – | – |

Nous remplaçons les traits (« pas de transition ») dans ce tableau par un nouvel état appelé poubelle (P), qui a la propriété que toutes les transitions à partir de cet état reviennent sur lui-même :

© Dunod – La photocopie non autorisée est un délit

Tableau 5.6

| | État | États résultant | | |
|---|---|---|---|---|
| | | en lisant a | en lisant b | en lisant c |
| (entrée) | (0) | (0,1) | (0) | P |
| | (0,1) | (0,1) | (0) | (2) |
| (sortie) | (2) | P | P | P |
| | P | P | P | P |

Voici donc l'automate déterminisé complet, dans lequel (0) est resté l'état initial, et le seul état terminal reste (2) :

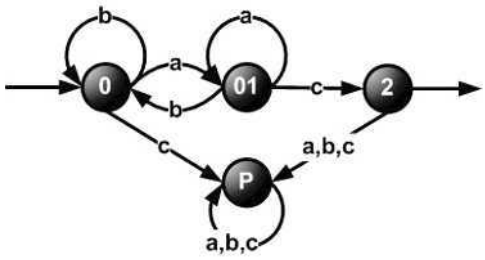


Figure 5.7

### *Remarque*

Il n'y a pas trop de sens de parler d'un automate non déterministe complet ou pas complet. De même, même si introduire un « état poubelle » dans un automate non déterministe est possible et ne nuit pas à son fonctionnement, ceci n'est pas utile pour la déterminisation de cet automate. Normalement, on n'introduit un état poubelle si besoin est qu'une fois l'automate est devenu déterministe.

Un automate est ***accessible*** si n'importe quel état est accessible à partir d'un état initial (c.à.d. s'il existe un chemin y menant à partir d'un état initial).

Un automate est ***coaccessible*** si à partir de n'importe quel état on peut accéder à un état terminal.

Accessible + coaccessible = ***émondé***.

Un ensemble de mots $X$ est un langage ***reconnaissable*** ssi il existe un automate fini $D$ qui le reconnaît.

Si $X$ est un langage reconnaissable sur l'alphabet $A$, on peut le reconnaître avec un automate déterministe et complet, car l'automate $D$ figurant dans la définition du langage reconnaissable est toujours équivalent à un automate déterministe et complet $F$.

Écrivant $F = (Q, i, T, E)$, on a alors $w \in X$ ssi $i.w \in T$ et donc $w \notin X$ ssi $i.w \notin T$.

• *Le complément d'un langage reconnaissable*

Le complément d'un langage reconnaissable (l'ensemble de mots sur le même alphabet n'appartenant pas au langage en question) est encore reconnaissable.

Soit $D$ un automate déterministe et complet reconnaissant le langage $X$.

Pour construire un automate reconnaissant le complément de $X$, il suffit de prendre comme ensemble d'états terminaux le complément de l'ensemble d'états terminaux de $D$.

***Exemple***

Construisons un automate sur $A = \{a, b\}$ qui reconnaît l'ensemble des mots n'ayant pas $aba$ en facteur.

On commence par la construction d'un automate (non déterministe) reconnaissant tous les mots ayant $aba$ en facteur.

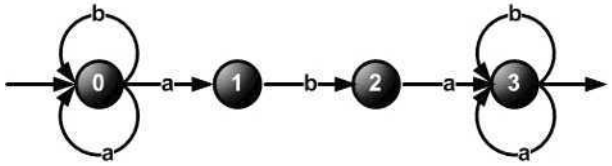


Figure 5.8

Appliquons l'algorithme de déterminisation :

Tableau 5.7

| | État | États résultant | |
|---|---|---|---|
| | | en lisant a | en lisant b |
| (initial) | (0) | (0,1) | (0) |
| | (0,1) | (0,1) | (0,2) |
| | (0,2) | (0,1,3) | (0) |
| (terminal) | (0,1,3) | (0,1,3) | (0,2,3) |
| (terminal) | (0,2,3) | (0,1,3) | (0,3) |
| (terminal) | (0,3) | (0,1,3) | (0,3) |

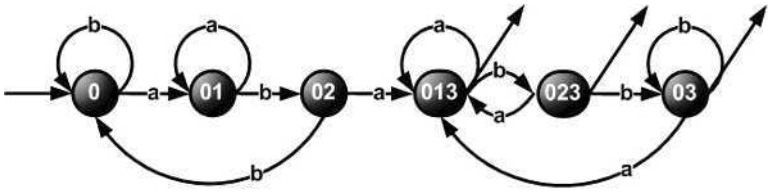


Figure 5.9

Cet automate déterministe et complet reconnaît tous les mots ayant ***aba*** en facteur. Il a trois états terminaux : (0,1,3), (0,2,3) et (0,3).

© Dunod – La photocopie non autorisée est un délit

Maintenant pour obtenir un automate (lui aussi déterministe et complet) reconnaissant tous les mots **qui n'ont pas *aba*** en facteur, il suffit de faire de tous les états terminaux, des états non terminaux, et vice versa :

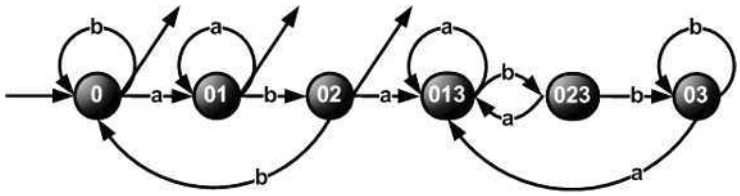


Figure 5.10

Ici, on a obtenu un automate complet sans qu'il y ait besoin de le compléter en introduisant l'état poubelle. Évidemment, ceci n'est pas toujours le cas. **Si on a affaire à un automate non complet, on n'a pas le droit de remplacer directement les états terminaux par des états non terminaux et vice versa : il faut d'abord le compléter.** Alors l'état poubelle (qui ne peut pas être terminal) devient un état terminal de l'automate reconnaissant le complément du langage.

## Minimisation

Pour tout langage reconnaissable il existe ***le plus petit*** (c.-à-d. contenant le plus petit nombre d'états) automate déterministe qui le reconnaît, et il est ***unique*** : c'est l'automate minimal du langage.

### • *Algorithme de minimisation par la méthode des équivalences*

Il est basé sur la notion de partitionnement de l'ensemble d'états de l'automate.

**Principe** Tout d'abord, si l'automate à minimiser n'est pas complet, *il faut lui ajouter l'état poubelle pour qu'il le devienne.* Sinon on risque de faire une erreur grave, qui se manifeste facilement au cas d'un automate déterministe non complet dont tous les états sont des états terminaux (essayez le voir vous-mêmes après avoir compris l'algorithme de minimisation).

On sépare tous les états de l'automate déterministe initial en deux groupes : terminaux et non-terminaux. Puis, on analyse la table des transitions en marquant vers quel groupe va chaque transition. On repartitionne les groupes selon les patterns des transitions en terme de groupes. On répète ce processus en utilisant cette nouvelle partition, et on réitère jusqu'à ce qu'on arrive à ne plus pouvoir partitionner. Les groupes restants forment l'automate minimal. On appelle souvent les itérations les étapes.

**Description du processus de partitionnement itératif** Soit un automate fini déterministe complet $D = (Q, i, T, E)$.

**Résultat à obtenir** : Un automate fini déterministe complet $D'$ qui reconnaît le même langage que $D$ et qui a aussi peu d'états que possible.

**Méthode** Construire une partition initiale $\Theta_0$ de l'ensemble des états avec deux groupes : les états terminaux $T$ et les états non-terminaux $Q - T$.

Procédure applicable à la partition courante $\Theta_i$, à commencer par $\Theta_0$ :

```
POUR chaque groupe G de Θi  FAIRE
début
partitionner G en sous-groupes de manière que deux états e et t de G
soient dans le même sous-groupe ssi pour tout symbole a ∈ A, les états e
et t ont des transitions sur a vers des états du même groupe de Θi ;
/* au pire, un état formera un sous-groupe par lui-même */
remplacer G dans par tous les sous-groupes ainsi formés ;
on obtient la partition de l'étape suivant, Θi+1 ;
fin
Si Θi+1 = Θi   alors Θfinal = Θi et continuer à l'étape 4.
Sinon, répéter l'étape (2) avec Θi+1 comme partition courante.
```

Choisir un état dans chaque groupe de la partition $\Theta_{final}$ en tant que représentant de ce groupe.

Les représentants seront les états de l'automate fini déterministe réduit $D'$.

Soit $e$ un état représentatif, et supposons que dans $D$ il y a une transition $e \xrightarrow{a} t$.

Soit $r$ le représentant du groupe de $t$ ($r$ peut être égal à $t$).

Alors $D'$ a une transition de $e$ vers $r$ sur $a$ ($e \xrightarrow{a} r$).

L'état initial de $D'$ est le représentant du groupe qui contient l'état initial $i$ de $D$ ; les états terminaux de $D'$ sont des représentants des états de $T$.

Pour tout groupe $G$ de $\Theta_{final}$, soit $G$ est entièrement composé d'états de $T$, soit $G$ n'en contient aucun.

### *Remarque*

En réalité, il est parfois plus facile, au lieu de choisir un représentant, marquer les groupes finaux comme tels, ou bien les renommer par, par exemple, A, B, C ... , et remplacer dans les transitions tout état par le nom de son groupe :
Si $e \xrightarrow{a} r$ , $e \in A$, $r \in B$ où les groupes $A, B \in \Theta_{final}$, alors dans l'automate minimisé on a $A \xrightarrow{a} B$ où maintenant on considère $A$ et $B$ comme états de l'automate minimisé. Cette remarque deviendra tout à fait claire à la fin de l'exemple suivant.

