

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

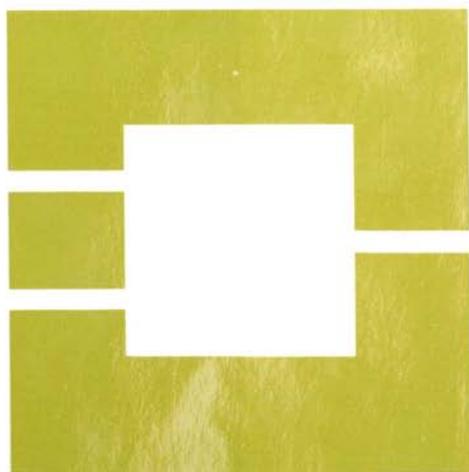
Traité d'Électricité

PUBLIÉ SOUS LA DIRECTION DE JACQUES NEIRYNCK

VOLUME XI

MACHINES SÉQUENTIELLES

Jacques Zahnd



PRESSES POLYTECHNIQUES ROMANDES

TRAITÉ D'ÉLECTRICITÉ

XI
MACHINES
SÉQUENTIELLES

TRAITÉ D'ÉLECTRICITÉ

DE L'ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE
PUBLIÉ SOUS LA DIRECTION DE JACQUES NEIRYNCK

VOLUME XI

MACHINES SÉQUENTIELLES

par Jacques Zahnd



PRESSES POLYTECHNIQUES ROMANDES

Cet ouvrage fait partie d'une série de vingt-deux volumes
dont les titres sont les suivants :

- I INTRODUCTION À L'ÉLECTROTECHNIQUE
- II MATÉRIAUX DE L'ÉLECTROTECHNIQUE
- III ÉLECTROMAGNÉTISME
- IV THÉORIE DES RÉSEAUX DE KIRCHHOFF
- V ANALYSE ET SYNTHÈSE DES SYSTÈMES LOGIQUES
- VI THÉORIE ET TRAITEMENT DES SIGNAUX
- VII DISPOSITIFS À SEMICONDUCTEUR
- VIII ÉLECTRONIQUE
- IX TRANSDUCTEURS ÉLECTROMÉCANIQUES
- X MACHINES ÉLECTRIQUES
- XI MACHINES SÉQUENTIELLES
- XII ÉNERGIE ÉLECTRIQUE
- XIII HYPERFRÉQUENCES
- XIV CALCULATRICES
- XV ÉLECTRONIQUE DE PUISSANCE
- XVI ÉLECTRONIQUE DE RÉGLAGE ET DE COMMANDE
- XVII SYSTÈMES DE MESURE
- XVIII SYSTÈMES DE TÉLÉCOMMUNICATIONS
- XIX FILTRES ÉLECTRIQUES
- XX TRAITEMENT NUMÉRIQUE DES SIGNAUX
- XXI ÉLECTROACOUSTIQUE
- XXII HAUTE TENSION



Le Traité d'Electricité est une publication des
Presses polytechniques romandes, fondation scientifique
dont le but est principalement la diffusion des travaux de
l'Ecole polytechnique fédérale de Lausanne.
Le catalogue de ces publications peut être obtenu aux
Presses polytechniques romandes, CH-1015 Lausanne.

Troisième édition
ISBN (série) : 2-604-00002-4
ISBN (ce volume) : 2-88074-051-7
© 1987, Presses polytechniques romandes
CH-1015 Lausanne
Imprimé en France

INTRODUCTION

Place de ce volume dans le traité d'Électricité, d'Électronique et d'Électrotechnique

La hiérarchie des modèles utilisée par l'ingénieur électricien, décrite dans l'introduction du Traité (*Introduction à l'électrotechnique*), comporte un niveau 3 intitulé schémas fonctionnels. A ce niveau, un système est constitué par la connexion de blocs remplissant des fonctions caractérisées par les relations entre grandeurs d'entrée et de sortie. Le volume « *Analyse et synthèse des systèmes logiques* » traite en particulier des schémas logiques, dont toutes les variables sont binaires.

Une généralisation de la classe des systèmes logiques est constituée par celle des systèmes discrets, dont les variables d'entrée et de sortie ont un nombre de valeurs fini quelconque. Ce sont les systèmes considérés dans le présent volume. On y suppose en outre que le temps n'est pas une variable continue, mais une suite discrète d'instant d'échantillonnage. Un signal d'entrée ou de sortie est alors une suite de valeurs représentée par une suite de symboles, qu'on appelle une séquence. La fonction la plus générale d'un tel système est d'associer à toute séquence d'entrée une ou plusieurs séquences de sortie. On aboutit ainsi à un modèle abstrait, extrêmement dépouillé, dans lequel un système n'est plus considéré que comme une transformation au sens mathématique de séquences d'entrée en séquences de sortie. C'est à cette transformation qu'on donne dans ce volume le nom de machine.

La notion de machine ainsi esquissée est très générale, et englobe quantité de cas particuliers qui n'entrent pas dans cet ouvrage, parce que leurs applications se situent dans des domaines qui ne concernent pas spécifiquement l'ingénieur électricien, par exemple la compilation des langages de programmation. On s'est restreint aux types de transformations de séquences qui sont réalisables, moyennant un codage binaire approprié, par les systèmes logiques étudiés dans le volume « *Analyse et synthèse des systèmes logiques* ». Il est clair que les systèmes logiques sont conçus en vue de réaliser des transformations de séquences. C'est pourquoi l'étude de ces transformations elles-mêmes, et en particulier des différentes manières de les spécifier ou représenter, de simplifier cette représentation, et d'en déduire des systèmes logiques capables de les réaliser, fournit de précieux outils pour la synthèse de ces derniers. C'est dans ce sens que le présent volume est destiné à compléter le volume « *Analyse et synthèse des systèmes logiques* »

Organisation générale de ce volume

La matière de ce volume correspond en bonne partie à ce qui se trouve, principalement dans la littérature de langue anglaise, sous le titre de théorie des automates finis. On s'est cependant écarté assez considérablement des exposés classiques dans le choix et la définition des concepts de base, en vue de la plus grande généralité, et de la plus grande unité possible entre les différentes parties de l'ouvrage. Tout l'exposé est construit logiquement sur un petit nombre de concepts de base, au moyen de définitions et

démonstrations mathématiques. Un accent est ainsi délibérément placé sur la rigueur. L'aridité de cette démarche devrait être atténuée toutefois par la quantité des exemples simples destinés à faciliter la compréhension.

Le premier chapitre contient un rappel des notions élémentaires de la théorie des ensembles, qui constitue le langage mathématique de l'ensemble de l'ouvrage. Le lecteur familiarisé avec ce langage pourra se contenter de lire les sections 1.3 à 1.5 (correspondances, séquences, graphes) qui sont plus spécifiques à cet ouvrage, et utilisées constamment par la suite. Comme le remarquent Aleksander et Hanna [27], le langage mathématique de la théorie des automates est simple. Cependant la plupart des ingénieurs, alors qu'ils sont à l'aise parmi les subtilités du calcul différentiel, perdent pied dans le plus simple des raisonnements ensemblistes. Cela ne tient qu'à leur formation traditionnelle. Aussi une lecture attentive du chapitre en question devrait les familiariser avec le style mathématique de l'ouvrage.

On élabore au chapitre 2 la notion générale de machine discrète esquissée ci-dessus. Considérés comme des boîtes noires munies d'entrées et de sorties, ces transformateurs de séquences peuvent être assemblés selon divers schémas de composition, et l'on définit la machine résultante pour les différentes formes d'assemblage de base. On introduit ensuite les types de machines fondamentaux considérés dans la suite. Ce sont les machines combinatoires, qui généralisent la notion de système logique combinatoire, puis les machines séquentielles, terme auquel il est donné un sens plus particulier que dans la littérature en général. On introduit encore les deux assemblages classiques d'une machine séquentielle et d'une machine combinatoire connus sous le nom de machine de Moore et machine de Mealy. Tous ces types de machines sont définis sous leur forme générale, non déterministe.

Les deux chapitres suivants traitent de différentes façons de spécifier la transformation de séquences que doit effectuer une machine. Le chapitre 3 est consacré spécifiquement au cas des machines binaires (ou logiques). On y étudie divers modes de représentation qui viennent s'ajouter à ceux du volume V. Ce sont les systèmes d'équations de récurrence booléennes, dont une classe particulière peut être avantageusement notée sous la forme de graphes, appelés graphes de récurrence booléens. Parmi ceux-ci, les graphes dits réceptifs correspondent à certains modes de représentation proposés récemment pour formaliser le cahier des charges des automatismes logiques [15, 16]. Le chapitre 4 traite de l'emploi des expressions régulières pour la spécification d'une transformation de séquences générales (binaires ou non). On y donne les bases de la théorie des langages réguliers et de leur relation avec les automates finis, puis on applique cette théorie à la spécification de machines, d'une manière qui inclut le cas de spécification incomplètes, généralement négligé dans la littérature. Les deux chapitres sont illustrés par des exemples de formalisation de cahiers des charges.

Le chapitre 5 traite de façon approfondie l'opération de réduction d'une table d'états, abordée au volume V. Le problème est posé en termes généraux, à savoir la réduction d'une machine de Mealy non déterministe. Certes des méthodes pratiques ne sont données que pour les cas particuliers plus usuels, pour lesquels seulement de bons algorithmes sont connus actuellement. Mais la théorie exposée cherche à mettre en évidence ce qui distingue ces cas particuliers du point de vue de la réduction, de façon à fournir une base à l'étude de méthodes plus générales.

Enfin le chapitre 6 est consacré à certaines formes de décomposition des machines séquentielles, en relation avec le problème du codage des états d'une table d'états.

Le problème est traité sous sa forme la plus générale : machines non déterministes, décompositions liées à des recouvrements. Contrairement à la réduction (chap. 5), le principe de la décomposition est le même pour les machines déterministes et non déterministes, à l'exception de la décomposition parallèle. L'accent dans ce chapitre est mis sur ce principe plutôt que sur les méthodes qui peuvent en découler. Toutefois, une section est consacrée à la méthode classique du calcul des partitions substitutives pour les machines déterministes.

Conventions

Le *Traité d'Electricité* est composé de volumes (vol.) repérés par un chiffre romain (vol. V). Chaque volume est partagé en chapitre (chap.) repérés par un nombre arabe (chap. 2). Chaque chapitre est divisé en sections (sect.) repérées par deux nombres arabes séparés par un point (sect. 2.3). Chaque section est divisée en paragraphes (§) repérés par trois nombres arabes séparés par deux points (§ 2.3.11). Les références internes stipulent le volume, le chapitre, la section ou le paragraphe du *Traité* auquel on renvoie. Dans le cas de la référence à une partie du même volume, on omet le numéro de celui-ci.

Les références bibliographiques sont numérotées continûment par volume et repérées par un seul nombre arabe entre crochets; les pages concernées sont éventuellement précisées entre parenthèses: [33] (pp. 12-15).

Un terme apparaît en *italique maigre* la première fois qu'il est défini dans le texte. Un passage important est mis en évidence lorsqu'il est composé en *italique gras*.

Les équations hors texte sont numérotées continûment par chapitre et repérées par deux nombres arabes placés entre parenthèses et séparés par un point (3.14); une équation est mise en évidence par son numéro imprimé en caractère gras. Les figures et tableaux sont numérotés continûment par chapitre et repérés par deux nombres arabes précédés de Fig. (Fig. 4.12) ou Tableau (Tableau 4.13).

TABLE DES MATIÈRES

	INTRODUCTION	v
CHAPITRE 1	PRÉLIMINAIRES	
	1.1 Ensembles et fonctions.	1
	1.2 Produits cartésiens.	6
	1.3 Correspondances.	8
	1.4 Séquences	11
	1.5 Graphes.	15
CHAPITRE 2	MACHINES	
	2.1 Notions générales	19
	2.2 Machines composées	26
	2.3 Machines combinatoires	33
	2.4 Machines séquentielles	37
	2.5 Machines séquentielles complètement spécifiées	46
	2.6 Machines de Moore et de Mealy	51
CHAPITRE 3	SPÉCIFICATION DES MACHINES BINAIRES	
	3.1 Expressions booléennes	61
	3.2 Equations de récurrence booléennes	74
	3.3 Graphes de récurrence booléens	92
	3.4 Graphes de récurrence réceptifs	110
CHAPITRE 4	EXPRESSIONS RÉGULIÈRES	
	4.1 Opérations régulières	123
	4.2 Diagrammes réguliers	135
	4.3 Automates finis.	145
	4.4 Application à la synthèse des machines	156
CHAPITRE 5	RÉDUCTION DES MACHINES DE MEALY	
	5.1 Simulation et réduction	175
	5.2 Machines quotients	178
	5.3 Classes de compatibilité	189
	5.4 Construction des recouvrements.	201
	5.5 Cas particuliers	214

CHAPITRE 6	DÉCOMPOSITION ET ASSIGNEMENT DES MACHINES SÉQUENTIELLES	
	6.1 Assignements	221
	6.2 Décomposition série.	235
	6.3 Décomposition parallèle	248
	6.4 Partitions.	251
	BIBLIOGRAPHIE	261
	INDEX ANALYTIQUE	263

PRÉLIMINAIRES

1.1 ENSEMBLES ET FONCTIONS

1.1.1 Introduction

Les notions de base de la théorie des ensembles sont supposées familières au lecteur, [1] (chap. 1 à 10). Le but de cette section n'est que d'en rappeler la terminologie et les notations essentielles.

Un *ensemble* E est une collection d'objets appelés les *éléments* de E . Il s'agit là d'une notion intuitive, le terme de collection n'étant pas mieux défini que celui d'ensemble. La relation $x \in E$, appelée *relation d'appartenance*, signifie que l'objet x est élément de E . Elle se lit "x appartient à E". Sa négation s'écrit $x \notin E$.

Le terme d'objet désigne n'importe quel objet concret ou abstrait, tel que nombre, point, fonction, etc. Un ensemble est lui-même un objet mathématique, de sorte que les éléments d'un ensemble E peuvent être eux-mêmes des ensembles. Dans ce cas, E est un ensemble d'ensembles.

1.1.2 Sous-ensembles

On dit qu'un ensemble E est un *sous-ensemble* d'un ensemble F , et l'on écrit $E \subset F$, si chaque élément de E est élément de F , c'est-à-dire si la relation $x \in E$ implique $x \in F$. Cette implication s'abrège usuellement $x \in E \Rightarrow x \in F$. On dit aussi que E est *inclus* dans F , ou encore que E est une *partie* de F . La relation $E \subset F$ est appelée *relation d'inclusion*. Il découle immédiatement de cette définition que l'on a $E \subset E$ pour tout ensemble E .

La négation de la relation $E \subset F$ s'écrit $E \not\subset F$. Elle signifie qu'il existe un objet x tel que $x \in E$ et $x \notin F$. Rappelons que l'expression "il existe un ..." signifie toujours en mathématique "il existe au moins un ...".

1.1.3 Egalité d'ensembles

La notion d'ensemble est précisée par la définition de l'*égalité* de deux ensembles, $E = F$. Cette égalité signifie par définition que $E \subset F$ et $F \subset E$. Lorsqu'on a défini deux ensembles E, F par des voies différentes, et qu'on veut démontrer leur égalité, on doit donc démontrer les deux implications $x \in E \Rightarrow x \in F$ et $x \in F \Rightarrow x \in E$.

Rappelons que si R et S sont deux relations quelconques, par exemple les relations $x \in E, x \in F$, la double implication $R \Rightarrow S$ et $S \Rightarrow R$ s'écrit $R \iff S$ et se lit "R équivaut à S".

1.1.4 Ensemble vide

On postule l'existence d'un *ensemble vide* noté \emptyset ayant la propriété suivante : il n'existe pas d'objet x tel que $x \in \emptyset$.

On peut en déduire logiquement que pour tout ensemble E l'on a :

$$\emptyset \subset E. \quad (1.1)$$

L'ensemble vide est donc un sous-ensemble de tout ensemble E . Par suite, l'implication

$$E \subset \emptyset \Rightarrow E = \emptyset \quad (1.2)$$

est vraie, car si l'on joint l'hypothèse $E \subset \emptyset$ à (1.1), il vient $E = \emptyset$ (§ 1.1.3). L'ensemble vide n'a donc qu'un seul sous-ensemble : lui-même.

1.1.5 Ensembles finis

Soient x_1, \dots, x_n ($n \geq 1$) des objets quelconques. La notation $\{x_1, \dots, x_n\}$ désigne l'ensemble dont les éléments sont x_1, \dots, x_n . Cet ensemble est défini plus exactement par la relation

$$y \in \{x_1, \dots, x_n\} \iff y = x_1 \text{ ou } \dots \text{ ou } y = x_n. \quad (1.3)$$

Si les objets x_1, \dots, x_n sont distincts deux à deux, c'est-à-dire si l'on a $x_i \neq x_j$ pour $i \neq j$, l'ensemble $\{x_1, \dots, x_n\}$ est appelé un *ensemble à n éléments*.

Un cas particulier de cette définition est celui d'un ensemble à un seul élément $\{x\}$, défini par

$$y \in \{x\} \iff y = x. \quad (1.4)$$

L'ensemble vide (§ 1.1.4) est appelé aussi *ensemble à zéro élément*. Les ensembles à n éléments ($n \geq 0$) sont appelés *ensembles finis*.

Le nombre d'éléments d'un ensemble fini E sera noté $|E|$. On a :

$$|E| = 0 \iff E = \emptyset. \quad (1.5)$$

1.1.6 Exemple

Soient a, b, c trois objets. L'ensemble $\{a, b, c\}$ est un ensemble à trois éléments si $a \neq b \neq c \neq a$. Par contre si $a \neq b = c$, alors $\{a, b, c\} = \{a, b\} = \{a, c\}$ est un ensemble à deux éléments. L'ordre dans lequel on écrit les éléments entre les parenthèses $\{, \}$ n'a pas d'importance : $\{a, b, c\} = \{c, b, a\} = \dots$

1.1.7 Remarque

Il faut distinguer en général un objet x de l'ensemble $\{x\}$. La distinction est évidente si l'objet x est lui-même un ensemble à n éléments avec $n \neq 1$. On aura dans ce cas $|x| = n \neq |\{x\}| = 1$.

1.1.8 Extension d'une relation

L'ensemble des objets x qui vérifient une relation $R(x)$ est appelé l'*extension en x* de cette relation, et noté $\{x \mid R(x)\}$. Lire : "ensemble des x tels que $R(x)$ ".

Par exemple soit $R(x)$ la relation " x est un nombre entier et $1 < x < 5$ ". On a $\{x \mid R(x)\} = \{2, 3, 4\}$.

1.1.9 Réunion, intersection, complément

Soient X, Y deux sous-ensembles d'un ensemble E . La *réunion* $X \cup Y$, l'*intersection* $X \cap Y$, et le *complément* \bar{X} de X par rapport à E , sont les sous-ensembles de E suivants :

$$X \cup Y = \{x \mid x \in X \text{ ou } x \in Y\} \quad (1.6)$$

$$X \cap Y = \{x \mid x \in X \text{ et } x \in Y\} \quad (1.7)$$

$$\bar{X} = \{x \mid x \in E \text{ et } x \notin X\}. \quad (1.8)$$

Il faut préciser que dans (1.6) la conjonction "ou" est prise au sens mathématique usuel, non exclusif : la relation $x \in X$ ou $x \in Y$ signifie que x appartient à l'un au moins des ensembles X, Y , ce qui n'exclut pas le cas où x appartient aux deux ensembles.

On dit que X et Y sont *disjoints* lorsque $X \cap Y = \emptyset$.

1.1.10 Ensemble des parties d'un ensemble

Pour tout ensemble E on désigne par $\mathbf{P}(E)$ l'*ensemble des parties* (ou sous-ensembles) de E . Ainsi :

$$\mathbf{P}(E) = \{X \mid X \subset E\}. \quad (1.9)$$

La relation d'inclusion $X \subset E$ est donc équivalente à la relation d'appartenance $X \in \mathbf{P}(E)$. Pour tout ensemble E on a $E \in \mathbf{P}(E)$ et $\emptyset \in \mathbf{P}(E)$. Ceci montre que l'ensemble $\mathbf{P}(E)$ n'est jamais vide. En particulier (§ 1.1.4), $\mathbf{P}(\emptyset) = \{\emptyset\}$, donc $|\mathbf{P}(\emptyset)| = 1$.

1.1.11 Exemples

Soient a, b, c des objets distincts. Nous avons :

$$\mathbf{P}(\emptyset) = \{\emptyset\}$$

$$\mathbf{P}(\{a\}) = \{\emptyset, \{a\}\}$$

$$\mathbf{P}(\{a, b\}) = \{\emptyset, \{a\}, \{b\}, \{a, b\}\}$$

$$\mathbf{P}(\{a, b, c\}) = \{\emptyset, \{a\}, \{b\}, \{c\}, \{a, b\}, \{a, c\}, \{b, c\}, \{a, b, c\}\}.$$

On peut montrer de façon générale que pour tout ensemble fini E à n éléments, l'ensemble $\mathbf{P}(E)$ possède 2^n éléments.

1.1.12 Algèbre de Boole

L'ensemble $\mathbf{P}(E)$ muni des opérations $\cup, \cap, \bar{}$, définies au paragraphe 1.1.9 est appelé *algèbre de Boole des sous-ensembles* de E . Les formules principales de cette algèbre sont rappelées ci-dessous. Il suffira au lecteur de s'assurer de leur évidence intuitive, en s'aidant au besoin de diagrammes de Venn (vol. V).

$$\left. \begin{aligned} X \cup Y &= Y \cup X \\ X \cap Y &= Y \cap X \end{aligned} \right\} \quad (1.10)$$

$$\left. \begin{aligned} X \cup (Y \cap Z) &= (X \cup Y) \cap Z \\ X \cap (Y \cup Z) &= (X \cap Y) \cup Z \end{aligned} \right\} \quad (1.11)$$

$$\left. \begin{aligned} X \cup (Y \cap Z) &= (X \cup Y) \cap (X \cup Z) \\ X \cap (Y \cup Z) &= (X \cap Y) \cup (X \cap Z) \end{aligned} \right\} \quad (1.12)$$

$$\left. \begin{aligned} X \cup \emptyset &= X \\ X \cap E &= X \end{aligned} \right\} \quad (1.13)$$

$$\left. \begin{aligned} X \cup \bar{X} &= E \\ X \cap \bar{X} &= \emptyset \end{aligned} \right\} \quad (1.14)$$

$$\left. \begin{aligned} X \cup X &= X \\ X \cap X &= X \end{aligned} \right\} \quad (1.15)$$

$$\left. \begin{aligned} X \cup E &= E \\ X \cap \emptyset &= \emptyset \end{aligned} \right\} \quad (1.16)$$

$$\left. \begin{aligned} X \cup (X \cap Y) &= X \\ X \cap (X \cup Y) &= X \end{aligned} \right\} \quad (1.17)$$

$$\bar{\bar{X}} = X \quad (1.18)$$

$$\left. \begin{aligned} \overline{X \cup Y} &= \bar{X} \cap \bar{Y} \\ \overline{X \cap Y} &= \bar{X} \cup \bar{Y} \end{aligned} \right\} \quad (1.19)$$

$$\left. \begin{aligned} \bar{E} &= \emptyset \\ \overline{\emptyset} &= E \end{aligned} \right\} \quad (1.20)$$

$$\left. \begin{aligned} X \cup Y = \emptyset &\Rightarrow X = \emptyset \text{ et } Y = \emptyset \\ X \cap Y = E &\Rightarrow X = E \text{ et } Y = E \end{aligned} \right\} \quad (1.21)$$

On peut observer dans la liste de formules ci-dessus la *règle de dualité* bien connue de l'algèbre de Boole, à savoir la règle suivante : si dans une formule valable on remplace \cup par \cap , E par \emptyset , et vice-versa (sans toucher au signe $\bar{}$), on obtient une formule valable, appelée *duale* de la première. Les formules ci-dessus sont groupées par paires de formules duales, la formule (1.18) étant sa propre duale.

Les formules suivantes concernent la relation d'inclusion \subset . La règle de dualité s'y applique également, si on la complète en permutant \subset et \supset .

$$\left. \begin{aligned} X \subset Y &\Leftrightarrow X \cup Y = Y \\ X \supset Y &\Leftrightarrow X \cap Y = Y \end{aligned} \right\} \quad (1.22)$$

$$\left. \begin{aligned} X \subset X \cup Y \\ X \supset X \cap Y \end{aligned} \right\} \quad (1.23)$$

$$\left. \begin{aligned} X \subset Z \text{ et } Y \subset Z &\Leftrightarrow X \cup Y \subset Z \\ X \supset Z \text{ et } Y \supset Z &\Leftrightarrow X \cap Y \supset Z \end{aligned} \right\} \quad (1.24)$$

$$\left. \begin{aligned} X \subset Y &\Leftrightarrow X \cap \bar{Y} = \emptyset \\ X \supset Y &\Leftrightarrow X \cup \bar{Y} = E \end{aligned} \right\} \quad (1.25)$$

$$X \subset Y \Leftrightarrow \bar{Y} \subset \bar{X} \quad (1.26)$$

$$\left. \begin{aligned} X \subset Y &\Rightarrow X \cup Z \subset Y \cup Z \\ X \supset Y &\Rightarrow X \cap Z \supset Y \cap Z \end{aligned} \right\} \quad (1.27)$$

1.1.13 Commentaire

Les formules du paragraphe précédent peuvent être démontrées à partir des définitions du paragraphe 1.1.9. Cependant, une démonstration rigoureuse repose sur la logique des propositions, c'est-à-dire sur les règles d'emploi des connecteurs logiques OU, ET, NON qui figurent dans (1.6), (1.7), (1.8). Une telle démonstration n'a sa place que dans un exposé rigoureux de la théorie des ensembles, basé sur la logique formelle des propositions, ce qui n'est pas notre sujet. Nous admettrons donc sans démonstration le formulaire du paragraphe 1.1.12. Cela ne signifie pas que toutes ces formules doivent être considérées comme des axiomes. Il est possible de n'en prendre qu'un certain nombre comme axiomes, et sur cette base de démontrer les autres formules de façon algébrique. C'est le thème de l'exercice proposé ci-dessous.

1.1.14 Exercice

A partir des formules (1.10) à (1.14), considérées comme axiomes, démontrer les formules (1.15) à (1.21). Par exemple, la première des formules (1.15) peut se démontrer comme suit :

$$\begin{aligned} X \cup X &= (X \cup X) \cap E && \text{par (1.13)} \\ &= (X \cup X) \cap (X \cup \bar{X}) && \text{par (1.14)} \\ &= X \cup (X \cap \bar{X}) && \text{par (1.12)} \\ &= X \cup \emptyset && \text{par (1.14)} \\ &= X && \text{par (1.13)}. \end{aligned}$$

Les formules (1.15), (1.16), (1.17) se démontrent de la sorte sans difficulté. Pour la suite, on établira d'abord la formule auxiliaire :

$$(X \cup Y = E \text{ et } X \cap Y = \emptyset) \Rightarrow Y = \bar{X}. \quad (1.28)$$

Celle-ci permet de prouver que $Y = \bar{X} \Rightarrow X = \bar{Y}$, d'où (1.18). Pour démontrer la première des formules (1.19), on posera $Z = \bar{X} \cap \bar{Y}$, et l'on montrera que $(X \cup Y) \cup Z = E$ et $(X \cup Y) \cap Z = \emptyset$. On obtient alors $Z = \overline{X \cup Y}$ par (1.28).

Ajouter ensuite aux axiomes (1.10) à (1.14) les formules (1.22), et démontrer (1.23) à (1.27).

1.1.15 Fonctions

Une *fonction* $f : A \rightarrow B$ est définie par les données suivantes :

- un ensemble A appelé *domaine* de f
- un ensemble B appelé *codomaine* de f
- une règle qui associe à chaque élément x de A un élément de B désigné par $f(x)$.

Une fonction $f : A \rightarrow B$ est appelée aussi une *application* de A dans B .

Deux fonctions $f : A \rightarrow B$, $f' : A' \rightarrow B'$ sont *égales* si et seulement si : $A = A'$, $B = B'$, et $f(x) = f'(x)$ pour tout $x \in A$. On écrit alors $f = f'$.

Une fonction $f: A \rightarrow B$ est dite *injective* si pour deux éléments x_1, x_2 quelconques de A

$$x_1 \neq x_2 \implies f(x_1) \neq f(x_2). \quad (1.29)$$

Une fonction $f: A \rightarrow B$ est dite *surjective* si pour tout $y \in B$ il existe un $x \in A$ tel que $y = f(x)$.

Une fonction $f: A \rightarrow B$ est dite *bijective* si elle est à la fois injective et surjective. Ces deux propriétés peuvent s'exprimer par la phrase : pour tout $y \in B$ il existe un $x \in A$ et un seul tel que $y = f(x)$. On définit alors la fonction bijective $f^{-1}: B \rightarrow A$, appelée *réciproque* de f , par la règle

$$x = f^{-1}(y) \iff y = f(x). \quad (1.30)$$

Une fonction injective (resp. surjective, bijective) est appelée aussi une *injection* (resp. une *surjection*, une *bijection*).

1.1.16 Exercice

Soient A et B deux ensembles finis (§ 1.1.5). Vérifier les assertions suivantes. Il existe une injection $f: A \rightarrow B$ si et seulement si $|A| \leq |B|$. Il existe une surjection $f: A \rightarrow B$ si et seulement si $|A| \geq |B|$. Il existe une bijection $f: A \rightarrow B$ si et seulement si $|A| = |B|$.

1.1.17 Représentations

Une bijection $f: A \rightarrow B$ est appelée parfois une *représentation* de l'ensemble A par l'ensemble B . Pour tout $x \in A$, l'élément $y = f(x)$ de B *représente* x selon f . Réciproquement, x représente y selon f^{-1} .

Lorsque la bijection f est particulièrement "évidente", on utilise fréquemment un élément y de B comme *notation* de l'élément x de A qu'il représente, et réciproquement. On écrit alors abusivement $y = x$ au lieu de $y = f(x)$. On annonce cette pratique en disant qu'on *identifie* les ensembles A et B au moyen de la bijection f . Un premier exemple est donné au paragraphe suivant.

1.1.18 Convention

Soient E un ensemble non vide et $\mathbf{P}_1(E)$ l'ensemble des sous-ensembles $\{x\}$ à un seul élément de E . Il existe une bijection évidente $f: E \rightarrow \mathbf{P}_1(E)$, à savoir la fonction f telle que $f(x) = \{x\}$ pour tout $x \in E$. On identifiera E et $\mathbf{P}_1(E)$ au moyen de cette bijection. Tout symbole désignant un élément x de E sera utilisé pour désigner le sous-ensemble $\{x\}$ de E , et réciproquement. Un même symbole pourra donc avoir des significations distinctes selon le contexte. Ce genre de convention simplifie l'écriture.

1.2 PRODUITS CARTÉSIENS

1.2.1 Couples

Soient x, x', y, y' des objets entre lesquels on a les égalités $x = x'$ et $y = y'$. On peut résumer ces deux égalités par l'unique égalité $(x, y) = (x', y')$. On introduit ainsi la notion de couple (x, y) soumise à l'axiome suivant :

$$(x, y) = (x', y') \iff x = x' \text{ et } y = y'. \quad (1.31)$$

Cet axiome implique que $(x, y) = (y, x)$ si et seulement si $x = y$. Pour cette raison, un couple est appelé parfois un *couple ordonné*.

1.2.2 Produit cartésien

Le *produit cartésien* d'un ensemble A par un ensemble B , noté $A \times B$, est l'ensemble des couples (x, y) tels que $x \in A$ et $y \in B$. Par exemple :

$$\{0, 1\} \times \{0, 1, 2\} = \{(0, 0), (0, 1), (0, 2), (1, 0), (1, 1), (1, 2)\}.$$

1.2.3 Propriétés du produit cartésien

$$(A = \emptyset \text{ ou } B = \emptyset) \iff A \times B = \emptyset \quad (1.32)$$

$$(A \subset A' \text{ et } B \subset B') \implies A \times B \subset A' \times B' \quad (1.33)$$

Si $A \times B \neq \emptyset$, alors

$$A \times B \subset A' \times B' \implies A \subset A' \text{ et } B \subset B'. \quad (1.34)$$

$$\left. \begin{aligned} (A \cup B) \times C &= (A \times C) \cup (B \times C) \\ C \times (A \cup B) &= (C \times A) \cup (C \times B) \end{aligned} \right\} \quad (1.35)$$

$$\left. \begin{aligned} (A \cap B) \times C &= (A \times C) \cap (B \times C) \\ C \times (A \cap B) &= (C \times A) \cap (C \times B) \end{aligned} \right\} \quad (1.36)$$

$$(A \times B) \cap (A' \times B') = (A \cap A') \times (B \cap B'). \quad (1.37)$$

Si A et B sont des ensembles finis, alors

$$|A \times B| = |A| \cdot |B|. \quad (1.38)$$

La vérification de ces formules est laissée au lecteur. On obtient (1.36) comme cas particulier de (1.37) en posant soit $B = B'$ soit $A = A'$.

1.2.4 Exercice

Montrer que si $A \times B = \emptyset$, (1.34) peut être fausse. Donner des ensembles finis A, A', B, B' tels que

$$(A \times B) \cup (A' \times B') \neq (A \cup A') \times (B \cup B').$$

Comparer avec (1.37).

1.2.5 Notations

Soient A, B deux ensembles, et $x \in A, y \in B$. On a :

$$\{x\} \times \{y\} = \{(x, y)\}. \quad (1.39)$$

Si l'on applique la convention du paragraphe 1.1.18 aux sous-ensembles $\{x\}, \{y\}$ de A et B , et au sous-ensemble $\{(x, y)\}$ de $A \times B$, c'est-à-dire si l'on désigne $\{x\}$ par x , $\{y\}$ par y , et $\{(x, y)\}$ par (x, y) , la relation (1.39) s'écrit

$$x \times y = (x, y). \quad (1.40)$$

Ainsi nous désignerons fréquemment un couple $(x, y) \in A \times B$ par $x \times y$.

1.2.6 Convention

Soient A, B, C trois ensembles. Il existe une bijection évidente $f: (A \times B) \times C \rightarrow A \times (B \times C)$. Un élément de $(A \times B) \times C$ est un couple $((x, y), z)$ tel que $(x, y) \in A \times B$ et $z \in C$. La bijection évidente est $f((x, y), z) = (x, (y, z))$. On identifiera $(A \times B) \times C$ et $A \times (B \times C)$ au moyen de cette bijection (§ 1.1.17). Cela permet de considérer le produit cartésien comme associatif. La notation $A \times B \times C$ désigne l'un ou l'autre de ces ensembles. La notation (x, y, z) désigne $((x, y), z)$ ou $(x, (y, z))$. L'expression (x, y, z) , qui peut être notée aussi $x \times y \times z$ (§ 1.2.5) est appelée un *triplet*.

La généralisation à n ensembles A_1, \dots, A_n ($n > 1$) est immédiate. Un élément (x_1, \dots, x_n) de $A_1 \times \dots \times A_n$, noté aussi $x_1 \times \dots \times x_n$, est appelé un *n-uple*.

1.3 CORRESPONDANCES

1.3.1 Définitions

Une *correspondance* f entre un ensemble A et un ensemble B , notée $f: A \rightarrow B$, est définie par la donnée des ensembles A et B , appelés respectivement *ensemble de départ* et *ensemble d'arrivée* de f , et par une règle qui associe à tout élément x de A un *sous-ensemble* de B , noté $f(x)$. On a donc $f(x) \subset B$ pour tout $x \in A$, de sorte que f peut être définie comme une application de A dans $\mathbf{P}(B)$.

Etant donné un élément x de A , on dit qu'un élément y de B tel que $y \in f(x)$ *correspond* à x par f .

Deux correspondances $f: A \rightarrow B$ et $f': A' \rightarrow B'$ sont *égales* si et seulement si : $A = A'$, $B = B'$, et $f(x) = f'(x)$ pour tout $x \in A$.

1.3.2 Exemple

Soient $A = \{a, b, c, d\}$ et $B = \{0, 1, 2, 3\}$. La correspondance $f: A \rightarrow B$ telle que $f(a) = \{0, 1\}$, $f(b) = \emptyset$, $f(c) = \{1\}$, $f(d) = \{1, 2\}$

peut être représentée par le tableau 1.1. Toutefois cette représentation n'est pas complète, car elle n'indique pas quel est l'ensemble d'arrivée de f . Une représentation complète de f est le diagramme de la figure 1.2, dans lequel la relation $y \in f(x)$, c'est-à-dire "y correspond à x", est notée par une flèche $x \rightarrow y$.

x	$f(x)$
a	$\{0, 1\}$
b	\emptyset
c	$\{1\}$
d	$\{1, 2\}$

Tableau 1.1

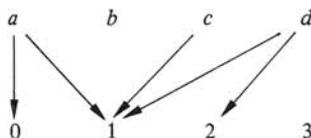


Fig. 1.2

1.3.3 Définition

Soient $f: A \rightarrow B$ une correspondance, et X un sous-ensemble de A . On appelle *transformé de X par f* , et l'on note $f(X)$, l'ensemble

$$f(X) = \bigcup_{x \in X} f(x). \quad (1.41)$$

L'ensemble $f(X)$ est la réunion des ensembles $f(x) \subset B$ tels que $x \in X$. Pour inclure le cas $X = \emptyset$, on pose

$$f(\emptyset) = \emptyset. \quad (1.42)$$

L'ensemble $f(X)$ est caractérisé par la propriété suivante :

$$y \in f(X) \iff \text{il existe un } x \in X \text{ tel que } y \in f(x). \quad (1.43)$$

1.3.4 Exemple

Si f est la correspondance du tableau 1.1, on a

$$f(\{a, b\}) = f(a) \cup f(b) = \{0, 1\} \cup \emptyset = \{0, 1\}$$

$$f(\{a, d\}) = f(a) \cup f(d) = \{0, 1\} \cup \{1, 2\} = \{0, 1, 2\}.$$

1.3.5 Propriétés

Soient $f: A \rightarrow B$ une correspondance, X et Y deux sous-ensembles quelconques de A . On a

$$X \subset Y \implies f(X) \subset f(Y) \quad (1.44)$$

$$f(X \cup Y) = f(X) \cup f(Y) \quad (1.45)$$

$$f(X \cap Y) \subset f(X) \cap f(Y). \quad (1.46)$$

1.3.6 Démonstration

Supposons que $X \subset Y$. Soit $y \in f(X)$. Selon (1.43), il existe un $x \in X$ tel que $y \in f(x)$. Comme $X \subset Y$, il vient $x \in Y$, d'où $y \in f(Y)$. Ceci prouve que $f(X) \subset f(Y)$.

Montrons que $f(X \cup Y) \subset f(X) \cup f(Y)$. Soit $y \in f(X \cup Y)$. Par (1.43), il existe un $x \in X \cup Y$ tel que $y \in f(x)$. On a $x \in X$ ou $x \in Y$. Par suite $y \in f(X)$ ou $y \in f(Y)$, donc $y \in f(X) \cup f(Y)$.

Montrons que $f(X) \cup f(Y) \subset f(X \cup Y)$. On a $X \subset X \cup Y$, donc $f(X) \subset f(X \cup Y)$ par (1.44). De même $f(Y) \subset f(X \cup Y)$. En vertu de (1.24), $f(X) \cup f(Y) \subset f(X \cup Y)$.

Montrons enfin (1.46). Soit $y \in f(X \cap Y)$. Il existe un $x \in X \cap Y$ tel que $y \in f(x)$. On a $x \in X$ et $x \in Y$, donc $y \in f(X)$ et $y \in f(Y)$. Par conséquent $y \in f(X) \cap f(Y)$.

1.3.7 Exercice

Donner un exemple de correspondance $f: A \rightarrow B$ dont l'ensemble de départ A possède deux sous-ensembles X, Y tels que $f(X \cap Y) \neq f(X) \cap f(Y)$.

1.3.8 Réciproque d'une correspondance

Considérons le diagramme de correspondance $f: A \rightarrow B$ de la figure 1.2. Si l'on inverse le sens de toutes les flèches, on obtient le diagramme d'une correspondance

$B \rightarrow A$, appelée *réciroque* de f , et notée f^{-1} . Pour deux éléments quelconques $x \in A$, $y \in B$, on a donc

$$y \in f(x) \iff x \in f^{-1}(y). \quad (1.47)$$

1.3.9 Propriétés

Soit $f: A \rightarrow B$ une correspondance quelconque. Il est clair que la correspondance réciproque de $f^{-1}: B \rightarrow A$ est elle-même, donc

$$(f^{-1})^{-1} = f. \quad (1.48)$$

Soient $X \subset A$ et $y \in B$. En vertu de (1.43) et (1.47)

$$y \in f(X) \iff f^{-1}(y) \cap X \neq \emptyset. \quad (1.49)$$

1.3.10 Correspondances déterministes

On dit qu'une correspondance $f: A \rightarrow B$ est *déterministe* si pour tout $x \in A$, l'ensemble $f(x) \subset B$ est soit l'ensemble vide, soit un ensemble à un seul élément. Ceci peut s'exprimer par $|f(x)| \leq 1$.

Pour une telle correspondance, la relation $y \in f(x)$ équivaut à $\{y\} = f(x)$. En appliquant la convention du paragraphe 1.1.18, on l'écrira sous la forme $y = f(x)$. De même, étant donné un $x \in A$ et un $y \in B$, pour exprimer la relation

$$f(x) = \emptyset \quad \text{ou} \quad f(x) = \{y\}$$

nous écrivons $f(x) \subset y$ au lieu de $f(x) \subset \{y\}$.

Une correspondance déterministe est appelée souvent une *fonction partielle*.

1.3.11 Correspondances fonctionnelles

Une correspondance $f: A \rightarrow B$ est dite *fonctionnelle* si pour tout $x \in A$, l'ensemble $f(x) \subset B$ possède un élément et un seul. Ceci peut s'exprimer par $|f(x)| = 1$. Il s'ensuit que pour tout ensemble $X \subset A$,

$$X \neq \emptyset \implies f(X) \neq \emptyset. \quad (1.50)$$

Une correspondance fonctionnelle est un cas particulier de correspondance déterministe.

1.3.12 Convention

Toute fonction $f: A \rightarrow B$ (§ 1.1.15) sera assimilée à la correspondance fonctionnelle $f': A \rightarrow B$ telle que $f'(x) = \{f(x)\}$. Selon cette convention, une fonction est considérée comme un cas particulier de fonction partielle (§ 1.3.10). Étant donnée une fonction $f: A \rightarrow B$, nous désignerons par $f(X)$ le transformé d'un ensemble $X \subset A$ par la correspondance fonctionnelle f' associée à f , et par f^{-1} la réciproque de f' .

1.3.13 Exercice

Soit $f: A \rightarrow B$ une correspondance. Exprimer au moyen des ensembles A , B , $f(A)$, $f^{-1}(B)$ et du symbole $=$ les propriétés suivantes de f .

- Pour tout $x \in A$, $f(x) \neq \emptyset$.
- Pour tout $y \in B$, il existe un $x \in A$ tel que $y \in f(x)$.

1.3.14 Exercice

Soit $f : A \rightarrow B$ une correspondance. Montrer que pour tout ensemble X contenu dans $f^{-1}(B)$ on a $X \subset f^{-1}(f(X))$. Montrer que si f est une fonction injective, on a $X = f^{-1}(f(X))$ pour tout $X \subset A$, et que si f est une fonction surjective, on a $Y = f(f^{-1}(Y))$ pour tout $Y \subset B$.

1.3.15 Définition

Soient f et g deux correspondances $A \rightarrow B$. On dit que f est *plus fine* que g , et l'on écrit $f \subset g$, si $f(x) \subset g(x)$ pour tout $x \in A$.

Si g est déterministe, $f \subset g$ implique que f est déterministe. Si f et g sont fonctionnelles, $f \subset g$ équivaut à $f = g$.

1.3.16 Exercice

Soient $f : A \rightarrow B$ une correspondance, X et Y deux sous-ensembles quelconques de A . Montrer que si f est une application injective (§ 1.1.15), la formule (1.46) peut être remplacée par l'égalité $f(X \cap Y) = f(X) \cap f(Y)$. Montrer que si f est une bijection (§ 1.1.15), on a la formule $f(\overline{X}) = \overline{f(X)}$, le premier complément étant pris par rapport à A , et le second par rapport à B . A cet effet, on développera $f(X \cap \overline{X})$ et $f(X \cup \overline{X})$ selon les formules précédentes, et on appliquera (1.28).

1.4 SÉQUENCES**1.4.1 Définitions**

Pour tout entier $n \geq 1$, nous désignons par \mathbf{n} l'ensemble des entiers k tels que $1 \leq k \leq n$. Ainsi

$$1 = \{1\}, \quad 2 = \{1,2\}, \quad 3 = \{1,2,3\}, \text{ etc.}$$

Soit A un ensemble non vide. Une fonction

$$x : \mathbf{n} \rightarrow A$$

(n entier ≥ 1) est appelée une *séquence de longueur n sur A* .

Par exemple, soient $A = \{a, b\}$ un ensemble à deux éléments, et $n = 5$. Le tableau 1.3 représente une application $x : \mathbf{5} \rightarrow A$, donc une séquence de longueur 5 sur A .

k	$x(k)$
1	b
2	a
3	a
4	b
5	a

Tableau 1.3

Une séquence $x : \mathbf{n} \rightarrow A$ est généralement représentée par l'expression $x(1)x(2) \dots x(n)$ qu'on appelle un *mot de longueur n sur A* . Par exemple, la séquence du tableau 1.3 se note simplement *baaba*.

La longueur d'une séquence x est désignée par $lg(x)$. Une séquence x de longueur n sera désignée par $x(1, n)$, ce qui abrège $x(1) \dots x(n)$.

Un ensemble *fini* et *non vide* A est appelé souvent un *alphabet*, lorsqu'on considère des séquences sur cet ensemble.

Deux séquences x, y sur un alphabet A sont *égales* si elles sont égales en tant que fonctions (§ 1.1.15), c'est-à-dire si et seulement si

$$lg(x) = lg(y) \quad \text{et} \quad x(k) = y(k) \quad \text{pour} \quad k = 1, \dots, lg(x). \quad (1.51)$$

L'ensemble des séquences de longueur n sur un alphabet A est désigné par A^n . On identifie A^1 et A .

L'ensemble des séquences de longueur quelconque sur A , c'est-à-dire l'ensemble infini $A^1 \cup A^2 \cup A^3 \cup \dots$ est noté A^* . Par exemple si $A = \{a, b\}$:

$$A^1 = \{a, b\}$$

$$A^2 = \{aa, ab, ba, bb\}$$

$$A^3 = \{aaa, aab, aba, abb, baa, bab, bba, bbb\}$$

1.4.2 Définition : produit de séquences

Soient $x(1, m)$ et $y(1, n)$ deux séquences sur un alphabet A . Le *produit* de x par y , noté xy , est la séquence $z(1, m+n)$ sur A qui est représentée par le mot

$$x(1) \dots x(m)y(1) \dots y(n).$$

On a donc

$$z(k) = \begin{cases} x(k) & \text{pour } k = 1, \dots, m \\ y(k-m) & \text{pour } k = m+1, \dots, m+n. \end{cases} \quad (1.52)$$

1.4.3 Proposition

Pour trois séquences quelconques x, y, z sur un alphabet A , on a

$$(xy)z = x(yz). \quad (1.53)$$

Le produit de séquences étant ainsi associatif, on l'écrira sans parenthèses.

1.4.4 Proposition

Soient x, x', y, y' des séquences sur un alphabet A . Les relations $xy = x'y'$ et $lg(x) = lg(x')$ impliquent $x = x'$ et $y = y'$. De même $xy = x'y'$ et $lg(y) = lg(y')$ impliquent $x = x'$ et $y = y'$.

1.4.5 Définition

Soient X un alphabet et A, B deux ensembles de séquences sur X : $A \subset X^+$, $B \subset X^+$.

On désigne par AB l'ensemble des séquences sur X de la forme xy avec $x \in A$ et $y \in B$.

1.4.6 Exemple

Soient $X = \{a, b\}$, $A = \{ab, baa\}$, $B = \{ab, b\}$. On a $AB = \{abab, abb, baaab, baab\}$.

1.4.7 Propriétés

Pour trois ensembles quelconques $A, B, C \subset X^+$, on a

$$\left. \begin{aligned} A \subset B &\Rightarrow AC \subset BC \\ A \subset B &\Rightarrow CA \subset CB \end{aligned} \right\} \quad (1.54)$$

$$(AB)C = A(BC) \quad (1.55)$$

$$\left. \begin{aligned} A(B \cup C) &= AB \cup AC \\ (A \cup B)C &= AC \cup BC \end{aligned} \right\} \quad (1.56)$$

$$\left. \begin{aligned} A(B \cap C) &\subset AB \cap AC \\ (A \cap B)C &\subset AC \cap BC \end{aligned} \right\} \quad (1.57)$$

$$A\emptyset = \emptyset A = \emptyset. \quad (1.58)$$

1.4.8 Démonstration

Les formules (1.54) et (1.55) sont évidentes. Nous démontrerons la première des formules (1.56). Soit $z \in A(B \cup C)$. Par définition (§ 1.4.5), il existe une séquence $x \in A$ et une séquence $y \in B \cup C$ telles que $z = xy$. On a $y \in B$ ou $y \in C$, donc $xy \in AB$ ou $xy \in AC$. Ceci montre que $A(B \cup C) \subset AB \cup AC$. Démontrons l'inclusion inverse. On a $B \subset B \cup C$, donc $AB \subset A(B \cup C)$ par (1.54). De même $AC \subset A(B \cup C)$. Il vient $AB \cup AC \subset A(B \cup C)$ par (1.24).

La deuxième des formules (1.56) se démontre de façon analogue, de même que les formules (1.57). Enfin (1.58) signifie qu'il n'existe pas de séquence xy telle que $x \in \emptyset$ ou $y \in \emptyset$.

1.4.9 Exercice

Soit $X = \{a, b\}$ un alphabet à deux éléments. Donner trois ensembles de séquences A, B, C sur X tels que $A(B \cap C) \neq AB \cap AC$. Donner quatre ensembles de séquences A, A', B, B' non vides sur X , tels que $A \not\subset A', B \not\subset B'$, et $AB \subset A'B'$.

1.4.10 Exercice

Soit $X = \{a, b\}$ un alphabet à deux éléments. Exprimer de deux manières différentes l'ensemble de séquences

$$\{abbbba, baa, aba, babba\}$$

sous la forme d'un produit de deux ensembles de séquences sur X .

1.4.11 Proposition

Soient A, A', B, B' des ensembles de séquences sur un alphabet X , tels que $A \cup A' \subset X^m$ et $B \cup B' \subset X^n$. Ces hypothèses signifient que A et A' (resp. B et B')

sont deux ensembles de séquences sur X , *toutes de même longueur m* (resp. n). On suppose en outre que $A \neq \emptyset$ et $B \neq \emptyset$. Avec ces hypothèses,

$$AB \subset A'B' \Rightarrow A \subset A' \text{ et } B \subset B'. \quad (1.59)$$

1.4.12 Démonstration

Supposons que $AB \subset A'B'$, et soient $x \in A$, $y \in B$. Il faut montrer que $x \in A'$ et $y \in B'$. On a $xy \in AB$ (§ 1.4.4), donc $xy \in A'B'$ par hypothèse. Il existe donc une séquence $x' \in A'$ et une séquence $y' \in B'$ telles que $xy = x'y'$. Selon les hypothèses faites sur A et A' , on a $lg(x) = lg(x')$. On en conclut par le paragraphe 1.4.4 que $x = x'$ et $y = y'$. Donc $x \in A'$ et $y \in B'$.

1.4.13 Produit cartésien de séquences

Soient X et Y deux alphabets, et $x(1, n)$, $y(1, n)$ deux séquences *de même longueur n* , la première sur X et la seconde sur Y . Nous appelons *produit cartésien* de x par y , et nous désignons par $x \times y$ la séquence de couples

$$x \times y = (x(1), y(1)) (x(2), y(2)) \dots (x(n), y(n)) \quad (1.60)$$

sur l'alphabet $X \times Y$.

Un couple (a, b) est noté parfois verticalement :

$$\begin{pmatrix} a \\ b \end{pmatrix}$$

Avec cette notation, (1.60) s'écrit plus clairement

$$x \times y = \begin{pmatrix} x(1) \\ y(1) \end{pmatrix} \begin{pmatrix} x(2) \\ y(2) \end{pmatrix} \dots \begin{pmatrix} x(n) \\ y(n) \end{pmatrix}. \quad (1.61)$$

Il faut noter que l'expression $x \times y$ n'a de sens, en tant que séquence, que si $lg(x) = lg(y)$, et que

$$lg(x \times y) = lg(x) = lg(y). \quad (1.62)$$

Les propositions suivantes sont évidentes.

1.4.14 Proposition

Soient X et Y deux alphabets. Pour toute séquence z sur l'alphabet $X \times Y$, il existe un couple de séquences de même longueur $x \in X^+$, $y \in Y^+$ et un seul tel que $z = x \times y$.

1.4.15 Proposition

Soient x et x' deux séquences sur X , y et y' deux séquences sur Y , telles que $lg(x) = lg(y)$ et $lg(x') = lg(y')$. On a

$$(x \times y)(x' \times y') = xx' \times yy'. \quad (1.63)$$

1.4.16 Définition

Soient X et Y deux alphabets, $A \subset X^n$ et $B \subset Y^n$ deux ensembles de séquences, *toutes de même longueur* n . On désigne par $A \times B$ l'ensemble des séquences $x \times y$ sur l'alphabet $X \times Y$ telles que $x \in A$ et $y \in B$.

1.4.17 Commentaire

Au paragraphe 1.2.2 nous avons défini le produit cartésien $A \times B$ comme l'ensemble des couples $(x \in A, y \in B)$. Le produit $A \times B$ du paragraphe 1.4.16 a donc un sens différent. Cependant, avec les hypothèses faites sur A et B (§ 1.4.16), les deux types de produits sont identifiables (§ 1.1.18), un couple de séquences (x, y) de même longueur n étant représentable par la séquence de couples (1.61) et vice-versa.

1.4.18 Proposition

Soient X et Y deux alphabets, $A \subset X^m$ et $B \subset Y^m$, $A' \subset X^n$ et $B' \subset Y^n$. On a

$$(A \times B)(A' \times B') = AA' \times BB'. \quad (1.64)$$

Cette proposition généralise celle du paragraphe 1.4.15. Sa démonstration, basée sur (1.63), est laissée au lecteur.

1.5 GRAPHERS

1.5.1 Définitions

Nous appelons *graphe* tout ensemble fini G dont chaque élément est un triplet (x, y, z) . Un élément (x, y, z) de G est appelé une *arête* de G . L'*origine* de cette arête est l'objet x , son *étiquette* est l'objet y , et son *extrémité* est l'objet z . L'origine ou l'extrémité d'une arête de G est un *sommet* de G .

1.5.2 Exemple

L'ensemble

$$G = \{(p, a, q), (p, a, s), (p, b, r), (q, s, p), (q, a, q), (q, r, q), (s, b, p)\}$$

est un graphe dont les sommets sont p, q, r, s .

1.5.3 Notations

Un graphe G est souvent représenté par un diagramme dans lequel une arête (x, y, z) est notée :

$$x \xrightarrow{y} z$$

Une arête (x, y, x) est notée généralement :

$$x \begin{array}{c} \circlearrowleft \\ y \end{array} x$$

Deux arêtes $(x, y, z), (x, y', z)$ sont notées :

$$x \xrightarrow{y, y'} z$$

Ainsi le graphe du paragraphe 1.5.2 peut être représenté par le diagramme de la figure 1.4. Il faut noter que cette représentation d'un graphe n'est pas unique. Rien n'empêche par exemple de représenter plusieurs fois un même sommet pour éviter un enchevêtrement de flèches.

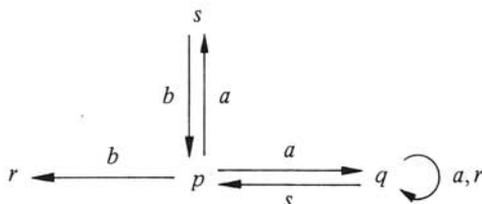


Fig. 1.4

1.5.4 Chemins

Un *chemin d'un graphe* G (ou *dans* un graphe G) est une séquence d'arêtes de G

$$a_1 a_2 \dots a_n$$

telle que si $n > 1$, alors pour $i = 1, \dots, n - 1$ l'extrémité de a_i est l'origine de a_{i+1} . L'*étiquette du chemin* est la séquence $x_1 \dots x_n$ des étiquettes des arêtes a_1, \dots, a_n . L'*origine du chemin* est l'origine de a_1 . L'*extrémité du chemin* est l'extrémité de a_n .

Pour $n = 1$, la notion de chemin s'identifie avec celle d'arête.

1.5.5 Exemple

La séquence

$$(s, b, p) (p, a, q) (q, a, q) (q, a, q) (q, s, p)$$

est un chemin de longueur 5 du graphe G du paragraphe 1.5.2 (fig. 1.4). L'origine de ce chemin est s , son extrémité p , et son étiquette est la séquence $baaas$.

1.5.6 Notations

Si c désigne un chemin d'origine p et d'extrémité q d'un graphe G , on écrit $c : p \rightarrow q$.

L'étiquette de c est désignée par $|c|$.

1.5.7 Propositions

Soient $c : p \rightarrow q$ et $c' : q \rightarrow r$ deux chemins d'un graphe G , tels que l'extrémité de c coïncide avec l'origine de c' . Alors le produit cc' (au sens du paragraphe 1.4.2, c et c' étant des séquences d'arêtes de G) est un chemin

$$cc' : p \rightarrow r$$

et l'on a la relation

$$|cc'| = |c| |c'| \tag{1.65}$$

entre les étiquettes de ces chemins.

Pour tout chemin c d'un graphe G :

$$lg(c) = lg(|c|). \quad (1.66)$$

Ces propositions sont évidentes.

1.5.8 Remarque

Le produit de deux chemins $c : p \rightarrow q$, $c' : p' \rightarrow r$ d'un graphe G est toujours défini, au sens du paragraphe 1.4.2. Cependant, ce produit cc' n'est un chemin de G que si $q = p'$. Néanmoins, la relation (1.65) reste valable si $q \neq p'$.

MACHINES

2.1 NOTIONS GÉNÉRALES

2.1.1 Introduction

Une *machine*, dans le présent volume, est un dispositif de traitement de l'information, schématisé sous la forme très grossière de la figure 2.1, c'est-à-dire considéré comme une "boîte noire" munie d'entrées et de sorties par lesquelles le dispositif reçoit et émet des messages. Un message n'est rien d'autre qu'une séquence (§ 1.4.1) sur un certain alphabet (ensemble de caractères).

La propriété essentielle d'une telle machine est la correspondance qui existe entre les messages reçus et les messages émis. Son fonctionnement peut être ainsi représenté par une correspondance $M : A \rightarrow B$ (§ 1.3.1), où A est un ensemble de messages d'entrée, et B un ensemble de messages de sortie. Ce modèle des dispositifs de traitement de l'information fait complètement abstraction de leur nature physique particulière. Il peut représenter aussi bien un système logique (vol. V) qu'un organisme administratif.

Le but du présent chapitre est de préciser ce modèle général, et d'introduire les types de machine particuliers qui font l'objet du livre : machines combinatoires, machines séquentielles, et leurs combinaisons.

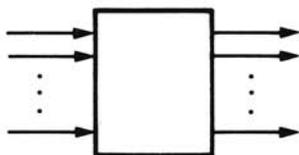


Fig. 2.1

2.1.2 Définitions

Une *machine à une entrée et une sortie* est représentée par un schéma tel que celui de la figure 2.2. Une telle machine est caractérisée par les données suivantes.

Un alphabet X , appelé *alphabet d'entrée* de la machine, est associé à son entrée, qui est marquée x dans le schéma. Une séquence sur cet alphabet est appelée une *séquence d'entrée* de la machine. La lettre x du schéma est appelée la *variable d'entrée du schéma*. Elle représente une *séquence d'entrée indéterminée* $x \in X^+$.

De même, un *alphabet de sortie* Y est associé à la sortie de la machine, marquée y dans le schéma. Une séquence sur cet alphabet est une *séquence de sortie* de la machine. La lettre y est la *variable de sortie du schéma*. Elle représente une *séquence de sortie indéterminée* $y \in Y^+$, correspondant à la séquence x .

Le symbole M du schéma représente le *fonctionnement* de la machine, à savoir une correspondance

$$M : X^+ \longrightarrow Y^+. \quad (2.1)$$

Conformément à la notion générale de correspondance (§ 1.3.1), ce fonctionnement associe à toute séquence d'entrée x un *ensemble de séquences de sortie* $M(x) \subset Y^+$. Une séquence de sortie y *correspond* à une séquence d'entrée x si

$$y \in M(x). \quad (2.2)$$

Nous supposons toujours que le fonctionnement M d'une machine *conserve la longueur des séquences*, c'est-à-dire que

$$y \in M(x) \Rightarrow \text{lg}(y) = \text{lg}(x). \quad (2.3)$$



Fig. 2.2

2.1.3 Commentaire

Il est clair que l'élément essentiel de la description d'une machine est la donnée de son fonctionnement, la correspondance (2.1). Pour cette raison, on dira souvent qu'on a défini une machine lorsqu'on a donné une correspondance de la forme (2.1) vérifiant la condition (2.3).

2.1.4 Exemple

Soient X un alphabet quelconque, et $R : X^+ \rightarrow X^+$ la correspondance définie comme suit : pour toute séquence $x = x(1) \dots x(n)$ de longueur n sur X , $R(x)$ est la séquence renversée $x(n) \dots x(1)$. Cette correspondance conserve la longueur des séquences. Nous pouvons l'appeler une machine. Elle est en outre fonctionnelle au sens du paragraphe 1.3.11.

2.1.5 Exemple

Soit G le graphe de la figure 2.3. Il se compose des quatre arêtes suivantes :

$$\alpha = (a, 0, b) \quad \beta = (b, 1, b) \quad \gamma = (b, 1, c) \quad \delta = (c, 0, a).$$

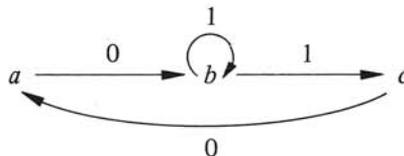


Fig. 2.3

Posons $X = \{0, 1\}$, et soit $T : X^+ \rightarrow G^+$ la correspondance définie comme suit : pour toute séquence $x \in X^+$, $T(x)$ est l'ensemble des chemins d'étiquette x dans G .

Par exemple :

$$\mathcal{T}(01) = \{\alpha\beta, \alpha\gamma\} \quad (\text{deux chemins d'étiquette } 01)$$

$$\mathcal{T}(000) = \emptyset \quad (\text{pas de chemin d'étiquette } 000).$$

La correspondance \mathcal{T} conserve la longueur des séquences. Nous pouvons l'appeler une machine.

2.1.6 Machines à entrées et sorties multiples

Une machine à m entrées ($m > 1$) et n sorties ($n > 1$) est représentée par le schéma de la figure 2.4. A chacune de ses entrées et chacune de ses sorties est associé un alphabet. La machine possède donc m alphabets d'entrée X_1, \dots, X_m et n alphabets de sortie Y_1, \dots, Y_n .

Une séquence d'entrée de la machine est une séquence x sur l'alphabet $X_1 \times \dots \times X_m$. Une telle séquence est le produit cartésien $x = x_1 \times \dots \times x_m$ de m séquences $x_i \in X_i^+$ ($i = 1, \dots, m$) (§ 1.4.13 et 1.4.14).

De même, une séquence de sortie de la machine est une séquence y sur l'alphabet $Y_1 \times \dots \times Y_n$. Elle est le produit $y = y_1 \times \dots \times y_n$ de n séquences $y_i \in Y_i^+$ ($i = 1, \dots, n$).

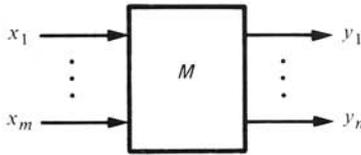


Fig. 2.4

Le fonctionnement de la machine fait correspondre à toute séquence d'entrée un ensemble de séquences de sortie. C'est donc une correspondance

$$M : (X_1 \times \dots \times X_m)^+ \longrightarrow (Y_1 \times \dots \times Y_n)^+. \quad (2.4)$$

Une séquence de sortie $y = y_1 \times \dots \times y_n$ correspond à une séquence d'entrée $x = x_1 \times \dots \times x_m$ si

$$y_1 \times \dots \times y_n \in M(x_1 \times \dots \times x_m). \quad (2.5)$$

On supposera toujours qu'une machine conserve la longueur des séquences. Dans le schéma de la figure 2.4, les variables d'entrée x_i ($i = 1, \dots, m$) et les variables de sortie y_j ($j = 1, \dots, n$) représentent des séquences indéterminées sur les alphabets respectifs X_i ($i = 1, \dots, m$) et Y_j ($j = 1, \dots, n$). Elles vérifient la relation de correspondance (2.5), et représentent donc des séquences de même longueur.

Nous appellerons format d'une machine le couple d'entiers (m, n) , où m est le nombre d'entrées, et n le nombre de sorties de la machine.

2.1.7 Machines binaires

Une machine est dite binaire si chacun de ses alphabets d'entrée et chacun de ses alphabets de sortie est l'alphabet $\{0, 1\}$. Cet alphabet sera désormais par \mathbf{B} . Pour une machine binaire on a donc $X_1 = \dots = X_m = Y_1 = \dots = Y_n = \mathbf{B}$ dans (2.4).

2.1.8 Exemple : machine OU

Un système logique (vol. V) peut être considéré comme une machine au sens du paragraphe 2.1.6. Il suffit pour cela d'échantillonner ses signaux logiques d'entrée et de sortie de façon appropriée. Nous traitons dans ce paragraphe un système combinatoire élémentaire; au paragraphe 2.1.9, un système séquentiel synchronisé élémentaire; au paragraphe 2.1.10, un système séquentiel asynchrone élémentaire.

Une porte OU (fig. 2.5) est un système logique combinatoire élémentaire bien connu (sect. V.1.3). On peut le représenter par une machine binaire M à deux entrées et une sortie (fig. 2.5). Les alphabets d'entrée de M sont $X_1 = X_2 = \mathbf{B}$, et son alphabet de sortie $Y = \mathbf{B}$. M est donc une correspondance $(\mathbf{B} \times \mathbf{B})^+ \rightarrow \mathbf{B}^+$, conservant la longueur des séquences. Une séquence d'entrée de longueur n est une séquence

$$x(1, n) = x_1(1, n) \times x_2(1, n) = \begin{pmatrix} x_1(1) \\ x_2(1) \end{pmatrix} \dots \begin{pmatrix} x_1(n) \\ x_2(n) \end{pmatrix} \quad (2.6)$$

avec $x_i(k) \in \{0, 1\}$. Par exemple :

$$x(1, 3) = 011 \times 101 = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

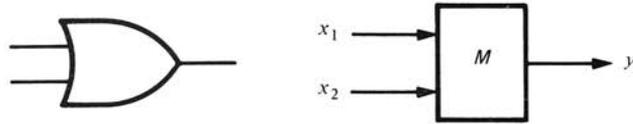


Fig. 2.5

La correspondance M associe à toute séquence d'entrée (2.6) une séquence de sortie $y(1, n)$ et une seule. Cette séquence de sortie $y = M(x)$ est la somme logique bit par bit des séquences x_1 et x_2 :

$$y(k) = x_1(k) + x_2(k) \quad (k = 1, \dots, n). \quad (2.7)$$

Ce modèle d'une porte OU résulte d'un échantillonnage. Le chronogramme de la figure 2.6 représente des signaux logiques $x_1(t)$, $x_2(t)$, $y(t)$ observables sur les entrées et la sortie d'une porte OU. Le temps t est une variable réelle (continue). Une séquence d'entrée (2.6) et la séquence de sortie $y(1, n)$ correspondante peuvent être définies en choisissant une suite de n instants de mesure distincts $t_1 < t_2 < \dots < t_n$, appelés ins-

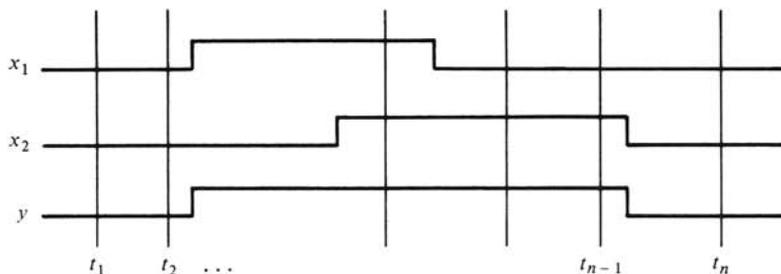


Fig. 2.6

tants d'échantillonnage des signaux. Les séquences $x_1(1, n)$, $x_2(1, n)$, $y(1, n)$ sont les séquences des valeurs des signaux x_1, x_2, y aux instants t_1, \dots, t_n .

Le chronogramme de la figure 2.6 est celui d'une porte idéale, c'est-à-dire sans retard. Dans ce cas, la correspondance M définie par (2.7) est vérifiée pour n'importe quel choix de n instants d'échantillonnage. Ce n'est plus le cas si la porte considérée est affectée d'un certain retard Δ (fig. 2.7). La relation (2.7) ne sera pas vérifiée si un instant d'échantillonnage est choisi comme t_2 dans la figure 2.7.

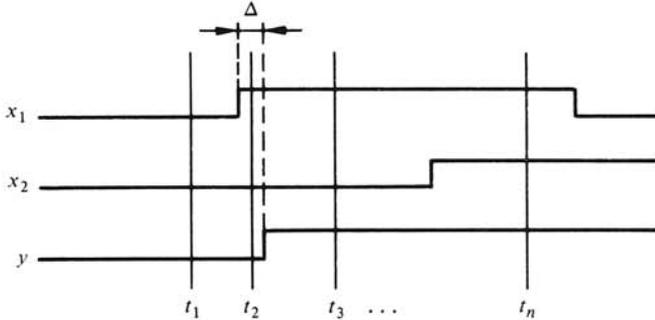


Fig. 2.7

2.1.9 Exemple : machine JK

La figure 2.8 représente une bascule bistable JK, et son graphe des états, selon les notations du chapitre V.3.

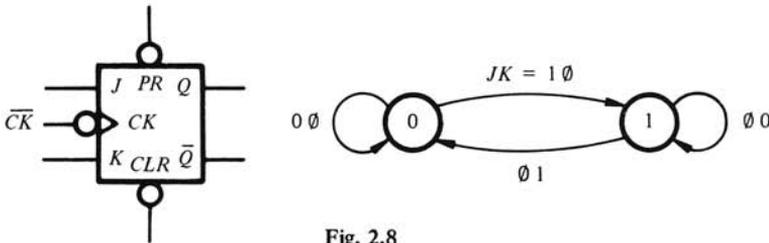


Fig. 2.8

Le graphe décrit le fonctionnement *synchrone* de la bascule, à savoir (chap. V.3) : les entrées PR , CLR sont inactives, et les entrées J, K ne varient pas pendant que $CK = 0$. A ces conditions, si l'on choisit n instants d'échantillonnage $t_1 < t_2 < \dots < t_n$ de la façon illustrée par la figure 2.9 (c'est-à-dire tels que $CK(t_i) = 0$ et CK varie deux fois entre t_i et t_{i+1}), alors chacun des triplets

$$Q(t_i) \xrightarrow{J(t_i) \times K(t_i)} Q(t_{i+1})$$

est une arête du graphe.

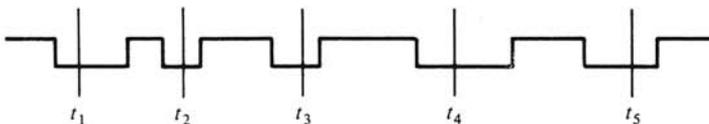


Fig. 2.9

En faisant abstraction de la sortie \bar{Q} , nous considérons ce système comme une machine binaire M à deux entrées et une sortie (fig. 2.10). Le graphe G de cette figure reproduit celui de la figure 2.8 avec d'autres notations. La correspondance $M : (\mathbf{B} \times \mathbf{B})^+ \rightarrow \mathbf{B}^+$ peut être définie au moyen de ce graphe. Considérons une séquence d'entrée $x = x(1) \dots x(n)$, par exemple la séquence $x = x(1) x(2) x(3) = (0 \times 0)(1 \times 0)(1 \times 0)$. Il y a dans le graphe G deux chemins d'étiquette x , à savoir les chemins

$$c_1 = (0, 0 \times 0, 0) (0, 1 \times 0, 1) (1, 1 \times 0, 1)$$

$$c_2 = (1, 0 \times 0, 1) (1, 1 \times 0, 1) (1, 1 \times 0, 1).$$

Ces chemins peuvent être notés aussi

$$c_1 : 0 \xrightarrow{0 \times 0} 0 \xrightarrow{1 \times 0} 1 \xrightarrow{1 \times 0} 1$$

$$c_2 : 1 \xrightarrow{0 \times 0} 1 \xrightarrow{1 \times 0} 1 \xrightarrow{1 \times 0} 1.$$

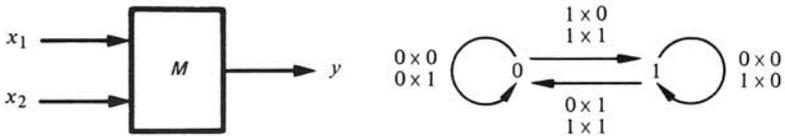


Fig. 2.10

Si pour trois instants d'échantillonnage consécutifs t_1, t_2, t_3 les valeurs des signaux J, K d'une bascule JK sont $0 \times 0, 1 \times 0, 1 \times 0$, alors la séquence $Q(t_1) Q(t_2) Q(t_3)$ peut être soit 001, soit 111. Ces deux séquences sont les séquences des origines des arêtes des chemins c_1 et c_2 . La correspondance M associe à toute séquence d'entrée $x = x(1) \dots x(n)$ l'ensemble des séquences $y(1) \dots y(n)$ telles qu'il existe un chemin

$$y(1) \xrightarrow{x(1)} y(2) \xrightarrow{x(2)} \dots y(n) \xrightarrow{x(n)} y(n+1)$$

dans G . On a donc

$$M((0 \times 0)(1 \times 0)(1 \times 0)) = \{001, 111\}$$

ou encore

$$M(011 \times 000) = \{00, 11\} 1.$$

2.1.10 Exemple : élément de mémoire $\bar{s}\bar{r}$

La figure 2.11 représente un élément de mémoire $\bar{s}\bar{r}$ (sect. V.3.1). La figure 2.12 est le chronogramme de trois signaux s, r, y caractéristiques de ce système séquentiel asynchrone. On suppose que celui-ci fonctionne en mode normal, c'est-à-dire que le produit logique $s(t)r(t)$ est nul à tout instant t , et que l'intervalle de temps écoulé entre deux variations consécutives de l'état d'entrée (s, r) est toujours supérieur au retard Δ du système.

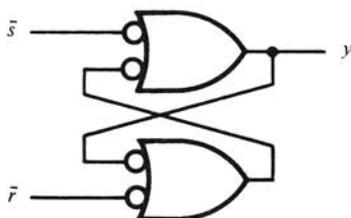


Fig. 2.11

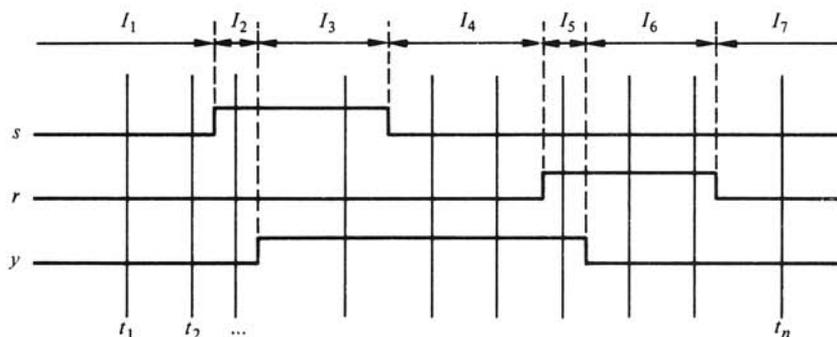


Fig. 2.12

Considérons une suite de n instants d'échantillonnage $t_1 < t_2 < \dots < t_n$ vérifiant les conditions suivantes :

- dans tout intervalle de temps I délimité par deux variations successives de l'état total (s, r, y) il y a au moins un instant d'échantillonnage. Ces intervalles sont désignés par I_1, \dots, I_7 dans la figure 2.12;
- dans chaque intervalle de retard Δ précédant une variation de y (intervalles I_2, I_5 dans la figure), il y a un instant d'échantillonnage et un seul.

A ces conditions, le graphe G de la figure 2.13 représente le fonctionnement du système de la manière suivante : pour chaque instant t_i , le triplet

$$y(t_i) \xrightarrow{s(t_i) \times r(t_i)} y(t_{i+1})$$

est une arête de G .

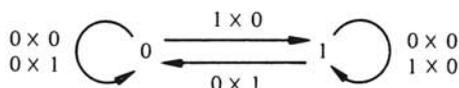


Fig. 2.13

Pour définir une machine $M : (\mathbf{B} \times \mathbf{B})^* \rightarrow \mathbf{B}^*$ qui représente ce système (fig. 2.14), il faut compléter le graphe G (fig. 2.13), de façon que l'ensemble de séquences de sortie $M(x)$ soit défini pour toute séquence d'entrée $x = x(1)x(2)\dots x(n)$, c'est-à-dire même si $x(i) = 1 \times 1$. La façon naturelle de procéder est d'ajouter au graphe G les quatre arêtes notées dans la figure 2.15. Elles expriment le fait que $y(t_{i+1})$ est indéterminé lorsque $r(t_i) \times s(t_i) = 1 \times 1$. En réunissant les graphes des figures 2.13 et 2.15, on obtient le gra-

phe H de la figure 2.14. Le fonctionnement M est défini au moyen de H comme au paragraphe 2.1.9. Une séquence de sortie $y(1, n)$ correspond à une séquence d'entrée $x(1, n)$ si et seulement s'il existe dans H un chemin

$$y(1) \xrightarrow{x(1)} y(2) \xrightarrow{x(2)} \dots y(n) \xrightarrow{x(n)} y(n+1).$$

On a par exemple

$$M(011 \times 010) = M((0 \times 0)(1 \times 1)(1 \times 0)) = \{000, 001, 110, 111\}.$$

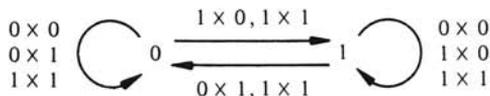


Fig. 2.14

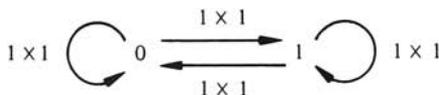


Fig. 2.15

2.2 MACHINES COMPOSÉES

2.2.1 Introduction

Plusieurs machines (§ 2.1.2 et 2.1.6) peuvent être assemblées, c'est-à-dire reliées entre elles pour former une nouvelle machine, de même que l'on assemble des composants logiques pour former des systèmes logiques. On peut aussi relier par des connexions plusieurs bornes (entrées ou sorties) d'une même machine. Les opérations de ce genre sont appelées *opérations de composition*. Les machines assemblées sont les *composantes*, et la machine résultante est la *machine composée*. On définit dans cette section certaines *opérations de composition fondamentales* (juxtaposition, connexion, projection, permutation). Toute opération de composition complexe peut être décomposée en une succession de ces opérations fondamentales.

2.2.2 Réduction de format

Il est toujours possible de considérer deux entrées adjacentes d'une même machine comme une seule entrée, l'alphabet associé à celle-ci étant le produit cartésien des alphabets des deux entrées considérées. Ce faisant, on réduit le format de la machine sans changer son fonctionnement.

Soient par exemple X_1, X_2, X_3 les alphabets d'entrée et Y l'alphabet de sortie d'une machine A (fig. 2.16) de format $(3, 1)$.

Le fonctionnement de A est une correspondance

$$M : (X_1 \times X_2 \times X_3)^+ \longrightarrow Y^+.$$

Posons $(X_2 \times X_3) = X_{23}$. Par l'associativité du produit cartésien (§ 1.2.6), on a

$$M : (X_1 \times X_{23})^+ \longrightarrow Y^+.$$

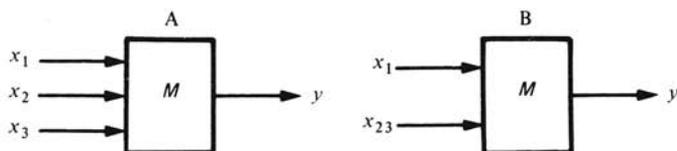


Fig. 2.16

On peut donc représenter la machine A par une machine B de format $(2, 1)$ ayant le même fonctionnement que A. Cette machine B (fig. 2.16) a pour alphabets d'entrée les ensembles X_1 et X_{23} . La variable d'entrée x_{23} est une séquence indéterminée sur l'alphabet X_{23} .

La même opération peut se faire pour deux sorties adjacentes d'une machine. A la limite, toute machine peut être représentée par une machine à une entrée et une sortie. Nous ferons fréquemment un usage implicite de cette représentation.

2.2.3 Juxtaposition

L'opération de *juxtaposition* est illustrée par la figure 2.17. Elle porte sur deux machines composantes

$$M : X^+ \longrightarrow Y^+ ; \quad N : U^+ \longrightarrow W^+$$

et fournit une machine composée

$$R : (X \times U)^+ \longrightarrow (Y \times W)^+.$$

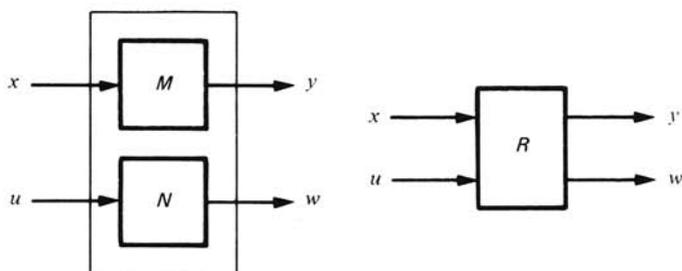


Fig. 2.17

Les variables x, u, y, w de la figure 2.17 sont des séquences indéterminées sur les alphabets respectifs X, U, Y, W . La machine composée R est définie par la relation

$$y \times w \in R(x \times u) \iff y \in M(x) \text{ et } w \in N(u) \quad (2.8)$$

On notera que la machine composée R dépend de l'ordre dans lequel sont prises les composantes M, N .

2.2.4 Connexion sortie-entrée

Nous distinguerons deux types d'opération de connexion (§ 2.2.4 et 2.2.9). Chacune de ces opérations porte sur une seule machine. La *connexion sortie-entrée* est illustrée par la figure 2.18. Elle porte sur une machine

$$M : (X_1 \times \dots \times X_m)^+ \longrightarrow (Y_1 \times \dots \times Y_n)^+$$

vérifiant la *condition*

$$X_1 = Y_1. \quad (2.9)$$

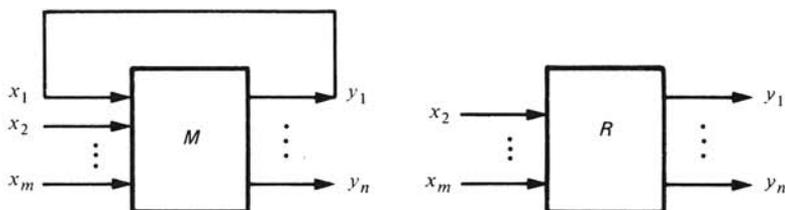


Fig. 2.18

La machine composée est la machine

$$R : (X_2 \times \dots \times X_m)^+ \longrightarrow (Y_1 \times \dots \times Y_n)^+$$

telle que

$$y_1 \times \dots \times y_n \in R(x_2 \times \dots \times x_m) \iff y_1 \times \dots \times y_n \in M(y_1 \times x_2 \times \dots \times x_m) \quad (2.10)$$

En français : une séquence $y_1 \times \dots \times y_n$ correspond à une séquence $x_2 \times \dots \times x_m$ par la machine composée R , si et seulement si elle correspond à la séquence $y_1 \times x_2 \times \dots \times x_m$ par la machine composante M .

Le format de la machine composée est $(m-1, n)$. On suppose ici que $m > 1$. Le cas $m = 1$ est traité au paragraphe 2.2.8.

2.2.5 Exemple

Supposons que la machine M de la figure 2.19 soit la machine OU (§ 2.1.8).

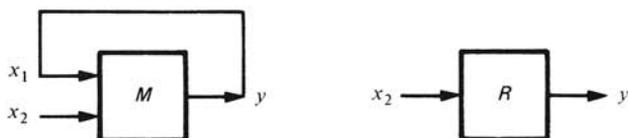


Fig. 2.19

Considérons la séquence $x_2(1, 3) = 101$. Les séquences $y(1, 3)$ qui correspondent à x_2 par la machine composée R sont les séquences telles que $y(i) + x(i) = y(i)$, ($i = 1, 2, 3$).

On a donc $R(101) = \{101, 111\}$.

2.2.6 Permutations

Une *permutation de deux entrées* d'une machine M est illustrée par la figure 2.20. La machine résultante R est définie par

$$R(x_1 \times \dots \times x_i \times \dots \times x_j \times \dots \times x_m) = M(x_1 \times \dots \times x_j \times \dots \times x_i \times \dots \times x_m). \quad (2.11)$$

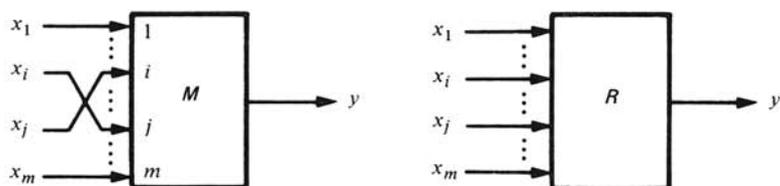


Fig. 2.20

La permutation de deux sorties d'une machine N (fig. 2.21) fournit une machine S , qui est définie par

$$y_1 \times \dots \times y_i \times \dots \times y_j \times \dots \times y_n \in S(x) \iff y_1 \times \dots \times y_j \times \dots \times y_i \times \dots \times y_n \in N(x). \quad (2.12)$$

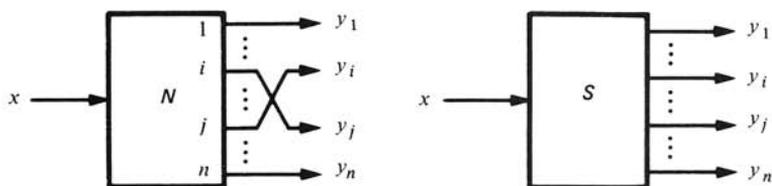


Fig. 2.21

2.2.7 Schémas de composition

Les opérations de juxtaposition, connexion, permutation introduites jusqu'ici définissent la signification d'un schéma de composition tel que celui de la figure 2.22. La machine composée R est définie par les trois opérations qui apparaissent dans la figure

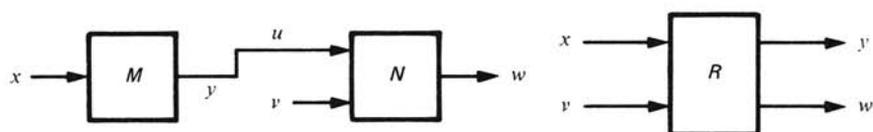


Fig. 2.22

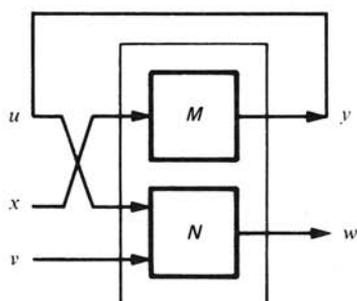


Fig. 2.23

2.23, à savoir : juxtaposition de M et N , permutation des deux premières entrées de la machine obtenue, puis connexion sortie-entrée. La permutation est nécessaire, du fait que l'opération de connexion sortie-entrée n'a été définie (§ 2.2.4) que pour la première entrée d'une machine.

2.2.8 Machines sans entrée

L'opération de connexion sortie-entrée (§ 2.2.4) sur une machine M n'a été définie que si le nombre d'entrées de M est un nombre $m > 1$. On peut étendre cette opération au cas $m = 1$ en introduisant la notion de *machine sans entrée*.

Une machine R sans entrée, et à n variables de sortie $y_1 \in Y_1^+, \dots, y_n \in Y_n^+$ (fig. 2.24) est simplement un sous-ensemble

$$R \subset (Y_1 \times \dots \times Y_n)^+.$$

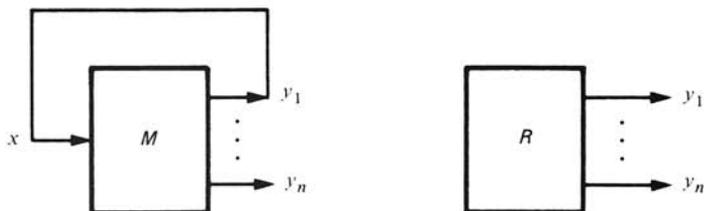


Fig. 2.24

Une telle machine peut être considérée comme un générateur de séquences.

L'opération de connexion illustrée par la figure 2.24 suppose que l'alphabet d'entrée X de M et son alphabet de sortie Y_1 sont identiques. La machine composée R n'a pas d'entrée. Elle est définie par

$$y_1 \times \dots \times y_n \in R \iff y_1 \times \dots \times y_n \in M(y_1). \quad (2.13)$$

2.2.9 Connexion divergente

L'opération de *connexion divergente* (ou *connexion entrée-entrée*) est illustrée par la figure 2.25. Elle porte sur une machine

$$M : (X_1 \times \dots \times X_m)^+ \longrightarrow Y^+$$

telle que $m > 1$, et vérifiant la *condition* $X_1 = X_2$.

La machine composée est la machine à $m - 1$ entrées

$$R : (X_2 \times \dots \times X_m)^+ \longrightarrow Y^+$$

telle que pour toute séquence $x_2 \times \dots \times x_m \in (X_2 \times \dots \times X_m)^+$ on a

$$R(x_2 \times \dots \times x_m) = M(x_2 \times x_2 \times \dots \times x_m). \quad (2.14)$$

L'opération de connexion divergente n'est définie ci-dessus que pour la première et la seconde entrée d'une machine M . Elle s'étend immédiatement à deux entrées quelconques, en effectuant des permutations (§ 2.2.6) qui permettent de se ramener au cas précédent.