***Exemple 1***

Minimisons l'automate déterministe complet :

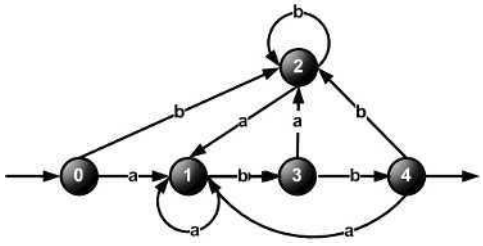


Figure 5.11

© Dunod – La photocopie non autorisée est un délit

décrit par le tableau de transitions suivant :

**Tableau 5.8**

| | État | États résultant | |
|---|---|---|---|
| | | en lisant a | en lisant b |
| (initial) | 0 | 1 | 2 |
| | 1 | 1 | 3 |
| | 2 | 1 | 2 |
| | 3 | 2 | 4 |
| (terminal) | 4 | 1 | 2 |

L'unique état terminal est l'état 4.
Donc, la partition initiale : $\Theta_0 = \{(0,1,2,3),(4)\}$.
Notons les groupes non terminal et terminal : $I = \{(0,1,2,3)\}$ et $II = \{(4)\}$.
On ne peut pas, évidemment, essayer de séparer le groupe II qui consiste déjà en un seul état.
On regarde donc dans quel groupe tombent les transitions à partir des états du groupe I :

**Tableau 5.9**

| État | Groupes résultant | |
|---|---|---|
| | en lisant a | en lisant b |
| 0 | I | I |
| 1 | I | I |
| 2 | I | I |
| 3 | I | II |

Donc, le groupe $I$ se sépare en deux, et on a $\Theta_1 = \{(0,1,2),(3),(4)\}$.
Notons les groupes (en recyclant la notation) : $I = \{(0,1,2)\},\ II = \{(3)\},\ III = \{(4)\}$.
Maintenant essayons de séparer le groupe $I$ en regardant où tombent les transitions à partir de ses états dans les termes de la séparation $\Theta_1 = \{(0,1,2),(3),(4)\}$.

**Tableau 5.10**

| État | Groupes résultant | |
|---|---|---|
| | en lisant a | en lisant b |
| 0 | I | I |
| 1 | I | II |
| 2 | I | I |

Donc, le groupe $I$ se sépare en deux, et on a $\Theta_2 = \{(0,2),(1),(3),(4)\}$.
Notons les groupes (en recyclant la notation) :

$$I = \{(0,2)\},\ II = \{(1)\},\ III = \{(3)\},\ IV = \{(4)\}.$$

Essayons de séparer le groupe $I$ en regardant où tombent les transitions à partir de ses états dans les termes de la séparation $\Theta_2 = \{(0,2),(1),(3),(4)\}$.

Tableau 5.11

| État | Groupes résultant | |
|---|---|---|
| | en lisant a | en lisant b |
| 0 | II | I |
| 2 | II | I |

Le groupe I ne se sépare pas, $\Theta_3$reste identique à la séparation de l'étape précédent :

$$\Theta_3 = \Theta_2 = \{(0,2), (1), (3), (4)\}.$$

Fin de la procédure itérative de séparation.
Voici les itérations qu'on a effectuées :
Étape 1 : $\Theta_{courant} = \{(0,1,2,3), (4)\}$

$$\Theta_{new} = \{(0,1,2), (3), (4)\}$$

Étape 2 : $\Theta_{courant} = \{(0,1,2), (3), (4)\}$

$$\Theta_{new} = \{(0,2), (1), (3), (4)\}$$

Étape 3 : $\Theta_{courant} = \{(0,2), (1), (3), (4)\}$

$$\Theta_{new} = \{(0,2), (1), (3), (4)\} = \Theta_{final}$$

Passons aux représentants :

Tableau 5.12

| Groupe | Représentant |
|---|---|
| I | 0 ou 2 |
| II | 1 |
| III | 3 |
| IV | 4 |

Il devient clair maintenant qu'on peut marquer les états de l'automate minimisé soit par un représentant, soit par les chiffres I, II, III, IV suivant le groupe du dernier étape, soit par le contenu du groupe (dans ce cas, l'état initial sera marqué (0,2) ; cette dernière solution est pratique tant que le groupe consiste en peu d'états).
L'automate minimisé :

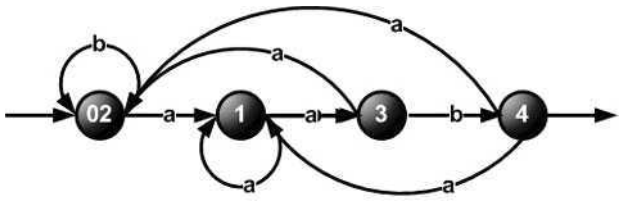


Figure 5.12

© Dunod – La photocopie non autorisée est un délit

On peut maintenant poser la question suivante : pourquoi les états 0 et 2 sont resté ensemble après la minimisation ? Qu'y a-t-il de spécial concernant ces deux états par rapport aux autres ? Regardons à nouveau l'automate initial :

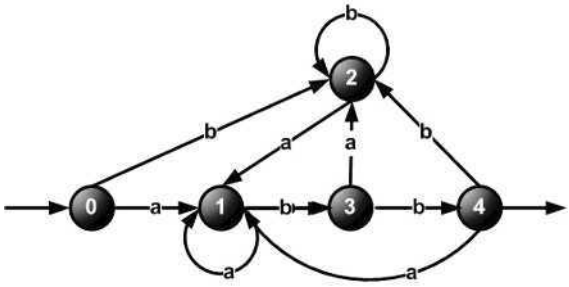


Figure 5.13

et introduisons la terminologie suivante :
*on dit que la chaîne $w$ distingue l'état $s$ de l'état $t$ si quand on commence dans l'état $s$ et lit $w$ on arrive dans un état terminal, et si on commence dans $t$ et lit $w$ on arrive dans un état non terminal ou vice-versa.*

Ici, tous les états peuvent être distingués par une chaîne ou une autre, sauf les états 0 et 2. C'est facile à voir : et à partir de 0, et à partir de 2 on arrive en 1 en lisant un nombre quelconque (y compris zéro) de $b$ et un $a$; puis, comme on est dans le même état (1), il ne nous reste que les mêmes chaînes pour arriver à l'état final (ici, il y en a un seul). Donc, les états non distinguables se fondent dans un même état (en l'occurrence (0,2)) de l'automate minimisé.

En fait, on peut montrer que l'algorithme de minimisation qu'on a expliqué, est basé sur la recherche des tous les groupes qui peuvent être distingués par une chaîne ou $I = \{1\}$, $T = \{1,2\}$ une autre.

***Exemple 2***

Voici un automate déterministe complet avec deux états terminaux, avec l'alphabet A $= \{0, 1\}$ :

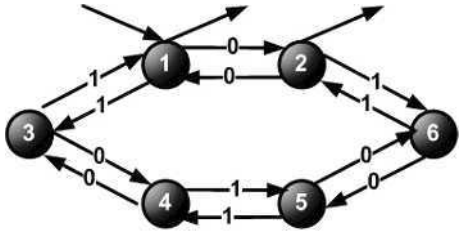


Figure 5.14

La procédure de minimisation donne :

$I = \{1\}, \quad T = \{1,2\}$

$\Theta_0 = \{(1,2),(3,4,5,6)\} = (I, II\}$
$\Theta_1 = (\{(1,2).(3,6),(4,5)\} = \{I, II, III\}$ (en recyclant les chiffres romains)
$\Theta_2 = \Theta_1$
Voici l'automate minimisé :

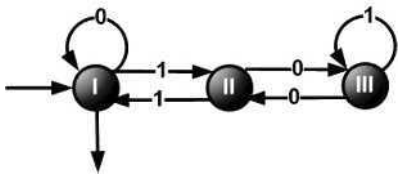


Figure 5.15

### 5.3.2 Automate asynchrone

Souvenons-nous que jusqu'à ce point nous avons évité **le mot vide**.

Un automate fini dans lequel certaines flèches sont étiquetées par le mot vide (« ***ε-transitions*** ») s'appelle **automate asynchrone**. L'ensemble des flèches vérifie donc $E \subset Q \times \left(A \cup \{\varepsilon\}\right) \times Q$.

**Proposition** : pour tout automate asynchrone, il existe un automate fini ordinaire (c.à.d. sans $\varepsilon$-transitions) qui reconnaît le même langage.

Comme tout automate non déterministe est équivalent à un automate déterministe, il en suit que **tout automate asynchrone est équivalent à un automate déterministe**.

Il existe un algorithme de suppressions de $\varepsilon$-transitions.

Pour un automate asynchrone $C = (Q, I, T, E)$ avec $E \subset Q \times \left(A \cup \{\varepsilon\}\right) \times Q$ nous allons construire un automate $B = (Q, I, T, F)$ qui ne diffère de $C$ que par l'ensemble de ses flèches et ce, suivant une règle bien déterminée :

- par définition on a $(p.a.q) \in F$ s'il existe un chemin

$$c : p_0 \xrightarrow{\varepsilon} p_1 \xrightarrow{\varepsilon} \ldots \xrightarrow{\varepsilon} p_n \xrightarrow{a} q_0 \xrightarrow{\varepsilon} q_1 \xrightarrow{\varepsilon} \ldots \xrightarrow{\varepsilon} q_m$$

$$\text{et } p_0 = p \wedge q_m = q$$

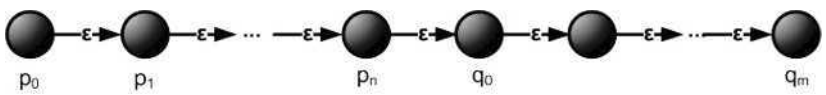


Figure 5.16

- Le nombre de $\varepsilon$-transitions à droite ou à gauche de la transition $p_n \xrightarrow{a} q_0$ peut être nul (dans ce cas $p_n = p_0$ou $p_n = p_0$ ).

© Dunod – La photocopie non autorisée est un délit

***Exemple***

a) Prenons un automate asynchrone :

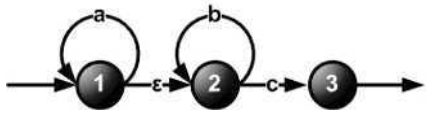


Figure 5.17 **(A1)**

Ici, suivant le raisonnement précédent, il existent les chemins : $1a1,\ 1a2,\ 1b2,\ 1c3,\ 2b2,\ 2c3$.

b) Donc cet automate est équivalent à l'automate non déterministe suivant :

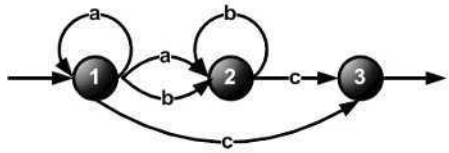


Figure 5.18 **(A2)**

c) Maintenant on peut le déterminiser :

Tableau 5.13

| État | États résultant | | |
|---|---|---|---|
| | en lisant a | en lisant b | en lisant c |
| 1 | 1,2 | 2 | 3 |
| 1,2 | 1,2 | 2 | 3 |
| 2 | – | 2 | 3 |
| 3 | – | – | – |

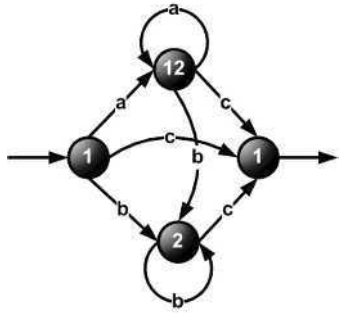


Figure 5.19 **(A3)**

ou bien, en ajoutant l'état poubelle :

Tableau 5.14

| État | États résultant | | |
|---|---|---|---|
| | en lisant a | en lisant b | en lisant c |
| 1 | 1,2 | 2 | 3 |
| 1,2 | 1,2 | 2 | 3 |
| 2 | P | 2 | 3 |
| 3 | P | P | P |
| P | P | P | P |

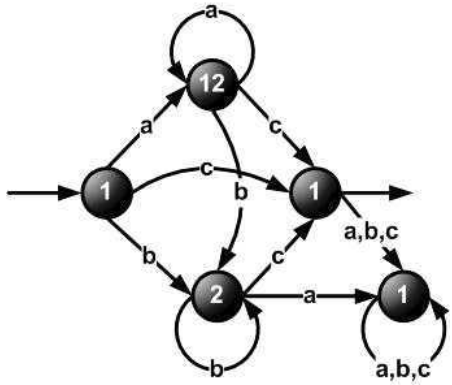


Figure 5.20 (A4)

d) Mais en réalité, il est plus simple de se passer de l'étape (b) et construire un automate déterministe directement à partir d'un automate asynchrone.
Redessinons l'automate (A1) :

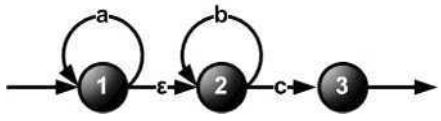


Figure 5.21 (A1)

L'état initial de l'automate déterministe correspondant à (A1) est (1,2), car en lisant le mot vide on arrive et en 1, et en 2. Continuons à déterminiser en marquant dans les colonnes correspondant à la lecture de chaque caractère, tous les états où on arrive après la lecture du caractère en question, c.à.d. non seulement l'état où mène la flèche étiquetée par ce caractère (disons, $a$), mais aussi tous les états où on peut arriver en lisant $a\varepsilon$, $a\varepsilon\varepsilon$ etc. Dans notre cas particulier, il n'y a pas de telles transitions, mais il faut systématiquement en tenir compte.

© Dunod – La photocopie non autorisée est un délit

Tableau 5.15

| État | États résultant | | |
|---|---|---|---|
| | en lisant a | en lisant b | en lisant c |
| 1,2 | 1,2 | 2 | 3 |
| 2 | – | 2 | 3 |
| 3 | – | – | – |

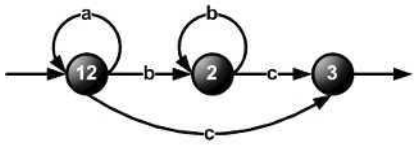


Figure 5.22 **(A5)**

On voit que l'automate (A5) n'est pas le même que l'automate (A3) ou (A4) ! Pourquoi ?

Parce que l'automate déterministe reconnaissant un certain langage n'est pas unique, ce n'est qu'en minimisant qu'on arrive à un automate unique. En minimisant (A4) et (A5), on doit obtenir la même chose.

Minimisons (A4) donc on rappelle la table de transitions :

Tableau 5.16

| État | États résultant | | |
|---|---|---|---|
| | en lisant a | en lisant b | en lisant c |
| 1 | 1,2 | 2 | 3 |
| 1,2 | 1,2 | 2 | 3 |
| 2 | P | 2 | 3 |
| 3 | P | P | P |
| P | P | P | P |

Ici , on voit immédiatement que les états 1 et 1,2 ne se sépareront jamais !

$\Theta_0 = \{(1,(1,2),2,P),\ (3)\} = \{I,(3)\}$ (Utilisons une notation un peu plus intelligente pour ne pas modifier le sens des chiffres romains sans cesse)

Tableau 5.17

| État | Groupes résultant | | |
|---|---|---|---|
| | en lisant a | en lisant b | en lisant c |
| 1 | I | I | 3 |
| 1,2 | I | I | 3 |
| 2 | I | I | 3 |
| P | I | I | I |

(5) se sépare en formant un groupe à part (c'est toujours le cas)

$\Theta_1 = \{(1,(1,2),2),\ (P),\ (3)\} = \{I,(P),(3)\}$

Tableau 5.18

| État | Groupes résultant | | |
|---|---|---|---|
| | en lisant a | en lisant b | en lisant c |
| 1 | I | I | 3 |
| 1,2 | I | I | 3 |
| 2 | P | I | 3 |

(2) se sépare.
$\Theta_2 = \{(1,(1,2)),\ (2),\ (P),\ (3)\} = \{I,(2),(P),(3)\}$

Tableau 5.19

| État | Groupes résultant | | |
|---|---|---|---|
| | en lisant a | en lisant b | en lisant c |
| 1 | I | 2 | 3 |
| 1,2 | I | 2 | 3 |

Le groupe (1,(1,2)) n'a aucun chance à se scinder en deux, car dès le début (1) et (1,2) ont les mêmes transitions. Donc on a terminé, et on obtient :

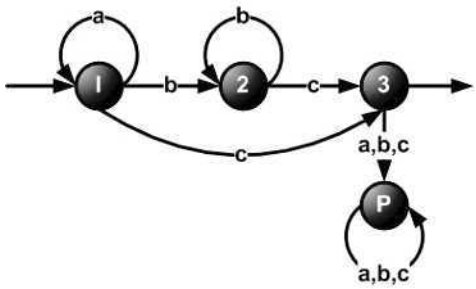


Figure 5.23

ce qui est le même automate que le résultat de compléter l'automate (A5) ! (en minimisant l'automate (A5), la seule chose qu'il reste à faire c'est de le compléter, autrement il est déjà minimal).

## ÉNONCÉS DES EXERCICES

**Exercice 5.1** Construire un automate fini dont les états correspondent aux situations de famille possibles d'une personne (célibataire, marié, divorcé, veuf) et dont les flèches correspondent aux changements de situation possible. Étiqueter ces flèches par M (mariage), D (divorce) et V (veuvage).

**Exercice 5.2** Construire des automates finis qui reconnaissent les langages suivants :
- (a) L'ensemble des mots sur l'alphabet $\{0, 1\}$ dont l'avant dernier symbole est 0.