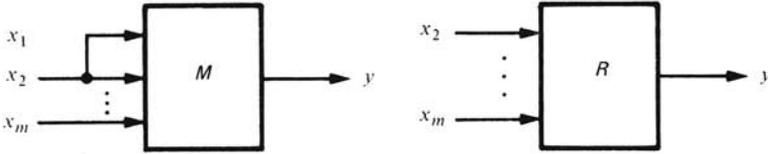


Fig. 2.25

2.2.10 Projection

L'opération de *projection* consiste à "ignorer" une sortie d'une machine. Elle est décrite par la figure 2.26, où

$$M : X^+ \longrightarrow (Y_1 \times \dots \times Y_n)^+$$

est une machine à n sorties ($n > 1$). La sortie ignorée est la première sortie, ce qui est indiqué par la mise entre parenthèses $[\]$ de la variable de sortie y_1 . La machine composée est la machine

$$R : X^+ \longrightarrow (Y_2 \times \dots \times Y_n)^+$$

définie par la propriété suivante :

$$y_2 \times \dots \times y_n \in R(x) \iff \exists y_1 : y_1 \times y_2 \times \dots \times y_n \in M(x). \quad (2.15)$$

En français : une séquence $y_2 \times \dots \times y_n$ correspond à une séquence x par R si et seulement s'il *existe* une séquence y_1 telle que $y_1 \times y_2 \times \dots \times y_n$ corresponde à x par M .

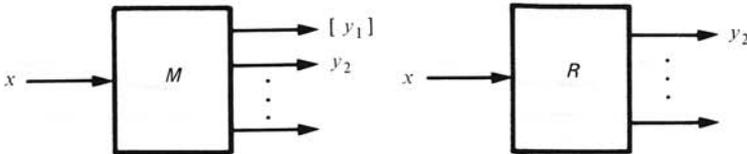


Fig. 2.26

Le choix du terme de projection pour désigner cette opération est commenté au paragraphe 2.2.12.

En général, l'opération de projection est utilisée conjointement avec une opération de connexion (§ 2.2.11).

2.2.11 Exemple

Le schéma de composition de la figure 2.27 représente la succession d'opérations suivante :

- juxtaposition de M et N (fig. 2.28), R_1 étant la résultante de cette opération; la relation $y \times z \in R_1(x \times u)$ équivaut à

$$y \in M(x) \quad \text{et} \quad z \in N(u);$$

- connexion sortie-entrée (fig. 2.29), R_2 étant la résultante; la relation $y \times z \in R_2(x)$ équivaut à $y \times z \in R_1(x \times y)$, donc à

$$y \in M(x) \quad \text{et} \quad z \in N(y);$$

- projection (fig. 2.30), R étant la résultante; la relation $z \in R(x)$ équivaut à $\exists y : y \times z \in R_2(x)$, donc à $\exists y : y \in M(x)$ et $z \in N(y)$.

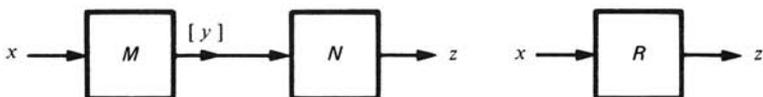


Fig. 2.27

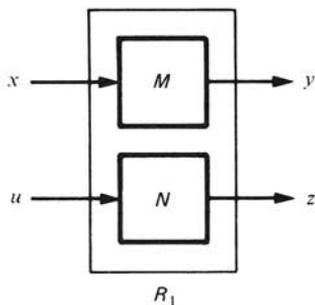


Fig. 2.28

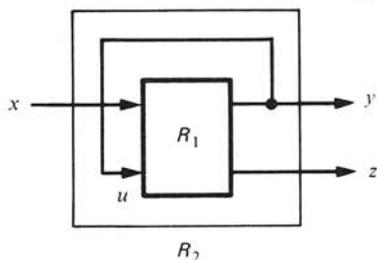


Fig. 2.29

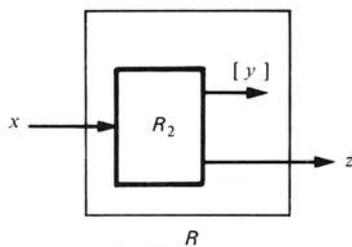


Fig. 2.30

2.2.12 Commentaire

Le choix du terme de projection (§ 2.2.10) découle des réflexions suivantes. Considérons l'opération de projection appliquée à une machine M (fig. 2.31) de format $(1, 2)$

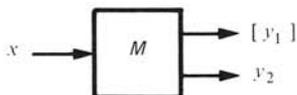


Fig. 2.31

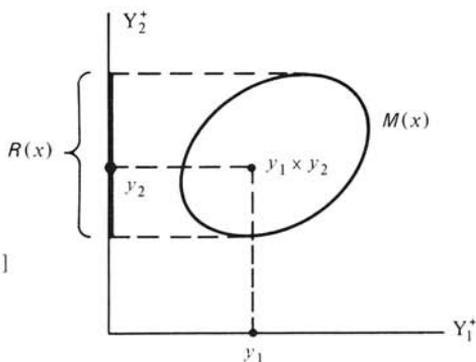


Fig. 2.32

Pour une séquence d'entrée x , l'ensemble $M(x)$ est un ensemble de séquences de la forme $y_1 \times y_2$. Si nous imaginons chaque séquence y_1 sur l'alphabet Y_1 représentée par un point d'une droite Y_1^+ (fig. 2.32), et de même pour chaque séquence y_2 sur l'alphabet Y_2 , alors l'ensemble $M(x)$ est représenté par un ensemble de points du plan. R étant la machine composée définie par (2.15), l'ensemble $R(x)$ apparaît comme la projection de l'ensemble $M(x)$ sur la droite Y_2^+ . En effet, un point y_2 appartient à cette projection si et seulement s'il existe un point $y_1 \in Y_1^+$ tel que $y_1 \times y_2 \in M(x)$.

2.2.13 Schémas de composition

Un schéma de composition représente une série d'opérations de composition fondamentales (juxtaposition, connexion, projection, permutation). En général, l'ordre d'exécution de ces opérations n'est pas précisé, de sorte que la machine composée n'est définie qu'à des permutations près. En outre, on n'associe en général qu'une seule variable à un ensemble d'entrées et/ou de sorties reliées entre elles par des connexions.

Les relations qui caractérisent la machine composée peuvent s'obtenir en décomposant le schéma en ses opérations élémentaires comme nous l'avons vu par exemple au paragraphe 2.2.11. Il est cependant aisé de les écrire directement.

Considérons par exemple le schéma de composition de la figure 2.33. La machine composée R est définie comme suit : une séquence $y \times z$ correspond à une séquence $x \times u$ par R si et seulement s'il existe une séquence w telle que les relations

$$y \times w \in M(x \times u) \quad ; \quad z \in N(u \times w)$$

soient vérifiées.

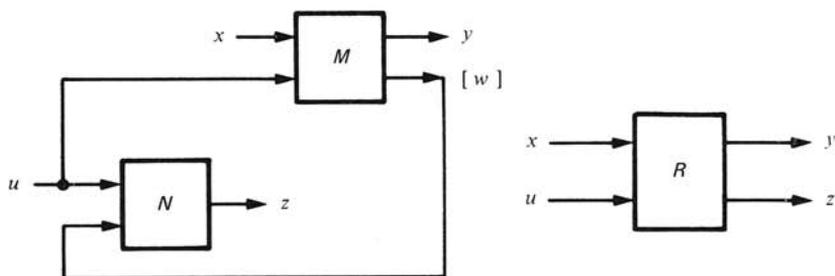


Fig. 2.33

2.2.14 Exercice

Détailler la succession d'opérations de composition fondamentales définie par la figure 2.33, et définir à chaque pas la machine composée (cf. § 2.2.11).

2.3 MACHINES COMBINATOIRES

2.3.1 Introduction

La machine OU étudiée au paragraphe 2.1.8 est un exemple de machine combinatoire binaire. On définit ci-dessous la notion générale de machine combinatoire, les alphabets d'entrée et de sortie pouvant être quelconques. Ces machines n'interviendront par la suite qu'associées à des machines séquentielles.

2.3.2 Définition

Une machine $M : X^+ \rightarrow Y^+$ (alphabet d'entrée X , alphabet de sortie Y) est dite *combinatoire* si elle possède les propriétés suivantes :

- à toute séquence d'entrée x correspond au moins une séquence de sortie y ;
pour toute séquence $x \in X^+$, on a donc

$$M(x) \neq \emptyset; \quad (2.16)$$

- pour deux séquences d'entrée x, x' quelconques, on a

$$M(x x') = M(x) M(x'); \quad (2.17)$$

le produit $M(x) M(x')$ est le produit de deux ensembles de séquences (§ 1.4.5).

Comme *conséquence* de la deuxième propriété ci-dessus, on peut énoncer : si $x = x(1) x(2) \dots x(n)$ est une séquence de longueur $n > 1$, on a

$$M(x) = M(x(1)) M(x(2)) \dots M(x(n)). \quad (2.18)$$

En vertu de cette dernière propriété, la correspondance M est entièrement déterminée lorsqu'on connaît l'ensemble $M(x)$ pour toute séquence x de longueur 1. Pour cette raison, une machine combinatoire est généralement donnée sous la forme d'une correspondance $X \rightarrow Y$.

2.3.3 Exemple

Considérons les alphabets $X = \{a, b, c, d\}$ et $Y = \{0, 1, 2\}$. Le tableau 2.34 définit une correspondance $X \rightarrow Y$. La machine combinatoire $M : X^+ \rightarrow Y^+$ définie par cette correspondance est déterminée par la règle (2.17). On a par exemple :

$$M(ab) = M(a) M(b) = 0 \{0, 2\} = \{00, 02\}$$

$$M(abc) = M(ab) M(c) = \{00, 02\} 1 = \{001, 021\}.$$

x	$M(x)$
a	$\{0\}$
b	$\{0, 2\}$
c	$\{1\}$
d	$\{1, 2\}$

Tableau 2.34

2.3.4 Commentaire

La définition d'une machine combinatoire n'a été donnée au paragraphe 2.3.2 que pour une machine à une entrée et une sortie. Elle se généralise immédiatement pour une machine à m entrées et n sorties. Il suffit de remplacer X par le produit cartésien $X_1 \times \dots \times X_m$ des m alphabets d'entrée, et Y par $Y_1 \times \dots \times Y_n$ (§ 2.2.2).

2.3.5 Exemple

La figure 2.35 définit une machine combinatoire M à une entrée, d'alphabet $X = \{0, 1, 2\}$, et deux sorties, d'alphabets $Y_1 = Y_2 = \{0, 1\}$. On a par exemple :

$$M(210) = M(2)M(1)M(0) = \{1 \times 1\} \{0 \times 1, 1 \times 0\} \{0 \times 0\} \\ = \{(1 \times 1)(0 \times 1)(0 \times 0), (1 \times 1)(1 \times 0)(0 \times 0)\}.$$

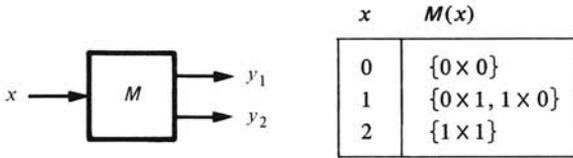


Fig. 2.35

2.3.6 Définition

Une machine combinatoire $M : X^+ \rightarrow Y^+$ est dite *complètement spécifiée* lorsque $|M(x)| = 1$ pour toute séquence $x \in X^+$. Cela signifie que la correspondance M est fonctionnelle (§ 1.3.11).

2.3.7 Machines combinatoires composées

À l'exception de l'opération de connexion sortie-entrée (§ 2.2.4) qui demande une précaution, toutes les opérations de composition définies dans la section 2.2 conservent le caractère combinatoire des machines. Ainsi, la juxtaposition (§ 2.2.3) de deux machines combinatoires est une machine combinatoire. Les opérations de permutation (§ 2.2.6), connexion divergente (§ 2.2.9), projection (§ 2.2.10), portant sur une machine combinatoire, fournissent une machine combinatoire.

Si M est une machine combinatoire, l'opération de connexion sortie-entrée de la figure 2.18 ne produit pas nécessairement une machine combinatoire. On peut montrer plus exactement que la machine composée R possède la propriété (2.17), mais pas nécessairement la propriété (2.16).

Par contre, si la machine combinatoire M est la juxtaposition de deux machines combinatoires M_1 et M_2 (fig. 2.36) et si la connexion relie une sortie de M_1 à une entrée de M_2 (ou vice-versa), alors la machine composée est toujours combinatoire.

La démonstration de ces assertions repose sur la définition formelle des opérations de composition (sect. 2.2). Elle est laissée au lecteur.

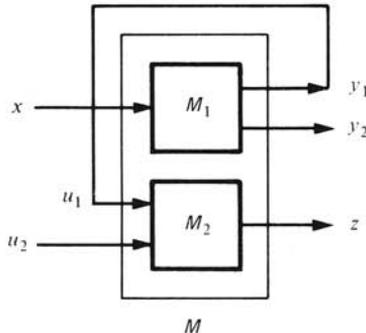


Fig. 2.36

2.3.8 Définition

Les sorties d'une machine M à deux sorties (fig. 2.37) sont dites *indépendantes*, si M peut être définie par la composition de deux machines M_1 et M_2 selon le schéma de la figure 2.37. Cela revient à dire que pour toute séquence d'entrée x , l'ensemble $M(x)$ est un produit cartésien $M_1(x) \times M_2(x)$ avec $M_1(x) \subset Y_1^+$ et $M_2(x) \subset Y_2^+$ (Y_1 et Y_2 étant les alphabets de sortie de M).

La définition se généralise de façon évidente pour une machine à n sorties ($n > 2$).

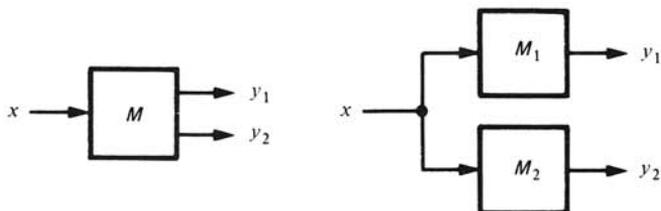


Fig. 2.37

2.3.9 Exemple

La machine combinatoire $M : X^+ \rightarrow (Y_1 \times Y_2)^+$ ($X = \{0, 1, 2\}$, $Y_1 = Y_2 = \mathbf{B}$) définie par le tableau 2.38 est à sorties indépendantes. En effet, pour $x = 0, 1, 2$, l'ensemble $M(x)$ est le produit cartésien d'un ensemble $M_1(x)$ par un ensemble $M_2(x)$. Par suite, pour une séquence $x = x(1)x(2)\dots$ de longueur $n > 1$, on a

$$\begin{aligned}
 M(x) &= M(x(1)) M(x(2)) \dots && \text{par (2.18)} \\
 &= [M_1(x(1)) \times M_2(x(1))] [M_1(x(2)) \times M_2(x(2))] \dots \\
 &= M_1(x(1)) M_1(x(2)) \dots \times M_2(x(1)) M_2(x(2)) \dots && \text{par (1.64)} \\
 &= M_1(x) \times M_2(x). && \text{par (2.18)}.
 \end{aligned}$$

x	$M(x)$	$M_1(x)$	$M_2(x)$
0	0×0	0	0
1	$\{0 \times 0, 0 \times 1\}$	0	$\{0, 1\}$
2	1×1	1	1

Tableau 2.38

2.3.10 Exemple

Les sorties de la machine combinatoire M du paragraphe 2.3.5 ne sont pas indépendantes, car l'ensemble $M(1) = \{0 \times 1, 1 \times 0\}$ n'est pas un produit cartésien. La dépendance des deux sorties peut s'exprimer en disant que la relation $y_1 \times y_2 \in M(1)$ équivaut à $y_1 = \bar{y}_2$.

2.3.11 Exercice

Soit M une machine combinatoire binaire à deux entrées x_1, x_2 et une sortie y , telle que $M(1 \times 0) = 0$, $M(0 \times 0) = M(0 \times 1) = M(1 \times 1) = 1$. Considérons la machine R définie par la connexion sortie-entrée $y \rightarrow x_1$ (fig. 2.19). Montrer que R ne possède

pas la propriété (2.16), c'est-à-dire qu'il existe une séquence x_2 telle que $R(x_2) = \emptyset$. Montrer que R possède la propriété (2.17), c'est-à-dire que $R(x_2 x'_2) = R(x_2) R(x'_2)$ quelles que soient les séquences x_2, x'_2 .

2.4 MACHINES SÉQUENTIELLES

2.4.1 Introduction

Les machines JK (§ 2.1.9) et $\bar{s}\bar{r}$ (§ 2.1.10) sont des exemples de machines séquentielles. La définition générale de cette classe de machines est donnée dans la présente section. Elle est précédée d'un exemple destiné à faciliter la lecture de la définition formelle.

2.4.2 Exemple

Considérons les alphabets $X = \{0, 1\}$ et $Y = \{a, b, c, d\}$. Nous allons définir une machine $M : X^+ \rightarrow Y^+$ au moyen du tableau 2.39. Le procédé de définition caractérise la classe des machines séquentielles.

		X	
		0	1
Y	a	b	a, c
	b	b	d
	c	c, d	d
	d	a	a, b, c, d

Tableau 2.39

Le tableau 2.39 associe à chaque couple $y \times x \in Y \times X$ un ensemble $f(y \times x) \subset Y$, à savoir l'ensemble des éléments $y' \in Y$ qui figurent dans la ligne y et la colonne x . En d'autres termes, le tableau représente une correspondance $f : Y \times X \rightarrow Y$. On a par exemple $f(c \times 0) = \{c, d\}$, et $f(c \times 1) = \{d\}$.

Nous utiliserons la notation $y \cdot x$ pour désigner l'ensemble $f(y \times x)$. Par exemple : $c \cdot 0 = \{c, d\}$, $c \cdot 1 = \{d\}$.

Soit $x = x(1) \dots x(n)$ une séquence d'entrée de longueur n . On définit les séquences de sortie $y(1, n)$ correspondant à x comme étant les séquences que l'on peut construire en appliquant les règles suivantes :

- $y(1)$ peut être choisi arbitrairement dans Y ,
- ayant choisi $y(1), \dots, y(k)$ ($k < n$), on peut choisir $y(k+1)$ dans l'ensemble $y(k) \cdot x(k)$.

Considérons par exemple la séquence d'entrée $x = x(1,4) = 0110$. La séquence de sortie $y(1,4) = ccdb$ correspond à x , car elle peut être construite en respectant les règles ci-dessus. On a en effet :

$$\begin{aligned}
 y(1) &\in Y \\
 y(2) &\in y(1) \cdot x(1) = c \cdot 0 = \{c, d\} \\
 y(3) &\in y(2) \cdot x(2) = c \cdot 1 = \{d\} \\
 y(4) &\in y(3) \cdot x(3) = d \cdot 1 = \{a, b, c, d\}.
 \end{aligned}$$

L'ensemble $M(0110)$ est l'ensemble de toutes les séquences $y(1,4)$ que l'on peut construire de cette façon. Ce procédé définit $M(x)$ pour toute séquence d'entrée x , et définit donc la machine $M : X^+ \rightarrow Y^+$. Pour une séquence x de longueur 1, on a $M(x) = Y$, puisque $y(1)$ peut être choisi arbitrairement.

2.4.3 Exercice

Dénombrer les séquences de sortie y qui correspondent à la séquence d'entrée $x = 0110$ par la machine M (§ 2.4.2).

2.4.4 Définition

Soient X et Y des alphabets quelconques. Une machine $M : X^+ \rightarrow Y^+$ est dite *séquentielle* s'il existe une correspondance $f : Y \times X \rightarrow Y$ avec les propriétés suivantes, où pour chaque couple $y \times x \in Y \times X$, l'ensemble $f(y \times x) \subset Y$ est noté $y \cdot x$.

- Pour chaque couple $y \times x \in Y \times X$,

$$y \cdot x \neq \emptyset. \quad (2.19)$$

- $x \in X \implies M(x) = Y$. (2.20)

- Pour deux séquences quelconques $x(1, n) \in X^+$, $y(1, n) \in Y^+$ de longueur $n > 1$,

$$y \in M(x) \iff \begin{cases} y(k+1) \in y(k) \cdot x(k) \\ \text{pour } k = 1, \dots, n-1 \end{cases} \quad (2.21)$$

On montre aisément que la correspondance f est unique. Elle est appelée *correspondance de transition* de M . La table qui représente cette correspondance (exemple : tab. 2.39) est appelée *table de transition* de M .

La définition ci-dessus s'étend immédiatement à une machine M à m entrées (alphabets d'entrée X_1, \dots, X_m) et n sorties (alphabets de sortie Y_1, \dots, Y_n). Il suffit de remplacer X par $X_1 \times \dots \times X_m$ et Y par $Y_1 \times \dots \times Y_n$.

2.4.5 Graphe de transition d'une machine séquentielle

Le *graphe de transition* G (§ 1.5.1) d'une machine séquentielle $M : X^+ \rightarrow Y^+$ est par définition l'ensemble des triplets $y \times x \times y' \in Y \times X \times Y$ tels que $y' \in y \cdot x$. On a donc

$$y' \in y \cdot x \iff y \times x \times y' \in G. \quad (2.22)$$

2.4.6 Exemple

La figure 2.40 représente le graphe de transition G de la machine séquentielle M du paragraphe 2.4.2, dont la table de transition est le tableau 2.39. On voit par exemple que la relation $a \cdot 1 = \{a, c\}$ se traduit par les arêtes $a \times 1 \times a$ et $a \times 1 \times c$ du graphe.

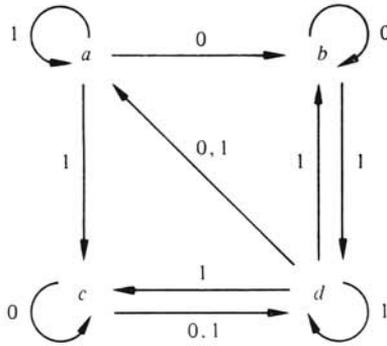


Fig. 2.40

2.4.7 Remarque

Une machine séquentielle $M : X^+ \rightarrow Y^+$ peut être donnée aussi bien au moyen de sa table de transition que de son graphe de transition. Par exemple, le graphe de la figure 2.40 permet de retrouver la table de transition (tab. 2.39) en appliquant (2.22). L'ensemble $y \cdot x$ est l'ensemble des y' tels que $y \times x \times y' \in G$.

2.4.8 Proposition

Soient G le graphe d'une machine séquentielle $M : X^+ \rightarrow Y^+$, et $x(1, n) \in X^+$, $y(1, n) \in Y^+$ deux séquences de longueur $n > 1$. On a $y \in M(x)$ si et seulement s'il existe un chemin

$$y(1) \xrightarrow{x(1)} y(2) \xrightarrow{x(2)} \dots \xrightarrow{x(n-1)} y(n)$$

dans G . En effet, la relation (2.21) peut s'écrire en vertu de (2.22) :

$$y \in M(x) \iff \begin{cases} y(k) \times x(k) \times y(k+1) \in G \\ \text{pour } k = 1, \dots, n-1 \end{cases} \quad (2.23)$$

2.4.9 Exemple

Le graphe de transition G de la figure 2.40 admet le chemin

$$c \xrightarrow{0} c \xrightarrow{1} d \xrightarrow{1} b.$$

On a donc $ccdb \in M(011x)$ quel que soit $x \in \{0, 1\}$, M étant la machine du paragraphe 2.4.2.

2.4.10 Remarque

Si $M : X^+ \rightarrow Y^+$ est une machine séquentielle, et $x(1, n)$ une séquence d'entrée de M , l'ensemble $M(x)$ ne dépend pas de $x(n)$.

2.4.11 Remarque

Les machines JK (§ 2.1.9) et $\bar{s}\bar{r}$ (§ 2.1.10) ont été définies chacune au moyen d'un graphe G , par une relation de la forme (2.23). On voit donc que ce sont des ma-

chines séquentielles. Par exemple, le graphe de la figure 2.14 est le graphe de transition de la machine $\bar{s}\bar{r}$. On en tire la table de transition de cette machine (tab. 2.41).

	0×0	0×1	1×1	1×0
0	0	0	0, 1	1
1	1	0	0, 1	1

Tableau 2.41

2.4.12 Proposition

Soient :

- $M : X^+ \rightarrow Y^+$, une machine séquentielle
- $x(1, m), x'(1, n)$, deux séquences sur X
- $y(1, m), y'(1, n)$, deux séquences sur Y .

On a :

$$yy' \in M(xx') \iff \begin{cases} y \in M(x) \\ y'(1) \in y(m) \cdot x(m) \\ y' \in M(x'). \end{cases} \quad (2.24)$$

2.4.13 Preuve

Soit G le graphe de transition de M . Supposons que $yy' \in M(xx')$. Il existe dans G un chemin

$$y(1) \xrightarrow{x(1)} \dots \xrightarrow{x(m-1)} y(m) \xrightarrow{x(m)} y'(1) \xrightarrow{x'(1)} \dots \xrightarrow{x'(n-1)} y'(n) \quad (2.25)$$

Ce chemin se décompose en trois chemins

$$\left. \begin{array}{l} y(1) \xrightarrow{x(1)} \dots \xrightarrow{x(m-1)} y(m) \\ y(m) \xrightarrow{x(m)} y'(1) \\ y'(1) \xrightarrow{x'(1)} \dots \xrightarrow{x'(n-1)} y'(n) \end{array} \right\} \quad (2.26)$$

qui prouvent les trois relations $y \in M(x)$, $y'(1) \in y(m) \cdot x(m)$, $y' \in M(x')$. Supposons réciproquement que ces trois relations soient vraies. Il existe alors dans G trois chemins de la forme (2.26). Le produit de ces trois chemins est un chemin de la forme (2.25) prouvant que $yy' \in M(xx')$.

2.4.14 Définition

Soient $M : X^+ \rightarrow Y^+$ une machine séquentielle, P un sous-ensemble non vide de Y ($P \subset Y$), et $x \in X$ une séquence d'entrée de longueur 1. On définit l'ensemble $P \cdot x \subset Y$ par la formule

$$P \cdot x = \bigcup_{y \in P} y \cdot x. \quad (2.27)$$

Pour $P = \emptyset$, on pose

$$\emptyset \cdot x = \emptyset. \quad (2.28)$$

L'ensemble $P \cdot x$ est appelé le *transformé de P par x* selon M . Lorsque plusieurs machines M, M', \dots sont considérées simultanément, on peut écrire $(P \cdot x)_M, (P \cdot x)_{M'}, \dots$ pour éviter des confusions.

2.4.15 Exemple

Soient M la machine du paragraphe 2.4.2, $P = \{b, c\} \subset Y$, $x = 0$. On a (tab. 2.39):

$$P \cdot x = (b \cdot x) \cup (c \cdot x) = \{b\} \cup \{c, d\} = \{b, c, d\}.$$

2.4.16 Commentaire

Si $f: Y \times X \rightarrow Y$ est la correspondance de transition (§ 2.4.4) d'une machine séquentielle, $M: X^+ \rightarrow Y^+$, la définition de $P \cdot x$ (2.27), (2.28) revient à dire que $P \cdot x = f(P \times x)$ au sens du paragraphe 1.3.3.

2.4.17 Définition

Soient $M: X^+ \rightarrow Y^+$ une machine séquentielle, $P \subset Y$, et $x = x(1, n)$ une séquence d'entrée de longueur $n > 1$. On définit l'ensemble $P \cdot x \subset Y$ comme étant l'ensemble P_n obtenu par la construction suivante :

$$\left. \begin{array}{l} P \cdot x(1) = P_1 \\ P_1 \cdot x(2) = P_2 \\ \vdots \\ P_{n-1} \cdot x(n) = P_n \end{array} \right\} \text{ (selon § 2.4.14)} \quad (2.29)$$

Si l'ensemble P possède un seul élément $p \in Y$, $P \cdot x$ est noté $p \cdot x$.

2.4.18 Exemple

Soient M la machine du paragraphe 2.4.2, $P = \{b, d\}$, $x = x(1, 5) = 00101$. On a (tab. 2.39) :

$$\begin{aligned} P \cdot x(1) &= \{b, d\} \cdot 0 = \{b, a\} = P_1 \\ P_1 \cdot x(2) &= \{b, a\} \cdot 0 = \{b\} = P_2 \\ P_2 \cdot x(3) &= b \cdot 1 = d = P_3 \\ P_3 \cdot x(4) &= d \cdot 0 = a = P_4 \\ P_4 \cdot x(5) &= a \cdot 1 = \{a, c\} = P_5 = P \cdot x. \end{aligned}$$

On peut donc écrire pour cette machine :

$$\{b, d\} \cdot 00101 = \{a, c\}.$$

2.4.19 Propriétés

Soient $M : X^+ \rightarrow Y^+$ une machine séquentielle, P et Q des sous-ensembles de Y , x et x' des séquences sur X . On a

$$P \cdot (xx') = (P \cdot x) \cdot x' \quad (2.30)$$

$$P \cdot x = \bigcup_{p \in P} p \cdot x \quad (2.31)$$

$$\emptyset \cdot x = \emptyset \quad (2.32)$$

$$P \subset Q \Rightarrow P \cdot x \subset Q \cdot x \quad (2.33)$$

$$(P \cup Q) \cdot x = (P \cdot x) \cup (Q \cdot x) \quad (2.34)$$

$$(P \cap Q) \cdot x \subset (P \cdot x) \cap (Q \cdot x) \quad (2.35)$$

2.4.20 Preuve

Supposons que dans la construction (2.29) k soit un entier tel que $1 \leq k < n$. On a $P \cdot x(1, k) = P_k$ et $P_k \cdot x(k+1, n) = P_n$, donc $P \cdot x(1, n) = (P \cdot x(1, k)) \cdot x(k+1, n)$. Ceci prouve (2.30).

Les relations (2.31) et (2.32) sont vraies pour une séquence x de longueur 1 (§ 2.4.14). Par (2.29) elles s'étendent de proche en proche à une séquence de longueur quelconque.

La correspondance $f : Y \times X^+ \rightarrow Y$ qui associe à chaque couple $p \times x \in Y \times X^+$ l'ensemble $f(p \times x) = p \cdot x \subset Y$ est une extension de la correspondance de transition de la machine. Les relations (2.31) et (2.32) montrent que l'on a $P \cdot x = f(P \times x)$ au sens du paragraphe 1.3.3. Les relations (2.33) à (2.35) découlent du paragraphe 1.3.5.

2.4.21 Exercice

Pour la machine M du paragraphe 2.4.2, $P = \{b, d\}$, $x = x(1, 5) = 00101$, vérifier la formule (2.31), c'est-à-dire vérifier que $P \cdot x = b \cdot x \cup d \cdot x$. On a $P \cdot x = \{a, c\}$ (§ 2.4.18).

2.4.22 Proposition

Soient $M : X^+ \rightarrow Y^+$ une machine séquentielle, et $x(1, n) \in X^+$, $y(1, n) \in Y^+$ deux séquences de longueur $n > 1$. On a

$$y \in M(x) \Rightarrow \begin{cases} y(k+1) \in y(1) \cdot x(1, k) \\ \text{pour } k=1, \dots, n-1. \end{cases} \quad (2.36)$$

La notation $x(1, k)$ représente naturellement la séquence $x(1) \dots x(k)$.

2.4.23 Démonstration

On suppose que $y \in M(x)$, et l'on montre par récurrence que la relation

$$y(k+1) \in y(1) \cdot x(1, k) \quad (2.37)$$

est vraie pour $k=1, \dots, n-1$. Pour $k=1$, elle s'écrit $y(2) \in y(1) \cdot x(1)$, ce qui est vrai par hypothèse, en vertu de (2.21). Supposons que (2.37) soit vraie pour un $k < n-1$.

Posons $P_k = y(1) \cdot x(1, k)$. On a donc $y(k+1) \in P_k$. En vertu de (2.21), $y(k+2) \in y(k+1) \cdot x(k+1)$. Par (2.27), $y(k+2) \in P_k \cdot x(k+1)$. Or, $P_k \cdot x(k+1) = (y(1) \cdot x(1, k)) \cdot x(k+1) = y(1) \cdot x(1, k+1)$ par (2.30). Donc $y(k+2) \in y(1) \cdot x(1, k+1)$, et la relation (2.37) est vraie pour $k+1$.

2.4.24 Exemple

Pour la machine M du paragraphe 2.4.2, on a $ccdb \in M(0110)$. En vertu de (2.36), cette relation implique

$$\begin{aligned} y(2) \in y(1) \cdot x(1) & \text{ soit } c \in c \cdot 0 \\ y(3) \in y(1) \cdot x(1, 2) & \text{ soit } d \in c \cdot 01 \\ y(4) \in y(1) \cdot x(1, 3) & \text{ soit } b \in c \cdot 011. \end{aligned}$$

Ces trois relations sont vraies, car $c \cdot 0 = \{c, d\}$, $c \cdot 01 = c \cdot 011 = \{a, b, c, d\}$.

2.4.25 Commentaire

La réciproque de (2.36) est fautive en général. On ne peut pas inverser le sens de l'implication, sauf si M est une machine complètement spécifiée (sect. 2.5). Ceci est illustré par l'exercice suivant.

2.4.26 Exercice

Pour la machine M du paragraphe 2.4.2, montrer que les séquences $x(1, 3) = 110$ et $y(1, 3) = aca$ vérifient le second membre de (2.36) mais non le premier : $y \notin M(x)$.

2.4.27 Conditions initiales

Considérons une machine séquentielle $M : X^+ \rightarrow Y^+$, et un sous-ensemble non vide $P \subset Y$. Pour toute séquence $x(1, n) \in X^+$, on désigne par $M_P(x)$ l'ensemble des séquences $y(1, n) \in M(x)$ telles que $y(1) \in P$. On a donc

$$y \in M_P(x) \iff y \in M(x) \text{ et } y(1) \in P. \quad (2.38)$$

On définit ainsi une *machine* $M_P : X^+ \rightarrow Y^+$. Si $P = Y$, les machines M_P et M sont identiques. Si $P \neq Y$, la machine M_P ne vérifie pas (2.20); elle n'est plus une machine séquentielle au sens strict du paragraphe 2.4.4. On dit que M_P est définie en imposant à M la *condition initiale* $y(1) \in P$.

Lorsque P est un sous-ensemble de Y à un seul élément, $P = \{p\}$, on écrit M_p au lieu de $M_{\{p\}}$.

2.4.28 Exemple

Soit M la machine séquentielle du paragraphe 2.4.2, dont le graphe de transition G est représenté par la figure 2.40. Considérons une séquence d'entrée $x(1, 4) = 110u$, où $u \in \{0, 1\}$ est quelconque. La figure 2.42 représente tous les chemins d'origine a et d'étiquette $x(1, 3) = 110$ de G . Les séquences des sommets de ces chemins constituent l'ensemble

$$M_a(110u) = \{aaab, aacc, aacd, acda\}. \quad (2.39)$$

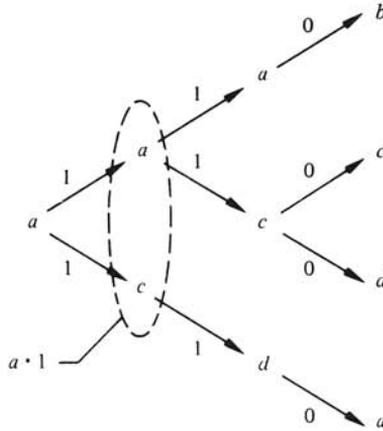


Fig. 2.42

2.4.29 Proposition

Soient $M : X^+ \rightarrow Y^+$ une machine séquentielle, $p \in Y$, x une séquence d'entrée de longueur 1 ($x \in X$), et x' une séquence d'entrée de longueur quelconque ($x' \in X^+$). On a :

$$M_p(x) = p \quad (2.40)$$

$$M_p(xx') = M_p(x) M_{p \cdot x}(x'). \quad (2.41)$$

La première formule est évidente. La seconde est démontrée au paragraphe 2.4.31.

2.4.30 Exemple

La formule (2.41) peut être illustrée par la figure 2.42, relative à la machine M du paragraphe 2.4.2. On peut mettre (2.39) sous la forme

$$M_a(110u) = a \{aab, acc, acd, cda\}.$$

On voit, dans la figure 2.42, que l'ensemble $\{aab, acc, acd, cda\}$ est l'ensemble $M_p(10u)$ où $P = \{a, c\} = a \cdot 1$. D'autre part, $M_a(1) = a$. On a donc

$$M_a(110u) = M_a(1) M_{a \cdot 1}(10u).$$

La formule (2.41) est vérifiée ici, pour $p = a$, $x = 1$, $x' = 10u$.

2.4.31 Démonstration

La formule (2.41) peut être démontrée comme suit. Soient $y \in Y$, et $y' \in Y^+$ telle que $lg(y') = lg(x')$. Supposons que $yy' \in M_p(xx')$. Alors, par (2.38) et (2.24), il vient

$$y = p, \quad y' \in M(x'), \quad y'(1) \in p \cdot x$$

d'où

$$y \in M_p(x) \quad \text{et} \quad y' \in M_{p \cdot x}(x') \quad \text{par (2.40) et (2.38).}$$

Par conséquent (§ 1.4.5), on a $yy' \in M_p(x) M_{p \cdot x}(x')$.

Partant de cette dernière relation, on montre en suivant le cheminement logique inverse, qu'elle implique $yy' \in M_p(xx')$.

2.4.32 Exercice

Montrer par un exemple que la relation (2.41) peut être fausse si $lg(x) \neq 1$.

2.4.33 Proposition

Soit $M : X^+ \rightarrow Y^+$ une machine séquentielle. Pour tout sous-ensemble non vide $P \subset Y$, et pour toute séquence $x \in X^+$, on a

$$M_P(x) = \bigcup_{p \in P} M_p(x). \quad (2.42)$$

La proposition est évidente (2.38).

2.4.34 Définition

Nous dirons qu'une machine séquentielle $M : X^+ \rightarrow Y^+$ est de type *standard*, si pour chaque couple $y \times x \in Y \times X$ on a

$$y \cdot x \in Y \quad \text{ou} \quad y \cdot x = Y. \quad (2.43)$$

Cela signifie que la table de transition de M contient dans chaque case : ou bien un seul élément de Y , ou bien tous les éléments de Y . Dans ce dernier cas, on utilise généralement la *notation standard* du tiret “-” qui représente l'ensemble $y \cdot x = Y$.

Le graphe de transition d'une machine séquentielle de type standard est caractérisé par la propriété suivante. Pour chaque couple $y \times x \in Y \times X$ le graphe possède ou bien une seule arête d'origine y et d'étiquette x , ou bien une arête $y \times x \times y'$ pour chaque $y' \in Y$. Dans ce dernier cas, la *notation standard* du graphe consiste à omettre de représenter les arêtes d'origine y et d'étiquette x .

2.4.35 Exemples

Soient $X = \{0, 1\}$ et $Y = \{a, b, c, d\}$. La figure 2.43 représente la table de transition et le graphe de transition, en notation standard, d'une machine séquentielle $M : X^+ \rightarrow Y^+$ de type standard. Chaque tiret de la table représente l'ensemble $\{a, b, c, d\}$. Douze arêtes sont omises dans le graphe, à savoir toutes les arêtes de la forme $a \times 1 \times y$, $c \times 0 \times y$, $d \times 1 \times y$, pour $y = a, b, c, d$.

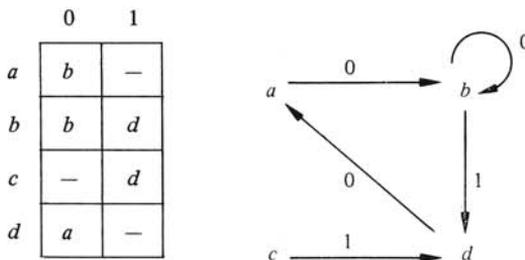


Fig. 2.43

Un autre exemple de machine séquentielle de type standard est donné par la machine $\bar{s}\bar{f}$ (tableau 2.41).

2.4.36 Remarques

Si $M : X^+ \rightarrow Y^+$ est une machine séquentielle de type standard, la relation (2.43) est vraie pour chaque couple $y \times x \in Y \times X$, mais elle peut être fausse pour un couple $y \times x \in Y \times X^+$.

Les machines séquentielles complètement spécifiées (sect. 2.5) constituent un cas particulier du type standard.

2.4.37 Exercice

Illustrer par un exemple la première remarque du paragraphe 2.4.36.

2.5 MACHINES SÉQUENTIELLES COMPLÈTEMENT SPÉCIFIÉES

2.5.1 Définition

Une machine séquentielle $M : X^+ \rightarrow Y^+$ est dite *complètement spécifiée* si pour chaque couple $y \times x \in Y \times X$, l'ensemble $y \cdot x \subset Y$ possède un seul élément. On peut formuler ceci par

$$y \cdot x \in Y \quad (\forall y \in Y, \forall x \in X) \quad (2.44)$$

l'expression $\forall y \in Y$ signifiant "quel que soit $y \in Y$ ".

Chaque case de la table de transition de M contient un seul élément. Le graphe de transition de M possède, pour chaque couple $y \times x \in Y \times X$, une seule arête d'origine y et d'étiquette x .

Pour une telle machine, et pour deux séquences $x(1, n) \in X^+$, $y(1, n) \in Y^+$, la relation (2.21) s'écrit

$$y \in M(x) \iff \begin{cases} y(k+1) = y(k) \cdot x(k) \\ \text{pour } k = 1, \dots, n-1. \end{cases} \quad (2.45)$$

La machine JK (fig. 2.10) est un exemple de machine séquentielle complètement spécifiée.

2.5.2 Proposition

Si $M : X^+ \rightarrow Y^+$ est une machine séquentielle complètement spécifiée, on a

$$y \cdot x \in Y \quad (\forall y \in Y, \forall x \in X^+). \quad (2.46)$$

Cette proposition étend la propriété (2.44) aux séquences $x \in X^+$.

2.5.3 Démonstration

La relation $y \cdot x \in Y$ est vraie pour chaque couple $y \times x \in Y \times X$, par définition (§ 2.5.1). Pour un $y \in Y$ et une séquence $x(1, n)$ de longueur $n > 1$, on a donc (§ 2.4.17) :

$$\begin{aligned} y \cdot x(1) &= y_1 \in Y \\ y \cdot x(1, 2) &= y_1 \cdot x(2) = y_2 \in Y \\ &\vdots \\ y \cdot x(1, n) &= y_{n-1} \cdot x(n) \in Y. \end{aligned}$$

2.5.4 Proposition

Soient $M : X^+ \rightarrow Y^+$ une machine séquentielle complètement spécifiée, et $x(1, n) \in X^+$, $y(1, n) \in Y^+$ deux séquences de longueur $n > 1$. On a

$$y \in M(x) \iff \begin{cases} y(k+1) = y(1) \cdot x(1, k) \\ \text{pour } k = 1, \dots, n-1. \end{cases} \quad (2.47)$$

(Comparer avec le paragraphe 2.4.22).

2.5.5 Démonstration

Le système des $n-1$ relations

$$y(k+1) = y(1) \cdot x(1, k) \quad (k = 1, \dots, n-1) \quad (2.48)$$

implique le système des $n-1$ relations

$$y(k+1) = y(k) \cdot x(k) \quad (k = 1, \dots, n-1) \quad (2.49)$$

et réciproquement. En effet, (2.48) implique

$$\begin{aligned} y(2) &= y(1) \cdot x(1) \\ &\vdots \\ y(k+1) &= y(1) \cdot x(1, k) \\ &= (y(1) \cdot x(1, k-1)) \cdot x(k) && \text{par (2.30)} \\ &= y(k) \cdot x(k) \end{aligned}$$

Réciproquement, (2.49) implique

$$\begin{aligned} y(2) &= y(1) \cdot x(1) \\ y(3) &= y(2) \cdot x(2) = (y(1) \cdot x(1)) \cdot x(2) \\ &= y(1) \cdot x(1, 2) && \text{par (2.30)} \end{aligned}$$

et ainsi de suite. On conclut en rappelant que (2.49) équivaut à $y \in M(x)$.

2.5.6 Définition

Considérons deux machines A et B de même format (fig. 2.44), ayant les mêmes alphabets d'entrée X_1, \dots, X_m et les mêmes alphabets de sortie Y_1, \dots, Y_n . Posons $X = X_1 \times \dots \times X_m$, $Y = Y_1 \times \dots \times Y_n$, et soient $M : X^+ \rightarrow Y^+$ et $R : X^+ \rightarrow Y^+$ les fonctionnements respectifs de A et B, (§ 2.1.6).

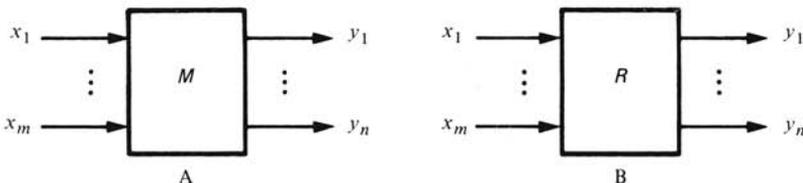


Fig. 2.44

On dit que B est une *réalisation* de A, si quelles que soient les séquences $y(1, l) \in Y^+$, $x(1, l) \in X^+$, on a

$$y \in R(x) \Rightarrow y \in M(x). \quad (2.50)$$

En français : toute séquence de sortie y qui correspond à une séquence d'entrée x par R correspond aussi à x par M .

Plus simplement, on peut écrire

$$R(x) \subset M(x) \quad (\forall x \in X^+) \quad (2.51)$$

ou encore (§ 1.3.15)

$$R \subset M. \quad (2.52)$$

2.5.7 Commentaire

Si x est une séquence d'entrée pour les machines A et B, on peut considérer toute séquence de sortie $y \in R(x)$ (respectivement $y \in M(x)$) comme une réponse de la machine B (resp. A) à la séquence x . On exprime alors sommairement la notion de réalisation, en disant que toute réponse de B est une réponse de A. Cette propriété permet de *substituer* B à A dans n'importe quel contexte, sans introduire la possibilité d'une réponse erronée.

On notera que la définition du paragraphe 2.5.6 s'applique à des machines de type quelconque (séquentielles, combinatoires, ou autres).

2.5.8 Proposition

Supposons que les machines A et B (§ 2.5.6) soient séquentielles. Pour que B soit une réalisation de A, il faut et il suffit que l'on ait

$$(y \cdot x)_B \subset (y \cdot x)_A \quad (2.53)$$

pour chaque couple $y \times x \in Y \times X$. Le contenu d'une case quelconque de la table de transition de B est un sous-ensemble du contenu de la case correspondante de la table de transition de A.

2.5.9 Démonstration

Supposons que B soit une réalisation de A, et considérons un couple $y \times x \in Y \times X$. On montre (2.53) comme suit. Soit $y' \in (y \cdot x)_B$. Il faut établir que $y' \in (y \cdot x)_A$. A cet effet, on choisit un $x' \in X$ quelconque. On a $yy' \in R(xx')$ par (2.21). Comme $R \subset M$ par hypothèse, il vient $yy' \in M(xx')$, d'où $y' \in (y \cdot x)_A$ par (2.21).

Supposons que (2.53) soit vraie pour chaque couple $y \times x \in Y \times X$. Pour une séquence d'entrée x de longueur 1, on a $R(x) = M(x) = Y$ par (2.20), donc $R(x) \subset M(x)$. Pour une séquence d'entrée $x(1, n)$ de longueur $n > 1$, on a

$$\begin{aligned} y(1, n) \in R(x) &\iff y(k+1) \in (y(k) \cdot x(k))_B \quad \text{par (2.21)} \\ &\quad k = 1, \dots, n-1 \\ &\implies y(k+1) \in (y(k) \cdot x(k))_A \quad \text{par (2.53)} \\ &\quad k = 1, \dots, n-1 \\ &\iff y(1, n) \in M(x) \quad \text{par (2.21)}. \end{aligned}$$

Ceci prouve que $R(x) \subset M(x)$ pour toute séquence d'entrée x , donc que B est une réalisation de A.

2.5.10 Exemple

Soient $X = \{0, 1\}$, et $Y = \{a, b, c, d\}$. Le tableau 2.45 est la table de transition d'une machine séquentielle A de fonctionnement $M : X^+ \rightarrow Y^+$ (machine du paragraphe 2.4.2). Le tableau 2.46 est la table de transition d'une machine séquentielle B de fonctionnement $R : X^+ \rightarrow Y^+$. On observe que $(y \cdot x)_B \subset (y \cdot x)_A$ pour chaque couple $y \cdot x \in Y \times X$. Donc B est une réalisation de A.

	0	1
a	b	a, c
b	b	d
c	c, d	d
d	a	a, b, c, d

Tableau 2.45

	0	1
a	b	a
b	b	d
c	c, d	d
d	a	b, d

Tableau 2.46

2.5.11 Délais

Nous appelons *machine séquentielle de délai unité*, ou simplement *délai*, toute machine séquentielle $\Delta : X^+ \rightarrow Y^+$ telle que

$$X = Y \quad (2.54)$$

et

$$y \cdot x = x \quad (\forall y \in Y, \forall x \in X). \quad (2.55)$$

Les conséquences immédiates de cette dernière propriété sont les suivantes :

- Δ est complètement spécifiée;
- pour deux séquences quelconques $x(1, n)$ et $y(1, n)$ de longueur $n > 1$ sur l'alphabet $X = Y$, on a

$$y \in \Delta(x) \iff \begin{cases} y(k+1) = x(k) \\ \text{pour } k = 1, \dots, n-1. \end{cases} \quad (2.56)$$

2.5.12 Exemples

La figure 2.47 donne la table et le graphe de transition d'un délai $\Delta : \{0, 1\}^+ \rightarrow \{0, 1\}^+$. On reconnaît dans cette machine le modèle d'une bascule bistable D (voir sect. V.3.4).

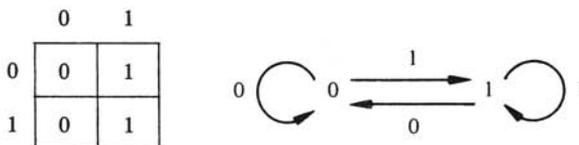


Fig. 2.47

La figure 2.48 donne la table et le graphe de transition d'un délai $\Delta : \{0, 1, 2\}^+ \rightarrow \{0, 1, 2\}^+$. Elle montre comment la notion de délai généralise le modèle d'une bascule D .

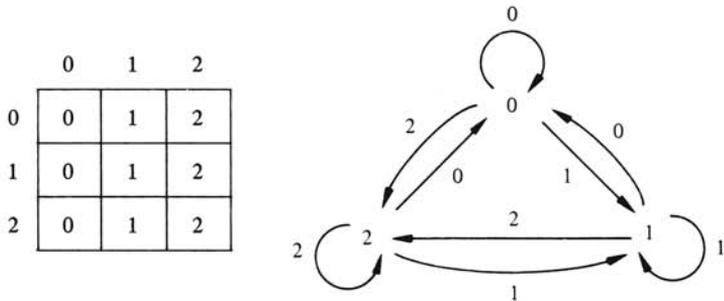


Fig. 2.48

2.5.13 Proposition

Soient A une machine séquentielle *complètement spécifiée* de fonctionnement $M : X^+ \rightarrow Y^+$ (fig. 2.49), et B une machine combinatoire (fig. 2.49) ayant les propriétés suivantes :

- les alphabets d'entrée de B sont X et Y;
- l'alphabet de sortie de B est Y;
- le fonctionnement $F : (X \times Y)^+ \rightarrow Y^+$ de B est tel que

$$F(x \times y) = (y \cdot x)_A \quad (\forall y \in Y, \forall x \in X). \quad (2.57)$$

Avec ces hypothèses, la machine C définie par le schéma de composition de la figure 2.50, où $\Delta : Y^+ \rightarrow Y^+$ est un délai, est une *réalisation* (dite *canonique*) de A.

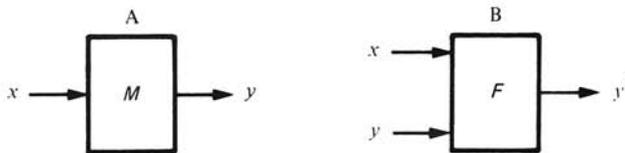


Fig. 2.49

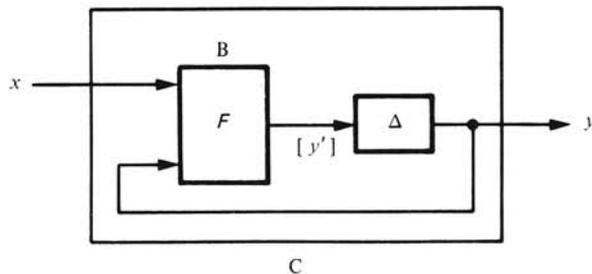


Fig. 2.50

2.5.14 Démonstration

Soit $R : X^+ \rightarrow Y^+$ le fonctionnement de C. Nous devons montrer que $R \subset M$ (§ 2.5.6). A cet effet, nous considérons deux séquences $x(1, n) \in X^+$, $y(1, n) \in Y^+$ de longueur $n > 1$, et nous supposons que $y \in R(x)$. Il s'agit de montrer que $y \in M(x)$.

En vertu du schéma de C (sect. 2.2), il existe une séquence $y'(1, n) \in Y^+$ telle que

$$y' \in F(x \times y) \quad (2.58)$$

$$y \in \Delta(y'). \quad (2.59)$$

La machine B étant combinatoire (§ 2.3.2), on peut écrire

$$\begin{aligned} F(x \times y) &= F(x(1) \times y(1)) F(x(2) \times y(2)) \dots F(x(n) \times y(n)) \\ &= (y(1) \cdot x(1))_A (y(2) \cdot x(2))_A \dots (y(n) \cdot x(n))_A \quad (\text{par (2.57)}). \end{aligned}$$

Par suite, (2.58) s'écrit

$$y'(k) = (y(k) \cdot x(k))_A \quad (k = 1, \dots, n).$$

D'autre part, (2.59) signifie selon (2.56) :

$$y(k+1) = y'(k) \quad (k = 1, \dots, n-1).$$

Il vient donc

$$y(k+1) = (y(k) \cdot x(k))_A \quad (k = 1, \dots, n-1)$$

d'où $y \in M(x)$.

2.6 MACHINES DE MOORE ET DE MEALY

2.6.1 Machines de Moore

On appelle *machine de Moore* toute machine C composée d'une machine séquentielle A et d'une machine combinatoire B selon le schéma de la figure 2.51.

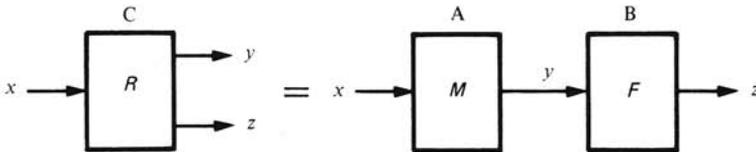


Fig. 2.51

Soient X l'alphabet d'entrée de A, et Y son alphabet de sortie. L'alphabet d'entrée de B est Y. Soit Z l'alphabet de sortie de B. Le fonctionnement de A est une correspondance $M : X^+ \rightarrow Y^+$. Celui de B est une correspondance $F : Y^+ \rightarrow Z^+$.

Le fonctionnement de la machine composée C (sect. 2.2) est une correspondance $R : X^+ \rightarrow (Y \times Z)^+$. Pour trois séquences quelconques x, y, z de longueur n sur les alphabets respectifs X, Y, Z, on a

$$y \times z \in R(x) \iff y \in M(x) \text{ et } z \in F(y). \quad (2.60)$$

Si x, y, z sont de longueur $n = 1$, la relation $y \in M(x)$ est vérifiée quelles que soient x, y (2.20), et l'on a simplement

$$y \times z \in R(x) \iff z \in F(y). \quad (2.61)$$

Si x, y, z sont de longueur $n > 1$, la relation (2.60) peut s'expliciter selon (2.21) et (2.18):

$$y \times z \in R(x) \iff \begin{cases} y(k+1) \in y(k) \cdot x(k) \\ \text{pour } k = 1, \dots, n-1 \end{cases} \text{ et } \begin{cases} z(k) \in F(y(k)) \\ \text{pour } k = 1, \dots, n \end{cases} \quad (2.62)$$

La notation $y(k) \cdot x(k)$ signifie naturellement $(y(k) \cdot x(k))_A$.

Les variables x, y, z du schéma (fig. 2.51) sont appelées respectivement *variable primaire*, *variable secondaire*, et *variable tertiaire* (ou *variable de sortie*).

La machine de Moore C est dite *complètement spécifiée* si A et B sont complètement spécifiées (§ 2.5.1 et 2.3.6). Dans ce cas, les signes \in se trouvant à droite de \iff dans (2.60), (2.61), (2.62) peuvent être remplacés par $=$.

La définition d'une machine de Moore se généralise immédiatement pour des machines A et B à entrées et/ou sorties multiples, (fig. 2.52). Une telle machine de Moore possède m variables primaires, n variables secondaires, et p variables tertiaires. Il suffit, pour la définir, de remplacer ci-dessus X par $X_1 \times \dots \times X_m$, x par $x_1 \times \dots \times x_m$, et de même pour Y, y, Z, z .

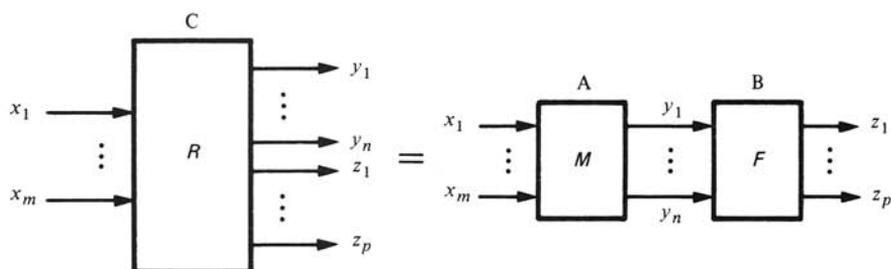


Fig. 2.52

2.6.2 Exemple

Soient $X = \{a, b, c, d\}$, $Y = \{p, q, r\}$, et $Z = \{0, 1\}$. Le tableau 2.53 est la table de transition d'une machine séquentielle $M : X^+ \rightarrow Y^+$ (de type standard). Le tableau 2.54 définit une machine combinatoire $F : Y^+ \rightarrow Z^+$. Soit $R : X^+ \rightarrow (Y \times Z)^+$ la machine de Moore de composante séquentielle M et de composante combinatoire F . On peut vérifier, en appliquant (2.62), que les séquences

$$x = accdb ; y = pqqpr ; z = 00011$$

vérifient la relation $y \times z \in R(x)$. Le tiret $-$ du tableau 2.54 est mis pour $\{0, 1\}$.

	a	b	c	d
p	q	-	p	-
q	q	r	-	p
r	-	r	q	-

Tableau 2.53

y	F(y)
p	-
q	0
r	1

Tableau 2.54

On rassemble généralement les données des tableaux 2.53 et 2.54 dans une seule table (tab. 2.55).

	<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>	
<i>p</i>	<i>q</i>	-	<i>p</i>	-	-
<i>q</i>	<i>q</i>	<i>r</i>	-	<i>p</i>	0
<i>r</i>	-	<i>r</i>	<i>q</i>	-	1

Tableau 2.55

2.6.3 Machines de Mealy

On appelle *machine de Mealy* toute machine C composée d'une machine séquentielle A (de fonctionnement $M : X^+ \rightarrow Y^+$) et d'une machine combinatoire B (de fonctionnement $F : (Y \times X)^+ \rightarrow Z^+$) selon le schéma de la figure 2.56.

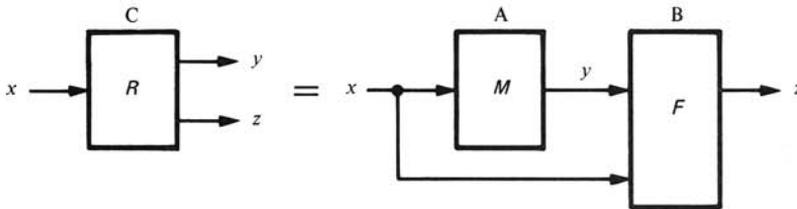


Fig. 2.56

Soit $R : X^+ \rightarrow (Y \times Z)^+$ le fonctionnement de la machine composée (sect. 2.2). Pour trois séquences quelconques x, y, z de longueur n sur les alphabets respectifs X, Y, Z , on a

$$y \times z \in R(x) \iff y \in M(x) \text{ et } z \in F(y \times x). \quad (2.63)$$

Si x, y, z sont de longueur $n = 1$, la relation $y \in M(x)$ est vérifiée quelles que soient x, y (2.20), et l'on a simplement

$$y \times z \in R(x) \iff z \in F(y \times x). \quad (2.64)$$

Si x, y, z sont de longueur $n > 1$, la relation (2.63) peut s'expliciter selon (2.21) et (2.18) :

$$y \times z \in R(x) \iff \begin{cases} y(k+1) \in y(k) \cdot x(k) \\ \text{pour } k = 1, \dots, n-1 \end{cases} \text{ et } \begin{cases} z(k) \in F(y(k) \times x(k)) \\ \text{pour } k = 1, \dots, n \end{cases} \quad (2.65)$$

Les variables x, y, z du schéma (fig. 2.56) sont appelées respectivement variable primaire, variable secondaire, et variable tertiaire (ou variable de sortie).

La définition d'une machine de Mealy se généralise immédiatement pour des composantes A, B à entrées et/ou sorties multiples, comme celle d'une machine de Moore.

2.6.4 Exemple

Soient $X = \{a, b, c, d\}$, $Y = \{p, q, r\}$, et $Z = \{0, 1\}$. Le tableau 2.57 est la table de transition d'une machine séquentielle $M : X^+ \rightarrow Y^+$ (de type standard). Le tableau 2.58 est la table d'une machine combinatoire $F : (Y \times X)^+ \rightarrow Z^+$. Cette table donne l'ensemble $F(y \times x) \subset Z$ pour chaque couple $y \times x \in Y \times X$; le tiret représente $\{0, 1\}$.

	<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>
<i>p</i>	<i>q</i>	—	<i>p</i>	—
<i>q</i>	<i>q</i>	<i>r</i>	—	<i>p</i>
<i>r</i>	—	<i>r</i>	<i>q</i>	—

Tableau 2.57

	<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>
<i>p</i>	—	1	0	0
<i>q</i>	0	—	—	1
<i>r</i>	—	1	1	0

Tableau 2.58

Soit $R : X^+ \rightarrow (Y \times Z)^+$ la machine de Mealy de composante séquentielle M et de composante combinatoire F . On peut vérifier en appliquant (2.65), que les séquences

$$x = acdba ; y = pqqpq ; z = 01110$$

vérifient la relation $y \times z \in R(x)$.

On rassemble généralement les données des tableaux 2.57 et 2.58 dans une seule table (tab. 2.59).

	<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>
<i>p</i>	<i>q</i> ; —	—; 1	<i>p</i> ; 0	—; 0
<i>q</i>	<i>q</i> ; 0	<i>r</i> ; —	—; —	<i>p</i> ; 1
<i>r</i>	—; —	<i>r</i> ; 1	<i>q</i> ; 1	—; 0

Tableau 2.59

2.6.5 Proposition

Pour toute machine de Moore C (fig. 2.60), il existe une machine de Mealy C' (fig. 2.61) ayant le même fonctionnement que C , et la même composante séquentielle.

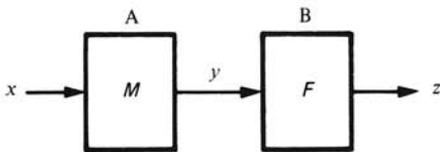


Fig. 2.60

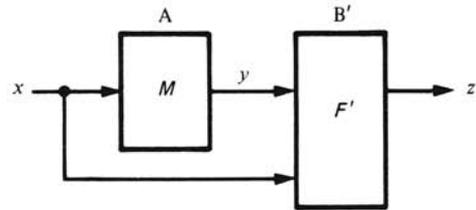


Fig. 2.61

2.6.6 Preuve

Il suffit de prendre pour B' (fig. 2.61) une machine combinatoire de fonctionnement F' tel que $F'(y \times x) = F(y)$. Le fonctionnement R' de C' , défini par (2.63), est alors identique au fonctionnement R de C , défini par (2.60). On dit que la machine B' est *dégénérée* en x .

2.6.7 Exemple

Supposons que la machine de Moore C de la figure 2.60 soit la machine définie au paragraphe 2.6.2 (tab. 2.55). Alors, la machine C' (fig. 2.61) est définie par le tableau

2.62, obtenu en recopiant simplement la dernière colonne du tableau 2.55, dans chacune des colonnes a, b, c, d .

	a	b	c	d
p	$q; -$	$-; -$	$p; -$	$-; -$
q	$q; 0$	$r; 0$	$-; 0$	$p; 0$
r	$-; 1$	$r; 1$	$q; 1$	$-; 1$

Tableau 2.62

2.6.8 Notation

Une machine de Mealy de fonctionnement $R : X^+ \rightarrow (Y \times Z)^+$, (fig. 2.56), composée d'une machine séquentielle de fonctionnement $M : X^+ \rightarrow Y^+$, et d'une machine combinatoire de fonctionnement $F : (Y \times X)^+ \rightarrow Z^+$, sera désignée par la notation $R[X, Y, Z, M, F]$ ou plus simplement par $R[X, Y, Z]$.

2.6.9 Définition

Une machine de Mealy $R[X, Y, Z, M, F]$ est dite *complètement spécifiée* si sa composante séquentielle M et sa composante combinatoire F sont complètement spécifiées (§ 2.5.1 et 2.3.6). Dans ce cas, les signes \in se trouvant à droite du signe \Leftrightarrow dans (2.63), (2.64), (2.65) peuvent être remplacés par des signes $=$.

2.6.10 Définitions

Soit $R[X, Y, Z, M, F]$ une machine de Mealy. Pour chaque couple $y \times x \in Y \times X$, la notation $(y \cdot x)_R$ désigne l'ensemble $y \cdot x$ défini par la table de transition de M . De façon générale, dans toutes les expressions relatives à R , le point \cdot est interprété selon M .

Pour chaque couple $y \times x \in Y \times X$, on pose

$$(y|x)_R = y|x = F(y \times x). \quad (2.66)$$

Par exemple, pour la machine R du paragraphe 2.6.4 (tab. 2.59), on a $q|a = \{0\}$, et $q|b = \{0, 1\}$.

Avec cette notation, la relation fondamentale (2.65) s'écrit

$$y \times z \in R(x) \Leftrightarrow \begin{cases} y(k+1) \in y(k) \cdot x(k) \\ \text{pour } k = 1, \dots, n-1 \end{cases} \text{ et } \begin{cases} z(k) \in y(k)|x(k) \\ \text{pour } k = 1, \dots, n. \end{cases} \quad (2.67)$$

Pour un sous-ensemble non vide $P \subset Y$ et un $x \in X$, on pose

$$P|x = \bigcup_{p \in P} p|x. \quad (2.68)$$

Par exemple, pour la machine du tableau 2.59, on a

$$\begin{aligned} \{p, r\}|b &= p|b \cup r|b = \{1\} \cup \{1\} = \{1\} \\ \{p, q\}|a &= p|a \cup q|a = \{0, 1\} \cup \{0\} = \{0, 1\}. \end{aligned}$$

Pour un sous-ensemble non vide $P \subset Y$ et une séquence d'entrée $x(1, n)$ de longueur $n > 1$, on pose

$$P | x(1, n) = (P \cdot x(1, n-1)) | x(n). \quad (2.69)$$

Par exemple (tab. 2.59) :

$$\begin{aligned} p | aabb &= (p \cdot aab) | b = r | b = 1 \\ \{p, q\} | aca &= (\{p, q\} \cdot ac) | a = \{p, q, r\} | a = \{0, 1\}. \end{aligned}$$

2.6.11 Proposition

Soient $R[X, Y, Z]$ une machine de Mealy, P et Q des sous-ensembles non vides de Y , x et x' des séquences sur X . On a

$$P | xx' = (P \cdot x) | x' \quad (2.70)$$

$$P | x = \bigcup_{p \in P} p | x \quad (2.71)$$

$$P \subset Q \Rightarrow P | x \subset Q | x \quad (2.72)$$

$$(P \cup Q) | x = P | x \cup Q | x \quad (2.73)$$

$$(P \cap Q) | x \subset P | x \cap Q | x \quad (2.74)$$

La preuve de ces relations est analogue au paragraphe 2.4.20.

2.6.12 Graphe de transition d'une machine de Mealy

Soit $R[X, Y, Z, M, F]$ une machine de Mealy telle que

$$Y \cap Z = \emptyset. \quad (2.75)$$

Le *graphe de transition* G de R est défini comme la réunion des deux ensembles suivants :

- l'ensemble des triplets $y \times x \times y' \in Y \times X \times Y$ tels que $y' \in y \cdot x$
- l'ensemble des triplets $y \times x \times z \in Y \times X \times Z$ tels que $z \in y | x$.

Le premier ensemble n'est autre que le graphe de transition de la composante séquentielle M (§ 2.4.5). En vertu de l'hypothèse (2.75), les deux ensembles de triplets sont disjoints. Les arêtes de G appartenant au premier (respectivement au second) ensemble sont appelées arêtes de *première espèce* (resp. *seconde espèce*) de G . Pour distinguer les deux types d'arêtes dans la représentation graphique, les arêtes de seconde espèce seront notées

$$y \xrightarrow{x} [z].$$

On adopte encore la convention d'écriture suivante, qui simplifie les figures dans la plupart des cas. Deux arêtes

$$\begin{array}{l} y \xrightarrow{x} y' \\ y \xrightarrow{x} [z] \end{array}$$

de première et seconde espèce, ayant même origine et même étiquette, peuvent être notées

$$y \xrightarrow{x[z]} y' \quad (2.76)$$

Enfin, si pour un couple $y \times x \in Y \times X$ on a $y | x = Z$, on omet de représenter les arêtes

$$y \xrightarrow{x} [z].$$

2.6.13 Exemple

La figure 2.63 donne la table et le graphe de transition de la machine de Mealy $R [X, Y, Z]$ du paragraphe 2.6.4. On a utilisé les notations standard, en omettant par exemple toutes les arêtes $p \times b \times y$ ($y \in Y$), et toutes les arêtes $p \times a \times z$ ($z \in Z$).

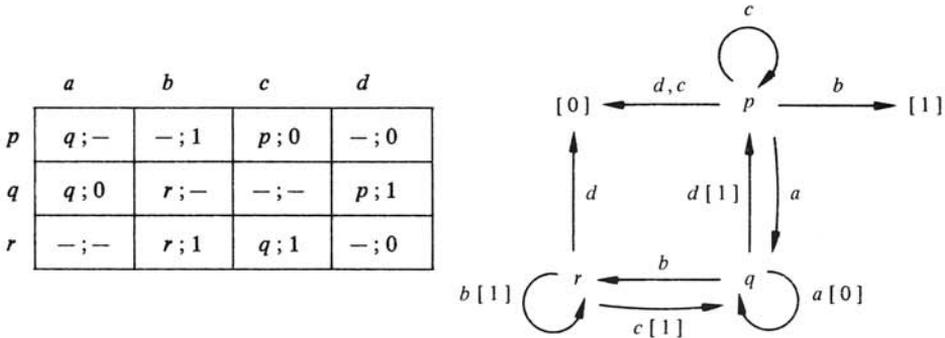


Fig. 2.63

2.6.14 Proposition

Soient $R [X, Y, Z]$ une machine de Mealy vérifiant (2.75), et $x(1, n) \in X^+$, $y(1, n) \in Y^+$, $z(1, n) \in Z^+$ des séquences de longueur $n > 1$. On a $y \times z \in R(x)$ si et seulement s'il existe dans le graphe de transition de R un "chemin" :

$$\begin{array}{ccccccc}
 y(1) & \xrightarrow{x(1)} & y(2) & \xrightarrow{x(2)} & \dots & \xrightarrow{x(n-1)} & y(n) \\
 \downarrow x(1) & & \downarrow x(2) & & & & \downarrow x(n) \\
 [z(1)] & & [z(2)] & & & & [z(n)]
 \end{array} \quad (2.77)$$

La proposition découle immédiatement de (2.67) et de la définition du graphe (§ 2.6.12).

2.6.15 Définition

Considérons une machine de Mealy (fig. 2.64) de fonctionnement $R [X, Y, Z, M, F]$, et soit P un sous-ensemble non vide de Y . On désignera par

$$R_P : X^+ \rightarrow Z^+$$

le fonctionnement de la machine de la figure 2.65, définie en substituant à la composante séquentielle A une machine A' de fonctionnement M_P (§ 2.4.27), et en ignorant la variable secondaire y .

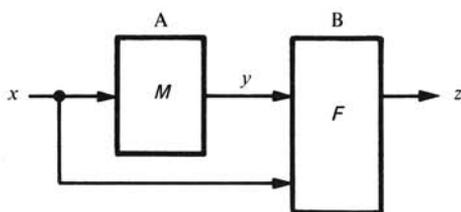


Fig. 2.64

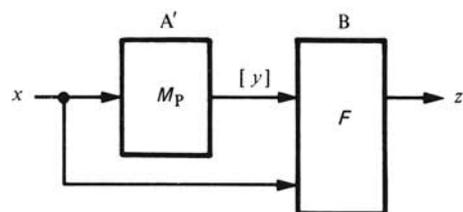


Fig. 2.65

D'après le paragraphe 2.2.10, on a

$$z \in R_P(x) \iff \begin{cases} \text{il existe une séquence } y \in M_P(x) \\ \text{telle que } z \in F(y \times x). \end{cases} \quad (2.78)$$

Ceci peut s'écrire

$$R_P(x) = \bigcup_{y \in M_P(x)} F(y \times x) \quad (2.79)$$

ou encore (§ 1.3.3)

$$R_P(x) = F(M_P(x) \times x). \quad (2.80)$$

Lorsque l'ensemble P possède un seul élément p , on écrit R_p au lieu de $R\{p\}$.

D'après (2.38) et (2.63), la relation (2.78) peut encore se mettre sous la forme

$$z \in R_P(x) \iff \begin{cases} \text{il existe une séquence } y \text{ telle que} \\ y(1) \in P \text{ et } y \times z \in R(x). \end{cases} \quad (2.81)$$

Etant donné une séquence d'entrée $x(1, n)$ de longueur $n > 1$, l'ensemble $R_P(x)$ peut donc s'obtenir au moyen du graphe de R en construisant tous les "chemins" de la forme (2.77) tels que $y(1) \in P$.

2.6.16 Exemple

Soit R la machine de Mealy du paragraphe 2.6.13. La figure 2.66 représente la construction de toutes les séquences $z \in R_P(x)$ pour $P = \{q\}$ et $x = bdd$. On a donc

$$R_q(bdd) = \{000, 001, 100, 101\} = \{0, 1\} \{00, 01\}.$$

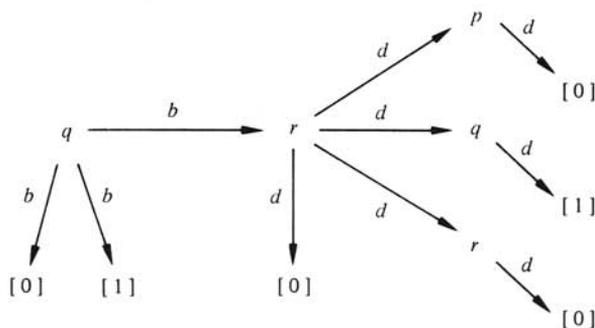


Fig. 2.66

2.6.17 Proposition

Soient $R[X, Y, Z, M, F]$ une machine de Mealy, $p \in Y$, x une séquence d'entrée de longueur 1 ($x \in X$), et x' une séquence d'entrée de longueur quelconque ($x' \in X^+$). On a

$$R_p(x) = p | x \quad (2.82)$$

$$R_p(xx') = R_p(x) R_{p \cdot x}(x'). \quad (2.83)$$

2.6.18 Démonstration

La relation (2.82) découle de (2.80), (2.40) et (2.66). On démontre (2.83) en écrivant

$$\begin{aligned} R_p(xx') &= F(M_p(xx') \times xx') && \text{par (2.80)} \\ &= F((M_p(x) M_{p \cdot x}(x')) \times xx') && \text{par (2.41)} \\ &= F[(M_p(x) \times x) (M_{p \cdot x}(x') \times x')] && \text{par (1.64)} \\ &= F(M_p(x) \times x) F(M_{p \cdot x}(x') \times x') && \text{par (2.17)} \\ &= R_p(x) R_{p \cdot x}(x'). && \text{par (2.80)} \end{aligned}$$

2.6.19 Bibliographie

Les types de machines introduits dans les sections 2.4 (machines séquentielles) et 2.6 (machines de Moore, machines de Mealy) sont généralement présentés dans la littérature sous la dénomination globale de "machines séquentielles", ou "automates finis" [2-7]. Les noms de Moore et Mealy ont été associés à deux types de machines depuis les travaux de ces auteurs [8, 9].

Il n'existe à notre connaissance aucun ouvrage en français sur la théorie des machines séquentielles. On en trouve cependant certains éléments dans [10].

SPÉCIFICATION DES MACHINES BINAIRES

3.1 EXPRESSIONS BOOLÉENNES

3.1.1 Introduction

Rappelons qu'une machine est dite binaire (§ 2.1.7) si l'alphabet associé à chacune de ses variables d'entrée et de sortie est l'alphabet $\mathbf{B} = \{0, 1\}$. Le but du présent chapitre est d'étudier certains modes de représentation des machines binaires (équations de récurrence, graphes de récurrence, diagrammes de réceptivité) pouvant être utilisés pour spécifier ces machines, c'est-à-dire pour exprimer leur fonctionnement de façon mathématique précise, en particulier lorsque celui-ci est décrit par un cahier des charges informel. Ces modes de représentation reposent sur l'algèbre des expressions booléennes, expressions dans lesquelles interviennent des variables booléennes (pouvant prendre les valeurs 0, 1) et les opérations booléennes fondamentales OU, ET, NON (sect. V.1).

La présente section établit l'algèbre des expressions booléennes sur la base de l'algèbre de Boole des sous-ensembles d'un ensemble (§ 1.1.12). On suppose chez le lecteur une certaine familiarité avec cette algèbre. La section a pour but de préciser et distinguer clairement les trois concepts fondamentaux d'expression booléenne, de fonction booléenne, et d'équation booléenne, et de montrer les relations entre ces notions. On peut la lire rapidement en omettant toutes les démonstrations.

3.1.2 Notations

Dans le volume V, les opérations booléennes OU, ET, NON sont appelées respectivement *somme logique*, *produit logique*, *complémentation*, et notées respectivement $x + y$, xy , \bar{x} .

Dans le présent volume, elles seront appelées respectivement *disjonction*, *conjonction*, *complémentation*, et notées $x \vee y$, $x \wedge y$, \bar{x} .

La raison de ce changement de notation est d'éviter la confusion d'un produit logique xy avec la *séquence* xy sur l'alphabet \mathbf{B} . Par ailleurs, le signe $+$ est utilisé dans la notation X^+ (§ 1.4.1).

Un avantage de la notation \vee , \wedge est la ressemblance formelle de ces symboles avec les symboles \cup , \cap de réunion et d'intersection (§ 1.1.9).

Les inconvénients de la nouvelle notation sont la longueur des expressions, et la nécessité de parenthèses supplémentaires. Par exemple, l'expression $x + yz$ est interprétée spontanément comme $x + (yz)$ et non comme $(x + y)z$, car on est accoutumé à effectuer les produits avant les sommes. Par contre, avec la nouvelle notation, l'expression équivalente doit s'écrire $x \vee (y \wedge z)$, car $x \vee y \wedge z$ serait ambiguë. Pour ces raisons, on utilisera souvent dans les applications la notation mixte $x \vee yz$.

3.1.3 Variables et constantes booléennes

On appelle *variable* un symbole quelconque, par exemple la lettre x , auquel on associe un ensemble X , appelé *ensemble des valeurs* de x . On dit généralement que x est un “élément indéterminé” de X . Cette locution prête à confusion, car elle incite à écrire $x \in X$. Or x n’est pas un élément de X , mais un symbole auxiliaire auquel on peut substituer un élément quelconque de X . On dira plutôt que x est une *variable de type X*, comme dans le langage de programmation PASCAL [11], et l’on écrira $x : X$. Par opposition, un élément de X est appelé une *constante de type X*.

Une *variable booléenne* est une variable de type $\mathbf{B} = \{0, 1\}$. Plusieurs variables booléennes x_1, \dots, x_n seront déclarées par la notation $x_1, \dots, x_n : \mathbf{B}$. Les éléments 0, 1 sont les *constantes booléennes*.

3.1.4 Expressions booléennes

Soient $x_1, \dots, x_n : \mathbf{B}$. On appelle *expressions booléennes sur x_1, \dots, x_n* , en abrégé $\mathbf{EB}(x_1, \dots, x_n)$, les expressions que l’on peut écrire au moyen des symboles

$$0, 1, x_1, \dots, x_n, \vee, \wedge, \bar{}, (,)$$

en se conformant aux règles (3.1) à (3.4) ci-dessous, appelées *règles d’écriture* (ou *syntaxe*) des expressions booléennes. La règle (3.1) énumère explicitement $n + 2$ expressions booléennes dites *élémentaires*. Les règles (3.2) à (3.4) sont des règles d’assemblage, déterminant les nouvelles expressions que l’on peut former à partir d’expressions déjà écrites. Un assemblage de symboles est une $\mathbf{EB}(x_1, \dots, x_n)$ si et seulement s’il peut être engendré (construit) de proche en proche, en partant d’expressions élémentaires et en appliquant les règles d’assemblage. Une telle définition est dite *inductive* (ou *générative*).

$$\text{Les expressions } 0, 1, x_1, \dots, x_n \text{ sont des } \mathbf{EB}(x_1, \dots, x_n). \quad (3.1)$$

$$\text{Si } F \text{ et } G \text{ sont deux } \mathbf{EB}(x_1, \dots, x_n), \text{ l’expression } (F \vee G) \text{ est une } \mathbf{EB}(x_1, \dots, x_n). \quad (3.2)$$

$$\text{Si } F \text{ et } G \text{ sont deux } \mathbf{EB}(x_1, \dots, x_n), \text{ l’expression } (F \wedge G) \text{ est une } \mathbf{EB}(x_1, \dots, x_n). \quad (3.3)$$

$$\text{Si } F \text{ est une } \mathbf{EB}(x_1, \dots, x_n), \text{ l’expression } \bar{F} \text{ est une } \mathbf{EB}(x_1, \dots, x_n). \quad (3.4)$$

Pour déclarer que les symboles F, G, \dots désignent des $\mathbf{EB}(x_1, \dots, x_n)$, nous écrirons $F, G, \dots : \mathbf{EB}(x_1, \dots, x_n)$.

3.1.5 Exemple

Soient $a, b, c : \mathbf{B}$. Les expressions $a, 0, b, (a \vee 0), (\overline{a \vee 0}), ((\overline{a \vee 0}) \wedge b)$ sont des $\mathbf{EB}(a, b, c)$ en vertu de (3.1) à (3.4), comme le montre la figure 3.1.

3.1.6 Remarques

La phrase “ $F : \mathbf{EB}(x_1, \dots, x_n)$ ” n’implique pas que toutes les lettres x_1, \dots, x_n figurent dans F . En particulier, F peut ne comporter que des constantes 0, 1. Par contre, la phrase implique qu’aucune variable distincte de x_1, \dots, x_n ne figure dans F . Si x_{n+1}

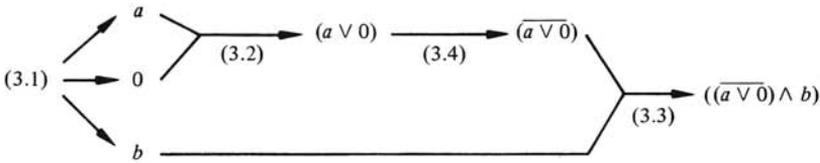


Fig. 3.1

est une variable distincte de x_1, \dots, x_n , la phrase $F : \mathbf{EB}(x_1, \dots, x_n)$ implique $F : \mathbf{EB}(x_1, \dots, x_{n+1})$. Si x_n ne figure pas dans F , la phrase $F : \mathbf{EB}(x_1, \dots, x_n)$ implique $F : \mathbf{EB}(x_1, \dots, x_{n-1})$.

3.1.7 Ensembles représentés par une expression booléenne

Nous désignerons par \mathbf{B}_n le produit cartésien de n facteurs $\mathbf{B} \times \dots \times \mathbf{B}$, et nous posons $\mathbf{B}_1 = \mathbf{B}$. Un élément

$$\xi = (\xi_1, \dots, \xi_n) = \xi_1 \times \dots \times \xi_n \in \mathbf{B}_n$$

est appelé un *vecteur binaire* de dimension n , et de *composantes* ξ_1, \dots, ξ_n .

Soit x_1, \dots, x_n une liste *ordonnée* de variables booléennes distinctes. Nous allons associer à toute expression $F : \mathbf{EB}(x_1, \dots, x_n)$ un *sous-ensemble* de \mathbf{B}_n , noté $(\sigma x_1, \dots, x_n)F$, et appelé le *sous-ensemble de \mathbf{B}_n représenté par F selon la liste x_1, \dots, x_n* .

Ce sous-ensemble est défini d'une manière inductive calquée sur la définition des expressions booléennes, (3.1) à (3.4). Les règles (3.5), (3.6), (3.7) définissent explicitement les sous-ensembles représentés par les expressions élémentaires (3.1). Ainsi, selon (3.7), le sous-ensemble de \mathbf{B}_n représenté par l'expression x_i (selon la liste x_1, \dots, x_n) est l'ensemble des vecteurs binaires dont la i -ème composante vaut 1. Les règles (3.8) à (3.10) définissent les sous-ensembles représentés par des expressions de la forme $(F \vee G)$, $(F \wedge G)$, \overline{F} comme la réunion, l'intersection et le complément par rapport à \mathbf{B}_n des sous-ensembles représentés par F et G . On abrège le préfixe $(\sigma x_1, \dots, x_n)$ par $(\sigma x_{1,n})$.

$$(\sigma x_{1,n}) 0 = \emptyset \quad (3.5)$$

$$(\sigma x_{1,n}) 1 = \mathbf{B}_n \quad (3.6)$$

$$(\sigma x_{1,n}) x_i = \{\xi \in \mathbf{B}_n \mid \xi_i = 1\} \quad (3.7)$$

$$(\sigma x_{1,n})(F \vee G) = (\sigma x_{1,n})F \cup (\sigma x_{1,n})G \quad (3.8)$$

$$(\sigma x_{1,n})(F \wedge G) = (\sigma x_{1,n})F \cap (\sigma x_{1,n})G \quad (3.9)$$

$$(\sigma x_{1,n})\overline{F} = \overline{(\sigma x_{1,n})F}. \quad (3.10)$$

3.1.8 Exemples

Soient $a, b, c : \mathbf{B}$. Le tableau 3.2 représente l'ensemble \mathbf{B}_3 , c'est-à-dire les 8 vecteurs binaires de dimension 3. On veut déterminer le sous-ensemble $(\sigma a, b, c)(\overline{a} \wedge b) \subset \mathbf{B}_3$. L'ensemble $(\sigma a, b, c)a$ se compose des quatre derniers vecteurs du tableau, en vertu de (3.7), et $(\sigma a, b, c)\overline{a}$ des quatre premiers vecteurs, en vertu de (3.10). De même, $(\sigma a, b, c)b$ se compose des vecteurs 3) à 6) en vertu de (3.7). Enfin, $(\sigma a, b, c)(\overline{a} \wedge b)$ se

compose des vecteurs 3) et 4) en vertu de (3.9). On a donc

$$(\sigma a, b, c)(\bar{a} \wedge b) = \{(0, 1, 0), (0, 1, 1)\}. \tag{3.11}$$

		<i>a</i>	<i>b</i>	<i>c</i>		
$(\sigma a, b, c) \bar{a}$	{	1)	0	0	0	}
		2)	0	0	1	
		3)	0	1	0	
		4)	0	1	1	
$(\sigma a, b, c) a$	{	5)	1	1	0	
		6)	1	1	1	
		7)	1	0	0	
		8)	1	0	1	

Tableau 3.2

Les tableaux 3.3 et 3.4 montrent que pour la même expression $\bar{a} \wedge b$, on a

$$(\sigma a, b)(\bar{a} \wedge b) = \{(0, 1)\} \tag{3.12}$$

$$(\sigma b, a)(\bar{a} \wedge b) = \{(1, 0)\}. \tag{3.13}$$

		<i>a</i>	<i>b</i>
$(\sigma a, b) \bar{a}$	{	0	0
		0	1
		1	1
		1	0

Tableau 3.3

		<i>b</i>	<i>a</i>
$(\sigma b, a) \bar{a}$	{	0	0
		1	0
		1	1
		0	1

Tableau 3.4

Ces exemples montrent comment est modifié l'ensemble $(\sigma x_1, \dots, x_n)F$ représenté par une expression booléenne F lorsqu'on ajoute une variable x_{n+1} au préfixe $(\sigma \dots)$, ou lorsqu'on permute deux variables dans ce préfixe. L'effet de ces deux opérations est étudié de façon détaillée dans les paragraphes 3.1.32 à 3.1.35.

3.1.9 Définition

Deux expressions $F, G : \mathbf{EB}(x_1, \dots, x_n)$ sont dites *équivalentes* si elles représentent le même sous-ensemble de \mathbf{B}_n selon la liste x_1, \dots, x_n , c'est-à-dire si

$$(\sigma x_1, \dots, x_n)F = (\sigma x_1, \dots, x_n)G. \tag{3.14}$$

On montre (§ 3.1.36) que si l'on modifie le préfixe ($\sigma \dots$) de la même manière dans les deux membres de (3.14), l'égalité des deux ensembles est conservée. Pour cette raison, on peut se permettre d'omettre le préfixe, et d'écrire simplement $F = G$ pour exprimer l'équivalence de F et G .

3.1.10 Algèbre des expressions booléennes

En abrégant $(\sigma x_1, \dots, x_n)F$ par F , les définitions (3.5) à (3.10) deviennent simplement $0 = \emptyset$, $1 = \mathbf{B}_n$, $x_i = \{\xi \in \mathbf{B}_n \mid \xi_i = 1\}$, $F \vee G = F \cup G$, $F \wedge G = F \cap G$, $\overline{F} = \overline{F}$. Ceci définit l'interprétation des expressions booléennes comme sous-ensembles de \mathbf{B}_n , dans laquelle les symboles \vee , \wedge , $\overline{}$ sont interprétés comme les symboles de réunion, d'intersection, et de complémentation par rapport à \mathbf{B}_n .