© Dunod – La photocopie non autorisée est un délit

- (b) L'ensemble des mots sur l'alphabet $\{0, 1\}$ qui commencent et qui finissent par 0. (On considère que le mot consistant en un seul 0 vérifie cette condition).
- (c) L'ensemble des mots sur l'alphabet $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, +, -, .\}$ qui représentent en langage C des constantes numériques.
- (d) L'ensemble des mots sur l'alphabet $\{0, 1\}$ qui comportent au moins une fois le motif 10 et au moins une fois le motif 01 (ces deux motifs ne sont pas obligés d'être séparés l'un de l'autre – par exemple, la séquence 010 comporte les deux motifs).
- (e) $\{a^n b^m | n + m \text{ pair}\}$

**Exercice 5.3** Construire un automate fini reconnaissant les entiers écrits en base deux divisibles par cinq.

**Exercice 5.4** Construire un automate fini reconnaissant les lignes contenant des commentaires d'un programme écrit en langage C. La sortie de l'automate correspond à la fin du commentaire.
On prendra comme alphabet $\{/, *, ", a\}$ où $a$ représentera tous les caractères autres que ", * et /.
Tout commentaire commence par /* et finit par */.
Le commentaire peut contenir des / et des *, mais il ne peut pas contenir le motif */, sauf à l'intérieur d'une chaîne qui commence par " et finit par " sans contenir d'autres ". En fait, toute chaîne de caractères placée entre les guillemets doubles, est déspécialisée ; donc le nombre de guillemets doubles doit être pair et avant le commentaire, et à l'intérieur du commentaire.
La séquence /*/ n'est pas considérée comme comportant et le motif /*, et le motif */.
Exemple d'un commentaire reconnu par l'automate : /*commentaire "/*" toto"*/" **/

**Exercice 5.5** Quel est le langage reconnu par l'automate suivant ? Quels états de cet automate sont inutiles ?

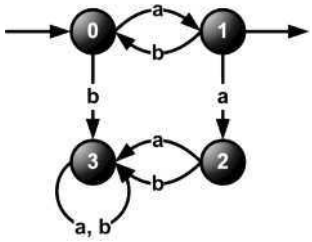


Figure 5.24

**Exercice 5.6** Voici cinq automates notés $A_n$ (ils n'ont rien à voir les uns avec les autres, ce n'est pas une séquence !). L'alphabet A = {a, b} sauf pour $A_4$ où il est {0,1}. Pour chacun de ces automates :
- Décrire $L(A_n)$ en langage ordinaire (sauf pour $A_5$).
- Caractériser $A_n$ (dire s'il est accessible, coaccessible, émondé...)
- Calculer l'automate déterministe équivalent à $A_n$.

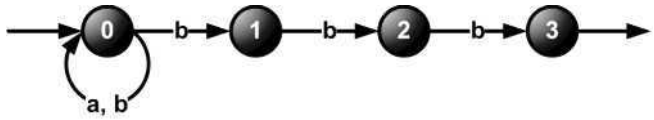


Figure 5.25 A1

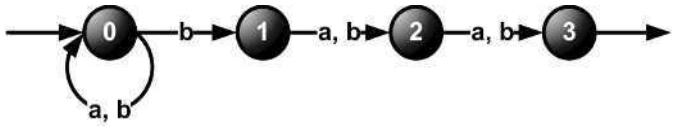


Figure 5.26 A2

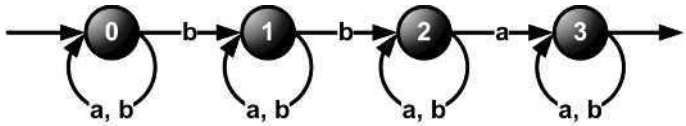


Figure 5.27 A3

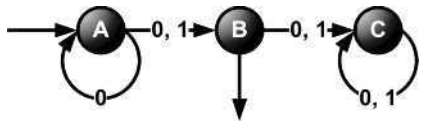


Figure 5.28 A4

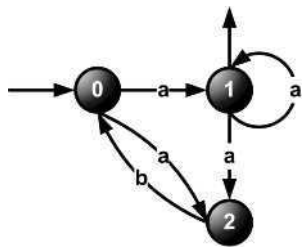


Figure 5.29 A5

(Ici, ne pas décrire $L(A_5)$ avant que l'automate ne soit déterminisé)

**Exercice 5.7** Déterminiser l'automate suivant :

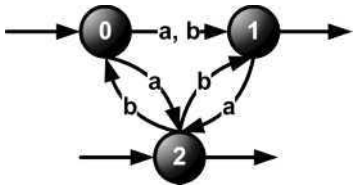


Figure 5.30

**Exercice 5.8** Construire des automates déterministes qui reconnaissent les langages suivants. En déduire des automates déterministes qui reconnaissent les complémentaires de ces langages :

- (a) L'ensemble des mots sur l'alphabet $\{0, 1\}$ qui contiennent **exactement** trois fois le symbole 1.
- (b) L'ensemble des mots sur l'alphabet $\{0, 1\}$ qui contiennent **au moins** un 1.

**Exercice 5.9** Construire un automate fini reconnaissant l'ensemble des mots sur l'alphabet $A = \{a,b\}$ qui **ne se terminent pas** par *abaab*. Le déterminiser et minimiser.

**Exercice 5.10** Montrer que les automates déterministes calculés à l'exercice 6 sont tous minimaux sauf un. Lequel ?

**Exercice 5.11** Déterminiser et minimiser les automates suivants :

- a)

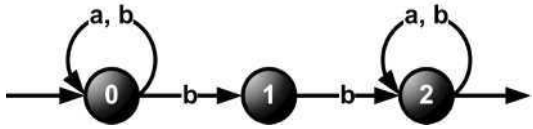


Figure 5.31

- b)

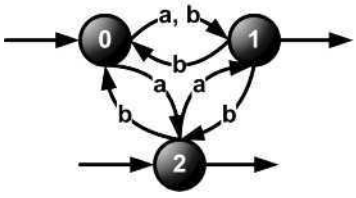


Figure 5.32

**Exercice 5.12** On définit la famille d'automates suivants :

$$\widetilde{A}_n = (Q_n, I, T, E),\ n \geqslant 1$$

avec :

- $A = \{a, b\}$
- $Q_n = \{0,1, ..., n-1\}$
- $I = T = \{0\}$
- Comme flèches étiquetées par $a$ l'ensemble de $(q.a.((q+1) \bmod n))$ pour $\forall q : 0 \leqslant q \leqslant n-1$
- Comme flèches étiquetées par b l'ensemble de $(q.b.0)$ et $(q.b.q)$ pour $\forall q : 1 \leqslant q \leqslant n-1$ (*attention : la première inégalité commence par 0, et la seconde, par 1*)

Dessiner $\widetilde{A_3}$ et $\widetilde{A_4}$.

Puis montrer que le déterminisé complet de $\widetilde{A_n}$ a toujours $2^n$ états.

## Corrigés des exercices

### Exercice 5.1

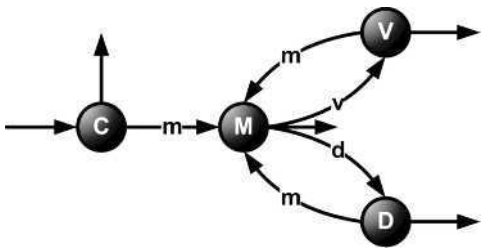


**Figure 5.33** **Cet automate est déterministe.**

### Exercice 5.2

Solution (a)

L'automate le plus simple qui reconnaît l'ensemble des mots sur l'alphabet {0, 1} dont l'avant dernier symbole est 0, est le suivant :

© Dunod – La photocopie non autorisée est un délit

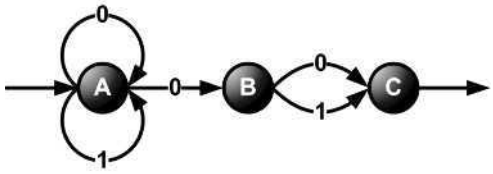


**Figure 5.34** **Cet automate n'est pas déterministe.**

### Solution (b)

L'automate le plus simple qui reconnaît l'ensemble des mots sur l'alphabet $\{0, 1\}$ qui commencent et qui finissent par 0, est le suivant :

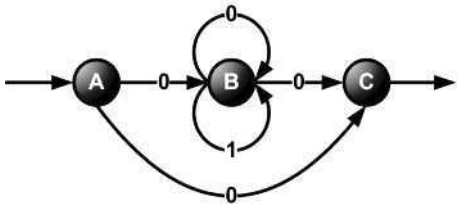


**Figure 5.35** **Cet automate n'est pas déterministe.**

### Solution (c)

Un des nombreux automates équivalents qui reconnaissent l'ensemble des mots sur l'alphabet $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, +, -, .\}$ représentant en C les constantes numériques, est le suivant :

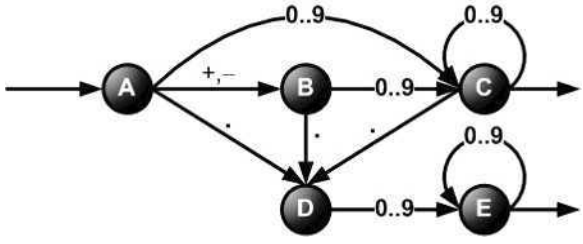


**Figure 5.36** **Cet automate est déterministe.**

### Solution (d)

Un des nombreux automates équivalents qui reconnaît l'ensemble des mots sur l'alphabet $\{0, 1\}$ comportant au moins une fois le motif 10 et au moins une fois le motif 01, est le suivant :

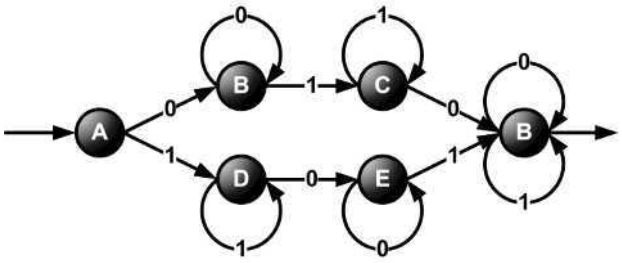


**Figure 5.37** **Cet automate est déterministe.**

Solution (e)

Un des nombreux automates équivalents qui reconnaît le langage $\{a^n b^m | n + m \text{ pair}\}$, est le suivant :

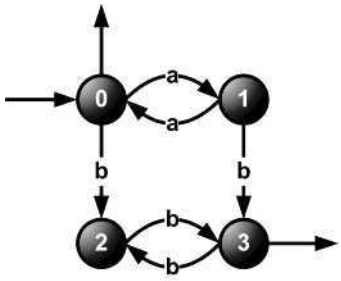


**Figure 5.38** **Cet automate est déterministe.**

**Exercice 5.3** Ajout d'un 0 à la fin d'un nombre binaire le multiplie par 2.
Ajout d'un 0 à la fin d'un nombre binaire le multiplie par 2 et lui ajoute 1.