L'algèbre des expressions booléennes est basée sur cette interprétation. On peut donc appliquer toutes les formules du paragraphe 1.1.12 (algèbre de Boole des sous-ensembles d'un ensemble E), en y remplaçant \emptyset , E , \cup , \cap respectivement par 0 , 1 , \vee , \wedge , et les lettres X , Y , Z par des expressions booléennes quelconques F , G , H . On obtient ainsi le formulaire ci-dessous, (3.15) à (3.24), d'après (1.10) à (1.21).

$$F \vee G = G \vee F \quad ; \quad F \wedge G = G \wedge F \quad (3.15)$$

$$\left. \begin{aligned} F \vee (G \vee H) &= (F \vee G) \vee H \\ F \wedge (G \wedge H) &= (F \wedge G) \wedge H \end{aligned} \right\} \quad (3.16)$$

$$\left. \begin{aligned} F \vee (G \wedge H) &= (F \vee G) \wedge (F \vee H) \\ F \wedge (G \vee H) &= (F \wedge G) \vee (F \wedge H) \end{aligned} \right\} \quad (3.17)$$

$$\left. \begin{aligned} F \vee 0 &= F \quad ; \quad F \wedge 1 = F \\ F \vee 1 &= 1 \quad ; \quad F \wedge 0 = 0 \\ F \vee \overline{F} &= 1 \quad ; \quad F \wedge \overline{F} = 0 \end{aligned} \right\} \quad (3.18)$$

$$F \vee F = F \quad ; \quad F \wedge F = F \quad (3.19)$$

$$F \vee (F \wedge G) = F \quad ; \quad F \wedge (F \vee G) = F \quad (3.20)$$

$$\overline{\overline{F}} = F \quad (3.21)$$

$$\overline{F \vee G} = \overline{F} \wedge \overline{G} \quad ; \quad \overline{F \wedge G} = \overline{F} \vee \overline{G} \quad (3.22)$$

$$\overline{1} = 0 \quad ; \quad \overline{0} = 1 \quad (3.23)$$

$$\left. \begin{aligned} F \vee G = 0 &\implies F = 0 \text{ et } G = 0 \\ F \wedge G = 1 &\implies F = 1 \text{ et } G = 1 \end{aligned} \right\} \quad (3.24)$$

On tire en particulier de (3.18) et (3.15):

$$0 \vee 1 = 1 \vee 0 = 1 \vee 1 = 1 \quad (3.25)$$

$$0 \wedge 1 = 1 \wedge 0 = 0 \wedge 0 = 0 \quad (3.26)$$

$$0 \vee 0 = 0 \quad ; \quad 1 \wedge 1 = 1. \quad (3.27)$$

On reconnaît dans (3.23), (3.25), (3.26), (3.27) les formules usuelles de la complémentation, de la somme logique (ou disjonction \vee), et du produit logique (ou conjonction \wedge).

3.1.11 Inégalités

Soient $x_1, \dots, x_n : \mathbf{B}$, et $F, G : \mathbf{EB}(x_1, \dots, x_n)$. On écrit $F \leq G$ lorsque $(\sigma x_1, \dots, x_n)F \subset (\sigma x_1, \dots, x_n)G$. Les formules (1.22) à (1.27) permettent donc d'écrire pour des expressions booléennes F, G, H quelconques :

$$F \leq G \iff F \vee G = G \quad ; \quad F \geq G \iff F \wedge G = G \quad (3.28)$$

$$F \leq F \vee G \quad ; \quad F \geq F \wedge G \quad (3.29)$$

$$F \leq H \text{ et } G \leq H \iff F \vee G \leq H \quad (3.30)$$

$$F \geq H \text{ et } G \geq H \iff F \wedge G \geq H \quad (3.31)$$

$$F \leq G \iff F \wedge \bar{G} = 0 \quad ; \quad F \geq G \iff F \vee \bar{G} = 1 \quad (3.32)$$

$$F \leq G \iff \bar{G} \leq \bar{F} \quad (3.33)$$

$$F \leq G \implies F \vee H \leq G \vee H \quad (3.34)$$

$$F \geq G \implies F \wedge H \geq G \wedge H. \quad (3.35)$$

Lorsque $F \leq G$ et $F \neq G$, on écrit naturellement $F < G$.

3.1.12 Proposition

Soient $x_1, \dots, x_n : \mathbf{B}$, et $F : \mathbf{EB}(x_1, \dots, x_n)$. Un vecteur binaire $\xi = (\xi_1, \dots, \xi_n) \in \mathbf{B}_n$ appartient au sous-ensemble $(\sigma x_1, \dots, x_n)F$ de \mathbf{B}_n si et seulement si l'expression obtenue en remplaçant x_1 par ξ_1, \dots, x_n par ξ_n dans F est égale à 1.

La démonstration est donnée au paragraphe 3.1.40.

3.1.3 Exemple

Soient $a, b, c : \mathbf{B}$, et F l'expression $(\bar{a} \wedge b) \vee c$. Considérons le vecteur $\xi = (\xi_1, \xi_2, \xi_3) = (0, 1, 0) \in \mathbf{B}_3$. En remplaçant a par ξ_1, b par ξ_2 , et c par ξ_3 dans F , on obtient l'expression

$$(\bar{0} \wedge 1) \vee 0 = (1 \wedge 1) \vee 0 = 1 \vee 0 = 1.$$

Ce vecteur ξ appartient donc à l'ensemble $(\sigma a, b, c)F$. Ce n'est pas le cas du vecteur $(1, 1, 0)$, pour lequel la substitution donne $(\bar{1} \wedge 1) \vee 0 = 0 \vee 0 = 0$.

3.1.14 Notations

La proposition du paragraphe 3.1.12 peut s'exprimer de façon succincte au moyen des deux notations introduites ci-dessous.

- Si R est une relation quelconque, nous désignons par $\langle R \rangle$ la *valeur de vérité* de R , à savoir l'élément de \mathbf{B} défini comme suit :

$$\langle R \rangle = \begin{cases} 0 & \text{si } R \text{ est fausse} \\ 1 & \text{si } R \text{ est vraie} \end{cases} \quad (3.36)$$

- L'opération de substitution décrite au paragraphe 3.1.2 est notée

$$(x_1 \leftarrow \xi_1, \dots, x_n \leftarrow \xi_n)F. \quad (3.37)$$

Lire : " x_1 est remplacé par ξ_1, \dots dans F ". Cette notation désigne l'expression résultant de la substitution.

Avec ces notations, la proposition (§ 3.1.12) s'écrit :

$$\langle \xi \in (\sigma x_1, \dots, x_n)F \rangle = (x_1 \leftarrow \xi_1, \dots, x_n \leftarrow \xi_n)F. \quad (3.38)$$

3.1.15 Minternes

Soient $x_1, \dots, x_n : \mathbf{B}$. On appelle *minterme* sur x_1, \dots, x_n toute expression $F : \mathbf{EB}(x_1, \dots, x_n)$ de la forme $F_1 \wedge \dots \wedge F_n$, où chacune des expressions F_i ($i = 1, \dots, n$) est soit \bar{x}_i , soit x_i .

3.1.16 Exemple

Soient $a, b, c : \mathbf{B}$. Le tableau 3.5 montre tous les mintermes sur (a) , sur (a, b) , et sur (a, b, c) .

3.1.17 Proposition

Pour tout minterme F sur x_1, \dots, x_n le sous-ensemble $(\sigma x_1, \dots, x_n)F \subset \mathbf{B}_n$ possède un élément et un seul; pour chaque élément $\xi \in \mathbf{B}_n$, il existe un minterme F sur (x_1, \dots, x_n) tel que $(\sigma x_1, \dots, x_n)F = \{\xi\}$.

La démonstration, basée sur (3.38), est laissée au lecteur.

mintermes sur		
(a)	(a, b)	(a, b, c)
\bar{a}	$\bar{a} \wedge \bar{b}$	$\bar{a} \wedge \bar{b} \wedge \bar{c}$
		$\bar{a} \wedge \bar{b} \wedge c$
	$\bar{a} \wedge b$	$\bar{a} \wedge b \wedge \bar{c}$
a		$\bar{a} \wedge b \wedge c$
	$a \wedge \bar{b}$	$a \wedge \bar{b} \wedge \bar{c}$
		$a \wedge \bar{b} \wedge c$
	$a \wedge b$	$a \wedge b \wedge \bar{c}$
		$a \wedge b \wedge c$

Tableau 3.5

3.1.18 Exemple

Soient $a, b, c : \mathbf{B}$, et F le minterme $\bar{a} \wedge b \wedge \bar{c}$. Le vecteur $\xi = (0, 1, 0)$ est le seul vecteur $\xi \in \mathbf{B}_3$ tel que $(a \leftarrow \xi_1, b \leftarrow \xi_2, c \leftarrow \xi_3)F = 1$. En vertu de (3.38), c'est le seul vecteur $\xi \in (\sigma a, b, c)F$. On a donc $(\sigma a, b, c)(\bar{a} \wedge b \wedge \bar{c}) = \{(0, 1, 0)\}$.

3.1.19 Proposition

Soient $x_1, \dots, x_n : \mathbf{B}$. Chaque sous-ensemble $S \subset \mathbf{B}_n$ peut être représenté par une expression $F : \mathbf{EB}(x_1, \dots, x_n)$.

En effet, le sous-ensemble $S = \emptyset$ est représenté par l'expression 0 (3.5). Si S ne possède qu'un élément, S est représenté par un minterme (§ 3.1.17). Si S possède plusieurs éléments, S est la réunion de plusieurs sous-ensembles à un élément; par (3.8), on voit que S peut être représenté par une disjonction (somme) de mintermes.

3.1.20 Fonctions booléennes

Nous appelons *fonction booléenne* toute application $f: \mathbf{B}_n \rightarrow \mathbf{B}$, et *noyau* d'une telle application l'ensemble des vecteurs $\xi \in \mathbf{B}_n$ tels que $f(\xi) = 1$. Nous désignons cet ensemble par $\mathbf{Noy}(f)$. On peut écrire selon le paragraphe 1.3.8 :

$$\mathbf{Noy}(f) = f^{-1}(1). \quad (3.39)$$

Avec la notation (3.36), on a pour tout vecteur $\xi \in \mathbf{B}_n$:

$$f(\xi) = \langle \xi \in \mathbf{Noy}(f) \rangle. \quad (3.40)$$

Il est clair qu'une fonction booléenne $f: \mathbf{B}_n \rightarrow \mathbf{B}$ est entièrement définie par son noyau.

3.1.21 Exemple

Le tableau 3.6 représente une fonction booléenne $f: \mathbf{B}_3 \rightarrow \mathbf{B}$. Son noyau est l'ensemble

$$\mathbf{Noy}(f) = \{(0, 1, 0), (0, 1, 1)\}. \quad (3.41)$$

a	b	c	f
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	1	0	0
1	1	1	0
1	0	0	0
1	0	1	0

Tableau 3.6

x	y	z	f	g
0	0	0	1	0
0	0	1	1	1
0	1	0	0	0
0	1	1	0	1
1	0	0	1	0
1	0	1	1	0
1	1	0	1	0
1	1	1	1	0

Tableau 3.7

3.1.22 Fonctions représentées par une expression booléenne

Soient $x_1, \dots, x_n \in \mathbf{B}$, et $F \in \mathbf{EB}(x_1, \dots, x_n)$. La *fonction booléenne* $f: \mathbf{B}_n \rightarrow \mathbf{B}$ représentée par F selon l'ordre x_1, \dots, x_n des variables est la fonction dont le noyau est le sous-ensemble de \mathbf{B}_n représenté par F selon le même ordre des variables. Cette fonction est désignée par la notation $(\lambda x_1, \dots, x_n)F$. Elle est donc entièrement définie par la relation

$$\mathbf{Noy}((\lambda x_1, \dots, x_n)F) = (\sigma x_1, \dots, x_n)F. \quad (3.42)$$

Pour cette fonction booléenne $f = (\lambda x_1, \dots, x_n)F$, on a d'après (3.40), (3.42), et (3.38) :

$$f(\xi) = (x_1 \leftarrow \xi_1, \dots, x_n \leftarrow \xi_n)F \quad (3.43)$$

pour tout vecteur $\xi \in \mathbf{B}_n$.

3.1.23 Exemple

Le noyau de la fonction $f: \mathbf{B}_3 \rightarrow \mathbf{B}$ du tableau 3.6 est l'ensemble $(\sigma a, b, c)(\bar{a} \wedge b)$, cf. (3.11). Cette fonction peut donc être notée $(\lambda a, b, c)(\bar{a} \wedge b)$.

Soulignons que le choix des variables a, b, c est tout à fait arbitraire. L'expression $(\lambda x, y, z)(\bar{x} \wedge y)$ désigne exactement la même fonction. On dit que dans une expression telle que $(\lambda x, y, z)(\bar{x} \wedge y)$ ou $(\sigma x, y, z)(\bar{x} \wedge y)$, les variables x, y, z sont *muettes*.

3.1.24 Equations booléennes

Soient $x_1, \dots, x_n : \mathbf{B}$. On appelle *équation booléenne sur x_1, \dots, x_n* toute expression de la forme $F = G$, où F et G sont deux expressions booléennes sur x_1, \dots, x_n . On appelle *solution dans \mathbf{B}_n* de l'équation $F = G$ selon l'ordre x_1, \dots, x_n des variables, tout vecteur $\xi = (\xi_1, \dots, \xi_n) \in \mathbf{B}_n$ tel que

$$(x_1 \leftarrow \xi_1, \dots, x_n \leftarrow \xi_n)F = (x_1 \leftarrow \xi_1, \dots, x_n \leftarrow \xi_n)G. \quad (3.44)$$

Si l'on désigne respectivement par f et g les fonctions booléennes $(\lambda x_1, \dots, x_n)F$ et $(\lambda x_1, \dots, x_n)G$, on peut, d'après (3.43), exprimer qu'un vecteur $\xi \in \mathbf{B}_n$ est solution de l'équation $F = G$ par

$$f(\xi) = g(\xi). \quad (3.45)$$

L'ensemble des solutions dans \mathbf{B}_n de l'équation $F = G$ selon l'ordre x_1, \dots, x_n est noté $(\sigma x_1, \dots, x_n)(F = G)$.

3.1.25 Exemple

Soient $x, y, z : \mathbf{B}$, et F, G les expressions $x \vee \bar{y}$, $z \wedge \bar{x}$. Les fonctions $f = (\lambda x, y, z)F$ et $g = (\lambda x, y, z)G$ sont représentées par le tableau 3.7. On voit que les solutions dans \mathbf{B}_3 de l'équation $F = G$ selon l'ordre x, y, z sont les vecteurs $(0, 0, 1)$, $(0, 1, 0)$. On écrit donc

$$(\sigma x, y, z)(x \vee \bar{y} = z \wedge \bar{x}) = \{(0, 0, 1), (0, 1, 0)\}. \quad (3.46)$$

3.1.26 Proposition

Soient $x_1, \dots, x_n : \mathbf{B}$, et $F, G : \mathbf{EB}(x_1, \dots, x_n)$. L'ensemble des solutions dans \mathbf{B}_n de l'équation $F = G$ selon l'ordre x_1, \dots, x_n est égal au sous-ensemble de \mathbf{B}_n représenté par l'expression $(\bar{F} \wedge \bar{G}) \vee (F \wedge G)$ selon le même ordre des variables. On a donc

$$(\sigma x_1, \dots, x_n)(F = G) = (\sigma x_1, \dots, x_n)((\bar{F} \wedge \bar{G}) \vee (F \wedge G)). \quad (3.47)$$

3.1.27 Démonstration

Soient $f = (\lambda x_1, \dots, x_n)F$ et $g = (\lambda x_1, \dots, x_n)G$. Un vecteur $\xi \in \mathbf{B}_n$ est solution de $F = G$ (§ 3.1.24) si et seulement si

$$(f(\xi) = 0 \text{ et } g(\xi) = 0) \quad \text{ou} \quad (f(\xi) = 1 \text{ et } g(\xi) = 1).$$

En vertu de (3.40) et (3.42), cette relation peut s'écrire, en abrégéant $(\sigma x_1, \dots, x_n)$ par σ :

$$(\xi \in \overline{\sigma F} \text{ et } \xi \in \overline{\sigma G}) \quad \text{ou} \quad (\xi \in \sigma F \text{ et } \xi \in \sigma G)$$

ce qui équivaut (§ 1.1.9) à

$$\xi \in (\overline{\sigma F} \cap \overline{\sigma G}) \cup (\sigma F \cap \sigma G)$$

et par (3.8), (3.9), (3.10) :

$$\xi \in \sigma((\overline{F} \wedge \overline{G}) \vee (F \wedge G)).$$

3.1.28 Proposition

Pour toute expression booléenne F sur $x_1, \dots, x_n : \mathbf{B}$, le sous-ensemble de \mathbf{B}_n représenté par F (§ 3.1.7) est l'ensemble des solutions dans \mathbf{B}_n de l'équation $F = 1$. Autrement dit

$$(\sigma x_1, \dots, \sigma x_n)F = (\sigma x_1, \dots, \sigma x_n)(F = 1). \quad (3.48)$$

Cette proposition découle immédiatement du paragraphe 3.1.24, de (3.38), et du fait que $(x_1 \leftarrow \xi_1, \dots, x_n \leftarrow \xi_n)1$ est identiquement 1.

3.1.29 Définition

Deux équations booléennes $F = G$, $F' = G'$ sur x_1, \dots, x_n sont *équivalentes* si elles admettent les mêmes solutions, c'est-à-dire si

$$(\sigma x_1, \dots, \sigma x_n)(F = G) = (\sigma x_1, \dots, \sigma x_n)(F' = G'). \quad (3.49)$$

3.1.30 Proposition

Toute équation booléenne est équivalente à une équation booléenne de la forme $F = 1$.

En effet, soient $F, G : \mathbf{EB}(x_1, \dots, x_n)$. L'équation $F = G$ est équivalente à l'équation $(\overline{F} \wedge \overline{G}) \vee (F \wedge G) = 1$ d'après (3.47) et (3.48).

3.1.31 Commentaire

La fin de la présente section, à partir du paragraphe 3.1.32, peut être considérée comme une annexe. Elle complète les paragraphes 3.1.8, 3.1.9 et 3.1.12.

3.1.32 Proposition

Soient $x_1, \dots, x_n, x_{n+1} : \mathbf{B}$ et $F : \mathbf{EB}(x_1, \dots, x_n)$. On a

$$(\sigma x_1, \dots, \sigma x_{n+1})F = (\sigma x_1, \dots, \sigma x_n)F \times \mathbf{B}. \quad (3.50)$$

Cette relation est illustrée par (3.11) et (3.12).

3.1.33 Démonstration

Soit $R(F)$ la relation (3.50). Pour démontrer que $R(F)$ est vraie pour toute expression $F : \mathbf{EB}(x_1, \dots, x_n)$, il suffit d'établir les points suivants :

- $R(0), R(1), R(x_i)$ ($i = 1, \dots, n$) sont vraies cf. (3.1)
- $R(F)$ et $R(G)$ impliquent $R(F \vee G)$ cf. (3.2)

- $R(F)$ et $R(G)$ impliquent $R(F \wedge G)$ cf.(3.3)
- $R(F)$ implique $R(\bar{F})$ cf.(3.4)

Premier point : on a $(\sigma x_{1, n+1})0 = (\sigma x_{1, n})0 = \emptyset$ par (3.5), et $\emptyset = \emptyset \times \mathbf{B}$ par (1.32).
 $(\sigma x_{1, n+1})1 = \mathbf{B}_{n+1}$ par (3.6), $\mathbf{B}_{n+1} = \mathbf{B}_n \times \mathbf{B} = (\sigma x_{1, n})1 \times \mathbf{B}$. Donc $R(0)$ et $R(1)$
sont vraies. Pour établir $R(x_i)$, on peut mettre la définition (3.7) sous la forme

$$\left. \begin{aligned} (\sigma x_{1, n})x_i &= \mathbf{X}_1 \times \dots \times \mathbf{X}_n \\ \text{avec } \mathbf{X}_j &= \begin{cases} \mathbf{B} & \text{pour } j \neq i \\ \{1\} & \text{pour } j = i. \end{cases} \end{aligned} \right\} \quad (3.51)$$

Il en découle immédiatement $(\sigma x_{1, n+1})x_i = (\sigma x_{1, n})x_i \times \mathbf{B}$ pour $i = 1, \dots, n$.
Deuxième point : supposons que $R(F)$ et $R(G)$ soient vraies. On a

$$\begin{aligned} (\sigma x_{1, n+1})(F \vee G) &= (\sigma x_{1, n+1})F \cup (\sigma x_{1, n+1})G && \text{par (3.8)} \\ &= ((\sigma x_{1, n})F \times \mathbf{B}) \cup ((\sigma x_{1, n})G \times \mathbf{B}) && \text{par hypothèse} \\ &= ((\sigma x_{1, n})F \cup (\sigma x_{1, n})G) \times \mathbf{B} && \text{par (1.35)} \\ &= (\sigma x_{1, n})(F \vee G) \times \mathbf{B} && \text{par (3.8)}. \end{aligned}$$

Troisième point : démonstration analogue basée sur (3.9) et (1.36). Quatrième point :
supposons que $R(F)$ soit vraie. On démontre $R(\bar{F})$ en utilisant (3.10) ainsi que le
lemme suivant, facile à vérifier.

Lemme : pour deux ensembles quelconques A, B et un sous-ensemble quelconque
 $X \subset A$, on a $\overline{X \times B} = \bar{X} \times B$, le premier complément étant pris par rapport à $A \times B$, et
le second par rapport à A.

3.1.34 Permutations dans \mathbf{B}_n

Considérons un vecteur $\xi = (\xi_1, \dots, \xi_n) \in \mathbf{B}_n$, de dimension $n > 1$, et soient i, j
deux entiers tels que $1 \leq i \leq j \leq n$. Nous désignons par $\pi_{i,j}(\xi)$ le vecteur obtenu en
permutant les composantes ξ_i et ξ_j de ξ . Par exemple, $\pi_{1,3}(0, 0, 1, 1) = (1, 0, 0, 1)$.
Ceci définit clairement une bijection $\pi_{i,j} : \mathbf{B}_n \rightarrow \mathbf{B}_n$. Les formules suivantes sont rela-
tivement transformés par $\pi_{i,j}$ de sous-ensembles de \mathbf{B}_n (§ 1.3.3).

$$\left. \begin{aligned} \pi_{i,j}(\emptyset) &= \emptyset \\ \pi_{i,j}(\mathbf{B}_n) &= \mathbf{B}_n \\ \pi_{i,j}(X \cup Y) &= \pi_{i,j}(X) \cup \pi_{i,j}(Y) \\ \pi_{i,j}(X \cap Y) &= \pi_{i,j}(X) \cap \pi_{i,j}(Y) \\ \pi_{i,j}(\bar{X}) &= \overline{\pi_{i,j}(X)}. \end{aligned} \right\} \quad (3.52)$$

Ces formules sont une simple extension des paragraphes 1.3.3 et 1.3.5 au cas où f est
une bijection (§ 1.3.16).

Soit **liste** une liste ordonnée de n variables booléennes distinctes. Nous désignons
par $\pi_{i,j}(\mathbf{liste})$ la liste obtenue en permutant les i -ème et j -ème variables dans **liste**.

3.1.35 Proposition

Soient **liste** une liste ordonnée de n variables booléennes distinctes, et F une expression booléenne sur **liste**. On a pour toute permutation $\pi_{i,j}$ ($1 \leq i \leq j \leq n$) :

$$(\sigma \pi_{i,j}(\mathbf{liste})) F = \pi_{i,j}((\sigma \mathbf{liste}) F). \quad (3.53)$$

La démonstration se fait selon le même schéma qu'au paragraphe 3.1.33, en appelant $R(F)$ la relation (3.53), et en se basant sur (3.5) à (3.10) et (3.52). La relation (3.53) est illustrée par (3.12) et (3.13), (tab. 3.3 et 3.4).

3.1.36 Proposition

Considérons deux listes ordonnées de variables booléennes, que nous désignons par **liste**₁ et **liste**₂, et deux expressions F, G telles que

$$F, G : \mathbf{EB}(\mathbf{liste}_1) \quad \text{et} \quad F, G : \mathbf{EB}(\mathbf{liste}_2). \quad (3.54)$$

Si

$$(\sigma \mathbf{liste}_1) F = (\sigma \mathbf{liste}_1) G \quad (3.55)$$

alors

$$(\sigma \mathbf{liste}_2) F = (\sigma \mathbf{liste}_2) G. \quad (3.56)$$

3.1.37 Démonstration

Etant donné deux expressions booléennes F, G , il existe une infinité de listes de variables booléennes **liste** pour lesquelles on peut écrire $F, G : \mathbf{EB}(\mathbf{liste})$. Toutes ces listes peuvent être obtenues en partant d'une liste minimale, et en y effectuant des adjonctions et/ou des permutations. Il suffit donc de montrer que (3.55) implique (3.56) dans les deux cas suivants :

- **liste**₂ est obtenue par adjonction d'une variable à **liste**₁
- **liste**₂ est obtenue par permutation de deux variables dans **liste**₁.

Cela se fait immédiatement en appliquant (3.50) à F et à G dans le premier cas, et (3.53) dans le second.

3.1.38 Substitutions

Dans une expression $F : \mathbf{EB}(x_1, \dots, x_n)$, on peut substituer des expressions $X_1, \dots, X_n : \mathbf{EB}(x_1, \dots, x_n)$ aux variables x_1, \dots, x_n . L'expression résultant de l'opération est notée

$$(x_1 \leftarrow X_1, \dots, x_n \leftarrow X_n) F. \quad (3.57)$$

Lorsque les expressions X_i sont des constantes (3.57), les substitutions $x_1 \leftarrow X_1, \dots, x_n \leftarrow X_n$ peuvent être effectuées séquentiellement dans n'importe quel ordre. Lorsque les expressions X_i contiennent des variables, les substitutions doivent s'effectuer simultanément. Pour lever toute ambiguïté quant à cette opération, on en donne dans le tableau 3.8 une définition inductive, calquée sur celle des $\mathbf{EB}(x_1, \dots, x_n)$.

expression	$(x_1 \leftarrow X_1, \dots, x_n \leftarrow X_n)$ expression
0	0
1	1
x_i	X_i
$(F \vee G)$	$(x_1 \leftarrow X_1, \dots, x_n \leftarrow X_n) F \vee (x_1 \leftarrow X_1, \dots, x_n \leftarrow X_n) G$
$(F \wedge G)$	$(x_1 \leftarrow X_1, \dots, x_n \leftarrow X_n) F \wedge (x_1 \leftarrow X_1, \dots, x_n \leftarrow X_n) G$
\bar{F}	$\overline{(x_1 \leftarrow X_1, \dots, x_n \leftarrow X_n) F}$

Tableau 3.8

3.1.39 Axiomes

Soient R, S des relations quelconques. Avec la notation (3.36), on a :

$$\langle R \text{ ou } S \rangle = \langle R \rangle \vee \langle S \rangle \quad (3.58)$$

$$\langle R \text{ et } S \rangle = \langle R \rangle \wedge \langle S \rangle \quad (3.59)$$

$$\langle \text{non } R \rangle = \overline{\langle R \rangle}. \quad (3.60)$$

Ces axiomes précisent le sens des mots “ou, et, non”.

3.1.40 Démonstration

Nous sommes en mesure maintenant de démontrer la proposition du paragraphe 3.1.12, formulée par (3.38). On procède selon le schéma du paragraphe 3.1.33, en appelant $R(F)$ la relation (3.38).

Si F est l'expression 0, le premier membre de (3.38) est $\langle \xi \in \emptyset \rangle = 0$, en vertu de (3.5) et (3.36); le second membre est 0 selon le tableau 3.8. Si F est l'expression 1, le premier membre est $\langle \xi \in \mathbf{B}_n \rangle = 1$ en vertu de (3.6), et le second est 1 (tab. 3.8). Si F est l'expression x_i ($i = 1, \dots, n$), le premier membre de (3.38) est égal à $\langle \xi_i = 1 \rangle$ en vertu de (3.7); or $\langle \xi_i = 1 \rangle = \xi_i$ puisque $\xi_i \in \mathbf{B}$, et ξ_i est précisément le second membre de (3.38) selon le tableau 3.8.

Supposons que (3.38) soit vraie pour deux expressions $F, G : \mathbf{EB}(x_1, \dots, x_n)$, et abrégeons les préfixes $(\sigma x_1, \dots, x_n), (x_1 \leftarrow \xi_1, \dots, x_n \leftarrow \xi_n)$ par σ et \leftarrow . On a

$$\begin{aligned} \langle \xi \in \sigma(F \vee G) \rangle &= \langle \xi \in \sigma F \cup \sigma G \rangle && \text{par (3.8)} \\ &= \langle \xi \in \sigma F \text{ ou } \xi \in \sigma G \rangle && \text{par (1.6)} \\ &= \langle \xi \in \sigma F \rangle \vee \langle \xi \in \sigma G \rangle && \text{par (3.58)} \\ &= (\leftarrow F) \vee (\leftarrow G) && \text{par hypothèse} \\ &= \leftarrow (F \vee G) && \text{selon tableau 3.8} \end{aligned}$$

Donc (3.38) est vraie pour $F \vee G$. On montre de même qu'elle est vraie pour $F \wedge G$, et enfin que si (3.38) est vraie pour F , elle est vraie pour \bar{F} .

3.2 ÉQUATIONS DE RÉCURRENCE BOOLÉENNES

3.2.1 Introduction

Le but de cette section est de montrer comment une certaine classe d'équations, les équations de récurrence booléennes, peut être utilisée pour effectuer la synthèse de machines binaires de type séquentiel (machines séquentielles, machines de Moore ou de Mealy), c'est-à-dire pour :

- mettre un cahier des charges sous la forme mathématique d'un système d'équations;
- transformer ce système d'équations jusqu'à ce qu'il ait la forme caractéristique de l'un des types de machines mentionnés ci-dessus.

L'exposé sera basé essentiellement sur des exemples.

3.2.2 Définitions

Une machine binaire de type quelconque dont les entrées sont désignées par x_1, \dots, x_m , les sorties par y_1, \dots, y_n , et le fonctionnement par M , est notée

$$M(x_1, \dots, x_m)(y_1, \dots, y_n).$$

Cette notation est équivalente au schéma de la figure 3.9.

Un *état d'entrée* de M est un vecteur $x = x_1 \times \dots \times x_m \in \mathbf{B}_m$. Un *état de sortie* de M est un vecteur $y = y_1 \times \dots \times y_n \in \mathbf{B}_n$. Une séquence d'entrée $x = x(1, p) = x(1) \dots x(p)$ de longueur p est donc une séquence d'états d'entrée $x(k) = x_1(k) \times \dots \times x_m(k)$ ($k = 1, \dots, p$), et de même pour une séquence de sortie.

Le fonctionnement de cette machine est une correspondance $M : X^+ \rightarrow Y^+$, avec $X = \mathbf{B}_m$ et $Y = \mathbf{B}_n$. Une séquence de sortie y correspond à une séquence d'entrée x , si $y \in M(x)$.

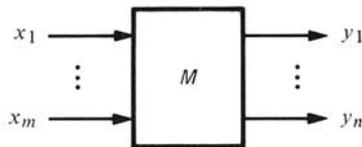


Fig. 3.9

3.2.3 Equations de récurrence des machines séquentielles

Pour une machine séquentielle, les mots entrée et sortie sont souvent remplacés par primaire et secondaire. Un *état total* d'une machine séquentielle binaire

$$M(x_1, \dots, x_m)(y_1, \dots, y_n)$$

est un couple $y \times x$ où y est un état secondaire, et x un état primaire. Un état total $y \times x$ est un élément de l'ensemble $\mathbf{B}_n \times \mathbf{B}_m = \mathbf{B}_{n+m}$.

Le fonctionnement M est défini par une table de transition (sect. 2.4) qui associe à chaque état total $y \times x$ un ensemble d'états secondaires noté $y \cdot x$. La correspondance M est définie par la relation

$$y(k+1) \in y(k) \cdot x(k). \quad (3.61)$$

Une séquence secondaire $y(1, p)$ correspond à une séquence primaire $x(1, p)$ si et seulement si (3.61) est vérifiée pour $k = 1, \dots, p - 1$.

Lorsque la machine est complètement spécifiée (sect. 2.5), l'ensemble $y(k) \cdot x(k)$ ne possède qu'un élément, et (3.61) peut s'écrire

$$y(k+1) = y(k) \cdot x(k). \quad (3.62)$$

On dit, dans ce cas, que l'état secondaire futur $y(k+1)$ est entièrement déterminé par l'état total présent $y(k) \times x(k)$. Dans le cas (3.61), l'état secondaire futur n'est que partiellement déterminé par l'état total présent.

Dans le cas (3.62), il existe pour chaque variable secondaire y_i une fonction booléenne $f_i : \mathbf{B}_{n+m} \rightarrow \mathbf{B}$ (§ 3.1.20) telle que

$$y_i(k+1) = f_i(y_1(k), \dots, y_n(k), x_1(k), \dots, x_m(k)). \quad (3.63)$$

La fonction booléenne f_i détermine la i -ème composante de l'état secondaire futur en fonction de l'état total présent. Nous abrégeons (3.63) par

$$y_i(k+1) = f_i(y(k), x(k)). \quad (3.64)$$

Autrement dit, le fonctionnement d'une machine séquentielle binaire complètement spécifiée $M(x_1, \dots, x_m)(y_1, \dots, y_n)$ peut être défini par un système d'équations de la forme

$$\left. \begin{array}{l} y_1(k+1) = f_1(y(k), x(k)) \\ \vdots \\ y_n(k+1) = f_n(y(k), x(k)). \end{array} \right\} \quad (3.65)$$

Pratiquement, les fonctions booléennes f_i seront représentées par des expressions booléennes sur les variables booléennes $x_1(k), \dots, x_m(k), y_1(k), \dots, y_n(k)$. Le système (3.65) sera donc un système d'équations booléennes sur les variables $x_1(k), \dots, x_m(k), y_1(k), \dots, y_n(k), y_1(k+1), \dots, y_n(k+1)$.

Un tel système d'équations est appelé un *système d'équations de récurrence booléennes d'ordre 1*. La notion de récurrence d'ordre p (p entier > 0) se précisera progressivement. Disons pour l'instant qu'un système d'équations de récurrence d'ordre p permet de déterminer le $(k+p)$ -ième élément $y(k+p)$ d'une séquence de sortie en fonction de $y(k, k+p-1)$ et $x(k, k+p)$.

3.2.4 Exemple

Le tableau 3.10 est la table de transition d'une machine séquentielle binaire $M(x_1, x_2)(y_1, y_2)$. Cette table se traduit par le système d'équations

$$\left. \begin{array}{l} y_1(k+1) = \overline{x_1(k)} \wedge \overline{x_2(k)} \\ y_2(k+1) = y_1(k) \wedge \overline{x_1(k)} \wedge x_2(k) \end{array} \right\} \quad (3.66)$$

3.2.5 Equations de récurrence des machines combinatoires

Il est clair que le fonctionnement d'une machine combinatoire (sect. 2.3) binaire $F(x_1, \dots, x_m)(y_1, \dots, y_n)$ complètement spécifiée peut être défini par un système

	$x_1 \times x_2$			
	0×0	0×1	1×1	1×0
0×0	1×0	0×0	0×0	0×0
0×1	1×0	0×0	0×0	0×0
1×1	1×0	0×1	0×0	0×0
1×0	1×0	0×1	0×0	0×0
$y_1 \times y_2$				

Tableau 3.10

d'équations de la forme

$$\left. \begin{aligned} y_1(k) &= f_1(x_1(k), \dots, x_m(k)) \\ \vdots \\ y_n(k) &= f_n(x_1(k), \dots, x_m(k)) \end{aligned} \right\} \quad (3.67)$$

où f_1, \dots, f_n sont des fonctions booléennes $\mathbf{B}_m \rightarrow \mathbf{B}$.

Nous dirons que le système (3.67) est un système d'équations de récurrence d'ordre 0.

3.2.6 Exemple

Le tableau 3.11 est la table de vérité d'une machine combinatoire binaire $F(x_1, x_2, y_1, y_2)(z)$ complètement spécifiée. Le fonctionnement de cette machine est donc défini par l'équation

$$z(k) = y_2(k) \wedge x_1(k) \wedge \overline{x_2(k)}. \quad (3.68)$$

3.2.7 Equations de récurrence des machines de Moore et de Mealy

Le fonctionnement d'une machine de Moore ou de Mealy (sect. 2.6) binaire peut être défini par le système d'équations formé par la juxtaposition des systèmes d'équations de sa composante séquentielle et de sa composante combinatoire.

Supposons par exemple que l'on assemble la machine séquentielle M (§ 3.2.4) et la machine combinatoire F (§ 3.2.6) selon le schéma de la figure 3.12. La machine résultante $R(x_1, x_2)(y_1, y_2, z)$ est une machine de Mealy. Son fonctionnement peut être décrit par le tableau 3.13, ou par le système d'équations

$$\left. \begin{aligned} y_1(k+1) &= \overline{x_1(k)} \wedge \overline{x_2(k)} \\ y_2(k+1) &= y_1(k) \wedge \overline{x_1(k)} \wedge x_2(k) \\ z(k) &= y_2(k) \wedge x_1(k) \wedge \overline{x_2(k)} \end{aligned} \right\} \quad (3.69)$$

formé par (3.66) et (3.68).

3.2.8 Cahier des charges : détecteur de séquences

Nous allons effectuer, dans les paragraphes qui suivent, la synthèse d'une machine satisfaisant le cahier des charges suivant (§ V.6.3.1).

		$x_1 \times x_2$			
		0×0	0×1	1×1	1×0
$y_1 \times y_2$	0×0	0	0	0	0
	0×1	0	0	0	1
	1×1	0	0	0	1
	1×0	0	0	0	0

Tableau 3.11

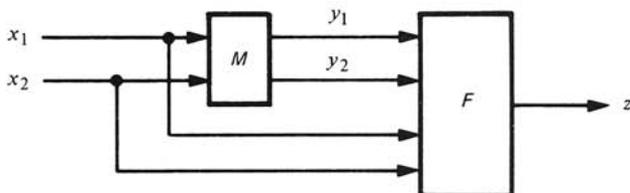


Fig. 3.12

		$x_1 \times x_2$			
		0×0	0×1	1×1	1×0
$y_1 \times y_2$	0×0	$1 \times 0 ; 0$	$0 \times 0 ; 0$	$0 \times 0 ; 0$	$0 \times 0 ; 0$
	0×1	$1 \times 0 ; 0$	$0 \times 0 ; 0$	$0 \times 0 ; 0$	$0 \times 0 ; 1$
	1×1	$1 \times 0 ; 0$	$0 \times 1 ; 0$	$0 \times 0 ; 0$	$0 \times 0 ; 1$
	1×0	$1 \times 0 ; 0$	$0 \times 1 ; 0$	$0 \times 0 ; 0$	$0 \times 0 ; 0$

Tableau 3.13

On veut reconnaître dans un texte français, analysé caractère par caractère, les mots terminés par *er*. Une tête de lecture code chacun des caractères en un vecteur $x_1 \times x_2 \in \mathbf{B}_2$ selon le tableau 3.14. Une machine $M(x_1, x_2)(z)$ doit signaler la fin de chaque mot détecté, au moyen de sa variable de sortie z .

		$x_1 \times x_2$	
lettre <i>e</i>	0×0	<i>e</i>	
lettre <i>r</i>	0×1	<i>r</i>	
autres lettres et tiret	1×1	<i>l</i>	
signes de ponctuation	1×0	<i>p</i>	

Tableau 3.14

3.2.9 Mise en forme du cahier des charges

En tant que description d'une machine $M(x_1, x_2)(z)$, le cahier des charges précédent est imprécis, ce qui est toujours le cas dans la pratique. Nous pouvons le préciser au moyen :

- d'une *équation de récurrence*
- de *conditions initiales*.

L'équation de récurrence s'écrit

$$z(k+2) = e(k) \wedge r(k+1) \wedge p(k+2) \quad (3.70)$$

avec

$$\left. \begin{aligned} e(k) &= \overline{x_1(k)} \wedge \overline{x_2(k)} \\ r(k) &= \overline{x_1(k)} \wedge x_2(k) \\ p(k) &= x_1(k) \wedge \overline{x_2(k)}. \end{aligned} \right\} \quad (3.71)$$

L'équation (3.70) est une équation de récurrence booléenne d'ordre 2. Etant donné une séquence d'entrée $x(1, n)$ de longueur $n > 2$, l'équation détermine la séquence $z(3, n)$. Elle ne détermine pas $z(1, 2)$. Pour cela nous ajoutons au cahier des charges les conditions initiales

$$\left. \begin{aligned} z(1) &= 0 \\ z(2) &= 0 \end{aligned} \right\} \quad (3.72)$$

Les relations (3.70) et (3.72) définissent une machine $M(x_1, x_2)(z)$ qui fait correspondre à toute séquence d'entrée x une séquence $z = M(x)$ conforme au cahier des charges. Un exemple est donné par les deux premières lignes du tableau 3.15.

	1	2	...	15											
$x(1, 15)$	<i>e</i>	<i>l</i>	<i>e</i>	<i>r</i>	<i>e</i>	<i>r</i>	<i>p</i>	<i>l</i>	<i>r</i>	<i>e</i>	<i>e</i>	<i>r</i>	<i>p</i>	<i>p</i>	<i>l</i>
$z(1, 15)$	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0
$y_1(1, 15)$	0	1	0	1	0	1	0	0	0	0	1	1	0	0	0
$y_2(1, 15)$	0	0	0	0	1	0	1	0	0	0	0	0	1	0	0

Tableau 3.15

3.2.10 Enoncé du problème de synthèse

La machine $M(x_1, x_2)(z)$ définie au paragraphe précédent n'est pas une machine de Mealy. Mais on espère qu'il existe une machine de Mealy $R(x_1, x_2)(y_1, \dots, y_n; z)$ avec n variables secondaires y_1, \dots, y_n , possédant un état secondaire q tel que la machine $R_q(x_1, x_2)(z)$ (cf. § 2.6.15) soit une réalisation (§ 2.5.6) de M .

En d'autres termes, on cherche un système d'équations de la forme

$$\left. \begin{aligned} y_1(k+1) &= f_1(y_1(k), \dots, y_n(k), x_1(k), x_2(k)) \\ &\vdots \\ y_n(k+1) &= f_n(y_1(k), \dots, y_n(k), x_1(k), x_2(k)) \\ z(k) &= g(y_1(k), \dots, y_n(k), x_1(k), x_2(k)) \end{aligned} \right\} \quad (3.73)$$

c'est-à-dire définissant une machine de Mealy (§ 3.2.7), et des conditions initiales

$$y_1(1) = q_1 ; \dots ; y_n(1) = q_n \quad (q_i \in \mathbf{B}) \quad (3.74)$$

de telle sorte que (3.73) et (3.74) impliquent (3.70) et (3.72).

La démarche consiste à introduire des variables secondaires y_1, y_2, \dots de façon à abaisser l'ordre de (3.70), jusqu'à ce qu'on obtienne pour z une équation d'ordre 0 et pour y_1, y_2, \dots des équations d'ordre 1.

On traitera séparément les équations de récurrence et les conditions initiales.

3.2.11 Abaissement de l'ordre de récurrence

En considérant l'équation (3.70), on pose

$$y_1(k+1) = e(k). \quad (3.75)$$

L'équation (3.70) s'écrit alors

$$z(k+2) = y_1(k+1) \wedge r(k+1) \wedge p(k+2).$$

En remplaçant le paramètre k par $k-1$, il vient

$$z(k+1) = y_1(k) \wedge r(k) \wedge p(k+1). \quad (3.76)$$

L'équation (3.70) est remplacée maintenant par (3.75) et (3.76). En considérant (3.76), on pose

$$y_2(k+1) = y_1(k) \wedge r(k). \quad (3.77)$$

Il vient

$$z(k+1) = y_2(k+1) \wedge p(k+1)$$

et en changeant k en $k-1$:

$$z(k) = y_2(k) \wedge p(k). \quad (3.78)$$

L'équation (3.70) est remplacée maintenant par le système

$$\left. \begin{aligned} y_1(k+1) &= e(k) = \overline{x_1(k)} \wedge \overline{x_2(k)} \\ y_2(k+1) &= y_1(k) \wedge r(k) = y_1(k) \wedge \overline{x_1(k)} \wedge x_2(k) \\ z(k) &= y_2(k) \wedge p(k) = y_2(k) \wedge x_1(k) \wedge \overline{x_2(k)} \end{aligned} \right\} \quad (3.79)$$

Ce système est celui d'une machine de Mealy $R(x_1, x_2)(y_1, y_2, z)$. On reconnaît la machine du paragraphe 3.2.7. Le fonctionnement est illustré par le tableau 3.15.

3.2.12 Conditions initiales

La première des conditions initiales (3.72) s'écrit selon (3.79) :

$$z(1) = y_2(1) \wedge p(1) = 0.$$

Pour que cette condition soit vérifiée quelle que soit la séquence d'entrée x , c'est-à-dire quel que soit $p(1)$, il faut et il suffit que $y_2(1) = 0$. La condition initiale $z(2) = 0$ s'écrit selon (3.79) :

$$z(2) = y_2(2) \wedge p(2) = y_1(1) \wedge r(1) \wedge p(2) = 0.$$

Pour qu'elle soit vérifiée quels que soient $r(1)$ et $p(2)$, il faut et il suffit que $y_1(1) = 0$. Les conditions initiales (3.74) cherchées sont donc

$$y_1(1) = q_1 = 0 ; y_2(1) = q_2 = 0. \tag{3.80}$$

Le lecteur vérifiera facilement que (3.79) et (3.80) impliquent (3.70) et (3.72). Le problème est résolu. La machine de Mealy (3.79) peut être réalisée par le schéma de la figure 3.16, qui comporte deux éléments de délai unité (§ 2.5.11) et des éléments combinatoires dont le symbolisme est supposé connu (vol. V).

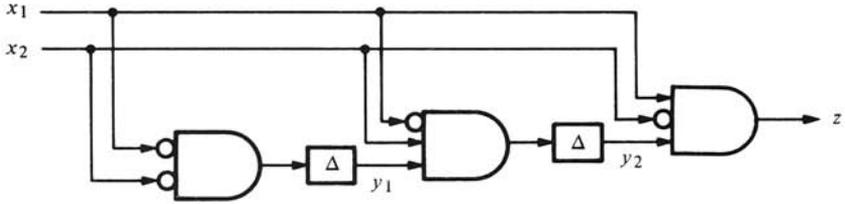


Fig. 3.16

3.2.13 Comparaison de méthodes de synthèse

La table de transition (ou table d'états) de la machine (3.79) est le tableau 3.13. En désignant ses états secondaires par q_0, \dots, q_3 selon le code $q_0 = 0 \times 0, q_1 = 0 \times 1, q_2 = 1 \times 0, q_3 = 1 \times 1$, on peut donner le graphe de transition de la figure 3.17. L'état q_0 fixé par les conditions initiales (3.80) est encadré.

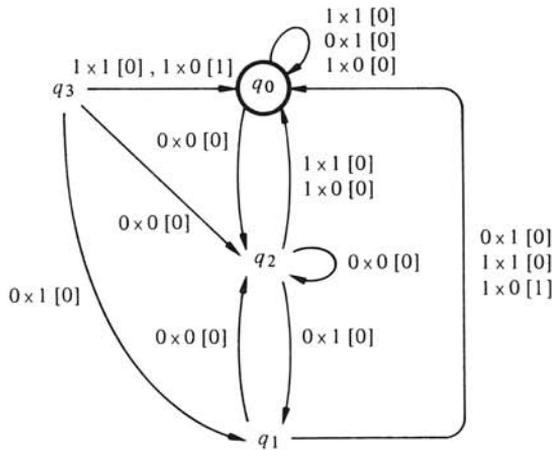


Fig. 3.17

En "oubliant" les vecteurs binaires représentés par q_0, q_1, q_2, q_3 , on peut considérer ce graphe de transition comme celui d'une machine de Mealy $R[X, Y, Z]$ (notation du paragraphe 2.6.8), avec les alphabets $X = \mathbf{B}_2, Y = \{q_0, q_1, q_2, q_3\}, Z = \mathbf{B}$.

La méthode de synthèse exposée dans le volume V (sect. 6.3) consiste à construire d'abord un graphe tel que celui de la figure 3.17, puis à coder l'alphabet $\{q_0, q_1, q_2, q_3\}$ au moyen de l'alphabet \mathbf{B}_2 . Si le graphe et le code choisis sont les mêmes que ci-dessus, on en tire la table de transition (tab. 3.13) et enfin les équations (3.79).

Nous ne saurions recommander de façon générale l'emploi de l'une ou l'autre méthode de synthèse, et cette remarque est valable pour toutes les méthodes de conception. Chacune d'elles présente des avantages ou des inconvénients selon le cahier des charges donné. Le savoir-faire consiste à connaître un éventail de méthodes et à choisir la plus commode dans chaque cas particulier.

3.2.14 Exercice

Le cahier des charges suivant est tiré textuellement du volume V (§ 6.3.12). On demande de le mettre sous la forme d'une équation de récurrence en z , x_1, x_2 , de préciser les conditions initiales, et de le traiter selon la méthode exposée ci-dessus (§ 3.2.9 à 3.2.12).

Un système séquentiel possède deux entrées x_1, x_2 et une sortie z définie de la façon suivante : si $x_2 = 0$, la valeur de z à l'instant présent est égale à celle de x_1 observée à l'instant précédent; si $x_2 = 1$, la valeur de z à l'instant présent est égale à celle de x_1 observée deux instants auparavant.

3.2.15 Exemple

Soit $M(x)(z)$ la machine binaire définie par l'équation de récurrence

$$z(k+3) = x(k)z(k) \vee \overline{z(k+2)} \quad (3.81)$$

et les conditions initiales

$$z(1) = z(2) = z(3) = 0. \quad (3.82)$$

La conjonction \wedge est notée comme un produit logique. En remarquant que la valeur de z à un instant quelconque est indépendante de la valeur de x au même instant, on se propose de trouver une machine séquentielle (et non une machine de Mealy) $R(x)(y_1, \dots, y_n, z)$ ainsi qu'un état secondaire initial q de cette machine, tel que la machine R_q soit une réalisation de M . La première idée qui vient à l'esprit est de poser

$$\left. \begin{aligned} y_1(k+1) &= x(k)z(k) \\ y_2(k+1) &= y_1(k) \\ z(k+1) &= y_2(k) \vee \overline{z(k)}. \end{aligned} \right\} \quad (3.83)$$

Le système (3.83) définit une machine séquentielle $R(x)(y_1, y_2, z)$ (équations de récurrence d'ordre 1). Il implique

$$\begin{aligned} x(k)z(k) \vee \overline{z(k+2)} &= y_1(k+1) \vee \overline{z(k+2)} \\ &= y_2(k+2) \vee \overline{z(k+2)} \\ &= z(k+3). \end{aligned}$$

Donc l'équation (3.81) est satisfaite par le système (3.83). Malheureusement les conditions initiales (3.82) ne peuvent pas être satisfaites, car si $z(1) = 0$, la troisième équation (3.83) implique $z(2) = 1$. Pour remédier à cet inconvénient, il suffit de multiplier le membre de droite de la troisième équation (3.83) par un terme $y_4(k)$ tel que $y_4(1) = y_4(2) = 0$ et $y_4(k+2) = 1$. On remplace donc le système (3.83) par le système

$$\left. \begin{aligned} y_1(k+1) &= x(k) z(k) \\ y_2(k+1) &= y_1(k) \\ y_3(k+1) &= 1 \\ y_4(k+1) &= y_3(k) \\ z(k+1) &= y_4(k)(y_2(k) \vee \overline{z(k)}). \end{aligned} \right\} \quad (3.84)$$

Le système (3.84) définit une machine séquentielle $R(x)(y_1, \dots, y_4, z)$. Il implique

$$y_4(k+2) = y_3(k+1) = 1 \quad (3.85)$$

et

$$\begin{aligned} x(k) z(k) \vee \overline{z(k+2)} &= y_1(k+1) \vee \overline{z(k+2)} \\ &= y_2(k+2) \vee \overline{z(k+2)} \\ &= y_4(k+2)(y_2(k+2) \vee \overline{z(k+2)}) \text{ par (3.85)} \\ &= z(k+3). \end{aligned}$$

L'équation de récurrence (3.81) est donc vérifiée par le système (3.84). Cherchons un état secondaire initial $q = y_1(1) \times \dots \times y_4(1) \times z(1)$ tel que les conditions (3.82) soient vérifiées. Il faut évidemment poser $z(1) = 0$. Il vient $z(2) = y_4(1)$ par (3.84). On pose donc $y_4(1) = 0$, et il vient $z(2) = 0$. Ceci entraîne $z(3) = y_4(2) = y_3(1)$ par (3.84). On pose donc $y_3(1) = 0$. Pour que les conditions (3.82) soient vérifiées, il suffit donc de fixer $y_3(1) = y_4(1) = z(1) = 0$; $y_1(1)$ et $y_2(1)$ peuvent être choisis arbitrairement. La machine séquentielle (3.84) peut être réalisée par le schéma de la figure 3.18.

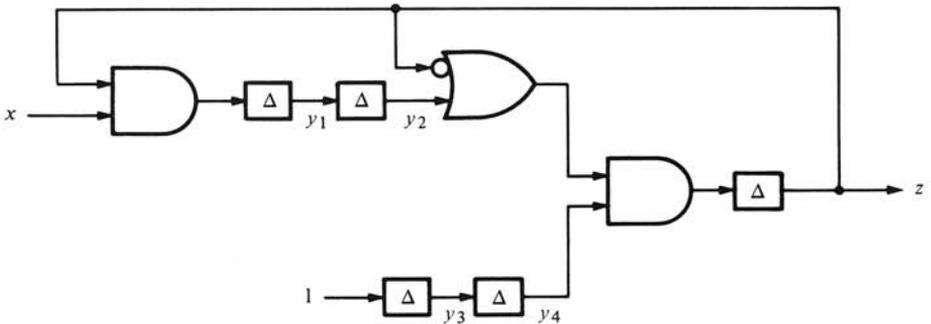


Fig. 3.18

3.2.16 Exercice

Soit $\zeta(1, p) \in \mathbf{B}^+$ une séquence binaire quelconque, de longueur p . Trouver une machine de Mealy binaire $R(s)(y_1, \dots, y_n; z)$ et un état secondaire initial $q = y_1(1) \times \dots \times y_n(1)$, tel que l'on ait pour toute séquence d'entrée s de longueur $> p$:

$$\begin{aligned} z(k+p) &= s(k+p) \quad (k = 1, 2, \dots) \\ z(1, p) &= \zeta(1, p). \end{aligned}$$

On s'inspirera du schéma de la figure 3.19, où F est une machine combinatoire. On pourra trouver d'autres solutions.

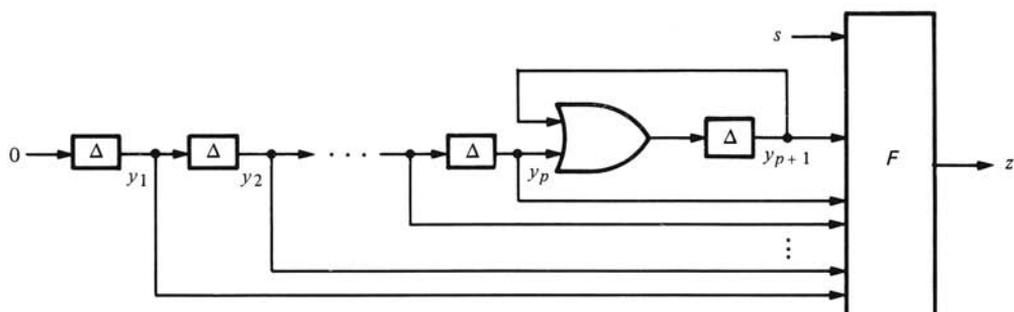


Fig. 3.19

3.2.17 Exercice

Soient $\zeta(1, p) \in \mathbf{B}^+$ une séquence binaire quelconque de longueur p , et

$$z(k+p) = f(x(k), \dots, x(k+p), z(k), \dots, z(k+p-1))$$

une équation de récurrence booléenne d'ordre p en x et z . Montrer qu'il existe une machine de Mealy $M(x)(y_1, \dots, y_n; z)$ satisfaisant cette équation de récurrence, et possédant un état secondaire initial q tel que l'on ait $z(1, p) = \zeta(1, p)$. On s'inspirera du schéma de la figure 3.20, où G est une machine combinatoire, et R une machine du type étudié dans l'exercice précédent. Les variables secondaires y_1, \dots, y_n seront les variables $u_1, \dots, u_p, w_1, \dots, w_p$ du schéma, plus les variables secondaires de R (fig. 3.19).

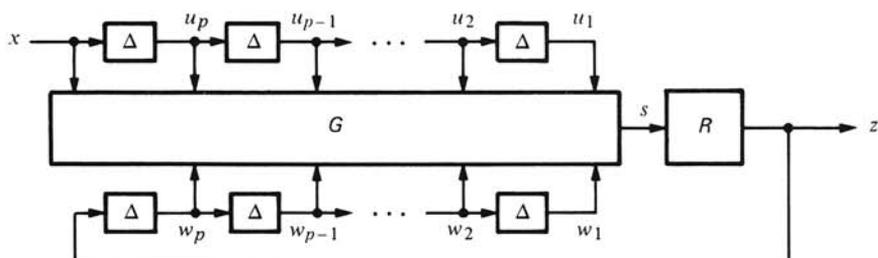


Fig. 3.20

3.2.18 Cahier des charges : convertisseur de code

Nous allons traiter dans les paragraphes qui suivent, par la méthode classique (sect. V.6.3) et par la méthode algébrique des équations de récurrence, le cahier des charges relativement complexe exposé ci-dessous.

On désire convertir, au moyen d'une machine de type séquentiel, toute séquence binaire $x(1, n) \in \mathbf{B}^+$ en une séquence ternaire $u(1, n)$ sur l'alphabet $\{-U, 0, U\}$, où U représente un nombre réel > 0 (tension électrique), de telle façon

- que la somme (algébrique usuelle) des $u(i)$ ($i = 1, \dots, n$) ne dépasse pas U en valeur absolue;
- que la séquence u ne comporte jamais quatre éléments consécutifs $u(i) u(i+1) u(i+2) u(i+3)$ identiques;
- que deux séquences binaires x, x' distinctes soient converties en des séquences ternaires u, u' distinctes (possibilité de décodage).

Le problème pratique dont provient ce cahier des charges est de transmettre des messages binaires sur une ligne électrique simple de façon que la tension moyenne soit toujours voisine de 0 et que les variations de tension soient suffisamment fréquentes, car elles sont utilisées comme signal de synchronisation des horloges de l'émetteur et du récepteur. On adopte à cet effet les règles de conversion suivantes.

- Si la séquence x ne comporte pas plus de trois zéros consécutifs, alors

$$x(i) = 0 \Rightarrow u(i) = 0$$

$$x(i) = 1 \Rightarrow u(i) = \pm U \text{ alternativement (les signes des } u(i) \text{ non nuls sont alternés).} \quad (3.86)$$

- Une sous-séquence 0000 de x (quatre zéros consécutifs) est convertie en une sous-séquence de u de la forme P00V qui se calcule comme suit : si le nombre des $u(i)$ non nuls qui précèdent P dans la séquence u est pair, alors $P = 0$; si ce nombre est impair, alors $P = \pm U$ en alternance avec le dernier $u(i)$ non nul; enfin $V = \pm U$ en violation d'alternance (le signe de V est le même que celui de P si $P \neq 0$, ou le même que celui du dernier $u(i)$ non nul précédant P si $P = 0$).

$$(3.87)$$

Un exemple de conversion est donné ci-dessous (3.88). On écrit \tilde{U} pour $-U$. L'exemple indique comment la séquence x doit être découpée en tranches auxquelles on applique (3.86) ou (3.87).

$$\begin{array}{l}
 x(1,29) = \left| \begin{array}{c|c|c|c|c|c|c|c}
 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0
 \end{array} \right| \\
 u(1,29) = \left| \begin{array}{c|c|c|c|c|c|c|c}
 0 & U & 0 & \tilde{U} & U & \tilde{U} & 0 & 0 & 0 & \tilde{U} & U & 0 & 0 & U & 0 & 0 & \tilde{U} & 0 & 0 & 0 & \tilde{U} & U & \tilde{U} & U & 0 & 0 & U & \tilde{U} & 0
 \end{array} \right| \quad (3.88) \\
 \underbrace{\hspace{10em}}_{(3.86)} \quad \underbrace{\hspace{10em}}_{(3.87)} \quad \underbrace{\hspace{10em}}_{(3.87)} \quad \underbrace{\hspace{10em}}_{(3.86)} \quad \underbrace{\hspace{10em}}_{(3.87)} \quad \underbrace{\hspace{10em}}_{(3.86)} \quad \underbrace{\hspace{10em}}_{(3.87)} \quad \underbrace{\hspace{10em}}_{(3.86)}
 \end{array}$$

3.2.19 Modification du cahier des charges

Il est clair que la machine $M : \mathbf{B}^+ \rightarrow \{-U, 0, U\}^+$ définie par le cahier des charges précédent ne saurait être une machine de Mealy ou de Moore, ou une machine dérivée (§ 2.6.15) de l'une ou l'autre. En effet ces machines sont du type *temps réel*, en ce sens qu'un élément $z(k)$ d'une séquence de sortie z ne dépend pas des éléments $x(k+1)$, $x(k+2)$, ... de la séquence d'entrée x à laquelle elle correspond. Ce n'est pas le cas de la machine M ci-dessus puisque $u(k)$ dépend en général de $x(k+1)$, $x(k+2)$, $x(k+3)$.

Pour obtenir une réalisation en temps réel, il faut donc modifier le cahier des charges, en décalant à droite la séquence de sortie u de trois unités, en introduisant trois éléments arbitraires à sa gauche (qui seront ignorés par le récepteur), et en supprimant ses trois derniers éléments (toute machine doit conserver la longueur des séquences). On définit ainsi une nouvelle machine en temps réel $R : \mathbf{B}^+ \rightarrow \{-U, 0, U\}^+$ pour laquelle l'exemple (3.88) devient :

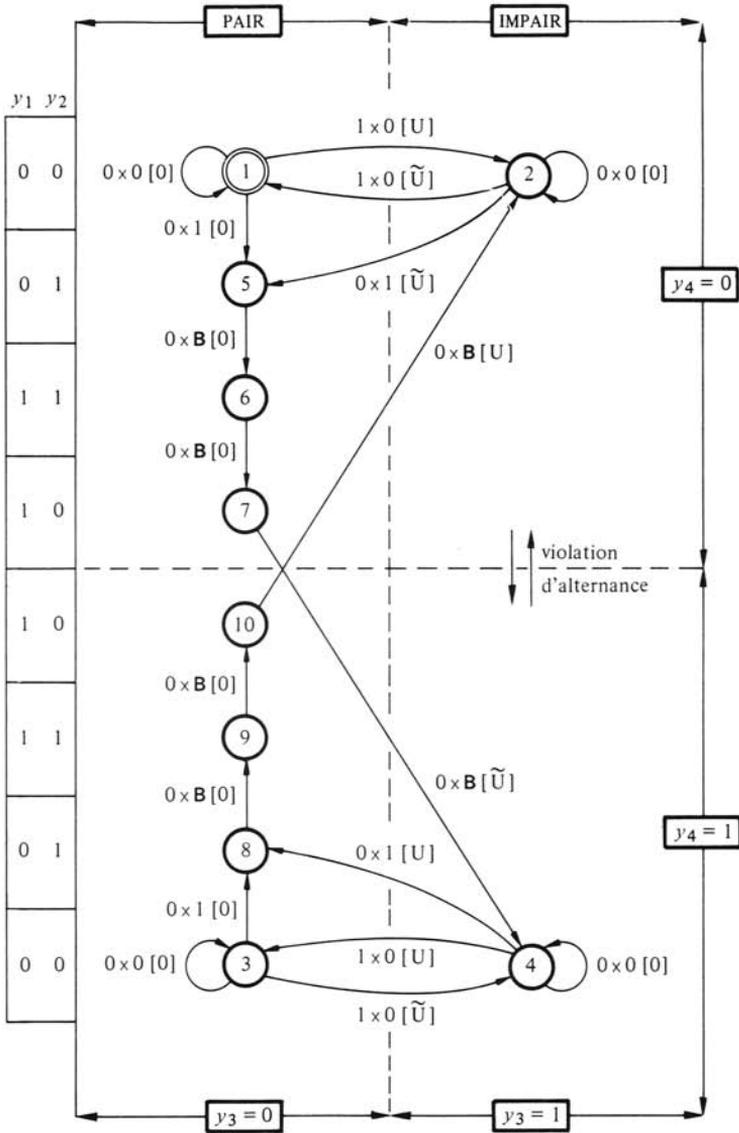


Fig. 3.22

sé n'est pas choisi au hasard, mais selon un critère mentionné ci-dessous et étudié de façon approfondie au chapitre 6.

Pour obtenir une machine binaire, il faut encore coder l'alphabet ternaire $\{-U, 0, U\}$. On choisit le code *signe et valeur absolue* $s \times v \in \mathbf{B}_2$ selon le tableau 3.23. Le nombre 0 peut être codé arbitrairement $+0$ ou -0 .

La figure 3.22 et le tableau 3.23 définissent une machine de Mealy $S(a_1, a_2)(y_1, y_2, y_3, y_4; s, v)$ incomplètement spécifiée. Sa table de transition est le tableau 3.24. En remplaçant chaque tiret par 0 ou 1, de façon appropriée (technique de

u	$s \times v$
-U	1×1
U	0×1
0	$\mathbf{B} \times 0$

Tableau 3.23

Karnaugh), on peut obtenir les équations de récurrence ci-dessous (3.90). On adopte ici la notation concise du volume V, qui consiste à écrire pour les variables secondaires $y^+ = f(y, x)$ au lieu de $y(k+1) = f(y(k), x(k))$, et pour les variables tertiaires, $z = g(y, x)$ au lieu de $z(k) = g(y(k), x(k))$. Les équations (3.90) définissent une réalisation complètement spécifiée de S (§ 2.5.6).

$$\left. \begin{aligned}
 y_1^+ &= y_2 \\
 y_2^+ &= \bar{y}_1 y_2 \vee \bar{y}_1 a_2 \\
 y_3^+ &= y_1 \bar{y}_2 \vee \bar{a}_1 \bar{a}_2 y_3 \vee a_1 \bar{a}_2 \bar{y}_3 \\
 y_4^+ &= \bar{y}_4 y_1 \bar{y}_2 \vee y_4 \bar{y}_1 \bar{y}_2 \\
 s &= \bar{y}_4 \bar{a}_1 \vee y_3 \bar{y}_4 a_1 \vee \bar{y}_3 y_4 a_1 \\
 v &= y_1 \bar{y}_2 \vee a_1 \bar{a}_2 \vee y_3 a_2
 \end{aligned} \right\} \quad (3.90)$$

On remarque que les variables y_1, y_2 ne dépendent pas de y_3 et y_4 . C'est en vue de cette indépendance qu'a été choisi le codage des sommets 1 à 10 dans la figure 3.22. Ce sujet sera traité au chapitre 6.

Les équations de la machine complète (fig. 3.21) sont formées de (3.90) et des équations

$$\left. \begin{aligned}
 w_1^+ &= x \\
 w_2^+ &= w_1 \\
 a_1^+ &= w_2 \\
 a_2 &= \overline{x \vee w_1 \vee w_2 \vee a_1} .
 \end{aligned} \right\} \quad (3.91)$$

En éliminant a_2 entre (3.90) et (3.91), on obtient un système d'équations ayant une variable primaire (x), sept variables secondaires ($w_1, w_2, a_1, y_1, y_2, y_3, y_4$), et deux variables tertiaires (s, v). L'état secondaire initial fixé précédemment est (1, 1, 0, 0, 0, 0, 0).

3.2.21 Méthode algébrique

On peut traiter le même problème de façon plus élégante en formalisant algébriquement le cahier des charges (3.89) par des équations de récurrence. La méthode est illustrée par la figure 3.25 qui reprend l'exemple (3.89). Les séquences binaires sont représentées par des chronogrammes.

Les variables v et s sont les variables valeur absolue et signe de u . La séquence v est déterminée immédiatement par la séquence u . La séquence s n'est pas déterminée

		a_2				a_1			
		0 0		0 1		1 1		1 0	
		$y_{1,2,3,4}$	$y_{1,2,3,4}^+$	s	y				
y_1	y_2	1	0 0 0 0	0 0 0 0	- 0	0 1 0 0	- 0	- - - -	0 0 1 0
		5	0 1 0 0	1 1 0 0	- 0	1 1 0 0	- 0	- - - -	- - - -
		6	1 1 0 0	1 0 0 0	- 0	1 0 0 0	- 0	- - - -	- - - -
		7	1 0 0 0	0 0 1 1	1 1	0 0 1 1	1 1	- - - -	- - - -
		2	0 0 1 0	0 0 1 0	- 0	0 1 0 0	1 1	- - - -	0 0 0 0
			0 1 1 0	- - - -	- -	- - - -	- -	- - - -	- - - -
			1 1 1 0	- - - -	- -	- - - -	- -	- - - -	- - - -
			1 0 1 0	- - - -	- -	- - - -	- -	- - - -	- - - -
		4	0 0 1 1	0 0 1 1	- 0	0 1 0 1	0 1	- - - -	0 0 0 1
			0 1 1 1	- - - -	- -	- - - -	- -	- - - -	- - - -
			1 1 1 1	- - - -	- -	- - - -	- -	- - - -	- - - -
			1 0 1 1	- - - -	- -	- - - -	- -	- - - -	- - - -
		3	0 0 0 1	0 0 0 1	- 0	0 1 0 1	- 0	- - - -	0 0 1 1
		8	0 1 0 1	1 1 0 1	- 0	1 1 0 1	- 0	- - - -	- - - -
		9	1 1 0 1	1 0 0 1	- 0	1 0 0 1	- 0	- - - -	- - - -
		10	1 0 0 1	0 0 1 0	0 1	0 0 1 0	0 1	- - - -	- - - -

Tableau 3.24

On peut écrire maintenant une équation de récurrence pour la variable v , en observant ceci : $v(k) = 1$ aux instants k pour lesquels l'une (au moins) des conditions suivantes est vérifiée :

- $x(k-3) = 1$
- $y(k) = 1$ et $p(k) = 1$ (instants P impair)
- $y(k-3) = 1$ (instants V).

Ceci s'exprime par l'équation

$$v(k+3) = x(k) \vee y(k) \vee (y(k+3) \wedge p(k+3)). \quad (3.96)$$

Pour la variable s , nous posons l'équation

$$s(k+1) = s(k) \oplus v(k) \oplus y(k). \quad (3.97)$$

Elle signifie que s change de valeur entre k et $k+1$, chaque fois que l'une et l'une seulement des conditions $v(k) = 1$, $y(k) = 1$ est vérifiée. Nous rappelons les formules

$$F \oplus 0 = F ; F \oplus 1 = \bar{F}. \quad (3.98)$$

Les équations (3.92), (3.95), (3.96), (3.97) constituent un système d'équations de récurrence complet en x, y, p, v, s . Il ne nous reste qu'à abaisser son ordre. Un peu d'habileté permet de le faire en n'introduisant qu'un nombre minimum de variables secondaires. L'équation (3.92) peut se transformer par De Morgan (3.22) en

$$y(k+3) = \overline{x(k) \vee y(k)} \wedge \overline{x(k+1) \vee y(k+1)} \wedge \overline{x(k+2) \vee y(k+2)} \wedge \overline{x(k+3)}. \quad (3.99)$$

Posons :

$$\left. \begin{aligned} w_1(k+1) &= \overline{x(k) \vee y(k)} \\ w_2(k+1) &= w_1(k) \\ w_3(k+1) &= w_2(k) \end{aligned} \right\} \quad (3.100)$$

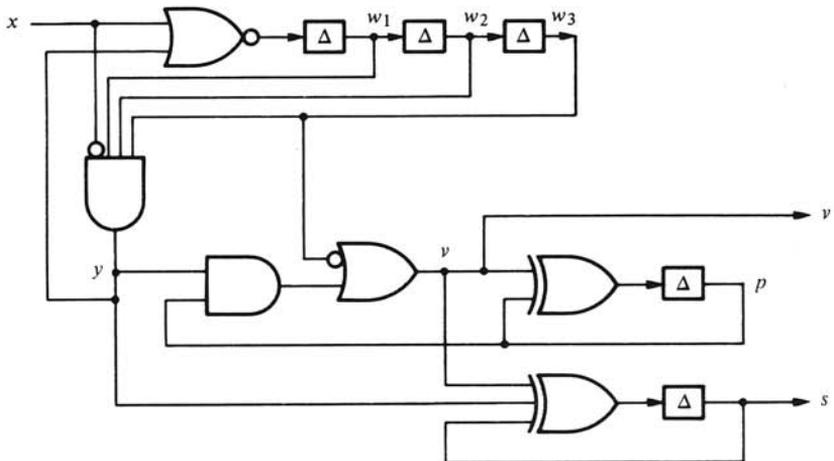


Fig. 3.26

et portons ceci dans (3.99). Il vient :

$$y(k+3) = w_3(k+3) \wedge w_2(k+3) \wedge w_1(k+3) \wedge \overline{x(k+3)}$$

ce que nous pouvons écrire sous la forme

$$y(k) = w_3(k) \wedge w_2(k) \wedge w_1(k) \wedge \overline{x(k)}. \quad (3.101)$$

L'équation (3.96) s'écrit au moyen de (3.100) :

$$v(k+3) = \overline{w_3(k+3)} \vee (y(k+3) \wedge p(k+3))$$

soit

$$v(k) = \overline{w_3(k)} \vee (y(k) \wedge p(k)). \quad (3.102)$$

Rassemblons les équations obtenues (3.95), (3.97), (3.100), (3.101), (3.102) :

$$\left. \begin{aligned} w_1(k+1) &= \overline{x(k)} \vee y(k) \\ w_2(k+1) &= w_1(k) \\ w_3(k+1) &= w_2(k) \\ p(k+1) &= p(k) \oplus v(k) \\ s(k+1) &= s(k) \oplus v(k) \oplus y(k) \\ y(k) &= w_3(k) \wedge w_2(k) \wedge w_1(k) \wedge \overline{x(k)} \\ v(k) &= \overline{w_3(k)} \vee (y(k) \wedge p(k)). \end{aligned} \right\} \quad (3.103)$$

Ces équations sont réalisées par le logigramme de la figure 3.26.

Ce schéma représente une machine de Mealy $M(x)(w_1, w_2, w_3, p, s; s, v)$ à cinq variables secondaires w_1, w_2, w_3, p, s et deux variables tertiaires s, v (s est à la fois secondaire et tertiaire; en toute rigueur, il faudrait introduire une variable tertiaire s' avec l'équation triviale $s'(k) = s(k)$).

Pour se conformer au cahier des charges illustré par la figure 3.25, il faut fixer un état secondaire initial $(w_1(1), w_2(1), w_3(1), p(1), s(1))$ de telle sorte que l'on obtienne par les équations (3.103) :

- $y(1) = y(2) = y(3) = 0$
(il ne faut pas empêcher $y(4) = 1$ au cas où $x(1,4) = 0000$);
- $s(4) = 0$
(le signe correspondant à $x(1)$, c'est-à-dire le signe de $u(4)$, est +);
- $p(4) = 0$.

Le tableau 3.27 montre que l'on peut choisir

$$(w_1(1), w_2(1), w_3(1), p(1), s(1)) = (0, 0, 0, 1, 1).$$

Les colonnes $k = 2, 3, 4$ sont calculées de proche en proche par (3.103).

3.2.22 Conclusions

Comme tout autre mode de représentation d'une machine, les équations de récurrence booléennes demandent un certain entraînement pour être employées avec succès. Dans l'exemple qui précède, ni le graphe de transition (fig. 3.22), ni le système d'équa-

k	1	2	3	4
$w_1(k)$	0	$\overline{x(1)}$	$\overline{x(2)}$	$\overline{x(3)}$
$w_2(k)$	0	0	$\overline{x(1)}$	$\overline{x(2)}$
$w_3(k)$	0	0	0	$\overline{x(1)}$
$p(k)$	1	0	1	0
$s(k)$	1	0	1	0
$y(k)$	0	0	0	$\overline{x(1)} \wedge \overline{x(2)} \wedge \overline{x(3)} \wedge \overline{x(4)}$
$v(k)$	1	1	1	$x(1)$

Tableau 3.27

tions (3.103) ne sont faciles à établir. Les deux solutions demandent une certaine habileté. Dans cet exemple, la seconde méthode conduit à un système d'équations plus simple que la première : comparer (3.103) avec (3.90), (3.91). Pour obtenir par la première méthode une solution aussi simple que par la seconde, il faudrait s'abstenir de décomposer a priori (fig. 3.21) la machine définie par le cahier des charges, et traduire celui-ci par un graphe de transition global, ayant au plus 32 sommets (5 variables secondaires). C'est là un problème très difficile. Toutefois, comme nous l'avons déjà suggéré au paragraphe 3.2.13, il faut se garder de conclure à la supériorité générale d'une méthode, d'après l'étude de quelques exemples.

Le lecteur aura remarqué l'absence dans cette section d'une théorie mathématique rigoureuse des équations de récurrence. Notamment, nous n'avons pas donné de méthode systématique pour abaisser l'ordre d'un système d'équations de récurrence, ni de théorème sur le nombre de variables secondaires qu'il suffit d'introduire à cet effet, en tenant compte de conditions initiales. Le paragraphe 3.2.17 suggère toutefois une réponse à ces questions. On trouvera dans [5] (chap. 6) une étude plus approfondie de ce sujet, et dans [12] (chap. 5) une généralisation du concept d'équation de récurrence, fournissant un langage plus puissant pour la formalisation de cahiers des charges. Nous nous sommes bornés à présenter les équations de récurrence comme un mode de description de machines, et à illustrer leur traitement par des exemples.

Enfin, nous n'avons considéré que des équations de récurrence définissant des machines complètement spécifiées. L'équation de récurrence d'ordre p la plus générale en x et z , est de la forme

$$f(x(k), \dots, x(k+p), z(k), \dots, z(k+p)) = 1$$

où f est une fonction booléenne (cf. § 3.1.30). Nous n'avons envisagé que des cas où cette équation peut se résoudre par rapport à $z(k+p)$, c'est-à-dire peut se mettre sous la forme

$$z(k+p) = g(x(k), \dots, x(k+p), z(k), \dots, z(k+p-1)).$$

3.3 GRAPHES DE RÉCURRENCE BOOLÉENS

3.3.1 Introduction

On étudie dans cette section un type particulier d'équations de récurrence booléennes, qu'il est commode, en vue des applications, de noter sous la forme de graphes. Il est toujours possible de formaliser le cahier des charges d'une machine séquentielle au

moyen d'un tel graphe, en introduisant suffisamment de variables secondaires. Ce mode de représentation se révèle particulièrement commode pour la conception des systèmes logiques de commande de processus.

Les paragraphes 3.3.2 à 3.3.10 apportent quelques compléments préalables sur les systèmes d'équations booléennes, et sur une classe particulière de fonctions booléennes.

3.3.2 Systèmes d'équations booléennes

Soit

$$\left. \begin{array}{l} F_1 = G_1 \\ \vdots \\ F_k = G_k \end{array} \right\} \quad (3.104)$$

Un système d'équations booléennes (§ 3.1.24) sur les variables $x_1, \dots, x_n : \mathbf{B}$. Une *solution dans \mathbf{B}_n* du système (3.104) selon l'ordre x_1, \dots, x_n des variables, est un vecteur $\xi \in \mathbf{B}_n$ qui est solution de chacune des équations du système selon cet ordre. L'ensemble des solutions dans \mathbf{B}_n du système (3.104) selon l'ordre x_1, \dots, x_n est noté

$$(\sigma x_1, \dots, x_n)(F_1 = G_1, \dots, F_k = G_k)$$

ou, de façon abrégée $(\sigma)(F_1 = G_1, \dots, F_k = G_k)$. On a donc par définition :

$$(\sigma)(F_1 = G_1, \dots, F_k = G_k) = \bigcap_{i=1}^k (\sigma)(F_i = G_i) \quad (3.105)$$

La fonction booléenne $f : \mathbf{B}_n \rightarrow \mathbf{B}$ ayant pour noyau (§ 3.1.20) l'ensemble

$$(\sigma)(F_1 = G_1, \dots, F_k = G_k)$$

est notée $(\lambda x_1, \dots, x_n)(F_1 = G_1, \dots, F_k = G_k)$, ou de façon abrégée

$$(\lambda)(F_1 = G_1, \dots, F_k = G_k).$$

Elle est appelée *fonction caractéristique* du système d'équations (3.104) selon l'ordre x_1, \dots, x_n des variables.

3.3.3 Exemple

Soient $a, b, c : \mathbf{B}$. Dans le tableau 3.28, on construit la fonction f_7 , caractéristique du système d'équations $(\bar{a} = b \wedge c, c \vee \bar{a} = b \wedge a)$, selon l'ordre a, b, c . On cons-

	a	b	c	f_1	f_2	f_3	f_4	f_5	f_6	f_7	
1)	0	0	0	1	0	0	1	0	0	0	$f_1 = (\lambda a, b, c)(\bar{a})$
2)	0	0	1	1	0	0	1	0	0	0	$f_2 = (\lambda a, b, c)(b \wedge c)$
3)	0	1	0	1	0	0	1	0	0	0	$f_3 = (\lambda a, b, c)(\bar{a} = b \wedge c)$
4)	0	1	1	1	1	1	1	0	0	0	$f_4 = (\lambda a, b, c)(c \vee \bar{a})$
5)	1	1	0	0	0	1	0	1	0	0	$f_5 = (\lambda a, b, c)(b \wedge a)$
6)	1	1	1	0	1	0	1	1	1	0	$f_6 = (\lambda a, b, c)(c \vee \bar{a} = b \wedge a)$
7)	1	0	0	0	0	1	0	0	1	1	$f_7 = (\lambda a, b, c)(\bar{a} = b \wedge c, c \vee \bar{a} = b \wedge a)$
8)	1	0	1	0	0	1	1	0	0	0	

Tableau 3.28

truit d'abord les fonctions f_1, f_2 , de noyaux respectifs $\bar{a}, b \wedge c$. La fonction f_3 , caractéristique de l'équation $\bar{a} = b \wedge c$, a pour tout vecteur ξ la valeur $f_3(\xi) = \langle f_1(\xi) = f_2(\xi) \rangle$. L'ensemble $(\sigma a, b, c)(\bar{a} = b \wedge c)$, noyau de f_3 , se compose donc des vecteurs numérotés 4, 5, 7, 8. On détermine de façon analogue l'ensemble $(\sigma a, b, c)(c \vee \bar{a} = b \wedge a)$, noyau de f_6 , lequel se compose des vecteurs 6, 7. L'ensemble des solutions du système $(\bar{a} = b \wedge c, c \vee \bar{a} = b \wedge a)$ est selon (3.105) :

$$\begin{aligned} (\sigma)(\bar{a} = b \wedge c, c \vee \bar{a} = b \wedge a) &= ((\sigma)(\bar{a} = b \wedge c)) \cap ((\sigma)(c \vee \bar{a} = b \wedge a)) \\ &= \{(1, 0, 0)\}. \end{aligned}$$

3.3.4 Systèmes d'équations équivalents

Deux systèmes d'équations booléens $(F_1 = G_1, \dots, F_k = G_k)$ et $(F'_1 = G'_1, \dots, F'_p = G'_p)$ sur x_1, \dots, x_n sont dits *équivalents* lorsqu'ils admettent les mêmes solutions, c'est-à-dire lorsque

$$(\sigma x_1, \dots, x_n)(F_1 = G_1, \dots, F_k = G_k) = (\sigma x_1, \dots, x_n)(F'_1 = G'_1, \dots, F'_p = G'_p).$$

Tout système d'équations booléennes $(F_1 = G_1, \dots, F_k = G_k)$ est équivalent à une seule équation, de la forme $F = 1$. En effet, soit S l'ensemble

$$(\sigma x_1, \dots, x_n)(F_1 = G_1, \dots, F_k = G_k).$$

En vertu du paragraphe 3.1.19, il existe une expression booléenne F sur x_1, \dots, x_n qui représente ce sous-ensemble $S \subset \mathbf{B}_n$, et l'on a $S = (\sigma x_1, \dots, x_n)(F = 1)$, en vertu du paragraphe 3.1.28.

Par exemple (tab. 3.28), le système $(\bar{a} = b \wedge c, c \vee \bar{a} = b \wedge a)$ est équivalent à l'équation $a \wedge \bar{b} \wedge \bar{c} = 1$.

3.3.5 Proposition

Soient $F_1, \dots, F_k : \mathbf{EB}(x_1, \dots, x_n)$. Le système d'équations $(F_1 = 1, \dots, F_k = 1)$ est équivalent à l'équation $F_1 \wedge \dots \wedge F_k = 1$. Le système $(F_1 = 0, \dots, F_k = 0)$ est équivalent à l'équation $F_1 \vee \dots \vee F_k = 0$.

En effet, en abrégant $(\sigma x_1, \dots, x_n)$ par (σ) , on a

$$\begin{aligned} (\sigma)(F_1 = 1, \dots, F_k = 1) &= ((\sigma)(F_1 = 1)) \cap \dots \cap ((\sigma)(F_k = 1)) && \text{par (3.105)} \\ &= ((\sigma) F_1) \cap \dots \cap ((\sigma) F_k) && \text{par (3.48)} \\ &= (\sigma)(F_1 \wedge \dots \wedge F_k) && \text{par (3.9)} \\ &= (\sigma)(F_1 \wedge \dots \wedge F_k = 1) && \text{par (3.48)}. \end{aligned}$$

Quant au système $(F_1 = 0, \dots, F_k = 0)$, il est équivalent au système $(\bar{F}_1 = 1, \dots, \bar{F}_k = 1)$, donc à $\bar{F}_1 \wedge \dots \wedge \bar{F}_k = 1$, et finalement à $F_1 \vee \dots \vee F_k = 0$ par (3.22).

3.3.6 Notation

Soit f une fonction booléenne $\mathbf{B}_n \rightarrow \mathbf{B}$. Rappelons (§ 1.3.3) que si S est un sous-ensemble de \mathbf{B}_n , la notation $f(S)$ désigne l'ensemble des $f(\xi)$ tels que $\xi \in S$, avec $f(\emptyset) = \emptyset$.

Supposons que S soit l'ensemble des solutions dans \mathbf{B}_n d'un système d'équations $F_1 = G_1, \dots, F_k = G_k$ sur les variables $x_1, \dots, x_n : \mathbf{B}$. L'ensemble $f(S)$ est alors noté $f(F_1 = G_1, \dots, F_k = G_k)$. Cette notation abrège la forme exacte

$$f((\sigma x_1, \dots, x_n)(F_1 = G_1, \dots, F_k = G_k)).$$

Elle est utilisée surtout lorsque le système d'équations considéré est de la forme $x_1 = \alpha_1, \dots, x_k = \alpha_k$ ($k \leq n$), où $\alpha_1, \dots, \alpha_k$ sont des constantes (0 ou 1).

Soient par exemple $f : \mathbf{B}_3 \rightarrow \mathbf{B}$ la fonction représentée dans le tableau 3.29, et $a, b, c : \mathbf{B}$. Les solutions dans \mathbf{B}_3 du système d'équations ($a = \bar{c}, b = a \vee c$) selon l'ordre a, b, c sont les vecteurs (0, 1, 1) et (1, 1, 0). On a donc pour cette fonction f :

$$f(a = \bar{c}, b = a \vee c) = \{f(0, 1, 1), f(1, 1, 0)\} = \{0, 1\}.$$

a	b	c	f
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	1	0	1
1	1	1	0
1	0	0	1
1	0	1	0

Tableau 3.29

3.3.7 Définition

Soit $F : \mathbf{EB}(x_1, \dots, x_n)$. On dit que la fonction booléenne $f = (\lambda x_1, \dots, x_n)F$ est nulle en $(x_1 = 0, \dots, x_k = 0)$ ($k \leq n$), si $f(x_1 = 0, \dots, x_k = 0) = \{0\}$.

Par exemple soit $f = (\lambda a, b, c)((b \vee a) \wedge \bar{c})$. C'est la fonction représentée dans le tableau 3.29. Elle est nulle en $(a = 0, b = 0)$, car

$$f(a = 0, b = 0) = \{f(0, 0, 0), f(0, 0, 1)\} = \{0\}.$$

Elle n'est pas nulle en $(b = 0, c = 0)$, car

$$f(b = 0, c = 0) = \{f(0, 0, 0), f(1, 0, 0)\} = \{0, 1\}.$$

3.3.8 Proposition

Soit $F : \mathbf{EB}(x_1, \dots, x_n)$. Pour que la fonction $f = (\lambda x_1, \dots, x_n)F$ soit nulle en $(x_1 = 0, \dots, x_k = 0)$, il faut et il suffit que l'on ait, au sens du paragraphe 3.1.11 :

$$F \leq x_1 \vee \dots \vee x_k. \quad (3.106)$$

En effet, la nullité de f en $(x_1 = 0, \dots, x_k = 0)$ signifie que pour tout vecteur

$$\xi \in (\sigma x_1, \dots, x_n)(x_1 = 0, \dots, x_k = 0)$$

on a $f(\xi) = 0$, donc que l'ensemble $(\sigma)(x_1 = 0, \dots, x_k = 0)$ est contenu dans l'ensemble $\mathbf{Noy}(f)$. Autrement dit la nullité de f en $(x_1 = 0, \dots, x_k = 0)$ peut s'exprimer par l'une quelconque des relations suivantes :

$$\begin{aligned}
(\sigma)(x_1 = 0, \dots, x_k = 0) &\subset (\sigma)\bar{F} \\
(\sigma)(x_1 \vee \dots \vee x_k = 0) &\subset (\sigma)\bar{F} && (\S 3.3.5) \\
(\sigma)(\overline{x_1 \vee \dots \vee x_k} = 1) &\subset (\sigma)\bar{F} \\
(\sigma)(\overline{x_1 \vee \dots \vee x_k}) &\subset (\sigma)\bar{F} && \text{par (3.48)} \\
\overline{x_1 \vee \dots \vee x_k} &\leq \bar{F} && (\S 3.1.11) \\
F &\leq x_1 \vee \dots \vee x_k && \text{par (3.33)}.
\end{aligned}$$

3.3.9 Proposition

Soit $F : \mathbf{EB}(x_1, \dots, x_n)$. Pour $i = 1, \dots, n$, on a

$$\left. \begin{aligned}
x_i \wedge F &= x_i \wedge (x_i \leftarrow 1)F \\
\bar{x}_i \wedge F &= \bar{x}_i \wedge (x_i \leftarrow 0)F
\end{aligned} \right\} \quad (3.107)$$

Rappelons que $(x_i \leftarrow 1)F$, $(x_i \leftarrow 0)F$ sont les expressions obtenues en remplaçant la variable x_i par 1 ou par 0 dans F . Pour démontrer les égalités (3.107), il suffit de constater qu'elles sont vraies lorsque F est une expression élémentaire 0, 1, x_j , et de vérifier que si elles sont vraies pour deux expressions F, F' , alors elles sont vraies pour les expressions $F \vee F', F \wedge F', \bar{F}$. Le détail de cette démonstration est laissé au lecteur.