**Tableau 5.20**

| N | N mod 5 | Ajout d'un 0 à la fin de l'écriture binaire donne N' : | N' mod 5 | Ajout d'un 1 à la fin de l'écriture binaire donne N" : | N" mod 5 |
|---|---|---|---|---|---|
| $5n$ | 0 | $10n$ | 0 | $10n + 1$ | 1 |
| $5n + 1$ | 1 | $10n + 2$ | 2 | $10n + 3$ | 3 |
| $5n + 2$ | 2 | $10n + 4$ | 4 | $10n + 5$ | 0 |
| $5n + 3$ | 3 | $10n + 6$ | 1 | $10n + 7$ | 2 |
| $5n + 4$ | 4 | $10n + 8$ | 3 | $10n + 9$ | 4 |

Cela se traduit par l'automate suivant :

© Dunod – La photocopie non autorisée est un délit

Tableau 5.21

| État | En lisant 0 | En lisant 1 |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 2 | 3 |
| 2 | 4 | 0 |
| 3 | 1 | 2 |
| 4 | 3 | 4 |

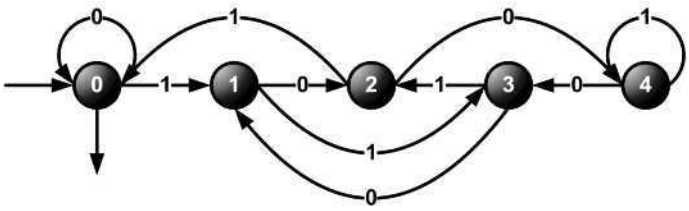


Figure 5.39

**Exercice 5.4**

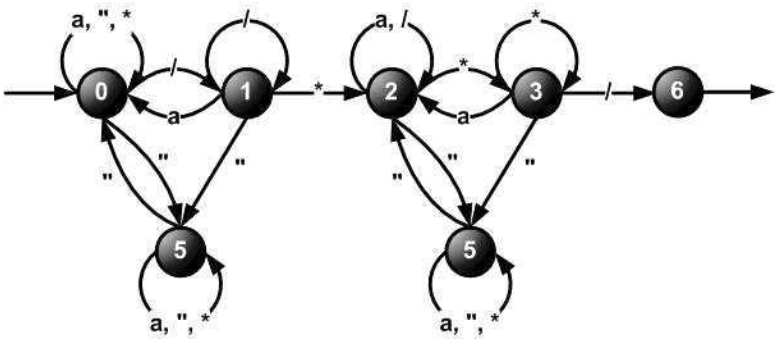


Figure 5.40

**Exercice 5.5** États 2 et 3 ne mènent pas à la sortie. Ils sont donc inutiles. Le langage reconnu par cet automate est le même que le langage reconnu par :

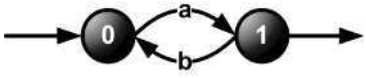


Figure 5.41

Il consiste en mots *a*, *aba*, *ababa*, *abababa*... qu'on peut écrire comme *a(ba)** ou *(ab)*a*.

## Exercice 5.6

### Solution $A_1$

- Cet automate lit tous les mots se terminant par *bbb*.
- Il est émondé.
- *Déterminisation*

Tableau 5.22

| État | a | b |
|---|---|---|
| 0 | 0 | 01 |
| 01 | 0 | 012 |
| 012 | 0 | 0123 |
| 0123 | 0 | 0123 |

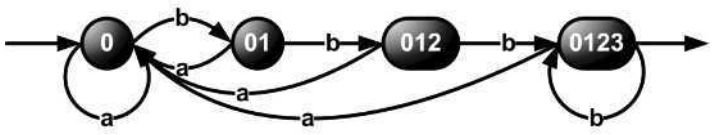


Figure 5.42 $A_1$

### Solution $A_2$

- Cet automate lit tous les mots avec un *b* en 3$^{ième}$ position de la fin.
- Il est émondé.
- *Déterminisation*

Tableau 5.23

| | État | A | b |
|---|---|---|---|
| Initial | 0 | 0 | 01 |
| | 01 | 02 | 012 |
| | 012 | 023 | 0123 |
| | 02 | 03 | 013 |
| Terminal | 0123 | 023 | 0123 |
| Terminal | 023 | 03 | 013 |
| Terminal | 03 | 0 | 01 |
| Terminal | 013 | 02 | 012 |

© Dunod – La photocopie non autorisée est un délit

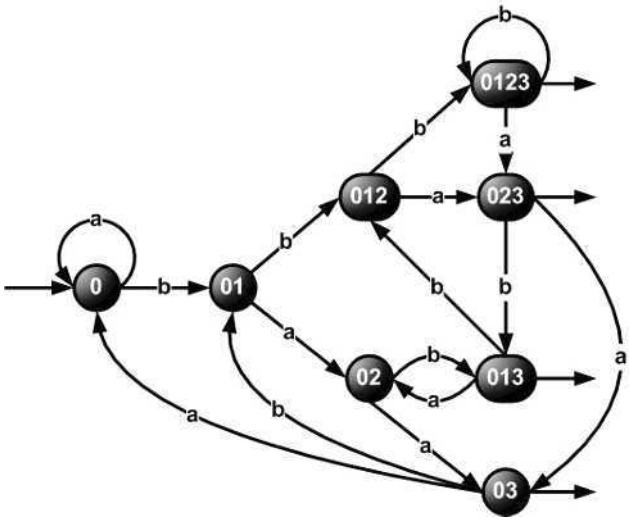


Figure 5.43

Solution $A_3$

- Cet automate lit tous les mots contenant au moins trois caractères *...a...b...a...* où « ... » peut être vide.
- Il est émondé.
- • *Déterminisation*

Tableau 5.24

| État | a | b |
|---|---|---|
| 0 | 01 | 01 |
| 01 | 01 | 012 |
| 012 | 0123 | 012 |
| 0123 | 0123 | 0123 |

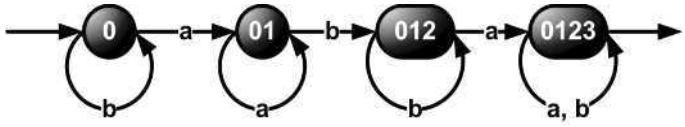


Figure 5.44

Solution $A_4$

- Cet automate lit tous les mots consistant en des 0 ou en des 0 suivis d'un seul 1, ou le mot 1.

- Il est accessible mais non pas coaccessible.
- *Déterminisation*

Tableau 5.25

| État | 0 | 1 |
|---|---|---|
| A | AB | B |
| AB | ABC | BC |
| ABC | ABC | BC |
| B | C | C |
| C | C | C |
| BC | C | C |

On voit que C est l'état poubelle.

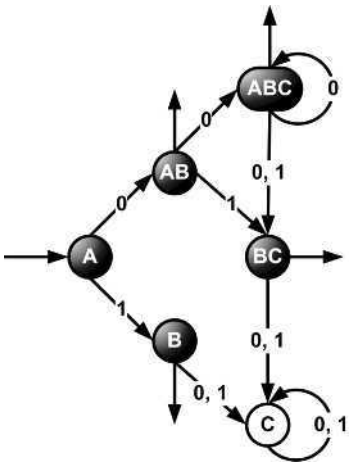


Figure 5.45

Ici on a obtenu donc un automate déterministe avec poubelle. Mais il y avait une « poubelle » dès le début – l'état C dont on ne pouvait pas revenir. On aurait pu construire d'abord un automate non déterministe plus simple qui reconnaît le même langage qu'$A_4$ :

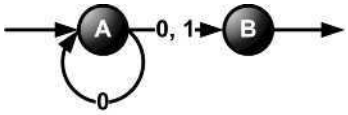


Figure 5.46

et le déterminiser. En ce faisant on arrive à un automate déterministe nettement plus simple que tout à l'heure :

© Dunod – La photocopie non autorisée est un délit

Tableau 5.26

| État | 0 | 1 |
|---|---|---|
| A | AB | B |
| AB | AB | B |
| B | P | P |
| P | P | P |

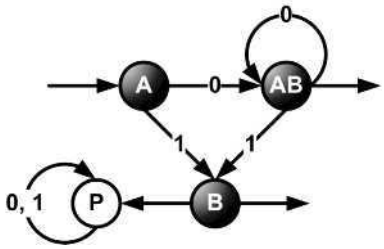


Figure 5.47

Dans l'exercice 10 on va voir que cet automate est en effet minimal.

Solution $A_5$

- On ne décrit pas le langage (c'est trop compliqué à ce stade).
- Il est émondé.
- *Déterminisation :*

Tableau 5.27

| État | a | b |
|---|---|---|
| 0 | 12 | P |
| 12 | 12 | 0 |

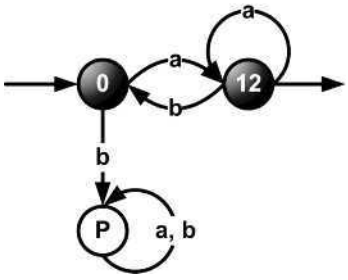


Figure 5.48

(Maintenant il devient bien plus facile de décrire $L(A_5)$ si l'on veut : il consiste en tous les mots commençant par *a*, finissant par *a*, et où tout *b* est entouré par des *a* de façon à ne jamais avoir *bb*).

## Exercice 5.7

Tableau 5.28

| État | a | b |
|---|---|---|
| 02 | 12 | 01 |
| 01 | 12 | 1 |
| 12 | 2 | 01 |
| 1 | 2 | P |
| 2 | P | 01 |
| P | P | P |

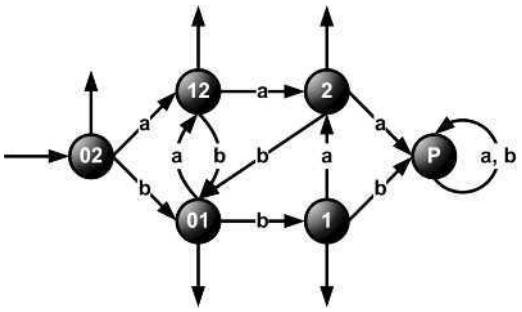


Figure 5.49

## Exercice 5.8

Solution (a)

Un automate déterministe complet qui reconnaît l'ensemble des mots sur l'alphabet $\{0, 1\}$ qui contiennent exactement trois fois le symbole 1 se construit directement, sans qu'on ait besoin de construire un automate non déterministe d'abord :

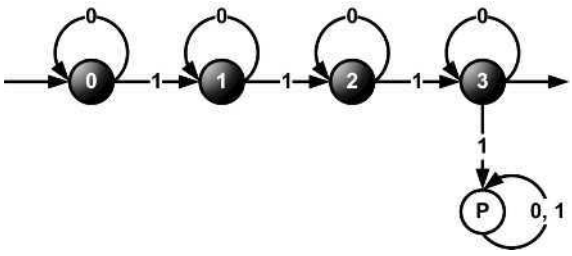


Figure 5.50

© Dunod – La photocopie non autorisée est un délit

D'où l'automate reconnaissant le complément de ce langage :

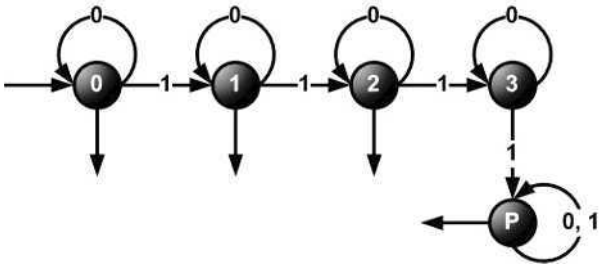


Figure 5.51

L'état P n'est plus un état poubelle, car il est un état terminal.

Solution (b)

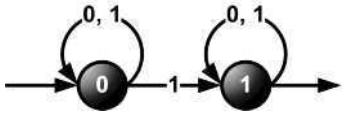


Figure 5.52

Cet automate n'est pas déterministe, il faut le déterminiser.

- *Déterminisation*

Tableau 5.29

| État | 0 | 1 |
|---|---|---|
| 0 | 0 | 01 |
| 01 | 01 | 01 |

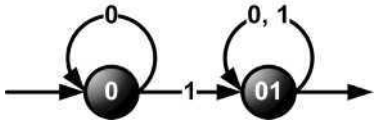


Figure 5.53

(En fait, on aurait pu dessiner cet automate dès le début).
Cet automate déterministe est complet ; donc pour construire l'automate reconnaissant le complément du langage, il suffit de modifier la position de la sortie :

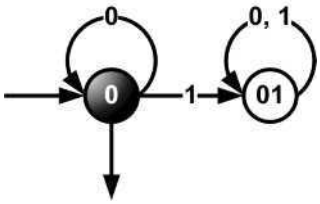


Figure 5.54 **Il est facile de voir qu'ici l'état 01 est la poubelle.**

**Exercice 5.9** Nous construisons d'abord un automate qui lit tous les mots qui **se terminent** par *abaab*. Puis nous le déterminiserons et compléterons, et puis nous ferons de tous les états finaux, états non finaux et vice versa. La minimisation se fera ensuite.

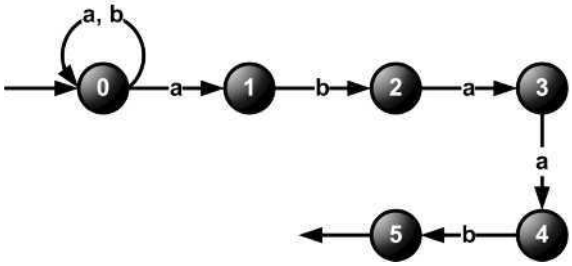


Figure 5.55

Cet automate lit tous les mots se terminant en .

- *Déterminisation*

Tableau 5.30

| État | a | b |
|---|---|---|
| 0 | 01 | 0 |
| 01 | 01 | 02 |
| 02 | 013 | 0 |
| 013 | 014 | 02 |
| 014 | 01 | 025 |
| 025 | 013 | 0 |

© Dunod – La photocopie non autorisée est un délit

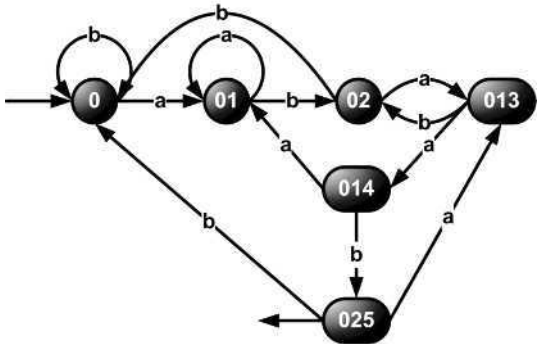


Figure 5.56

Cet automate est déjà complet, sinon il faudrait le compléter en ajoutant une poubelle.
L'automate déterministe lisant tous les mots **ne se terminant pas** par *abaab*, s'obtient comme suit :

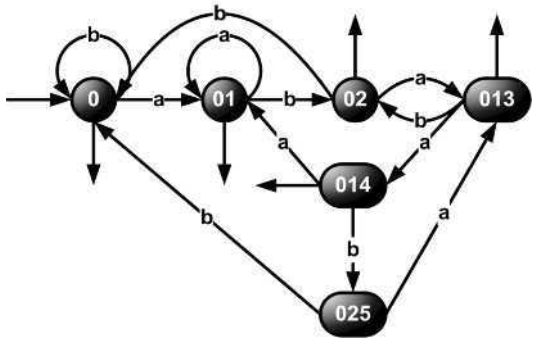


Figure 5.57

Il n'est pas minimisable, c.-à-d. qu'il est déjà minimal. Pour le voir, il faut essayer de le minimiser :
$\Theta_0 = \{I, II\}$ avec $I = \{025\}$, $II = \{0,01,02,013,014\}$

Tableau 5.31

| État | a | b | a | b |
|---|---|---|---|---|
| 0 | 01 | 0 | II | II |
| 01 | 01 | 02 | II | II |
| 02 | 013 | 0 | II | II |
| 013 | 014 | 02 | II | II |
| 014 | 01 | 025 | II | I |
| 025 | 013 | 0 | II | II |

Le groupe non terminal consiste en un seul état et n'a pas lieu à se séparer.
Le groupe terminal se sépare en deux :
$\Theta_1 = \{I, II, III\}$ avec $I = \{025\}$, $II = \{0{,}01, 02{,}013\}$, $III = \{014\}$

Tableau 5.32

| État | a | b | a | b |
|---|---|---|---|---|
| 0 | 01 | 0 | II | II |
| 01 | 01 | 02 | II | II |
| 02 | 013 | 0 | II | II |
| 013 | 014 | 02 | III | II |

Le groupe II se sépare en deux :
$\Theta_2 = \{I, II, III, IV\}$ avec $I = \{025\}$, II = {0, 01, 02}, III={014}, IV = {013}

Tableau 5.33

| État | a | b | a | b |
|---|---|---|---|---|
| 0 | 01 | 0 | II | II |
| 01 | 01 | 02 | II | II |
| 02 | 013 | 0 | IV | II |

Le groupe II se sépare en deux :
$\Theta_3 = \{I, II, III, IV, V\}$ avec $I = \{025\}$, II = {0, 01}, III = {014}, IV = {013}, V={02}

Tableau 5.34

| État | a | b | a | b |
|---|---|---|---|---|
| 0 | 01 | 0 | II | II |
| 01 | 01 | 02 | II | V |

$\Theta_4 = \Theta_3$ ; tout les états se sont séparés ; l'automate était minimal.

## Exercice 5.10

### Solution $A_1$

Reprenons l'automate déterministe complet $A_1$ :

© Dunod – La photocopie non autorisée est un délit

Tableau 5.35

| État | a | b |
|---|---|---|
| 0 | 0 | 01 |
| 01 | 0 | 012 |
| 012 | 0 | 0123 |
| 0123 | 0 | 0123 |

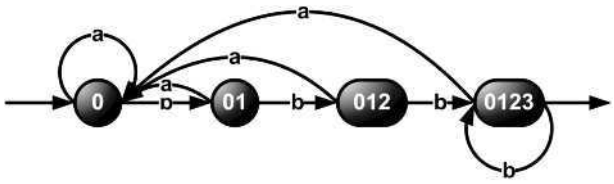


Figure 5.58

• *Minimisation*

$Q_0 = \{I,\ II\}$ avec $I = \{0,\ 01,\ 012\}$, $II = \{0123\}$.