3.3.10 Proposition

Soit $F : \mathbf{EB}(x_1, \dots, x_n)$. Pour que la fonction $f = (\lambda x_1, \dots, x_n)F$ soit nulle en $(x_1 = 0, \dots, x_k = 0)$, il faut et il suffit que F soit équivalente à une expression de la forme

$$(F_1 \wedge x_1) \vee \dots \vee (F_k \wedge x_k) \quad (3.108)$$

où pour $i = 1, \dots, k$, F_i est une $\mathbf{EB}(x_1, \dots, x_n)$ dans laquelle x_i ne figure pas.

En effet, la relation (3.106) peut s'écrire

$$F \wedge (x_1 \vee \dots \vee x_k) = F \quad \text{par (3.28)}$$

$$(F \wedge x_1) \vee \dots \vee (F \wedge x_k) = F \quad \text{par (3.17)}$$

$$\left. \begin{aligned}
(F_1 \wedge x_1) \vee \dots \vee (F_k \wedge x_k) &= F \\
\text{avec } F_i &= (x_i \leftarrow 1)F \text{ pour } i = 1, \dots
\end{aligned} \right\} \quad \text{par (3.107)}$$

3.3.11 Etats secondaires stables d'une machine séquentielle

Soient $M(x_1, \dots, x_m)(y_1, \dots, y_n)$ une machine séquentielle binaire complètement spécifiée, y un état secondaire, et x un état primaire de cette machine. On dit que y est *stable pour* x si $y \cdot x = y$. On dit que y est *totalelement stable* s'il est stable pour tout état primaire x .

Par exemple, l'état secondaire $y = 1 \times 1$ de la machine séquentielle

$$M(x_1, x_2)(y_1, y_2)$$

définie par la table de transition 3.30 est stable pour l'état primaire $x = 0 \times 1$, et instable pour tout autre état primaire. L'état secondaire 0×0 est totalelement stable.

	$x_1 \times x_2$			
	0×0	0×1	1×1	1×0
0×0	0×0	0×0	0×0	0×0
0×1	1×0	1×0	0×0	0×0
1×1	1×0	1×1	0×1	0×1
1×0	0×1	0×1	1×1	1×1
$y_1 \times y_2$				

Tableau 3.30

3.3.12 Proposition

Soit $M(x_1, \dots, x_m)(y_1, \dots, y_n)$ une machine séquentielle binaire complètement spécifiée. Pour que l'état secondaire nul de M , c'est-à-dire l'état secondaire $y_1 \times \dots \times y_n = 0 \times \dots \times 0$, soit totalement stable, il faut et il suffit que le fonctionnement M puisse être défini par un système d'équations de récurrence de la forme

$$\left. \begin{aligned} y_1(k+1) &= F_{11}(k)y_1(k) \vee \dots \vee F_{1n}(k)y_n(k) \\ \vdots \\ y_n(k+1) &= F_{n1}(k)y_1(k) \vee \dots \vee F_{nn}(k)y_n(k) \end{aligned} \right\} \quad (3.109)$$

dans lequel, pour $i, j = 1, \dots, n$, $F_{ij}(k)$ est une expression booléenne sur $x_1(k), \dots, x_m(k), y_1(k), \dots, y_n(k)$ dans laquelle $y_j(k)$ ne figure pas. Les produits $F_{ij}y_j$ signifient naturellement $(F_{ij} \wedge y_j)$.

En effet soit $y_i(k+1) = F_i(k)$ ($i = 1, \dots, n$) un système d'équations de récurrence quelconque de M . Dire que l'état secondaire nul est totalement stable, c'est dire que chacune des fonctions booléennes $(\lambda)F_i(k)$ est nulle en $y_1(k) = 0, \dots, y_n(k) = 0$. Par suite la proposition résulte immédiatement du paragraphe 3.3.10.

3.3.13 Exemple

La machine $M(x_1, x_2)(y_1, y_2)$ du tableau 3.30 peut être définie par le système d'équations

$$\left. \begin{aligned} y_1(k+1) &= F_{11}(k)y_1(k) \vee F_{12}(k)y_2(k) \\ y_2(k+1) &= F_{21}(k)y_1(k) \vee F_{22}(k)y_2(k) \end{aligned} \right\} \quad (3.110)$$

dans lequel

$$\begin{aligned} F_{11}(k) &= x_1(k)\overline{y_2(k)} & F_{12}(k) &= \overline{x_1(k)} \\ F_{21}(k) &= x_1(k) \vee x_2(k) \vee \overline{y_2(k)} & F_{22}(k) &= 0. \end{aligned}$$

3.3.14 Exercice

Modifier le tableau 3.30 de telle façon que l'état secondaire 0×0 ne soit pas totalement stable, et vérifier que M ne peut plus être définie par un système d'équations de la forme (3.110).

3.3.15 Notation matricielle

La forme du système d'équations (3.109) suggère immédiatement la notation matricielle

$$\begin{pmatrix} y_1(k+1) \\ \vdots \\ y_n(k+1) \end{pmatrix} = \begin{pmatrix} F_{11}(k) & \dots & F_{1n}(k) \\ \vdots & & \vdots \\ F_{n1}(k) & \dots & F_{nn}(k) \end{pmatrix} \begin{pmatrix} y_1(k) \\ \vdots \\ y_n(k) \end{pmatrix} \quad (3.111)$$

dans laquelle on applique la règle usuelle du produit matriciel, la somme étant la somme logique \vee , et le produit étant le produit logique \wedge . L'équation (3.111) pourra s'abrégier $y(k+1) = F(k)y(k)$, les notations $y(k+1)$, $F(k)$, $y(k)$ désignant les matrices considérées.

3.3.16 Exemple

Considérons la machine séquentielle $M(x_1, x_2)(y_1, \dots, y_4)$ définie par les équations

$$\left. \begin{aligned} y_1(k+1) &= 0 \\ y_2(k+1) &= x_1(k)y_1(k) \vee \overline{x_2(k)}y_2(k) \\ y_3(k+1) &= x_2(k)y_4(k)y_2(k) \\ y_4(k+1) &= \overline{x_1(k)}\overline{y_3(k)}y_2(k) \vee y_3(k). \end{aligned} \right\} \quad (3.112)$$

Ce système d'équations peut s'écrire sous la forme matricielle (3.111), en prenant pour $F(k)$ la matrice

$$F = \begin{pmatrix} 0 & 0 & 0 & 0 \\ x_1 & \overline{x_2} & 0 & 0 \\ 0 & x_2 y_4 & 0 & 0 \\ 0 & \overline{x_1} \overline{y_3} & 1 & 0 \end{pmatrix} \quad (\text{le paramètre } k \text{ étant omis}).$$

Il faut remarquer que la forme matricielle d'un système d'équations de récurrence n'est pas unique en général. Par exemple, la troisième équation (3.112) peut s'écrire $y_3(k+1) = x_2(k)y_2(k)y_4(k)$, ce qui change la troisième ligne de la matrice F en $0, 0, 0, x_2 y_2$.

3.3.17 Graphes de récurrence

Ce que nous appelons *graphe de récurrence* n'est qu'une autre façon de noter un système d'équations de récurrence de la forme (3.109). Le graphe G de ce système est simplement l'ensemble des triplets (arêtes)

$$y_j \xrightarrow{F_{ij}} y_i \quad (i, j = 1, \dots, n) \quad (3.113)$$

où F_{ij} est l'expression $F_{ij}(k)$ dans laquelle on omet le paramètre k (attention à l'ordre des indices i, j dans (3.113)). Dans le dessin de G , on omet les arêtes (3.113) telles que $F_{ij} = 0$.

3.3.18 Exemple

Le système d'équations (3.112) peut être représenté par le graphe de récurrence de la figure 3.31.

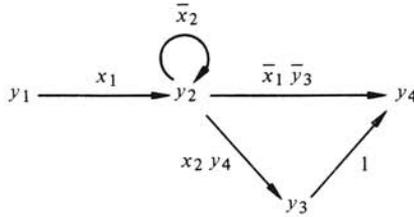


Fig. 3.31

3.3.19 Remarques

Par définition (§ 3.3.17), un graphe de récurrence représente un système d'équations de récurrence de la forme (3.109), donc une machine séquentielle binaire complètement spécifiée dont l'état secondaire nul est totalement stable. Réciproquement, toute machine de ce type peut être définie par un graphe de récurrence.

De même que la forme matricielle, le graphe d'un système d'équations de récurrence n'est pas unique en général.

3.3.20 Exercice

Représenter par un graphe de récurrence la machine du paragraphe 3.3.13. Dans le système (3.112), écrire la troisième équation sous la forme

$$y_3(k+1) = x_2(k) y_2(k) y_4(k)$$

et modifier le graphe de la figure 3.31 en conséquence.

3.3.21 Règles de passage du graphe aux équations de récurrence

La définition du paragraphe 3.3.17 permet le passage immédiat d'un système d'équations de récurrence de la forme (3.109) à son graphe. Il n'est pas inutile de préciser l'opération inverse. Etant donné un graphe G ayant les propriétés suivantes :

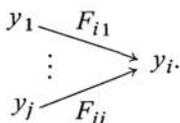
- les sommets de G sont y_1, \dots, y_n ;
- les étiquettes de G sont des expressions booléennes sur $x_1, \dots, x_m, y_1, \dots, y_n$;
- pour chaque couple y_j, y_i il y a au plus une arête

$$y_j \xrightarrow{F_{ij}} y_i$$

et y_j ne figure pas dans F_{ij} .

Alors, le système d'équations de récurrence représenté par G s'obtient en écrivant, pour $i = 1, \dots, n$, l'équation

- $y_i(k+1) = 0$ s'il n'y a aucune arête d'extrémité y_i ;
- $y_i(k+1) = F_{i1}(k) y_1(k) \vee \dots \vee F_{ij}(k) y_j(k)$ si les arêtes d'extrémité y_i sont



l'équation (3.115) définit la composante combinatoire de R . Fixons l'état secondaire initial

$$y_1(1) \times y_2(1) \times y_3(1) = 1 \times 0 \times 0. \quad (3.116)$$

Seule la phase y_1 est active à l'instant initial. La condition de franchissement 1 est vérifiée à tout instant. Donc la phase y_1 reste active à tout instant k . Il s'ensuit que la phase y_2 est active à un instant $k + 1$ si et seulement si $e(k) = 1$. La phase y_3 est active à un instant $k + 2$ si et seulement si $y_2(k + 1) = 1$ et $r(k + 1) = 1$, donc si et seulement si $e(k) = r(k + 1) = 1$. Il en découle par (3.115)

$$z(k + 2) = e(k)r(k + 1)p(k + 2).$$

Par ailleurs l'état initial (3.116) implique que la phase y_2 ne peut être active qu'au plus tôt à l'instant 2, et y_3 au plus tôt à l'instant 3. Par suite $z(1) = z(2) = 0$. La machine proposée satisfait ainsi le cahier des charges (3.70), (3.72).

Le graphe de récurrence (3.114) est d'un type particulier, en ce sens que les variables secondaires y_i ne figurent pas dans les étiquettes. Cette classe de machines sera traitée plus loin (§ 3.3.30).

3.3.24 Cahier des charges

Une machine binaire $S(x_1, x_2)(z)$ est définie comme suit. La variable de sortie z prend la valeur 1 à la fin de tout intervalle de temps $T = \{t, t + 1, \dots, t + k\}$ vérifiant les quatre conditions ci-dessous, et vaut 0 à tout autre instant.

- $k \geq 2$;
- $x_1(t) = x_1(t + k) = 1$;
- $x_1(t') = 0$ pour tout instant t' tel que $t < t' < t + k$;
- le nombre des instants t' tels que $t < t' < t + k$ et $x_2(t') = 1$ est *impair*.

Ce fonctionnement est illustré par la figure 3.32. Les traits verticaux représentent les instants 1, 2, 3, ... Les instants t tels que $x_1(t) = 1$ sont marqués d'un point \bullet sur la ligne x_1 , et de même pour les autres variables. On a mis en évidence les intervalles T vérifiant les conditions ci-dessus.

On demande de trouver une machine de Mealy $R(x_1, x_2)(y_1, \dots, y_n; z)$ et un état secondaire initial q , tel que la machine $R_q(x_1, x_2)(z)$ satisfasse ce cahier des charges.

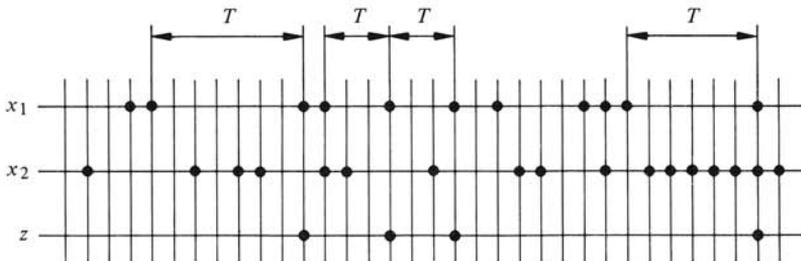


Fig. 3.32

3.3.25 Solutions

Deux solutions sont données dans les figures 3.33 et 3.34. Leur analyse approfondie est un exercice vivement recommandé pour développer la faculté d'utilisation des

graphes de récurrence. Nous commenterons la figure 3.33 par les points suivants.

- Grâce à l'état secondaire initial fixé, la phase y_1 est active à tout instant t . En effet, l'arête $(y_1, 1, y_1)$ est franchie entre $t = 1$ et $t = 2$, donc entre $t = 2$ et $t = 3$, etc...
- Si $x_1(t) = 1$, il y a une arête et une seule d'extrémité y_1 ou y_2 qui est franchie entre t et $t + 1$, à savoir l'arête (y_1, x_1, y_2) . Donc $x_1(t) = 1$ implique $y_2(t + 1) = 1$ et $y_3(t + 1) = 0$.
- Si la condition $\bar{x}_1 x_2$ est vérifiée à un instant t , alors

$$y_2(t) = 1 \implies y_2(t + 1) = 0 \text{ et } y_3(t + 1) = 1$$

$$y_3(t) = 1 \implies y_2(t + 1) = 1 \text{ et } y_3(t + 1) = 0$$
- Si les phases y_2 et y_3 sont respectivement active et inactive au début d'un intervalle de temps dans lequel x_1 vaut 0, il faut et il suffit que la condition x_2 soit vérifiée un nombre impair de fois dans cet intervalle pour que la phase y_3 soit active à la fin de l'intervalle.

Ces remarques ne constituent pas une démonstration de validité de la solution, mais devraient aider le lecteur à se convaincre de cette validité. Les équations de récurrence représentées par le graphe (fig. 3.33) s'écrivent

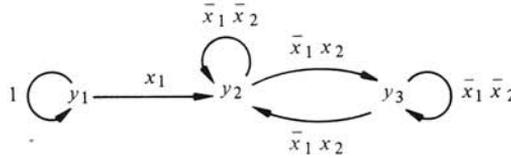
$$y_1(k + 1) = y_1(k)$$

$$y_2(k + 1) = y_1(k) x_1(k) \vee y_2(k) \overline{x_1(k)} \overline{x_2(k)} \vee y_3(k) \overline{x_1(k)} x_2(k)$$

$$y_3(k + 1) = y_2(k) \overline{x_1(k)} x_2(k) \vee y_3(k) \overline{x_1(k)} \overline{x_2(k)}.$$

Il est recommandé de tester la solution en construisant, au moyen de ces équations, les séquences y_1, y_2, y_3 correspondant aux séquences x_1, x_2 de la figure 3.32 (avec l'état initial donné), et de vérifier que l'on a bien $z(k) = y_3(k) x_1(k)$.

composante séquentielle :



composante combinatoire :

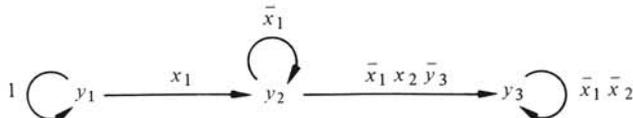
$$z(k) = y_3(k) x_1(k)$$

état secondaire initial :

$$y_1(1) \times y_2(1) \times y_3(1) = 1 \times 0 \times 0$$

Fig. 3.33

composante séquentielle :



composante combinatoire :

$$z(k) = y_3(k) x_1(k)$$

état secondaire initial :

$$y_1(1) \times y_2(1) \times y_3(1) = 1 \times 0 \times 0$$

Fig. 3.34

La seconde solution (fig. 3.30) fait figurer une variable secondaire dans l'étiquette d'une arête. On impose ici au franchissement de l'arête $y_2 \longrightarrow y_3$ entre deux instants $t, t+1$ la condition que y_3 soit inactive à l'instant t . Ce genre de condition de franchissement n'est guère indispensable dans cet exemple de cahier des charges. En théorie, il n'est même jamais indispensable. Nous verrons cependant que son usage peut être extrêmement commode, voire pratiquement indispensable. On peut faire, sur cette seconde solution les remarques qui suivent.

- La phase y_1 est active à tout instant, comme dans la première solution.
- Si $x_1(t) = 1$, alors la phase y_2 est active à tout instant $t' > t$. En effet, $x_1(t) = 1$ implique $y_2(t+1) = 1$, et par suite, pour chaque instant $t' \geq t$, l'une des arêtes $(y_1, x_1, y_2), (y_2, \bar{x}_1, y_2)$ est franchie entre t' et $t'+1$.
- Si $x_1(t) = 1$, alors $y_3(t+1) = 0$, car aucune arête d'extrémité y_3 n'est franchissable entre t et $t+1$.
- Considérons un intervalle $T = \{t, t+1, \dots, t+k\}$ tel que décrit dans le cahier des charges. On a $x_1(t) = 1$, donc y_2 est active et reste active dès l'instant $t+1$. Considérons un instant $t' (t < t' < t+k)$ tel que $x_2(t') = 1$. Si $y_3(t') = 0$, alors l'arête $y_2 \longrightarrow y_3$ est franchie et $y_3(t'+1) = 1$. Si $y_3(t') = 1$, alors aucune des arêtes d'extrémité y_3 n'est franchie, et $y_3(t'+1) = 0$. La condition $x_2(t') = 1$ a donc pour effet d'inverser l'activité de y_3 entre t' et $t'+1$. Il faut et il suffit qu'elle soit vérifiée un nombre impair de fois dans l'intervalle pour que l'on ait $y_3(t+k) = 1$.

La solution est testée dans la figure 3.35, que l'on peut construire directement à partir du graphe de récurrence, lorsqu'on a bien assimilé les règles du paragraphe 3.3.21, exprimées dans les termes du paragraphe 3.3.22.

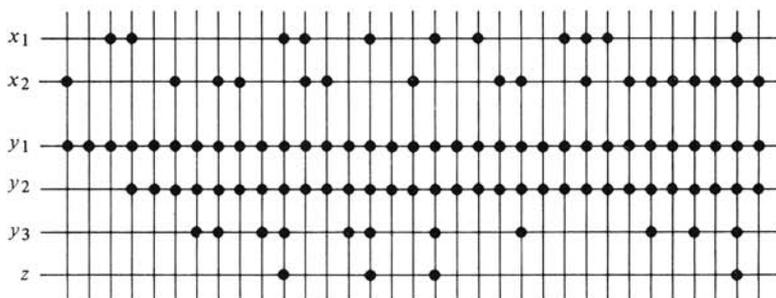


Fig. 3.35

3.3.26 Exercice : cahier des charges

Le fonctionnement d'une machine binaire $S(x_1, x_2, x_3)(z_1, z_2)$ est décrit par les points suivants, illustrés dans la figure 3.36.

- $z_1(k) = 1$ si et seulement si $k > 1$ et $x_1(k-1) = x_2(k) = 1$.
- $z_2(k) = 1$ si et seulement s'il existe un instant $k' < k$ tel que $x_3(k') = 1$ et $x_2(k'') = 1$ pour tout instant $k'' (k' < k'' \leq k)$.

On demande de trouver une machine de Mealy $R(x_1, x_2, x_3)(y_1, \dots, y_n; z_1, z_2)$ et un état secondaire initial q , telle que R_q soit une réalisation de S .

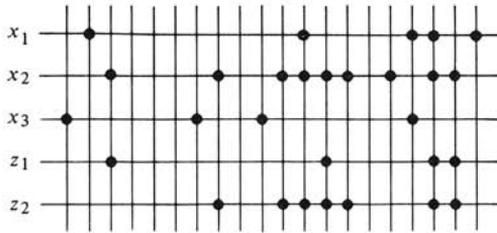


Fig. 3.36

3.3.27 Exercice : cahier des charges

Nous disons qu'une séquence $x(1, k) \in \mathbf{B}^+$ se termine par un nombre pair (resp. impair) de 0 s'il existe un instant $h < k$ tel que $x(h) = 1, x(h+1) = x(h+2) = \dots = x(k) = 0$, et $k-h$ est pair (resp. impair). Par exemple, les séquences 100, 010000 se terminent par un nombre pair de 0; les séquences 110, 0101000 se terminent par un nombre impair de 0. On définit de façon symétrique les séquences se terminant par un nombre pair de 1 (ainsi 011, 1001111), et celles se terminant par un nombre impair de 1 (ainsi 101, 0111). Ceci posé, on définit le fonctionnement d'une machine binaire $M(x)(z_1, z_2, z_3, z_4)$ par les points suivants :

- $z_1(k) = 1$ ssi (si et seulement si) la séquence $x(1, k)$ se termine par un nombre pair de 0;
- $z_2(k) = 1$ ssi $x(1, k)$ se termine par un nombre impair de 0;
- $z_3(k) = 1$ ssi $x(1, k)$ se termine par un nombre pair de 1;
- $z_4(k) = 1$ ssi $x(1, k)$ se termine par un nombre impair de 1.

On demande de réaliser cette machine par une machine de Mealy binaire munie d'un état secondaire initial.

3.3.28 Cahier des charges

Le système d'enclenchement d'un moteur électrique comporte trois boutons poussoirs A, B, C. Le moteur se met en marche lorsqu'on a poussé et relâché A et B dans n'importe quel ordre. Il s'arrête lorsqu'on a poussé et relâché C. Une action sur A ou sur B pendant que le moteur est en marche, une action sur C pendant qu'il est arrêté n'ont aucun effet.

Ce fonctionnement est illustré par le chronogramme de la figure 3.37 où a, b, c sont les signaux logiques associés aux boutons A, B, C, et z le signal qui commande la marche du moteur. On effectue un échantillonnage des signaux a, b, c, z , avec une période d'échantillonnage assez courte pour que toute variation des signaux soit prise en compte. On obtient ainsi les séquences a, b, c, z de la figure 3.37, et l'on cherche une machine de Mealy $R(a, b, c)(y_1, \dots, y_n; z)$ avec un état secondaire initial p tel que la machine $R_p(a, b, c)(z)$ satisfasse ce cahier des charges.

3.3.29 Solutions

Deux solutions sont proposées dans les figures 3.38 et 3.39. Le lecteur les analysera de la manière indiquée au paragraphe 3.3.25. Les deux solutions sont des machines de Moore (z ne dépend pas des variables primaires). On observera dans les deux solutions

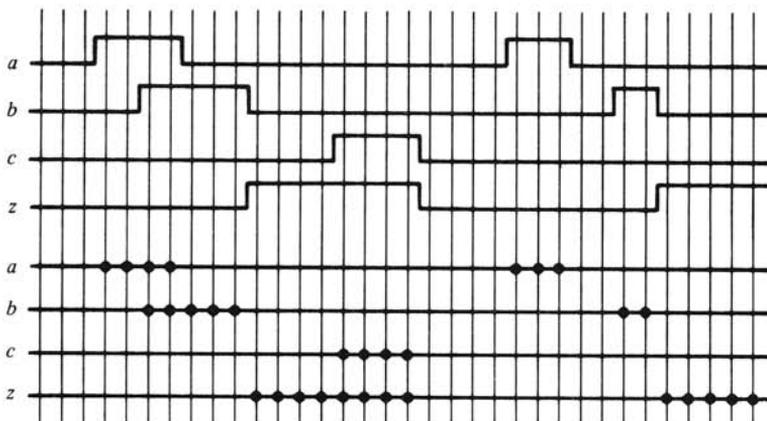
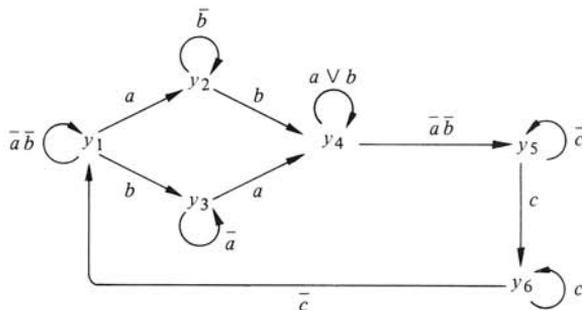


Fig. 3.37

composante séquentielle :



composante combinatoire :

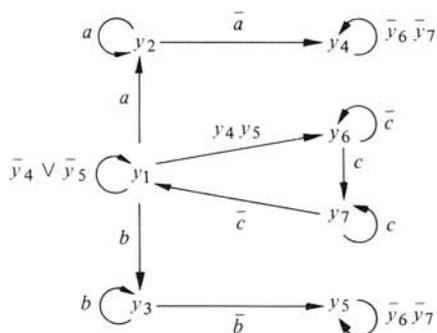
$$z(k) = y_5(k) \vee y_6(k)$$

état secondaire initial :

$$y_1(1) \times \dots \times y_6(1) = 1 \times 0 \times 0 \times 0 \times 0 \times 0$$

Fig. 3.38

composante séquentielle :



composante combinatoire :

$$z(k) = y_6(k) \vee y_7(k)$$

état secondaire initial :

$$y_1(1) \times \dots \times y_7(1) = 1 \times 0 \times \dots \times 0$$

Fig. 3.39

un retard à l'enclenchement et au déclenchement de z , par rapport à la figure 3.37. Dans la première solution, la seconde phrase du cahier des charges (§ 3.3.28) est interprétée dans le sens suivant : le moteur se met en marche lorsqu'on a appuyé une fois au moins sur chacun des boutons A, B et lorsque ces deux boutons sont relâchés. En outre, si l'on appuie simultanément sur A et B, la première solution suppose que cette action s'étend sur au moins deux périodes d'échantillonnage. Un cahier des charges informel autorise toujours une certaine liberté d'interprétation.

3.3.30 Machines séquentielles B-linéaires

Une machine séquentielle binaire $M(x_1, \dots, x_m)(y_1, \dots, y_n)$ est dite **B-linéaire**, ou *linéaire* au sens *booléen* du terme, si elle peut être définie par un système d'équations de récurrence de la forme

$$y_1(k+1) = F_{i1}(k)y_1(k) \vee \dots \vee F_{in}(k)y_n(k) \quad (i = 1, \dots, n) \quad (3.117)$$

où toutes les expressions $F_{ij}(k)$ ($i, j = 1, \dots, n$) sont des expressions booléennes sur $x_1(k), \dots, x_m(k)$. Les variables secondaires ne figurent pas dans ces expressions.

Une telle machine peut être représentée par un graphe de récurrence dans lequel les conditions de franchissement de toutes les arêtes sont indépendantes des variables secondaires.

Par exemple, les machines séquentielles définies par les graphes de récurrence des figures 3.33, 3.38 sont linéaires.

3.3.31 Exemple

Soit $M(x_1, x_2)(y_1, y_2, y_3)$ la machine séquentielle définie par le graphe de récurrence de la figure 3.40. Cette machine est linéaire. Nous pouvons écrire ses équations de récurrence :

$$\left. \begin{aligned} y_1(k+1) &= x_1(k)y_1(k) \vee \overline{x_1(k)}y_3(k) \\ y_2(k+1) &= x_1(k)\overline{x_2(k)}y_1(k) \vee (\overline{x_1(k)} \vee x_2(k))y_2(k) \\ y_3(k+1) &= x_2(k)y_2(k). \end{aligned} \right\} \quad (3.118)$$

La table de transition de la machine (tab. 3.41) est tirée des équations (3.118). La linéarité en y_1, y_2, y_3 se traduit par deux propriétés de la table de transition. Premièrement l'état secondaire nul $(0, 0, 0)$ est totalement stable. Secondement, si pour deux états secondaires quelconques $y = (y_1, y_2, y_3)$, $y' = (y'_1, y'_2, y'_3)$ on désigne par $y \vee y'$ l'état secondaire $(y_1 \vee y'_1, y_2 \vee y'_2, y_3 \vee y'_3)$, on observe que pour tout état primaire x

$$(y \vee y') \cdot x = (y \cdot x) \vee (y' \cdot x). \quad (3.119)$$

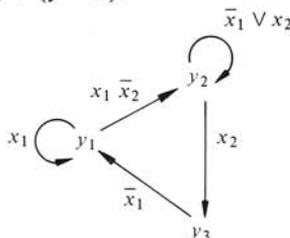


Fig. 3.40

	x_1		x_2					
	0	0	0	1	1	1	1	0
0 0 0	0	0	0	0	0	0	0	0
0 0 1	1	0	0	0	0	0	0	0
0 1 1	1	1	0	1	1	1	0	0
0 1 0	0	1	0	0	1	1	0	0
1 1 0	0	1	0	0	1	1	1	0
1 1 1	1	1	0	1	1	1	1	0
1 0 1	1	0	0	1	0	0	1	0
1 0 0	0	0	0	0	0	0	1	0

 $y_1 y_2 y_3$

Tableau 3.41

Par exemple, pour $y = (0, 0, 1)$, $y' = (1, 1, 0)$, on a $y \vee y' = (1, 1, 1)$, et l'on observe dans la table que la ligne correspondant à $(1, 1, 1)$ est la disjonction bit par bit des lignes correspondant à $(0, 0, 1)$ et $(1, 1, 0)$.

Grâce à ces deux propriétés, la table de transition est entièrement déterminée lorsqu'on connaît ses trois lignes correspondant aux états secondaires *unitaires* $(0, 0, 1)$, $(0, 1, 0)$, $(1, 0, 0)$, puisque tout autre état secondaire peut s'exprimer comme une disjonction d'états unitaires. Par ailleurs, une ligne correspondant à un état unitaire peut se construire directement d'après le graphe de récurrence. Par exemple, on construit la dernière ligne de la table en se plaçant à un instant k où seule la phase y_1 est active, et en suivant pour chaque état primaire les arêtes franchies, donc les phases actives à l'instant suivant.

La propriété de linéarité (3.119) peut se démontrer de façon générale. En vertu de la forme du système d'équations (3.117), le vecteur $y \cdot x$, noté en colonne, se calcule par un produit matriciel Fy , où F est une matrice carrée dont les composantes ne dépendent pas de celles de y . Ainsi dans notre exemple

$$y \cdot x = \begin{pmatrix} x_1 & 0 & \bar{x}_1 \\ x_1 \bar{x}_2 & \bar{x}_1 \vee x_2 & 0 \\ 0 & x_2 & 0 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}.$$

Il est clair que cette opération matricielle Fy est linéaire en y , c'est-à-dire que $F(y \vee y') = Fy \vee Fy'$.

Réciproquement, si une machine séquentielle binaire complètement spécifiée (donnée par sa table de transition) possède la propriété (3.119), et si son état secondaire nul est totalement stable, alors elle est linéaire au sens du paragraphe 3.3.30.

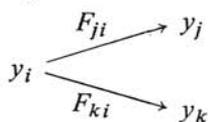
Il est à noter que pour une machine linéaire, la matrice F est unique, contrairement au cas d'une machine non linéaire (§ 3.3.16).

3.3.32 Machines séquentielles uniformes

Nous appelons *uniforme* une machine séquentielle binaire

$$M(x_1, \dots, x_m)(y_1, \dots, y_n)$$

qui est **B**-linéaire, et dont le graphe de récurrence G possède en outre la propriété suivante : pour deux arêtes quelconques



ayant même origine et des extrémités distinctes ($j \neq k$), les expressions F_{ji}, F_{ki} sont telles que $F_{ji} \wedge F_{ki} = 0$. Cette propriété s'entend comme valable au cas où $i = j$.

3.3.33 Exemple : cahier des charges

Sur une ligne de chemin de fer à voie unique se trouve un passage à niveau gardé (fig. 3.42). On détecte en trois points A, B, C de la voie la présence d'un train. Le point C est situé sur le passage à niveau, A et B sont en amont et en aval du passage. Des variables binaires $a(k), b(k), c(k)$ sont associées aux trois points. La relation $a(k) = 1$ signifie qu'un train se trouve sur A à l'instant k , et de même pour $b(k)$ et $c(k)$. On demande de construire une machine ayant une variable de sortie binaire $z(k)$ qui commande la fermeture des barrières selon la règle suivante : les barrières se ferment sitôt qu'un train arrive de l'extérieur sur A ou sur B (les trains peuvent passer dans les deux sens); elles s'ouvrent dès que le train a fini de passer sur C. La longueur L d'un train peut être quelconque : $0 < L < \infty$. Ce cahier des charges est tiré de [13].

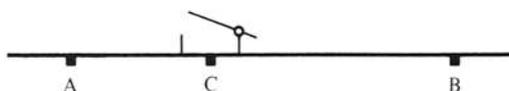
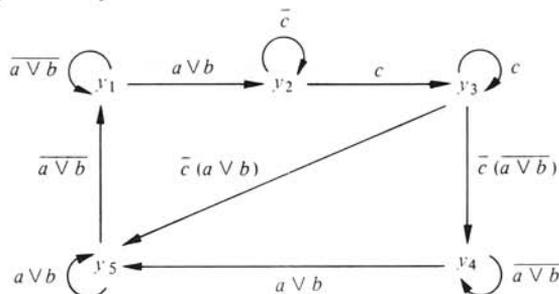


Fig. 3.42

Une solution est proposée dans la figure 3.43, dont la composante séquentielle est uniforme. Dans cet exemple en outre, la disjonction de toutes les conditions de franchissement des arêtes issues d'un même sommet est égale à 1. L'analyse de cette solution est laissée au lecteur.

composante séquentielle :



composante combinatoire :

$$z(k) = y_2(k) \vee y_3(k)$$

état secondaire initial :

$$y_1(1) \times \dots \times y_5(1) = 1 \times 0 \times 0 \times 0 \times 0$$

Fig. 3.43

3.3.34 Définition

On appelle *norme* d'un vecteur binaire $x = (x_1, \dots, x_n)$ le nombre des composantes non nulles de x , et l'on désigne ce nombre par $\|x\|$. Ainsi

$$\|x\| = \sum_{i=1}^n x_i \quad (\text{somme arithmétique}). \quad (3.120)$$

Un vecteur nul est un vecteur de norme 0, un vecteur unitaire est un vecteur de norme 1. Si x, x' sont deux vecteurs de même dimension,

$$\|x \vee x'\| \leq \|x\| + \|x'\|. \quad (3.121)$$

3.3.35 Proposition

Pour tout état total $y \times x$ d'une machine séquentielle binaire *uniforme*

$M(x_1, \dots, x_m)(y_1, \dots, y_n)$,

$$\|y \cdot x\| \leq \|y\|. \quad (3.122)$$

En effet, la relation (3.122) est vraie si $\|y\| = 0$, en vertu de la stabilité de l'état secondaire nul. Si $\|y\| = 1$ la relation est vraie d'après le paragraphe 3.3.32 : si seule une phase y_i est active à un instant k , alors il y a au plus une phase active à l'instant $k + 1$. Enfin, si $\|y\| = p > 1$, y peut s'exprimer comme la disjonction de p vecteurs unitaires η_1, \dots, η_p et par la propriété de linéarité $y \cdot x = \eta_1 \cdot x \vee \dots \vee \eta_p \cdot x$. Chacun des vecteurs $\eta_i \cdot x$ étant de norme ≤ 1 , il vient $\|y \cdot x\| \leq p$ par (3.121).

3.3.36 Remarque

Il y a une grande ressemblance entre le graphe de récurrence d'une machine séquentielle uniforme, et un graphe d'états (graphe de transition) au sens classique. Considérons par exemple la machine séquentielle définie par la table ou le graphe de transition de la figure 3.44. C'est la composante séquentielle d'une machine longuement étudiée dans le volume V (chap. 6 : discriminateur de sens de rotation). On peut la considérer comme une machine séquentielle $R(x_1, x_2)(y)$, dont l'alphabet primaire est l'alphabet binaire B_2 , et l'alphabet secondaire est l'alphabet "quelconque" $Y = \{a, b, c, d\}$. On sait que la méthode classique consiste, pour obtenir une machine

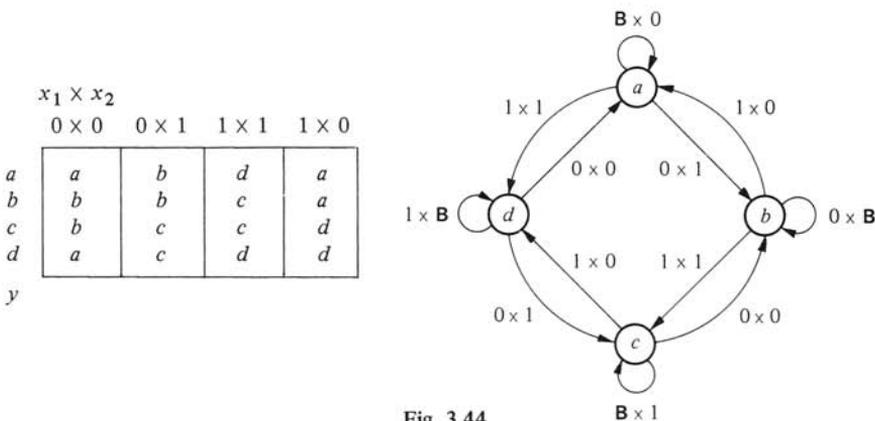


Fig. 3.44

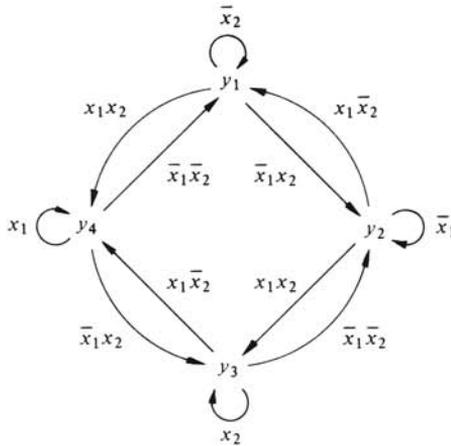


Fig. 3.45

binaire, à coder les états a, b, c, d par des vecteurs binaires distincts, de même dimension. Ce code peut être quelconque. Mais si l'on choisit les quatre vecteurs *unitaires* de dimension 4, ce qu'on appelle de façon générale codage 1 parmi n (§ V.6.2.12), on obtient la machine binaire $M(x_1, x_2)(y_1, \dots, y_4)$ uniforme dont le graphe de récurrence est donné dans la figure 3.45. La ressemblance des deux graphes est évidente, et le passage de l'un à l'autre immédiat. Mais il faut bien voir que les machines R et M sont distinctes : notamment la seconde possède 16 états secondaires, et non 4.

3.4 GRAPHES DE RÉCURRENCE RÉCEPTIFS

3.4.1 Définition

Soient

$$y_i \xrightarrow{F_{ji}} y_j \quad (i, j = 1, \dots, n)$$

les arêtes d'un graphe de récurrence G . Nous appellerons *réceptivité* d'une phase y_i dans G , et nous désignerons par $\rho_G(y_i)$, ou simplement $\rho(y_i)$, la disjonction des conditions de franchissement F_{ji} telles que $j \neq i$. Nous écrirons ceci :

$$\rho_G(y_i) = \bigvee_{j \neq i} F_{ji}. \quad (3.123)$$

Les F_{ji} intervenant dans cette définition sont les conditions de franchissement de toutes les arêtes d'origine y_i et d'extrémité différente de y_i . Si l'on considère la matrice des conditions de franchissement

$$F = \begin{pmatrix} F_{11} & \dots & F_{1i} & \dots & F_{1n} \\ F_{i1} & \dots & F_{ii} & \dots & F_{in} \\ \vdots & & \vdots & & \vdots \\ F_{n1} & \dots & F_{ni} & \dots & F_{nn} \end{pmatrix} \quad (3.124)$$

on voit que la réceptivité de y_i est la disjonction des éléments non diagonaux de la i -ème colonne de F .

3.4.2 Exemples

La figure 3.46 reproduit le graphe de récurrence de la figure 3.38. Dans ce graphe, on a $\rho(y_1) = a \vee b$, $\rho(y_2) = b$, $\rho(y_3) = a$, $\rho(y_4) = \bar{a}\bar{b}$, etc. Notons que ces réceptivités sont les mêmes que celles de la figure 3.47.

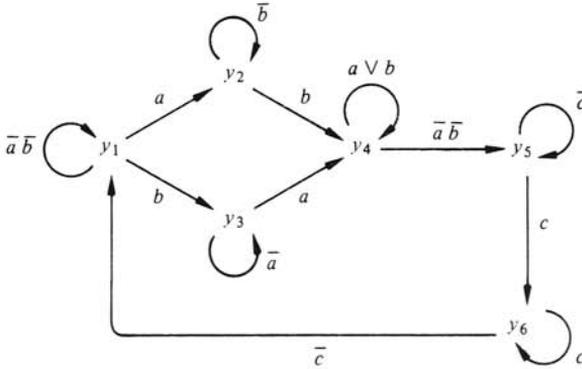


Fig. 3.46

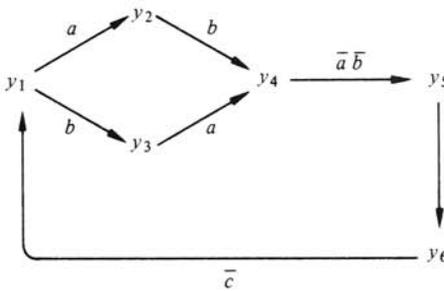


Fig. 3.47

Pour le graphe de la figure 3.48, où R, S, T, U sont des expressions booléennes non précisées, la matrice des conditions de franchissement s'écrit :

$$F = \begin{pmatrix} 1 & 0 & 0 & 0 \\ R & S & 0 & 0 \\ 0 & U & 0 & 0 \\ 0 & T & 1 & 0 \end{pmatrix} \tag{3.125}$$

Dans ce graphe, $\rho(y_1) = R$, $\rho(y_2) = U \vee T$, $\rho(y_3) = 1$, $\rho(y_4) = 0$. On rappelle (§ 3.3.17) que les arêtes dont la condition de franchissement est 0 sont omises dans le dessin d'un graphe de récurrence.

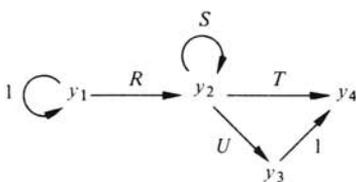


Fig. 3.48

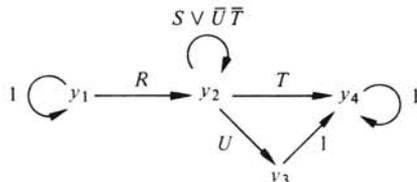


Fig. 3.49

3.4.3 Définition

Un graphe de récurrence G , dont les arêtes sont

$$y_i \xrightarrow{F_{ji}} y_j \quad (i, j = 1, \dots, n)$$

est dit *réceptif* si pour chaque phase y_i la condition suivante est vérifiée :

$$F_{ii} \vee \overline{\rho_G(y_i)} = F_{ii}. \quad (3.126)$$

Il s'agit là d'une équivalence d'expressions booléennes, au sens de la section 3.1. Cette condition peut s'écrire selon (3.28) :

$$\overline{\rho_G(y_i)} \leq F_{ii}. \quad (3.127)$$

Nous écrirons encore, en appliquant (3.22) à (3.123) :

$$\overline{\rho_G(y_i)} = \bigwedge_{j \neq i} \overline{F_{ji}}. \quad (3.128)$$

Le graphe de la figure 3.46 est réceptif, et même plus, puisqu'il vérifie l'égalité $\overline{\rho(y_i)} = F_{ii}$ pour chaque phase y_i . Le graphe de la figure 3.48 n'est pas réceptif, car $\overline{\rho(y_4)} = \overline{0} = 1$ et $F_{44} = 0$. La condition (3.127) est vérifiée cependant pour y_1 et y_3 ; pour y_2 on ne peut rien dire tant que les expressions S, T, U restent indéterminées.

3.4.4 Définition

Soient les arêtes d'un graphe de récurrence G :

$$y_i \xrightarrow{F_{ji}} y_j \quad (i, j = 1, \dots, n)$$

et F (3.124) la matrice des conditions de franchissement. Supposons que l'on modifie les éléments diagonaux F_{ii} de cette matrice, en les remplaçant par

$$F'_{ii} = F_{ii} \vee \overline{\rho_G(y_i)}, \quad (3.129)$$

ceci pour $i = 1, \dots, n$ et en laissant inchangés les éléments non diagonaux. On définit ainsi une nouvelle matrice F' , avec

$$F'_{ij} = F_{ij} \quad \text{pour } i \neq j \quad (3.130)$$

$$F'_{ii} = F_{ii} \vee \left(\bigwedge_{j \neq i} \overline{F_{ji}} \right) \quad \text{pour } i = 1, \dots, n \quad (3.131)$$

Nous désignerons par $|G$ le graphe de récurrence dont les arêtes sont

$$y_i \xrightarrow{F'_{ji}} y_j \quad (i, j = 1, \dots, n).$$

Il est clair que ce graphe $|G$ est réceptif. On a en effet $\rho_{|G}(y_i) = \rho_G(y_i)$ en vertu de (3.130). Par suite $F'_{ii} \vee \overline{\rho_{|G}(y_i)} = F'_{ii}$ selon (3.129).

Le graphe $|G$ est appelé le *graphe de récurrence réceptif associé* au graphe de récurrence G . Il est clair que si G est lui-même réceptif (3.126), alors les matrices F et F' sont égales.

Le graphe de la figure 3.46 est le graphe réceptif associé à celui de la figure 3.47. Le graphe de la figure 3.49 est le graphe réceptif associé à celui de la figure 3.48. On le

vérifie immédiatement en effectuant sur la matrice F (3.125) la modification décrite ci-dessus, et qui donne :

$$F' = \begin{pmatrix} 1 & 0 & 0 & 0 \\ R & S \vee \bar{U}\bar{T} & 0 & 0 \\ 0 & U & 0 & 0 \\ 0 & T & 1 & 1 \end{pmatrix}. \quad (3.132)$$

3.4.5 Notation

Etant donné un graphe de récurrence G , on représentera le graphe réceptif associé $|G$ par le diagramme de G lui-même dans lequel on aura barré les flèches par un trait $|$. Un tel diagramme est appelé un *diagramme de réceptivité*.

Ainsi, le graphe de la figure 3.46 sera représenté par la figure 3.50 (cf. fig. 3.47), et le graphe de la figure 3.49 par la figure 3.51 (cf. fig. 3.48).

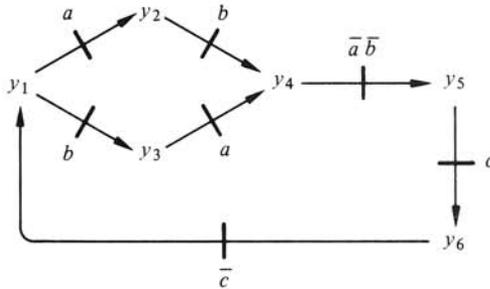


Fig. 3.50

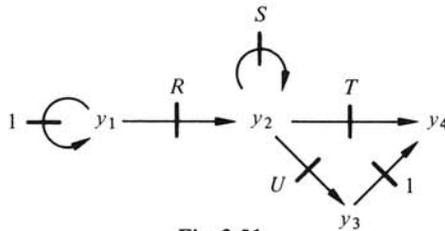


Fig. 3.51

Les figures 3.48 et 3.51 sont identiques aux barres $|$ près. Il faut bien voir cependant qu'elles représentent des systèmes d'équations de récurrence *non équivalents*. Pour écrire les équations de récurrence représentées par la seconde, il faut considérer le graphe réceptif qu'elle représente (fig. 3.49), et appliquer à *celui-ci* les règles du paragraphe 3.3.21. Il vient :

$$\left. \begin{aligned} y_1(k+1) &= y_1(k) \\ y_2(k+1) &= R(k)y_1(k) \vee (S(k) \vee \overline{U(k)}\overline{T(k)})y_2(k) \\ y_3(k+1) &= U(k)y_2(k) \\ y_4(k+1) &= T(k)y_2(k) \vee y_3(k) \vee y_4(k). \end{aligned} \right\} \quad (3.133)$$

On peut alors se demander si la nouvelle notation présente un intérêt quelconque, hormis l'économie d'écriture. Il se trouve que l'intérêt principal ne réside pas dans cette économie, mais dans une interprétation directe en termes de règles d'activité des phases, analogues à celles du paragraphe 3.3.22. Elles seront énoncées au paragraphe 3.4.8, et l'on verra par la suite comment cette manière de penser convient à la formalisation de toute une classe de cahiers des charges.

3.4.6 Proposition

Soient

$$y_i \xrightarrow{F_{ji}} y_j \quad (i, j = 1, \dots, n)$$

les arêtes d'un graphe de récurrence G . Le système d'équations de récurrence représenté par le graphe réceptif associé $|G$ est équivalent au système d'équations

$$y_i(k+1) = \left(\bigvee_{j=1}^n F_{ij}(k) y_j(k) \right) \vee \left(\bigwedge_{j=1}^n \overline{F_{ji}(k)} \right) y_i(k) \quad (3.134)$$

$$(i = 1, \dots, n).$$

3.4.7 Démonstration

Par définition (§ 3.4.4), le système d'équations de récurrence représenté par $|G$ s'écrit, en omettant le paramètre k au second membre :

$$y_i(k+1) = \left(\bigvee_{j \neq i} F_{ij} y_j \right) \vee \left(F_{ii} \vee \bigwedge_{j \neq i} \overline{F_{ji}} \right) y_i. \quad (3.135)$$

Il y a lieu d'appliquer ici la formule booléenne

$$F \vee G = F \vee (\overline{F} \wedge G) \quad (3.136)$$

involontairement omise au paragraphe 3.1.10, et qui se démontre comme suit :

$$F \vee (\overline{F} \wedge G) = (F \vee \overline{F}) \wedge (F \vee G) = 1 \wedge (F \vee G) = F \vee G.$$

Grâce à cette formule, on montre facilement que le second membre de (3.135) est égal à celui de (3.134). Il suffit de remarquer que

$$F_{ii} \vee \bigwedge_{j \neq i} \overline{F_{ji}} = F_{ii} \vee \overline{F_{ii}} \bigwedge_{j \neq i} \overline{F_{ji}} = F_{ii} \vee \bigwedge_{j=1}^n \overline{F_{ji}}.$$

3.4.8 Règles d'activité

Soient

$$y_i \xrightarrow{F_{ji}} y_j \quad (i, j = 1, \dots, n)$$

les arêtes d'un graphe de récurrence G . Le système d'équations de récurrence représenté par le graphe réceptif associé $|G$ peut se traduire par les *règles d'activité* suivantes, valables pour une phase y_i quelconque.

- Si à l'instant k il y a dans G une arête

$$y_j \xrightarrow{F_{ij}} y_i$$

d'extrémité y_i , telle que $y_j(k) = 1$ et $F_{ij}(k) = 1$, alors $y_i(k+1) = 1$.

- Si $y_i(k) = 1$ et si pour toute arête

$$y_i \xrightarrow{F_{ji}} y_j$$

d'origine y_i dans G on a $F_{ji}(k) = 0$, alors $y_i(k+1) = 1$.

- La phase y_i ne peut être active à l'instant $k+1$ qu'à l'une au moins des conditions énoncées dans les règles précédentes.

Ces règles découlent immédiatement du paragraphe 3.4.6. Elles ne font que traduire l'équation (3.134). On observe que si l'on supprime la seconde règle, ce qui revient à supprimer dans (3.134) le terme

$$\bigwedge_{j=1}^n \overline{F_{ji}(k)} y_i(k),$$

on obtient les règles d'activité relatives à G . L'adjonction de la deuxième règle a pour effet de maintenir en activité une phase *active* y_i lorsque son activité ne peut se communiquer à aucune phase par franchissement d'une arête d'origine y_i dans G .

3.4.9 Exemple

Considérons la phase y_2 de la figure 3.51 et énonçons les règles d'activité de cette phase. Premièrement : si $y_1(k) = 1$ et $R(k) = 1$, ou si $y_2(k) = 1$ et $S(k) = 1$, alors $y_2(k+1) = 1$. Secondement : si $y_2(k) = 1$ et $S(k) = T(k) = U(k) = 0$, alors $y_2(k+1) = 1$. Troisièmement : y_2 ne peut être active à l'instant $k+1$ qu'à l'une au moins de ces conditions.

On pouvait bien entendu énoncer ces règles en ignorant les paragraphes 3.4.6 à 3.4.8, et en se reportant à la figure équivalente 3.49. Il suffisait d'y remplacer l'expression $S \vee \bar{U} \bar{T}$ par l'expression équivalente $S \vee \bar{S} \bar{U} \bar{T}$. Les paragraphes mentionnés n'avaient pour but que de généraliser cet exemple.

Considérons encore la phase y_4 (fig. 3.51). La seconde règle d'activité entraîne ceci : si $y_4(k) = 1$, alors $y_4(k+1) = 1$, car toutes les arêtes d'origine y_4 dans G (fig. 3.48) ont une étiquette nulle (3.125). Cette règle peut aussi se tirer de la figure 3.49.

3.4.10 Commentaire

Les graphes de récurrence réceptifs se présentent de façon naturelle lorsqu'on cherche à formaliser par graphes de récurrence le cahier des charges de systèmes séquentiels à *comportement asynchrone* (§ V.6.1.6). Ces systèmes sont caractérisés approximativement par la propriété suivante : dans tout intervalle de temps dans lequel les variables d'entrée sont constantes, les variables de sortie sont constantes. Nous avons dit "approximativement", car la propriété s'entend en faisant abstraction d'un éventuel retard du changement des sorties par rapport au changement des entrées qui en est la cause. Nous ne pouvons entrer ici dans plus de détails.

On remarque que le système du paragraphe 3.3.28 est de ce type. Il a pu être représenté par un graphe réceptif (fig. 3.38, 3.46, 3.50). Nous allons en voir d'autres exemples, nous offrant l'occasion d'utiliser la notation du paragraphe 3.4.5.

3.4.11 Cahier des charges

Deux véhicules C_1, C_2 (fig. 3.52) peuvent se déplacer sur des voies parallèles entre un lieu A et un lieu B. L'état de mouvement du système à un instant t est décrit par 8 variables binaires $a_1(t), a_2(t), b_1(t), b_2(t), d_1(t), d_2(t), g_1(t), g_2(t)$ ayant les significations suivantes :

$$\begin{aligned} a_i(t) = 1 &\iff C_i \text{ est en A;} \\ b_i(t) = 1 &\iff C_i \text{ est en B;} \\ d_i(t) = 1 &\iff C_i \text{ est en mouvement vers la droite;} \\ g_i(t) = 1 &\iff C_i \text{ est en mouvement vers la gauche.} \end{aligned}$$

Le système est complété par un bouton poussoir S, représenté par une variable binaire $s(t)$, et il fonctionne de la façon suivante. Initialement les deux véhicules sont au repos en A. Ils y restent tant que l'on n'appuie pas sur S ($s(t) = 0$). Sitôt qu'on appuie sur S ($s(t) = 1$), chacun d'eux effectue à sa vitesse propre un aller et retour ABA. Ce mouvement est appelé le cycle du système. Le cycle est terminé lorsque les deux véhicules sont de retour en A. Un nouveau cycle s'effectue si les deux véhicules étant en A, on appuie à nouveau sur S. Mais si S est maintenu constamment poussé, un seul cycle a lieu. Par ailleurs, une manipulation quelconque de S pendant le mouvement des véhicules est sans effet. On demande de représenter ce fonctionnement par une machine binaire $M(s, a_1, a_2, b_1, b_2)(d_1, g_1, d_2, g_2)$.

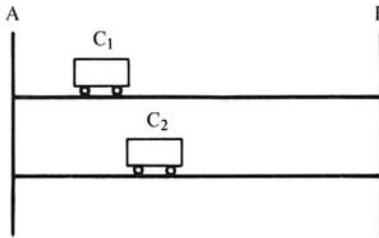


Fig. 3.52

Une solution est proposée dans la figure 3.53. Il s'agit d'une machine de Moore dont la composante séquentielle est définie par un graphe de récurrence réceptif, et munie d'un état secondaire initial. Dans cet état, seule la phase y_1 est active (phase encerclée deux fois). Les équations de la composante combinatoire sont simplement :

$$\begin{aligned} d_1(t) &= y_3(t) ; & d_2(t) &= y_2(t) \\ g_1(t) &= y_5(t) ; & g_2(t) &= y_4(t). \end{aligned}$$

Ceci est noté dans la figure par le placement des variables d_1, d_2, g_1, g_2 à côté des phases correspondantes. On peut généraliser cette notation lorsqu'une variable tertiaire z est déterminée par une équation de la forme $z(t) = y_i(t) \vee y_j(t) \vee \dots$, en écrivant z à côté des sommets y_i, y_j, \dots du graphe de récurrence.

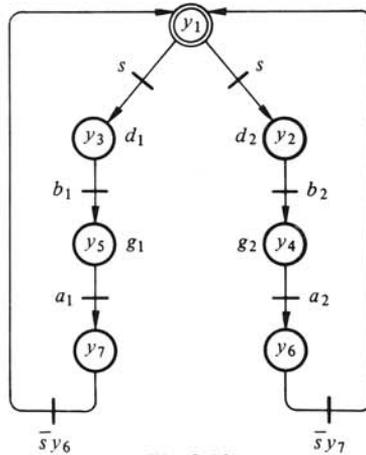


Fig. 3.53

3.4.12 Exercice : cahier des charges

Reprenre les données du paragraphe précédent, en y apportant l'une des modifications possibles indiquées ci-dessous.

- Le premier véhicule qui arrive en B y attend l'arrivée de l'autre, et alors seulement ils repartent vers A.
- Le premier véhicule qui arrive en B y attend l'arrivée de l'autre. Alors C_1 retourne en A pendant que C_2 reste en B, et lorsque C_1 est arrivé en A, C_2 quitte B et retourne en A.
- L'arrivée en B de l'un quelconque des véhicules détermine le retour immédiat des deux véhicules en A, quelle que soit la position de l'autre véhicule à ce moment.

3.4.13 Machines d'enclenchement et déclenchement

Pour toute séquence binaire $x(1, n)$, nous appelons *séquence des enclenchements* de x , et *séquence des déclenchements* de x , et nous notons respectivement $x \uparrow(1, n)$, $x \downarrow(1, n)$ les séquences binaires définies par :

$$\left. \begin{aligned} x \uparrow(1) &= x(1) \\ x \uparrow(k+1) &= \overline{x(k)} \wedge x(k+1) \quad (k = 1, \dots, n-1) \end{aligned} \right\} \quad (3.137)$$

$$\left. \begin{aligned} x \downarrow(1) &= \overline{x(1)} \\ x \downarrow(k+1) &= x(k) \wedge \overline{x(k+1)} \quad (k = 1, \dots, n-1). \end{aligned} \right\} \quad (3.138)$$

Cette définition est illustrée par la figure 3.54.

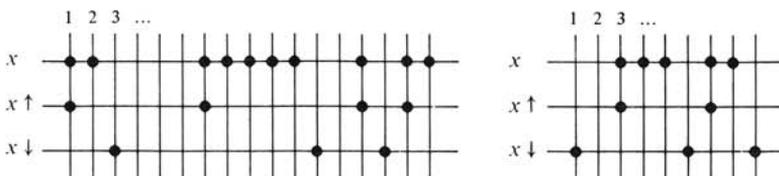
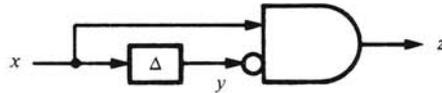


Fig. 3.54

Une *machine d'enclenchement* (resp. *déclenchement*) est une machine binaire $M(x)(z)$, telle que $M(x) = x \uparrow$ (resp. $M(x) = x \downarrow$) pour toute séquence d'entrée x . Les figures 3.55, 3.56, 3.57 sont trois réalisations d'une machine d'enclenchement. La première n'est pas représentable par un graphe de récurrence (§ 3.3.19). La seconde est définie au moyen d'un graphe de récurrence non réceptif, et la troisième par un graphe de récurrence réceptif. Une machine de déclenchement admet des réalisations analogues.



composante séquentielle: $y(k+1) = x(k)$
 composante combinatoire: $z(k) = \overline{y(k)} \wedge x(k)$
 état secondaire initial: $y(1) = 0$

Fig. 3.55

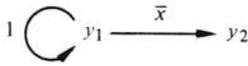
composante séquentielle: 
 composante combinatoire: $z(k) = y_2(k) \wedge x(k)$
 état secondaire initial: $y_1(1) \times y_2(1) = 1 \times 1$

Fig. 3.56

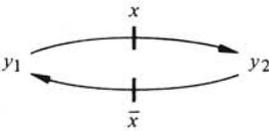
composante séquentielle: 
 composante combinatoire: $z(k) = y_1(k) \wedge x(k)$
 état secondaire initial: $y_1(1) \times y_2(1) = 1 \times 0$

Fig. 3.57

Les machines d'enclenchement ou déclenchement peuvent être des accessoires commodes. Reprenons par exemple le cahier des charges du paragraphe 3.4.11 (fig. 3.53). Une autre solution est présentée dans la figure 3.58, qui se compose de trois graphes réceptifs distincts en interaction. Le troisième réalise une machine d'enclen-

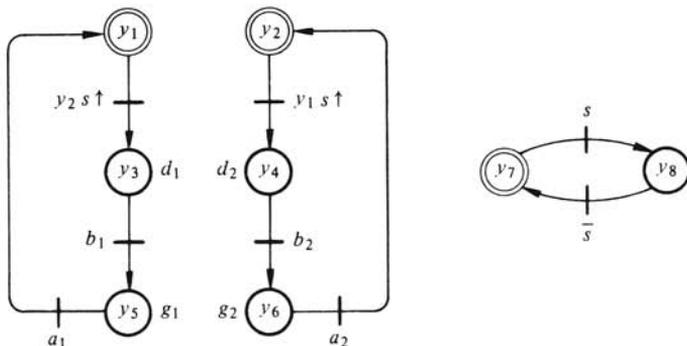


Fig. 3.58

chement, et dans les deux premiers l'expression $s \uparrow$ est mise pour $s y_7$. L'avantage de cette solution est que l'on peut omettre le graphe d'enclenchement. Il faut seulement se rappeler que la condition $s \uparrow$ est vérifiée à l'instant 1 si $s(1) = 1$, et à l'instant $k > 1$ si $s(k) = 1$ et $s(k - 1) = 0$.

3.4.14 Cahier des charges

Nous appelons *impulsion* d'une séquence binaire $x(1, n)$ tout intervalle $[[t_1, t_2]] = \{t_1, t_1 + 1, \dots, t_2\}$ tel que $1 \leq t_1 \leq t_2 \leq n$, et vérifiant les conditions suivantes :

- $x(t) = 1$ pour tout instant $t \in [[t_1, t_2]]$
- si $1 < t_1$, alors $x(t_1 - 1) = 0$
- si $t_2 < n$, alors $x(t_2 + 1) = 0$.

Par exemple, les impulsions de la séquence $x(1, 13) = 1101011101111$ sont les intervalles $[[1, 2]]$, $[[4, 4]]$, $[[6, 8]]$, $[[10, 13]]$.

Une machine binaire $M(x_1, x_2)(z)$ est définie par la règle suivante : étant donné une séquence d'entrée $x_1 \times x_2$, la séquence de sortie correspondante z est construite en prenant pour chaque impulsion $[[t_1, t_2]]$ de x_1 la première impulsion $[[t'_1, t'_2]]$ de x_2 telle que $t_1 < t'_1$. Cette règle est illustrée par la figure 3.59. On demande de réaliser cette machine par une machine de Mealy munie d'un état secondaire initial. On ne tolère aucun retard de z dans la solution, par rapport à ce qui est donné dans la figure 3.59.

Une solution est proposée dans la figure 3.60. Son analyse est laissée au lecteur.

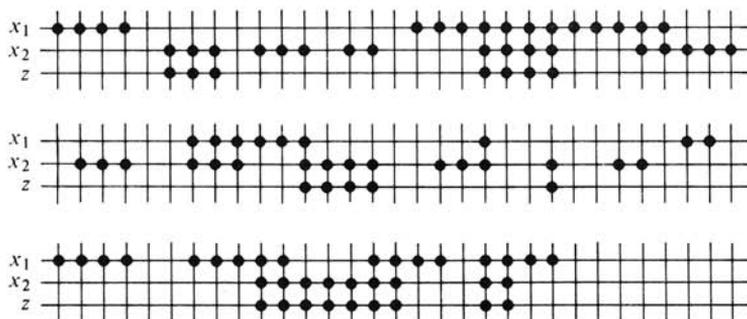
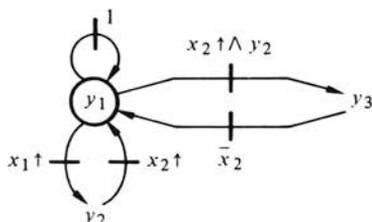


Fig. 3.59

Composante séquentielle :



Composante combinatoire : $z(k) = y_3(k) x_2(k) \vee y_2(k) (x_2 \uparrow(k))$
 Etat secondaire initial : $y_1(1) \times y_2(1) \times y_3(1) = 1 \times 0 \times 0$.

Fig. 3.60

3.4.15 Exercice : cahier des charges

Les entrées et les sorties d'une machine binaire $M(a_1, a_2, r_3)(r_1, r_2, a_3)$ sont groupées par paires (fig. 3.61) : (a_1, r_1) , (a_2, r_2) , (a_3, r_3) . Chacune de ces paires est appelée une *voie de communication* v_1, v_2, v_3 . Les voies $v_1 = (a_1, r_1)$, $v_2 = (a_2, r_2)$ sont dites *voies d'entrée* de M , et la voie $v_3 = (a_3, r_3)$ est dite *voie de sortie*. Une voie v_i est dite *libre* à un instant t si $r_i(t) = 0$, et *occupée* si $r_i(t) = 1$. On dit qu'il y a *appel* sur une voie v_i à l'instant t si $a_i(t) = 1$.

Un appel sur une voie d'entrée à un instant t est enregistré par la machine si cette voie est libre à cet instant. Sitôt que la machine a enregistré un appel sur une voie d'entrée, elle occupe cette voie et la maintient occupée. Lorsqu'elle a enregistré un appel sur chacune des voies d'entrée (dans n'importe quel ordre), elle appelle sur la voie de sortie sitôt que celle-ci est libre. Elle appelle ainsi jusqu'à ce qu'elle reçoive une réponse, c'est-à-dire jusqu'à ce que $r_3(t) = 1$. Sitôt qu'elle reçoit cette réponse, elle cesse d'appeler et libère les deux voies d'entrée.

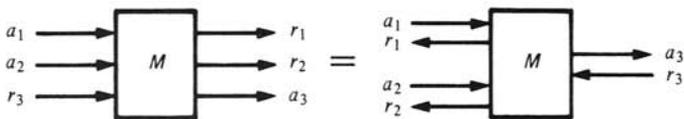


Fig. 3.61

3.4.16 Notation

Les figures 3.62, 3.63, 3.64 introduisent une notation abrégée pour les graphes de récurrence réceptifs. Dans chacune d'elles, le diagramme de droite est la notation abrégée de celui de gauche. F est une expression quelconque.

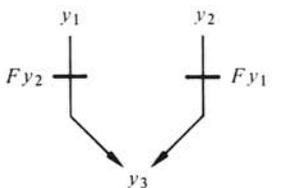


Fig. 3.62

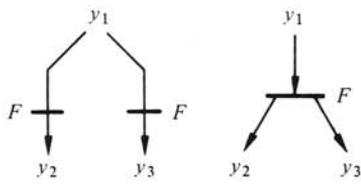


Fig. 3.63

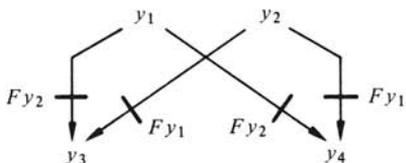


Fig. 3.64

Plus généralement, le diagramme de la figure 3.65 représente $k \times p$ arêtes :

$$y_i \xrightarrow{F_{ji}} y_j \\ (i = 1, \dots, k) \quad (j = m + 1, \dots, m + p)$$

dont les $k \times p$ conditions de franchissement F_{ji} sont toutes de la forme :

$$Fy_1 \dots y_{i-1} y_{i+1} \dots y_k.$$

La notation représente donc un faisceau d'arêtes qui sont toujours franchies simultanément, et ceci à la condition que y_1, \dots, y_k soient toutes actives et que F soit vérifiée. On ne suppose pas dans cette définition de notation, que $m > k$. Elle s'applique par exemple à la figure 3.66.

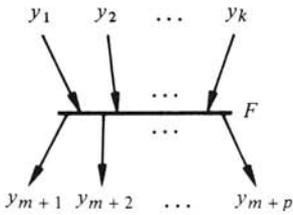


Fig. 3.65

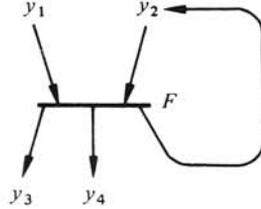


Fig. 3.66

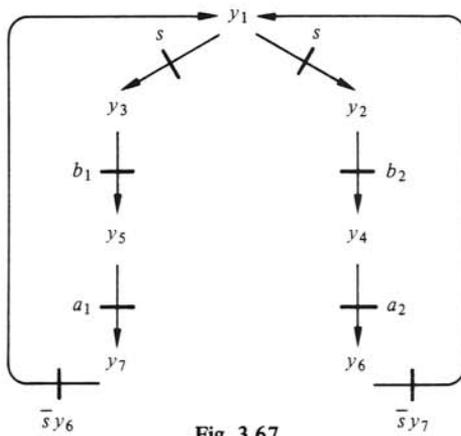


Fig. 3.67

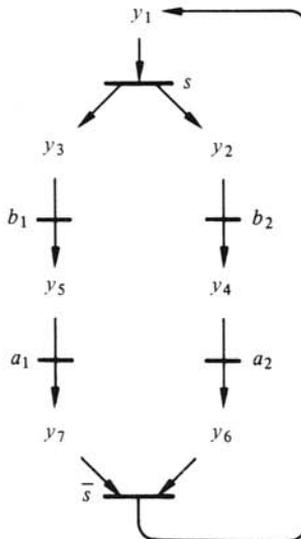


Fig. 3.68

Avec cette notation, le graphe de récurrence réceptif représenté par la figure 3.67 (cf. § 3.4.11, fig. 3.53) peut être représenté également par la figure 3.68.

3.4.17 Références bibliographiques

La représentation de systèmes logiques séquentiels au moyen de graphes qui ne sont pas des graphes d'états est assez récente. Divers types de graphes ont été proposés dans la littérature [15, 16, 17]. Ils ont tous la propriété commune suivante : les états secondaires de la machine séquentielle ne sont pas représentés chacun par un sommet du graphe, mais par un ensemble de sommets, que l'on dit actifs dans l'état considéré. Dès lors, chaque sommet n'est autre qu'une variable secondaire $y(t)$, qu'on l'appelle phase, place, ou étape comme il se trouve dans la littérature. Et le graphe représente un système d'équations de récurrence booléennes. Nous sommes donc partis de cette idée : noter sous forme de graphe un tel système d'équations. Il nous a paru raisonnable de n'appliquer cette idée qu'à un type particulier d'équations, celles qui définissent une machine séquentielle complètement spécifiée dont l'état secondaire nul est totalement stable (§ 3.3.19). Les machines réceptives (§ 3.4.3) en sont un cas particulier. L'*organiphase* [15] est un graphe qui s'interprète selon les règles du paragraphe 3.4.8. Il entre donc dans la classe des machines réceptives. Il en est de même du GRAFCET [16] auquel nous avons emprunté la notation ci-dessus (§ 3.4.16). Par contre, les *réseaux de Petri* [17] représentent des machines séquentielles incomplètement spécifiées et de type non standard (§ 2.4.34). Leurs équations de récurrence n'entrent pas dans la classe la plus générale considérée dans ce chapitre.

EXPRESSIONS RÉGULIÈRES

4.1 OPÉRATIONS RÉGULIÈRES

4.1.1 Introduction

Le chapitre précédent était consacré exclusivement aux machines binaires. Dans le présent chapitre, nous revenons aux machines plus générales présentées au chapitre 2. Un exemple nous servira d'introduction, tout en indiquant les notions qu'il pourrait être judicieux de revoir.

Considérons la figure 4.1 qui reproduit la figure 2.63. Elle présente la table de transition et le graphe de transition d'une machine de Mealy $R[X, Y, Z]$, avec l'alphabet primaire $X = \{a, b, c, d\}$, l'alphabet secondaire $Y = \{p, q, r\}$, et l'alphabet tertiaire $Z = \{0, 1\}$ (§ 2.6.8). Nous rappelons les notations du paragraphe 2.6.12.

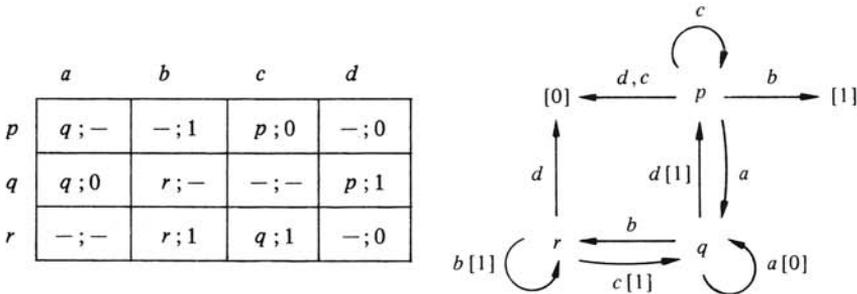


Fig. 4.1

Si l'on fixe un état secondaire initial, par exemple q , on définit par là-même une machine $R_q : X^+ \rightarrow Z^+$ (§ 2.6.15) dans laquelle l'alphabet secondaire Y est ignoré. Pour chaque séquence $x \in X^+$, la figure 4.1 permet de déterminer l'ensemble de séquences $R_q(x) \subset Z^+$. Ainsi pour $x = bdd$, on a $R_q(x) = \{000, 001, 100, 101\}$ (§ 2.6.16). On peut imaginer la correspondance R_q représentée par une table infinie, où dans une première colonne seraient écrites toutes les séquences d'entrée x , et dans une seconde colonne, en regard de chaque séquence x , serait noté l'ensemble $R_q(x)$ correspondant. Une table infinie n'est cependant qu'une vue de l'esprit, et l'on voudrait pouvoir la remplacer par une expression mathématique finie, définissant la correspondance R_q . Certes le graphe de la figure 4.1 est bien une "expression" finie définissant cette correspondance. Cependant il fait usage d'un alphabet secondaire Y . On voudrait pouvoir décrire la correspondance R_q par une expression dans laquelle ne figureraient que les lettres des alphabets X et Z .

La motivation de ce souhait est la suivante. Dans une machine de Mealy $R[X, Y, Z]$, la composante séquentielle et l'alphabet secondaire jouent un rôle auxiliaire. La

machine est construite en vue de réaliser une correspondance $X^+ \rightarrow Z^+$. Sa construction procède en partant d'un cahier des charges qui ne se réfère qu'aux alphabets X et Z . On l'a bien vu dans les exemples du chapitre 3. On peut ainsi imaginer que la machine ci-dessus (fig. 4.1) a été construite en vue de réaliser la correspondance R_q , celle-ci ayant été décrite au préalable dans un cahier des charges. On voit qu'il serait avantageux de disposer d'un formalisme permettant d'exprimer cette correspondance au moyen des seuls alphabets X, Z et d'en tirer la machine R (fig. 4.1) par une méthode systématique.

L'étude d'un tel formalisme et de ses applications sont précisément l'objet de ce chapitre.

4.1.2 Définitions

Rappelons qu'un alphabet X est simplement un ensemble fini non vide, et que X^+ désigne l'ensemble de toutes les séquences sur cet alphabet (§ 1.4.1). Un ensemble de séquences quelconque A sur X , c'est-à-dire un sous-ensemble quelconque $A \subset X^+$ est appelé un *langage* sur X , et nous désignerons par L_X l'ensemble des langages sur X .

L'ensemble vide $\emptyset \subset X^+$, l'ensemble X^+ lui-même ($X^+ \subset X^+$) sont des langages sur X . L'ensemble X , considéré comme l'ensemble des séquences de longueur 1 sur X , est un langage sur X . Rappelons que pour tout entier $n \geq 1$, X^n est l'ensemble des séquences de longueur n sur X . C'est aussi un langage sur X .

Au paragraphe 1.4.2 nous avons défini le produit xy de deux séquences x, y sur un alphabet X , et nous avons remarqué (§ 1.4.3) que ce produit est associatif : $(xy)z = x(yz)$. On définit par récurrence le produit $\Pi(x_1, \dots, x_n)$ d'un n -uplet de séquences (x_1, \dots, x_n) , pour tout $n \geq 1$, en posant

$$\Pi(x_1, \dots, x_n) = \begin{cases} x_1 & \text{si } n = 1 \\ (\Pi(x_1, \dots, x_{n-1}))x_n & \text{si } n > 1. \end{cases} \quad (4.1)$$

Etant donné une séquence x , on appelle *décomposition de x de dimension n* (n entier ≥ 1) tout n -uplet de séquences (x_1, \dots, x_n) tel que $x = \Pi(x_1, \dots, x_n)$. Toute composante x_i d'une décomposition (x_1, \dots, x_n) de x est appelée une *syllabe* de x . La première et la dernière composante d'une décomposition quelconque de x sont appelées respectivement un *préfixe* et une *terminaison* de x . En particulier x est elle-même une syllabe, un préfixe, et une terminaison de x : décomposition (x) de dimension 1.

Le produit AB de deux langages A, B sur un alphabet X a été défini au paragraphe 1.4.5. Utilisant la définition ci-dessus, nous pouvons exprimer la définition de AB par la relation

$$x \in AB \iff x \text{ admet une décomposition } (x_1, x_2) \text{ telle que } x_1 \in A \text{ et } x_2 \in B. \quad (4.2)$$

On n'exige pas dans (4.2) que la décomposition (x_1, x_2) soit unique. Considérons par exemple les langages finis $A = \{0, 01\}$, $B = \{12, 112\}$ sur l'alphabet $X = \{0, 1, 2\}$. La séquence $x = 0112$ appartient au langage AB , car elle admet la décomposition $(01 \in A, 12 \in B)$. Elle admet aussi la décomposition $(0 \in A, 112 \in B)$. Le langage AB est ici le langage fini $\{012, 0112, 01112\}$.

On a vu (§ 1.4.7) que le produit de langages est associatif : $(AB)C = A(BC)$. Le produit $\Pi(A_1, \dots, A_n)$ d'un n -uplet de langages (A_1, \dots, A_n) est défini par récurrence :

$$\Pi(A_1, \dots, A_n) = \begin{cases} A_1 & \text{si } n = 1 \\ (\Pi(A_1, \dots, A_{n-1})) A_n & \text{si } n > 1. \end{cases} \quad (4.3)$$

On en tire aussitôt la généralisation de (4.2) :

$$x \in \Pi(A_1, \dots, A_n) \iff x \text{ admet une décomposition } (x_1, \dots, x_n) \text{ telle que } x_i \in A_i \text{ pour } i = 1, \dots, n. \quad (4.4)$$

Lorsque les langages A_1, \dots, A_n sont tous égaux à un même langage A , le produit $\Pi(A_1, \dots, A_n)$ est noté A^n . On a donc

$$\left. \begin{aligned} A^1 &= A \\ A^{n+1} &= A^n A \text{ pour tout entier } n \geq 1. \end{aligned} \right\} \quad (4.5)$$

4.1.3 Propriétés

Les propriétés principales du produit de langages ont été vues au paragraphe 1.4.7. Nous rappelons ici :

$$A \subset B \implies \begin{cases} AC \subset BC \\ CA \subset CB \end{cases} \quad (4.6)$$

$$\begin{aligned} A(B \cup C) &= AB \cup AC \\ (B \cup C)A &= BA \cup CA \end{aligned} \quad (4.7)$$

$$A \emptyset = \emptyset A = \emptyset. \quad (4.8)$$

Nous ajoutons les formules suivantes qui découlent immédiatement de (4.5) :

$$A^m A^n = A^{m+n} \quad (4.9)$$

$$(A^m)^n = A^{mn}. \quad (4.10)$$

Enfin si A_1, \dots, A_n sont des langages quelconques, on a d'après (4.4) :

$$x \in \Pi(A_1, \dots, A_n) \implies \lg(x) \geq n. \quad (4.11)$$

4.1.4 Définition

Soit A un langage quelconque sur un alphabet X . On appelle *itération* de A la réunion des langages A^n ($n \geq 1$). Ce langage est noté A^+ . On a donc

$$A^+ = A \cup A^2 \cup A^3 \cup \dots = \bigcup_{n \geq 1} A^n. \quad (4.12)$$

Une séquence x sur X appartient au langage A^+ si et seulement si elle appartient à une puissance quelconque de A , autrement dit :

$$x \in A^+ \iff x \text{ admet une décomposition } (x_1, \dots, x_n) \text{ de dimension } n \geq 1 \text{ quelconque telle que } x_i \in A \text{ pour } i = 1, \dots, n. \quad (4.13)$$

4.1.5 Exemple

Considérons le langage fini $A = \{01, 12, 010, 112\}$ sur l'alphabet $X = \{0, 1, 2\}$. La séquence $x = 010112$ appartient au langage A^+ , car elle admet la décomposition $(01, 01, 12)$ selon laquelle elle appartient à A^3 . Elle admet aussi la décomposition $(010, 112)$ et appartient donc aussi à A^2 .

4.1.6 Remarque

En désignant par X^+ l'ensemble des séquences sur un alphabet X , on utilise le signe $+$ d'une manière compatible avec la définition de l'itération. En effet toute séquence x sur X admet une décomposition (x_1, \dots, x_n) avec $x_i \in X$ ($i = 1, \dots, n$). Les syllabes x_i sont de longueur 1, et la décomposition est unique.

4.1.7 Propriétés de l'itération

Soient A, B des langages quelconques sur un alphabet X . On a les formules suivantes :

$$A^n \subset A^+ \text{ (pour tout entier } n \geq 1) \quad (4.14)$$

$$A^+ = A \cup AA^+ = A \cup A^+A \quad (4.15)$$

$$(A^+)^+ = A^+ \quad (4.16)$$

$$\emptyset^+ = \emptyset \quad (4.17)$$

$$A \subset B \Rightarrow A^+ \subset B^+. \quad (4.18)$$

4.1.8 Démonstration

La relation (4.14) découle immédiatement de (4.12). Pour (4.15) il suffit d'écrire $AA^+ = A(A \cup A^2 \cup \dots) = A^2 \cup A^3 \cup \dots$ et $A^+A = (A \cup A^2 \cup \dots)A = A^2 \cup A^3 \cup \dots$. Pour (4.16), on a d'abord $A^+ = (A^+)^1 \subset (A^+)^+$ d'après (4.14). Par ailleurs si $x \in (A^+)^+$, x admet une décomposition (x_1, \dots, x_n) avec $x_i \in A^+$ pour $i = 1, \dots, n$. Chaque composante x_i admet à son tour une décomposition (x_{i1}, x_{i2}, \dots) de composantes $x_{ik} \in A$. Il est clair que $(x_{11}, x_{12}, \dots, x_{n1}, x_{n2}, \dots)$ est une décomposition de x dont toutes les composantes appartiennent à A . Par suite $x \in A^+$, et ceci montre que $(A^+)^+ \subset A^+$. Pour (4.17), on peut écrire $\emptyset^+ = \emptyset \cup \emptyset \emptyset^+ = \emptyset$ par (4.15) et (4.8). Enfin (4.18) est évidente.

4.1.9 Définitions

Les opérations de *réunion* $A \cup B$, de *produit* AB , et d'*itération* A^+ appliquées à des langages A, B sont appelées *opérations régulières*.

Soit Λ un ensemble de langages sur un alphabet X . Nous désignons par $\mathbf{Opr}(\Lambda)$ l'ensemble des langages de la forme $A \cup B$ ou AB ou A^+ , avec $A, B \in \Lambda$. Autrement dit $\mathbf{Opr}(\Lambda)$ est l'ensemble des langages qui peuvent être obtenus à partir des langages de l'ensemble Λ en effectuant *une* opération régulière.

Par exemple, si Λ se compose de trois langages A, B, C , alors l'ensemble $\mathbf{Opr}(\Lambda)$ se compose des langages $A \cup A, A \cup B, A \cup C, B \cup B, B \cup C, C \cup C, AA, AB, AC, BA, BB, BC, CA, CB, CC, A^+, B^+, C^+$. Il est clair que tous ces langages ne sont pas nécessairement distincts. On remarque, puisque $A \cup A = A$, que

$$\Lambda \subset \mathbf{Opr}(\Lambda). \quad (4.19)$$

On dira qu'un ensemble de langages Λ sur un alphabet X est *stable pour les opérations régulières*, si

$$\mathbf{Opr}(\Lambda) = \Lambda. \quad (4.20)$$

Pour montrer que Λ est stable, il suffit, en vertu de (4.19) de montrer que

$\mathbf{Opr}(\Lambda) \subset \Lambda$, c'est-à-dire que

$$A \in \Lambda \text{ et } B \in \Lambda \Rightarrow A \cup B \in \Lambda \quad (4.21)$$

$$A \in \Lambda \text{ et } B \in \Lambda \Rightarrow AB \in \Lambda \quad (4.22)$$

$$A \in \Lambda \Rightarrow A^+ \in \Lambda. \quad (4.23)$$

4.1.10 Définition

Soit Λ un ensemble de langages sur un alphabet X . Nous désignons par $\mathbf{LR}(\Lambda)$ l'ensemble des langages qui peuvent être construits à partir des langages de Λ en effectuant une *succession* quelconque d'opérations régulières. Autrement dit, l'ensemble $\mathbf{LR}(\Lambda)$ est la réunion des ensembles Λ , $\mathbf{Opr}(\Lambda)$, $\mathbf{Opr}(\mathbf{Opr}(\Lambda))$, ... Posons

$$\mathbf{LR}_0(\Lambda) = \Lambda \quad (4.24)$$

et pour tout entier $n \geq 0$

$$\mathbf{LR}_{n+1}(\Lambda) = \mathbf{Opr}(\mathbf{LR}_n(\Lambda)). \quad (4.25)$$

Ceci définit une suite infinie d'ensembles de langages $\mathbf{LR}_n(\Lambda)$ ($n = 0, 1, 2, \dots$). l'ensemble $\mathbf{LR}(\Lambda)$ est défini comme la réunion de tous les ensembles $\mathbf{LR}_n(\Lambda)$ de cette suite. Cela signifie qu'un langage A appartient à l'ensemble $\mathbf{LR}(\Lambda)$ si et seulement s'il existe un entier n tel que $A \in \mathbf{LR}_n(\Lambda)$.

On remarque que $\mathbf{LR}_n(\Lambda) \subset \mathbf{LR}_{n+1}(\Lambda)$ en vertu de (4.25) et (4.19), d'où

$$m \leq n \Rightarrow \mathbf{LR}_m(\Lambda) \subset \mathbf{LR}_n(\Lambda). \quad (4.26)$$

Nous dirons que les langages $A \in \mathbf{LR}(\Lambda)$ sont les langages *régulièrement engendrés* par l'ensemble de langages Λ . Lorsque Λ se compose d'un nombre fini de langages, $\Lambda = \{A_1, \dots, A_n\}$, l'ensemble $\mathbf{LR}(\Lambda)$ est noté $\mathbf{LR}(A_1, \dots, A_n)$.

4.1.11 Proposition

L'ensemble $\mathbf{LR}(\Lambda)$ contient Λ ; il est stable pour les opérations régulières; tout ensemble de langages Ω qui contient Λ et qui est stable pour les opérations régulières contient $\mathbf{LR}(\Lambda)$.

Cette proposition peut se résumer par la phrase : $\mathbf{LR}(\Lambda)$ est le plus petit ensemble de langages contenant Λ et stable pour les opérations régulières.

4.1.12 Démonstration

La première assertion résulte immédiatement de (4.24). Pour la seconde, il suffit de montrer que les relations (4.21), (4.22), (4.23) sont vraies lorsqu'on y remplace Λ par $\mathbf{LR}(\Lambda)$. Soient donc $A, B \in \mathbf{LR}(\Lambda)$. Il existe deux entiers m, n tels que $A \in \mathbf{LR}_m(\Lambda)$ et $B \in \mathbf{LR}_n(\Lambda)$. Admettons que $m \leq n$. On a $A \in \mathbf{LR}_n(\Lambda)$ par (4.26). En vertu

de (4.25), les langages $A \cup B$, AB , A^+ appartiennent à $\mathbf{LR}_{n+1}(\Lambda)$, donc à $\mathbf{LR}(\Lambda)$. Pour démontrer la troisième assertion, considérons un ensemble de langages Ω tel que $\Lambda \subset \Omega$ et $\mathbf{Opr}(\Omega) = \Omega$. On a $\mathbf{LR}_0(\Lambda) \subset \Omega$ par (4.24). Supposons que $\mathbf{LR}_n(\Lambda) \subset \Omega$. Il est clair que cette hypothèse entraîne $\mathbf{Opr}(\mathbf{LR}_n(\Lambda)) \subset \mathbf{Opr}(\Omega)$, d'où $\mathbf{LR}_{n+1} \subset \Omega$. Ainsi par récurrence on a $\mathbf{LR}_n(\Lambda) \subset \Omega$ pour tout entier n . Par suite $\mathbf{LR}(\Lambda) \subset \Omega$.

4.1.13 Exemple

Soient A, B, C trois langages sur un alphabet X . L'expression $(A \cup (B^+C))^+$ représente un langage régulièrement engendré par l'ensemble $\Lambda = \{A, B, C\}$. On a en effet $B^+ \in \mathbf{LR}_1(\Lambda)$, $B^+C \in \mathbf{LR}_2(\Lambda)$, $A \cup (B^+C) \in \mathbf{LR}_3(\Lambda)$, $(A \cup (B^+C))^+ \in \mathbf{LR}_4(\Lambda)$.

4.1.14 Corollaire

Soient Λ, Λ' deux ensembles de langages sur un alphabet X . Si $\Lambda' \subset \mathbf{LR}(\Lambda)$, alors $\mathbf{LR}(\Lambda') \subset \mathbf{LR}(\Lambda)$.

En d'autres termes, tout langage A qui est régulièrement engendré par un ensemble de langages Λ' régulièrement engendrés par Λ , est lui-même régulièrement engendré par Λ .