Tableau 5.36

| État | a | B | A | b |
|---|---|---|---|---|
| 0 | 0 | 01 | I | I |
| 01 | 0 | 012 | I | I |
| 012 | 0 | 0123 | I | II |

Le groupe I se sépare en deux. Modifiant la notation, on a :
$Q_1 = \{I,\ II,\ III\}$ avec $I = \{0,\ 01\}$, $II = \{012\}$, $III = \{0123\}$.

Tableau 5.37

| État | a | B | a | b |
|---|---|---|---|---|
| 0 | 0 | 01 | I | I |
| 01 | 0 | 012 | I | II |

Les deux états du groupe I se sont séparés, donc tous les états se sont séparés – l'automate était minimal dès le début.

Solution $A_2$

Reprenons l'automate déterministe complet $A_2$ mais modifions la notation des états pour plus de clarté :

Tableau 5.38

| | | | | En renommant les états | | |
|---|---|---|---|---|---|---|
| | **État** | **a** | **b** | **État** | **a** | **b** |
| Initial | 0 | 0 | 01 | A | A | B |
| | 01 | 02 | 012 | B | D | C |
| | 012 | 023 | 0123 | C | F | E |
| | 02 | 03 | 013 | D | G | H |
| Terminal | 0123 | 023 | 0123 | E | F | E |
| Terminal | 023 | 03 | 013 | F | G | H |
| Terminal | 03 | 0 | 01 | G | A | B |
| Terminal | 013 | 02 | 012 | H | D | C |

(Le dessin n'est pas trop utile pour cet automate.)

- *Minimisation*

$\Theta_0 = \{I, II\}$ avec $I = \{A, B, C, D\}$, $II = \{E, F, G, H\}$.

Tableau 5.39

| | **État** | **a** | **b** | **a** | **b** |
|---|---|---|---|---|---|
| Groupe non terminal | A | A | B | I | I |
| | B | D | C | I | I |
| | C | F | E | II | II |
| | D | G | H | II | II |
| Groupe terminal | E | F | E | II | II |
| | F | G | H | II | II |
| | G | A | B | I | I |
| | H | D | C | I | I |

$\Theta_1 = \{I, II, III, IV\}$ avec $I = \{A, B\}$, $II=\{C, D\}$, $III=\{E, F\}$, $IV=\{G, H\}$.

© Dunod – La photocopie non autorisée est un délit

Tableau 5.40

| État | a | b | A | b |
|---|---|---|---|---|
| A | A | B | I | I |
| B | D | C | II | II |
| C | F | E | III | III |
| D | G | H | IV | IV |
| E | F | E | III | III |
| F | G | H | IV | IV |
| G | A | B | I | I |
| H | D | C | II | II |

Tous les états se sont séparés, l'automate était minimal dès le début.

### Solution $A_3$

Reprenons l'automate déterministe complet $A_3$ :

Tableau 5.41

| État | a | b |
|---|---|---|
| 0 | 01 | 01 |
| 01 | 01 | 012 |
| 012 | 0123 | 012 |
| 0123 | 0123 | 0123 |

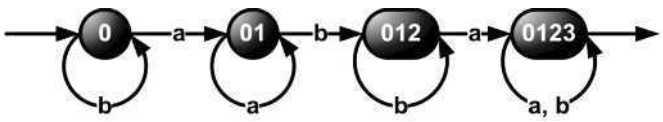


Figure 5.59

- *Minimisation*

$\Theta_0 = \{I, II\}$ avec $I = \{0, 01, 012\}$, $II = \{0123\}$

Tableau 5.42

| État | a | b | a | b |
|---|---|---|---|---|
| 0 | 01 | 01 | I | I |
| 01 | 01 | 012 | I | I |
| 012 | 0123 | 012 | II | I |

L'état 012 est sorti du groupe I. Nouveaux groupes :
$\Theta_1 = \{I, II, III\}$ avec $I = \{0, 01\}$, $II = \{012\}$, $III = \{0123\}$.

Tableau 5.43

| État | a | b | a | B |
|---|---|---|---|---|
| 0 | 01 | 01 | I | I |
| 01 | 01 | 012 | I | II |

Tous les états se sont séparés, l'automate était minimal dès le début.

### Solution $A_4$

Reprenons l'automate déterministe complet $A_4$ :

Tableau 5.44

| État | 0 | 1 |
|---|---|---|
| A | AB | B |
| AB | ABC | BC |
| ABC | ABC | BC |
| B | C | C |
| C | C | C |
| BC | C | C |

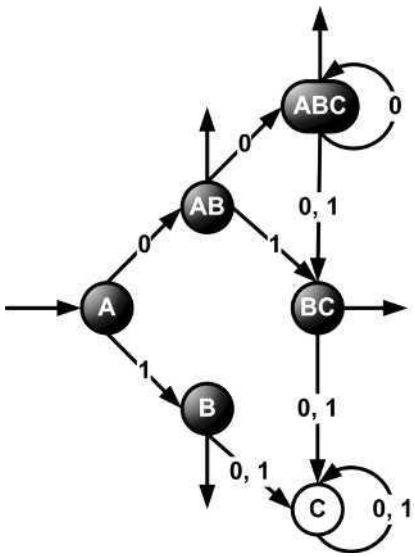


Figure 5.60

- *Minimisation*

$\Theta_0 = \{I, II\}$ avec $I = \{A, C\}$, $II = \{AB, ABC, BC, B\}$.

© Dunod – La photocopie non autorisée est un délit

Tableau 5.45

| | État | 0 | 1 | 0 | 1 |
|---|---|---|---|---|---|
| Groupe non terminal | A | AB | B | II | II |
| | C | C | C | I | I |
| Groupe terminal | AB | ABC | BC | II | II |
| | ABC | ABC | BC | II | II |
| | BC | C | C | I | I |
| | B | C | C | I | I |

On peut voir immédiatement que les états A et C se séparent, et que le groupe II se divise en deux groupes, qui ne pourront plus se séparer. On obtient donc les états de l'automate minimisé : A, (B, BC), (AB, ABC), et C.

Graphiquement, cet automate minimal se présente comme :

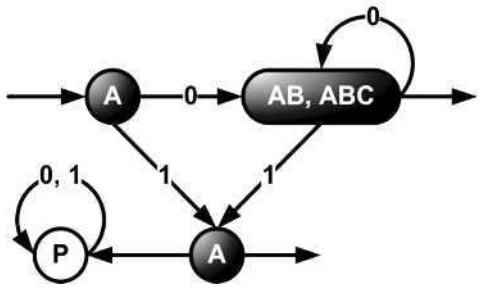


Figure 5.61

où l'état C sert comme l'état poubelle. L'automate déterministe correspondant à $A_4$ n'était donc pas minimal. On voit qu'en le minimisant on est arrivé au même automate qu'on a obtenu en déterminisant l'automate $A_4$ initial (non déterministe) duquel on a coupé sa « poubelle ».

### *Remarque*

On rappelle encore une fois que la « poubelle » d'un automate non déterministe (l'état ou le groupe d'états dont on ne peut pas allez vers la ou les sortie(s) n'est pas la même chose que la poubelle de l'automate déterministe qui en résulte après la déterminisation.
La seule utilité de la « poubelle » d'un automate non déterministe consiste en la possibilité de la couper pour simplifier la déterminisation.

Solution $A_5$

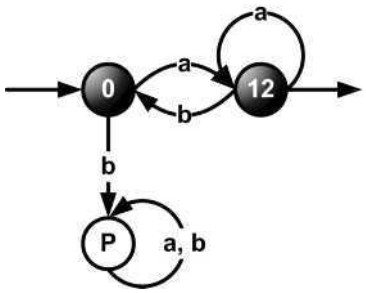


Figure 5.62 $A_5$

• *Minimisation*

$\Theta_0 = \{I, II\}$ avec $I = \{0, P\}$, $II = \{12\}$.

Tableau 5.46

| État | a | b | a | b |
|---|---|---|---|---|
| 0 | 12 | P | II | I |
| P | P | P | I | I |
| 12 | 12 | 0 | II | I |

Les deux états du groupe I se sont séparés ; donc tous les états se sont séparés, l'automate était minimal dès le début.

## Exercice 5.11

Solution (a)

• *Déterminisation*

Tableau 5.47

| État | a | b |
|---|---|---|
| 0 | 0 | 01 |
| 01 | 0 | 012 |
| 012 | 02 | 012 |
| 02 | 02 | 012 |

© Dunod – La photocopie non autorisée est un délit

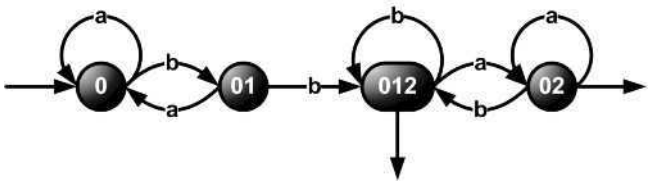


Figure 5.63

• *Minimisation*

$\Theta_0 = \{I, II\}$ avec $I = \{0, 01\}$, $II = \{012, 02\}$.

Tableau 5.48

| État | a | b | a | b |
|---|---|---|---|---|
| 0 | 0 | 01 | I | I |
| 01 | 0 | 012 | I | II |
| 012 | 02 | 012 | II | II |
| 02 | 02 | 012 | II | II |

Le groupe II ne peut pas se séparer. Le groupe I se sépare en 0 et 01.
Le résultat :

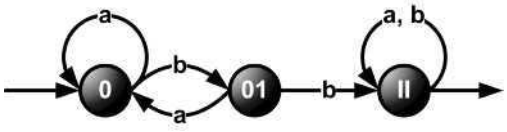


Figure 5.64

Solution (b)

• *Déterminisation*

Tableau 5.49

| | | | En modifiant les étiquettes des états | | |
|---|---|---|---|---|---|
| **État** | **a** | **b** | **État** | **a** | **b** |
| 02 | 12 | 01 | A | B | D |
| 01 | 12 | 012 | D | B | E |
| 12 | 1 | 02 | B | C | A |
| 012 | 12 | 012 | E | B | E |
| 1 | P | 02 | C | P | A |
| P | P | P | P | P | P |

Tous les états sont des états finaux sauf l'état poubelle.