Pour preuve, il suffit de poser $\Omega = \mathbf{LR}(\Lambda)$. L'ensemble Ω est stable pour les opérations régulières (§ 4.1.11). L'hypothèse $\Lambda' \subset \Omega$ entraîne $\mathbf{LR}(\Lambda') \subset \Omega$ (§ 4.1.11).

4.1.15 Opérations *

On rencontre fréquemment des expressions de la forme $A^+B \cup B$ ou $BA^+ \cup B$, dans lesquelles A et B sont des langages sur un alphabet X . Pour les abréger, on pose

$$A^+B \cup B = A^*B \quad (4.27)$$

$$BA^+ \cup B = BA^* \quad (4.28)$$

En développant A^+ selon (4.12), il vient

$$A^*B = B \cup AB \cup A^2B \cup A^3B \cup \dots \quad (4.29)$$

$$BA^* = B \cup BA \cup BA^2 \cup BA^3 \cup \dots \quad (4.30)$$

Une séquence x du langage A^*B (resp. BA^*) est donc soit une séquence de B soit une séquence de B précédée (resp. suivie) de n séquences de A ($n \geq 1$ quelconque).

Il est clair que les langages A^*B et BA^* sont régulièrement engendrés par l'ensemble $\{A, B\}$.

4.1.16 Exemple

Soient X l'alphabet $\{0, 1, 2\}$ et B le langage fini $\{01, 02\}$. L'expression X^*B représente l'ensemble de toutes les séquences x sur X qui possèdent la terminaison 01 ou la terminaison 02 (§ 4.1.2). De même BX^* est l'ensemble des séquences qui possèdent le préfixe 01 ou le préfixe 02 .

4.1.17 Commentaire

Les opérations A^*B , BA^* sont *binaires*, en ce sens qu'elles portent sur *deux* langages A, B . Par opposition l'opération A^+ portant sur *un* langage est *unaire*. Mais l'ex-

pression A^* seule n'a pas de sens dans cet ouvrage. Nous attirons l'attention sur ce point, car la plupart des auteurs donnent un sens à l'expression A^* . Ils introduisent à cet effet une séquence spéciale λ de longueur nulle, dite séquence neutre, telle que $\lambda x = x\lambda = x$ pour toute séquence x . Ils définissent alors A^* par $A^* = A^+ \cup \{\lambda\}$. La relation (4.27) s'en déduit comme conséquence : $A^*B = (A^+ \cup \{\lambda\})B = A^+B \cup \{\lambda\}B = A^+B \cup B$. La relation (4.28) se déduit semblablement. L'introduction d'une séquence neutre présente des avantages algébriques, mais entraîne des complications théoriques. Nous l'écartons définitivement.

4.1.18 Proposition

Soient A, B, C des langages quelconques sur un alphabet X . On a les formules suivantes :

$$A^+ = A^*A = AA^* \quad (4.31)$$

$$\left. \begin{aligned} A^*(B \cup C) &= A^*B \cup A^*C \\ (B \cup C)A^* &= BA^* \cup CA^* \end{aligned} \right\} \quad (4.32)$$

$$\emptyset^*A = A\emptyset^* = A \quad (4.33)$$

$$A^*\emptyset = \emptyset A^* = \emptyset \quad (4.34)$$

$$(A^*B)C = A^*(BC) \quad (4.35)$$

$$(AB^*)C = A(B^*C) \quad (4.36)$$

$$(AB)C^* = A(BC^*) \quad (4.37)$$

En vertu des trois dernières relations, nous écrivons A^*BC , AB^*C , ABC^* sans parenthèses. Pour commenter ces notations, on peut dire qu'une séquence du langage A^*BC est formée d'un nombre quelconque (éventuellement nul) de séquences de A suivies d'une séquence de B et d'une séquence de C . Une séquence du langage AB^*C est formée d'une séquence de A suivie d'un nombre quelconque (éventuellement nul) de séquences de B , suivies d'une séquence de C .

La démonstration des formules ci-dessus est un simple exercice d'application de formules précédentes. Il suffit d'éliminer les signes $*$ au moyen de (4.27), (4.28) et d'appliquer notamment (4.15), (4.7), (4.17), (4.8).

4.1.19 Lemme

Si A et S sont deux langages sur un alphabet X , alors

$$A \subset SA \implies A = \emptyset. \quad (4.38)$$

Autrement dit le seul langage A vérifiant la relation $A \subset SA$ est le langage $A = \emptyset$.

4.1.20 Démonstration

Supposons que $A \subset SA$. Désignons par A_n l'ensemble des séquences $x \in A$ de longueur n , ceci pour tout entier $n \geq 1$. Nous allons montrer que $A_n = \emptyset$ quel que soit n . Premièrement $A_1 = \emptyset$, car l'hypothèse $A \subset SA$ implique $lg(x) \geq 2$ pour toute séquence $x \in A$ (4.11). Par ailleurs, notre hypothèse permet d'écrire successivement par (4.6) :

$$A \subset SA \subset S^2A \subset \dots \subset S^nA \subset \dots$$

En vertu de (4.11), on a donc $lg(x) \geq n + 1$ pour toute séquence $x \in A$, donc $A_n = \emptyset$, et ceci quel que soit n . Par suite $A = \emptyset$.

4.1.21 Proposition

Si R, S, T sont des langages sur un alphabet X , on a l'équivalence

$$R = SR \cup T \iff R = S^*T. \quad (4.39)$$

4.1.22 Démonstration

Supposons d'abord que $R = S^*T$. Alors $SR \cup T = S(S^*T) \cup T = (SS^*)T \cup T = S^+T \cup T = S^*T = R$ par (4.36), (4.31), (4.27).

Supposons inversement que

$$R = SR \cup T. \quad (4.40)$$

Nous en déduisons premièrement que

$$S^*T \subset R.$$

En effet, en appliquant alternativement (4.40) et (4.6), on peut écrire successivement $T \subset R, ST \subset SR, ST \subset R, S^2T \subset SR, S^2T \subset R$, et ainsi $S^nT \subset R$ quel que soit n , d'où $S^*T \subset R$. Il reste à montrer que $R = S^*T$. Pour cela désignons par A l'ensemble des séquences $x \in R$ qui n'appartiennent pas au langage S^*T . Le langage A est le complément de l'ensemble S^*T par rapport à l'ensemble R :

$$x \in A \iff x \in R \text{ et } x \notin S^*T \quad (4.41)$$

Pour montrer que $R = S^*T$, il suffit de montrer que $A = \emptyset$. A cet effet nous établirons que $A \subset SA$ et nous appliquerons (4.38). Soit donc $x \in A$. On a $x \in R$ et $x \notin S^*T$ par (4.41), donc $x \in SR$ par (4.40). Par suite x se décompose en un produit $x = x_1x_2$ avec $x_1 \in S$ et $x_2 \in R$. De plus $x_2 \notin S^*T$, sinon l'on aurait $x \in S^*T$. Donc $x_2 \in A$, et par suite $x \in SA$. Ceci prouve que $A \subset SA$.

4.1.23 Corollaire

Si S et T sont deux langages sur un alphabet X , il existe un langage R sur X et un seul tel que $R = SR \cup T$, et ce langage est régulièrement engendré par S, T .

En effet selon (4.39), la relation $R = SR \cup T$ admet une solution R et une seule, à savoir $R = S^*T$. Ce corollaire est généralisé au paragraphe 4.1.25.

4.1.24 Corollaire

Si R et S sont deux langages sur un alphabet X ,

$$R = SR \iff R = \emptyset. \quad (4.42)$$

En effet la relation $R = SR$ peut s'écrire $R = SR \cup \emptyset$. Elle équivaut à $R = S^*\emptyset$, donc à $R = \emptyset$ par (4.39) et (4.34).

4.1.25 Proposition

Soient $S_{ij}, T_i (i, j = 1, \dots, n)$ des langages sur un alphabet X , et considérons le système de relations

$$\left. \begin{aligned} R_1 &= S_{11} R_1 \cup S_{12} R_2 \cup \dots \cup S_{1n} R_n \cup T_1 \\ R_2 &= S_{21} R_1 \cup S_{22} R_2 \cup \dots \cup S_{2n} R_n \cup T_2 \\ &\vdots \\ R_n &= S_{n1} R_1 \cup S_{n2} R_2 \cup \dots \cup S_{nn} R_n \cup T_n. \end{aligned} \right\} \quad (4.43)$$

Ce système admet une solution R_1, \dots, R_n (langages sur X) et une seule, et chacun des langages R_1, \dots, R_n de cette solution est régulièrement engendré par l'ensemble des langages S_{ij}, T_i .

Nous dirons que (4.43) est un *système d'équations régulières linéaires* d'ordre n en R_1, \dots, R_n .

4.1.26 Démonstration

La proposition est vraie pour un système d'ordre 1 (§ 4.1.23). Nous raisonnons par récurrence. Supposons que la proposition soit vraie pour tout système d'ordre $n - 1$, n étant un entier > 1 , et considérons un système (4.43) d'ordre n . Nous pouvons écrire sa dernière équation sous la forme $R_n = S_{nn} R_n \cup T$, avec

$$T = S_{n1} R_1 \cup \dots \cup S_{nn-1} R_{n-1} \cup T_n.$$

Cette équation est équivalente à $R_n = S_{nn}^* T$ (4.39), soit à

$$R_n = S_{nn}^* S_{n1} R_1 \cup \dots \cup S_{nn}^* S_{nn-1} R_{n-1} \cup S_{nn}^* T_n. \quad (4.44)$$

Si l'on élimine R_n au moyen de (4.44) dans les $n - 1$ premières équations de (4.43), on obtient un système d'équations linéaires d'ordre $n - 1$ en R_1, \dots, R_{n-1} . Ce système aura la forme

$$R_i = S'_{i1} R_1 \cup \dots \cup S'_{in-1} R_{n-1} \cup T'_i \quad (i = 1, \dots, n - 1) \quad (4.45)$$

avec

$$\left. \begin{aligned} S'_{ij} &= S_{ij} \cup S_{in} S_{nn}^* S_{nj} \\ T'_i &= T_i \cup S_{in} S_{nn}^* T_n \end{aligned} \right\} \quad (i, j = 1, \dots, n - 1). \quad (4.46)$$

Par hypothèse, le système (4.45) admet une solution R_1, \dots, R_{n-1} et une seule, et les langages R_1, \dots, R_{n-1} de cette solution sont régulièrement engendrés par l'ensemble des langages S'_{ij}, T'_i . Or ceux-ci sont régulièrement engendrés par l'ensemble des S_{ij}, T_i selon (4.46). Il s'ensuit que R_1, \dots, R_{n-1} sont régulièrement engendrés par l'ensemble des S_{ij}, T_i (§ 4.1.14). En portant cette solution R_1, \dots, R_{n-1} dans (4.44), on détermine R_n , qui se trouve à son tour régulièrement engendré par l'ensemble des S_{ij}, T_i .

4.1.27 Définition : langages réguliers

On appelle *langages élémentaires* sur un alphabet $X = \{a_1, \dots, a_n\}$ le langage vide \emptyset et les n langages $\{a_1\}, \dots, \{a_n\}$ comportant chacun une seule séquence (de longueur 1).

On appelle *langage régulier* sur X tout langage qui est régulièrement engendré (§ 4.1.10) par l'ensemble des langages élémentaires. Autrement dit, l'ensemble des lan-

gages réguliers sur X est l'ensemble $LR(\{a_1\}, \dots, \{a_n\}, \emptyset)$ au sens du paragraphe 4.1.10. Un langage sur X est régulier si et seulement s'il peut être construit à partir des langages élémentaires par une succession d'opérations régulières.

4.1.28 Expressions régulières

On convient de représenter les langages élémentaires $\{a_1\}, \dots, \{a_n\}$ sur l'alphabet $X = \{a_1, \dots, a_n\}$ par les expressions a_1, \dots, a_n . Cela revient à assimiler la séquence a_i au langage $\{a_i\}$. Cette notation adoptée, il est clair que tout langage régulier sur l'alphabet X peut être représenté par une expression formée au moyen des seuls symboles $a_1, \dots, a_n, \emptyset, \cup, +$ et de parenthèses. L'opération de produit est notée sans symbole spécifique. On peut utiliser encore le symbole $*$, qui est commode mais non indispensable (§ 4.1.15).

Ces expressions sont appelées *expressions régulières* sur a_1, \dots, a_n , en abrégé $ER(a_1, \dots, a_n)$. On peut préciser leur forme au moyen de règles d'écritures analogues à celles des expressions booléennes (§ 3.1.4).

Les expressions $\emptyset, a_1, \dots, a_n$ sont des $ER(a_1, \dots, a_n)$. (4.47)

Si F, G sont deux $ER(a_1, \dots, a_n)$, alors l'expression $(F \cup G)$ est une $ER(a_1, \dots, a_n)$. (4.48)

Si F, G sont deux $ER(a_1, \dots, a_n)$, alors l'expression (FG) est une $ER(a_1, \dots, a_n)$. (4.49)

Si F est une $ER(a_1, \dots, a_n)$, alors l'expression F^+ est une $ER(a_1, \dots, a_n)$. (4.50)

Si F, G sont deux $ER(a_1, \dots, a_n)$, alors les expressions (F^*G) et (FG^*) sont des $ER(a_1, \dots, a_n)$. (4.51)

Considérons par exemple l'alphabet $X = \{0, 1, 2\}$. L'expression

$((0 \cup (2^*1)^+)(01))$ (4.52)

est une expression régulière sur cet alphabet, comme le montre la figure 4.2.

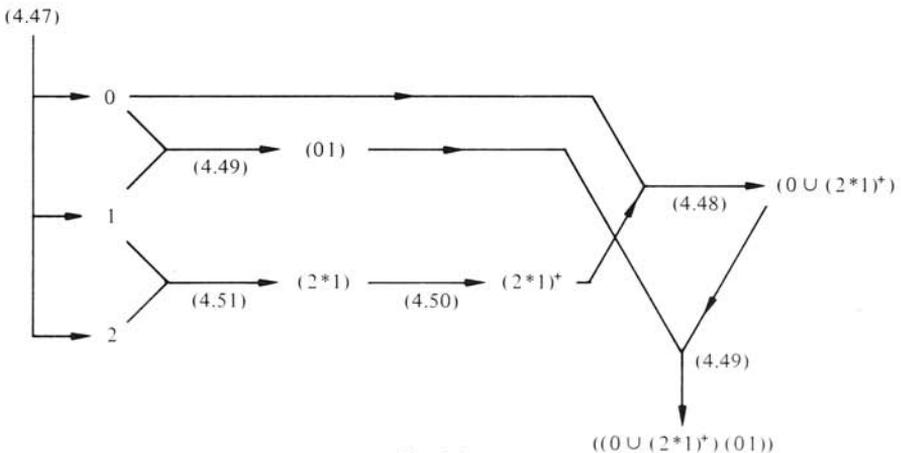


Fig. 4.2

Les règles ci-dessus font un large emploi de parenthèses. Pratiquement on omet les parenthèses extérieures d'une expression ainsi que les parenthèses qui sont superflues en vertu de l'associativité de la réunion et du produit. Ainsi l'expression (4.52) s'abrège $(0 \cup (2^* 1)^+)$ 01. En outre, on omet des parenthèses en convenant que les opérations de produit sont effectuées avant les réunions. Par exemple l'expression $0 \cup 21$ signifie $0 \cup (21)$ et non $(0 \cup 2)1$.

4.1.29 Exemples

Nous considérons l'alphabet $X = \{0, 1, 2\}$ pour fixer les idées, et quelques exemples de langages réguliers sur cet alphabet.

Un langage comportant une seule séquence x est régulier. En effet si $lg(x) = 1$ ce langage est élémentaire. Si $lg(x) = n > 1$, le langage s'obtient par $n - 1$ produits à partir des langages élémentaires. Par exemple le langage $\{201\}$ s'obtient en effectuant le produit $\{2\} \{0\} = \{20\}$, puis le produit $\{20\} \{1\} = \{201\}$. Ce langage est représenté par l'expression régulière 201.

Un langage fini $\{x_1, \dots, x_n\}$, c'est-à-dire composé d'un nombre fini n de séquences est régulier. Il s'obtient en effet à partir des langages réguliers $\{x_1\}, \dots, \{x_n\}$ par $n - 1$ opérations de réunion. On le notera par l'expression régulière $x_1 \cup \dots \cup x_n$. Par exemple le langage $X = \{0, 1, 2\}$ lui-même peut être noté par l'expression régulière $0 \cup 1 \cup 2$. Le langage $\{02, 111\}$ est noté par l'expression régulière $02 \cup 111$.

Il est clair que si, partant des langages élémentaires, on n'effectue que des réunions et/ou des produits, on ne peut obtenir que des langages réguliers finis. Ainsi l'expression régulière $(02 \cup 111)(0 \cup 2)$ représente le langage fini $\{020, 022, 1110, 1112\}$. Il faut utiliser l'opération d'itération pour engendrer un langage régulier infini. Par exemple l'expression 01^+ représente l'ensemble des séquences de la forme 01^n : 01, 011, 0111, ...

Un langage régulier peut être défini autrement que par une expression régulière. Considérons par exemple la phrase suivante, définissant un langage A sur l'alphabet X : A est l'ensemble des séquences qui possèdent la terminaison 01. Cette phrase définit un langage, mais on ne peut affirmer qu'il est régulier que lorsqu'on a réussi à le représenter par une expression régulière montrant par quelle succession d'opérations régulières A se trouve engendré par les langages élémentaires. Ceci ne présente pas de difficulté dans l'exemple choisi. Les séquences qui possèdent la terminaison 01 sont : la séquence 01, et toutes les séquences de la forme $x01$ où x est une séquence quelconque sur X . L'ensemble des séquences x quelconques est le langage $X^+ = (0 \cup 1 \cup 2)^+$. L'ensemble des séquences de la forme $x01$ est le langage régulier $(0 \cup 1 \cup 2)^+ 01$. Finalement, A est le langage $(0 \cup 1 \cup 2)^+ 01 \cup 01$. On peut le noter aussi $(0 \cup 1 \cup 2)^* 01$. Signalons que certaines descriptions de langages réguliers peuvent être beaucoup plus difficiles à traduire par une expression régulière.

Ces exemples amènent immédiatement la question : existe-t-il des langages non réguliers? La réponse est affirmative. Un exemple classique est l'ensemble des séquences de la forme $0^n 1^n$, c'est-à-dire 01, 0011, 000111, ... On peut s'essayer vainement à représenter ce langage par une expression régulière. Toutefois, l'insuccès d'une telle tentative ne constitue pas une preuve de la non régularité du langage. Cette preuve repose sur d'autres notions et sera reportée au paragraphe 4.2.20.

4.1.30 Exercice

Représenter par une expression régulière chacun des langages A_1, \dots, A_{10} sur l'alphabet $\{0, 1, 2\}$ décrits ci-dessous.

- A_1 : ensemble des séquences qui possèdent la terminaison 100.
- A_2 : ensemble des séquences qui possèdent le préfixe 100 (§ 4.1.2).
- A_3 : ensemble des séquences qui possèdent une syllabe 2 (§ 4.1.2).
- A_4 : ensemble des séquences qui possèdent une syllabe 2 et une seule.
- A_5 : ensemble des séquences qui ne possèdent aucune syllabe 2.
- A_6 : ensemble des séquences de longueur ≥ 3 .
- A_7 : ensemble des séquences qui comportent un nombre pair de 1.
- A_8 : ensemble des séquences qui comportent un nombre impair de 1.
- A_9 : ensemble des séquences qui ne possèdent aucune syllabe 01.
- A_{10} : ensemble des séquences qui possèdent une syllabe 01 et une seule.

4.1.31 Solution

$$A_1 = (0 \cup 1 \cup 2)^* 100.$$

$$A_2 = 100(0 \cup 1 \cup 2)^*$$

$$A_3 = (0 \cup 1 \cup 2)^* 2(0 \cup 1 \cup 2)^*.$$

Les séquences de A_3 sont de la forme $2, x2, 2y, x2y$, où x et y sont des séquences quelconques. On a donc, en posant $X = 0 \cup 1 \cup 2$: $A_3 = 2 \cup X^* 2 \cup 2 X^* \cup X^* 2 X^*$, ce qui se transforme par (4.27), (4.28) en l'expression ci-dessus.

$$A_4 = (0 \cup 1)^* 2(0 \cup 1)^*.$$

$$A_5 = (0 \cup 1)^+.$$

$$A_6 = (0 \cup 1 \cup 2)(0 \cup 1 \cup 2)^+.$$

$$A_7 = (0 \cup 2 \cup 1(0 \cup 2)^* 1)^+.$$

Une séquence de A_7 admet toujours une décomposition en syllabes de la forme $0, 2, 11, 1x1$, où x est une séquence quelconque ne comportant pas de 1. Par exemple la séquence 201201011 admet la décomposition $(2, 0, 1201, 0, 11)$. Ceci se traduit par l'expression régulière ci-dessus. A noter que les séquences qui ne comportent pas de 1 font partie du langage A_7 (le nombre 0 est pair).

$$A_8 = (0 \cup 2)^* 1(0 \cup 2 \cup 1(0 \cup 2)^* 1)^*.$$

Les séquences de A_8 sont de la forme $1, x1, 1y, x1y$, où x est une séquence quelconque qui ne comporte pas de 1, et y est une séquence quelconque comportant un nombre pair (éventuellement nul) de 1. Ceci se traduit par l'expression ci-dessus.

$$A_9 = 0^+(21^*0^*)^* \cup 1^+0^*(21^*0^*)^* \cup (21^*0^*)^+.$$

Le langage A_9 est exprimé comme la réunion de trois langages, à savoir les trois ensembles de séquences ne comportant aucune syllabe 01, et commençant respectivement par 0, par 1, et par 2. Désignons ces trois langages par R_0, R_1, R_2 . L'expression $R_2 = (21^*0^*)^+$ est facile à comprendre. L'ensemble $P = 21^*0^*$ est l'ensemble des séquences qui commencent par 2, qui ne comportent qu'un seul 2 et aucune syllabe 01.

Il est clair que toute séquence de R_2 admet une décomposition en syllabes du type P , d'où $R_2 = P^+$. Une séquence de l'ensemble R_0 appartient au langage 0^+ si elle ne comporte pas de 2, et au langage 0^+R_2 si elle comporte un 2. D'où l'expression $R_0 = 0^+(21^*0^*)^*$. L'expression de R_1 s'explique de façon analogue.

$$A_{10} = 01 \cup A_9 01 \cup 01 A_9 \cup A_9 01 A_9.$$

En effet les séquences de A_{10} sont de la forme $01, x01, 01y, x01y$, où x, y sont des séquences qui ne comportent pas de syllabe 01 .

4.1.32 Exercice

En s'inspirant du paragraphe 4.1.26, résoudre le système d'équations régulières linéaires

$$R_1 = 1 R_1 \cup 0 R_2$$

$$R_2 = 1 R_1 \cup 2 R_2 \cup 0.$$

On obtiendra pour R_1, R_2 des expressions régulières sur l'alphabet $\{0,1,2\}$.

4.1.33 Solution

Résolvons la première équation en R_1 par (4.39). Il vient $R_1 = 1^*0R_2$. Éliminons R_1 dans la deuxième équation : $R_2 = 11^*0R_2 \cup 2R_2 \cup 0 = (1^*0 \cup 2)R_2 \cup 0$. Tirons-en par (4.39) : $R_2 = (1^*0 \cup 2)^*0$. D'où $R_1 = 1^*0(1^*0 \cup 2)^*0$.

4.2 DIAGRAMMES RÉGULIERS

4.2.1 Introduction; exemples

Pour fixer les idées, nous considérons généralement dans cette section l'alphabet $X = \{0,1,2\}$. La généralisation est immédiate. Les langages réguliers sur X peuvent être représentés de façon très suggestive au moyen de diagrammes que nous appelons *diagrammes réguliers*, dont la forme est empruntée à Wirth [14], et rappelle celle de réseaux ferroviaires. La figure 4.3 en donne un exemple. On y voit essentiellement des *places* (cercles) contenant chacune un des symboles 0, 1, 2 de l'alphabet considéré. Ces places sont situées sur des voies à sens unique (sens des flèches), qui divergent ou convergent par des aiguillages. Les places sont désignées par a, b, c, d, e . Le diagramme représente un langage A sur l'alphabet X , et ceci de la façon suivante. Les séquences $x \in A$ sont les séquences que l'on obtient en parcourant sur les voies, depuis l'entrée E jusqu'à la sortie S , un chemin compatible avec le sens des voies, et en énumérant les symboles 0, 1, 2 rencontrés successivement dans ce parcours. On peut effectuer par exemple le parcours $\rightarrow a \rightarrow b \rightarrow d \rightarrow b \rightarrow d \rightarrow$, suivant lequel on obtient la séquence $x = 00202$. Celle-ci appartient donc au langage A , et seules les séquences obtenues de cette manière appartiennent à A . Vu la boucle fermée que possède le diagramme, il est clair que le langage A est infini. Si A est régulier, il doit y avoir une opération d'itération dans toute expression régulière de A . On a par exemple, de façon évidente $A = (0 \cup 02)^+ \cup 12$.

Nous fixons ainsi les buts de la présente section : premièrement de montrer que tout langage régulier est représentable par un diagramme régulier; secondement de montrer que tout diagramme régulier représente un langage régulier. Ce faisant, nous don-

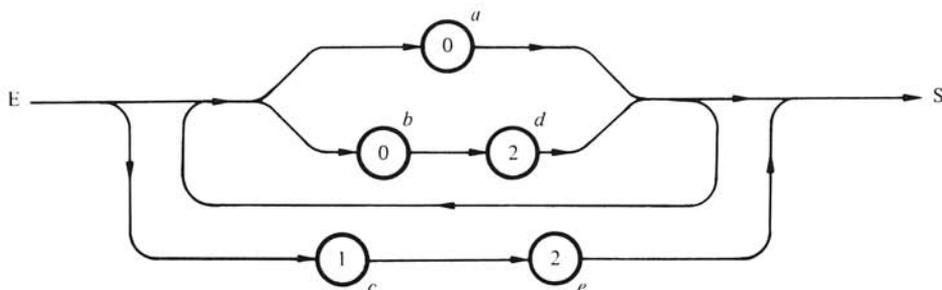


Fig. 4.3

nerons des procédés de passage d'une expression régulière à un diagramme régulier, et vice-versa.

L'énoncé même de ces objectifs indique qu'il nous faut préciser ce qu'est un diagramme régulier, et comment il est interprété en tant que représentation d'un langage. C'est le but des paragraphes suivants.

Auparavant, il est suggéré au lecteur comme exercice, de représenter les langages du paragraphe 4.1.30 par des diagrammes réguliers. Par exemple le langage A_9 peut être représenté par le diagramme de la figure 4.4, et cet exemple montre qu'un diagramme régulier peut être plus simple et plus clair qu'une expression régulière.

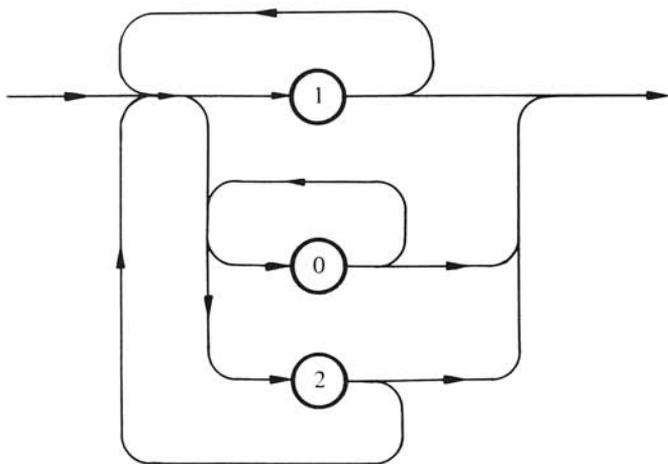


Fig. 4.4

4.2.2 Définitions

Considérons le diagramme de la figure 4.3, et désignons par $P = \{a, b, c, d, e\}$ l'ensemble de ses places. Certaines places sont directement accessibles de l'entrée E, sans passer par d'autres places. Ce sont les places a, b, c , et nous les appellerons *places initiales*. Symétriquement, les places a, d, e sont dites *terminales*, car depuis ces places on peut accéder directement à la sortie S sans passer par d'autres places.

Le diagramme associé à chaque place p un ensemble de places auxquelles on peut accéder directement depuis p . Nous désignerons cet ensemble par $p \cdot$. Par exemple de la place a on peut accéder directement à la place b ou à la place a elle-même. Nous écrivons donc $a \cdot = \{a, b\}$. De la place e on ne peut accéder à aucune place. L'ensemble $e \cdot$ est vide. Enfin chaque place p contient un symbole 0,1,2 que nous désignerons par $\mu(p)$ et appellerons la *marque* de p .

Toutes ces informations sont rassemblées dans le tableau 4.5. Dans sa première colonne, les places initiales sont notées $\rightarrow p$, les places terminales $\leftarrow p$, et les places à la fois initiales et terminales $\leftrightarrow p$. Il se trouve dans cet exemple que toute place est initiale ou/et terminale. C'est évidemment un cas particulier. Si l'on insérait une place supplémentaire entre c et e dans le diagramme, elle ne serait ni initiale ni terminale.

p	$p \cdot$	$\mu(p)$
$\leftrightarrow a$	$\{a, b\}$	0
$\rightarrow b$	$\{d\}$	2
$\rightarrow c$	$\{e\}$	1
$\leftarrow d$	$\{a, b\}$	2
$\leftarrow e$	\emptyset	2

Tableau 4.5

Nous inversons maintenant l'ordre dans lequel la figure 4.3 et le tableau 4.5 ont été pris en considération. Les données essentielles du diagramme sont dans le tableau 4.5. La figure 4.3 n'est qu'un moyen graphique de définir ces données. Toute autre figure qui conduirait au même tableau serait considérée comme le même diagramme régulier. Nous ne cherchons donc pas à définir la notion de diagramme régulier par le moyen de quelconques normes graphiques. N'importe quel dessin est considéré comme un diagramme régulier s'il définit sans ambiguïté un tableau de la forme ci-dessus. Ceci conduit aux définitions qui suivent.

4.2.3 Définitions

Un *diagramme régulier* D sur l'alphabet X se compose des données suivantes :

- un ensemble fini et non vide P appelé *ensemble des places*;
- un sous-ensemble $\rightarrow P \subset P$ appelé *ensemble des places initiales*;
- un sous-ensemble $\leftarrow P \subset P$ appelé *ensemble des places terminales*;
- une correspondance $\varphi : P \rightarrow P$ qui associe à chaque place p un sous-ensemble $\varphi(p) \subset P$, noté $p \cdot$ et appelé *ensemble des places suivantes* de p ;
- une application $\mu : P \rightarrow X$ qui associe à chaque place p un élément $\mu(p) \in X$ appelé *marque* de p .

On désigne le diagramme D par le 5-uple $(P, \rightarrow P, \leftarrow P, \varphi, \mu)$.

Un *parcours* du diagramme est une *séquence de places* $y(1, n)$ vérifiant les trois conditions suivantes :

$$y(1) \in \rightarrow P \tag{4.53}$$

$$y(n) \in \leftarrow P \quad (4.54)$$

$$\text{si } n > 1, \text{ alors } y(k+1) \in y(k) \cdot \text{ pour } k = 1, \dots, n-1. \quad (4.55)$$

Pour tout parcours $y(1, n)$ nous désignons par $\mu(y)$ la séquence $\mu(y_1) \dots \mu(y_n)$ sur l'alphabet X . Cette séquence $\mu(y)$ est appelée la *marque* du parcours y .

Le langage A représenté par le diagramme est l'ensemble des marques de tous les parcours du diagramme. Autrement dit, une séquence $x \in X^+$ appartient au langage A si et seulement s'il existe un parcours y tel que $x = \mu(y)$.

4.2.4 Proposition

Tout langage élémentaire (§ 4.1.27) est représentable par un diagramme régulier.

En effet on peut représenter le langage vide par n'importe quel diagramme régulier n'admettant aucun parcours; il suffit par exemple que l'ensemble $\rightarrow P$ des places initiales soit vide, c'est-à-dire qu'aucune place ne soit désignée comme initiale. Un langage élémentaire non vide, par exemple le langage 0 sur l'alphabet $\{0, 1, 2\}$, peut être représenté par un *diagramme élémentaire* (fig. 4.6). Ce diagramme possède une place et une seule, initiale et terminale, de marque 0 , et sans places suivantes. Il admet un parcours et un seul, de marque 0 .

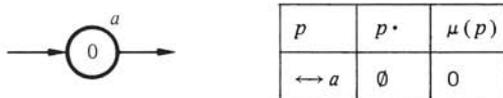


Fig. 4.6

4.2.5 Proposition

Si deux langages A, B sur l'alphabet X sont représentables chacun par un diagramme régulier, alors le langage $A \cup B$ est représentable par un diagramme régulier.

4.2.6 Démonstration

On suppose que A et B sont représentés par des diagrammes respectifs $D_A = (P_A, \rightarrow P_A, \leftarrow P_A, \varphi_A, \mu_A)$ et $D_B = (P_B, \rightarrow P_B, \leftarrow P_B, \varphi_B, \mu_B)$, tels que $P_A \cap P_B = \emptyset$ (pas de place commune). On peut alors représenter le langage $A \cup B$ par le diagramme

$$D = \begin{array}{c} \text{---} \text{---} \text{---} \\ \text{---} \text{---} \text{---} \end{array} \begin{array}{l} D_A \\ D_B \end{array} \quad (4.56)$$

La notation (4.56) signifie que le diagramme $D = (P, \rightarrow P, \leftarrow P, \varphi, \mu)$ est la "réunion" des diagrammes D_A, D_B . Plus précisément, ce diagramme D est défini par les relations

$$P = P_A \cup P_B; \quad \rightarrow P = \rightarrow P_A \cup \rightarrow P_B; \quad \leftarrow P = \leftarrow P_A \cup \leftarrow P_B; \quad (4.57)$$

et par les relations suivantes, valables pour toute place $p \in P$:

$$\varphi(p) = \begin{cases} \varphi_A(p) & \text{si } p \in P_A, \\ \varphi_B(p) & \text{si } p \in P_B, \end{cases} \quad \mu(p) = \begin{cases} \mu_A(p) & \text{si } p \in P_A \\ \mu_B(p) & \text{si } p \in P_B. \end{cases} \quad (4.58)$$

Il est clair qu'un parcours de D est soit un parcours de D_A , soit un parcours de D_B , et

La notation (4.64) signifie que le diagramme $D = (P, \rightarrow P, \leftarrow P, \varphi, \mu)$ est construit de la façon suivante :

$$P = P_A; \quad \rightarrow P = \rightarrow P_A; \quad \leftarrow P = \leftarrow P_A; \quad (4.65)$$

et pour toute place p ,

$$\varphi(p) = \begin{cases} \varphi_A(p) & \text{si } p \notin \leftarrow P_A \\ \varphi_A(p) \cup \rightarrow P_A & \text{si } p \in \leftarrow P_A \end{cases} \quad (4.66)$$

$$\mu(p) = \mu_A(p). \quad (4.67)$$

Il est intuitivement évident que le langage B représenté par D est le langage $B = A^+$. Pour le prouver, on peut remarquer qu'un parcours de D est soit un parcours de D_A , soit un parcours de D_A prolongé par un nouveau parcours de D . Ceci se traduit par la relation $B = AB \cup A$ entre les langages A et B , d'où l'on tire $B = A^*A = A^+$ par (4.39) et (4.31).

4.2.11 Proposition

Tout langage régulier sur un alphabet X peut être représenté par un diagramme régulier sur X .

4.2.12 Démonstration

Par définition (§ 4.1.27), un langage sur l'alphabet X est régulier s'il peut être construit à partir des langages élémentaires par une succession d'opérations régulières. En appliquant les constructions des paragraphes 4.2.4, 4.2.6, 4.2.8, 4.2.10, on pourra toujours représenter un tel langage par un diagramme régulier.

Pour raisonner plus formellement, désignons par Λ l'ensemble des langages élémentaires sur X . Il s'agit de montrer que pour tout entier $n \geq 0$, les langages de l'ensemble $\mathbf{LR}_n(\Lambda)$ (§ 4.1.10) sont représentables par des diagrammes réguliers. Pour $n = 0$ cela résulte du paragraphe 4.2.4. En supposant le fait établi pour n , on peut représenter par un diagramme régulier n'importe quel langage de $\mathbf{LR}_{n+1}(\Lambda)$, en vertu des paragraphes 4.2.5, 4.2.7, 4.2.9.

4.2.13 Exemple

La donnée d'une expression régulière pour un langage A sur un alphabet X , permet une construction systématique d'un diagramme régulier représentant A , par une succession d'opérations de la forme (4.56), (4.59), (4.64).

Soit à représenter par exemple le langage $A = (01 \cup 2)^+ 01$ sur l'alphabet $X = \{0, 1, 2\}$. La figure 4.7 montre la succession des opérations de construction du diagramme.

4.2.14 Commentaires

La démarche illustrée par la figure 4.7 est une démarche standard, toujours applicable à une expression régulière donnée, mais ne fournissant pas toujours le diagramme le plus simple.

Langage	Diagramme	Opération
0		
1		
01		(4.59)
2		
$01 \cup 2$		(4.56)
$(01 \cup 2)^+$		(4.64)
$(01 \cup 2)^+ 01$		(4.59)

Fig. 4.7

Considérons par exemple une expression de la forme $A \cup AB$, où A et B sont deux langages réguliers. La démarche standard conduit à un diagramme de la forme



où D_A, D'_A sont deux diagrammes semblables représentant chacun le langage A , et D_B un diagramme représentatif de B . Or la notation



suggère un diagramme plus simple.

Considérons de même l'expression A^*B . Pour appliquer la démarche standard, il faut écrire $A^*B = A^+B \cup B$. On est alors conduit à un diagramme de la forme



Ici aussi, le même langage A^*B peut être représenté plus simplement par un diagramme



4.2.15 Exercice

Soient A et B deux langages sur un alphabet X , et $D_A = (P_A, \rightarrow P_A, \leftarrow P_A, \varphi_A, \mu_A)$, $D_B = (P_B, \rightarrow P_B, \leftarrow P_B, \varphi_B, \mu_B)$ deux diagrammes qui représentent ces langages. On suppose que $P_A \cap P_B = \emptyset$. Définir de façon précise le diagramme $D = (P, \rightarrow P, \leftarrow P, \varphi, \mu)$ représenté par la notation (4.69), et celui représenté par (4.71).

4.2.16 Définition

Soit $D = (P, \rightarrow P, \leftarrow P, \varphi, \mu)$ un diagramme régulier sur un alphabet X . Nous appellerons *trajet* dans D toute séquence de places $y(1, n)$ vérifiant les deux conditions

$$y(n) \in \leftarrow P; \quad (4.72)$$

$$\text{si } n > 1, \quad \text{alors } y(k+1) \in y(k) \cdot \text{ pour } k = 1, \dots, n-1. \quad (4.73)$$

La place $y(1)$ est appelée *place de départ* du trajet. Elle peut être quelconque. Si elle est initiale, c'est-à-dire si $y(1) \in \rightarrow P$, le trajet est un parcours de D (§ 4.2.3). Un trajet $y(1, n)$ est une séquence sur l'alphabet P . La séquence $\mu(y(1)) \mu(y(2)) \dots \mu(y(n))$, notée $\mu(y)$, est la *marque* du trajet.

4.2.17 Exemple

Considérons le diagramme de la figure 4.8, reproduisant la figure 4.3. Pour chaque place p , désignons par R_p l'ensemble des trajets de départ p . On a les relations suivantes :

$$\left. \begin{aligned} R_a &= a R_a \cup a R_b \cup a \\ R_b &= b R_d \\ R_c &= c R_e \\ R_d &= d R_a \cup d R_b \cup d \\ R_e &= e. \end{aligned} \right\} \quad (4.74)$$

La première relation s'explique ainsi : les trajets de départ a sont la séquence a et les séquences de la forme ay où y est un trajet de départ a ou b (car a, b sont les places suivantes de a). Les autres relations s'expliquent de façon semblable.

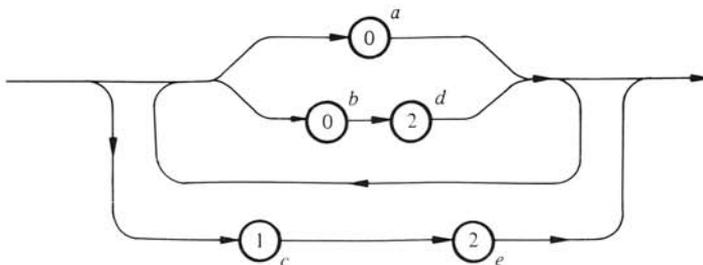


Fig. 4.8

Si maintenant nous désignons par R_p non plus l'ensemble des trajets de départ p , mais l'ensemble des marques de ces trajets, il vient :

$$\left. \begin{aligned} R_a &= 0R_a \cup 0R_b \cup 0 \\ R_b &= 0R_d \\ R_c &= 1R_e \\ R_d &= 2R_a \cup 2R_b \cup 2 \\ R_e &= 2. \end{aligned} \right\} \quad (4.75)$$

Nous voyons que les langages R_a, \dots, R_e vérifient un système d'équations régulières linéaires, dont les coefficients sont des langages élémentaires sur l'alphabet $X = \{0, 1, 2\}$. On peut écrire ce système sous la forme plus homogène :

$$\begin{aligned} R_a &= 0R_a \cup 0R_b \cup \emptyset R_c \cup \emptyset R_d \cup \emptyset R_e \cup 0 \\ R_b &= \emptyset R_a \cup \emptyset R_b \cup \emptyset R_c \cup 0R_d \cup \emptyset R_e \cup \emptyset \\ &\dots \end{aligned}$$

En vertu du paragraphe 4.1.25, les langages R_a, \dots, R_e sont réguliers. Par suite, le langage représenté par le diagramme est régulier, puisque c'est le langage $R = R_a \cup R_b \cup R_c$.

Le traitement de cet exemple est généralisé par la proposition suivante, que nous admettons comme évidente.

4.2.18 Proposition

Soient p_1, \dots, p_n les places d'un diagramme régulier $D = (P, \rightarrow P, \leftarrow P, \varphi, \mu)$ sur l'alphabet X , et pour toute place p_i soit R_i l'ensemble des marques des trajets de départ p_i dans D . Les langages R_i vérifient le système d'équations régulières linéaires

$$R_i = \bigcup_{j=1}^n S_{ij} R_j \cup T_i \quad (i = 1, \dots, n) \quad (4.76)$$

avec

$$S_{ij} = \begin{cases} \mu(p_i) & \text{si } p_j \in \varphi(p_i) \\ \emptyset & \text{si } p_j \notin \varphi(p_i) \end{cases} \quad (i, j = 1, \dots, n) \quad (4.77)$$

et

$$T_i = \begin{cases} \mu(p_i) & \text{si } p_i \in \leftarrow P \\ \emptyset & \text{si } p_i \notin \leftarrow P \end{cases} \quad (i = 1, \dots, n). \quad (4.78)$$

Par suite les langages R_i sont réguliers (car régulièrement engendrés (§ 4.1.25) par les langages élémentaires S_{ij} (4.77), T_i (4.78)). Enfin, le langage représenté par D , à savoir le langage

$$R = \bigcup_{p_i \in \rightarrow P} R_i \quad (4.79)$$

est régulier.

4.2.19 Commentaire

La proposition précédente fournit un procédé permettant de trouver une expression régulière pour le langage représenté par un diagramme régulier : il suffit de résoudre le système (4.76) et de former le langage (4.79).

Signalons cependant que la façon de résoudre le système (4.76) a une grande influence sur l'expression finale obtenue. Considérons par exemple le système (4.75), relatif à la figure 4.8. En éliminant R_d et R_e , on obtient

$$\begin{aligned} R_a &= 0R_a \cup 0R_b \cup 0 \\ R_b &= 02R_a \cup 02R_b \cup 02 \\ R_c &= 12. \end{aligned}$$

On s'intéresse au langage $R_a \cup R_b \cup R_c$ représenté par le diagramme, et l'on peut écrire

$$R_a \cup R_b = (0 \cup 02)R_a \cup (0 \cup 02)R_b \cup (0 \cup 02)$$

en réunissant les deux premières équations qui précèdent. Par suite il vient

$$R_a \cup R_b = (0 \cup 02)(R_a \cup R_b) \cup (0 \cup 02)$$

et en appliquant (4.39) :

$$\begin{aligned} R_a \cup R_b &= (0 \cup 02)^*(0 \cup 02) \\ &= (0 \cup 02)^+. \end{aligned}$$

Le langage représenté par la figure 4.8 est donc $(0 \cup 02)^+ \cup 12$. On trouverait une expression équivalente, mais beaucoup plus compliquée si l'on cherchait à calculer séparément R_a et R_b .

4.2.20 Exemple de langage non régulier

Soit A l'ensemble des séquences de la forme $0^n 1^n$ ($n \geq 1$) sur l'alphabet $\{0,1\}$, c'est-à-dire l'ensemble des séquences 01, 0011, 000111, ... Nous allons montrer par l'absurde que ce langage ne peut pas être représenté par un diagramme régulier, et que par conséquent (§ 4.2.11) il n'est pas régulier.

Supposons qu'il existe un diagramme régulier D qui représente le langage A . Pour chaque entier $n \geq 1$ soit y_n un parcours de marque $0^n 1^n$ dans ce diagramme. Le parcours y_n est de longueur $2n$ et s'écrit

$$y_n = y_n(1, 2n) = y_n(1, n)y_n(n+1, 2n)$$

la marque de $y_n(1, n)$ étant 0^n , et celle de $y_n(n+1, 2n)$ étant 1^n . Considérons pour chaque entier $n \geq 1$ la n -ième place $y_n(n)$ du parcours y_n correspondant. Le nombre de places d'un diagramme régulier est fini par définition (§ 4.2.3). Il existe donc deux entiers distincts $1 \leq m < n$ tels que $y_m(m) = y_n(n)$. Il s'ensuit que la séquence de places $y_m(1, m)y_n(n+1, 2n)$ est un parcours de D , de marque $0^m 1^n$. Or ceci est impossible puisque la séquence $0^m 1^n$ n'appartient pas au langage A .

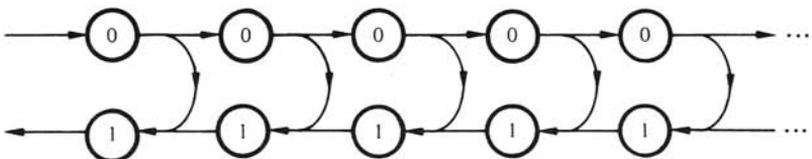


Fig. 4.9

On remarque que l'impossibilité de représenter ce langage par un diagramme régulier tient essentiellement au caractère *fini* de ces diagrammes. Si l'on admet la possibilité d'un nombre infini de places, un "diagramme" tel que celui de la figure 4.9 représente de façon évidente le langage en question.

4.3. AUTOMATES FINIS

4.3.1 Définitions

Soient A un graphe (§ 1.5.1) et X un alphabet. On dira que A est un graphe *sur* l'alphabet X si l'étiquette de toute arête de A est un élément de X . On désignera par $\Sigma(A)$ l'ensemble des sommets d'un graphe A quelconque.

On appelle *automate fini* (ou simplement *automate*) sur un alphabet X , un triplet (A, I, T) composé d'un graphe A non vide sur X , et de deux sous-ensembles $I, T \subset \Sigma(A)$ appelés respectivement *ensemble des sommets initiaux* et *ensemble des sommets terminaux* de l'automate.

4.3.2 Exemple

La figure 4.10 représente un automate (A, I, T) sur l'alphabet $X = \{0, 1, 2\}$. Le graphe A se compose des arêtes $(\alpha, 0, \alpha)$, $(\alpha, 0, \beta)$, $(\beta, 2, \alpha)$, $(\gamma, 1, \delta)$, $(\delta, 2, \epsilon)$. L'ensemble des sommets de A est $\Sigma(A) = \{\alpha, \beta, \gamma, \delta, \epsilon\}$. L'ensemble des sommets initiaux est $I = \{\alpha, \gamma\}$ et l'ensemble des sommets terminaux $T = \{\alpha, \epsilon\}$. Les sommets initiaux sont marqués d'une flèche entrante $\rightarrow s$, les sommets terminaux d'une flèche sortante $s \rightarrow$, et les sommets à la fois initiaux et terminaux sont marqués d'une seule flèche entrante et sortante $\leftrightarrow s$.

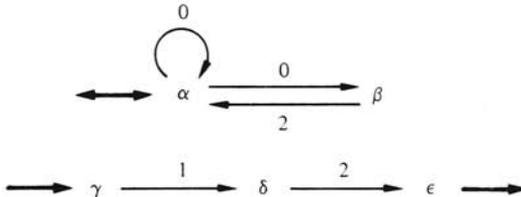


Fig. 4.10

4.3.3 Définitions

Tout automate (A, I, T) sur un alphabet X représente un certain langage sur cet alphabet. Pour définir ce langage, appelons *parcours* de l'automate tout chemin du graphe A (§ 1.5.4) ayant pour origine un sommet initial et pour extrémité un sommet terminal. Le langage représenté par l'automate est l'ensemble des étiquettes de ses parcours.

Par exemple, la séquence d'arêtes $(\alpha, 0, \alpha)$ $(\alpha, 0, \beta)$ $(\beta, 2, \alpha)$ est un parcours de l'automate de la figure 4.10. Son étiquette 002 appartient donc au langage représenté, qu'on reconnaît aisément comme le langage régulier $(0 \cup 02)^+ \cup 12$.

4.3.4 Diagramme régulier associé à un automate

A tout automate (A, I, T) on peut associer un diagramme régulier $D = (P, \rightarrow P, \leftarrow P, \varphi, \mu)$ qui de façon évidente représente le même langage. Pour fixer les idées, nous considérons

l'automate de la figure 4.10. Le *diagramme régulier associé*, dont nous allons énoncer les règles de construction, est représenté par la figure 4.11 ou le tableau 4.12.

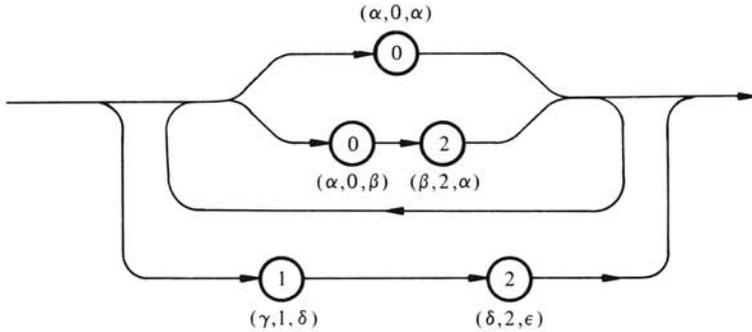


Fig. 4.11

p	$p \cdot$	$\mu(p)$
$\leftrightarrow (\alpha, 0, \alpha)$	$\{(\alpha, 0, \alpha), (\alpha, 0, \beta)\}$	0
$\rightarrow (\alpha, 0, \beta)$	$\{(\beta, 2, \alpha)\}$	0
$\leftarrow (\beta, 2, \alpha)$	$\{(\alpha, 0, \alpha), (\alpha, 0, \beta)\}$	2
$\rightarrow (\gamma, 1, \delta)$	$\{(\delta, 2, \epsilon)\}$	1
$\leftarrow (\delta, 2, \epsilon)$	\emptyset	2

Tableau 4.12

L'ensemble des places du diagramme est l'ensemble des arêtes du graphe A. On peut donc écrire $P = A$. Les places initiales du diagramme sont les arêtes (r, x, s) telles que $r \in I$. Les places terminales du diagramme sont les arêtes (r, x, s) telles que $s \in T$. Pour chaque place $p = (r, x, s)$, l'ensemble $p \cdot$ des places suivantes est l'ensemble des arêtes d'origine s . Enfin la marque $\mu(p)$ d'une place $p = (r, x, s)$ est son étiquette x .

Il est évident qu'un parcours de l'automate (au sens du paragraphe 4.3.3) est un parcours du diagramme associé (au sens du paragraphe 4.2.3) et réciproquement, et que l'étiquette de l'un est la marque de l'autre. Par suite l'automate et son diagramme associé représentent le même langage. Il en découle la proposition suivante, d'après le paragraphe 4.2.18.

4.3.5 Proposition

Tout automate fini (A, I, T) sur l'alphabet X représente un langage régulier sur X .

4.3.6 Exercice

Le langage représenté par l'automate de la figure 4.13 sur l'alphabet $X = \{0, 1\}$ est l'ensemble des séquences possédant un nombre pair de 1. Construire le diagramme régulier associé.

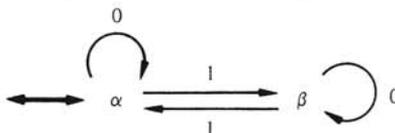


Fig. 4.13

4.3.7 Remarque

Etant donné un diagramme régulier D , il n'existe pas nécessairement un automate (A, I, T) dont le diagramme associé soit semblable à D . Considérons par exemple le diagramme de la figure 4.14, représentant le langage 0^+1 . S'il existait un automate (A, I, T) auquel ce diagramme soit associé selon les règles du paragraphe 4.3.4, le graphe A posséderait deux arêtes a, b . L'origine de a serait égale à l'extrémité de a et à l'origine de b . Ces arêtes seraient donc de la forme $a = (\alpha, 0, \alpha)$, $b = (\alpha, 1, \beta)$. L'origine de a devrait être un sommet initial de l'automate, mais non l'origine de b , ce qui est contradictoire. Ainsi le diagramme considéré ne peut être le diagramme associé à un automate au sens du paragraphe 4.3.4.

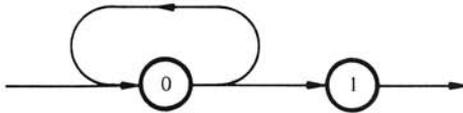


Fig. 4.14

Cela ne signifie pas que le langage 0^+1 n'est pas représentable par un automate. On peut le représenter par exemple par l'automate de la figure 4.15. Le diagramme régulier associé à ce dernier est donné par la figure 4.16.

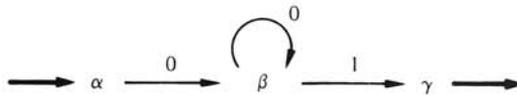


Fig. 4.15

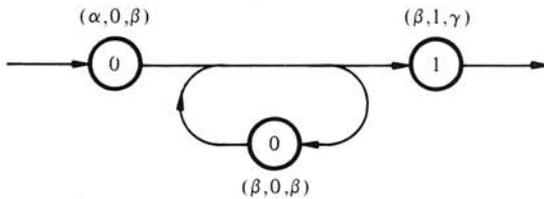


Fig. 4.16

4.3.8 Automate associé à un diagramme régulier

Nous allons voir qu'à tout diagramme régulier $D = (P, \rightarrow P, \leftarrow P, \varphi, \mu)$ sur un alphabet X on peut associer un automate (A, I, T) sur X , qui représente le même langage. Pour fixer les idées, nous considérons le diagramme de la figure 4.17 (tab. 4.18). L'automate associé, dont nous allons énoncer les règles de construction, est représenté par la figure 4.19. Le langage représenté est $0^+1 \cup 21 \cup 2$.

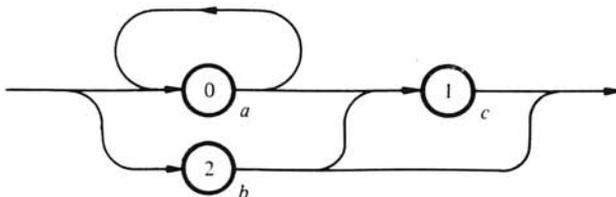


Fig. 4.17

p	$p \cdot$	$\mu(p)$
$\rightarrow a$	$\{a, c\}$	0
$\leftarrow b$	$\{c\}$	2
$\leftarrow c$	\emptyset	1

Tableau 4.18

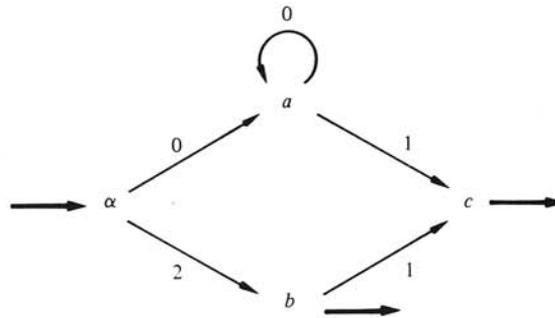


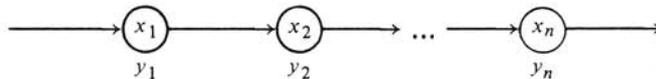
Fig. 4.19

L'ensemble des sommets du graphe A est l'ensemble des places du diagramme, augmenté d'un élément α distinct de toutes les places. L'ensemble des arêtes du graphe est construit selon les règles suivantes :

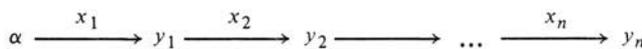
- pour toute place initiale p du diagramme, le triplet $(\alpha, \mu(p), p)$ est une arête du graphe (exemple $(\alpha, 0, a)$);
- pour toute place p du diagramme et pour toute place $q \in p \cdot$, le triplet $(p, \mu(q), q)$ est une arête du graphe (exemple $(a, 0, a)$, $(a, 1, c)$);
- toute arête du graphe est obtenue par l'une des règles précédentes.

Enfin α est l'unique sommet initial de l'automate, et les sommets terminaux sont les places terminales du diagramme.

Il est clair que l'automate ainsi construit représente le même langage que le diagramme régulier donné, car si



est un parcours du diagramme, alors



est un parcours de l'automate et réciproquement.

4.3.9 Proposition

Tout langage régulier sur un alphabet X peut être représenté par un automate (A, I, T) sur X , dans lequel l'ensemble initial I ne comporte qu'un seul sommet α et le graphe A n'a aucune arête d'extrémité α .

La proposition découle du paragraphe 4.2.11 et de la construction ci-dessus (§ 4.3.8).

4.3.10 Exercice

Construire l'automate associé au diagramme régulier de la figure 4.14 selon les règles du paragraphe 4.3.8. Effectuer la même construction pour le diagramme de la figure 4.4.

4.3.11 Exercice

Les règles du paragraphe 4.3.8 associent à un diagramme $D = (P, \rightarrow P, \leftarrow P, \varphi, \mu)$ un automate dont les arêtes sont de la forme $(p, \mu(q), q)$ où q est une place du diagramme. Montrer qu'on peut associer au même diagramme un autre automate, dont les arêtes sont de la forme $(p, \mu(p), q)$ où p est une place du diagramme. Cet automate aura pour sommets l'ensemble des places du diagramme augmenté d'un unique sommet **terminal** ω . Préciser cette construction, et l'appliquer aux diagrammes des figures 4.17, 4.14, 4.4.

4.3.12 Définition

Considérons un graphe A sur un alphabet X , par exemple le graphe de la figure 4.20 sur l'alphabet $X = \{0,1\}$, et soit Y l'ensemble des sommets de A . Etant donné un sommet p et un élément x de l'alphabet X , associons au couple (p, x) l'ensemble des sommets q tels que (p, x, q) soit une arête de A , et notons cet ensemble $(p \cdot x)_A$, ou simplement $p \cdot x$. Par exemple (fig. 4.20) au couple $(a, 0)$ est associé l'ensemble $a \cdot 0 = \{b, c\}$, et au couple $(a, 1)$ l'ensemble $a \cdot 1 = \emptyset$.

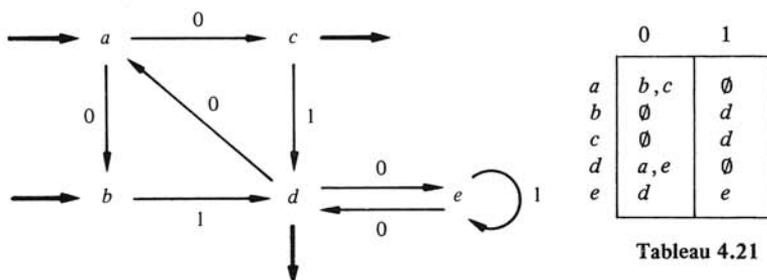


Fig. 4.20

Tableau 4.21

Ce faisant nous avons défini une correspondance $f: Y \times X \rightarrow Y$, représentée par le tableau 4.21 pour l'exemple considéré. Il y a la même relation entre le tableau 4.21 et le graphe A (fig. 4.20) qu'entre la table de transition et le graphe de transition d'une machine séquentielle (§ 2.4.5), à savoir

$$q \in (p \cdot x)_A \iff (p, x, q) \in A. \tag{4.80}$$

Le tableau 4.21 pourrait être considéré comme la table de transition d'une machine séquentielle s'il n'avait pas de case vide. Malgré cette circonstance, on peut définir le transformé $(P \cdot x)_A$ d'un **sous ensemble** quelconque $P \subset Y$ par une **séquence** quelconque x sur X , de la même façon qu'aux paragraphes 2.4.14 et suivants. Il suffit en fait, pour définir cette opération, de poser les règles suivantes, où P et Q sont des sous-ensembles quelconques de Y , et x, x' des séquences quelconques sur X :

$$(P \cup Q) \cdot x = (P \cdot x) \cup (Q \cdot x) \tag{4.81}$$

$$\emptyset \cdot x = \emptyset \quad (4.82)$$

$$P \cdot (xx') = (P \cdot x) \cdot x' \quad (4.83)$$

Comme exemple d'application de ces règles, calculons le transformé de l'ensemble $P = \{b, c, d\}$ (tab. 4.21) par la séquence $x = 10$. On a $P \cdot 10 = (P \cdot 1) \cdot 0$ par (4.83). Or $P \cdot 1 = b \cdot 1 \cup c \cdot 1 \cup d \cdot 1 = d$ par (4.81), et $d \cdot 0 = \{a, e\}$. Donc $P \cdot 10 = \{a, e\}$.

On peut définir le transformé $(P \cdot x)_A$ d'une autre manière en posant ceci : $(P \cdot x)_A$ est l'ensemble des extrémités des chemins de A dont l'origine appartient à P et dont l'étiquette est x . Le lecteur peut vérifier que cette définition est compatible avec la première. Par exemple (fig. 4.20) pour $P = \{b, c, d\}$ et $x = 10$, l'ensemble des chemins avec origine dans P et étiquette x est représenté par la figure 4.22. Les extrémités de ces chemins constituent l'ensemble $P \cdot x = \{a, e\}$.

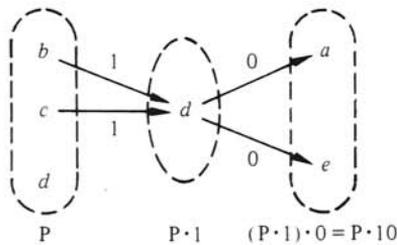


Fig. 4.22

4.3.13 Proposition

Soit (A, I, T) un automate sur l'alphabet X . Pour qu'une séquence x sur X appartienne au langage représenté par cet automate, il faut et il suffit que

$$(I \cdot x)_A \cap T \neq \emptyset. \quad (4.84)$$

En effet d'après la définition du transformé $I \cdot x$, la relation (4.84) signifie : il existe un chemin de A dont l'origine appartient à I , dont l'étiquette est x , et dont l'extrémité appartient à T .

4.3.14 Définition

Un graphe A est *déterministe* s'il ne possède pas deux arêtes (p, x, q) , (p, x, q') ayant même origine, même étiquette, et des extrémités différentes ($q \neq q'$). Si A est un graphe sur l'alphabet X , cette définition peut s'exprimer sous la forme suivante : pour tout sommet p de A et pour tout $x \in X$, l'ensemble $(p \cdot x)_A$ possède au plus un élément. Un automate (A, I, T) est dit *déterministe* si son graphe A est déterministe.

Par exemple, l'automate de la figure 4.20 n'est pas déterministe, puisque son graphe possède les deux arêtes $(a, 0, b)$, $(a, 0, c)$. Par contre l'automate de la figure 4.19 est déterministe.

4.3.15 Construction

A tout automate (A, I, T) sur un alphabet X , on peut associer par une certaine construction un automate déterministe $D(A, I, T)$ sur l'alphabet X , qui représente le

même langage. Cette construction, appelée *construction des sous-ensembles*, est exposée ici en prenant comme exemple l'automate non déterministe (A, I, T) de la figure 4.20.

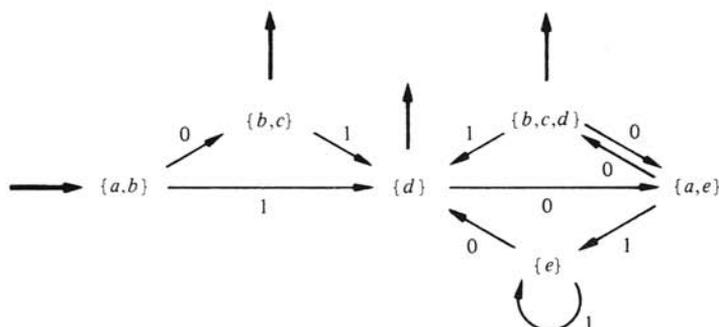


Fig. 4.23

	0	1
$\{a,b\}$	$\{b,c\}$	$\{d\}$
$\{b,c\}$	\emptyset	$\{d\}$
$\{d\}$	$\{a,e\}$	\emptyset
$\{a,e\}$	$\{b,c,d\}$	$\{e\}$
$\{b,c,d\}$	$\{a,e\}$	$\{d\}$
$\{e\}$	$\{d\}$	$\{e\}$

Tableau 4.24

Le résultat de la construction dans cet exemple est l'automate de la figure 4.23. On remarque d'abord que les sommets de cet automate sont tous des sous-ensembles (non vides) de l'ensemble des sommets de A. On remarque ensuite que si (P, x, Q) est une arête de l'automate $D(A, I, T)$, alors Q est le transformé $(P \cdot x)_A$. Par exemple on a dans la figure 4.23 l'arête $(\{a, b\}, 0, \{b, c\})$, et l'on observe que l'ensemble $\{b, c\}$ est le transformé de $\{a, b\}$ par 0 selon A.

La construction commence par le choix du sous-ensemble $I = \{a, b\}$ comme premier sommet de $D(A, I, T)$. On calcule ensuite les transformés $(I \cdot 0)_A$ et $(I \cdot 1)_A$ au moyen du tableau 4.21. Ces transformés sont les ensembles non vides $\{b, c\}$ et $\{d\}$ (tableau 4.24, 1ère ligne), et ceci nous donne les deux arêtes d'origine $\{a, b\}$ dans la figure 4.23. Nous répétons cette opération en partant des nouveaux sommets obtenus $\{b, c\}$ et $\{d\}$. Leurs transformés respectifs par 0 et 1 selon A sont calculés au moyen du tableau 4.21, et notés dans le tableau 4.24 (2e et 3e lignes). On néglige les transformés vides, et l'on obtient les nouvelles arêtes $(\{b, c\}, 1, \{d\})$ et $(\{d\}, 0, \{a, e\})$ dans la figure 4.23. Un nouveau sommet est ainsi apparu, le sommet $\{a, e\}$, et l'on répète l'opération à partir de celui-ci, et ainsi de suite jusqu'à ce qu'elle ne fasse plus apparaître de nouveau sommet. En fait on construit en entier le tableau 4.24, en partant de $\{a, b\}$, et en calculant au moyen du tableau 4.21 les transformés de $\{a, b\}$, puis successivement les transformés de tous les ensembles non vides qui apparaissent. Le graphe de la figure 4.23 est tiré du tableau 4.24 en négligeant l'ensemble vide.

L'automate $D(A, I, T)$ possède un seul sommet initial par définition à savoir le sommet $I = \{a, b\}$. Les sommets terminaux de $D(A, I, T)$ sont tous les sommets Q qui contiennent au moins un élément de l'ensemble terminal $T = \{c, d\}$ du premier automate, à savoir les sommets $\{b, c\}$, $\{d\}$, $\{b, c, d\}$.

L'automate $D(A, I, T)$ est déterministe par construction. Il ne peut pas avoir deux arêtes distinctes (P, x, Q) et (P, x, Q') puisque l'extrémité d'une arête quelconque d'origine P et d'étiquette x doit être le transformé $(P \cdot x)_A$.

Les automates (A, I, T) et $D(A, I, T)$ représentent le même langage. En effet supposons que

$$I \xrightarrow{x(1)} Q(1) \xrightarrow{x(2)} \dots \xrightarrow{x(n)} Q(n) \quad (4.85)$$

soit un parcours de l'automate $D(A, I, T)$. Par construction de cet automate, et par (4.83), l'ensemble $Q(n)$ est le transformé $(I \cdot x)_A$ de I par la séquence $x = x(1, n)$. D'autre part $Q(n) \cap T \neq \emptyset$ puisque $Q(n)$ est un sommet terminal de $D(A, I, T)$. On a donc $(I \cdot x)_A \cap T \neq \emptyset$, et en vertu du paragraphe 4.3.13, la séquence x appartient au langage représenté par (A, I, T) . Réciproquement, pour toute séquence $x(1, n)$ du langage représenté par (A, I, T) on a $(I \cdot x)_A \cap T \neq \emptyset$. En posant $Q(1) = (I \cdot x(1))_A$, $Q(2) = (Q(1) \cdot x(2))_A$, etc., on obtient un parcours (4.85) de $D(A, I, T)$ d'étiquette x .

4.3.16 Proposition

Tout langage régulier peut être représenté par un automate déterministe $(A, \{\alpha\}, T)$ possédant un sommet initial α et un seul.

En effet on peut toujours représenter un langage régulier par un automate (A', I', T') et appliquer à celui-ci la construction des sous-ensembles (§ 4.3.15), c'est-à-dire prendre pour automate déterministe $(A, \{\alpha\}, T)$ l'automate $D(A', I', T')$.

4.3.17 Exercice

Appliquer la construction des sous-ensembles (§ 4.3.15) à l'automate de la figure 4.10.

4.3.18 Automates complets

Un graphe A sur l'alphabet X est dit *complet sur X* , si pour chaque sommet p de A et chaque élément x de X , le graphe possède une arête et une seule d'origine p et d'étiquette x . Un automate (A, I, T) sur l'alphabet X est *complet sur X* si son graphe A est complet sur X . Il est clair qu'un graphe complet sur X est déterministe (§ 4.3.14).

La figure 4.25 est un exemple d'automate déterministe sur l'alphabet $X = \{0, 1, 2\}$ qui n'est pas complet sur X . Il est facile de rendre un tel automate complet sans modifier le langage représenté. Il suffit de lui ajouter un sommet β (fig. 4.26) et d'introduire de nouvelles arêtes, toutes d'extrémité β . L'ensemble des parcours de l'automate, donc le langage représenté, n'est pas modifié par ces adjonctions.

La généralisation de cet exemple, jointe à la proposition précédente (§ 4.3.16), permet d'affirmer que tout langage régulier sur un alphabet X peut être représenté par un automate $(A, \{\alpha\}, T)$, complet sur X et possédant un seul sommet initial α .

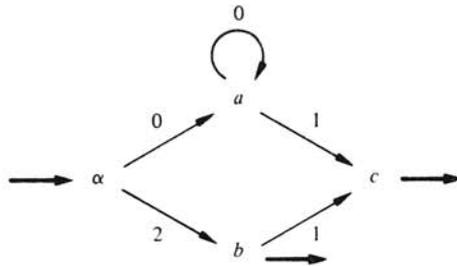


Fig. 4.25

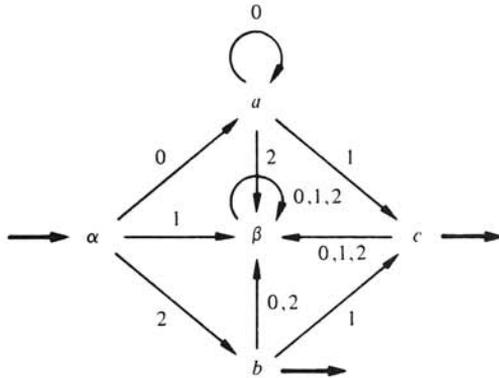


Fig. 4.26

4.3.19 Proposition

Soient $(A, \{\alpha\}, T)$ un automate complet sur l'alphabet X , et \bar{T} l'ensemble des sommets q de A tels que $q \notin T$. Si L est le langage sur X représenté par $(A, \{\alpha\}, T)$, alors le langage représenté par l'automate $(A, \{\alpha\}, \bar{T})$ est le langage \bar{L} , à savoir l'ensemble des séquences x sur X telles que $x \notin L$.

Il suffit en effet de remarquer que dans un automate de la forme considérée, pour toute séquence x sur X il existe un chemin et un seul d'étiquette x et d'origine α , et que x appartient à L ou à \bar{L} selon que l'extrémité de ce chemin appartient à T ou à \bar{T} .

4.3.20 Corollaire

Si R est un langage régulier sur l'alphabet X , alors le langage complémentaire \bar{R} est régulier.

En effet R peut être représenté par un automate $(A, \{\alpha\}, T)$ complet sur X . En vertu de la proposition précédente (§ 4.3.19) et du paragraphe 4.3.5, le langage \bar{R} est régulier.

4.3.21 Familles d'automates

Soient X et Z deux alphabets. Supposons qu'à chaque élément z de l'alphabet Z on ait associé un automate (A_z, I_z, T_z) sur l'alphabet X . Nous dirons qu'on a défini une *famille d'automates* d'indices $z \in Z$ sur l'alphabet X . Nous nous intéresserons particulièrement au cas où tous les automates de la famille ont le même graphe A et le même ensemble de sommets initiaux I , c'est-à-dire où l'on a $A_z = A$ et $I_z = I$ pour tout $z \in Z$.

Une telle famille d'automates sera notée $(A, I, T_z)_{z \in Z}$, et appelée une *famille d'automates à graphe et ensemble initial commun*.

Considérons par exemple le graphe A de la figure 4.27 sur l'alphabet $X = \{0, 1, 2\}$. Soit Z l'alphabet $\{a, b\}$, et associons à a l'ensemble de sommets $T_a = \{\gamma, \delta\}$, et à b l'ensemble de sommets $T_b = \{\delta, \epsilon\}$. Posons encore $I = \{\alpha, \beta\}$. Ainsi se trouve définie une famille d'automates (A, I, T_z) d'indices $z \in \{a, b\}$ sur l'alphabet X , à graphe et ensemble initial commun. Cette famille est représentée dans la figure 4.28. Les flèches marquant les sommets terminaux portent les lettres a ou/et b afin qu'on distingue les ensembles T_a et T_b .

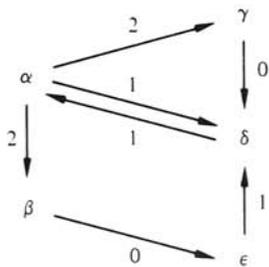


Fig. 4.27

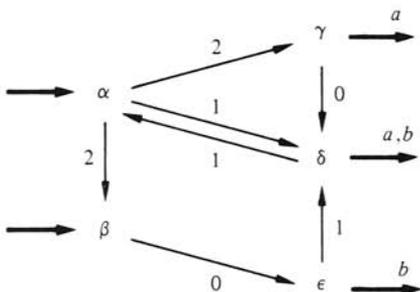


Fig. 4.28

Il est clair qu'une famille d'automates (A, I, T_z) d'indices $z \in Z$ sur l'alphabet X représente une famille de langages réguliers L_z sur l'alphabet X .

Réciproquement supposons donnée une famille de langages réguliers L_z sur un alphabet X , les indices z étant les éléments d'un alphabet Z . Il existe une famille d'automates (A, I, T_z) d'indices $z \in Z$ sur l'alphabet X , à graphe et ensemble initial commun, qui représente la famille des langages L_z . En effet, on peut toujours construire une famille d'automates (A_z, I_z, T_z) , symbolisés dans la figure 4.29 (où l'on a pris pour Z l'alphabet $1, \dots, n$), qui représentent les langages L_z , et qui n'ont deux-à-deux aucun sommet commun. Il suffit alors de prendre pour graphe A la réunion des graphes A_z , et pour ensemble I la réunion des ensembles I_z . Pour un $z \in Z$ quelconque, le langage représenté par l'automate (A, I, T_z) est égal au langage représenté par (A_z, I_z, T_z) , car en vertu du fait que les graphes A_z n'ont pas de sommet commun, tout parcours de (A, I, T_z) est un parcours de (A_z, I_z, T_z) et réciproquement.

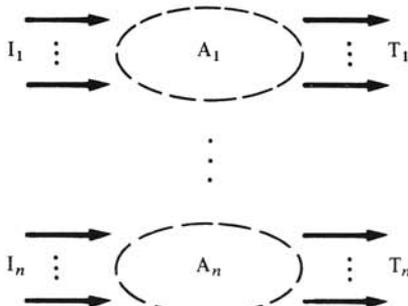


Fig. 4.29

4.3.22 Construction

La construction des sous-ensembles (§ 4.3.15) peut s'appliquer à une famille d'automates à graphe et ensemble initial commun. Illustrons-le par l'exemple de la famille (A, I, T_z) représentée par la figure 4.28, sur l'alphabet $X = \{0, 1, 2\}$, et où l'ensemble des indices z est $Z = \{a, b\}$. On construit premièrement le tableau des transformés $(p \cdot x)_A$ pour chaque sommet p du graphe et chaque $x \in X$ (tableau 4.30). Puis le tableau 4.31 est construit selon la méthode précédemment expliquée (§ 4.3.15), et que nous répétons brièvement. La première ligne du tableau contient les transformés de l'ensemble $I = \{\alpha, \beta\}$, noté $\alpha\beta$ pour la commodité d'écriture. Les transformés de $\alpha\beta$ sont $\epsilon, \delta, \beta\gamma$ (mis pour $\{\epsilon\}, \{\delta\}, \{\beta, \gamma\}$). On note ces ensembles dans la première colonne et l'on calcule leurs transformés, et ainsi de suite pour chaque nouvel ensemble obtenu. Le graphe déterministe correspondant au tableau 4.31 est représenté dans la figure 4.32. On prend le sommet $I = \alpha\beta$ comme unique sommet initial. Enfin on associe aux indices a et b des ensembles de sommets terminaux définis à partir des ensembles $T_a = \{\gamma, \delta\}$, $T_b = \{\delta, \epsilon\}$ (fig. 4.28) comme au paragraphe 4.3.15. Un sommet K (fig. 4.32) est terminal avec indice z (a ou b) si et seulement si $K \cap T_z \neq \emptyset$. Ainsi le sommet $\beta\gamma$ est terminal d'indice a puisque $\gamma \in T_a$; le sommet $\epsilon\delta$ est terminal d'indice a et d'indice b , puisque δ appartient à T_a et à T_b .

	0	1	2
α	\emptyset	δ	β, γ
β	ϵ	\emptyset	\emptyset
γ	δ	\emptyset	\emptyset
δ	\emptyset	α	\emptyset
ϵ	\emptyset	δ	\emptyset

Tableau 4.30

	0	1	2
$\alpha\beta$	ϵ	δ	$\beta\gamma$
ϵ	\emptyset	δ	\emptyset
δ	\emptyset	α	\emptyset
$\beta\gamma$	$\epsilon\delta$	\emptyset	\emptyset
α	\emptyset	δ	$\beta\gamma$
$\epsilon\delta$	\emptyset	$\alpha\delta$	\emptyset
$\alpha\delta$	\emptyset	$\alpha\delta$	$\beta\gamma$

Tableau 4.31

La famille d'automates ainsi construite (fig. 4.32) n'est autre que la famille des automates déterministes $D(A, I, T_z)$ associés aux automates (A, I, T_z) de la figure 4.28, au sens du paragraphe 4.3.15. Ces deux familles d'automates représentent donc la même famille de langages $L_z (L_a, L_b)$ sur l'alphabet X .

Considérant que toute famille de langages réguliers L_z sur un alphabet X peut être représentée par une famille d'automates (A, I, T_z) , donc aussi par la famille d'automates déterministes $D(A, I, T_z)$, nous pouvons énoncer la proposition qui suit.

4.3.23 Proposition

Toute famille de langages réguliers L_z sur un alphabet X , d'indices $z \in Z$, peut être représentée par une famille d'automates déterministes $(A, \{p\}, T_z)_{z \in Z}$, ayant graphe commun et ensemble initial commun à un seul sommet.

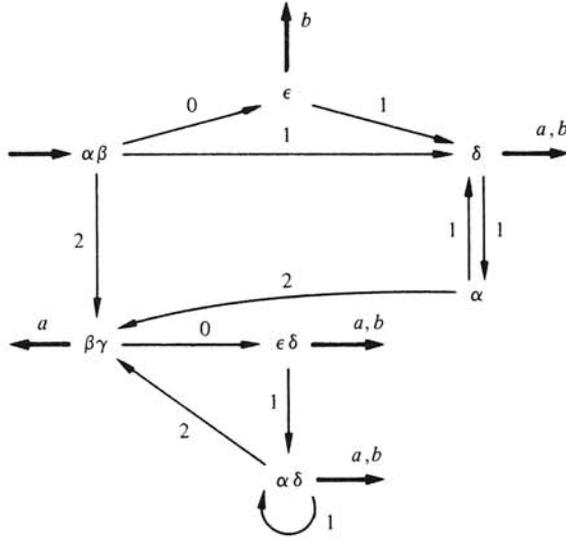


Fig. 4.32

4.4. APPLICATION À LA SYNTHÈSE DES MACHINES

4.4.1 Introduction

Nous allons mettre en relation la théorie des automates et langages réguliers avec le problème posé dans l'introduction de ce chapitre (§ 4.4.1), à savoir la description d'une machine $R_q : X^+ \rightarrow Z^+$, où $R[X, Y, Z]$ est une machine de Mealy, et $q \in Y$ un état secondaire initial (§ 2.6.15). Les automates que nous allons considérer sont les automates $(A, \{q\}, \{z\})$ où A est le graphe de transition de R (§ 2.6.12), q est l'unique sommet initial, et $z \in Z$ un unique sommet terminal.

4.4.2 Exemple

Reprenons l'exemple de la figure 4.1. Il convient de rappeler que le graphe de transition de cette machine est noté de façon abrégée dans la figure 4.1, selon les conventions des paragraphes 2.4.34 et 2.6.12. La table de transition de cette machine est reproduite ci-dessous (tableau 4.33). Sachant que le tiret représente soit l'ensemble $Y = \{p, q, r\}$ soit l'ensemble $Z = \{0, 1\}$, on peut donner une représentation complète du graphe de transition A de cette machine (fig. 4.34), conforme à la définition du paragraphe 2.6.12.

	a	b	c	d
p	$q; -$	$-; 1$	$p; 0$	$-; 0$
q	$q; 0$	$r; -$	$-; -$	$p; 1$
r	$-; -$	$r; 1$	$q; 1$	$-; 0$

Tableau 4.33

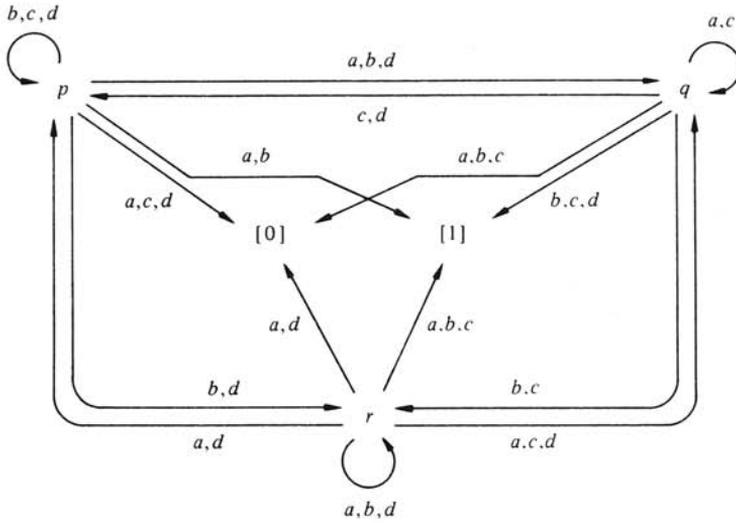


Fig. 4.34

Nous verrons que les langages réguliers représentés par les automates $(A, \{q\}, \{0\})$ et $(A, \{q\}, \{1\})$ peuvent être utilisés comme un mode de représentation de la machine R_q , et que la machine R peut être construite de façon systématique à partir de ces langages.

4.4.3 Propriétés du graphe de transition d'une machine de Mealy

Soit A le graphe de transition d'une machine de Mealy $R[X, Y, Z]$. On suppose que $Y \cap Z = \emptyset$. Nous voulons mentionner quelques propriétés élémentaires de ce graphe, qui peuvent être illustrées par l'exemple de la figure 4.34.

Premièrement, l'ensemble $\Sigma(A)$ des sommets de ce graphe est l'ensemble $Y \cup Z$. Cependant il n'y a pas d'arête dont l'origine soit un sommet $z \in Z$. Ainsi tout chemin de A a pour origine un élément de Y . Par conséquent, si I et T sont deux sous-ensembles quelconques de $\Sigma(A)$, le langage représenté par l'automate (A, I, T) est égal au langage représenté par l'automate $(A, I \cap Y, T)$.

Soient P un sous-ensemble de Y , et x une séquence sur X . Il ne faut pas confondre l'ensemble $(P \cdot x)_A$ tel qu'il est défini au paragraphe 4.3.12, avec les ensembles $(P \cdot x)_R$ et $(P|x)_R$ définis au paragraphe 2.6.10. On a précisément

$$(P \cdot x)_R = (P \cdot x)_A \cap Y \tag{4.86}$$

$$(P|x)_R = (P \cdot x)_A \cap Z. \tag{4.87}$$

Par exemple pour la machine R de la figure 4.34 (tab. 4.33), on a pour $P = \{p, q\}$ et $x = ab$:

$$(P \cdot x)_A = \{r, 0, 1\}$$

$$(P \cdot x)_R = \{r\}$$

$$(P|x)_R = \{0, 1\}.$$

4.4.4 Proposition

Soient (A, I, T) un automate sur un alphabet X , et x, x' deux séquences sur X . En désignant par $\text{Lang}(A, I, T)$ le langage représenté, on a

$$xx' \in \text{Lang}(A, I, T) \iff x' \in \text{Lang}(A, (I \cdot x)_A, T). \quad (4.88)$$

4.4.5 Démonstration

On a

$$xx' \in \text{Lang}(A, I, T) \iff (I \cdot xx')_A \cap T \neq \emptyset \quad \text{par (4.84)}$$

$$\iff ((I \cdot x)_A \cdot x')_A \cap T \neq \emptyset \quad \text{par (4.83)}$$

$$\iff x' \in \text{Lang}(A, (I \cdot x)_A, T). \quad \text{par (4.84)}$$

La relation (4.88) est d'ailleurs assez évidente lorsqu'on la formule comme suit : pour qu'il existe dans A un chemin d'étiquette xx' avec origine dans I et extrémité dans T , il faut et il suffit qu'il existe un chemin d'étiquette x' avec origine dans $(I \cdot x)_A$ et extrémité dans T .

4.4.6 Corollaire

Soient $R[X, Y, Z]$ une machine de Mealy, A son graphe de transition, $P \subset Y$, $z \in Z$, et x, x' deux séquences sur X . On a

$$xx' \in \text{Lang}(A, P, \{z\}) \iff x' \in \text{Lang}(A, (P \cdot x)_R, \{z\}). \quad (4.89)$$

En effet, il suffit d'appliquer la formule (4.88), de noter que le langage représenté par $(A, (P \cdot x)_A, \{z\})$ est égal au langage représenté par $(A, (P \cdot x)_A \cap Y, \{z\})$, et d'appliquer (4.86).

4.4.7 Définitions

Pour tout alphabet X on définit l'opérateur fin de séquence sur X , noté ω_X , qui associe à toute séquence $x(1, n)$ sur X le dernier élément $\omega(x) = x(n)$ de cette séquence. En d'autres termes ω_X est une application de X^+ dans X . Elle est complètement caractérisée par les propriétés suivantes :

$$x \in X \implies \omega_X(x) = x \quad (4.90)$$

$$\omega_X(xx') = \omega_X(x') \quad (4.91)$$

la relation (4.91) étant vraie pour deux séquences x, x' quelconques sur X .

Si L est un langage sur l'alphabet X , la notation $\omega_X(L)$ désigne, conformément au paragraphe 1.3.3, l'ensemble des fins de séquence $\omega_X(x)$ des séquences $x \in L$. Les formules suivantes, où L et L' sont deux langages sur X sont immédiates :

$$\omega_X(L) \subset X \quad (4.92)$$

$$\omega_X(L \cup L') = \omega_X(L) \cup \omega_X(L') \quad (4.93)$$

$$\omega_X(LL') = \omega_X(L') \quad (\text{pour } L \neq \emptyset) \quad (4.94)$$

$$\omega_X(L^+) = \omega_X(L). \quad (4.95)$$

On écrira souvent $\omega_X L$ pour $\omega_X(L)$.

4.4.8 Définitions

Considérons une machine $M : X^+ \rightarrow Z^+$ de type quelconque (§ 2.1.2), et soient x une séquence d'entrée ($x \in X^+$), et z un élément de l'alphabet Z . Supposons que x et z vérifient la relation

$$z \in \omega_z M(x). \quad (4.96)$$

L'ensemble $M(x)$ est l'ensemble des séquences de sortie $y \in Z^+$ qui correspondent à la séquence d'entrée x . Nous pouvons dire que le langage $M(x) \subset Z^+$ est l'ensemble des réponses possibles de la machine M pour la séquence d'entrée x . L'ensemble $\omega_z M(x)$ est l'ensemble des fins de séquence de ces réponses, ou *ensemble des fins de réponse* de la machine pour cette séquence x . La relation (4.96) signifie simplement que z est une fin de réponse de M pour x , autrement dit qu'il existe une séquence $y(1, n) \in M(x)$ telle que $z = y(n)$.

Considérons maintenant pour un élément fixé z de Z , l'ensemble des séquences d'entrée x qui vérifient la relation (4.96). Nous dirons que ce sont les séquences d'entrée qui admettent la fin de réponse z . Leur ensemble sera noté $L_z(M)$. On a donc par définition

$$x \in L_z(M) \iff z \in \omega_z M(x). \quad (4.97)$$

4.4.9 Exemple

Considérons le cahier des charges du détecteur de séquences *erp* (§ 3.2.9). Soient X l'alphabet $\{e, r, l, p\}$ et Z l'alphabet $\{0, 1\}$. Le cahier des charges définit une machine $M : X^+ \rightarrow Z^+$. Il peut s'exprimer sous la forme suivante :

$$\left. \begin{aligned} L_1(M) &= (e \cup r \cup l \cup p)^* erp \\ L_0(M) &= \overline{L_1(M)}. \end{aligned} \right\} \quad (4.98)$$

En effet une séquence d'entrée x admet la fin de réponse $z = 1$ si et seulement si elle se termine par *erp*. L'ensemble des séquences qui se terminent par *erp* est représenté par l'expression régulière donnée dans (4.98). Une séquence d'entrée x admet la fin de réponse 0 si et seulement si elle ne se termine pas par *erp*. L'ensemble $L_0(M)$ est donc le complément de l'ensemble $L_1(M)$ par rapport à X^+ .

Nous sommes en présence d'un cas particulier sur deux points. Premièrement les langages $L_z(M)$ sont réguliers (en vertu du paragraphe 4.3.20 pour $z = 0$). Secondement, toute séquence d'entrée x admet une fin de réponse z et une seule, et appartient ainsi à l'un et seulement l'un des langages $L_z(M)$. La famille des langages $L_z(M)$ constitue dans cet exemple une partition de l'ensemble X^+ .

4.4.10 Proposition

Soit $R[X, Y, Z]$ une machine de Mealy. On a défini au paragraphe 2.6.15 la machine $R_P : X^+ \rightarrow Z^+$ pour un sous-ensemble $P \subset Y$ quelconque (non vide). La révision de cette notion est conseillée au lecteur, ainsi que celle des formules (2.82), (2.83).

Si P et P' sont deux sous-ensembles (non vides) de Y , on a pour toute séquence $x \in X^+$:

$$R_{P \cup P'}(x) = R_P(x) \cup R_{P'}(x). \quad (4.99)$$

Il en découle aussitôt

$$R_P(x) = \bigcup_{p \in P} R_p(x). \quad (4.100)$$

Enfin si $x \in X$ (séquence de longueur 1), et $x' \in X^+$, on a

$$\left. \begin{aligned} \omega_Z R_P(xx') &= \omega_Z R_Q(x') \\ \text{avec } Q &= (P \cdot x)_R. \end{aligned} \right\} \quad (4.101)$$

4.4.11 Démonstration

La formule (4.99) découle immédiatement de (2.81). Pour démontrer (4.101), on peut écrire (en omettant l'indice Z de ω_Z) :

$$\begin{aligned} \omega R_P(xx') &= \omega \bigcup_{p \in P} R_p(xx') && \text{par (4.100)} \\ &= \bigcup_{p \in P} \omega (R_p(x) R_{Q(p)}(x')) && \\ &\quad \text{avec } Q(p) = (p \cdot x)_R && \text{par (4.93) et (2.83)} \\ &= \bigcup_{p \in P} \omega R_{Q(p)}(x') && \text{par (4.94)} \\ &= \omega \bigcup_{p \in P} R_{Q(p)}(x') && \text{par (4.93)} \\ &= \omega R_Q(x') && \\ &\quad \text{avec } Q = \bigcup_{p \in P} Q(p) && \text{par (4.99)}. \end{aligned}$$

Il suffit de noter finalement que

$$Q = \bigcup_{p \in P} (p \cdot x)_R = (P \cdot x)_R \quad \text{par (2.31).}$$

4.4.12 Proposition

Soient $R[X, Y, Z]$ une machine de Mealy (avec $Y \cap Z = \emptyset$), A son graphe de transition, P un sous-ensemble non vide de Y , et $z \in Z$. Pour toute séquence x sur X , on a

$$z \in \omega_Z R_P(x) \iff x \in \text{Lang}(A, P, \{z\}). \quad (4.102)$$

Autrement dit z est une fin de réponse de la machine R_P pour la séquence d'entrée x si et seulement s'il existe dans le graphe de transition de R un chemin d'étiquette x , d'extrémité z , avec origine dans P .

4.4.13 Démonstration

On raisonne par récurrence sur la longueur de x . Supposons que $lg(x) = 1$. Alors $R_P(x)$ est un ensemble de séquences de longueur 1, donc

$$\omega_Z R_P(x) = R_P(x) = \bigcup_{p \in P} R_p(x) = \bigcup_{p \in P} (p|x)_R$$

par (4.90), (4.100), et (2.82). On a donc $z \in \omega_Z R_P(x)$ si et seulement si $z \in (p|x)_R$ pour un $p \in P$. Or $z \in (p|x)_R$ équivaut à $(p, x, z) \in A$ par définition du graphe A . La

relation $z \in \omega_Z R_P(x)$ est donc vraie si et seulement s'il existe un chemin (p, x, z) de A avec $p \in P$, ce qui prouve (4.102) pour $lg(x) = 1$.

Supposons que (4.102) soit vraie pour toute séquence x de longueur n , et quel que soit $P \subset Y$. Considérons une séquence xx' de longueur $n+1$ sur X , avec $lg(x) = 1$ et $lg(x') = n$. En vertu de (4.101), on peut écrire

$$\begin{aligned} z \in \omega_Z R_P(xx') &\iff z \in \omega_Z R_Q(x') \\ &\text{avec } Q = (P \cdot x)_R \\ &\iff x' \in \mathbf{Lang}(A, (P \cdot x)_R, \{z\}) \quad (\text{car } lg(x') = n) \\ &\iff xx' \in \mathbf{Lang}(A, P, \{z\}). \quad \text{par (4.89)} \end{aligned}$$

La relation (4.102) est donc vraie pour toute séquence de longueur $n+1$, ce qui achève la démonstration.

4.4.14 Commentaire

Compte tenu de la définition (4.97), la relation (4.102) peut s'écrire encore

$$x \in \mathbf{L}_Z(R_P) \iff x \in \mathbf{Lang}(A, P, \{z\}) \quad (4.103)$$

et comme cette relation est vraie quel que soit x , on a

$$\mathbf{L}_Z(R_P) = \mathbf{Lang}(A, P, \{z\}). \quad (4.104)$$

On appliquera surtout cette relation au cas où P comporte un seul élément $p \in Y$, et où elle s'écrit

$$\mathbf{L}_Z(R_p) = \mathbf{Lang}(A, \{p\}, \{z\}). \quad (4.105)$$

La relation (4.105) montre que pour une machine de Mealy $R[X, Y, Z]$ quelconque et un état initial $p \in Y$ quelconque, la famille des langages $\mathbf{L}_Z(R_p)$ ($z \in Z$) est une famille de langages réguliers, puisqu'elle est représentée par la famille d'automates $(A, \{p\}, \{z\})_{z \in Z}$.

On peut ajouter que la famille des langages $\mathbf{L}_Z(R_p)$ constitue un recouvrement de l'ensemble X^+ , en ce sens que toute séquence $x \in X^+$ appartient à l'un au moins des langages $\mathbf{L}_Z(R_p)$. En effet l'ensemble $R_p(x)$ n'étant jamais vide, toute séquence x admet au moins une fin de réponse $z \in Z$ pour la machine R_p .

En général les langages $\mathbf{L}_Z(R_p)$ ne sont pas disjoints deux-à-deux. On le voit par exemple dans la figure 4.34, où l'on a

$$a \in \mathbf{L}_0(R_p) \text{ et } a \in \mathbf{L}_1(R_p).$$

La séquence a , pour cette machine R_p , admet la fin de réponse 0 *et* la fin de réponse 1.

4.4.15 Problème de synthèse

Nous avons vu au paragraphe 4.4.9 que le cahier des charges d'une machine $M: X^+ \rightarrow Z^+$ peut être formulé parfois de façon claire et succincte par la donnée de la famille de langages $\mathbf{L}_Z(M)$ ($z \in Z$). On se pose alors le problème de synthèse suivant.

Soient X et Z deux alphabets, $(R_z)_{z \in Z}$ une famille de langages sur l'alphabet X . Trouver une machine de Mealy $R[X, Y, Z]$ et un état initial $p \in Y$ tel que $R_z = \mathbf{L}_Z(R_p)$ pour tout $z \in Z$.

Pour que le problème posé ait une solution, on sait (§ 4.4.14) que la famille des langages R_z doit vérifier les conditions suivantes :

$$R_z \text{ est régulier quel que soit } z \in Z \quad (4.106)$$

$$\bigcup_{z \in Z} R_z = X^*. \quad (4.107)$$

La seconde condition est souvent peu commode à satisfaire lors de la formalisation d'un cahier des charges. Nous allons donc modifier le problème posé, et lui donner une forme plus générale. Supposons donnée une famille $(R_z)_{z \in Z}$ de langages sur X , satisfaisant (4.106), et posons

$$R = \bigcup_{z \in Z} R_z.$$

Soit \bar{R} l'ensemble complémentaire de R par rapport à X^+ (\bar{R} est vide si la condition (4.107) est satisfaite). On cherche une machine de Mealy $R[X, Y, Z]$ avec un état initial $p \in Y$ telle que

$$R_z \subset L_z(R_p) \subset R_z \cup \bar{R} \quad (4.108)$$

pour tout $z \in Z$.

La condition (4.108) peut s'énoncer comme suit. Si une séquence d'entrée x appartient à l'un des langages R_z donnés dans le cahier des charges, alors dans la machine R_p cherchée, cette séquence x doit admettre la fin de réponse z (condition $R_z \subset L_z(R_p)$). Si une séquence x admet dans la machine R_p la fin de réponse z , alors ou bien x appartient au langage R_z donné dans le cahier des charges, ou bien x n'appartient à aucun des langages du cahier des charges (celui-ci n'impose aucune fin de réponse pour cette séquence x).

Nous allons montrer que le problème ainsi posé admet toujours une solution, et donner une méthode pour le résoudre. Nous commençons par un exemple.

4.4.16 Exemple

Soient $X = \{0,1\}$, $Z = \{u, v\}$, et

$$\left. \begin{aligned} R_u &= 0^*(00 \cup 01) \\ R_v &= 0^*(01 \cup 10). \end{aligned} \right\} \quad (4.109)$$

On s'est donné ainsi une famille de langages réguliers $(R_z)_{z \in Z}$ sur l'alphabet X . Notre premier objectif sera de construire une famille d'automates déterministes $(A, \{p\}, T_z)$ représentant la famille de langages R_z . On sait (§ 4.3.23) que c'est toujours possible. Pour obtenir une telle famille d'automates, on peut commencer par une famille d'automates non déterministes telle que celle de la figure 4.35, qui représente de façon évidente les langages R_z . On applique à cette famille d'automates la construction des sous-ensembles (§ 4.3.22), qui est donnée ici par les tableaux 4.36, 4.37, et la figure 4.38. Pour alléger l'écriture, on désigne par les nombres 1 à 6 les sommets de la figure 4.38, qu'on reproduit avec cette notation dans la figure 4.39. Cette dernière figure sera la famille d'automates déterministes $(A, \{p\}, T_z)$ représentant les langages R_z , avec $\Sigma(A) = \{1, \dots, 6\}$, $p = 1$, $T_u = \{4, 5\}$, et $T_v = \{5, 6\}$.

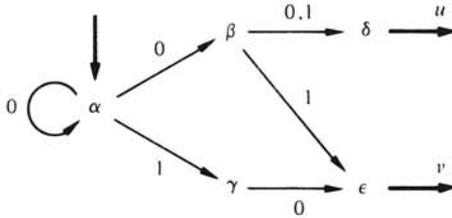


Fig. 4.35

	0	1
α	$\alpha\beta$	γ
β	δ	$\delta\epsilon$
γ	ϵ	\emptyset
δ	\emptyset	\emptyset
ϵ	\emptyset	\emptyset

Tableau 4.36

	0	1
α	$\alpha\beta$	γ
$\alpha\beta$	$\alpha\beta\delta$	$\gamma\delta\epsilon$
γ	ϵ	\emptyset
$\alpha\beta\delta$	$\alpha\beta\delta$	$\gamma\delta\epsilon$
$\gamma\delta\epsilon$	ϵ	\emptyset
ϵ	\emptyset	\emptyset

Tableau 4.37

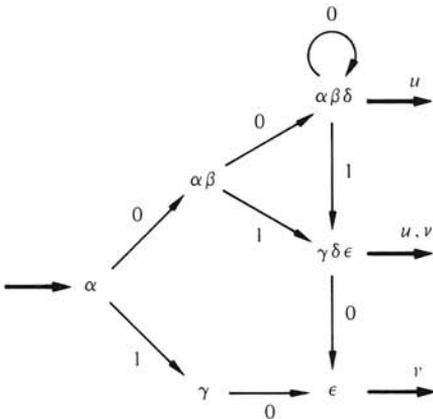


Fig. 4.38

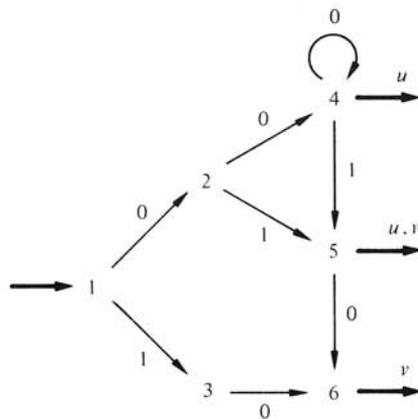


Fig. 4.39

A partir de la figure 4.39, on construit la table de transition d'une machine de Mealy $R[X, Y, Z]$, avec $Y = \Sigma(A) = \{1, \dots, 6\}$. Cette machine R (tab. 4.40) est construite en appliquant les règles suivantes, où T désigne la réunion des ensembles T_z , et x, y, z sont trois éléments quelconques de X, Y, Z respectivement.

$$(y \cdot x)_R = \begin{cases} (y \cdot x)_A & \text{si } (y \cdot x)_A \neq \emptyset \\ Y & \text{si } (y \cdot x)_A = \emptyset \end{cases} \quad (4.110)$$

$$z \in (y \cdot x)_R \iff (y \cdot x)_A \cap T_z \neq \emptyset \quad \text{ou} \quad (y \cdot x)_A \cap T = \emptyset. \quad (4.111)$$

Considérons quelques cas pour illustrer ces règles.

- Soient $y=1, x=0$.

On a $(y \cdot x)_A = 2$ (fig. 4.39). Comme $(y \cdot x)_A \neq \emptyset$, il vient par (4.110) $(y \cdot x)_R = (y \cdot x)_A = 2$ (tab. 4.40, 1ère ligne, 1ère colonne). D'autre part $(y \cdot x)_A \cap T = \emptyset$ puisque $T = \{4, 5, 6\}$. Selon la règle (4.111) on a donc

$z \in (y|x)_R$ quel que soit $z \in Z$, autrement dit $(y|x)_R = Z = \{u, v\}$. C'est ce qui est noté par le tiret dans le tableau 4.40 (1ère ligne, 1ère colonne).

- Soient $y = 2, x = 0$.

On a $(y \cdot x)_A = 4$ (fig. 4.39), d'où $(y \cdot x)_R = 4$ par (4.110). D'autre part $(y \cdot x)_A \cap T \neq \emptyset$, et le seul élément $z \in Z$ pour lequel on a $(y \cdot x)_A \cap T_z \neq \emptyset$ est $z = u$. Selon la règle (4.111), on pose $(y|x)_R = u$.

- Soient $y = 2, x = 1$.

On a $(y \cdot x)_A = 5$ (fig. 4.39), d'où $(y \cdot x)_R = 5$ par (4.110). D'autre part on a $(y \cdot x)_A \cap T_z \neq \emptyset$ pour $z = u$ et $z = v$. On pose donc $(y|x)_R = Z$ (tiret).

- Soient $y = 3, x = 1$.

On a $(y \cdot x)_A = \emptyset$ (fig. 4.39), d'où $(y \cdot x)_R = Y$ par (4.110). Ceci est représenté par le premier tiret dans la case correspondante du tableau 4.40. D'autre part $(y \cdot x)_A = \emptyset$ implique évidemment $(y \cdot x)_A \cap T = \emptyset$. Selon (4.111), on a $z \in (y|x)_R$ quel que soit $z \in Z$, d'où $(y|x)_R = Z$. Ceci est représenté par le deuxième tiret dans la case correspondante du tableau 4.40.

	0	1
1	2; -	3; -
2	4; u	5; -
3	6; v	-; -
4	4; u	5; -
5	6; v	-; -
6	-; -	-; -

Tableau 4.40

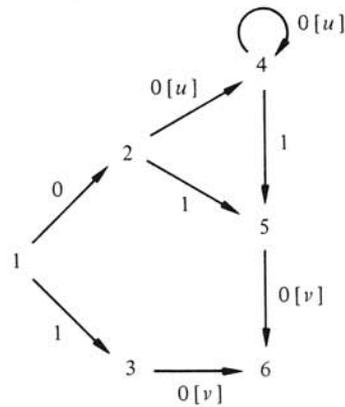


Fig. 4.41

La figure 4.41 représente le graphe de transition de la machine de Mealy R , avec les notations abrégées (§ 2.4.34, 2.6.12). Il ne faut pas confondre ce graphe de transition G avec le graphe A de la figure 4.39. Vu le nombre de tirets dans la table de transition de R , la représentation complète (non abrégée) de G serait pratiquement illisible. Par exemple, pour le sommet 3 et l'étiquette $x = 1$, il y a dans G une arête $(3, x, q)$ pour chacun des sommet $q = 1, \dots, 6$.

Nous affirmons que la machine R , munie de l'état secondaire initial $p = 1$, vérifie (4.108), c'est-à-dire que

$$R_u \subset L_u(R_1) \subset R_u \cup \bar{R}$$

$$R_v \subset L_v(R_1) \subset R_v \cup \bar{R}$$

\bar{R} étant le complément du langage $R = R_u \cup R_v$ par rapport à X^+ . Cette assertion est généralisée et démontrée dans les deux paragraphes qui suivent.

4.4.17 Proposition

Soient X, Z deux alphabets, et $(A, \{p\}, T_z)_{z \in Z}$ une famille d'automates *déterministes* sur X , à graphe commun A et sommet initial unique p commun. On désigne par Y l'ensemble des sommets de A , et l'on suppose que $Y \cap Z = \emptyset$. Soit encore $(R_z)_{z \in Z}$ la famille des langages sur X représentés par les automates $(A, \{p\}, T_z)$. Alors la machine de Mealy $R[X, Y, Z]$ définie par (4.110) et (4.111) vérifie (4.108) pour chaque élément z de Z , R étant la réunion des langages R_z et \bar{R} le complément de R par rapport à X^+ .

4.4.18 Démonstration

Commençons par remarquer que si G est le graphe de transition de R , alors $A \subset G$. En effet si (y, x, y') est une arête de A , alors $y' \in (y \cdot x)_R$ par (4.110), donc (y, x, y') est une arête de G . Par suite tout chemin de A est un chemin de G .

Montrons que $R_z \subset L_z(R_p)$. Soit $x \in R_z$. Il existe un chemin $c : p \rightarrow q$ dans A tel que $q \in T_z$ et d'étiquette $|c| = x$. Le chemin c est aussi un chemin de G . Soit (p', x', q) sa dernière arête. En vertu de (4.111), on a $z \in (p' | x')_R$ puisque $(p' \cdot x')_A = \{q\} \subset T_z$, donc $(p', x', z) \in G$. En remplaçant la dernière arête de c par (p', x', z) , on obtient un chemin $c' : p \rightarrow z$ de G , d'étiquette $|c| = x$. Ceci prouve que $x \in \text{Lang}(G, \{p\}, \{z\})$, et par (4.105) que $x \in L_z(R_p)$.

Montrons enfin que $L_z(R_p) \subset R_z \cup \bar{R}$. On peut énoncer cette relation sous la forme suivante : si $x \in L_z(R_p)$ et si $x \notin R_z$, alors $x \in \bar{R}$. Nous allons le prouver en démontrant la relation plus générale, pour un élément $q \in Y$ quelconque :

$$x \in L_z(R_q) \text{ et } x \notin \text{Lang}(A, \{q\}, T_z) \implies x \notin \text{Lang}(A, \{q\}, T) \quad (4.112)$$

Dans cette relation, T désigne la réunion des ensembles T_z . Avant de la démontrer, notons qu'elle donne bien la relation précédente pour $q = p$, car $\text{Lang}(A, \{p\}, T_z) = R_z$ et $\text{Lang}(A, \{p\}, T) = R$. Nous démontrerons (4.112) par récurrence sur la longueur de x .

Supposons que $lg(x) = 1$, et qu'on ait $x \in L_z(R_q)$. Il vient $z \in \omega_z R_q(x)$ par (4.97). Comme toute séquence de $R_q(x)$ est de longueur 1, $\omega_z R_q(x) = R_q(x) = (q | x)_R$ par (2.82). On a donc $z \in (q | x)_R$, et par (4.111) l'une des relations $(q \cdot x)_A \cap T_z \neq \emptyset$, $(q \cdot x)_A \cap T = \emptyset$ doit être vraie. Or la première est fautive si l'on suppose que $x \notin \text{Lang}(A, \{q\}, T_z)$, en vertu du paragraphe 4.3.13. En faisant cette hypothèse on a donc $(q \cdot x)_A \cap T = \emptyset$, d'où $x \notin \text{Lang}(A, \{q\}, T)$ (§ 4.3.13). La relation (4.112) est ainsi démontrée pour $lg(x) = 1$.

Supposons que (4.112) soit vraie pour toute séquence x de longueur n et quel que soit $q \in Y$. Soit xx' une séquence sur X de longueur $n + 1$, avec $lg(x) = 1$ et $lg(x') = n$. Supposons que

$$xx' \in L_z(R_q) \tag{4.113}$$

$$xx' \notin \text{Lang}(A, \{q\}, T_z). \tag{4.114}$$

Il vient

$$z \in \omega_z R_q(xx') \tag{par (4.97)}$$

$$z \in \omega_z R_Q(x') \text{ avec } Q = (q \cdot x)_R \tag{par (4.101).}$$

Il y a lieu de considérer ici deux cas.

- $(q \cdot x)_A = \emptyset$.

Dans ce cas $(q \cdot xx')_A = \emptyset$, d'où $(q \cdot xx')_A \cap T = \emptyset$, et $xx' \notin \text{Lang}(A, \{q\}, T)$ (§ 4.3.13).

- $(q \cdot x)_A \neq \emptyset$.

Dans ce cas $(q \cdot x)_R = (q \cdot x)_A$ par (4.110), et comme A est un graphe déterministe, on a $(q \cdot x)_R = \{q'\}$ avec $q' \in Y$. Il vient alors $z \in \omega_Z R_{q'}(x')$, soit

$$x' \in L_z(R_{q'}). \quad (4.115)$$

Or l'hypothèse (4.114) peut s'écrire selon (4.89)

$$x' \notin \text{Lang}(A, \{q'\}, T_z). \quad (4.116)$$

Comme $lg(x') = n$, les relations (4.115) et (4.116) impliquent $x' \notin \text{Lang}(A, \{q'\}, T)$, et il vient (§ 4.3.13)

$$xx' \notin \text{Lang}(A, \{q\}, T).$$

4.4.19 Commentaire

Le terme "déterministe" employé dans l'énoncé du paragraphe 4.4.17 est essentiel. La proposition serait fautive si l'on ne faisait pas cette hypothèse. On peut le voir en reprenant l'exemple du paragraphe 4.4.16. Si l'on construit une machine de Mealy $R[X, Y, Z]$ selon les règles (4.110) et (4.111), en prenant pour automates $(A, \{p\}, T_z)$ non pas les automates déterministes de la figure 4.39, mais les automates non déterministes de la figure 4.35, en posant $Y = \{\alpha, \beta, \gamma, \delta, \epsilon\}$, $p = \alpha$, $T_u = \{\delta\}$, $T_v = \{\epsilon\}$, on aura

$$(\alpha \cdot 0)_R = \{\alpha, \beta\} \quad \text{par (4.110)}$$

$$(\alpha \mid 0)_R = \{u, v\} \quad \text{par (4.111)}.$$

Le graphe de transition de R comprend donc les arêtes

$$0 \begin{array}{c} \circlearrowleft \\ \alpha \end{array} \xrightarrow{0} [v]$$

et l'on voit que $00 \in L_v(R_\alpha)$. Or $00 \notin R_v$ et $00 \in R$ (4.109). La relation

$$L_v(R_\alpha) \subset R_v \cup \bar{R}$$

n'est donc pas vérifiée.

4.4.20 Exemple

Considérons le cahier des charges suivant. Le système séquentiel S de la figure 4.42 doit éliminer l'effet des rebondissements du commutateur. On doit avoir $z = 0$ lorsque $x_1 = 1$ (commutateur à gauche), $z = 1$ lorsque $x_2 = 1$ (commutateur à droite), et z conserve sa valeur antérieure lorsque le commutateur est en position intermédiaire ($x_1 = x_2 = 0$).

Ce cahier des charges est illustré par le chronogramme de la figure 4.43. On remarque que x_1 et x_2 n'ont jamais simultanément la valeur 1. Désignons par a, b, c les trois valeurs possibles de $x_1 \times x_2$ à savoir $a = 1 \times 0$, $b = 0 \times 1$, $c = 0 \times 0$.

Si nous échantillons le chronogramme de façon arbitraire, mais assez fine

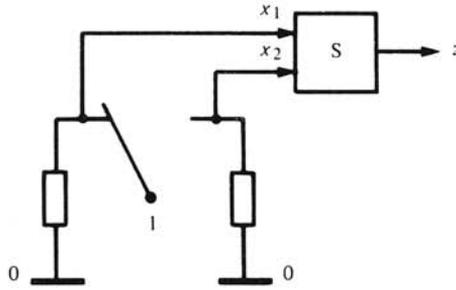


Fig. 4.42

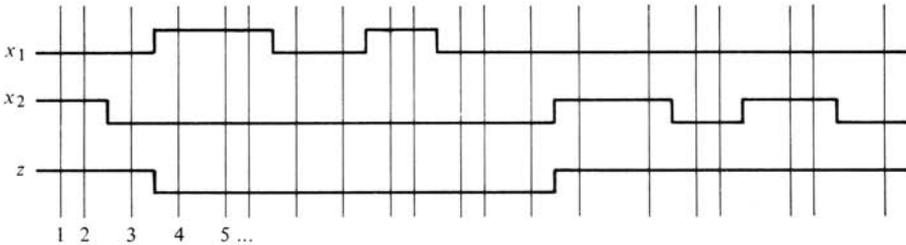


Fig. 4.43

pour que toute variation des signaux soit prise en compte, nous observons ceci :

- $z(t) = 0$ si et seulement si la séquence $x(1) \dots x(t)$ se termine par a ou par ac^n (n quelconque);
- $z(t) = 1$ si et seulement si la séquence $x(1) \dots x(t)$ se termine par b ou par bc^n (n quelconque).

Posons

$$\left. \begin{aligned} R_0 &= (a \cup b \cup c)^* a c^* \\ R_1 &= (a \cup b \cup c)^* b c^* \end{aligned} \right\} \quad (4.117)$$

Le langage R_0 (resp. R_1) est l'ensemble des séquences sur l'alphabet $\{a, b, c\}$ qui se terminent par a ou ac^n (resp. par b ou bc^n), n étant quelconque.

Posons encore $d = 1 \times 1$, et soient X l'alphabet $\{a, b, c, d\}$ et Z l'alphabet $\{0,1\}$. On cherche une machine de Mealy $R[X, Y, Z]$ avec un état secondaire initial $p \in Y$ telle que

$$\left. \begin{aligned} R_0 &\subset L_0(R_p) \subset R_0 \cup \bar{R} \\ R_1 &\subset L_1(R_p) \subset R_1 \cup \bar{R} \end{aligned} \right\} \quad (4.118)$$

Les langages R_z ($z = 0, 1$) sont représentés par les automates non déterministes de la figure 4.44. La construction des sous-ensembles, dont le détail est laissé au lecteur, fournit les automates déterministes de la figure 4.45. Pour la commodité d'écriture, on change en p, q, r les noms des sommets de cette figure, qu'on reproduit dans la figure 4.46. Cette dernière figure est une famille d'automates déterministes $(A, \{p\}, T_z)$, qui représente les langages R_z .

La machine de Mealy $R[X, Y, Z]$ dont la table de transition est le tableau 4.47, est construite selon (4.110) et (4.111). Elle est solution du problème.

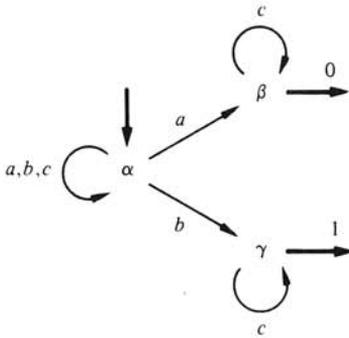


Fig. 4.44

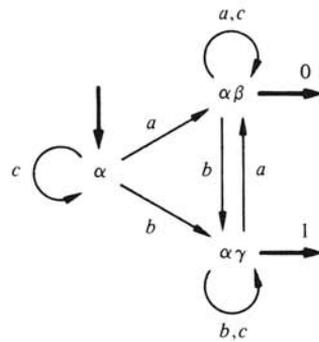


Fig. 4.45

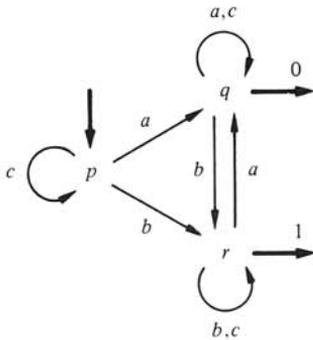


Fig. 4.46

	<i>c</i>	<i>b</i>	<i>d</i>	<i>a</i>
<i>p</i>	0×0	0×1	1×1	1×0
<i>q</i>	$p ; -$	$r ; 1$	$- ; -$	$q ; 0$
<i>r</i>	$q ; 0$	$r ; 1$	$- ; -$	$q ; 0$
	$r ; 1$	$r ; 1$	$- ; -$	$q ; 0$

Tableau 4.47

On verra au chapitre suivant que la machine *R* peut être réduite. Dans le cas présent, la réduction revient simplement à supprimer la première ligne de la table de transition, et à prendre comme état secondaire initial l'un quelconque des états *q*, *r*. En codant *r* par 0, et *q* par 1, on obtient une machine dont la composante séquentielle est un élément de mémoire $\bar{s}r$ (fig. 2.14). L'emploi d'un filp-flop de type $\bar{s}r$ pour supprimer l'effet des rebondissements d'un commutateur est bien connu dans la pratique. L'exemple a été choisi seulement pour sa valeur illustrative et sa simplicité.

4.4.21 Exemple

Considérons le cahier des charges suivant, décrivant une machine binaire $M(x_1, x_2, x_3)(z_1, z_2)$. On suppose qu'à tout instant *t* une seule au plus des variables

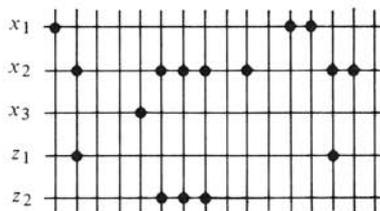


Fig. 4.48

x_1, x_2, x_3 vaut 1. Dans ces conditions, les variables z_1, z_2 obéissent aux règles suivantes :

- $z_1(t) = 1$ si et seulement si $t > 1$ et $x_1(t-1) = x_2(t) = 1$;
- $z_2(t) = 1$ si et seulement s'il existe un instant $t' < t$ tel que $x_3(t') = 1$ et $x_2(t'') = 1$ pour tout instant $t'' (t' < t'' \leq t)$.

Ces règles sont illustrées par la figure 4.48.

Désignons par X et Z les ensembles respectifs B_3 et B_2 (§ 3.1.7). On a $M: X^+ \rightarrow Z^+$. Soit E le sous-ensemble de X formé des vecteurs $0 \times 0 \times 0, 1 \times 0 \times 0, 0 \times 1 \times 0, 0 \times 0 \times 1$, que nous coderons par les nombres 0, 1, 2, 3 de la façon suivante :

$$0 \times 0 \times 0 \rightarrow 0$$

$$1 \times 0 \times 0 \rightarrow 1$$

$$0 \times 1 \times 0 \rightarrow 2$$

$$0 \times 0 \times 1 \rightarrow 3.$$

Nous codons de même les éléments de Z par les nombres 0, 1, 2, 3 :

$$0 \times 0 \rightarrow 0; \quad 1 \times 0 \rightarrow 1; \quad 0 \times 1 \rightarrow 2; \quad 1 \times 1 \rightarrow 3.$$

Le cahier des charges ne définit les réponses de la machine à une séquence d'entrée x que pour $x \in E^+$. Dans ce cas, si $y \in M(x)$, on a

$$\omega_Z(y) = 1 \iff x \in R_1 = (0 \cup 1 \cup 2 \cup 3)^* 12. \quad (4.119)$$

Le langage R_1 est l'ensemble des séquences x sur E qui se terminent par 12. Une séquence $y \in M(x)$ se termine par 1 si et seulement si x se termine par 12. De même y se termine par 2 si et seulement si x se termine par 32^n (n entier quelconque ≥ 1), ce qui s'écrit

$$\omega_Z(y) = 2 \iff x \in R_2 = (0 \cup 1 \cup 2 \cup 3)^* 32^+. \quad (4.120)$$

La séquence y se termine par 0 si et seulement si x n'appartient ni à R_1 ni à R_2 , c'est-à-dire x appartient au complément du langage $R_1 \cup R_2$ par rapport à E^+ , ce qu'on note

$$\omega_Z(y) = 0 \iff x \in R_0 = E^+ - (R_1 \cup R_2). \quad (4.121)$$

Enfin y ne se termine jamais par 3, ce qu'on peut noter

$$\omega_Z(y) = 3 \iff x \in R_3 = \emptyset. \quad (4.122)$$

Rappelons que les relations (4.119) à (4.122) font l'hypothèse $x \in E^+$ et $y \in M(x)$. Le cahier des charges ne dit rien des réponses de la machine M pour une séquence d'entrée x qui n'appartient pas à E^+ . En remarquant que le langage $R = R_0 \cup R_1 \cup R_2 \cup R_3$ est égal à E^+ , et en désignant par \bar{R} le complément de R par rapport à X^+ , on a donc

$$R_z \subset L_z(M) \subset R_z \cup \bar{R} \quad \text{pour } z = 0, 1, 2, 3. \quad (4.123)$$

On cherche pour M une machine R_p où $R[X, Y, Z]$ est une machine de Mealy et $p \in Y$ un état secondaire initial.

La figure 4.49 (non-déterministe) représente les langages R_1 (ensemble terminal $\{\delta\}$), R_2 (ensemble terminal $\{\epsilon\}$), et R_3 (ensemble terminal vide). Nous verrons qu'il n'est pas nécessaire de représenter à ce stade le langage

$$R_0 = E^+ - (R_1 \cup R_2 \cup R_3). \quad (4.124)$$

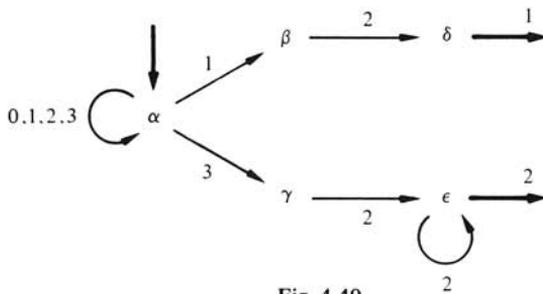


Fig. 4.49

La construction des sous-ensembles appliquée aux automates de la figure 4.49, considérés comme automates sur l'alphabet X , est donnée par le tableau 4.50, dans lequel x représente n'importe quel élément de X différent de $0, 1, 2, 3$. On pourra coder les éléments de $X - E$ par les nombres $4, 5, 6, 7$ de façon quelconque.

	0	1	2	3	x
$0 \leftarrow \alpha$	α	$\alpha\beta$	α	$\alpha\gamma$	\emptyset
$0 \leftarrow \alpha\beta$	α	$\alpha\beta$	$\alpha\delta$	$\alpha\gamma$	\emptyset
$0 \leftarrow \alpha\gamma$	α	$\alpha\beta$	$\alpha\epsilon$	$\alpha\gamma$	\emptyset
$1 \leftarrow \alpha\delta$	α	$\alpha\beta$	α	$\alpha\gamma$	\emptyset
$2 \leftarrow \alpha\epsilon$	α	$\alpha\beta$	$\alpha\epsilon$	$\alpha\gamma$	\emptyset

Tableau 4.50

Le tableau définit une famille d'automates déterministes $(A, \{p\}, T_z)$ ($z = 0, 1, 2, 3$), avec $p = \alpha$, $T_1 = \{\alpha\delta\}$, $T_2 = \{\alpha\epsilon\}$, $T_3 = \emptyset$ et

$$T_0 = \Sigma(A) - (T_1 \cup T_2 \cup T_3). \quad (4.125)$$

Les ensembles T_1, T_2, T_3 sont définis par la construction des sous-ensembles. L'ensemble T_0 est défini (choisi) par (4.125). Ce choix de T_0 implique que le langage représenté par $(A, \{p\}, T_0)$ est le langage R_0 (4.124), en vertu du fait que le graphe A , considéré comme graphe sur l'alphabet E , est complet (§ 4.3.18, 4.3.19).

En désignant respectivement par a, b, c, d, e les sommets $\alpha, \alpha\beta, \alpha\gamma, \alpha\delta, \alpha\epsilon$ du graphe A , la machine de Mealy $R[X, Y, Z]$ définie par (4.110) et (4.111) a pour table de transition le tableau 4.51. Avec l'état secondaire initial a , elle est solution du problème.

	0	1	2	3	x
a	$a; 0$	$b; 0$	$a; 0$	$c; 0$	$—; —$
b	$a; 0$	$b; 0$	$d; 1$	$c; 0$	$—; —$
c	$a; 0$	$b; 0$	$e; 2$	$c; 0$	$—; —$
d	$a; 0$	$b; 0$	$a; 0$	$c; 0$	$—; —$
e	$a; 0$	$b; 0$	$e; 2$	$c; 0$	$—; —$

Tableau 4.51

Cette machine peut être réduite comme on le verra au chapitre suivant. Il nous suffit de constater ici que les lignes a et d du tableau 4.51 sont identiques, ainsi que

les lignes c et e . Dans un certain sens, qui sera précisé ultérieurement, les états secondaires a et d (resp. c et e) peuvent être identifiés.

4.4.22 Exercice

Une machine binaire $M(x_1, x_2)(z)$ est décrite par le cahier des charges suivant. La variable z vaut toujours 0 lorsque x_1 vaut 0. Le premier changement de x_2 qui se produit pendant que $x_1 = 1$ fait passer z à 1; alors z reste à 1 jusqu'à ce que x_1 retourne à 0. Ce cahier des charges est illustré par la figure 4.52.

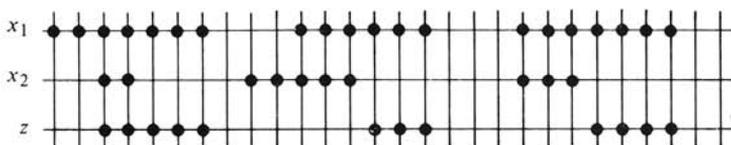


Fig. 4.52

On code les valeurs $0 \times 0, 0 \times 1, 1 \times 0, 1 \times 1$ de $x_1 \times x_2$ respectivement par 0, 1, 2, 3. On demande de représenter les langages $R_z = \mathbf{L}_z(M)$ ($z = 0, 1$) par des expressions régulières sur l'alphabet $X = \{0, 1, 2, 3\}$. Remarquer que $R_1 = \bar{R}_0$ (cf. § 4.4.9). Construire un automate déterministe complet $(A, \{p\}, T_0)$ sur X représentant le langage R_0 , en passant par un automate non déterministe et par la construction des sous-ensembles. Poser $T_1 = \Sigma(A) - T_0$. L'automate $(A, \{p\}, T_1)$ représente alors R_1 . Construire la table de transition d'une machine de Mealy $R[X, Y, Z]$ à partir de $(A, \{p\}, T_z)$, avec $Y = \Sigma(A), Z = \{0, 1\}$, selon (4.110) et (4.111).

4.4.23 Exercice

Même exercice que le précédent, sur le cahier des charges du paragraphe 4.4.9.

4.4.24 Exercice

La figure 4.53 représente une voie de circulation à sens unique, parcourue par deux types de véhicules : des véhicules longs (de longueur L) et des véhicules courts (de longueur $l < L$). Une machine binaire $M(a, b, c)(z)$ dont les variables d'entrée proviennent de trois capteurs photoélectriques A, B, C détecte le passage de chaque véhicule long. Les capteurs A et C sont équidistants de B (distance d). Le signal délivré par un capteur vaut 1 pendant toute la durée du passage d'un véhicule devant celui-ci. On suppose que les longueurs des véhicules satisfont la relation $d < l < 2d < L$. L'espacement de deux véhicules consécutifs peut être quelconque (non nul). Leur vitesse est quelconque, mais ils ne reculent jamais.

On code par les nombres $0, \dots, 7$ les huit vecteurs binaires $a \times b \times c$ selon le code usuel $0 = 0 \times 0 \times 0, 1 = 0 \times 0 \times 1, \dots, 7 = 1 \times 1 \times 1$. En supposant que le passage d'un véhicule long est signalé par un enclenchement de la variable z , on demande de définir la machine M en donnant une expression régulière du langage $\mathbf{L}_z(M)$ pour $z = 1$. On suppose naturellement que les signaux a, b, c sont échantillonnés avec une

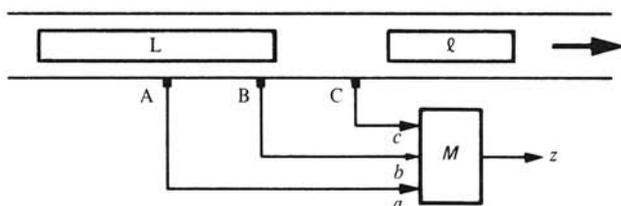


Fig. 4.53

période très courte vis-à-vis du temps de passage d'un véhicule en un point, et de l'intervalle de temps entre les passages de deux véhicules consécutifs en un point.

4.4.25 Solution

On examine ce qui se passe entre un instant t où un véhicule long se trouve devant les trois capteurs ($a \times b \times c = 7$) et un instant t' où il ne se trouve plus que devant C . Les événements possibles entre ces deux instants sont représentés dans les trois "films" de la figure 4.54. Les instants t, t' sont choisis tels que $a(t+1) = 0$ et $b(t'-1) = 1$. Dans le premier des films la séquence $x(t, t') = a(t, t') \times b(t, t') \times c(t, t')$ sur l'alphabet $X = \{0, \dots, 7\}$ appartient au langage 73^+1 . Dans le second elle appartient au langage 73^+7^+5 , et dans le troisième, au langage 73^+5 . Désignons par S la réunion de ces trois langages, qu'on peut écrire $73^+(1 \cup 7^+5 \cup 5)$. On vérifie aisément que pour deux instants $t < t'$, la séquence $x(t, t')$ appartient au langage S seulement si un véhicule long se trouve à l'instant t devant les trois capteurs. Nous dirons que le langage S est l'ensemble des séquences décisives pour la machine M . A un instant t' quelconque, la machine signalera le passage d'un véhicule long ($z(t') = 1$) si et seule-

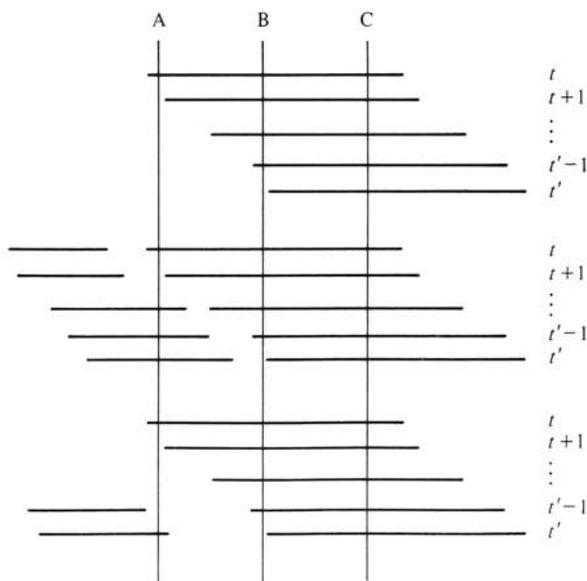


Fig. 4.54

ment s'il existe un instant $t < t'$ tel que $x(t, t') \in S$, autrement dit si et seulement si la séquence $x(1, t')$ se termine par une séquence décisive. Le langage $L_1(M)$ sera donc le langage régulier

$$R_1 = (0 \cup 1 \cup \dots \cup 7)^* 73^+(1 \cup 7^+5 \cup 5)$$

et le langage $L_0(M) = R_0$ sera le complément de R_1 par rapport à X^+ . Le langage R_1 peut être représenté par l'automate de la figure 4.55. A partir de cet automate, la synthèse d'une machine de Mealy s'effectue comme dans les exemples précédents.

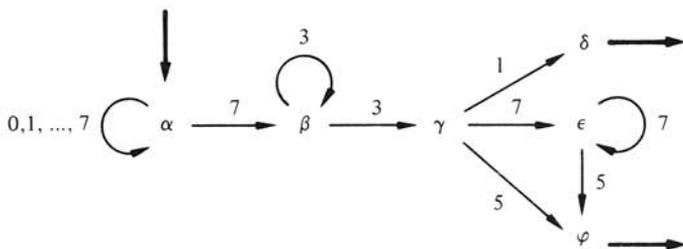


Fig. 4.55

4.4.26 Références bibliographiques

La théorie des langages réguliers et automates finis fait partie de la théorie générale des langages formels et automates, dans laquelle on étudie différentes classes de langages (en particulier celle des langages réguliers) qu'on met en relation avec des classes d'automates (en particulier celle des automates finis), chaque langage d'une certaine classe pouvant être représenté par un automate de la classe correspondante [18]. Cette théorie générale trouve son application principale dans la définition et la compilation des langages de programmation.

La théorie particulière des langages réguliers et automates finis dont la relation a été établie initialement par Kleene [19], a fait l'objet des sections 4.1 à 4.3 de ce volume. On en trouvera de bons exposés dans [6] et [20]. A notre connaissance cependant, l'emploi de diagrammes réguliers dans cette théorie (sect. 4.2, § 4.3.4 à 4.3.8) est particulière à notre exposé.

Il en est de même nous semble-t-il de la manière dont nous avons appliqué cette théorie à la synthèse des machines de Mealy. Il nous est difficile de citer une référence posant le problème de synthèse du paragraphe 4.4.15 avec la même généralité. En langue française, on trouvera une approche plus élémentaire de ce sujet dans [10].

RÉDUCTION DES MACHINES DE MEALY

5.1 SIMULATION ET RÉDUCTION

5.1.1 Introduction

Dans ce chapitre, où il est question essentiellement de machines de Mealy, nous les appellerons souvent machines tout court.

La possibilité de réduire la table de transition (table d'états) d'une machine a été mentionnée dans le volume V (§ 6.1.13), par la présentation d'un exemple dans lequel cette possibilité apparaît avec une évidence particulière. Nous reprenons cet exemple comme introduction au présent chapitre (tab. 5.1, 5.2).

	0 × 0	0 × 1	1 × 1	1 × 0
1	1 ; 0	2 ; 0	- ; -	5 ; 1
2	8 ; 1	2 ; 0	3 ; 0	- ; -
3	- ; -	7 ; 1	3 ; 0	4 ; 0
4	1 ; 0	- ; -	6 ; 1	4 ; 0
5	1 ; 0	- ; -	6 ; 1	5 ; 1
6	- ; -	7 ; 1	6 ; 1	4 ; 0
7	8 ; 1	7 ; 1	3 ; 0	- ; -
8	8 ; 1	2 ; 0	- ; -	5 ; 1

	0 × 0	0 × 1	1 × 1	1 × 0
<i>a</i>	<i>a</i> ; 0	<i>b</i> ; 0	<i>d</i> ; 1	<i>a</i> ; 1
<i>b</i>	<i>b</i> ; 1	<i>b</i> ; 0	<i>c</i> ; 0	<i>a</i> ; 1
<i>c</i>	<i>b</i> ; 1	<i>c</i> ; 1	<i>c</i> ; 0	<i>d</i> ; 0
<i>d</i>	<i>a</i> ; 0	<i>c</i> ; 1	<i>d</i> ; 1	<i>d</i> ; 0

Tableau 5.2

Tableau 5.1

La ligne *a* de la seconde table remplace les lignes 1 et 5 de la première. De même *b* remplace 2 et 8, *c* remplace 3 et 7, et *d* remplace 4 et 6.

Dans d'autres exemples, la possibilité de réduction est beaucoup moins évidente. Le but de ce chapitre est de présenter une méthode de réduction générale. Pour amener cette méthode et la justifier, il convient d'abord de préciser son objectif.

On voit dans l'exemple ci-dessus qu'il s'agit de remplacer une machine $R[X, Y, Z]$ (tab. 5.1) par une machine $R'[X, U, Z]$ (tab. 5.2), ayant les mêmes alphabets primaires et tertiaires X, Z que la première, mais avec un alphabet secondaire U ayant moins d'éléments que Y . Ici $X = \mathbf{B}_2$, $Z = \mathbf{B}$, $Y = \{1, \dots, 8\}$, $U = \{a, b, c, d\}$. Mais ce n'est pas tout. Il faut encore que la machine R' puisse valablement **remplacer** R . Nous dirons que R' doit simuler R , et le but de la présente section sera de donner un sens mathématique précis à cette notion de simulation. Le problème de la réduction sera posé ainsi en des termes exacts se prêtant à un traitement rigoureux.

5.1.2 Définition

Soient $R[X, Y, Z]$ et $R'[X, U, Z]$ deux machines de Mealy. On dit que R' *simule* R , si pour tout élément $p \in Y$ il existe un élément $u \in U$ tel que

$$R'_u \subset R_p. \quad (5.1)$$

5.1.3 Commentaire

Rappelons que R'_u et R_p sont deux correspondances $X^+ \rightarrow Z^+$ (§ 2.6.15). La relation (5.1) signifie (§ 1.3.15) que

$$R'_u(x) \subset R_p(x) \quad (5.2)$$

pour toute séquence $x \in X^+$. Toute réponse z de la machine R'_u à une séquence d'entrée x est aussi une réponse de la machine R_p à cette même séquence x :

$$z \in R'_u(x) \Rightarrow z \in R_p(x). \quad (5.3)$$

La machine R dans l'état initial p peut donc être remplacée par la machine R' dans l'état initial u . Et quel que soit le choix de l'état initial p de R , il existe un état u de R' ayant cette propriété. Tel est le sens de la notion de simulation.

5.1.4 Définition

On dit que la machine $R'[X, U, Z]$ est une *réduction* de la machine $R[X, Y, Z]$, si R' simule R (§ 5.1.2) et si $|U| \leq |Y|$ (le nombre d'éléments de U est inférieur ou égal à celui de Y).

5.1.5 Définition

On dit que $R'[X, U, Z]$ est une *réduction minimale* de $R[X, Y, Z]$ si R' est une réduction de R , et s'il n'existe pas de réduction $R''[X, W, Z]$ de R avec $|W| < |U|$.

5.1.6 Exemple

Soient $X=Z=\{0,1\}$, $Y=\{a,b,c,d,e,f\}$, $U=\{\alpha,\beta\}$, et $R[X, Y, Z]$, $R'[X, U, Z]$ les machines dont les tables de transition sont données ci-dessous (tab. 5.3, 5.4).

	0	1
a	$c; 0$	$d; 1$
b	$c; -$	$d; -$
c	$-; -$	$e; 0$
d	$f; 1$	$-; -$
e	$-; -$	$e; 0$
f	$f; 1$	$-; -$

Tableau 5.3

	0	1
α	$\beta; 0$	$\beta; 1$
β	$\beta; 1$	$\beta; 0$

Tableau 5.4

Nous allons montrer que R' simule R , et plus précisément, en considérant les ensembles

$$\sigma(\alpha) = \{a, b\} \quad \text{et} \quad \sigma(\beta) = \{c, d, e, f\},$$

nous allons montrer que la relation (5.2) est vraie quels que soient $u \in \{\alpha, \beta\}$ et $p \in \sigma(u)$, c'est-à-dire que

$$p \in \sigma(u) \implies R'_u(x) \subset R_p(x) \quad (5.4)$$

pour toute séquence $x \in X^+$. On raisonne par récurrence sur la longueur de x . Considérons d'abord les séquences $x = 0, x = 1$ de longueur 1. On vérifie directement dans les tables de transition que les relations $R'_u(0) \subset R_p(0)$ et $R'_u(1) \subset R_p(1)$ sont vraies pour tous les choix possibles de $u \in \{\alpha, \beta\}$ et $p \in \sigma(u)$. Par exemple pour $u = \alpha$ et $p = b$, on a par (2.82)

$$R'_\alpha(0) = \alpha | 0 = \{0\} \subset R_b(0) = b | 0 = \{0, 1\}.$$

$$R'_\alpha(1) = \alpha | 1 = \{1\} \subset R_b(1) = b | 1 = \{0, 1\}.$$

La vérification pour d'autres choix de u et p est laissée au lecteur.

Supposons que la relation (5.4) soit vraie pour toute séquence x de longueur n . Les séquences sur X de longueur $n+1$ sont toutes de la forme $0x$ ou $1x$, où x est une séquence de longueur n . Il s'agit donc de montrer que les relations $R'_u(0x) \subset R_p(0x)$ et $R'_u(1x) \subset R_p(1x)$ sont vraies pour n'importe quel choix de $u \in \{\alpha, \beta\}$ et de $p \in \sigma(u)$. Grâce à la formule (2.83), ceci peut se faire par vérification directe dans les tables de transition. Par exemple, pour $u = \alpha$ et $p = a$, on peut écrire

$$R'_\alpha(0x) = R'_\alpha(0) R'_{\alpha \cdot 0}(x) = R'_\alpha(0) R'_\beta(x)$$

$$R_a(0x) = R_a(0) R_{a \cdot 0}(x) = R_a(0) R_c(x).$$

Or on a déjà vérifié que $R'_\alpha(0) \subset R_a(0)$, et l'on a $R'_\beta(x) \subset R_c(x)$ parce que $c \in \sigma(\beta)$ et $lg(x) = n$. Il vient donc $R'_\alpha(0x) \subset R_a(0x)$.

Considérons encore $u = \beta$ et $p = c$. On peut écrire

$$R'_\beta(0x) = R'_\beta(0) R'_{\beta \cdot 0}(x) = R'_\beta(0) R'_\beta(x)$$

$$R_c(0x) = R_c(0) R_{c \cdot 0}(x).$$

Ici encore on a déjà vérifié que $R'_\beta(0) \subset R_c(0)$. Il reste à voir que $R'_\beta(x) \subset R_{c \cdot 0}(x)$. Or $c \cdot 0 = \{a, \dots, f\} = Y$ et $R_{c \cdot 0}(x)$ est la réunion des ensembles $R_q(x)$ ($q \in Y$). Si l'on choisit un élément $q \in \sigma(\beta)$, on a d'une part $R_q(x) \subset R_{c \cdot 0}(x)$ et d'autre part $R'_\beta(x) \subset R_q(x)$ puisque $lg(x) = n$.

On voit que sur la base de la définition de la simulation (§ 5.1.2) il est possible de vérifier que R' est une réduction de R , mais que cette vérification est fastidieuse. La section suivante nous fournira une manière beaucoup plus rapide de prouver la simulation de R par R' .

5.1.7 Proposition

Toute machine $R[X, Y, Z]$ est une réduction d'elle-même. Si $R'[X, U, Z]$ est une réduction de $R[X, Y, Z]$ et si $R''[X, W, Z]$ est une réduction de R' , alors R'' est une réduction de R .

La proposition découle immédiatement des définitions (§ 5.1.2, 5.1.4).

5.1.8 Définition

Une machine $R[X, Y, Z]$ est *réduite* si elle est une réduction minimale d'elle-même (§ 5.1.5).

5.1.9 Commentaire

Si R' est une réduction de R , et si R' est réduite, R' n'est pas nécessairement une réduction minimale de R . Nous en verrons un exemple plus loin (§ 5.3.25).

5.2 MACHINES QUOTIENTS

5.2.1 Définition: recouvrements

Soient U et Y deux ensembles non vides. Une correspondance $\sigma : U \rightarrow Y$ (§ 1.3.1) est appelée un *recouvrement* de Y si

$$\bigcup_{u \in U} \sigma(u) = Y. \quad (5.5)$$

Par exemple, si $U = \{\alpha, \beta, \gamma, \epsilon\}$ et $Y = \{a, b, c, d, e\}$, la correspondance $\sigma : U \rightarrow Y$ représentée par le tableau 5.5 est un recouvrement de Y .

u	$\sigma(u)$
α	$\{a, c\}$
β	$\{a, e\}$
γ	$\{b, d\}$
ϵ	\emptyset

Tableau 5.5

p	$\delta(p)$
a	a, b
b	d
c	b
d	—
e	a, d

Tableau 5.6

Les sous-ensembles de Y de la forme $\sigma(u)$ ($u \in U$) sont appelés les *classes* du recouvrement. La relation (5.5) signifie que chaque élément de Y appartient à **au moins** une classe.

La définition générale d'un recouvrement permet que l'on ait $\sigma(u) = \sigma(v)$ pour deux éléments $u \neq v$ de U (ce n'est pas le cas dans l'exemple ci-dessus). Le nombre des classes d'un recouvrement $\sigma : U \rightarrow Y$ peut donc être inférieur à $|U|$.

Notons enfin qu'il existe toujours un élément u de U tel que $\sigma(u) \neq \emptyset$ (Y étant supposé non vide).

5.2.2 Première notion de compatibilité: exemple

Le tableau 5.6 représente une correspondance $\delta : Y \rightarrow Y$, avec $Y = \{a, b, c, d, e\}$. On a $\delta(a) = \{a, b\}$, $\delta(c) = \{b\}$, $\delta(d) = Y$ (tiret), etc. Nous allons mettre en évidence une propriété du recouvrement $\sigma : U \rightarrow Y$ défini par le tableau 5.5 vis-à-vis de cette correspondance δ .

Considérons les éléments α, γ de U . On constate entre α et γ la relation suivante :

$$\forall p \in \sigma(\alpha) : \delta(p) \cap \sigma(\gamma) \neq \emptyset. \tag{5.6}$$

Cette relation se lit : pour tout élément p de $\sigma(\alpha)$, $\delta(p) \cap \sigma(\gamma)$ est non vide. On le vérifie immédiatement, puisque $\delta(a) \cap \sigma(\gamma) = \{b\}$ et $\delta(c) \cap \sigma(\gamma) = \{b\}$.

Abrégeons par $P(\alpha, \gamma)$ cette propriété (5.6). On peut alors vérifier ceci : **pour tout élément u de U ($U = \{\alpha, \dots, \epsilon\}$), il existe un élément v de U tel que $P(u, v)$** . On vérifie en effet

- pour α : $P(\alpha, \gamma)$
- pour β : $P(\beta, \alpha)$
- pour γ : $P(\gamma, \gamma)$
- pour ϵ : $P(\epsilon, v)$ quel que soit $v \in U$.

La dernière assertion concernant ϵ découle d'une propriété fondamentale de l'ensemble vide, à savoir qu'une relation de la forme $x \in \emptyset \Rightarrow R(x)$ est toujours vraie, puisque $\emptyset \subset \{x \mid R(x)\}$. Comme $\sigma(\epsilon) = \emptyset$, on peut écrire $p \in \sigma(\epsilon) \Rightarrow \delta(p) \cap \sigma(v) \neq \emptyset$ quel que soit $v \in U$.

Cette propriété du recouvrement σ vis-à-vis de δ (imprimée en italique gras ci-dessus) s'abrège en disant que σ est *compatible avec* δ .

Considérons maintenant la machine séquentielle $M : X^+ \rightarrow Y^+$ définie par la table de transition ci-dessous (tab. 5.7), avec $X = \{0, 1\}$. On peut considérer que cette table représente deux correspondances $Y \rightarrow Y$: une correspondance δ_0 définie par la colonne 0 (et identique à la correspondance du tableau 5.6), et une correspondance δ_1 définie par la colonne 1. On a $\delta_0(p) = p \cdot 0$ et $\delta_1(p) = p \cdot 1$ pour tout $p \in Y$.

On vérifie que le recouvrement σ est aussi compatible avec δ_1 . En effet, abrégeons par $P_x(u, v)$ (pour $x = 0$ et $x = 1$) la relation

$$\forall p \in \sigma(u) : \delta_x(p) \cap \sigma(v) \neq \emptyset. \tag{5.7}$$

On vérifie alors $P_0(\alpha, \alpha)$, $P_1(\beta, \beta)$, $P_1(\gamma, \alpha)$, $P_1(\epsilon, v)$ (v quelconque).

	0	1
a	a, b	c, e
b	d	—
c	b	a, b, c
d	—	c
e	a, d	e

Tableau 5.7

Cette double compatibilité du recouvrement σ avec δ_0 et δ_1 s'exprime en disant que σ est compatible avec M . Cette définition est généralisée dans le paragraphe suivant.

5.2.3 Définition

Soient $M: X^+ \rightarrow Y^+$ une machine séquentielle, et $\sigma: U \rightarrow Y$ un recouvrement de Y . On dit que σ est *compatible avec M* si pour chaque $u \in U$ et chaque $x \in X$, il existe un $v \in U$ tel que

$$\forall p \in \sigma(u): (p \cdot x) \cap \sigma(v) \neq \emptyset. \quad (5.8)$$

Lorsque $\sigma(u) = \emptyset$, la relation (5.8) est vraie pour n'importe quel élément v de U .

5.2.4 Cas particulier

Nous n'aurons à faire par la suite qu'à des machines séquentielles $M: X^+ \rightarrow Y^+$ de type standard (§ 2.4.34), où pour chaque couple $p \in Y, x \in X$ l'ensemble $p \cdot x$ est soit l'ensemble Y (noté par un tiret), soit un sous-ensemble à un seul élément $\{q\} \subset Y$. Un exemple est donné par le tableau 5.8.

	0	1	2	3
a	b	e	e	b
b	c	d	—	—
c	—	e	a	b
d	a	b	—	—
e	b	—	—	d

Tableau 5.8

u	$\sigma(u)$
α	{a, c}
β	{a, e}
γ	{b, d}
δ	\emptyset

Tableau 5.9

Dans ce cas, la compatibilité d'un recouvrement de Y avec M peut être exprimée commodément en introduisant la notation suivante, où $p \in Y$ et $x \in X$:

$$p + x = \begin{cases} \emptyset & \text{si } p \cdot x = Y \\ p \cdot x & \text{si } p \cdot x \neq Y. \end{cases} \quad (5.9)$$

Par exemple, dans le tableau 5.8, $a + 0 = \{b\}$ et $c + 0 = \emptyset$.

Pour un sous-ensemble $P \subset Y$, et $x \in X$, on écrit naturellement

$$P + x = \left. \bigcup_{p \in P} (p + x) \right\} \quad (5.10)$$

$$\emptyset + x = \emptyset.$$

Ainsi dans le tableau 5.8, $\{a, b, c\} + 0 = \{b, c\}$. Avec cette notation, on peut énoncer la proposition suivante.

5.2.5 Proposition

Soient $M: X^+ \rightarrow Y^+$ une machine séquentielle de type standard, $\sigma: U \rightarrow Y$ un recouvrement de Y , et soient $u \in U, v \in U, x \in X$. Si $\sigma(u) \neq \emptyset$, la relation (5.8) équivaut à

$$\sigma(u) + x \subset \sigma(v) \quad \text{et} \quad \sigma(v) \neq \emptyset. \quad (5.11)$$

5.2.6 Démonstration

Faisons l'hypothèse (5.8), en supposant $\sigma(u) \neq \emptyset$. Alors $\sigma(v) \neq \emptyset$. Soit $p \in \sigma(u)$. Si $p \cdot x = Y$, alors $p + x = \emptyset \subset \sigma(v)$. Si $p \cdot x \neq Y$, alors (5.8) implique

$p \cdot x = p + x \subset \sigma(v)$. Ainsi $p + x \subset \sigma(v)$ quel que soit $p \in \sigma(u)$, donc $\sigma(u) + x \subset \sigma(v)$.

Faisons l'hypothèse (5.11). Soit $p \in \sigma(u)$. Si $p \cdot x = Y$, alors $(p \cdot x) \cap \sigma(v) = \sigma(v) \neq \emptyset$. Si $p \cdot x \neq Y$, alors $p \cdot x = p + x \subset \sigma(v)$, d'où $(p \cdot x) \cap \sigma(v) = p \cdot x \neq \emptyset$.

5.2.7 Corollaire

Soient $M: X^+ \rightarrow Y^+$ une machine séquentielle de type standard, et $\sigma: U \rightarrow Y$ un recouvrement de Y . Pour que σ soit compatible avec M , il faut et il suffit que la condition suivante soit vérifiée: pour tout $u \in U$, et pour tout $x \in X$, il existe un $v \in U$ tel que

$$\sigma(u) + x \subset \sigma(v). \quad (5.12)$$

En effet la condition est nécessaire en vertu du paragraphe 5.2.5. Elle est suffisante, car s'il existe un $v \in U$ vérifiant (5.12), il en existe un tel que $\sigma(v) \neq \emptyset$.

5.2.8 Exemple

Le recouvrement σ de l'ensemble $Y = \{a, b, c, d, e\}$ donné dans le tableau 5.9 est compatible avec la machine séquentielle M du tableau 5.8. On le vérifie rapidement en appliquant le corollaire ci-dessus. Par exemple:

$$\begin{aligned} \sigma(\alpha) + 0 &= \{a, c\} + 0 = \{b\} \subset \sigma(\gamma) \\ \sigma(\alpha) + 2 &= \{a, c\} + 2 = \{a, e\} \subset \sigma(\beta) \\ \sigma(\gamma) + 2 &= \{b, d\} + 2 = \emptyset \subset \sigma(v) \quad (\forall v \in U) \\ \sigma(\delta) + x &= \emptyset + x = \emptyset \subset \sigma(v) \quad (\forall v \in U). \end{aligned}$$

5.2.9 Définition

Soient $R[X, Y, Z, M, F]$ une machine de Mealy et $\sigma: U \rightarrow Y$ un recouvrement de Y . Rappelons que dans cette notation (§ 2.6.8), M est la composante séquentielle, et F la composante combinatoire de R . On dit que σ est compatible avec F si pour tout $u \in U$ et tout $x \in X$,

$$\bigcap_{p \in \sigma(u)} (p|x)_R \neq \emptyset. \quad (5.13)$$

On dit que σ est compatible avec R s'il est compatible avec M (§ 5.2.3) et avec F . Notons que

$$\sigma(u) = \emptyset \implies \bigcap_{p \in \sigma(u)} (p|x)_R = Z \quad (5.14)$$

car l'intersection des ensembles $(p|x)_R$ tels que $p \in \sigma(u)$ est l'ensemble des éléments z de Z tels que

$$p \in \sigma(u) \implies z \in (p|x)_R. \quad (5.15)$$

Comme on l'a vu plus haut (§ 5.2.2), une relation de la forme $p \in \emptyset \implies R$ est toujours vraie, de sorte que pour $\sigma(u) = \emptyset$, la relation (5.15) est vraie quel que soit $z \in Z$. Par suite la relation (5.13) est toujours vraie lorsque $\sigma(u) = \emptyset$.

5.2.10 Exemple

Soit $R[X, Y, Z]$ la machine définie par le tableau 5.10. Sa composante séquentielle est la machine de type standard déjà vue dans le tableau 5.8. L'alphabet Z est l'alphabet à quatre éléments

$$Z = \mathbf{B}_2 = \{0 \times 0, 0 \times 1, 1 \times 0, 1 \times 1\},$$

les éléments de Z étant notés 00, 01, 10, 11. Un sous-ensemble de Z tel que $\{00, 01\}$ est noté $0-$. L'ensemble Z lui-même est noté $--$.

	0	1	2	3
a	$b; --$	$e; 0-$	$e; 00$	$b; -1$
b	$c; 00$	$d; --$	$-; -1$	$-; -0$
c	$-; -1$	$e; -1$	$a; 00$	$b; 01$
d	$a; -0$	$b; 1-$	$-; --$	$-; --$
e	$b; 0-$	$-; -0$	$-; -0$	$d; 11$

Tableau 5.10

u	$\sigma(u)$
α	$\{a, c\}$
β	$\{a, e\}$
γ	$\{b, d\}$
δ	\emptyset

Tableau 5.11

Le recouvrement $\sigma: U \rightarrow Y$ rappelé dans le tableau 5.11 est compatible avec R . Nous savons déjà qu'il est compatible avec la composante séquentielle de R (§ 5.2.8). Il reste à vérifier (5.13) pour $u = \alpha, \beta, \gamma$ et $x = 0, 1, 2, 3$. Considérons par exemple $\sigma(\alpha) = \{a, c\}$ et $x = 1$. On a

$$a|1 = 0- = \{0 \times 0, 0 \times 1\}$$

$$c|1 = -1 = \{0 \times 1, 1 \times 1\}$$

$$(a|1) \cap (c|1) = \{0 \times 1\} \neq \emptyset.$$

La condition (5.13) est donc vérifiée pour $u = \alpha$ et $x = 1$. Le lecteur pourra la vérifier pour les autres valeurs de u et x .

Supposons qu'un recouvrement ρ de Y possède une classe $\rho(u) = \{a, b\}$. Alors ce recouvrement ne serait pas compatible avec la composante combinatoire de R , car

$$a|3 = \{0 \times 1, 1 \times 1\}$$

$$b|3 = \{0 \times 0, 1 \times 0\}$$

$$(a|3) \cap (b|3) = \emptyset.$$

5.2.11 Définition: machine quotient

Soient $R[X, Y, Z]$ une machine de Mealy et $\sigma: U \rightarrow Y$ un recouvrement de Y compatible avec R . On appelle *machine quotient de R par σ* la machine $R'[X, U, Z]$ définie par les relations (5.16) et (5.17), valables pour des éléments $u \in U$ et $x \in X$ quelconques, et déterminant la table de transition de R' .

$$(u \cdot x)_{R'} = \{v \in U \mid \forall p \in \sigma(u) : (p \cdot x)_R \cap \sigma(v) \neq \emptyset\} \quad (5.16)$$

$$(u|x)_{R'} = \bigcap_{p \in \sigma(u)} (p|x)_R. \quad (5.17)$$

La compatibilité de σ avec R (§ 5.2.3, 5.2.9) garantit que les membres de droite des égalités (5.16), (5.17) sont des sous-ensembles *non vides* de U et Z , et c'est ce qui

permet de prendre ces relations comme définition d'une machine $R'[X,U,Z]$. Cette définition entraîne

$$\sigma(u) = \emptyset \implies \begin{cases} (u \cdot x)_{R'} = U \\ (u|x)_{R'} = Z \end{cases} \quad (5.18)$$

comme nous l'avons vu aux paragraphes 5.2.3, 5.2.9.

Au cas où la composante séquentielle de R est de *type standard*, la relation (5.16) équivaut à

$$(u \cdot x)_{R'} = \begin{cases} \{v \in U \mid \sigma(u) + x \subset \sigma(v) \text{ et } \sigma(v) \neq \emptyset\} & \text{si } \sigma(u) \neq \emptyset \\ U & \text{si } \sigma(u) = \emptyset. \end{cases} \quad (5.19)$$

5.2.12 Exemple

Soient $R[X,Y,Z]$ la machine du tableau 5.10, et $\sigma:U \rightarrow Y$ le recouvrement du tableau 5.11, compatible avec R . La machine quotient $R'[X,U,Z]$ de R par σ a pour table de transition le tableau 5.12, construit selon (5.17), (5.18), (5.19). Soient par exemple $u = \alpha$ et $x = 0$. On a $\sigma(\alpha) + 0 = \{a,c\} + 0 = \{b\}$. Le seul élément v de U tel que $\sigma(\alpha) + 0 \subset \sigma(v)$ est γ . On a donc $(\alpha \cdot 0)_{R'} = \{\gamma\}$ en vertu de (5.19). D'autre part, en vertu de (5.17),

$$(\alpha|0)_{R'} = (a|0)_R \cap (c|0)_R = \{0 \times 1, 1 \times 1\}.$$

Si l'on applique (5.19) à $u = \gamma$ et $x = 2$, il vient $\sigma(\gamma) + 2 = \{b,d\} + 2 = \emptyset$, donc $(\gamma \cdot 2)_{R'} = \{\alpha, \beta, \gamma\}$, noté $\alpha\beta\gamma$ dans le tableau 5.12. Enfin $(\delta \cdot x)_{R'} = U$ (tiret) quel que soit x , en vertu de (5.18). Le lecteur contrôlera par lui-même le reste du tableau.

	0	1	2	3
α	$\gamma; -1$	$\beta; 0 \ 1$	$\beta; 0 \ 0$	$\gamma; 0 \ 1$
β	$\gamma; 0 \ -$	$\beta; 0 \ 0$	$\beta; 0 \ 0$	$\gamma; 1 \ 1$
γ	$\alpha; 0 \ 0$	$\gamma; 1 \ -$	$\alpha\beta\gamma; -1$	$\alpha\beta\gamma; -0$
δ	$-; ---$	$-; ---$	$-; ---$	$-; ---$

Tableau 5.12

Nous verrons au paragraphe suivant que toute machine quotient d'une machine R simule R . Sachant cela, on peut affirmer que la machine R' (tab. 5.12) est une réduction de R (tab. 5.10). Ce n'est pas une réduction minimale, car le recouvrement σ (tab. 5.13) est aussi compatible avec R , et fournit la machine quotient du tableau 5.14.

u	$\sigma(u)$
α	$\{a, c\}$
β	$\{a, e\}$
γ	$\{b, d\}$

Tableau 5.13

	0	1	2	3
α	$\gamma; -1$	$\beta; 0 \ 1$	$\beta; 0 \ 0$	$\gamma; 0 \ 1$
β	$\gamma; 0 \ -$	$\beta; 0 \ 0$	$\beta; 0 \ 0$	$\gamma; 1 \ 1$
γ	$\alpha; 0 \ 0$	$\gamma; 1 \ -$	$-; -1$	$-; -0$

Tableau 5.14

Il est clair qu'un recouvrement possédant une classe vide (tab. 5.11) ne donne jamais pour machine quotient une réduction minimale, puisque la suppression de cette classe vide ne détruit pas la compatibilité du recouvrement avec R .

La seconde machine quotient (tab. 5.14) possède une composante séquentielle de type standard. C'est un fait accidentel. Si l'on modifiait légèrement le tableau 5.10, en remplaçant $b \cdot 0 = \{c\}$ par $b \cdot 0 = Y$ (tiret), on aurait $\sigma(\gamma) + 0 = \{b, d\} + 0 = \{a\}$. Or $\{a\}$ est contenu dans $\sigma(\alpha)$ et $\sigma(\beta)$. Il viendrait alors $\gamma \cdot 0 = \{\alpha, \beta\}$ dans le tableau 5.14.

5.2.13 Proposition

Soient $R[X, Y, Z]$ une machine de Mealy, $\sigma: U \rightarrow Y$ un recouvrement de Y compatible avec R , et $R'[X, U, Z]$ la machine quotient de R par σ . Si $p \in Y$ et $u \in U$, on a

$$p \in \sigma(u) \implies R'_u \subset R_p. \quad (5.20)$$

Par suite, la machine R' simule R (§ 5.1.2), puisque chaque élément $p \in Y$ appartient à au moins une classe $\sigma(u)$.

5.2.14 Démonstration

Nous allons montrer que la relation

$$p \in \sigma(u) \implies R'_u(x) \subset R_p(x) \quad (5.21)$$

est vraie quels que soient $p \in Y$, $u \in U$, $x \in X^+$, par récurrence sur la longueur de x . Supposons que $lg(x) = 1$, et que $p \in \sigma(u)$. Alors

$$R'_u(x) = (u|x)_{R'} = \bigcap_{q \in \sigma(u)} (q|x)_R \quad \text{selon (2.82) et (5.17)}$$

$$R_p(x) = (p|x)_R.$$

Comme p appartient à $\sigma(u)$, l'intersection des ensembles $(q|x)_R$ tels que $q \in \sigma(u)$ est contenue dans l'ensemble $(p|x)_R$. Donc $R'_u(x) \subset R_p(x)$, et la relation (5.21) est prouvée pour $lg(x) = 1$.

Supposons que (5.21) soit vraie pour $lg(x) = n$. Pour la clarté de ce qui suit, nous l'écrivons en changeant de symboles. Nous faisons l'hypothèse que la relation

$$q \in \sigma(v) \implies R'_v(x') \subset R_q(x') \quad (5.22)$$

est vraie quels que soient $q \in Y$, $v \in U$, $x' \in X^n$. Considérons alors des éléments $u \in U$, $p \in Y$, $x \in X$, $x' \in X^n$. Nous allons voir que

$$p \in \sigma(u) \implies R'_u(xx') \subset R_p(xx'), \quad (5.23)$$

donc que (5.21) est vraie pour $lg(x) = n + 1$. On a en effet par (2.83) et (4.100),

$$R'_u(xx') = R'_u(x) \bigcup_{v \in V} R'_v(x') \quad \text{avec } V = (u \cdot x)_{R'}$$

et de même

$$R_p(xx') = R_p(x) \bigcup_{q \in Q} R_q(x') \quad \text{avec } Q = (p \cdot x)_R.$$

Supposons que $p \in \sigma(u)$. Alors $R'_u(x) \subset R_p(x)$ puisque $lg(x) = 1$. Il reste à montrer que

$$\bigcup_{v \in V} R'_v(x') \subset \bigcup_{q \in Q} R_q(x'). \quad (5.24)$$

Or $\nu \in V$ implique $\sigma(\nu) \cap Q \neq \emptyset$ par définition de V et Q , par (5.16), et par l'hypothèse $p \in \sigma(u)$. Pour chaque $\nu \in V$, il existe donc un $q \in Q$ tel que $q \in \sigma(\nu)$. Alors (5.24) découle de (5.22).

5.2.15 Exemple

La machine R' (tab. 5.12) simule la machine R (tab. 5.10), puisque R' est la machine quotient de R par σ (tab. 5.11). Plus précisément, selon (5.20), on a

$$R'_\alpha \subset R_a; R'_\alpha \subset R_c$$

$$R'_\beta \subset R_a; R'_\beta \subset R_e$$

$$R'_\gamma \subset R_b; R'_\gamma \subset R_d.$$

Ainsi la machine R avec l'état initial a peut être remplacée par la machine R' avec l'état initial α ou β , etc.

5.2.16 Remarque

La proposition du paragraphe 5.2.13 ne suppose pas que la composante séquentielle de R soit de type standard. Elle est valable pour le type de machine le plus général. Cependant les exemples de réduction qui suivront porteront toujours sur des composantes séquentielles de type standard.

5.2.17 Exercice

Vérifier que le recouvrement $\sigma : \{\alpha, \beta\} \rightarrow \{a, \dots, f\}$ défini au paragraphe 5.1.6 est compatible avec la machine R du tableau 5.3, et que R' (tab. 5.4) est la machine quotient de R par σ .

5.2.18 Exercice

Pour chacune des machines $R[X, Y, Z]$ données par les tables de transition ci-dessous (tab. 5.15 à 5.18), trouver un recouvrement de Y , $\sigma : U \rightarrow Y$, compatible avec R , tel que $|U| < Y$, et construire la machine quotient de R par σ . L'alphabet Z est dans chacun des cas \mathbf{B} ou \mathbf{B}_2 .

Il existe une méthode pour résoudre ce problème, qui sera exposée dans les sections suivantes. Mais il est bon de se familiariser avec le sujet, en traitant de façon heuristique quelques exemples simples comme ceux-ci.

	<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>
1	1; 0	1; 0	2; 0	3; 1
2	—; —	—; —	2; 0	3; 1
3	3; 1	1; 0	4; 1	3; 1
4	4; 1	1; 0	4; 1	3; 1

Tableau 5.15

	<i>a</i>	<i>b</i>	<i>c</i>
1	1; 0	1; —	2; 0
2	—; —	1; —	4; 1
3	1; —	1; —	2; 0
4	1; 1	1; 1	5; 0
5	3; 0	1; 0	3; 1

Tableau 5.16

	0	1
a	b ; 1	h ; 1
b	a ; 0	g ; 0
c	- ; 0	f ; -
d	d ; 0	- ; 1
e	c ; 0	d ; -
f	a ; -	c ; 0
g	- ; -	b ; -
h	g ; 0	e ; -

Tableau 5.17

	a	b	c	d
1	1 ; 0 -	2 ; -1	- ; - -	- ; - -
2	3 ; 1 1	2 ; 0 1	5 ; 1 -	- ; - -
3	3 ; 1 1	- ; - -	- ; - -	4 ; 0 -
4	6 ; 1 0	4 ; 0 0	- ; - -	4 ; 0 1
5	- ; - -	4 ; - 0	5 ; - 1	4 ; 0 -
6	6 ; 1 0	- ; - -	- ; - -	7 ; 1 -
7	1 ; 0 -	- ; - -	- ; - -	7 ; 1 1

Tableau 5.18

5.2.19 Commentaire

Nous avons vu qu'étant donné une machine $R[X, Y, Z]$, lorsqu'on a trouvé un recouvrement $\sigma: U \rightarrow Y$ compatible avec R , tel que $|U| \leq |Y|$, on peut construire une réduction $R'[X, U, Z]$ de R , à savoir la machine quotient de R par σ . La question à laquelle sont consacrés les paragraphes suivants est de savoir si toute réduction de R peut être obtenue de cette manière.

5.2.20 Proposition

Soient $R[X, Y, Z]$, $R'[X, U, Z]$ deux machines, et soient $p \in Y$, $u \in U$, $x \in X$. On a

$$R'_u \subset R_p \implies R'_{u \cdot x} \subset R_{p \cdot x}. \quad (5.25)$$

Par suite, si $v \in U$ et $q \in Y$,

$$\left. \begin{array}{l} R'_u \subset R_p \\ \{q\} = (p \cdot x)_R \\ v \in (u \cdot x)_{R'} \end{array} \right\} \implies R'_v \subset R_q. \quad (5.26)$$

5.2.21 Démonstration

Supposons que $R'_u \subset R_p$, et soit x' une séquence quelconque sur X . On a par hypothèse

$$R'_u(xx') \subset R_p(xx')$$

et en appliquant (2.83),

$$\underbrace{R'_u(x)}_A \underbrace{R'_{u \cdot x}(x')}_B \subset \underbrace{R_p(x)}_{A'} \underbrace{R_{p \cdot x}(x')}_{B'}.$$

On peut appliquer aux ensembles désignés ci-dessus par A, B, A', B' la proposition du paragraphe 1.4.11, car on a ici $A \cup A' \subset Z$ et $B \cup B' \subset Z^n$, n étant la longueur de x' . Il vient $B \subset B'$, c'est-à-dire

$$R'_{u \cdot x}(x') \subset R_{p \cdot x}(x').$$

Ceci étant vrai quelle que soit la séquence x' , on peut écrire $R'_{u \cdot x} \subset R_{p \cdot x}$. Si l'on suppose que $v \in (u \cdot x)$, il vient $R'_v \subset R'_{u \cdot x}$ selon (4.100), donc $R'_v \subset R_{p \cdot x}$. Si de plus $p \cdot x = \{q\}$, alors $R'_v \subset R_q$.

5.2.22 Proposition

Soient $R[X, Y, Z]$ une machine à composante séquentielle *de type standard*, et $R'[X, U, Z]$ une machine simulant R . Pour chaque élément $u \in U$ soit $\sigma(u)$ l'ensemble des $p \in Y$ tels que $R'_u \subset R_p$. La correspondance $\sigma: U \rightarrow Y$ est un recouvrement de Y , compatible avec R , et l'on a plus précisément les relations suivantes :

$$v \in (u \cdot x)_{R'} \Rightarrow \sigma(u) + x \subset \sigma(v) \quad (5.27)$$

$$(u|x)_{R'} \subset \bigcap_{p \in \sigma(u)} (p|x)_R. \quad (5.28)$$

5.2.23 Démonstration

Par définition, l'on a

$$p \in \sigma(u) \iff R'_u \subset R_p. \quad (5.29)$$

Comme R' simule R (§ 5.1.2), σ est un recouvrement de Y . Établissons (5.27). On suppose que $v \in (u \cdot x)_{R'}$. Soit $q \in \sigma(u) + x$. On a $\{q\} = (p \cdot x)_R$ pour un $p \in \sigma(u)$. En vertu de (5.29) et (5.26) il vient $q \in \sigma(v)$. La relation (5.27) est ainsi prouvée. Elle entraîne selon le paragraphe 5.2.7 que σ est compatible avec la composante séquentielle de R . La relation (5.28) découle immédiatement de (5.29) et (2.82). Elle montre que σ est compatible avec la composante combinatoire de R (5.13).

5.2.24 Commentaire

La proposition (§ 5.2.22) n'implique pas que R' soit la machine quotient de R par σ . Mais en considérant la machine quotient $R''[X, U, Z]$ de R par σ , on voit que pour toute réduction R' d'une machine R à composante séquentielle de type standard, il existe une réduction R'' de R ayant le même nombre d'états secondaires que R' , et pouvant être obtenue comme quotient de R par un recouvrement σ . Ceci répond partiellement à la question du paragraphe 5.2.19.

5.2.25 Définition

Soient $R'[X, U, Z]$ et $R''[X, U, Z]$ deux machines avec les mêmes alphabets X, U, Z . Nous dirons que R' est une *diminution* de R'' si l'on a, quels que soient $u \in U$ et $x \in X$:

$$\left. \begin{aligned} (u \cdot x)_{R'} &\subset (u \cdot x)_{R''} \\ (u|x)_{R'} &\subset (u|x)_{R''} \end{aligned} \right\} \quad (5.30)$$

5.2.26 Proposition

Si $R'[X, U, Z]$ est une diminution de $R''[X, U, Z]$, alors $R'_u \subset R''_u$ quel que soit $u \in U$. Par suite R' simule R'' .

5.2.27 Démonstration

La relation $R'_u(x) \subset R''_u(x)$ est vraie pour $lg(x) = 1$ en vertu de (5.30) et (2.82). Supposons qu'elle soit vraie pour $lg(x) \leq n$. Soient $x \in X$ et $x' \in X^n$. On a

selon (2.83),

$$\begin{aligned}
 R'_u(xx') &= R'_u(x) \bigcup_{v \in Q'} R'_v(x') && \text{avec } Q' = (u \cdot x)_{R'} \\
 &\subset R'_u(x) \bigcup_{v \in Q} R'_v(x') && \text{avec } Q = (u \cdot x)_R \supset Q' \\
 &\subset R''_u(x) \bigcup_{v \in Q} R''_v(x') && \text{par hypothèse} \\
 &= R''_u(xx').
 \end{aligned}$$

5.2.28 Proposition

On considère à nouveau les machines $R[X, Y, Z]$, $R'[X, U, Z]$ et le recouvrement $\sigma: U \rightarrow Y$ du paragraphe 5.2.22. On fait l'hypothèse supplémentaire que σ n'a pas de classe vide, autrement dit que pour tout $u \in U$ il existe un $p \in Y$ tel que $R'_u \subset R_p$. Soit alors $R''[X, U, Y]$ la machine quotient de R par σ . Avec ces hypothèses, R' est une diminution de R'' .

5.2.29 Démonstration

Comme σ n'a pas de classe vide, la relation (5.27) peut s'écrire

$$v \in (u \cdot x)_{R'} \implies v \in (u \cdot x)_{R''}$$

en vertu de (5.19) appliqué à R'' . D'autre part, en vertu de (5.17) appliqué à R'' , la relation (5.28) s'écrit

$$(u|x)_{R'} \subset (u|x)_{R''}.$$

5.2.30 Définition

Nous dirons qu'une réduction $R'[X, U, Z]$ d'une machine $R[X, Y, Z]$ possède un *état superflu* $u \in U$, si pour cet élément u il n'existe pas d'élément $p \in Y$ tel que $R'_u \subset R_p$. Cela revient à dire que le recouvrement $\sigma: U \rightarrow Y$ défini par (5.29) possède une classe $\sigma(u) = \emptyset$.

5.2.31 Proposition

Soient $R[X, Y, Z]$ une machine à composante séquentielle *de type standard*, et $R'[X, U, Z]$ une réduction minimale de R . La machine R' n'a pas d'état superflu.

5.2.32 Démonstration

Supposons que le recouvrement $\sigma: U \rightarrow Y$ défini par (5.29) possède une classe $\sigma(v) = \emptyset$. Alors la famille des classes $\sigma(u)$ telles que $u \neq v$ constitue aussi un recouvrement compatible avec R (§ 5.2.7), et permet de construire une machine quotient $R''[X, V, Z]$ de R , avec $V = U - \{u\}$. Par suite R' n'est pas minimale.

5.2.33 Conclusions

Les propositions qui précèdent (§ 5.2.22, 5.2.28) permettent de donner la réponse suivante à la question du paragraphe 5.2.19.

Si $R[X, Y, Z]$ est une machine à composante séquentielle de type standard, toute réduction $R'[X, U, Z]$ sans état superflu peut être obtenue en effectuant les opérations suivantes :

- trouver un recouvrement $\sigma : U \rightarrow Y$ compatible avec R et sans classe vide;
- construire la machine quotient R'' de R par σ ;
- diminuer R'' au sens du paragraphe 5.2.25, c'est-à-dire construire R' selon (5.30).

Réciproquement, on peut affirmer en vertu des paragraphes 5.2.13, 5.2.26 et 5.1.7, que ces trois opérations, effectuées de façon quelconque, c'est-à-dire quels que soient le recouvrement trouvé (sans classe vide) et la diminution effectuée sur R'' , fournissent une réduction R' de R , sans état superflu. De plus cette assertion réciproque reste valable au cas où la composante séquentielle de R n'est pas de type standard. Par contre nous ne sommes pas en mesure d'affirmer que dans ce cas les trois opérations permettent de déterminer *toutes* les réductions R' (sans état superflu) de R .

Il nous reste à exposer une méthode pour la première des trois opérations. C'est l'objet des sections suivantes.

5.3. CLASSES DE COMPATIBILITÉ

5.3.1 Introduction

Dans cette section et la suivante, R désigne toujours une machine à composante séquentielle de type standard, et X, Y, Z sont les alphabets de R (machine $R[X, Y, Z]$). On cherche les réductions de R sans état superflu (§ 5.2.30), et en particulier (§ 5.2.31), les réductions minimales de R .

On sait (§ 5.2.33) que le problème de trouver une telle réduction se ramène au **problème** suivant : trouver un recouvrement $\sigma : U \rightarrow Y$ compatible avec R et sans classe vide.

Dans l'exposé qui suit, on prendra toujours pour U un ensemble de la forme $\{1, 2, \dots, n\}$, au lieu de $\{\alpha, \beta, \dots\}$. Les classes $\sigma(1), \dots, \sigma(n)$ seront notées C_1, \dots, C_n , et le recouvrement σ sera noté (C_1, \dots, C_n) .

Le terme de recouvrement désigne toujours un recouvrement de Y . La proposition suivante n'est qu'une traduction, dans ces nouvelles notations, de la définition d'un recouvrement compatible avec R .

5.3.2 Proposition

Soient C_1, \dots, C_n des sous-ensembles de Y . Le n -uple $\sigma = (C_1, \dots, C_n)$ est un recouvrement compatible avec R si et seulement s'il vérifie les trois conditions suivantes.