Tableau 5.50

| État | a | b | a | b |
|---|---|---|---|---|
| A | B | D | II | II |
| D | B | E | II | II |
| B | C | A | II | II |
| E | B | E | II | II |
| C | P | A | I | II |

• *Minimisation*

$\Theta_0 = \{I, II\}$ avec $I = \{P\}$, $II = \{A, B, C, D, E\}$.

Nouveaux groupes d'états : $I = \{P\}$, $II = \{C\}$, $III = \{A, B, D, E\}$.

Tableau 5.51

| État | a | b | a | b |
|---|---|---|---|---|
| A | B | D | III | III |
| D | B | E | III | III |
| B | C | A | II | III |
| E | B | E | III | III |

Nouveaux groupes d'états : $I = \{P\}$, $II = \{C\}$, $III = \{B\}$, $IV = \{A, D, E\}$.

Tableau 5.52

| État | a | b | a | b |
|---|---|---|---|---|
| A | B | D | III | IV |
| D | B | E | III | IV |
| E | B | E | III | IV |

Le groupe IV ne se sépare pas.

Le résultat :

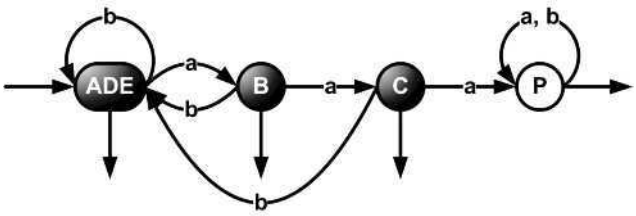


Figure 5.65

© Dunod – La photocopie non autorisée est un délit

## Exercice 5.12

• $\tilde{A}_3$

états 0, 1, 2;
flèches 0.a.(1 mod 3) = 0.a.1 ; 1.a.(2 mod 3) = 1.a.2 ; 2.a.(3 mod 3) = 2.a.0 ;
1.b.0; 1.b.1; 2.b.0; 2.b.2.

Tableau 5.53

| État | a | b |
|---|---|---|
| 0 | 1 | – |
| 1 | 2 | 0,1 |
| 2 | 0 | 0,2 |

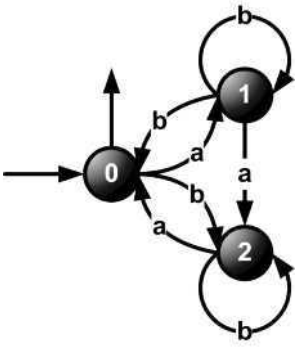


Figure 5.66

• $\tilde{A}_4$

états 0, 1, 2, 3;
flèches 0.a.(1 mod 4) = 0.a.1 ; 1.a.(2 mod 4) = 1.a.2 ;
2.a.(3 mod 4) = 2.a.3 ; 2.a.(4 mod 4) = 2.a.0 ;
1.b.0; 1.b.1; 2.b.0; 2.b.2; 3.b.0; 3.b.3.

Tableau 5.54

| État | a | B |
|---|---|---|
| 0 | 1 | – |
| 1 | 2 | 0,1 |
| 2 | 3 | 0,2 |
| 3 | 0 | 0,3 |

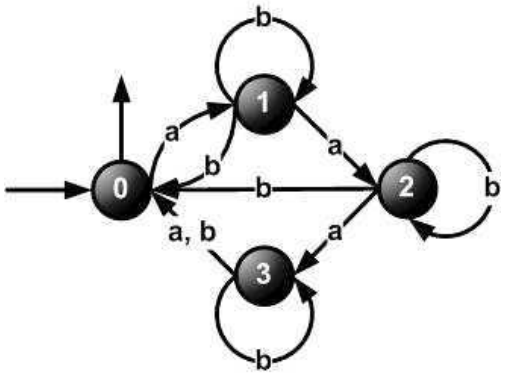


Figure 5.67

- $\tilde{A}_n$

Tableau 5.55

| État | a | B |
|---|---|---|
| 0 | 1 | – |
| 1 | 2 | 0,1 |
| 2 | 3 | 0,2 |
| 3 | 4 | 0,3 |
| ... | ... | ... |
| $n-1$ | 0 | 0, $n-1$ |

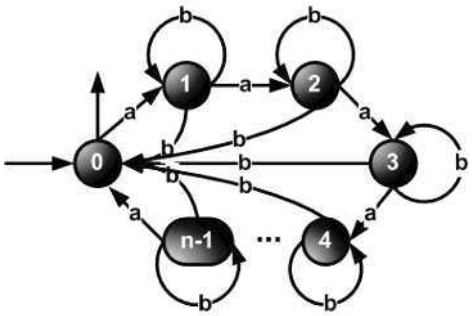


Figure 5.68

© Dunod – La photocopie non autorisée est un délit

- *Déterminisation*

Tableau 5.56

| État | a | B |
|---|---|---|
| 0 | 1 | – |
| 1 | 2 | 01 |
| 2 | 3 | 02 |
| 3 | 4 | 03 |
| ... | ... | ... |
| $n-1$ | 0 | $(0, n-1)$ |
| 01 | 12 | 01 |
| 02 | 13 | 02 |
| ... | ... | ... |
| $(0, n-1)$ | 01 | $(0, n-1)$ |
| 12 | 23 | 012 |
| 13 | 24 | 013 |
| ... | ... | ... |
| $(1, n-1)$ | 02 | $(0,1, n-1)$ |
| ... | ... | ... |

Ici figurent toutes les combinaisons possibles de $m$ états parmi $n$ c'est-à-dire : $1 \leqslant m \leqslant n$.

Il y a $\sum_{m=0}^{n} C_n^m = 2^n - 1$ de tels états. Si l'on y ajoute l'état poubelle (où mène la flèche étiquetée $b$ partant de l'état 0), pour que l'automate soit complet, on obtient $2^n$ états.

# Bibliographie


Baynat B., Chrétienne P., Hanen C., Kedad-Sidhoum S., Munier-Kordon A. et Picouleau C. – *Exercices et problèmes d'algorithmique* 2e édition. Dunod, Paris, 2007.

Carrez C. – *Des structures aux bases de données*. Dunod, Paris, 1994.

Cormen T., Leiserson C., Rivest R. et Stein C. – *Introduction à l'algorithmique* 3e édition. Dunod, Paris, 2010.

Froidevaux C., Gaudel M.-C. et Soria M. – *Types de données et algorithmes*. Ediscience International, Paris, 1994.

Guyot J. et Vial C. – *Arbres, tables et algorithmes*. Eyrolles, Paris, 1992.

Luc Bouge L., Kenyon C., Muller J.-M. et Robert Y. – *Algorithmique. Exercices corrigés*. Ellipses, Paris, 1993.

# INDEX


**A**

accepteurs 169
adresse 18
affectation 7
afficher 8, 45
algorithme 1
allocation dynamique 23
alphabet 170, 171, 174, 176, 177, 182
arbre 127–129, 131, 137–139, 143
  binaire 127–129, 131, 142, 145
  de Syracuse 145
automate
  fini 169, 170, 172, 173, 176, 178

**B**

boucle 12

**C**

calcul 1
champ 29
complexité algorithmique 1
concaténation 37, 45, 51, 52
condition 9
constante 7

**D**

décidabilité 1
  algorithmique 1
  logique 1
  pratique 1
  théorique 1
déterminisation 173–177, 195–198, 200, 201, 208–210, 214

**E**

émondé 176, 188, 195, 196, 198
équifinalité 1
ET 11
état 169–172, 174, 178, 182, 204
  accepteur 171
  initial 170, 174
  terminal 170, 174, 177, 178, 182
évaluation 13
expression 8

**F**

FIFO 90
file 90
fonction 23
  appel 25
  appelante 25
  appelée 25
  argument 26
  corps 24
  en-tête 24

**H**

Heqat 99

**I**

insertion 35, 37, 43, 99, 112
  point d' 112
itératif 12, 45, 49, 50, 54, 63, 142, 178


© Dunod – La photocopie non autorisée est un délit

**L**

LIFO 87
liste
    doublement chaînée 44, 99, 112
    linéaire 35
    simplement chaînée 38, 46, 112

**M**

machine de Turing 169
minimisation 178, 182, 201
mot
    vide 170

**N**

NON 11
NULL 21

**O**

OU 11

**P**

pile 87–89
`pile` 89
pointeur 18, 19
prédécesseur 60, 61, 112, 145, 166
problème 1
problème de Joseph 99
pseudo code IX

**R**

racine 127
raisonnement 1
récursif 45, 50–52, 61, 65, 136, 142
récursivité 127, 136

**S**

saisir 9, 40
solution 1
sommet de pile 87
sous-programme 23
structures de contrôle 9
successeur 56, 112
suppression 37, 41, 43, 46, 61, 99, 112, 138

**T**

tableaux 14
tables de vérité 11
transition 33, 169–171, 174, 175, 214
    table de 171
tri 2
type 5
    pointé 20

**V**

variable 6
    locale 24