- **Condition de couverture** (5.31)
Chaque élément $p \in Y$ appartient à l'un au moins des ensembles C_i ($i = 1, \dots, n$).
- **Condition de fermeture** (5.32)
Pour chaque ensemble C_i ($i = 1, \dots, n$) et pour chaque $x \in X$, il existe un ensemble C_j ($j = 1, \dots, n$) tel que $C_i + x \subset C_j$.

• **Condition Z** (5.33)

Pour chacun des ensembles $C_i (i = 1, \dots, n)$, on a quel que soit $x \in X$:

$$\bigcap_{p \in C_i} (p|x)_R \neq \emptyset.$$

La condition de couverture exprime que σ est un recouvrement; la condition de fermeture, que σ est compatible avec la composante séquentielle de R (§ 5.2.7); la condition Z, que σ est compatible avec la composante combinatoire de R (§ 5.2.9).

5.3.3 Définition : classes de compatibilité

On appelle classe de compatibilité de R tout sous-ensemble C non vide de Y qui est une classe d'un recouvrement compatible avec R .

5.3.4 Proposition

Si $Y = \{y_1, \dots, y_n\}$, alors le n -uplet $\sigma = (\{y_1\}, \dots, \{y_n\})$ est un recouvrement compatible avec R . Par suite chaque sous-ensemble à un seul élément $\{y\}$ de Y est une classe de compatibilité de R .

En effet le σ considéré vérifie de façon triviale les conditions (5.31), (5.32), (5.33). On dit qu'il s'agit du *recouvrement trivial*.

5.3.5 Remarque

Soient C_1, \dots, C_n des classes de compatibilité de R . Pour que le n -uplet $\sigma = (C_1, \dots, C_n)$ soit un recouvrement compatible avec R , il faut et il suffit qu'il vérifie la condition de couverture (5.31) et la condition de fermeture (5.32).

En effet chacun des ensembles C_i est une classe d'un recouvrement σ_i compatible avec R (§ 5.3.3) et vérifie ipso facto la condition (5.33).

En vertu de cette remarque, pour trouver les recouvrements compatibles avec R , on effectuera les opérations suivantes :

- déterminer *toutes* les classes de compatibilité de R ;
- choisir n classes de compatibilité C_1, \dots, C_n telles que (C_1, \dots, C_n) vérifie les conditions de couverture et de fermeture.

La présente section est consacrée à la première de ces opérations. On connaît déjà toutes les classes de compatibilité à un élément (§ 5.3.4). Une méthode exposée plus loin (§ 5.3.10) permet de déterminer de façon récurrente les classes de compatibilité à n éléments lorsqu'on connaît les classes à m éléments pour $m = 1, \dots, n - 1$.

5.3.6 Proposition

Si C est une classe de compatibilité de R , alors tout sous-ensemble non vide K de C est aussi une classe de compatibilité de R .

5.3.7 Démonstration

L'hypothèse faite sur C revient à dire que C est l'une des classes d'un recouvrement (C_1, \dots, C_n) compatible avec R . Supposons par exemple que $C = C_1$. Soit K

un sous ensemble de C . Posons $\sigma = (C_1, \dots, C_n, K)$. Il est facile de voir que σ est un recouvrement compatible avec R . En effet la condition (5.31) est vérifiée. La condition (5.32) est vérifiée pour $i = 1, \dots, n$. Pour $i = n + 1$, c'est-à-dire pour $C_{n+1} = K$, on a $K + x \subset C_1 + x$ quel que soit $x \in X$ de sorte que (5.32) est vérifiée. De même, la condition (5.33) est vérifiée pour $i = 1, \dots, n$. Pour $i = n + 1$ elle est aussi vérifiée, car $K \subset C$ implique

$$\bigcap_{p \in C} (p|x) \subset \bigcap_{p \in K} (p|x).$$

5.3.8 Proposition

Soient C une classe de compatibilité de R , et $x \in X$. Si l'ensemble $C + x$ n'est pas vide, c'est une classe de compatibilité de R . De plus $|C + x| \leq |C|$.

5.3.9 Démonstration

Soit (C_1, \dots, C_n) un recouvrement compatible avec R , tel que $C_1 = C$. Soit $K = C + x$. En vertu de (5.32), on a $K \subset C_j$ pour un $j \in \{1, \dots, n\}$. Comme C_j est une classe de compatibilité, si $K \neq \emptyset$, K est une classe de compatibilité en vertu du paragraphe 5.3.6. La relation $|C + x| \leq |C|$ est évidente.

5.3.10 Recherche des classes de compatibilité : méthode générale

Nous désignons par $P_n(Y)$ l'ensemble des sous-ensembles (ou parties) à n éléments de Y ; par $\Gamma(n)$ l'ensemble des classes de compatibilité à n éléments; par $\Omega(n)$ l'ensemble des classes de compatibilité C telles que $|C| \leq n$. On a

$$\Gamma(0) = \Omega(0) = \emptyset \tag{5.34}$$

puisque par définition, aucune classe de compatibilité n'est vide, et

$$\Gamma(1) = \Omega(1) = P_1(Y) \tag{5.35}$$

en vertu du paragraphe 5.3.4. Pour $n > 1$,

$$\Omega(n) = \Gamma(1) \cup \dots \cup \Gamma(n-1) \cup \Gamma(n) = \Omega(n-1) \cup \Gamma(n). \tag{5.36}$$

Pour déterminer toutes les classes de compatibilité, on dispose d'une méthode récurrente. On connaît $\Gamma(1)$. Supposons que l'on ait déterminé $\Gamma(1), \dots, \Gamma(n-1)$, c'est-à-dire $\Omega(n-1)$ pour un $n > 1$. On cherche $\Gamma(n)$, sachant que $\Gamma(n) \subset P_n(Y)$. L'idée de méthode est de partir d'un ensemble Γ de sous-ensemble de Y , tel que

$$\Gamma \subset P_n(Y) \tag{5.37}$$

et dont on sait qu'il contient $\Gamma(n)$:

$$\Gamma(n) \subset \Gamma. \tag{5.38}$$

On se propose alors de procéder par élimination, c'est-à-dire de supprimer progressivement les ensembles $C \in \Gamma$ qui n'appartiennent pas à $\Gamma(n)$. Comme on ne connaît pas $\Gamma(n)$, il faut disposer de critères de suppression, et il faut que ces critères suffisent à éliminer tous les $C \in \Gamma$ tels que $C \notin \Gamma(n)$. A cet effet, on impose à l'ensemble de départ Γ une condition supplémentaire. On sait que toute classe de compatibilité C véri-

fie la condition (5.33);

$$\bigcap_{p \in C} (p|x) \neq \emptyset \quad \text{quel que soit } x \in X. \quad (5.39)$$

On imposera donc à Γ la condition

$$\bigcap_{p \in C} (p|x) \neq \emptyset \quad \text{quels que soient } C \in \Gamma \quad \text{et } x \in X. \quad (5.40)$$

Il existe toujours un ensemble Γ vérifiant (5.37), (5.38), et (5.40): c'est l'ensemble de **tous** les ensembles $C \in \mathbf{P}_n(Y)$ qui vérifient (5.39).

Pour accélérer le processus d'élimination, on peut imposer à Γ des conditions supplémentaires qui diminuent sa taille au départ. Par exemple, on sait que si $C \in \Gamma(n)$, alors en vertu du paragraphe 5.3.6:

$$(K \subset C \text{ et } |K| = |C| - 1) \implies K \in \Gamma(n-1). \quad (5.41)$$

On peut donc imposer à l'ensemble de départ Γ la condition supplémentaire suivante:

$$\begin{aligned} &\text{Pour tout } C \in \Gamma, \text{ chaque sous-ensemble } K \text{ de } C \\ &\text{tel que } |K| = n-1 \text{ appartient à } \Gamma(n-1). \end{aligned} \quad (5.42)$$

Les conditions (5.40) et (5.42) sont vérifiables au moyen de la table de transition de R , et de la liste des ensembles $K \in \Gamma(n-1)$ supposée connue. Nous appellerons *ensemble de départ canonique* Γ , l'ensemble de **tous** les $C \in \mathbf{P}_n(Y)$ qui vérifient (5.39) et (5.41).

Cherchons maintenant des *critères d'élimination*. Nous en aurons deux. Premièrement, considérons un ensemble $C \in \Gamma$, et supposons que l'on ait

$$|C+x| = |C| \quad \text{et} \quad C+x \notin \Gamma \quad \text{pour un } x \in X. \quad (5.43)$$

Il est certain que $C \notin \Gamma(n)$. En effet si l'on avait $C \in \Gamma(n)$ et $|C+x| = |C|$, on aurait $C+x \in \Gamma(n)$ en vertu du paragraphe 5.3.8, donc $C+x \in \Gamma$ par (5.38). La condition (5.43) est donc un premier critère d'élimination. Deuxièmement, supposons que l'on ait

$$0 < |C+x| < |C| \quad \text{et} \quad C+x \notin \Omega(n-1) \quad \text{pour un } x \in X. \quad (5.44)$$

Il est certain encore que $C \notin \Gamma(n)$, en vertu du paragraphe 5.3.8. La condition (5.44) est donc un second critère d'élimination. On verra que ces deux critères suffisent pour obtenir $\Gamma(n)$.

Désignons par $\mathbf{Sp}(\Gamma)$ l'ensemble des $C \in \Gamma$ qui peuvent être supprimés (d'où le symbole \mathbf{Sp}) *parce qu'ils satisfont l'un ou l'autre des critères d'élimination* (5.43), (5.44). On se propose de déterminer $\Gamma(n)$ par l'algorithme suivant:

$$\text{Tant que } \mathbf{Sp}(\Gamma) \neq \emptyset \text{ remplacer } \Gamma \text{ par } \Gamma - \mathbf{Sp}(\Gamma).$$

Cet algorithme est représenté dans la figure 5.19.

Il reste à montrer que cet algorithme est correct, c'est-à-dire premièrement qu'il se termine (la boucle n'est pas parcourue indéfiniment), et secondement qu'à la sortie de la boucle, l'ensemble Γ est bien l'ensemble cherché $\Gamma(n)$. La propriété de terminaison est évidente, car Γ est un ensemble fini, et il diminue à chaque parcours de la boucle. La seconde propriété résulte de la proposition suivante.

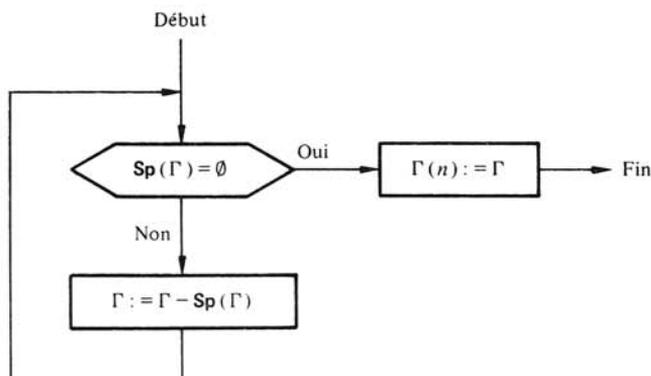


Fig. 5.19

5.3.11 Proposition

Si Γ vérifie les conditions (5.37), (5.38), et (5.40), alors l'ensemble $\Gamma' = \Gamma - \mathbf{Sp}(\Gamma)$ vérifie également ces conditions.

Si Γ vérifie les conditions (5.37), (5.38), (5.40), et si $\mathbf{Sp}(\Gamma) = \emptyset$, alors $\Gamma = \Gamma(n)$.

5.3.12 Démonstration

La première assertion est évidente : Γ' vérifie (5.37), (5.40) parce que $\Gamma' \subset \Gamma$, et vérifie (5.38) par définition de $\mathbf{Sp}(\Gamma)$.

Supposons que Γ vérifie les conditions mentionnées, et que $\mathbf{Sp}(\Gamma) = \emptyset$. La seconde hypothèse signifie qu'aucun élément $C \in \Gamma$ ne vérifie (5.43) ou (5.44), donc que quels que soient $C \in \Gamma$ et $x \in X$,

$$\left. \begin{array}{l} |C + x| = |C| \Rightarrow C + x \in \Gamma \\ 0 < |C + x| < |C| \Rightarrow C + x \in \Omega(n-1). \end{array} \right\} \quad (5.45)$$

Si $\Gamma = \emptyset$, alors $\Gamma = \Gamma(n)$ par (5.38). Supposons $\Gamma \neq \emptyset$, et soient C_1, \dots, C_m les éléments de Γ . Soient C_{m+1}, \dots, C_{m+r} les éléments de $\Omega(n-1)$. On peut voir que $(C_1, \dots, C_m, C_{m+1}, \dots, C_{m+r})$ est un recouvrement compatible avec R . La condition de couverture (5.31) est vérifiée parce que $\Omega(n-1)$ contient $\Omega(1)$ et par (5.35). La condition de fermeture (5.32) est vérifiée pour C_1, \dots, C_m en vertu de (5.45), et pour C_{m+1}, \dots, C_{m+r} en vertu du paragraphe 5.3.8. Enfin la condition (5.33) est vérifiée en vertu de (5.40) et par définition de $\Omega(n-1)$. Ceci prouve que C_1, \dots, C_m sont des classes de compatibilité (§ 5.3.3), donc que $\Gamma \subset \Gamma(n)$. Il vient donc $\Gamma = \Gamma(n)$ par (5.38).

5.3.13 Exemple

Considérons la machine $R[X, Y, Z]$ définie par le tableau 5.20, avec $X = Z = \{0, 1\}$ et $Y = \{a, \dots, h\}$. On veut déterminer les classes de compatibilité à deux éléments en appliquant l'algorithme du paragraphe 5.3.10 pour $n = 2$. Pour cette valeur de n , on peut utiliser une représentation tabulaire particulière. Un sous-ensemble $\{p, q\}$ ($p \neq q$) de Y est appelé une *paire d'états* ou simplement une *paire*. On forme un tableau triangulaire (tab. 5.21). Chaque paire $\{p, q\}$ est représentée par une case du

tableau, à savoir l'unique case de coordonnées (p, q) ou (q, p) . Nous l'appellerons brièvement la case $\{p, q\}$. Ainsi l'ensemble $\mathbf{P}_2(Y)$ est l'ensemble des cases du tableau. On note un sous-ensemble $\Gamma \subset \mathbf{P}_2(Y)$ en biffant les cases $\{p, q\} \notin \Gamma$.

Pour $n = 2$, l'ensemble de départ canonique Γ défini au paragraphe 5.3.10, est simplement l'ensemble des paires $\{p, q\}$ telles que $(p|x) \cap (q|x) \neq \emptyset$ (condition (5.40)), car la condition (5.42) est vérifiée pour toute paire $C = \{p, q\}$. Cet ensemble de départ est l'ensemble des cases non biffées du tableau 5.21. Par exemple on a biffé la case $\{a, c\}$ car $(a|0) \cap (c|0) = \emptyset$ (tab. 5.20). Avant d'appliquer l'algorithme, on doit donc biffer toutes les cases $C = \{p, q\}$ qui ne vérifient pas la condition (5.40).

	0	1
a	b ; 1	h ; 1
b	a ; 0	g ; 0
c	- ; 0	f ; -
d	d ; 0	- ; 1
e	c ; 0	d ; -
f	a ; -	c ; 0
g	- ; -	b ; -
h	g ; 0	e ; -

Tableau 5.20

b							
c		fg					
d							
e		ac	df	cd			
f		cg			ac	cd	
g	bh		bf		bd	bc	
h		ag	ef	dg	cg	ag	
		eg		de	ce	be	
	a	b	c	d	e	f	g

Tableau 5.21

$$\mathbf{Sp}(\Gamma) = \{be, ce, ef, eg\}$$

Pour $n = 2$, le critère d'élimination (5.44) n'est jamais vérifié, car si $|C| = 2$ et si $0 < |C+x| < |C|$, alors $|C+x| = 1$, donc $C+x \in \Omega(n-1)$ en vertu de (5.35). Le critère d'élimination (5.43) suffit donc à déterminer l'ensemble $\mathbf{Sp}(\Gamma)$. Pour appliquer ce critère, on convient de dire qu'une paire $\{p', q'\}$ est *impliquée* par la paire $\{p, q\}$, ou que $\{p, q\}$ *implique* $\{p', q'\}$, si l'on a $\{p', q'\} \neq \{p, q\}$ et $\{p', q'\} = \{p, q\} + x$ pour un $x \in X$. Par exemple $\{b, c\}$ implique $\{f, g\}$, car $\{b, c\} + 1 = \{f, g\}$ et $\{f, g\} \neq \{b, c\}$. Dans chaque case $\{p, q\}$ du tableau triangulaire 5.21, on note l'ensemble des paires impliquées par $\{p, q\}$. Par exemple $\{f, g\}$ est la seule paire impliquée par $\{b, c\}$. Elle est notée *fg* dans la case $\{b, c\}$. La paire $\{b, e\}$ implique $\{a, c\} = \{b, e\} + 0$ et $\{d, g\} = \{b, e\} + 1$. Ces paires sont notées *ac* et *dg* dans la case $\{b, e\}$. La paire $\{b, g\}$ n'implique aucune paire car $\{b, g\} + 0 = \{a\}$ et $\{b, g\} + 1 = \{b, g\}$. La case $\{b, g\}$ reste vide. Le tableau 5.21 est souvent appelé *table des implications* de la machine.

Le critère d'élimination (5.43) pour une paire $C = \{p, q\} \in \Gamma$ prend la forme suivante : $\{p, q\}$ implique une paire $\{p', q'\} \notin \Gamma$, ou encore **la case $\{p, q\}$ porte l'inscription d'une paire $\{p', q'\}$ dont la case est une case biffée.**

La table des implications permet ainsi une détermination rapide de l'ensemble $\mathbf{Sp}(\Gamma)$. Ainsi pour l'ensemble de départ Γ (tab. 5.21), on a (en écrivant *pq* pour $\{p, q\}$):

$$\mathbf{Sp}(\Gamma) = \{be, ce, ef, eg\} \neq \emptyset.$$

Par exemple $\{b, e\}$ implique $\{a, c\}$ dont la case est biffée, donc $\{b, e\}$ appartient à $\mathbf{Sp}(\Gamma)$.

L'opération $\Gamma := \Gamma - \mathbf{Sp}(\Gamma)$ (fig. 5.19) consiste à biffer les cases $\{p, q\} \in \mathbf{Sp}(\Gamma)$, et le tableau 5.21 devient le tableau 5.22.

b							
c		fg					
d							
e		ag dg	df	cd			
f		cg			ag cd		
g	bh		bf		bd	bc	
h		ag eg	ef	dg	cg de	ag ce	be
	a	b	c	d	e	f	g

Tableau 5.22

$$\mathbf{Sp}(\Gamma) = \{bh, ch, fh, gh\}$$

Le nouvel ensemble Γ est l'ensemble des cases non biffées du tableau 5.22, et le nouvel ensemble $\mathbf{Sp}(\Gamma)$ est $\{bh, ch, fh, gh\}$ noté en légende du tableau. L'itération du processus est représentée par le tableau 5.23, puis le tableau 5.24 où l'on a $\mathbf{Sp}(\Gamma) = \emptyset$. Aucune case de ce dernier tableau ne vérifie le critère d'élimination. Le tableau 5.24 représente donc l'ensemble des classes de compatibilité à deux éléments

$$\Gamma(2) = \{bc, bf, bg, cd, cf, cg, de, dg, dh, eh, fg\}. \tag{5.46}$$

b							
c		fg					
d							
e		ag dg	df	cd			
f		cg			ag cd		
g	bh		bf		bd	bc	
h		ag eg	ef	dg	cg de	ag ce	be
	a	b	c	d	e	f	g

Tableau 5.23

$$\mathbf{Sp}(\Gamma) = \{ag\}$$

b							
c		fg					
d							
e		ag dg	df	cd			
f		cg			ag cd		
g	bh		bf		bd	bc	
h		ag eg	ef	dg	cg de	ag ce	be
	a	b	c	d	e	f	g

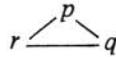
Tableau 5.24

$$\mathbf{Sp}(\Gamma) = \emptyset.$$

Cherchons maintenant les classes de compatibilité à trois éléments. L'ensemble de départ canonique $\Gamma \subset \mathbf{P}_3(Y)$ vérifie entre autres la condition (5.42). Pour $n = 3$, cette condition peut s'écrire

$$\{p, q, r\} \in \Gamma \Rightarrow \begin{cases} \{p, q\} \in \Gamma(2) \\ \{p, r\} \in \Gamma(2) \\ \{q, r\} \in \Gamma(2). \end{cases} \tag{5.47}$$

On commence donc par déterminer tous les ensembles $\{p, q, r\} \in \mathbf{P}_3(Y)$ tels que $\{p, q\}$, $\{p, r\}$, $\{q, r\}$ appartiennent à $\Gamma(2)$. A cet effet, on peut se servir de la figure 5.25 que l'on appelle *diagramme de compatibilité*, dans lequel chaque paire $\{p, q\} \in \Gamma(2)$, ou *paire d'états compatibles*, est notée par une liaison $p - q$. Ce diagramme n'est qu'une autre notation de (5.46). Les ensembles $\{p, q, r\}$ cherchés apparaissent dans le diagramme comme des triangles :



On les note pqr , et ce sont :

$$bcg, bcf, bgf, cgf, cdg, deh. \quad (5.48)$$

L'ensemble de départ Γ sera formé de tous ceux des ensembles C énumérés dans (5.48), qui vérifient (5.40). On peut voir qu'en fait tous les ensembles de cette liste vérifient (5.40). Le contrôle s'effectue avec la table de transition (tab. 5.20). Par exemple

$$b|0 \cap c|0 \cap g|0 = \{0\}$$

$$b|1 \cap c|1 \cap g|1 = \{0\}.$$

L'ensemble de départ Γ est donc l'ensemble (5.48). On peut vérifier que $\mathbf{Sp} \Gamma = \emptyset$, c'est-à-dire qu'aucun des ensembles $C \in \Gamma$ ne vérifie le critère d'élimination (5.43) ou (5.44). Par exemple

$$\{b, c, g\} + 0 = \{a\} \in \Omega(n-1) \quad (\text{ici } n=3)$$

$$\{b, c, g\} + 1 = \{b, f, g\} \in \Gamma.$$

Ainsi $\Gamma(3)$ est l'ensemble (5.48). Nous ne poursuivons pas l'application de la méthode générale (§ 5.3.10) à cet exemple.

En effet, on démontrera plus loin qu'en vertu d'un caractère particulier de la composante combinatoire de \mathcal{R} , cette machine jouit de la *propriété* suivante : pour qu'un sous-ensemble non vide C de Y soit une classe de compatibilité, il faut et il suffit que chaque sous-ensemble $\{p, q\}$ de C soit une classe de compatibilité.

En vertu de cette propriété, les classes de compatibilité à n éléments ($n \geq 2$) apparaissent dans le diagramme de compatibilité (fig. 5.25) comme n sommets tous reliés deux-à-deux. On trouve ainsi une classe à 4 éléments et une seule, à savoir $\{b, c, f, g\}$. Il n'y a pas de classe à n éléments pour $n > 4$.

Les paragraphes qui suivent ont pour but de démontrer la propriété particulière énoncée ci-dessus.

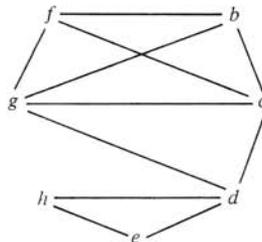


Fig. 5.25

5.3.14 Composantes combinatoires de type standard

Dans tous les exemples de machine $R[X, Y, Z]$ considérés dans ce chapitre, l'alphabet Z était l'alphabet $B = \{0, 1\}$ (tab. 5.20), ou l'alphabet $B_2 = B \times B$ (tab. 5.10, 5.18). De plus, pour chaque couple d'éléments $p \in Y, x \in X$, l'ensemble $p|x$ a pu être noté par un mot sur l'alphabet $\{0, 1, -\}$ (tiret).

Dans le cas où $Z = B$, tout sous-ensemble non vide de Z peut être noté de cette façon: $\{0\}, \{1\}, \{0, 1\}$ sont notés respectivement $0, 1, -$. Dans le cas où $Z = B_2$, on ne peut pas noter de cette façon n'importe quel sous-ensemble de Z . En effet Z possède quatre éléments $0 \times 0, 0 \times 1, 1 \times 0, 1 \times 1$, notés généralement $00, 01, 10, 11$ dans les tables. Les sous-ensembles de Z que l'on peut noter avec le tiret sont au nombre de neuf: $00, 01, 10, 11, -0, -1, 0-, 1-, --$. Or Z possède quinze sous-ensembles non vides ($2^4 - 1$). Par exemple, les sous-ensembles $\{01, 10\}, \{01, 10, 11\}$ ne sont pas représentables avec la notation du tiret.

Plus généralement, supposons que Z soit le produit cartésien $Z_1 \times \dots \times Z_n$ de n alphabets (distincts ou non). Nous appelons *sous-ensemble de type standard* de Z tout sous-ensemble $Q \subset Z$ qui peut être noté par un mot $\alpha_1 \alpha_2 \dots \alpha_n$ où pour $i = 1, \dots, n$, α_i est soit un élément de Z_i , soit le tiret $-$. On dira alors que la *composante combinatoire* d'une machine $R[X, Y, Z]$ est de *type standard* si pour chaque couple $p \in Y, x \in X$, l'ensemble $p|x$ est un sous-ensemble de type standard de Z .

Ainsi dans tous les exemples considérés jusqu'ici, la composante combinatoire de R était de type standard. Cependant on n'a jamais fait usage de cette propriété dans la théorie. La méthode générale de détermination des classes de compatibilité (§ 5.3.10) vaut pour une composante combinatoire de type quelconque.

On montre dans les paragraphes suivants que la propriété particulière énoncée à la fin du paragraphe précédent découle précisément du caractère standard des ensembles $p|x$ de la machine considérée. C'est ce qu'exprime la proposition suivante.

5.3.15 Proposition

Soit R une machine à composante combinatoire de type standard. Pour qu'un ensemble non vide $C \subset Y$ soit une classe de compatibilité de R , il faut et il suffit que chaque sous-ensemble à deux éléments $\{p, q\}$ de C soit une classe de compatibilité.

La démonstration se trouve au paragraphe 5.3.23. Les paragraphes intermédiaires en sont le préalable.

5.3.16 Définitions

On appelle *sous-ensembles triviaux* d'un ensemble Z , l'ensemble Z lui-même et tous les sous-ensembles à un élément $\{z\}$ de Z . On supposera toujours dans la suite que Z, Z_1, \dots, Z_p sont des ensembles non vides. Les sous-ensembles triviaux de ces ensembles ne sont donc jamais vides.

Supposons que $Z = Z_1 \times \dots \times Z_p$. On dit qu'un ensemble $Q \subset Z$ est un *sous-ensemble de type standard* de Z , si Q est de la forme

$$Q = Q_1 \times \dots \times Q_p$$

où pour $k = 1, \dots, p$, Q_k est un sous-ensemble trivial de Z_k .

5.3.17 Proposition

Soient Q_1, \dots, Q_m des sous-ensembles triviaux de Z . Si l'on a $Q_i \cap Q_j \neq \emptyset$ quels que soient $i, j = 1, \dots, m$, alors $Q_1 \cap \dots \cap Q_m \neq \emptyset$.

5.3.18 Démonstration

La relation $Q_i \cap Q_j \neq \emptyset$ implique $Q_i \cap Q_j = Q_i$ ou $Q_i \cap Q_j = Q_j$ par définition des sous-ensembles triviaux. La proposition est donc vraie pour $m = 2$ (elle est triviale pour $m = 1$). Pour $m > 2$, on montre par récurrence sur k ($2 \leq k \leq m$) que l'ensemble $Q_1 \cap \dots \cap Q_k$ est égal à l'un des ensembles Q_i ($i \leq k$). D'où la proposition.

5.3.19 Proposition

Soient $Z = Z_1 \times \dots \times Z_p$, et Q_1, \dots, Q_m des sous-ensembles de type standard de Z . Si l'on a $Q_i \cap Q_j \neq \emptyset$ quels que soient $i, j = 1, \dots, m$, alors $Q_1 \cap \dots \cap Q_m \neq \emptyset$.

5.3.20 Démonstration

Posons pour $i, j = 1, \dots, m$

$$Q_i = Q_{i1} \times \dots \times Q_{ip} \quad (Q_{ik} \text{ trivial } \subset Z_k)$$

$$Q_j = Q_{j1} \times \dots \times Q_{jp} \quad (Q_{jk} \text{ trivial } \subset Z_k)$$

En vertu de (1.37), on a

$$Q_i \cap Q_j = (Q_{i1} \cap Q_{j1}) \times \dots \times (Q_{ip} \cap Q_{jp}). \quad (5.49)$$

L'ensemble (5.49) étant supposé non vide quels que soient $i, j = 1, \dots, m$, les facteurs de ce produit cartésien sont non vides (1.32). On a donc

$$Q_{ik} \cap Q_{jk} \neq \emptyset \quad (i, j = 1, \dots, m) \quad (k = 1, \dots, p). \quad (5.50)$$

L'ensemble $Q_1 \cap \dots \cap Q_m$ s'écrit selon (1.37)

$$(Q_{11} \cap \dots \cap Q_{m1}) \times \dots \times (Q_{1p} \cap \dots \cap Q_{mp}). \quad (5.51)$$

Un facteur quelconque $(Q_{1k} \cap \dots \cap Q_{mk})$ du produit cartésien (5.51) est non vide en vertu de (5.50) et du paragraphe 5.3.17, puisque Q_{1k}, \dots, Q_{mk} sont des sous-ensembles triviaux de Z_k . Il s'ensuit que l'ensemble (5.51) n'est pas vide.

5.3.21 Remarque

La définition des sous-ensembles triviaux et sous-ensembles de type standard (§ 5.3.16) permet de préciser la notion de machine $R[X, Y, Z]$ à composante combinatoire de type standard (§ 5.3.14). On peut dire que la composante combinatoire est de type standard dans les deux cas suivants :

- quels que soient $p \in Y$ et $x \in X$, l'ensemble $p|x$ est un sous-ensemble trivial de Z ;
- Z est un produit cartésien $Z_1 \times \dots \times Z_n$, et quels que soient $p \in Y$ et $x \in X$, l'ensemble $p|x$ est un sous-ensemble de type standard de Z .

5.3.22 Proposition

Supposons que la composante combinatoire de R soit de type standard, et soient $C \subset Y$, $x \in X$. Si l'on a

$$(p|x) \cap (q|x) \neq \emptyset \quad \text{quels que soient } p, q \in C \quad (5.52)$$

alors

$$\bigcap_{p \in C} (p|x) \neq \emptyset. \quad (5.53)$$

La proposition découle immédiatement des paragraphes 5.3.17 et 5.3.19.

5.3.23 Démonstration

On démontre ici la proposition du paragraphe 5.3.15, en procédant par récurrence sur le nombre d'éléments de C . La proposition est triviale pour $|C| \leq 2$. Supposons qu'elle soit vraie pour $|C| < n$ ($n > 2$). Nous appellerons ceci l'*hypothèse H*. Soit Γ l'ensemble des $C \in \mathbf{P}_n(Y)$ qui possèdent la *propriété P* suivante : chaque sous-ensemble à deux éléments $\{p, q\} \subset C$ est une classe de compatibilité. Il est clair que si $C \in \Gamma(n)$, alors C possède la propriété **P** (§ 5.3.6). Donc $\Gamma(n) \subset \Gamma$. D'autre part si C possède la propriété **P**, alors C vérifie (5.52), donc (5.53). Par suite Γ vérifie les conditions (5.37), (5.38), et (5.40). On veut montrer que $\Gamma = \Gamma(n)$, et pour cela il suffit de prouver que $\mathbf{Sp}(\Gamma) = \emptyset$ (§ 5.3.11).

Remarquons d'abord que si C jouit de la propriété **P**, alors $C + x$ jouit de la même propriété, ceci quel que soit $x \in X$. En effet soit $\{p', q'\} \subset C + x$. On a $\{p', q'\} = \{p, q\} + x$ pour un sous-ensemble $\{p, q\} \subset C$. Donc $\{p', q'\}$ est une classe de compatibilité (§ 5.3.8).

En vertu de l'alinéa qui précède, si $C \in \Gamma$ et $|C + x| = |C|$, alors $C + x \in \Gamma$. Si $0 < |C + x| < |C|$, alors $C + x \in \Omega(n-1)$ en vertu de l'hypothèse H. Ainsi aucun ensemble $C \in \Gamma$ ne satisfait le critère d'élimination (5.43) ou (5.44). Donc $\mathbf{Sp}(\Gamma) = \emptyset$.

5.3.24 Exercice

Pour chacune des machines R données dans les tableaux 5.10, 5.15, 5.16, 5.18, déterminer toutes les classes de compatibilité de R , en suivant la méthode du paragraphe 5.3.13.

5.3.25 Exemple

Au paragraphe 5.1.9, nous avons affirmé que si R' est une réduction de R , et si R' est réduite (§ 5.1.8) c'est-à-dire irréductible, R' n'est pas nécessairement une réduction minimale de R . L'exemple qui suit, emprunté à Unger [21], prouve cette assertion. La machine R est donnée par la table 5.26, avec $X = \{a, b, c\}$, $Y = \{1, \dots, 8\}$, $Z = \{0, 1\}$. On considère le recouvrement $\sigma = (C_1, C_2, C_3, C_4)$ défini par

$$C_1 = \{1, 2, 3, 6\}, \quad C_2 = \{4, 5\}, \quad C_3 = \{7\}, \quad C_4 = \{8\}$$

et l'on vérifie immédiatement que ce recouvrement est compatible avec R . La machine quotient de R par σ est la machine R' donnée dans la table 5.27, avec $Y' = \{1, 2, 3, 4\}$.

La machine R' est une réduction de R par construction. Sa composante séquentielle est de type standard. Si l'on cherche les classes de compatibilité de R' (selon le

	<i>a</i>	<i>b</i>	<i>c</i>
1	1 ; 0	2 ; —	5 ; 0
2	2 ; 0	3 ; —	4 ; 0
3	2 ; 0	1 ; —	4 ; 0
4	— ; —	2 ; —	7 ; 1
5	— ; —	1 ; —	7 ; 1
6	1 ; —	2 ; —	5 ; 0
7	1 ; 1	2 ; 1	8 ; 0
8	6 ; 0	3 ; 0	6 ; 1

Tableau 5.26

	<i>a</i>	<i>b</i>	<i>c</i>
1	1 ; 0	1 ; —	2 ; 0
2	— ; —	1 ; —	3 ; 1
3	1 ; 1	1 ; 1	4 ; 0
4	1 ; 0	1 ; 0	1 ; 1

Tableau 5.27

procédé exposé dans cette section), on s'aperçoit que le seul recouvrement de Y' compatible avec R' et sans classe vide est le recouvrement trivial (§ 5.3.4). La vérification est laissée au lecteur comme exercice.

La composante séquentielle de R' étant de type standard, on peut affirmer en vertu des paragraphes 5.2.31 et 5.2.33 que R' est réduite. Or le recouvrement suivant de Y est aussi compatible avec R :

$$K_1 = \{1, 2, 3\}, \quad K_2 = \{4, 5, 8\}, \quad K_3 = \{6, 7\}.$$

La machine quotient de R par (K_1, K_2, K_3) est la machine R'' du tableau 5.28, avec $Y'' = \{1, 2, 3\}$. C'est aussi une réduction de R , par construction. Elle montre que R' , bien que réduite, n'est pas une réduction minimale de R . Comme R' est réduite, R'' n'est pas une réduction de R' .

	<i>a</i>	<i>b</i>	<i>c</i>
1	1 ; 0	1 ; —	2 ; 0
2	3 ; 0	1 ; 0	3 ; 1
3	1 ; 1	1 ; 1	2 ; 0

Tableau 5.28

On vérifie que R'' est réduite, tout comme R' . Cela ne signifie pas non plus que R'' soit une réduction minimale de R . On peut cependant le prouver. Considérons le sous-ensemble $S = \{1, 7, 8\}$ de Y . On voit dans le tableau 5.26 que

$$1 \mid a \cap 7 \mid a = \emptyset, \quad 1 \mid c \cap 8 \mid c = \emptyset, \quad 7 \mid a \cap 8 \mid a = \emptyset.$$

On en déduit qu'aucune classe de compatibilité de R ne peut contenir deux éléments de S . Par suite tout recouvrement de Y compatible avec R possède au moins trois classes distinctes. Ceci prouve que R'' est bien une réduction minimale de R . Cet argument est généralisé au paragraphe suivant.

5.3.26 Classes d'incompatibilité

On appelle classe d'incompatibilité d'une machine $R[X, Y, Z]$ tout sous-ensemble S de Y ayant la propriété suivante : aucun sous-ensemble $\{p, q\}$ ($p \neq q$) de S n'est une classe de compatibilité.

Il est clair que si m est le nombre d'éléments de la plus grande classe d'incompa-

tibilité de R , aucun recouvrement compatible avec R ne possède moins de m classes distinctes. Mais il n'existe pas nécessairement de recouvrement compatible ayant m classes. L'exercice suivant le montre.

5.3.27 Exercice

A partir du tableau 5.24, construire une *diagramme d'incompatibilité* semblable à celui de la figure 5.25, mais dans lequel les liaisons $p - q$ représentent toutes les paires d'états non compatibles. Vérifier au moyen de ce diagramme que le nombre maximum d'éléments pour une classe d'incompatibilité de R (tab. 5.20) est $m = 3$.

Connaissant toutes les classes de compatibilité de R (§ 5.3.13), vérifier qu'il y a exactement une façon de recouvrir Y avec trois classes de compatibilité seulement, et que ce recouvrement n'est pas compatible avec R .

5.4. CONSTRUCTION DES RECOUUREMENTS

5.4.1 Introduction

Cette section est consacrée à la seconde des opérations mentionnées au paragraphe 5.3.5, à savoir au problème suivant.

On suppose qu'on connaît toutes les classes de compatibilité d'une machine $R[X, Y, Z]$, et l'on veut choisir n classes C_1, \dots, C_n (n minimum) satisfaisant la condition de couverture (5.31) et la condition de fermeture (5.32).

Comme dans la section précédente, on suppose toujours que la composante séquentielle de R est de type standard. Par contre la composante combinatoire peut être quelconque.

Le terme de "classe" signifie toujours "classe de compatibilité".

5.4.2 Définition

Soient C et K deux classes. On dit que K est *impliquée* par C , si K vérifie les trois conditions suivantes:

$$K = C + x \quad \text{pour un } x \in X \quad (5.54)$$

$$K \not\subset C \quad (5.55)$$

$$|K| > 1. \quad (5.56)$$

On désigne par $\mathbf{Imp}(C)$ l'ensemble des classes K impliquées par C .

5.4.3 Exemple

Soit $R[X, Y, Z]$ la machine du tableau 5.29. La table des implications (tab. 5.30) dans sa forme finale, puis le diagramme de compatibilité (fig. 5.31), nous livrent toutes les classes de compatibilité de R . Celles-ci sont notées C_1, \dots, C_{12} dans la première colonne du tableau 5.32.

Dans la deuxième colonne du tableau 5.32, en regard de chaque classe C_i , figure l'ensemble $\mathbf{Imp}(C_i)$. Il est clair que $\mathbf{Imp}(C_i) = \emptyset$ lorsque $|C_i| = 1$, en vertu de (5.56). L'ensemble $\mathbf{Imp}(C_6)$ comporte les classes $\{2, 3\} = C_6 + a$ et $\{4, 5\} = C_6 + b$ déterminées par le tableau 5.29.

	<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>
1	2 ; -	5 ; 0	5 ; 0	2 ; -
2	3 ; 0	4 ; -	- ; -	- ; -
3	- ; -	5 ; -	1 ; 0	2 ; 0
4	1 ; -	2 ; 1	- ; -	- ; -
5	2 ; 1	- ; -	- ; -	4 ; 1

Tableau 5.29

2	23 45			
3	15	45		
4		13	25	
5	24			12
	1	2	3	4

Tableau 5.30

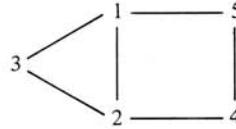


Fig. 5.31

Classes C	Imp (C)	Fermetures minimales de {C}
C ₁ = {1}		∅
C ₂ = {2}		∅
C ₃ = {3}		∅
C ₄ = {4}		∅
C ₅ = {5}		∅
C ₆ = {1,2}	{2,3}, {4,5}	{C ₉ , C ₁₁ }, {C ₁₂ , C ₁₁ }
C ₇ = {1,3}	{1,5}	{C ₈ }
C ₈ = {1,5}	{2,4}	{C ₁₀ }
C ₉ = {2,3}	{4,5}	{C ₁₁ }
C ₁₀ = {2,4}	{1,3}	{C ₇ }, {C ₁₂ }
C ₁₁ = {4,5}	{1,2}	{C ₆ }, {C ₁₂ }
C ₁₂ = {1,2,3}	{4,5}, {1,5}	{C ₁₁ , C ₈ }

Tableau 5.32

5.4.4 Définition

Soient Γ et Ω deux ensembles de classes. On dit que Ω est une *fermeture* de Γ si les deux conditions suivantes sont vérifiées :

$$\Gamma \cap \Omega = \emptyset \tag{5.57}$$

Pour toute classe $C \in \Gamma$ et pour toute classe $K \in \text{Imp}(C)$
 il existe une classe $C' \in \Gamma \cup \Omega$ telle que $K \subset C'$. (5.58)

On dit que Ω est une *fermeture minimale* de Γ si aucun sous-ensemble propre de Ω n'est une fermeture de Γ . Par sous-ensemble propre, on entend un sous-ensemble $\Omega' \subset \Omega$ tel que $\Omega' \neq \Omega$.

5.4.5 Exemple

La troisième colonne du tableau 5.32 contient les fermetures minimales des ensembles de classes $\Gamma_i = \{C_i\}$ ($i = 1, \dots, 12$). Il est clair que si C est une classe telle que $\text{Imp}(C) = \emptyset$, alors l'ensemble $\Omega = \emptyset$ est l'unique fermeture minimale de l'ensemble $\Gamma = \{C\}$. Considérons l'ensemble $\Gamma_6 = \{C_6\}$. Si Ω est une fermeture de Γ_6 , on doit avoir

$$(C_9 \in \Omega \text{ ou } C_{12} \in \Omega) \text{ et } C_{11} \in \Omega$$

car C_9 et C_{12} sont les seules classes contenant la classe $\{2, 3\}$ impliquée par C_6 , et C_{11} est la seule classe contenant la classe $\{4, 5\}$ impliquée par C_6 . D'où les fermetures minimales $\{C_9, C_{11}\}$, $\{C_{12}, C_{11}\}$.

5.4.6 Proposition

Soient $\Gamma = \{C_1, \dots, C_m\}$ un ensemble de classes, et $\Omega_1, \dots, \Omega_m$ des fermetures respectives de $\{C_1\}, \dots, \{C_m\}$. Alors l'ensemble

$$\Omega = (\Omega_1 - \Gamma) \cup \dots \cup (\Omega_m - \Gamma) \tag{5.59}$$

est une fermeture de Γ .

5.4.7 Démonstration

Rappelons d'abord que $\Omega_i - \Gamma$ est l'ensemble des $C \in \Omega_i$ tels que $C \notin \Gamma$. Il est donc clair que l'ensemble Ω défini par (5.59) vérifie (5.57). Montrons qu'il vérifie aussi (5.58). Considérons une classe $C_i \in \Gamma$, et soit $K \in \text{Imp}(C_i)$. Supposons qu'il n'existe pas de classe $C' \in \Gamma$ telle que $K \subset C'$. Alors par définition de Ω_i , il existe une classe $C' \in \Omega_i - \Gamma$ telle que $K \subset C'$.

5.4.8 Proposition

Si Ω est une fermeture de Γ (§ 5.4.4), alors Ω contient une fermeture minimale de Γ .

5.4.9 Démonstration

Supposons que Ω soit une fermeture de Γ . Si Ω n'est pas une fermeture minimale, il existe un sous-ensemble propre $\Omega_1 \subset \Omega$ qui est une fermeture de Γ . Si Ω_1 n'est pas minimal, Ω_1 possède un sous-ensemble propre Ω_2 qui est une fermeture de Γ , et ainsi de suite. Comme Ω est fini on doit nécessairement aboutir à un sous-ensemble Ω_k qui est une fermeture minimale de Γ .

5.4.10 Proposition

Soient Γ un ensemble de classes et Ω une fermeture de Γ . Pour toute classe $C \in \Gamma$, l'ensemble $(\Gamma - \{C\}) \cup \Omega$ est une fermeture de $\{C\}$.

5.4.11 Démonstration

Posons $\Gamma' = \{C\}$ et $\Omega' = (\Gamma - \{C\}) \cup \Omega$. Il est clair que $\Gamma' \cap \Omega' = \emptyset$ en vertu de (5.57). Soit $K \in \text{Imp}(C)$. En vertu de (5.55) et (5.58), il existe une classe $C' \in \Omega'$ telle que $K \subset C'$.

5.4.12 Proposition

Soit $\Gamma = \{C_1, \dots, C_m\}$ un ensemble de classes. Toute fermeture minimale Ω de Γ est de la forme

$$\Omega = (\Omega_1 - \Gamma) \cup \dots \cup (\Omega_m - \Gamma) \tag{5.60}$$

où $\Omega_1, \dots, \Omega_m$ sont des fermetures minimales respectives de $\{C_1\}, \dots, \{C_m\}$.

5.4.13 Démonstration

Supposons que Ω soit une fermeture minimale de Γ . Pour chacune des classes $C_i \in \Gamma$, soit $\Omega'_i = (\Gamma - \{C_i\}) \cup \Omega$. L'ensemble Ω'_i est une fermeture de $\{C_i\}$ (§ 5.4.11), et l'on a $\Omega'_i - \Gamma = \Omega$. L'ensemble Ω'_i contient une fermeture minimale Ω_i de $\{C_i\}$, et l'on a $\Omega_i - \Gamma \subset \Omega$. La réunion des ensembles $\Omega_i - \Gamma$ est une fermeture de Γ (§ 5.4.6). Elle est contenue dans Ω , donc égale à Ω puisque Ω est une fermeture minimale de Γ .

5.4.14 Exemple

Le tableau 5.33 est extrait du tableau 5.32. On désigne par Ω_6 l'une quelconque des fermetures minimales de $\{C_6\}$, et de même pour Ω_{11} . Soit à trouver les fermetures minimales de $\Gamma = \{C_6, C_{11}\}$. On écrit les ensembles de la forme $\Omega_6 - \Gamma, \Omega_{11} - \Gamma$ (tab. 5.33), et l'on sait que toute fermeture minimale de Γ sera l'une des quatre réunions de la forme $(\Omega_6 - \Gamma) \cup (\Omega_{11} - \Gamma)$. Ces quatre réunions sont les ensembles

$$\begin{aligned} \{C_9\} \cup \emptyset &= \{C_9\} \\ \{C_9\} \cup \{C_{12}\} &= \{C_9, C_{12}\} \\ \{C_{12}\} \cup \emptyset &= \{C_{12}\} \\ \{C_{12}\} \cup \{C_{12}\} &= \{C_{12}\}. \end{aligned}$$

On voit que les fermetures minimales de $\Gamma = \{C_6, C_{11}\}$ sont $\{C_9\}$ et $\{C_{12}\}$. En pratique, on peut noter les ensembles $\Omega_i - \Gamma$ en biffant dans Ω_i les classes $C \in \Gamma$ (tab. 5.34). La notation $\{\emptyset\}$ représente alors l'ensemble vide.

Le même procédé, appliqué à $\Gamma = \{C_7, C_8, C_{10}\}$ (tab. 5.35), montre que l'unique fermeture minimale de $\{C_7, C_8, C_{10}\}$ est l'ensemble vide, obtenu par la réunion $\{\emptyset_8\} \cup \{\emptyset_{10}\} \cup \{\emptyset_7\}$.

Classes	Fermetures minimales Ω_i	$\Omega_i - \Gamma$
$\Gamma \begin{cases} C_6 \\ C_{11} \end{cases}$	$\Omega_6 : \{C_9, C_{11}\}, \{C_{12}, C_{11}\}$ $\Omega_{11} : \{C_6\}, \{C_{12}\}$	$\Omega_6 - \Gamma : \{C_9\}, \{C_{12}\}$ $\Omega_{11} - \Gamma : \emptyset, \{C_{12}\}$

Tableau 5.33

Classes	$\Omega_i - \Gamma$
$\Gamma \begin{cases} C_6 \\ C_{11} \end{cases}$	$\{C_9, \emptyset_{11}\}, \{C_{12}, \emptyset_{11}\}$ $\{\emptyset_6\}, \{C_{12}\}$

Tableau 5.34

Classes	$\Omega_i - \Gamma$
$\Gamma \begin{cases} C_7 \\ C_8 \\ C_{10} \end{cases}$	$\{\emptyset_8\}$ $\{\emptyset_{10}\}$ $\{\emptyset_7\}, \{C_{12}\}$

Tableau 5.35

5.4.15 Définition

On dira qu'un ensemble de classes Γ est *fermé* si l'ensemble $\Omega = \emptyset$ est une fermeture de Γ .

En vertu du paragraphe 5.4.4, la phrase " Γ est fermé" signifie que pour toute classe $C \in \Gamma$ et pour toute classe $K \in \text{Imp}(C)$ il existe une classe $C' \in \Gamma$ telle que $K \subset C'$.

5.4.16 Proposition

Soit $\Gamma = \{C_1, \dots, C_n\}$ un ensemble de classes vérifiant la condition de couverture (5.31). Pour que Γ vérifie la condition de fermeture (5.32), il faut et il suffit que Γ soit fermé.

5.4.17 Démonstration

Si Γ vérifie la condition de fermeture (5.32), il est clair que Γ est fermé. Supposons que Γ soit fermé. Considérons une classe $C_i \in \Gamma$ et un $x \in X$. Si $|C_i + x| \leq 1$, il existe une classe $C_j \supset C_i + x$ parce que Γ satisfait la condition de couverture. Si $|C_i + x| > 1$, $C_i + x$ est soit contenue dans C_i , soit une classe impliquée par C_i , donc contenue dans une classe C_j . Par suite Γ satisfait la condition de fermeture (5.32).

5.4.18 Exemple

On a vu dans le tableau 5.35, que l'ensemble $\Gamma = \{C_7, C_8, C_{10}\}$ est fermé. Il vérifie la condition de couverture, car on a $C_7 \cup C_8 \cup C_{10} = \{1, \dots, 5\} = Y$ (tab. 5.32). Donc il vérifie la condition de fermeture.

5.4.19 Proposition

Si Γ est un ensemble de classes fermé, Γ contient une fermeture minimale de chacun de ses sous-ensembles.

En effet si $\Gamma' \subset \Gamma$ et si Γ est fermé, $\Gamma - \Gamma'$ est une fermeture de Γ' , et contient une fermeture minimale de Γ' (§ 5.4.8).

5.4.20 Méthode génératrice de recouvrements

La figure 5.36 représente une méthode de construction d'un ensemble de classes Γ fermé et satisfaisant la condition de couverture. Un tel ensemble sera appelé en bref un recouvrement. On procède en partant de l'ensemble $\Gamma = \emptyset$, et en lui ajoutant des classes jusqu'à ce que les deux conditions soient satisfaites. A chaque stade, on dit qu'un élément $y_i \in Y$ est couvert par Γ s'il y a une classe $C \in \Gamma$ telle que $y_i \in C$. La méthode n'est pas un algorithme au sens strict du terme, car elle comporte des opérations de sélection exprimées par le verbe choisir. Elle engendre différents recouvrements selon les choix effectués, d'où son nom de méthode génératrice. On verra plus loin qu'elle engendre tous les recouvrements irredondants, c'est-à-dire les recouvrements dans lesquels aucune classe ne peut être supprimée sans que l'une ou l'autre des conditions de couverture et fermeture cesse d'être vérifiée. Elle engendre en particulier tous les recouvrements minimaux, c'est-à-dire ayant le nombre minimum de classes.

La méthode suppose que l'on a au préalable fixé un *ordre* y_1, \dots, y_n pour les éléments de Y . Si S est un sous-ensemble non vide de Y , l'élément $y_i \in S$ avec i minimum est appelé le plus petit élément de S .

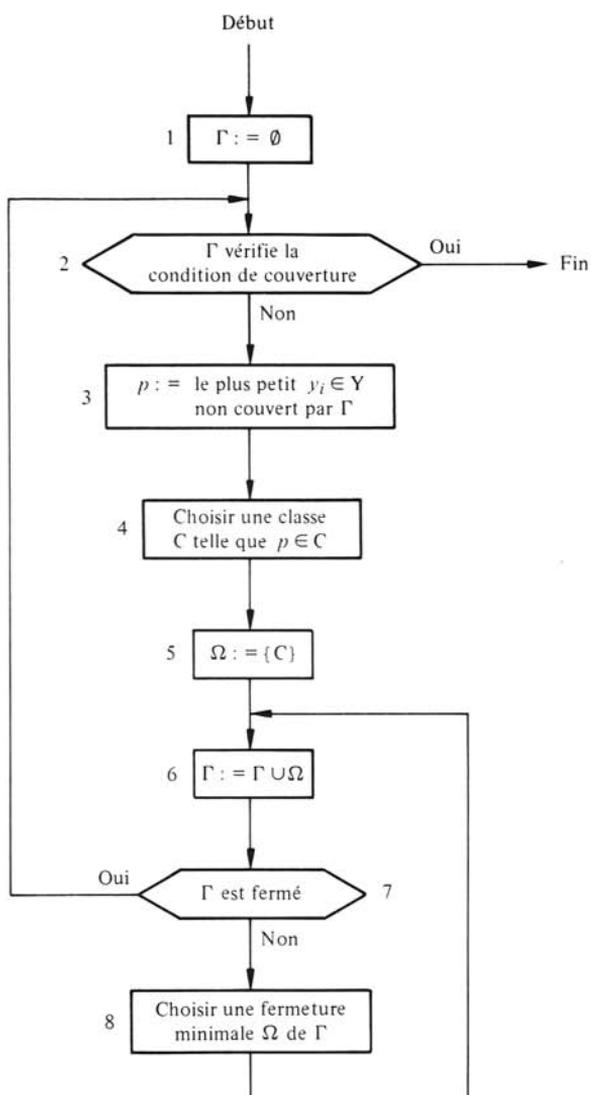


Fig. 5.36

5.4.21 Exemple

Appliquons la méthode de la figure 5.36 à l'exemple du paragraphe 5.4.3 (tab. 5.32). Auparavant, on choisit pour l'ensemble Y l'ordre suivant :

$$(y_1, y_2, y_3, y_4, y_5) = (4, 5, 3, 1, 2). \quad (5.61)$$

La raison de ce choix sera donnée plus loin. Les instructions de la méthode sont numérotées de 1 à 8 (fig. 5.36). Un déroulement possible des opérations est le suivant :

$$1. \Gamma = \emptyset \quad (\Gamma_0)$$

2. Non

$$3. p = 4$$

$$4. C_{10} = \{2, 4\}$$

$$5. \Omega = \{C_{10}\}$$

$$6. \Gamma = \{C_{10}\} \quad (\Gamma_1)$$

7. Non

$$8. \Omega = \{C_{12}\} (C_{12} = \{1, 2, 3\})$$

$$6. \Gamma = \{C_{10}, C_{12}\} \quad (\Gamma_2)$$

7. Non (par la méthode du paragraphe 5.4.14)

$$8. \Omega = \{C_{11}, C_8\} (C_{11} = \{4, 5\}, C_8 = \{1, 5\})$$

$$6. \Gamma = \{C_{10}, C_{12}, C_{11}, C_8\} \quad (\Gamma_3)$$

7. Oui

2. Oui.

Cette exécution de la méthode peut être représentée succinctement par la suite d'ensembles

$$\left. \begin{aligned} \Gamma_0 &= \emptyset \\ \Gamma_1 &= \{C_{10}\} \\ \Gamma_2 &= \{C_{10}, C_{12}\} \\ \Gamma_3 &= \{C_{10}, C_{12}, C_{11}, C_8\}. \end{aligned} \right\} \quad (5.62)$$

5.4.22 Propriétés de la méthode génératrice

La méthode ne peut pas donner lieu à une suite infinie d'opérations, car l'ensemble Γ augmente à chaque exécution de l'instruction $\Gamma := \Gamma \cup \Omega$. Dans le pire des cas on atteint l'ensemble de toutes les classes, qui satisfait toujours les conditions de couverture et fermeture.

Lorsqu'on atteint la fin de la méthode, l'ensemble Γ a passé avec succès les tests 2 et 7. La méthode n'engendre donc que des ensembles Γ vérifiant les conditions de couverture et fermeture (appelés en bref recouvrements).

On dit qu'un recouvrement Γ est *redondant* s'il existe un recouvrement Γ' qui est un sous-ensemble propre de Γ . Sinon, Γ est dit *irredondant*. Nous allons montrer que tout recouvrement irredondant est engendré par la méthode (mais elle engendre aussi des recouvrements redondants).

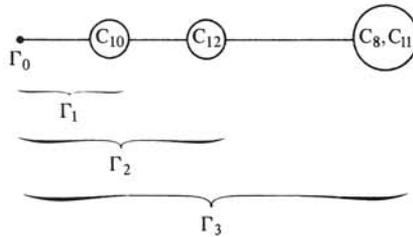
Considérons un déroulement de la méthode, et soient $\Gamma_0 = \emptyset$ et $\Gamma_1, \dots, \Gamma_n$ les ensembles obtenus par les exécutions successives de l'instruction No 6. Soit Γ un recouvrement irredondant. On peut faire en sorte que chacun des ensembles $\Gamma_0, \Gamma_1, \dots, \Gamma_n$ soit contenu dans Γ . En effet c'est possible pour Γ_0 . Supposons que l'on ait obtenu

$\Gamma_k \subset \Gamma$. Si Γ_k est fermé, alors on peut obtenir $\Gamma_{k+1} \subset \Gamma$, car en effectuant l'instruction 4 on peut toujours prendre une classe $C \in \Gamma$ (puisque Γ est un recouvrement). Si Γ_k n'est pas fermé, on peut aussi obtenir $\Gamma_{k+1} \subset \Gamma$, car $\Gamma_k \subset \Gamma$ implique que Γ contient une fermeture minimale de Γ_k (§ 5.4.19). Ainsi par récurrence, on peut obtenir $\Gamma_n \subset \Gamma$. Comme Γ_n et Γ sont deux recouvrements, et Γ est irredondant, on a nécessairement $\Gamma_n = \Gamma$. Ceci prouve que tout recouvrement irredondant est engendré par la méthode.

5.4.23 Arbre des recouvrements

Toutes les suites d'ensembles de classes $\Gamma_0 = \emptyset, \Gamma_1, \dots, \Gamma_n$ construites par exécution de la méthode génératrice peuvent être représentées par un arbre tel que celui de la figure 5.37, relatif à l'exemple du paragraphe 5.4.3.

Chaque branche de l'arbre représente une exécution de la méthode génératrice. Les embranchements correspondent aux divers choix que l'on peut effectuer. Par exemple, la suite (5.62) est représentée par la branche supérieure de l'arbre :



Chaque adjonction d'un nœud à l'extrémité d'une branche en cours de construction représente une exécution de l'instruction No 6 (fig. 5.36).

Les nœuds de l'arbre sont disposés en colonne, de façon que toutes les branches aboutissant dans une même colonne portent le même nombre de classes. Il est ainsi possible d'un coup d'œil, de comparer la taille des divers ensembles Γ engendrés.

L'idée qui donne à cette représentation tout son intérêt, est que l'on peut construire l'arbre *par colonnes*, ce qui revient à effectuer toutes les exécutions possibles de la méthode génératrice *en parallèle*. On obtient ainsi les recouvrements minimaux avant les autres, et l'on peut s'abstenir de poursuivre la construction d'une branche au delà de la colonne n , lorsque la construction d'une autre branche est déjà achevée et se termine dans une colonne $m \leq n$.

Les embranchements dessinés en traits forts correspondent aux divers choix possibles dans une exécution de l'instruction No 4 (fig. 5.36). On a noté à côté de ces embranchements le plus petit élément $y_i \in Y$ selon l'ordre (5.61) qui n'est pas encore couvert à cet endroit de la construction (instruction No 3).

Nous pouvons expliquer maintenant le choix de l'ordre (5.61) par le tableau 5.38, qu'on appelle *tableau de couverture*, et qui indique pour chaque élément $y \in Y$ les classes C_i auxquelles y appartient, ceci par une croix dans la colonne y et la ligne C_i . Le *score* d'un élément y est le nombre de classes auxquelles il appartient. Les éléments y sont alors ordonnés *par score croissant*, et par exemple dans l'ordre 4, 5, 3, 1, 2 (5.61). Le choix de cet ordre a pour vertu de diminuer, par rapport à d'autres ordres, le nombre de choix possibles dans l'exécution de l'instruction No 4, donc la taille des embranchements dont il est question ci-dessus.

	y				
	1	2	3	4	5
C ₁	X				
C ₂		X			
C ₃			X		
C ₄				X	
C ₅					X
C ₆	X	X			
C ₇	X		X		
C ₈	X				X
C ₉		X	X		
C ₁₀		X		X	
C ₁₁				X	X
C ₁₂	X	X	X		
Score	5	5	4	3	3

Tableau 5.38

(5.63), mais il n'est pas minimal. Les branches (3) et (4) représentent deux recouvrements redondants: $\{C_4, C_8, C_{10}, C_7\}$ et $\{C_4, C_{11}, C_6, C_9\}$, dans lesquels on peut supprimer C_4 .

5.4.24 Exercice

Pour chacun des recouvrements minimaux Γ (5.63), renuméroter C_1, C_2, C_3 les classes de Γ et construire la machine quotient de R (tab. 5.29) par (C_1, C_2, C_3) .

5.4.25 Remarque méthodologique

L'arbre des recouvrements prend rapidement une dimension encombrante, lorsque le nombre de classes de la machine augmente. Il vient donc à l'idée d'éliminer des classes a priori, à condition d'avoir un critère d'élimination qui garantisse la possibilité de trouver un recouvrement minimal dans l'ensemble des classes restantes. C'est cette idée que nous allons mettre en œuvre pour terminer cette section.

5.4.26 Définitions

Soient C et C' deux classes de compatibilité. On dit que C est *exclue par* C' si les deux conditions suivantes sont vérifiées:

$$C \subset C' \text{ et } C \neq C' \quad (5.64)$$

$$\text{Imp}(C') \subset \text{Imp}(C). \quad (5.65)$$

Une classe C est dite *première* s'il n'existe pas de classe C' telle que C soit exclue par C' .

Une classe C est dite *maximale* s'il n'existe pas de classe C' vérifiant (5.64). Il est clair que toute classe maximale est première.

5.4.27 Proposition

La relation d'exclusion est transitive : si C est exclue par C' et si C' est exclue par C'' , alors C est exclue par C'' .

La proposition est évidente. Il en découle que toute classe C non première est exclue par une classe première. En effet si C n'est pas première il existe une classe C' qui exclut C , et tant que C' n'est pas première, on peut remplacer C' par une classe C'' excluant C' , donc excluant C par transitivité. L'opération ne peut se répéter indéfiniment, et aboutit nécessairement à une classe première excluant C .

5.4.28 Proposition

Pour tout recouvrement $\Gamma = \{C_1, \dots, C_n\}$ il existe un recouvrement $\Gamma' = \{C'_1, \dots, C'_n\}$ formé uniquement de classes premières.

Il suffit en effet de prendre pour C'_i : la classe C_i elle-même si elle est première, sinon une classe première excluant C_i . Ce choix entraîne

$$C_i \subset C'_i \tag{5.66}$$

$$\text{Imp}(C'_i) \subset \text{Imp}(C_i) \tag{5.67}$$

pour $i = 1, \dots, n$. Comme Γ vérifie la condition de couverture, il en est de même de Γ' , en vertu de (5.66). Si $K \in \text{Imp}(C'_i)$, alors $K \in \text{Imp}(C_i)$, et comme Γ est fermé on a $K \subset C_j$ donc $K \subset C'_j$. Par suite Γ' est fermé.

5.4.29 Exemple

La proposition qui précède montre que l'on peut éliminer a priori toutes les classes non premières, car quel que soit le nombre de classes d'un recouvrement Γ , il existe toujours un recouvrement premier ayant le même nombre de classes que Γ .

On reprend ici pour exemple la machine du tableau 5.20, dont on a déterminé toutes les classes au paragraphe 5.3.13. La liste des classes C_i et les ensembles $\text{Imp}(C_i)$ sont donnés dans le tableau 5.39. Dans la troisième colonne, en regard de chaque classe C_i , on a noté une classe C_j excluant C_i , lorsqu'il en existe une. Ainsi C_4 est exclue par C_{12} , car $C_4 \subset C_{12}$ et $\text{Imp}(C_{12}) = \emptyset \subset \text{Imp}(C_4)$. Les classes premières sont donc

$$C_1, C_5, C_8, C_{12}, C_{15}, C_{16}, C_{17}, C_{18}, C_{23}, C_{25}, C_{26}. \tag{5.68}$$

En ne retenant que les classes premières (5.68), on construit le tableau 5.40, qui contient pour chaque classe première C_i l'ensemble $\text{Imp}(C_i)$, les fermetures minimales premières Ω_i de $\{C_i\}$, et donne le score des éléments $y = a, \dots, h$ par rapport aux classes premières. Comme dans le tableau 5.39, on a fait l'économie de toutes les parenthèses $\{, \}$, en écrivant systématiquement $ABC\dots$ pour $\{A, B, C, \dots\}$.

On choisit pour l'ensemble $Y = \{a, \dots, h\}$ l'ordre suivant, de score croissant

$$(y_1, \dots, y_8) = (b, a, f, c, g, e, h, d). \tag{5.69}$$

La méthode génératrice, pour cet ordre, donne l'arbre de la figure 5.41. Une seule branche s'achève dans la quatrième colonne. Il n'y a donc qu'un recouvrement minimal premier :

$$\Gamma = \{C_{26}, C_1, C_{25}, C_{23}\}. \tag{5.70}$$

Classes C_i	Imp (C_i)	Exclusion
$C_1 = a$		
$C_2 = b$		C_{26}
$C_3 = c$		C_{26}
$C_4 = d$		C_{12}
$C_5 = e$		
$C_6 = f$		C_{26}
$C_7 = g$		C_{26}
$C_8 = h$		
$C_9 = bc$	fg	C_{26}
$C_{10} = bf$	cg	C_{26}
$C_{11} = bg$		C_{26}
$C_{12} = cd$		
$C_{13} = cf$		C_{26}
$C_{14} = cg$	fg	C_{26}
$C_{15} = de$	cd	
$C_{16} = dg$		
$C_{17} = dh$	dg	
$C_{18} = eh$	cg, de	
$C_{19} = fg$	cd	C_{26}
$C_{20} = bcf$	cfg	C_{26}
$C_{21} = bcg$	bjg	C_{26}
$C_{22} = bfg$	bcg	C_{26}
$C_{23} = cdg$	bf	
$C_{24} = cfg$	bcf	C_{26}
$C_{25} = deh$	cdg	
$C_{26} = bcfg$		

Tableau 5.39

C_i	Imp (C_i)	Fermetures minimales	a	b	c	d	e	f	g	h	
$C_1 = a$			X								
$C_5 = e$							X				
$C_8 = h$										X	
$C_{12} = cd$					X	X					
$C_{15} = de$	cd	C_{12}, C_{23}			X	X					
$C_{16} = dg$					X	X			X		
$C_{17} = dh$	dg	C_{16}, C_{23}			X	X				X	
$C_{18} = eh$	cg, de	$C_{23}, C_{15}, C_{23}, C_{25}, C_{26}, C_{15}, C_{26}, C_{25}$					X			X	
$C_{23} = cdg$	bf	C_{26}			X	X			X		
$C_{25} = deh$	cdg	C_{23}			X	X				X	
$C_{26} = bcfg$				X	X			X	X		
			Score	1	1	3	6	4	1	3	4

Tableau 5.40

On constate que les classes de Γ sont toutes les classes *maximales* de la machine. C'est un fait accidentel.

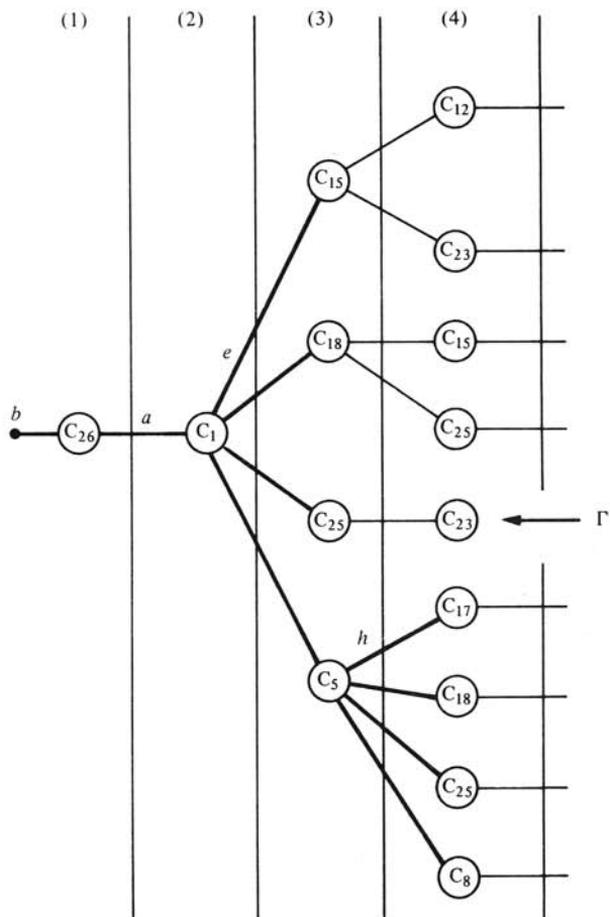


Fig. 5.41

5.4.30 Remarque

Il existe des machines dont *toutes* les classes de compatibilité sont premières. C'est le cas pour l'exemple du paragraphe 5.4.3, comme on peut le vérifier dans le tableau 5.32.

	1	2	3	4	5	6	7
<i>a</i>	<i>a</i> ; 0	— ; —	<i>d</i> ; 0	<i>e</i> ; 1	<i>b</i> ; 0	<i>a</i> ; —	— ; —
<i>b</i>	<i>b</i> ; 0	<i>d</i> ; 1	<i>a</i> ; —	— ; —	<i>a</i> ; —	<i>a</i> ; 1	— ; —
<i>c</i>	<i>b</i> ; 0	<i>d</i> ; 1	<i>a</i> ; 1	— ; —	— ; —	— ; —	<i>g</i> ; 0
<i>d</i>	— ; —	<i>e</i> ; —	— ; —	<i>b</i> ; —	<i>b</i> ; 0	— ; —	<i>a</i> ; —
<i>e</i>	<i>b</i> ; —	<i>e</i> ; —	<i>a</i> ; —	— ; —	<i>b</i> ; —	<i>e</i> ; —	<i>a</i> ; 1
<i>f</i>	<i>b</i> ; 0	<i>c</i> ; —	— ; 1	<i>h</i> ; 1	<i>f</i> ; 1	<i>g</i> ; 0	— ; —
<i>g</i>	— ; —	<i>c</i> ; 1	— ; —	<i>e</i> ; 1	— ; —	<i>g</i> ; 0	<i>f</i> ; 0
<i>h</i>	<i>a</i> ; 1	<i>e</i> ; 0	<i>d</i> ; 1	<i>b</i> ; 0	<i>b</i> ; —	<i>e</i> ; —	<i>a</i> ; 1

Tableau 5.42

5.4.31 Exercice

Déterminer un recouvrement minimal premier pour la machine R du tableau 5.42, et construire la machine quotient.

5.5 CAS PARTICULIERS

5.5.1 Classes de compatibilité maximales

Soit $\Gamma = \{C_1, \dots, C_n\}$ l'ensemble de *toutes* les classes de compatibilité maximales (§ 5.4.26) d'une machine $R[X, Y, Z]$. Il est clair que toute classe de compatibilité K est contenue dans une classe maximale. En particulier toute classe K à un élément $\{y\}$. Donc Γ vérifie la condition de couverture. De plus si $K \in \text{Imp}(C_i)$ ($C_i \in \Gamma$), alors K est contenu dans une classe maximale C_j . Donc Γ est fermé.

Ainsi l'ensemble des classes maximales est toujours un recouvrement. Dans certains cas il est minimum (exemple du paragraphe 5.4.29, Γ (5.70)), dans d'autres cas il ne l'est pas. Il peut même compter plus de classes qu'il n'y a d'éléments $y \in Y$. Supposons par exemple que $|Y| = 6$, et que le diagramme de compatibilité soit de la forme donnée dans la figure 5.43. Les classes maximales sont toutes les classes à deux éléments, et il y en a sept.

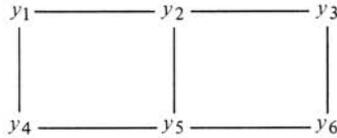


Fig. 5.43

Dans l'exemple du paragraphe 5.4.3, les classes maximales sont au nombre de quatre (fig. 5.31), et les recouvrements minimaux ont trois classes seulement (5.63).

5.5.2 Machines complètement spécifiées

Supposons que R soit complètement spécifiée, c'est-à-dire que

$$|p + x| = 1 \text{ et } |(p|x)| = 1 \quad \text{quels que soient } p \in Y, x \in X. \quad (5.71)$$

On montre dans ce cas que si $\{p, q\}$ et $\{q, r\}$ sont deux classes, alors $\{p, r\}$ est une classe. La démonstration est donnée au paragraphe suivant. En d'autres termes, si $\Gamma(2)$ désigne comme précédemment (sect. 5.3) l'ensemble des classes de compatibilité à deux éléments, la relation $\{p, q\} \in \Gamma(2)$ est transitive. Il en découle que si C et C' sont deux classes *maximales*, on a

$$C = C' \text{ ou } C \cap C' = \emptyset. \quad (5.72)$$

En effet, supposons que $C \cap C' \neq \emptyset$. Alors par transitivité, toute paire $\{p, q\} \subset C \cup C'$ est une classe de compatibilité. On peut en déduire que $C \cup C'$ est une classe de compatibilité, car la composante combinatoire de R est de type standard (§ 5.3.15). Comme C et C' sont contenues dans $C \cup C'$, et sont maximales, on a $C = C' = C \cup C'$.

On en conclut que pour une machine de ce type, *l'ensemble des classes maximales est l'unique recouvrement minimal*, puisque ces classes sont mutuellement disjointes.

5.5.3 Démonstration

On démontre ici la transitivité de la relation $\{p, q\} \in \Gamma(2)$ pour une machine complètement spécifiée. A cet effet on considère l'algorithme de détermination de $\Gamma(2)$ (fig. 5.19). On prend pour ensemble de départ Γ , l'ensemble des paires $\{p, q\}$ telles que $(p|x) \cap (q|x) \neq \emptyset (\forall x \in X)$. Dans le cas d'une machine complètement spécifiée cette condition équivaut à $p|x = q|x$. Par suite, pour l'ensemble de départ Γ , la relation $\{p, q\} \subset \Gamma$ est transitive. On dira en bref que Γ est transitif. On montre ensuite que si Γ est transitif, alors $\Gamma - \mathbf{Sp}(\Gamma)$ est transitif. Supposons Γ transitif, et soient $\{p, q\}$ et $\{q, r\}$ deux paires distinctes appartenant à $\Gamma - \mathbf{Sp}(\Gamma)$. La paire $\{q, r\}$ appartient à Γ par transitivité. Soient $x \in X$ et $p' = p + x, q' = q + x, r' = r + x$. On a

$$\begin{aligned} p' &= q' \quad \text{ou} \quad \{p', q'\} \in \Gamma \\ q' &= r' \quad \text{ou} \quad \{q', r'\} \in \Gamma \end{aligned}$$

en vertu de l'hypothèse que $\{p, q\}$ et $\{q, r\}$ n'appartiennent pas à $\mathbf{Sp}(\Gamma)$. Il vient $p' = r'$ ou $\{p', r'\} \in \Gamma$ (par transitivité). Donc $\{p, r\}$ n'appartient pas à $\mathbf{Sp}(\Gamma)$. Ainsi $\Gamma - \mathbf{Sp}(\Gamma)$ est transitif. Dans l'algorithme de la figure 5.19, l'opération $\Gamma := \Gamma - \mathbf{Sp}(\Gamma)$ conserve la transitivité. Il s'ensuit que l'ensemble final $\Gamma(2)$ est transitif.

5.5.4 Fusionnements

Supposons que pour une machine $R[X, Y, Z]$ on ait $Y = \{y_1, \dots, y_n\}$, et qu'un sous-ensemble $C = \{y_1, \dots, y_m\} \subset Y$ vérifie les conditions suivantes :

$$\bigcap_{p \in C} (p|x) \neq \emptyset \quad \text{quel que soit } x \in X \tag{5.73}$$

$$|C + x| \leq 1 \quad \text{ou} \quad C + x \subset C \quad \text{quel que soit } x \in X. \tag{5.74}$$

Posons

$$C_1 = C \quad \text{et} \quad C_i = \{y_i\} \quad \text{pour } i = m + 1, \dots, n. \tag{5.75}$$

Il est clair que $(C_1, C_{m+1}, \dots, C_n)$ vérifie les conditions de fermeture et de couverture. On dit que la machine quotient de R par ce recouvrement est obtenue par *fusionnement* de y_1, \dots, y_m .

Considérons par exemple la machine R du tableau 5.44. Le sous-ensemble $C = \{1, 2, 3\}$ de Y , vérifie (5.73) et (5.74). Posons donc $C_1 = \{1, 2, 3\}, C_4 = \{4\}, C_5 = \{5\}$. On obtient la machine quotient $R'[X, Y', Z]$ donnée dans le tableau 5.45. Il est évident que R' est réduite. Mais ce n'est pas une réduction minimale de R , car il existe un recouvrement de Y à deux classes seulement : $\{1, 2, 4\}, \{3, 5\}$.

Cet exemple montre qu'en général, si l'on passe d'une machine R à une machine R' par des fusionnements, on ne peut pas trouver une réduction minimale de R parmi les réductions de R' . Cependant dans la pratique, pour de grandes tables, l'opération de fusionnement permet souvent d'obtenir facilement des réductions quasi minimales.

5.5.5 Réductions des machines de Moore

On a vu au paragraphe 2.6.5 que pour toute machine de Moore M il existe une machine de Mealy R ayant même fonctionnement et même composante séquentielle

1	1 ; 0	2 ; -	- ; -	- ; -
2	2 ; -	2 ; 1	3 ; -	- ; -
3	- ; -	- ; -	3 ; 0	4 ; -
4	4 ; 0	2 ; -	5 ; -	4 ; 0
5	5 ; 1	5 ; 0	5 ; 0	4 ; -

Tableau 5.44

1	1 ; 0	1 ; 1	1 ; 0	4 ; -
4	4 ; 0	1 ; -	5 ; -	4 ; 0
5	5 ; 1	5 ; 0	5 ; 0	4 ; -

Tableau 5.45

que M . Cette machine R est unique, et l'on appellera réductions de M toutes les réductions de R .

Le tableau 5.46 représente une machine de Moore M , avec $X = \{a, b, c\}$, $Y = \{1, \dots, 7\}$, et $Z = \mathbf{B}_2$. La machine de Mealy correspondante $R[X, Y, Z]$ est donnée par le tableau 5.47. Elle a la propriété suivante : pour tout élément $p \in Y$,

$$p|x = p|x' \quad \text{quels que soient } x, x' \in X. \quad (5.76)$$

L'ensemble $p|x \subset Z$ ne dépend pas de x ; on pourra le noter $p|$.

On considère en somme une machine de Moore comme une machine de Mealy vérifiant la condition (5.76). Il est clair que toute réduction d'une telle machine vérifiera également cette condition. Par exemple la machine R' du tableau 5.48 est la machine quotient de R (tab. 5.47) par le recouvrement

$$(C_\alpha, C_\beta, C_\gamma) = (\{1, 2, 3\}, \{3, 4, 5\}, \{5, 6, 7\}).$$

Par définition, $\alpha|x = (1|x) \cap (2|x) \cap (3|x)$, ce qui ne dépend pas de x :

$$\alpha|x = 1| \cap 2| \cap 3| = \alpha| \quad (\text{tab. 5.49}).$$

	a	b	c	
1	3	1	7	0 -
2	-	2	5	0 0
3	5	3	-	0 -
4	6	4	5	- -
5	5	-	7	- 1
6	6	4	-	1 -
7	-	-	7	1 1

Tableau 5.46

	a	b	c
1	3 ; 0 -	1 ; 0 -	7 ; 0 -
2	- ; 0 0	2 ; 0 0	5 ; 0 0
3	5 ; 0 -	3 ; 0 -	- ; 0 -
4	6 ; - -	4 ; - -	5 ; - -
5	5 ; - 1	- ; - 1	7 ; - 1
6	6 ; 1 -	4 ; 1 -	- ; 1 -
7	- ; 1 1	- ; 1 1	7 ; 1 1

Tableau 5.47

	a	b	c
α	$\beta ; 0 0$	$\alpha ; 0 0$	$\gamma ; 0 0$
β	$\gamma ; 0 1$	$\beta ; 0 1$	$\gamma ; 0 1$
γ	$\gamma ; 1 1$	$\beta ; 1 1$	$\gamma ; 1 1$

Tableau 5.48

	a	b	c
α	β	α	γ
β	γ	β	γ
γ	γ	β	γ
			0 0
			0 1
			1 1

Tableau 5.49

5.5.6 Réductions asynchrones

Considérons la machine R' du tableau 5.50, avec $X = \{a, b, c, d\}$, $Y = \{1, \dots, 4\}$, et $Z = \{0, 1\}$. On observe la propriété suivante : pour chaque couple d'éléments $y \in Y$, $x \in X$,

$$\left. \begin{aligned} y \cdot xx &= y \cdot x \\ y | xx &= y | x. \end{aligned} \right\} \quad (5.77)$$

Par exemple

$$\begin{aligned} 1 \cdot c &= 2; & 1 \cdot cc &= 2 \cdot c = 2 \\ 1 | c &= 0; & 1 | cc &= (1 \cdot c) | c = 2 | c = 0. \end{aligned}$$

On dit qu'une machine R vérifiant (5.77) est une machine *asynchrone*.

Rappelons (§ 3.3.11) qu'un état secondaire $y \in Y$ est dit stable pour un état primaire $x \in X$, lorsque $y \cdot x = y$. On dit aussi que l'état total $y \times x$ est stable. On met souvent en évidence les états totaux stables $y \times x$, par l'encerclement de l'élément y noté dans la case de coordonnées x, y de la table de transition.

La machine du tableau 5.50 peut être réduite de façon évidente par fusionnement de 1, 2 et de 3, 4 ce qui donne la machine R'' du tableau 5.51.

	<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>
1	① ; 1	① ; 0	2 ; 0	3 ; 1
2	— ; —	— ; —	② ; 0	3 ; 1
3	③ ; 0	1 ; 0	4 ; 1	③ ; 1
4	④ ; 0	1 ; 0	④ ; 1	3 ; 1

Tableau 5.50 Machine R' .

	<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>
1	① ; 1	① ; 0	① ; 0	3 ; 1
3	③ ; 0	1 ; 0	③ ; 1	③ ; 1

Tableau 5.51 Machine R'' .

Considérons maintenant la machine R du tableau 5.52. Elle n'est pas asynchrone au sens ci-dessus. Cependant elle jouit d'une propriété voisine de (5.77), à savoir: il existe un entier $n \geq 1$ tel que

$$y \cdot x^{n+1} = y \cdot x^n \quad \text{et} \quad y | x^{n+1} = y | x^n \quad (5.78)$$

quels que soient $y \in Y$ et $x \in X$. Dans cet exemple (tab. 5.52), la propriété est vraie pour $n = 3$. On a par exemple

$$\left. \begin{aligned} 1 \cdot d^4 &= 1 \cdot d^3 = 3 \\ 1 | d^4 &= 1 | d^3 = 1 \end{aligned} \right\} \quad (5.79)$$

selon les définitions des paragraphes 2.4.17 et 2.6.10.

Il y a entre les machines R (tab. 5.52) et R' (tab. 5.50) la relation suivante (pour $n = 3$):

$$\left. \begin{aligned} (y \cdot x)_{R'} &= (y \cdot x^n)_{R'} \\ (y | x)_{R'} &= (y | x^n)_{R'} \end{aligned} \right\} \quad \text{quels que soient } y \in Y, x \in X. \quad (5.80)$$

	a	b	c	d
1	① ; 1	① ; 0	2 ; 0	2 ; 1
2	- ; -	- ; -	② ; 0	3 ; 0
3	③ ; 0	4 ; 0	4 ; 1	③ ; 1
4	④ ; 0	1 ; 1	④ ; 1	3 ; 1

Tableau 5.52 Machine R .

Ainsi $(1 \cdot d)_R = (1 \cdot d^3)_R = 3$ et $(1|d)_{R'} = (1|d^3)_R = 1$.

En fait la machine R' a été **construite** à partir de R , de façon à satisfaire (5.80). On dit que R' est la *machine asynchrone associée* à R , et que R'' est une *réduction asynchrone* de R . On appelle donc réduction asynchrone d'une machine R , vérifiant (5.78) pour un entier n , toute réduction de la machine asynchrone associée à R selon (5.80).

La figure 5.53 donne la justification de cette terminologie. On y voit une séquence d'entrée x de la forme $d^k a^l b^m$, les nombres k, l, m étant supérieurs à 3. La séquence z est la réponse de R à cette séquence x , pour l'état initial $p = 1$. La séquence z' est la réponse de R' pour le même état initial. On voit que z et z' coïncident à tout instant, sauf éventuellement dans les intervalles transitoires désignés par Tr.

$$\begin{array}{l}
 x = \left| \begin{array}{c} d \ d \\ 1 \ 0 \\ 1 \ 1 \end{array} \right| \left| \begin{array}{c} d \ d \ \dots \\ 1 \ 1 \ \dots \\ 1 \ 1 \ \dots \end{array} \right| \left| \begin{array}{c} d \\ 1 \\ 1 \end{array} \right| \left| \begin{array}{c} a \ a \\ 0 \ 0 \\ 0 \ 0 \end{array} \right| \left| \begin{array}{c} a \ a \ \dots \\ 0 \ 0 \ \dots \\ 0 \ 0 \ \dots \end{array} \right| \left| \begin{array}{c} a \\ 0 \\ 0 \end{array} \right| \left| \begin{array}{c} b \ b \\ 0 \ 1 \\ 0 \ 0 \end{array} \right| \left| \begin{array}{c} b \ b \ \dots \ b \\ 0 \ 0 \ \dots \ 0 \\ 0 \ 0 \ \dots \ 0 \end{array} \right| \\
 z = \left| \begin{array}{c} d \ d \\ 1 \ 0 \\ 1 \ 1 \end{array} \right| \left| \begin{array}{c} d \ d \ \dots \\ 1 \ 1 \ \dots \\ 1 \ 1 \ \dots \end{array} \right| \left| \begin{array}{c} d \\ 1 \\ 1 \end{array} \right| \left| \begin{array}{c} a \ a \\ 0 \ 0 \\ 0 \ 0 \end{array} \right| \left| \begin{array}{c} a \ a \ \dots \\ 0 \ 0 \ \dots \\ 0 \ 0 \ \dots \end{array} \right| \left| \begin{array}{c} a \\ 0 \\ 0 \end{array} \right| \left| \begin{array}{c} b \ b \\ 0 \ 1 \\ 0 \ 0 \end{array} \right| \left| \begin{array}{c} b \ b \ \dots \ b \\ 0 \ 0 \ \dots \ 0 \\ 0 \ 0 \ \dots \ 0 \end{array} \right| \\
 z' = \left| \begin{array}{c} d \ d \\ 1 \ 0 \\ 1 \ 1 \end{array} \right| \left| \begin{array}{c} d \ d \ \dots \\ 1 \ 1 \ \dots \\ 1 \ 1 \ \dots \end{array} \right| \left| \begin{array}{c} d \\ 1 \\ 1 \end{array} \right| \left| \begin{array}{c} a \ a \\ 0 \ 0 \\ 0 \ 0 \end{array} \right| \left| \begin{array}{c} a \ a \ \dots \\ 0 \ 0 \ \dots \\ 0 \ 0 \ \dots \end{array} \right| \left| \begin{array}{c} a \\ 0 \\ 0 \end{array} \right| \left| \begin{array}{c} b \ b \\ 0 \ 1 \\ 0 \ 0 \end{array} \right| \left| \begin{array}{c} b \ b \ \dots \ b \\ 0 \ 0 \ \dots \ 0 \\ 0 \ 0 \ \dots \ 0 \end{array} \right| \\
 \text{Tr} \qquad \qquad \qquad \text{Tr} \qquad \qquad \qquad \text{Tr}
 \end{array}$$

Fig. 5.53

Si l'on ne considère que les séquences d'entrée de la forme $x_1^{n_1} x_2^{n_2} \dots x_m^{n_m}$ telles que $x_i \in X$ et $n_i \geq n$ (n étant le plus petit entier vérifiant (5.78)), et si l'on fait abstraction des intervalles transitoires, les machines R et R' sont équivalentes. En ce sens, la machine R'' est une réduction de R . Mais au sens strict des sections précédentes, la machine R est *réduite*, comme on le voit immédiatement en construisant sa table des implications.

5.5.7 Exercice

Soit R la machine de Moore représentée de façon équivalente par les tableaux 5.54, 5.55. Vérifier que R satisfait (5.78) pour $n = 2$. En particulier, pour $y = 2$ et $x = a$, on a quel que soit $m \geq 2$,

$$y|x^m = (y \cdot x^{m-1})|x = Y|x = \{0,1\}$$

selon les paragraphes 2.4.17 et 2.6.10. Construire la machine asynchrone R' associée à R selon (5.80), et une réduction minimale R'' de R' . Observer sur cet exemple qu'une réduction *asynchrone* d'une machine de Moore peut ne pas être une machine de Moore. Comparer avec le paragraphe 5.5.5.

	<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>	
1	①	2	—	3	0
2	—	②	4	3	0
3	1	—	③	③	1
4	④	2	④	—	1

Tableau 5.54

	<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>
1	① ; 0	2 ; 0	— ; 0	3 ; 0
2	— ; 0	② ; 0	4 ; 0	3 ; 0
3	1 ; 1	— ; 1	③ ; 1	③ ; 1
4	④ ; 1	2 ; 1	④ ; 1	— ; 1

Tableau 5.55

5.5.8 Références bibliographiques

Historiquement, le problème de la réduction a été attaqué sur des cas particuliers [8], [9], [22]. Ainsi dans son travail de pionnier [22], Huffman traite de ce nous appellerions ici les réductions asynchrones de machines de Moore à composante combinatoire complètement spécifiée, ce qui est le dernier exercice ci-dessus.

Actuellement, la plupart des ouvrages théoriques portent sur des machines à composantes séquentielle et combinatoire de type standard, la seconde étant toujours du premier des types standards définis au paragraphe 5.3.21. Le problème théorique central est celui de la relation entre la notion de simulation (sect. 5.1) et celle de machine quotient (sect. 5.2). Ginsburg [4] a été le premier à l'étudier de façon rigoureuse, pour les machines particulières ci-dessus. Notre section 5.2 généralise ses résultats.

En ce qui concerne les méthodes pratiques, celle de la table des implications (§ 5.3.13) est due à Paull et Unger [25], et celle de l'arbre des recouvrements (§ 5.4.23) à Meisel [23]. La notion de classe première est due à Grasselli et Luccio [26].

Il existe de bons exposés plus succincts que le nôtre, parce que moins soucieux de la généralité et des problèmes théoriques qu'elle pose. On peut citer en particulier [10] et [24].

DÉCOMPOSITION ET ASSIGNEMENT DES MACHINES SÉQUENTIELLES

6.1 ASSIGNEMENTS

6.1.1 Introduction et exemple

Considérons la machine de Mealy $R[X, Y, Z]$ définie par le tableau 6.1, avec $X = \mathbf{B}_2$, $Y = \{0, \dots, 4\}$, $Z = \mathbf{B}$. Pour réaliser cette machine par un système logique, il faut la transformer en une machine binaire. A cet effet, on code les éléments de Y (états secondaires) au moyen de combinaisons de valeurs de variables secondaires binaires y_i . Ce procédé, appelé assignement, est décrit dans le volume V (§ V.6.2.1).

Dans notre exemple, trois variables binaires y_1, y_2, y_3 sont nécessaires et suffisantes. Le code choisi peut être quelconque, par exemple celui du tableau 6.3.

		$x_1 x_2$			
		00	01	11	10
0	— ; —	1 ; 0	3 ; 1	4 ; 0	
1	4 ; —	— ; 1	— ; —	2 ; 0	
2	3 ; 0	4 ; 1	— ; —	— ; —	
3	4 ; 1	1 ; 0	1 ; 1	4 ; 1	
4	0 ; 0	2 ; 0	— ; —	— ; —	

Tableau 6.1

		00	01	11	10
0	—	1	3	4	
1	4	—	—	2	
2	3	4	—	—	
3	4	1	1	4	
4	0	2	—	—	

Tableau 6.2

Nous ne nous intéresserons désormais qu'à la composante séquentielle de cette machine (tab. 6.2). Le procédé d'assignement consiste à recopier simplement le tableau 6.2 en utilisant le code choisi, ce qui donne le tableau 6.4, que l'on appelle une forme assignée du tableau 6.2. Il est sous-entendu que le tableau assigné se prolonge par des lignes 101, 110, 111 ne contenant que des tirets.

$y_1 y_2 y_3$	
000	0
001	1
010	2
011	3
100	4

Tableau 6.3

		$x_1 x_2$			
		00	01	11	10
$y_1 y_2 y_3$	000	—	001	011	100
001	100	—	—	010	
010	011	100	—	—	
011	100	001	001	100	
100	000	010	—	—	

$y_1 y_2 y_3$

Tableau 6.4

En remplaçant les tirets du tableau 6.4 par des combinaisons adéquates de valeurs de $y_1 y_2 y_3$, selon la technique de Karnaugh, on peut obtenir les équations de

réurrence suivantes (notation du vol. V), et le lecteur pourra vérifier que ce sont les plus simples possibles :

$$\left. \begin{aligned} y_1^+ &= \bar{y}_3 x_1 \bar{x}_2 + y_2 y_3 \bar{x}_2 + \bar{y}_2 y_3 \bar{x}_1 + y_2 \bar{y}_3 x_2 \\ y_2^+ &= y_2 \bar{y}_3 \bar{x}_2 + \bar{y}_2 x_1 x_2 + \bar{y}_2 y_3 x_1 + y_1 x_2 \\ y_3^+ &= \bar{y}_1 \bar{y}_2 x_2 + y_3 x_2 + y_2 \bar{y}_3 \bar{x}_2. \end{aligned} \right\} \quad (6.1)$$

Considérons maintenant un autre codage, celui du tableau 6.5, dans lequel on s'est permis d'attribuer deux codes à l'état 3. Comme précédemment, on recopie le tableau 6.2 avec ce code (tab. 6.6). Ce faisant, on permute les lignes de façon à faciliter l'application ultérieure de la technique de Karnaugh; on écrit une ligne pour chacun des codes de l'état 3; enfin, chaque fois qu'on rencontre l'état futur 3 dans le tableau 6.2, on s'autorise à le représenter par l'un ou l'autre de ses codes à choix. Ce choix se fait évidemment de façon à rendre le plus simple possible les équations futures, et l'on peut obtenir ici les équations

$$\left. \begin{aligned} y_1^+ &= \bar{y}_1 \bar{x}_2 + y_1 x_2 \\ y_2^+ &= \bar{y}_1 x_2 \\ y_3^+ &= \bar{y}_1 \bar{x}_1 \bar{x}_2 + y_1 \bar{y}_3 + \bar{y}_1 y_3 \bar{x}_2 + \bar{y}_2 \bar{y}_3 x_1. \end{aligned} \right\} \quad (6.2)$$

$y_1 y_2 y_3$			$x_1 x_2$			
			00	01	11	10
000	0	(0) 000	—	(1) 010	(3) 011	(4) 101
001	3	(3) 001	(4) 101	(1) 010	(1) 010	(4) 101
011	3	(3) 011	(4) 101	(1) 010	(1) 010	(4) 101
010	1	(1) 010	(4) 101	—	—	(2) 100
100	2	(2) 100	(3) 001	(4) 101	—	—
101	4	(4) 101	(0) 000	(2) 100	—	—

Tableau 6.5

 $y_1 y_2 y_3$

Tableau 6.6

La simplicité des équations (6.2) comparées aux équations (6.1) est frappante, et les questions que suscite immédiatement cette comparaison peuvent servir d'introduction au présent chapitre. Ce sont les questions suivantes :

- quelle est la notion d'assignement la plus générale ?
- comment trouve-t-on un assignement tel que celui du tableau 6.6 ?

La première question fait l'objet de la présente section, avec quelques autres notions préalables aux sections suivantes.

Quant à la seconde question, il faut évidemment en préciser les termes pour pouvoir lui donner une réponse. Il faut caractériser l'assignement considéré par une propriété bien précise.

La simplicité des équations (6.2) comparées aux équations (6.1) peut se mesurer selon différents critères. Le critère classique de la minimisation des expressions booléennes (vol. V) est le nombre des lettres des expressions. Ce n'est pas ce critère que nous utiliserons, et pour trois raisons : premièrement, trouver une méthode d'assignement qui garantisse l'obtention d'équations minimales selon ce critère, constitue un problème théorique d'une difficulté sans rapport avec l'importance du sujet; secondement le critère de minimisation classique des expressions booléennes est lié à une forme particu-

lière de systèmes combinatoires, à savoir les assemblages de portes à deux niveaux (somme de produits); troisièmement une théorie de l'assignement développée selon ce critère serait spécifique aux assignements binaires, alors qu'un autre critère vérifié par les équations (6.2) est indépendant de leur caractère booléen et permet de poser le problème de l'assignement sous une forme plus générale.

On observe dans les équations (6.2) que y_1^+ ne dépend pas de y_2 et y_3 , et que y_2^+ ne dépend pas de y_3 . En conséquence, la machine séquentielle $M(x_1, x_2)(y_1, y_2, y_3)$ définie par ces équations peut être réalisée par la composition de trois machines séquentielles $M_1(x_1, x_2)(y_1)$, $M_2(x_1, x_2, y_1)(y_2)$, $M_3(x_1, x_2, y_1, y_2)(y_3)$, selon le schéma de la figure 6.7, dans lequel x est mis pour $x_1 \times x_2$. On dit que la machine définie par les équations (6.2) est décomposable en série, et cette propriété n'est pas liée au caractère binaire de la machine.

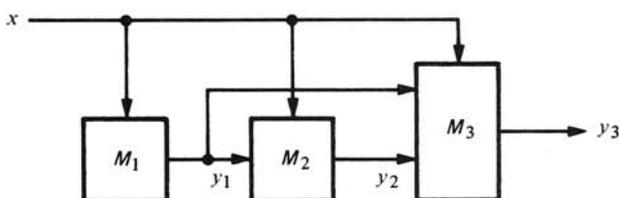


Fig. 6.7

C'est cette possibilité de décomposition que nous aurons en vue dans la recherche d'assignements, et l'expérience montre qu'en général un assignement décomposable conduit à des équations plus simples qu'un assignement non décomposable.

On étudie la décomposition série dans la section 6.2, et la décomposition parallèle dans la section 6.3. Etant donné une machine séquentielle $S: X^+ \rightarrow Y^+$ (tab. 6.2), tout assignement décomposable est lié à un recouvrement de Y compatible avec S (§ 5.2.3). Nous reviendrons sur cette notion fondamentale dans la présente section. La section 6.4 est consacrée au cas particulier où ce recouvrement est une partition de Y , et S est de type standard.

6.1.2 Notations et conventions

A quelques exceptions près, toutes les machines considérées dans ce chapitre sont des machines séquentielles au sens de la section 2.4, et seront appelées machines tout court.

Le terme de *machine générale* $M: X^+ \rightarrow Y^+$ désigne une machine au sens général de la section 2.1, avec la seule hypothèse supplémentaire que $M(x) \neq \emptyset$ pour toute séquence $x \in X^+$. Ce peut être une machine combinatoire, une machine séquentielle, une machine de Mealy, etc. Une telle machine est notée $M[X, Y]$.

Lorsqu'une lettre majuscule telle que U désigne un ensemble, les lettres minuscules correspondantes u, u', u'', \dots désignent, sauf mention contraire, des éléments de U .

6.1.3 Recouvrements

Rappelons qu'une correspondance $\sigma: U \rightarrow Y$ est appelée un recouvrement de Y si l'on a

$$\bigcup_{u \in U} \sigma(u) = Y$$

ce qui peut s'abrégé $\sigma(U) = Y$ (§ 1.3.3). Les ensembles $\sigma(u)$ sont appelés les classes du recouvrement.

6.1.4 Réciproque d'un recouvrement

La correspondance réciproque $\sigma^{-1} : Y \rightarrow U$ d'une correspondance $\sigma : U \rightarrow Y$ a été définie au paragraphe 1.3.8. Par exemple, le tableau 6.8 représente une correspondance $\sigma : U \rightarrow Y$, avec $U = \{\alpha, \beta, \gamma, \delta\}$ et $Y = \{0, 1, 2, 3, 4\}$. La figure 6.9 est le diagramme de cette correspondance et il suffit d'y inverser le sens des flèches pour obtenir le diagramme de σ^{-1} , d'où le tableau 6.10. On a par définition

$$y \in \sigma(u) \iff u \in \sigma^{-1}(y). \tag{6.4}$$

On fera souvent usage de la relation suivante : si P est un sous-ensemble de Y,

$$u \in \sigma^{-1}(P) \iff \sigma(u) \cap P \neq \emptyset. \tag{6.5}$$

On peut contrôler cette relation dans la figure 6.9 en prenant pour P l'ensemble $\{1, 2, 4\}$. On a $\sigma^{-1}(P) = \{\beta, \gamma\}$ et l'on vérifie que β et γ sont les seuls éléments u tels que $\sigma(u) \cap P \neq \emptyset$.

On observe dans cet exemple que σ est un recouvrement de Y, alors que σ^{-1} n'est pas un recouvrement de U : $\sigma^{-1}(Y) = \{\alpha, \beta, \gamma\}$. Cet exemple nous suffira comme justification de la proposition suivante.

u	$\sigma(u)$
α	$\{0, 3\}$
β	$\{1, 3\}$
γ	$\{2, 4\}$
δ	\emptyset

Tableau 6.8

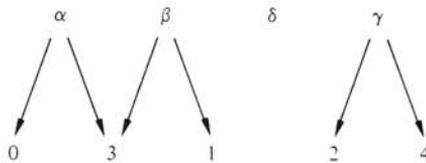


Fig. 6.9

y	$\sigma^{-1}(y)$
0	$\{\alpha\}$
1	$\{\beta\}$
2	$\{\gamma\}$
3	$\{\alpha, \beta\}$
4	$\{\gamma\}$

Tableau 6.10

6.1.5 Proposition

Soit $\sigma : U \rightarrow Y$ un recouvrement de Y. Pour tout $y \in Y$, $\sigma^{-1}(y) \neq \emptyset$. L'ensemble $\sigma^{-1}(Y)$ est l'ensemble des u tels que $\sigma(u) \neq \emptyset$. Pour que σ^{-1} soit un recouvrement de U, il faut et il suffit que σ n'ait pas de classe vide.

6.1.6 Composition de deux correspondances

Soient $\rho : U \rightarrow A$ et $\varphi : A \rightarrow Y$ deux correspondances, l'ensemble d'arrivée de la première étant l'ensemble de départ de la seconde. On définit la correspondance composée $\sigma = \varphi\rho : U \rightarrow Y$ (noter l'ordre des facteurs) en posant pour tout u

$$\sigma(u) = \bigcup_{a \in \rho(u)} \varphi(a) = \varphi(\rho(u)). \tag{6.6}$$

On a donc

$$\varphi\rho(u) = \varphi(\rho(u)) \tag{6.7}$$

et par suite

$$y \in \varphi \rho(u) \iff \text{il existe un } a \in \rho(u) \text{ tel que } y \in \varphi(a). \tag{6.8}$$

Par exemple, soient $U = \{p, q, r\}$, $A = \{\alpha, \beta, \gamma, \delta\}$, $Y = \{0, 1, 2, 3, 4\}$ et $\rho: U \rightarrow A$, $\varphi: A \rightarrow Y$ les correspondances représentées par le diagramme de la figure 6.11. Selon (6.8), la figure 6.12 représente la correspondance $\varphi\rho$.

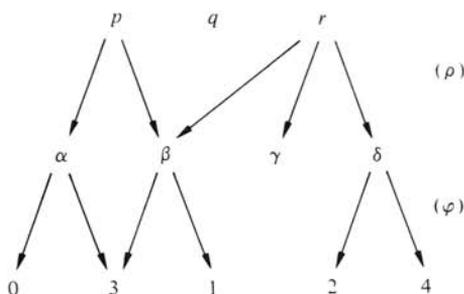


Fig. 6.11

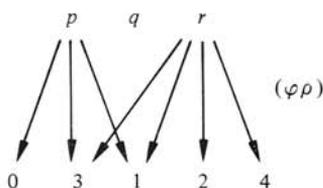


Fig. 6.12

Nous admettrons comme évidentes les trois propositions qui suivent.

6.1.7 Proposition

Soient $\rho: U \rightarrow A$ et $\varphi: A \rightarrow Y$ deux correspondances. La réciproque de la correspondance composée $\varphi\rho$ est la composée des réciproques dans l'ordre inverse :

$$(\varphi\rho)^{-1} = \rho^{-1}\varphi^{-1}. \tag{6.9}$$

Il suffit en effet de remarquer que l'inversion du sens des flèches dans la figure 6.11 produit simplement l'inversion des flèches dans la figure 6.12.

6.1.8 Proposition

Si $\rho: U \rightarrow A$ est un recouvrement de A , et $\varphi: A \rightarrow Y$ un recouvrement de Y , alors $\varphi\rho: U \rightarrow Y$ est un recouvrement de Y (figures 6.11, 6.12).

6.1.9 Proposition

Si $\varphi: A \rightarrow Y$ est un recouvrement de Y , on a

$$S \subset \varphi\varphi^{-1}(S) \text{ pour tout ensemble } S \subset Y. \tag{6.10}$$

Par exemple pour φ (fig. 6.11) et $S = \{0, 1\}$, on a $\varphi^{-1}(S) = \{\alpha, \beta\}$ et $\varphi(\varphi^{-1}(S)) = \varphi(\{\alpha, \beta\}) = \{0, 1, 3\}$.

6.1.10 Codages

Une correspondance $\varphi: A \rightarrow Y$ est appelée un *codage de Y*, ou simplement un codage, si φ est un *recouvrement* de Y et si l'on a

$$|\varphi(a)| \leq 1 \text{ quel que soit } a \in A. \tag{6.11}$$

Les classes du recouvrement ont au plus un élément. Un exemple de codage $\varphi: A \rightarrow Y$, avec $A = \{p, q, r, s, t\}$ et $Y = \{0, 1, 2\}$ est donné dans la figure 6.13 et le tableau correspondant 6.14. La relation $y \in \varphi(a)$ équivaut à $\varphi(a) = \{y\}$, et s'écrit généralement $\varphi(a) = y$. On dit que y est codé par a , ou que a code y . La définition générale d'un codage autorise qu'un élément y soit codé par plusieurs éléments a, a', \dots , mais un élément a quelconque code au plus un élément y . Enfin tout élément y est codé par au moins un élément a .



Fig. 6.13

a	$\varphi(a)$
p	$\{0\}$
q	$\{0\}$
r	$\{1\}$
s	\emptyset
t	$\{2\}$

Tableau 6.14

6.1.11 Proposition

Si $\varphi: A \rightarrow Y$ est un codage, on a

$$S = \varphi\varphi^{-1}(S) \quad \text{pour tout ensemble } S \subset Y \quad (6.12)$$

et en particulier

$$y = \varphi\varphi^{-1}(y) \quad \text{quel que soit } y \in Y. \quad (6.13)$$

La vérification de ces assertions est immédiate. On remarquera que la différence entre (6.12) et (6.10) provient de (6.11). Enfin l'on a

$$|\varphi(P)| \leq |P| \quad \text{pour tout ensemble } P \subset A. \quad (6.14)$$

6.1.12 Proposition

Soient A et Y deux ensembles finis. Pour qu'il existe un codage $\varphi: A \rightarrow Y$, il faut et il suffit que $|A| \geq |Y|$.

La proposition est évidente.

6.1.13 Partitions

Une correspondance $\pi: U \rightarrow A$ est appelée une *partition* de A si π est un *recouvrement* de A et si

$$u \neq u' \Rightarrow \pi(u) \cap \pi(u') = \emptyset. \quad (6.15)$$

On dit que les classes d'un tel recouvrement sont mutuellement disjointes. Un exemple avec $U = \{p, q, r\}$ et $A = \{\alpha, \beta, \gamma, \delta, \epsilon\}$ est donné par la figure 6.15 et le tableau 6.16.

On remarquera que la réciproque π^{-1} d'une partition $\pi: U \rightarrow A$ est une *application* de A dans U , et que l'on a

$$a \in \pi(u) \iff \pi^{-1}(a) = u. \quad (6.16)$$

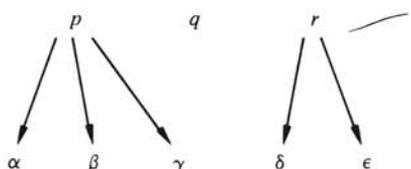


Fig. 6.15

u	$\pi(u)$
p	$\{\alpha, \beta, \gamma\}$
q	\emptyset
r	$\{\delta, \epsilon\}$

Tableau 6.16

6.1.14 Partitions canoniques d'un produit cartésien

Soient U et V deux ensembles finis et non vides. Nous appelons *partitions canoniques* de l'ensemble $U \times V$ les correspondances

$$\pi : U \rightarrow U \times V$$

$$\tau : V \rightarrow U \times V$$

définies par

$$\left. \begin{aligned} \pi(u) &= u \times V \\ \tau(v) &= U \times v \end{aligned} \right\} \text{quels que soient } u, v. \tag{6.17}$$

Pour bien "voir" ces deux partitions, on peut représenter les ensembles U et V par des ensembles de points sur deux axes de coordonnées (fig. 6.17), et associer à chaque couple $u \times v$ le point du plan de coordonnées u, v . L'ensemble $U \times V$ est ainsi représenté par une matrice de points du plan. Une classe $\pi(u) = u \times V$ (resp. $\tau(v) = U \times v$) est représentée par une colonne (resp. une ligne) de cette matrice. Il est évident que π et τ sont deux partitions de $U \times V$, et de plus n'ont pas de classe vide.

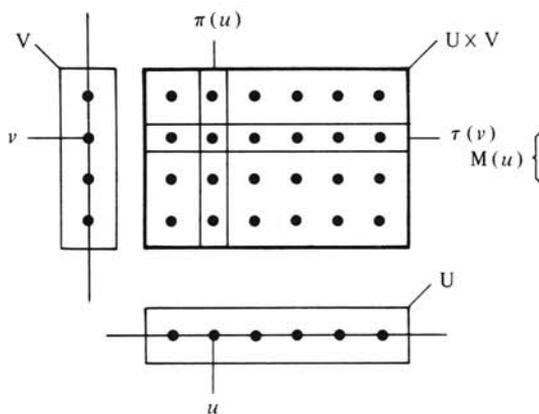


Fig. 6.17

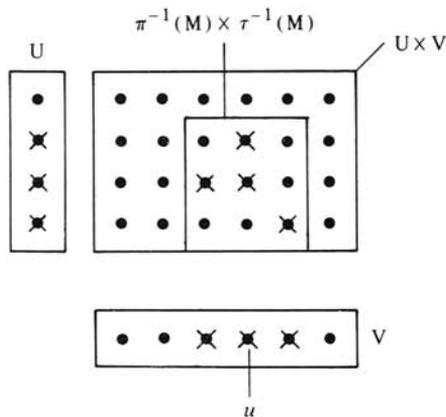


Fig. 6.18

Indépendamment de cette représentation, on dit que pour un couple $u \times v$, u est la première coordonnée et v la seconde coordonnée du couple. Ainsi l'ensemble $\pi(u)$ (resp. $\tau(v)$) est l'ensemble des éléments de $U \times V$ dont la première (resp. la seconde) coordonnée est u (resp. v). On a trivialement

$$\left. \begin{aligned} u \times v &\in \pi(u) \\ u \times v &\in \tau(v) \end{aligned} \right\} \text{quels que soient } u, v \tag{6.18}$$

d'où par (6.16)

$$\left. \begin{array}{l} \pi^{-1}(u \times v) = u \\ \tau^{-1}(u \times v) = v \end{array} \right\} \text{quels que soient } u, v. \quad (6.19)$$

De ce fait, les applications $\pi^{-1} : U \times V \rightarrow U$ et $\tau^{-1} : U \times V \rightarrow V$ sont appelées *projections* de $U \times V$ sur U et sur V .

Si M est un sous-ensemble de $U \times V$, l'ensemble $\pi^{-1}(M)$ (resp. $\tau^{-1}(M)$) est l'ensemble des premières (resp. des secondes) coordonnées des éléments de M , et l'on a

$$M \subset \pi^{-1}(M) \times \tau^{-1}(M). \quad (6.20)$$

Cette relation est illustrée par la figure 6.18, où M est l'ensemble des points marqués d'une croix dans $U \times V$. On a conformément à (6.5) :

$$\left. \begin{array}{l} u \in \pi^{-1}(M) \iff \pi(u) \cap M \neq \emptyset \\ v \in \tau^{-1}(M) \iff \tau(v) \cap M \neq \emptyset. \end{array} \right\} \quad (6.21)$$

Pour deux sous-ensembles $Q \subset U$ et $S \subset V$, on a

$$\pi^{-1}(Q \times S) = Q \quad \text{et} \quad \tau^{-1}(Q \times S) = S. \quad (6.22)$$

donc

$$Q \times S = \pi^{-1}(Q \times S) \times \tau^{-1}(Q \times S). \quad (6.23)$$

6.1.15 Notation

Continuant avec les définitions du paragraphe qui précède, pour un sous-ensemble M quelconque de $U \times V$, et pour un élément u , nous désignerons par $M(u)$ l'ensemble des éléments v tels que $u \times v \in M$. Cette définition est illustrée par la figure 6.18. On a par définition

$$v \in M(u) \iff u \times v \in M. \quad (6.24)$$

On voit dans la figure 6.18 que la relation

$$u \times M(u) = \pi(u) \cap M \subset M \quad (6.25)$$

est toujours vraie.

6.1.16 Compatibilité

Soit $S[X, Y]$ une machine séquentielle, et $\sigma : U \rightarrow Y$ un recouvrement de Y . Rappelons que σ est dit compatible avec S (§ 5.2.2, 5.2.3) si étant donné deux éléments quelconques u, x il existe un élément u' pour lequel la relation suivante est vérifiée :

$$\forall y \in \sigma(u) : (y \cdot x)_S \cap \sigma(u') \neq \emptyset. \quad (6.26)$$

Or la relation $(y \cdot x) \cap \sigma(u') \neq \emptyset$ équivaut à $u' \in \sigma^{-1}(y \cdot x)$ selon (6.5). Par conséquent la relation (6.26) peut s'écrire sous les formes suivantes :

$$\begin{aligned} \forall y \in \sigma(u) : u' \in \sigma^{-1}(y \cdot x) \\ u' \in \bigcap_{y \in \sigma(u)} \sigma^{-1}(y \cdot x). \end{aligned} \quad (6.27)$$

Ainsi, dire que σ est compatible avec S revient à dire qu'étant donné deux éléments u, x il existe un u' vérifiant (6.27), ou plus simplement encore :

$$\sigma \text{ est compatible avec } S \iff \bigcap_{y \in \sigma(u)} \sigma^{-1}(y \cdot x)_S \neq \emptyset \quad (\forall u, x). \quad (6.28)$$

6.1.17 Définition : machine quotient

Soient $S[X, Y]$ une machine séquentielle, et $\sigma : U \rightarrow Y$ un recouvrement de Y compatible avec S . On appelle *machine quotient* de S par σ , et l'on note S/σ la machine séquentielle d'alphabets $[X, U]$ définie par

$$(u \cdot x)_{S/\sigma} = \bigcap_{y \in \sigma(u)} \sigma^{-1}(y \cdot x). \quad (6.29)$$

On a en particulier

$$(u \cdot x)_{S/\sigma} = U \text{ lorsque } \sigma(u) = \emptyset. \quad (6.30)$$

6.1.18 Proposition

Soient $R[X, Y, Z]$ une machine de Mealy, $S[X, Y]$ sa composante séquentielle, et $\sigma : U \rightarrow Y$ un recouvrement compatible avec R (§ 5.2.9). On a défini au paragraphe 5.2.11 la machine quotient $R'[X, U, Z]$ de R par σ . Notons la R/σ . La composante séquentielle de R/σ est la machine S/σ définie par (6.29).

Il suffit pour le voir d'écrire (5.16) avec u' au lieu de v et de y au lieu de p . Selon le paragraphe 6.1.16, l'ensemble $(u \cdot x)_R$ n'est autre que l'ensemble des u' vérifiant (6.27).

6.1.19 Rappel

On sait que si $S[X, Y]$ est une machine séquentielle de type standard, alors (§ 5.2.7) la condition nécessaire et suffisante pour qu'un recouvrement $\sigma : U \rightarrow Y$ soit compatible avec S , est que pour deux éléments u, x quelconques, il existe un élément u' tel que $\sigma(u) + x \subset \sigma(u')$. On sait également (5.19) que $(u \cdot x)_{S/\sigma}$ est égal à : l'ensemble U si $\sigma(u) = \emptyset$; l'ensemble des u' tels que $\sigma(u) + x \subset \sigma(u')$ et $\sigma(u') \neq \emptyset$ si $\sigma(u) \neq \emptyset$.

6.1.20

Soient $S[X, Y]$ la machine du tableau 6.19 avec $X = \{0, \dots, 3\}$, $Y = \{0, \dots, 4\}$, et $\sigma : U \rightarrow Y$ le recouvrement du tableau 6.20 avec $U = \{\alpha, \beta, \gamma\}$. On vérifie immédia-

	0	1	3	2
0	—	1	3	4
1	4	—	—	2
2	3	4	—	—
3	4	1	1	4
4	0	2	—	—

S

Tableau 6.19

u	$\sigma(u)$
α	$\{0, 3\}$
β	$\{1, 3\}$
γ	$\{2, 4\}$

Tableau 6.20

	0	1	3	2
α	γ	β	β	γ
β	γ	β	β	γ
γ	α	γ	—	—

S/ σ
Tableau 6.21

tement que σ est compatible avec S selon le paragraphe précédent, et que la machine S/σ est celle du tableau 6.21. Cette machine quotient est aussi de type standard, mais c'est un fait accidentel.

6.1.21 Rappel

Pour deux machines générales $S[X, Y]$, $S'[X, Y]$ on écrit $S \subset S'$ lorsque $S(x) \subset S'(x)$ pour toute séquence $x \in X^+$ (§ 2.5.6). Si S et S' sont des machines séquentielles (§ 2.5.8),

$$S \subset S' \iff \begin{cases} (y \cdot x)_S \subset (y \cdot x)_{S'} \\ \text{quels que soient } y, x. \end{cases} \quad (6.31)$$

6.1.22 Proposition

Soient $S[X, Y]$, $S'[X, Y]$ deux machines séquentielles telles que $S \subset S'$, et $\sigma: U \rightarrow Y$ un recouvrement de Y . Si σ est compatible avec S , alors σ est compatible avec S' , et l'on a $S/\sigma \subset S'/\sigma$.

6.1.23 Démonstration

Si $S \subset S'$ on a $\sigma^{-1}(y \cdot x)_S \subset \sigma^{-1}(y \cdot x)_{S'}$ en vertu de (6.31), donc

$$\bigcap_{y \in \sigma(u)} \sigma^{-1}(y \cdot x)_S \subset \bigcap_{y \in \sigma(u)} \sigma^{-1}(y \cdot x)_{S'}. \quad (6.32)$$

La proposition découle alors de (6.28) et (6.29).

6.1.24 Proposition

Soit $S[X, Y]$ une machine séquentielle. Tout codage $\varphi: A \rightarrow Y$ est compatible avec S , et l'on a

$$(a \cdot x)_{S/\varphi} = \begin{cases} A & \text{si } \varphi(a) = \emptyset \\ \varphi^{-1}(\varphi(a) \cdot x)_S & \text{si } \varphi(a) \neq \emptyset. \end{cases} \quad (6.33)$$

Par suite

$$\varphi(a) \neq \emptyset \Rightarrow \varphi(a \cdot x) = \varphi(a) \cdot x. \quad (6.34)$$

Désignons par \tilde{A} l'ensemble des $a \in A$ tels que $\varphi(a) \neq \emptyset$.

$$\text{Si } (\varphi(a) \cdot x)_S = Y, \text{ alors } (a \cdot x)_{S/\varphi} = \tilde{A}. \quad (6.35)$$

6.1.25 Démonstration

Il suffit de considérer l'ensemble

$$\bigcap_{y \in \varphi(a)} \varphi^{-1}(y \cdot x). \quad (6.36)$$

Si $\varphi(a) = \emptyset$ cet ensemble est égal à A . Si $\varphi(a) \neq \emptyset$ la relation $y \in \varphi(a)$ équivaut à $y = \varphi(a)$ (§ 6.1.10), et l'ensemble (6.36) est égal à $\varphi^{-1}(\varphi(a) \cdot x)$, ce qui n'est pas vide.

Ceci démontre (6.33). Par suite, si $\varphi(a) \neq \emptyset$, on a (en omettant les indices S/φ et S)

$$\begin{aligned} \varphi(a \cdot x) &= \varphi\varphi^{-1}(\varphi(a) \cdot x) \quad \text{par (6.33)} \\ &= \varphi(a) \cdot x \quad \text{par (6.12), d'où (6.34).} \end{aligned}$$

Supposons que pour deux éléments a, x on ait $(\varphi(a) \cdot x)_S = Y$. Alors $\varphi(a) \neq \emptyset$, et l'on a $(a \cdot x)_{S/\varphi} = \varphi^{-1}(Y)$ par (6.33). Or $\varphi^{-1}(Y) = \tilde{A}$ (§ 6.1.5).

6.1.26 Exemple

Soient $X = \{0,1\}$, $Y = \{p,q,r\}$, $A = \{0,1,2,3,4\}$, et $S[X,Y]$ la machine du tableau 6.22. La machine quotient S/φ par le codage $\varphi : A \rightarrow Y$ du tableau 6.23 est donnée par le tableau 6.24. Le tiret du tableau 6.22 représente l'ensemble Y ; ceux du tableau 6.24 l'ensemble A . La relation (6.33) peut s'exprimer ainsi: si $a \neq 3$, le contenu d'une case d'adresse $a \times x$ dans le tableau 6.24 est l'ensemble des codes des éléments y se trouvant dans la case correspondante, d'adresse $\varphi(a) \times x$, du tableau 6.22. En particulier, pour $a \times x = 1 \times 1$, $\varphi(a) \cdot x = q \cdot 1 = Y$, d'où $a \cdot x = \tilde{A} = \{0,1,2,4\}$.

	0	1
p	p	q
q	r	—
r	q	p, r
	S	

Tableau 6.22

	a	$\varphi(a)$
0	p	
1	q	
2	r	
3	\emptyset	
4	p	

Tableau 6.23

	0	1
0	0,4	1
1	2	\tilde{A}
2	1	0,2,4
3	—	—
4	0,4	1
	S/ φ	

Tableau 6.24

6.1.27 Définition : assignements

Soient $S[X,Y]$ et $M[X,A]$ deux machines séquentielles. On dit que M est un *assignement* de S , s'il existe un codage $\varphi : A \rightarrow Y$ tel que $M \subset S/\varphi$.

6.1.28 Proposition

Si $M[X,A]$ est un assignement de $S[X,Y]$, alors pour toute machine de Mealy $R[X,Y,Z]$ de composante séquentielle S , il existe une machine de Mealy $R'[X,A,Z]$ de composante séquentielle M qui simule R au sens du paragraphe 5.1.2.

6.1.29 Démonstration

Soit $\varphi : A \rightarrow Y$ un codage tel que

$$M \subset S/\varphi \tag{6.37}$$

et soit $R[X,Y,Z]$ une machine de Mealy de composante séquentielle S . Il est clair que φ est compatible avec la composante combinatoire de R (§ 5.2.9) puisque

$$\varphi(a) \neq \emptyset \Rightarrow |\varphi(a)| = 1$$

donc

$$\varphi(a) \neq \emptyset \Rightarrow \bigcap_{y \in \varphi(a)} (y|x)_R = (\varphi(a)|x)_R \neq \emptyset. \tag{6.38}$$

Soit R/φ la machine quotient de R par φ (§ 5.2.11). On sait que la composante séquentielle de R/φ est S/φ (§ 6.1.18). On a donc

$$(a \cdot x)_{S/\varphi} = (a \cdot x)_{R/\varphi}. \quad (6.39)$$

Soit $R'[X, A, Z]$ la machine de Mealy de composante séquentielle $M[X, A]$, et de composante combinatoire identique à celle de R/φ . Autrement dit

$$(a \cdot x)_{R'} = (a \cdot x)_M \quad (6.40)$$

$$(a | x)_{R'} = (a | x)_{R/\varphi} \quad (6.41)$$

En vertu de (6.37) et (6.39), on a

$$(a \cdot x)_{R'} \subset (a \cdot x)_{R/\varphi}. \quad (6.42)$$

Les relations (6.41) et (6.42) signifient que R' est une diminution de R/φ au sens du paragraphe 5.2.25, et l'on a

$$R'_a \subset (R/\varphi)_a \quad \text{quel que soit } a \quad (6.43)$$

en vertu du paragraphe 5.2.26. Par ailleurs R/φ simule R (§ 5.2.13) et l'on a plus précisément

$$y = \varphi(a) \Rightarrow (R/\varphi)_a \subset R_y. \quad (6.44)$$

Par (6.43) et (6.44),

$$y = \varphi(a) \Rightarrow R'_a \subset R_y.$$

Comme il existe pour tout y un élément a tel que $y = \varphi(a)$, on voit que R' simule R (§ 5.1.2).

6.1.30 Commentaire

La notion d'assignement définie au paragraphe 6.1.27 est à la fois plus générale et plus restrictive que celle du volume V, rappelée dans notre introduction (§ 6.1.1). Elle est plus générale en ce sens qu'elle s'applique aux machines quelconques, de type standard ou non. Par contre elle est plus restrictive en ce qui concerne les machines de type standard, comme nous allons le voir sur un exemple.

Appliquons le procédé d'assignement classique à la machine de Mealy $R[X, Y, Z]$ du tableau 6.25, où $X = Z = \mathbf{B}$ et $Y = \{p, q, r\}$, en prenant le codage $\varphi: A = \mathbf{B}_2 \rightarrow Y$ du tableau 6.26.

	0	1
p	$p ; 1$	$q ; 0$
q	$r ; 1$	— ; —
r	$q ; 0$	$r ; 1$

$R[X, Y, Z]$

Tableau 6.25

$a = y_1 y_2$	$\varphi(a)$
0 = 00	$\{p\}$
1 = 01	$\{q\}$
3 = 11	\emptyset
2 = 10	$\{r\}$

Tableau 6.26

Le procédé classique est de recopier simplement la table de transition donnée en utilisant le code choisi, c'est-à-dire former le tableau (6.27), en écrivant une ligne de tirets pour l'élément $a = 11$ tel que $\varphi(a) = \emptyset$. Notons que le tableau 6.27 *n'est pas* la

machine quotient de R par φ , à cause du tiret qui s'y trouve encadré. La machine R/φ est donnée dans le tableau 6.28, où à la place du tiret incriminé se trouve l'ensemble plus restreint $\tilde{A} = \{0,1,2\}$.

	x	
	0	1
00	00 ; 1	01 ; 0
01	10 ; 1	— ; —
11	— ; —	— ; —
10	01 ; 0	10 ; 1
$y_1 y_2$		

Tableau 6.27

	0	1
0	0 ; 1	1 ; 0
1	2 ; 1	\tilde{A} ; —
3	— ; —	— ; —
2	1 ; 0	2 ; 1

$R/\varphi = R''[X, A, Z]$

Tableau 6.28

En appliquant la technique de Karnaugh au tableau 6.27, on écrit naturellement les équations les plus simples possibles, à savoir

$$\left. \begin{aligned} y_1^+ &= y_2 + y_1 x \\ y_2^+ &= \bar{y}_1 x + y_1 \bar{x} \\ z &= \bar{y}_1 \bar{x} + y_1 x. \end{aligned} \right\} \quad (6.45)$$

Ces équations définissent une machine $R'[X, A, Z]$ complètement spécifiée (tab. 6.29, 6.30).

	0	1
00	00 ; 1	01 ; 0
01	10 ; 1	11 ; 0
11	11 ; 0	10 ; 1
10	01 ; 0	10 ; 1

Tableau 6.29

	0	1
0	0 ; 1	1 ; 0
1	2 ; 1	3 ; 0
3	3 ; 0	2 ; 1
2	1 ; 0	2 ; 1

$R'[X, A, Z]$

Tableau 6.30

Notons que si l'on appelle $S[X, Y]$ la composante séquentielle de R (tab. 6.25), et $M[X, A]$ celle de R' (tab. 6.30), la relation $M \subset S/\varphi$ n'est pas vérifiée, à cause de l'élément 3 encadré dans le tableau 6.30, qui n'appartient pas à l'ensemble \tilde{A} figurant dans la case correspondante du tableau 6.28. La relation $M \subset S/\varphi$ aurait été vérifiée si l'on avait veillé, en appliquant la technique de Karnaugh au tableau 6.27, à ne pas obtenir la combinaison 11 à la place du tiret encadré, mais seulement l'une des combinaisons 00, 01, 10. Il eût été possible de le faire en ne compliquant que très peu les équations (6.45). Si l'on avait veillé à ceci, la machine R' (tab. 6.30) serait une diminution (§ 5.2.25) de la machine quotient $R''[X, A, Z] = R/\varphi$ (tab. 6.28). Il serait certain alors que l'état 1 de R' simulerait l'état q de R (codé par 1) en ce sens que l'on aurait $R'_1 \subset R_q$, car on aurait $R'_1 \subset R''_1$ (§ 5.2.26) et l'on a $R''_1 \subset R_q$ (§ 5.2.13). Et de fait, n'ayant pas pris cette précaution, nous allons voir que la relation $R'_1 \subset R_q$ n'est pas vérifiée. Il suffit de calculer $R'_1(x)$ et $R_q(x)$ pour la séquence $x = 100$ en appliquant (2.82) et (2.83). Ce calcul est effectué graphiquement dans les figures 6.31 et 6.32 (Cf. § 2.6.16), et l'on voit que $R'_1(100) = 000$, tandis que $R_q(100) = (0 \cup 1)(11 \cup 10 \cup 01)$, donc $R'_1(100) \not\subset R_q(100)$.

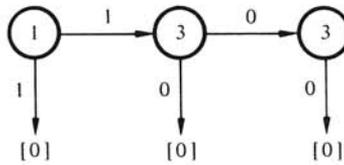


Fig. 6.31

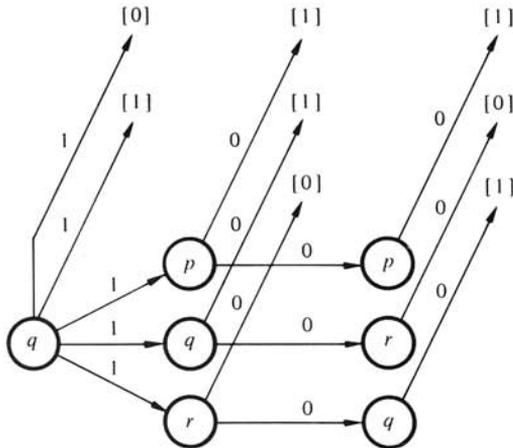


Fig. 6.32

Dans la plupart des ouvrages, on ne traite que des machines $R[X, Y, Z]$ de type standard. On dit alors qu'une séquence d'entrée $x(1, n)$ ($n > 1$) est *applicable* à un état secondaire y si dans le graphe de transition de la composante séquentielle de R , il y a un chemin et un seul d'origine y et d'étiquette $x(1, n-1)$. Ceci peut s'exprimer par $|y \cdot x(1, k)| = 1$ pour $k = 1, \dots, n-1$. Si a est un état secondaire d'une machine $R'[X, A, Y]$, on dit que a simule y si l'on a $R'_a(x) \subset R_y(x)$ pour toute séquence x de longueur 1 et pour toute séquence x *applicable* à y . En ce sens, l'état 1 de R' (tab. 6.30) simule l'état q de R (tab. 6.27). La séquence $x = 100$ que nous avons considérée n'est simplement pas applicable à q . Cette notion de simulation est moins stricte que celle du présent ouvrage, où il n'est pas question de séquences inapplicables. Toutefois c'est une notion ad hoc qui n'est pas généralisable aux machines de type non standard.

Nous nous en tiendrons à la notion de simulation stricte du paragraphe 5.1.2, tout au moins dans la théorie. Ceci n'entraîne aucune restriction dans les possibilités de réduction (chap. 5) d'une machine $R[X, Y, Z]$ de type standard car on n'utilise à cet effet que des recouvrements $A \rightarrow Y$ sans classe vide, de sorte que $\tilde{A} = A$. Dans les problèmes d'assignement, on est naturellement conduit à des codages $A \rightarrow Y$ possédant des classes vides, par exemple lorsque $A = \mathbf{B}_n$ et $|Y| < 2^n$ (tab. 6.26). Ceci entraîne la présence de $\tilde{A} \neq A$ dans la table de la machine quotient (tab. 6.28) et diminue quelque peu la liberté d'écriture des équations finales. Mais cette notion stricte conduit aux mêmes méthodes que la notion classique en ce qui concerne la recherche d'assignements décomposables, et permet d'englober dans une théorie unitaire les machines de type standard et non standard.

6.2 DÉCOMPOSITION SÉRIE

6.2.1 Définition

Soient $F[X, U]$ et $G[X \times U, V]$ deux machines générales (§ 6.1.2), l'alphabet d'entrée de la seconde étant le produit cartésien des alphabets de la première. La machine composée $M[X, U \times V]$ selon le schéma de la figure 6.33 est appelée *composition série* de F et G , et notée simplement $M = FG$. Selon les définitions de la section 2.2, on a pour trois séquences x, u, v (de même longueur) :

$$u \times v \in (FG)(x) \iff u \in F(x) \text{ et } v \in G(x \times u). \quad (6.46)$$

On écrira dorénavant $FG(x)$ pour $(FG)(x)$. On voit qu'une machine de Mealy est un cas particulier de composition série, à savoir la composition d'une machine F séquentielle avec une machine G combinatoire. On s'intéresse dans ce chapitre au cas où F et G sont des machines séquentielles, mais quelques notions préalables s'appliquent à des machines générales quelconques.

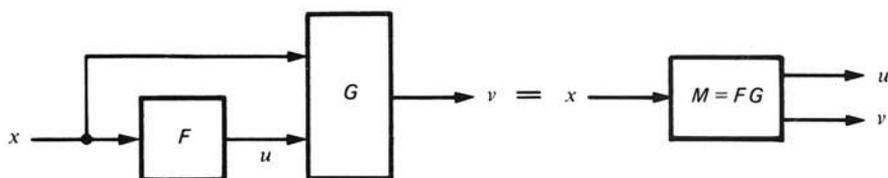


Fig. 6.33

6.2.2 Hypothèse

Dans toute cette section F, F', F'', \dots , et G, G', G'' désignent des machines avec les alphabets $[X, U]$ pour F, F', F'', \dots , et $[X \times U, V]$ pour G, G', G'', \dots . On pourra ainsi s'abstenir de mentionner ces alphabets.

6.2.3 Définition

Pour deux machines générales $M[X, Y], M'[X, Y]$, on définit la machine $(M \cup M')[X, Y]$ par

$$(M \cup M')(x) = M(x) \cup M'(x) \quad \forall x \in X^*. \quad (6.47)$$

Il est clair que si $M''[X, Y]$ est une troisième machine, alors selon la définition de la relation \subset entre machines générales (§ 6.1.21), on a

$$M \subset M'' \text{ et } M' \subset M'' \iff M \cup M' \subset M''. \quad (6.48)$$

6.2.4 Proposition

$$F \subset F' \text{ et } G \subset G' \Rightarrow FG \subset F'G'. \quad (6.49)$$

$$FG \subset F'G' \Rightarrow F \subset F'. \quad (6.50)$$

$$F(G \cup G') = FG \cup F'G'. \quad (6.51)$$

6.2.5 Démonstration

Si l'on suppose que $F \subset F'$ et $G \subset G'$, alors la relation $u \times v \in FG(x)$ implique $u \times v \in F'G'(x)$ par (6.46) appliquée à FG puis à $F'G'$.

Si l'on suppose que $FG \subset F'G'$ alors la relation $u \in F(x)$ implique $u \in F'(x)$. En effet soit $u \in F(x)$. Il existe une séquence $v \in G(x \times u)$ (§ 6.1.2), et l'on a $u \times v \in FG(x)$ par (6.46), donc $u \times v \in F'G'(x)$ par hypothèse, et enfin $u \in F'(x)$ par (6.46) appliquée à $F'G'$.

On a $G \subset G \cup G'$ donc $FG \subset F(G \cup G')$ par (6.49). De même $FG' \subset F(G \cup G')$. Selon (6.48) il vient $FG \cup FG' \subset F(G \cup G')$. Montrons l'inclusion inverse. Soit $u \times v \in F(G \cup G')(x)$. On a $u \in F(x)$ et $v \in G(x \times u)$ ou $v \in G'(x \times u)$, donc $u \times v \in FG(x)$ ou $u \times v \in FG'(x)$.

6.2.6 Définition : division

Soit $M[X, U \times V]$ une machine générale. On dit que F *divise* M s'il existe une machine G telle que $FG \subset M$. On dit que F *divise exactement* M s'il existe une machine G telle que $FG = M$.

On dit que M est *décomposable* s'il existe une machine F qui divise M , et que M est *exactement décomposable* s'il existe une machine F qui divise exactement M .

Si F divise M , on désigne par M/F la réunion de toutes les machines G telles que $FG \subset M$.

6.2.7 Proposition

Si F divise $M[X, U \times V]$, la machine M/F a les propriétés suivantes :

$$F(M/F) \subset M \quad (6.52)$$

$$FG \subset M \Rightarrow G \subset M/F. \quad (6.53)$$

En d'autres termes, M/F est la "plus grande" machine G telle que $FG \subset M$.

6.2.8 Démonstration

Par définition, M/F est la réunion des G telles que $FG \subset M$. Donc (6.53) est immédiate. En vertu de (6.51), $F(M/F)$ est la réunion des machines $FG \subset M$, et (6.52) découle de ceci en vertu de (6.48).

6.2.9 Proposition

Il existe au plus une machine F qui divise exactement M . En effet si l'on a $FG = M$ et $F'G' = M$, il vient $F = F'$ par (6.50).

6.2.10 Proposition

Si F divise $M[X, U \times V]$ et si $M \subset M'[X, U \times V]$, alors F divise M' et $F/M \subset F/M'$.

En effet supposons que F divise M et que $M \subset M'$. On a $F(M/F) \subset M'$ par (6.52), d'où $M/F \subset M'/F$ en appliquant (6.53) à M' .

6.2.11 Proposition

Si F et G sont deux machines séquentielles, alors FG est une machine séquentielle, et l'on a

$$((u \times v) \cdot x)_{FG} = (u \cdot x)_F \times (v \cdot (x \times u))_G. \tag{6.54}$$

On écrira $u \times v \cdot x$ pour $(u \times v) \cdot x$ et $v \cdot x \times u$ pour $v \cdot (x \times u)$. La formule (6.54) donne la façon de construire la table de transition de FG à partir de celles de F et de G . Le principe de cette construction est illustré par la figure 6.34, qui ne fait que traduire (6.54).

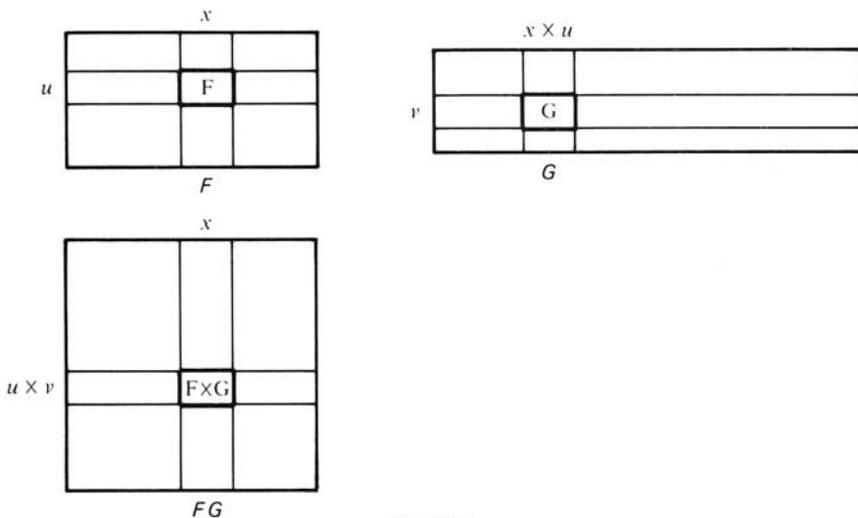


Fig. 6.34

6.2.12 Démonstration

Supposons que F et G soient deux machines séquentielles. Il convient de rappeler ce que cela signifie exactement selon le paragraphe 2.4.4. Premièrement si $lg(x) = 1$ on a $F(x) = U$ et $G(x \times u) = V$ quel que soit $u \in U$, conformément à (2.20). Selon (6.46) il vient

$$u \times v \in FG(x) \iff u \in U \text{ et } v \in V, \text{ donc} \\ FG(x) = U \times V \text{ pour tout } x \in X. \tag{6.55}$$

Supposons que $lg(x) = n > 1$. Alors on peut expliciter la relation (6.46) comme suit, en écrivant x_k, u_k, v_k pour $x(k), u(k), v(k)$:

$$u \times v \in FG(x) \iff u_{k+1} \in (u_k \cdot x_k)_F \text{ et } v_{k+1} \in (v_k \cdot x_k \times u_k)_G \quad (k=1, \dots, n-1) \\ \iff u_{k+1} \times v_{k+1} \in (u_k \cdot x_k)_F \times (v_k \cdot x_k \times u_k)_G \quad (k=1, \dots, n-1) \\ \iff (u \times v)_{k+1} \in \underbrace{(u_k \cdot x_k)_F \times (v_k \cdot x_k \times u_k)_G}_{M(u_k, v_k, x_k)} \quad (k=1, \dots, n-1).$$

L'ensemble désigné par $M(u_k, v_k, x_k)$ n'est pas vide, étant le produit cartésien de deux ensembles non vides par hypothèse. Ceci montre que FG est une machine séquentielle et que sa table de transition est définie par la formule (6.54).

6.2.13 Exemple

Les tableaux 6.35, 6.36, 6.37 illustrent la composition de deux machines séquentielles FG , avec $X = \{0,1\}$, $U = \{p,q,r\}$, $V = \{\alpha,\beta\}$. Le lecteur vérifiera que la table de FG est conforme au principe de la figure 6.34, c'est-à-dire à (6.54). Les tables sont découpées en blocs, et la figure 6.38 indique comment la table de FG peut se construire par blocs.

	x		
	0	1	
p	p	q	F
q	r	U	
r	q	r	
	u		

Tableau 6.35

	$x \times u$						
	$0 \times p$	$1 \times p$	$0 \times q$	$1 \times q$	$0 \times r$	$1 \times r$	
α	V	α	β	α	α	V	G
β	β	α	α	V	V	β	
	v						

Tableau 6.36

	x		
	0	1	
$p \times \alpha$	$p \times V$	$q \times \alpha$	
$p \times \beta$	$p \times \beta$	$q \times \alpha$	
$q \times \alpha$	$r \times \beta$	$U \times \alpha$	
$q \times \beta$	$r \times \alpha$	$U \times V$	
$r \times \alpha$	$q \times \alpha$	$r \times V$	
$r \times \beta$	$q \times V$	$r \times \beta$	
	$u \times v$		

Tableau 6.37

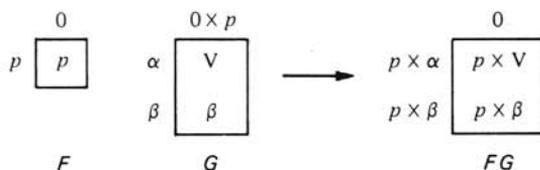


Fig. 6.38

6.2.14 Remarque

Si F et G sont deux machines séquentielles de type standard, la machine FG n'est pas nécessairement de type standard. On l'observe dans l'exemple qui précède. La machine FG serait de type standard si dans chaque case de son tableau se trouvait soit un élément $u \times v$, soit l'ensemble $U \times V$, c'est-à-dire un sous-ensemble trivial de $U \times V$ (§ 5.3.16). On remarque cependant que si F et G sont de type standard, alors l'ensemble $(u \times v \cdot x)_{FG}$ est toujours un sous-ensemble de type standard de $U \times V$ au sens du paragraphe 5.3.16, car le second membre de (6.54) est le produit cartésien d'un sous-ensemble trivial de U par un sous-ensemble trivial de V .

6.2.15 Proposition

Si F, G, F', G' sont des machines séquentielles, alors

$$FG \subset F'G' \iff F \subset F' \text{ et } G \subset G'. \quad (6.56)$$

Par suite

$$FG = F'G' \iff F = F' \text{ et } G = G' \quad (6.57)$$

et

$$(FG)/F = G. \quad (6.58)$$

6.2.16 Démonstration

Pour (6.56) il suffit d'établir la réciproque de (6.49). Supposons que $FG \subset F'G'$. En vertu de (6.31) et (6.54), on a

$$(u \cdot x)_F \times (v \cdot x \times u)_G \subset (u \cdot x)_{F'} \times (v \cdot x \times u)_{G'}.$$

Selon (1.34) il vient

$$(u \cdot x)_F \subset (u \cdot x)_{F'} \text{ et } (v \cdot x \times u)_G \subset (v \cdot x \times u)_{G'}$$

d'où $F \subset F'$ et $G \subset G'$. La relation (6.57) est une conséquence immédiate de (6.56).

Posons $FG = M$ et $G' = M/F = (FG)/F$. On a $G \subset G'$ par (6.53) et $FG' \subset FG = M$ par (6.52), d'où $G' \subset G$ par (6.56). Il vient donc $G' = G$, c'est-à-dire (6.58).

6.2.17 Hypothèse

Dans toute la suite de cette section on fait l'hypothèse que $F, F', F'', \dots, G, G', G'', \dots$ (Cf. § 6.2.2) sont des machines séquentielles, et que M, M', M'', \dots sont des machines séquentielles avec les alphabets $[X, U \times V]$, sauf mention contraire.

6.2.18 Proposition

Soit $\pi : U \rightarrow U \times V$ la partition canonique de $U \times V$ (§ 6.1.14). On a

$$\pi \text{ est compatible avec } M \iff \bigcap_{v \in V} \pi^{-1}(u \times v \cdot x)_M \neq \emptyset \ (\forall u, x). \quad (6.59)$$

Si π est compatible avec M , alors pour la machine quotient $M/\pi [X, U]$ on a

$$(u \cdot x)_{M/\pi} = \bigcap_{v \in V} \pi^{-1}(u \times v \cdot x)_M. \quad (6.60)$$

6.2.19 Démonstration

Il suffit d'appliquer (6.28) et (6.29) en remplaçant S par M et Y par $U \times V$. L'élément y qui parcourt $\sigma(u)$ dans (6.28) et (6.29) est remplacé par un élément $u' \times v$ parcourant $\pi(u)$. Or $u' \times v \in \pi(u)$ équivaut à $u' = u$ et $v \in V$. Donc

$$u' \times \bigcap_{v \in \pi(u)} \pi^{-1}(u' \times v \cdot x) = \bigcap_{v \in V} \pi^{-1}(u \times v \cdot x)$$

d'où (6.59) et (6.60).

6.2.20 Proposition

La partition canonique $\pi : U \rightarrow U \times V$ est compatible avec FG et l'on a

$$F = (FG)/\pi. \quad (6.61)$$

6.2.21 Démonstration

On applique (6.59) et (6.60) à $M = FG$. En vertu de (6.54) et (6.22), on a $\pi^{-1}(u \times v \cdot x)_{FG} = (u \cdot x)_F$ quel que soit v , d'où $(u \cdot x)_{M/\pi} = (u \cdot x)_F$.

6.2.22 Exemple

Considérons la machine $M = FG$ du tableau 6.37. Si l'on construit le tableau de M/π selon (6.60), on obtient exactement le tableau de F (tab. 6.35). Par exemple pour $u = p$ et $x = 0$, (6.60) devient

$$\begin{aligned} (p \cdot 0)_{M/\pi} &= \pi^{-1}(p \times \alpha \cdot 0)_M \cap \pi^{-1}(p \times \beta \cdot 0)_M \\ &= \pi^{-1}(p \times V) \cap \pi^{-1}(p \times \beta) \\ &= p \cap p \quad \text{par (6.22)} \\ &= p = (p \cdot 0)_F. \end{aligned}$$

6.2.23 Proposition

Si F divise M (§ 6.2.6), alors la partition canonique $\pi : U \rightarrow U \times V$ est compatible avec M et l'on a $F \subset M/\pi$.

En effet soit G telle que $FG \subset M$. La partition π est compatible avec FG (§ 6.2.20). En vertu du paragraphe 6.1.22, π est compatible avec M et l'on a $(FG)/\pi \subset M/\pi$, donc $F \subset M/\pi$ par (6.61).

6.2.24 Proposition

Pour que M soit décomposable (§ 6.2.6), il faut et il suffit que la partition canonique $\pi : U \rightarrow U \times V$ soit compatible avec M . Si cette condition est vérifiée, toute machine complètement spécifiée $F \subset M/\pi$ divise M .

6.2.25 Démonstration

La condition est nécessaire en vertu du paragraphe 6.2.23. Supposons que π soit compatible avec M , et soit F une machine complètement spécifiée telle que $F \subset M/\pi$. Il en existe évidemment une. Nous allons montrer que F divise M . Posons pour simplifier l'écriture

$$(u \cdot x)_F = u_x \quad (6.62)$$

$$(u \times v \cdot x)_M = M_{uvx} \quad (6.63)$$

On a $M_{uvx} \subset U \times V$. Posons encore, au sens du paragraphe 6.1.15,

$$G_{vXu} = M_{uvx}(u_x). \quad (6.64)$$

On a par (6.25),

$$u_x \times G_{v_x u} = \pi(u_x) \cap M_{u v x} \subset M_{u v x}. \tag{6.65}$$

En vertu des hypothèses faites sur F , on a $u_x \in (u \cdot x)_{M/\pi}$, donc

$$u_x \in \pi^{-1}(M_{u v x}) \tag{par (6.60), (6.63)}$$

$$\pi(u_x) \cap M_{u v x} \neq \emptyset \tag{par (6.21).}$$

$$G_{v_x u} \neq \emptyset \tag{par (6.65).}$$

On peut donc définir une machine G en posant $(v \cdot x \times u)_G = G_{v_x u}$ et il vient $(u \cdot x)_F \times (v \cdot x \times u)_G \subset (u \times v \cdot x)_M$ par (6.65), donc $FG \subset M$ par (6.54). Il est clair que $G = M/F$ (§ 6.2.6, 6.2.7).

6.2.26 Exemple

Si M n'est pas de type standard, et si la partition canonique π est compatible avec M , la machine M/π ne devise pas M en général. On le voit dans l'exemple du tableau 6.39, où $X = \{0,1\}$, $U = \{p,q\}$, $V = \{\alpha, \beta\}$. La machine M/π est donnée dans le tableau 6.40. On a notamment $(p \cdot 0)_{M/\pi} = \{p, q\}$ selon (6.60), car $\pi^{-1}(p \times \alpha \cdot 0)_M = \pi^{-1}(p \times \beta \cdot 0)_M = \{p, q\}$. Quelle que soit la machine G (tab. 6.42), on aura $(M/\pi)G \not\subset M$, car pour les ensembles désignés par M, F, G dans les tableaux on aura $F \times G \not\subset M$. Par contre, comme on l'a montré ci-dessus, si l'on élimine p ou q dans F , c'est-à-dire si l'on choisit une machine complètement spécifiée $F' \subset M/\pi$ (tab. 6.41), il est possible de remplir le tableau de G de telle sorte que $F'G \subset M$. La relation (6.64) donne la façon de procéder. On aura par exemple $G = M(q) = \beta$ et alors $G \times F' \subset M$.

	0	1
$p \times \alpha$	$p \times \alpha, q \times \beta$	$q \times V$
$p \times \beta$	$U \times \beta$	$U \times \beta$
$q \times \alpha$	$q \times V$	$p \times \beta$
$q \times \beta$	$U \times \alpha$	$U \times V$

M

Tableau 6.39

	0	1
p	p, q	q
q	q	p

Tableau 6.40

	0	1
p	q	q
q	q	p

Tableau 6.41

	$0 \times p$	$1 \times p$	$0 \times q$	$1 \times q$
α	G			
β				

Tableau 6.42

6.2.27 Cas particulier

On dira qu'une machine séquentielle $M[X, U \times V]$ est de *type cartésien* si quels que soient u, v, x l'ensemble

$$M_{u v x} = (u \times v \cdot x)_M$$

est un produit cartésien

$$M_{uvx} = Q_{uvx} \times S_{uvx}, (Q_{uvx} \subset U, S_{uvx} \subset V). \quad (6.66)$$

Supposons que la partition canonique π soit compatible avec M , et soit $F = M/\pi$. Posons $F_{ux} = (u \cdot x)_F$. On a

$$F_{ux} = \bigcap_{v \in V} Q_{uvx} \quad \text{par (6.60) et (6.22)}$$

donc

$$F_{ux} \times S_{uvx} \subset M_{uvx}. \quad (6.67)$$

On peut définir une machine G en posant $(v \cdot x \times u)_G = S_{uvx}$, et l'on a $FG \subset M$ par (6.67). Il est clair que $G = M/F$ (§ 6.2.6, 6.2.7).

On voit que pour ce type de machine M , si π est compatible avec M , alors M/π divise M . Comme exemple, on peut prendre pour M le tableau 6.39 en remplaçant l'ensemble M (encadré) par $U \times \alpha$.

Notons qu'une machine M de type standard est de type cartésien puisque M_{uvx} est de la forme $u' \times v'$ ou $U \times V$.

6.2.28 Proposition fondamentale

Soient :

- $S[X, Y]$ une machine séquentielle;
- $\varphi : A \rightarrow Y$ un codage de Y ;
- $M[X, A] = S/\varphi$ (§ 6.1.24);
- $\rho : U \rightarrow A$ un recouvrement de A sans classe vide;
- $\sigma = \varphi\rho : U \rightarrow Y$ (§ 6.1.6, 6.1.8).

Alors pour deux éléments u, x quelconques, on a

$$y \in \bigcap_{u \in \sigma^{-1}(y \cdot x)_S} \sigma^{-1}(y \cdot x)_S = \bigcap_{a \in \rho^{-1}(a \cdot x)_M} \rho^{-1}(a \cdot x)_M. \quad (6.68)$$

Par suite, selon (6.28),

$$\sigma \text{ compatible avec } S \iff \rho \text{ compatible avec } M \quad (6.69)$$

et si l'une des compatibilités (6.69) est vérifiée, on a selon (6.29)

$$S/\sigma = M/\rho. \quad (6.70)$$

On verra que la proposition est fondamentale en ce qui concerne la recherche d'assignements décomposables. Elle est illustrée par la figure 6.43 symbolisant les tables de transition de $S, M, M/\rho$ lorsque l'une quelconque des compatibilités (6.69) est vérifiée.

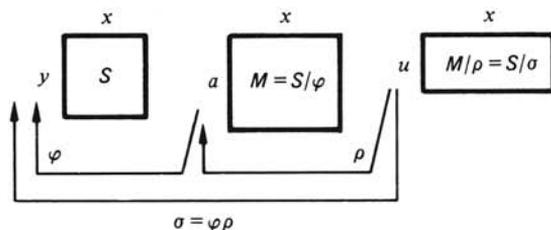


Fig. 6.43

6.2.29 Démonstration

Soient, pour l'ensemble de la démonstration, u et x deux éléments quelconques mais fixes. A gauche du signe = dans (6.68) se trouve l'intersection des ensembles

$$\sigma^{-1}(y \cdot x) \text{ tels que } y \in \sigma(u) \tag{6.71}$$

et à droite, l'intersection des ensembles

$$\rho^{-1}(a \cdot x) \text{ tels que } a \in \rho(u). \tag{6.72}$$

Remarquons d'abord que si $\varphi(a) = \emptyset$, on a $\rho^{-1}(a \cdot x) = \rho^{-1}(A) = U$ en vertu de (6.33) et du fait que ρ est supposé sans classe vide. Par suite l'intersection des ensembles (6.72) est égale à celle des ensembles

$$\rho^{-1}(a \cdot x) \text{ tels que } a \in \rho(u) \text{ et } \varphi(a) \neq \emptyset. \tag{6.73}$$

Il suffit donc de montrer que l'intersection des ensembles (6.71) est égale à celle des ensembles (6.73). A cet effet, il suffit de vérifier que chacun des ensembles (6.71) est égal à l'un des ensembles (6.73) et réciproquement, ce que l'on fait comme suit.

- Soit $y \in \sigma(u)$. Comme $\sigma = \varphi\rho$, il existe un $a \in \rho(u)$ tel que $y = \varphi(a)$ et ipso facto $\varphi(a) \neq \emptyset$. On a

$$\begin{aligned} \sigma^{-1}(y \cdot x) &= \rho^{-1} \varphi^{-1}(y \cdot x) && \text{par (6.9)} \\ &= \rho^{-1} \varphi^{-1}(\varphi(a) \cdot x) \\ &= \rho^{-1}(a \cdot x) && \text{par (6.33) et } \varphi(a) \neq \emptyset. \end{aligned}$$

- Soit $a \in \rho(u)$ tel que $\varphi(a) \neq \emptyset$, et soit $y = \varphi(a)$. Il vient $y \in \sigma(u)$ et l'on peut récrire la suite d'égalités ci-dessus.

6.2.30 Proposition

Soient $S[X, Y]$ une machine séquentielle, $\varphi : U \times V \rightarrow Y$ un codage de Y , et $M[X, U \times V] = S/\varphi$.

Si M est décomposable, le recouvrement $\sigma = \varphi\pi : U \rightarrow Y$ où π est la partition canonique $U \rightarrow U \times V$ est compatible avec S , et l'on a

$$S/\sigma = M/\pi \tag{6.74}$$

$$|\sigma(u)| \leq |V| \text{ quel que soit } u. \tag{6.75}$$

6.2.31 Démonstration

Il suffit d'appliquer la proposition fondamentale (§ 6.2.28) en posant $A = U \times V$, et $\varphi = \pi$ (π n'a pas de classe vide). Si M est décomposable, alors ρ est compatible avec M (§ 6.2.24). Le recouvrement $\sigma = \varphi\rho$ de Y est compatible avec S par (6.69), et l'on a (6.74) par (6.70). Par ailleurs, on a $\sigma(u) = \varphi(\pi(u)) = \varphi(u \times V)$, donc $|\sigma(u)| \leq |u \times V|$ par (6.14), et $|u \times V| = |V|$, d'où (6.75).

6.2.32 Construction

Supposons donnés une machine séquentielle $S[X, Y]$ et un recouvrement

$$\sigma : U \rightarrow Y \text{ compatible avec } S. \tag{6.76}$$

Choisissons un alphabet V avec la propriété (6.75). En vertu de cette propriété, il est possible de construire pour chaque élément u un codage de la classe $\sigma(u)$

$$\varphi_u : V \rightarrow \sigma(u). \tag{6.77}$$

Pour chaque couple $u \times v$ posons

$$\varphi(u \times v) = \varphi_u(v). \tag{6.78}$$

Ceci définit une correspondance

$$\varphi : U \times V \rightarrow Y$$

et il est clair que φ est un codage. De plus on a $\sigma(u) = \varphi_u(V)$ car φ_u (6.77) est un codage, donc un recouvrement de $\sigma(u)$. On a donc

$$\begin{aligned} \sigma(u) &= \varphi_u(V) \\ &= \varphi(u \times V) && \text{par (6.78)} \\ &= \varphi(\pi(u)) && \text{par (6.17)} \end{aligned}$$

π étant la partition canonique $U \rightarrow U \times V$. Donc $\sigma = \varphi\pi$. Construisons la machine quotient $M[X, U \times V] = S/\varphi$. En vertu de la proposition fondamentale (§ 6.2.28) et de (6.76), π est compatible avec M et (6.74) est vérifiée. M est décomposable (§ 6.2.24).

6.2.33 Exemple

Soit $S[X, Y]$ la machine du tableau 6.44 (Cf. tab. 6.2), avec $X = \{0, \dots, 3\}$ et $Y = \{0, \dots, 4\}$. Le recouvrement $\sigma : U \rightarrow Y$ du tableau 6.45, avec $U = \{p, q, r\}$ est compatible avec S , et la machine $F[X, U] = S/\sigma$ est donnée dans le tableau 6.46.

	0	1	3	2
0	—	1	3	4
1	4	—	—	2
2	3	4	—	—
3	4	1	1	4
4	0	2	—	—

S
Tableau 6.44

u	$\sigma(u)$
p	$\{0, 3\}$
q	$\{1, 3\}$
r	$\{2, 4\}$

Tableau 6.45

	0	1	3	2
p	r	q	q	r
q	r	q	q	r
r	p	r	U	U

$F = S/\sigma = M/\pi$
Tableau 6.46

Les classes de σ ont chacune deux éléments. On choisit donc un alphabet V à deux éléments $V = \{\alpha, \beta\}$ et l'on construit un codage $\varphi : U \times V \rightarrow Y$ (tab. 6.47) tel que $\varphi(u \times V) = \sigma(u)$. On construit ensuite la machine $M[X, U \times V] = S/\varphi$ (tab. 6.48) selon (6.33). Cette machine est un assignement décomposable de S par construction, et l'on a $M/\pi = S/\sigma = F$ (tab. 6.46). Il se trouve que M est de type cartésien (§ 6.2.27). Par suite la machine $F = M/\pi$ divise M , et l'on construit la machine $G = M/F$ (tab. 6.49) comme indiqué au paragraphe 6.2.27. Le lecteur vérifiera que $FG \subset M$.

Nous allons voir que cette méthode peut se généraliser, de façon à permettre de trouver un codage $\varphi : U \times V \times W \rightarrow Y$ tel que S/φ soit décomposable selon le schéma de la figure 6.50.

$u \times v$	$\varphi(u \times v)$
$p \times \alpha$ $p \times \beta$	0
$q \times \alpha$ $q \times \beta$	1
$r \times \alpha$ $r \times \beta$	2

Tableau 6.47

	0	1	3	2
(0) $p \times \alpha$	$U \times V$	$q \times \alpha$	$\{p, q\} \times \beta$	$r \times \beta$
(3) $p \times \beta$	$r \times \beta$	$q \times \alpha$	$q \times \alpha$	$r \times \beta$
(1) $q \times \alpha$	$r \times \beta$	$U \times V$	$U \times V$	$r \times \alpha$
(3) $q \times \beta$	$r \times \beta$	$q \times \alpha$	$q \times \alpha$	$r \times \beta$
(2) $r \times \alpha$	$\{p, q\} \times \beta$	$r \times \beta$	$U \times V$	$U \times V$
(4) $r \times \beta$	$p \times \alpha$	$r \times \alpha$	$U \times V$	$U \times V$

$$M = S/\varphi$$

Tableau 6.48

	0	1	3	2	0	1	3	2	0	1	3	2
α	V	α	β	β	β	V	V	α	β	β	V	V
β	β	α	α	β	β	α	α	β	α	α	V	V
	p				q				r			

$$G = M/F$$

Tableau 6.49

6.2.34 Proposition

Soient $F[X, U], G[X \times U, V], H[X \times U \times V, W]$ trois machines générales. On a $(FG)H = F(GH)$. (6.79)

Il suffit pour le voir, de développer les relations

$$u \times v \times w \in (FG)H(x) \text{ et } u \times v \times w \in F(GH)(x)$$

en appliquant (6.46). La machine composée $(FG)H$ est notée FGH (fig. 6.50). Dans la suite on suppose, sauf mention contraire, qu'il s'agit de machines séquentielles.

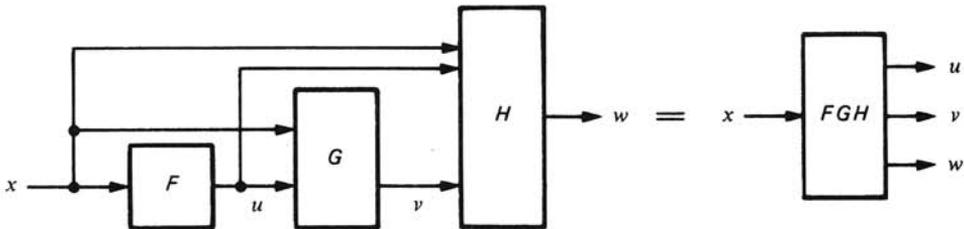


Fig. 6.50

6.2.35 Définition

On dira qu'une machine $M[X, U \times V \times W]$ est décomposable en trois niveaux s'il existe trois machines F, G, H telles que $FGH \subset M$. Il y a lieu de considérer ici les deux partitions canoniques $\pi_1 : U \rightarrow U \times V$, et $\pi_2 : U \times V \rightarrow U \times V \times W$ telles que

$$\pi_1(u) = u \times V \text{ et } \pi_2(u \times v) = u \times v \times W. \quad (6.80)$$

6.2.36 Proposition

Pour que $M[X, U \times V \times W]$ soit décomposable en trois niveaux, il faut et il suffit que π_2 soit compatible avec M et que π_1 soit compatible avec M/π_2 .

6.2.37 Démonstration

La condition est nécessaire. En effet supposons que $FGH \subset M$. En vertu des paragraphes 6.2.20 et 6.1.22, on peut écrire

$$FG = (FGH)/\pi_2 \subset M/\pi_2, \text{ puis } F = (FG)/\pi_1 \subset (M/\pi_2)/\pi_1.$$

La condition est suffisante. En effet si π_1 est compatible avec M/π_2 il existe une machine $FG \subset M/\pi_2$, et a fortiori une machine FG complètement spécifiée. Or une telle machine divise M (§ 6.2.24).

6.2.38 Construction

Soient $S[X, Y]$ une machine séquentielle (tab. 6.51), $\sigma : A \rightarrow Y$ un recouvrement compatible avec S (tab. 6.52) et $\rho : U \rightarrow A$ (tab. 6.54) un recouvrement compatible avec S/σ (tab. 6.53).

	0	1	3	2
0	—	1	3	4
1	4	—	—	2
2	3	4	—	—
3	4	1	1	4
4	0	2	—	—

$S[X, Y]$
Tableau 6.51

a	$\sigma(a)$
p	{0,3}
q	{1,3}
r	{2,4}

Tableau 6.52

	0	1	3	2
p	r	q	q	r
q	r	q	q	r
r	p	r	—	—

S/σ
Tableau 6.53

u	$\rho(u)$
α	{ p, q }
β	{ r }

Tableau 6.54

Soit V un alphabet tel que $|\rho(u)| \leq |V|$ quel que soit u , (dans l'exemple $V = \{0,1\}$) et $\psi : U \times V \rightarrow A$ (tab. 6.55) un codage tel que $\psi(u \times V) = \psi\pi_1(u) = \rho(u)$, c'est-à-dire

$$\psi\pi_1 = \rho. \tag{6.81}$$

$u \times v$	$\psi(u \times v)$
$\alpha \times 0$	p
$\alpha \times 1$	q
$\beta \times 0$	r
$\beta \times 1$	\emptyset

Tableau 6.55

$u \times v \times w$	$\varphi(u \times v \times w)$
$\alpha \times 0 \times 0$	0
$\alpha \times 0 \times 1$	3
$\alpha \times 1 \times 0$	1
$\alpha \times 1 \times 1$	3
$\beta \times 0 \times 0$	2
$\beta \times 0 \times 1$	4
$\beta \times 1 \times 0$	\emptyset
$\beta \times 1 \times 1$	\emptyset

Tableau 6.56

Soit W un ensemble ($W = \{0,1\}$ dans l'exemple) tel que $|\sigma(a)| \leq |W|$ quel que soit a , et $\varphi : U \times V \times W \rightarrow Y$ un codage (tab. 6.56) tel que $\varphi(u \times v \times W) = \varphi(\pi_2(u \times v)) = \sigma(\psi(u \times v))$, c'est-à-dire

$$\varphi\pi_2 = \sigma\psi. \tag{6.82}$$

Par exemple $\varphi(\alpha \times 0 \times \{0,1\}) = \{0,3\} = \sigma(p) = \sigma(\psi(\alpha \times 0))$.

Le codage φ permet de construire un assignement de S décomposable en trois niveaux. En effet soit

$$M[X, U \times V \times W] = S/\varphi. \tag{6.83}$$

On montrera plus loin (§ 6.2.39) que le fait que σ soit compatible avec S entraîne que $\sigma\psi$ est compatible avec S et que

$$S/(\sigma\psi) = (S/\sigma)/\psi.$$

Par suite, en vertu de (6.82) et de la proposition fondamentale (§ 6.2.28), π_2 est compatible avec M , et l'on a

$$M/\pi_2 = S/\sigma\psi = (S/\sigma)/\psi. \tag{6.84}$$

Comme ρ est compatible avec S/σ , π_1 est compatible avec $(S/\sigma)/\psi$, en vertu de (6.81) et de la proposition fondamentale (§ 6.2.28) appliquée à $S' = S/\sigma$. Par suite π_1 est compatible avec M/π_2 . En vertu du paragraphe 6.2.36, M est décomposable en trois niveaux.

De fait, si le lecteur remplace α par 0 et β par 1 dans le tableau de codage 6.56, il reconnaîtra le codage proposé dans le tableau 6.5. La construction ci-dessus montre comment on peut obtenir un tel codage.

6.2.39 Proposition

Soient $S[X, Y]$ une machine séquentielle, $\sigma : B \rightarrow Y$ un recouvrement compatible avec S , et $\psi : A \rightarrow B$ un recouvrement compatible avec S/σ . Alors le recouvrement $\sigma\psi : A \rightarrow Y$ est compatible avec S , et l'on a

$$(S/\sigma)/\psi \subset S/\sigma\psi. \tag{6.85}$$

Si de plus ψ est un codage, alors $(S/\sigma)/\psi = S/\sigma\psi$.

6.2.40 Démonstration

Il suffit d'appliquer le paragraphe 6.1.17. On a

$$\begin{aligned} (a \cdot x)_{(S/\sigma)/\psi} &= \bigcap_{b \in \psi(a)} \psi^{-1}(b \cdot x)_{S/\sigma} \\ \psi^{-1}(b \cdot x)_{S/\sigma} &= \psi^{-1} \bigcap_{y \in \sigma(b)} \sigma^{-1}(y \cdot x)_S \subset \bigcap_{y \in \sigma(b)} \psi^{-1} \sigma^{-1}(y \cdot x)_S \end{aligned} \tag{6.86}$$

selon (1.46)

donc

$$\begin{aligned} (a \cdot x)_{(S/\sigma)/\psi} &\subset \bigcap_{b \in \psi(a)} \left(\bigcap_{y \in \sigma(b)} \psi^{-1} \sigma^{-1}(y \cdot x)_S \right) = \bigcap_{y \in \sigma\psi(a)} (\sigma\psi)^{-1}(y \cdot x)_S \\ &= (a \cdot x)_{S/\sigma\psi}. \end{aligned} \tag{6.87}$$

On vérifie aisément que si $\psi : A \rightarrow B$ est un codage, alors

$$\psi^{-1}(Q \cap Q') = \psi^{-1}(Q) \cap \psi^{-1}(Q') \tag{6.88}$$

quels que soient $Q, Q' \subset B$. Dans ce cas, le signe \subset de (6.86) et (6.87) peut être remplacé par une égalité.

6.2.41 Corollaire

Si $M[X, B]$ est un assignement de $S[X, Y]$ et si $M'[X, A]$ est un assignement de M , alors M' est un assignement de S .

Il suffit d'appliquer la proposition précédente, en supposant que φ et ψ sont des codages et que $M \subset S/\varphi, M' \subset M/\psi$. Il vient $M' \subset (S/\varphi)/\psi$ en vertu du paragraphe 6.1.22, donc $M' \subset S/\varphi\psi$. Comme $\varphi\psi$ est un codage, M' est un assignement de S .

6.3 DÉCOMPOSITION PARALLÈLE

6.3.1 Définition

Soient $F[X, U]$ et $G[X, V]$ deux machines générales. On note $F \times G$ la machine définie par le schéma de la figure 6.57. On a par définition

$$u \times v \in (F \times G)(x) \iff u \in F(x) \text{ et } v \in G(x) \quad (6.89)$$

donc

$$(F \times G)(x) = F(x) \times G(x) \text{ pour toute séquence } x \in X^*. \quad (6.90)$$

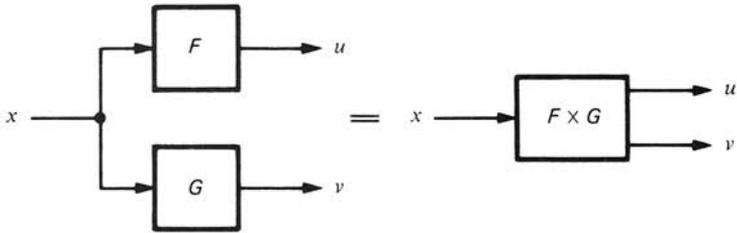


Fig. 6.57

6.3.2 Proposition

En vertu de (6.90) et du paragraphe 6.1.2, on a

$$F \times G \subset F' \times G' \iff F \subset F' \text{ et } G \subset G'. \quad (6.91)$$

6.3.3 Proposition

F divise exactement $F \times G$, au sens du paragraphe 6.2.6.

6.3.4 Démonstration

Il suffit de considérer la machine $G'[X \times U, V]$ telle que

$$G'(x \times u) = G(x). \quad (6.92)$$

En vertu de (6.89) et (6.46), on a $F \times G = FG'$.

6.3.5 Proposition

Si F et G sont deux machines séquentielles, alors $F \times G$ est une machine séquentielle et l'on a

$$(u \times v \cdot x)_{F \times G} = (u \cdot x)_F \times (v \cdot x)_G. \quad (6.93)$$

En effet il est évident que la machine G' (6.92) est séquentielle et que $(v \cdot x \times u)_{G'} = (v \cdot x)_G$. Il suffit alors d'appliquer le paragraphe 6.2.11 à FG' .

6.3.6 Définition

On dit qu'une machine séquentielle $M[X, U \times V]$ est *décomposable en parallèle* s'il existe deux machines séquentielles $F[X, U], G[X, V]$ telles que $F \times G \subset M$, c'est-à-dire telles que

$$(u \cdot x)_F \times (v \cdot x)_G \subset (u \times v \cdot x)_M. \tag{6.94}$$

6.3.7 Proposition

Si la machine séquentielle $M[X, U \times V]$ est décomposable en parallèle, alors les partitions canoniques $\pi: U \rightarrow U \times V$ et $\tau: V \rightarrow U \times V$ sont toutes deux compatibles avec M .

En effet si $F \times G \subset M$ alors F divise M (§ 6.3.3, 6.2.10); π est compatible avec M et $F \subset M/\pi$ (§ 6.2.23). Par symétrie, τ est compatible avec M et $G \subset M/\tau$.

6.3.8 Commentaire

La réciproque de la proposition précédente est fautive en général. On le voit dans l'exemple du tableau 6.58, où $X = \{x\}$, $U = V = \mathbf{B}$. Les partitions π et τ sont compatibles avec M , et les machines $F = M/\pi$ (tab. 6.59), $G = M/\tau$ (tab. 6.60) sont complètement spécifiées. Or $F \times G \not\subset M$ car

$$(0 \cdot x)_F \times (1 \cdot x)_G = 0 \times 0 \not\subset (0 \times 1 \cdot x)_M.$$

	x
0×0	$0 \times \mathbf{B}$
0×1	$\{0 \times 1, 1 \times 0\}$
1×1	$\mathbf{B} \times 0$
1×0	1×1
$u \times v$	M

Tableau 6.58

	x
0	0
1	1
u	

$$F = M/\pi$$

	x
0	1
1	0
v	

$$G = M/\tau$$

Tableau 6.60

Nous nous en tiendrons par la suite à une classe de machines séquentielles $M[X, U \times V]$ pour laquelle la réciproque de la proposition précédente est vraie, à savoir les machines de type cartésien (§ 6.2.27), auquel n'appartient précisément pas la machine ci-dessus.

6.3.9 Proposition

Soit $M[X, U \times V]$ une machine séquentielle de *type cartésien*. Si les partitions canoniques π et τ sont compatibles avec M , alors

$$M/\pi \times M/\tau \subset M. \tag{6.95}$$

Il suffit de faire le raisonnement symétrique de celui du paragraphe 6.2.27, pour la partition τ .

6.3.10 Construction

Soient $S[X, Y]$ une machine séquentielle (tab. 6.61) de *type standard*, $\sigma : U \rightarrow Y$ (tab. 6.62) et $\rho : V \rightarrow Y$ (tab. 6.63) deux recouvrements de Y tels que pour chaque couple $u \times v$ on ait

$$|\sigma(u) \cap \rho(v)| = 1. \tag{6.96}$$

On définit alors un codage $\varphi : U \times V \rightarrow Y$ (tab. 6.64) en posant

$$\varphi(u \times v) = \sigma(u) \cap \rho(v). \tag{6.97}$$

	0	1	3	2
0	—	1	3	4
1	4	—	—	2
2	3	4	—	—
3	4	1	1	4
4	0	2	—	—

$S[X, Y]$
Tableau 6.61

	u	$\sigma(u)$
0		{0,1,3}
1		{2,4}

Tableau 6.62

	v	$\rho(v)$
a		{0,4}
b		{1,2}
c		{3,4}

Tableau 6.63

$u \times v$	$\varphi(u \times v)$
$0 \times a$	0
$0 \times b$	1
$0 \times c$	3
$1 \times a$	4
$1 \times b$	2
$1 \times c$	4

Tableau 6.64

Du fait que $\varphi(u \times v) \neq \emptyset$ quels que soient u, v , la machine $M[X, U \times V] = S/\varphi$ sera de type standard, car dans (6.35), on aura $\tilde{A} = A = U \times V$.

On a $\varphi(\pi(u)) = \varphi(u \times V) = \sigma(u) \cap \rho(V) = \sigma(u) \cap Y = \sigma(u)$, et de même $\varphi(\tau(v)) = \rho(v)$, donc

$$\varphi\pi = \sigma \quad \text{et} \quad \varphi\tau = \rho. \tag{6.98}$$

Par la proposition fondamentale (§ 6.2.28), π et τ sont compatibles avec M et l'on a $M/\pi = S/\sigma, M/\tau = S/\rho$. Comme M est de type cartésien (étant de type standard), on a $S/\sigma \times S/\rho \subset M = S/\varphi$. Les machines $F[X, U] = S/\sigma$ et $G[X, V] = S/\rho$ sont données dans les tableaux 6.65 et 6.66. La machine $F \times G$ est un assignement de S .

	0	1	3	2
0	1	0	0	1
1	0	1	—	—

u $F = S/\sigma$

Tableau 6.65

	0	1	3	2
a	a	b	c	a, c
b	c	a, c	—	b
c	a	b	b	a, c

v $G = S/\rho$

Tableau 6.66

On remarque que la machine G (tab. 6.66) admet un assignement binaire décomposable en série, car le recouvrement $\mu : V' \rightarrow V$ tel que $V' = \{0, 1\}$ et

$$\mu(0) = \{a, c\}, \quad \mu(1) = \{b, c\} \tag{6.99}$$

est compatible avec G . Le codage $\psi : V' \times W \rightarrow V$ défini par le tableau 6.67 (où $V' = W = B$) est tel que $\psi(v \times w) = \mu(v)$, de sorte que G/ψ est décomposable en série. La construction du tableau de G/ψ et sa décomposition en série $H[X, V'] K[X \times V', W]$ est laissée au lecteur.

En vertu de la proposition suivante, la machine $F \times (HK)$ est un assignement de $F \times G$, donc un assignement de S . La machine S (tab. 6.61), considérée tout au

$v' \times w$	$\psi(v' \times w)$
0×0	a
0×1	c
1×0	b
1×1	c

Tableau 6.67

long de ce chapitre, admet donc un assignement binaire en forme de composition série/parallèle (fig. 6.68).

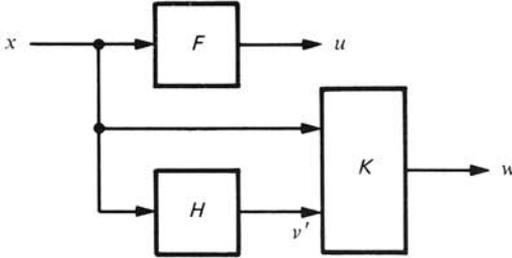


Fig. 6.68

6.3.11 Proposition

Soient $F[X, U], G[X, V]$ deux machines séquentielles. Si $M[X, A]$ est un assignement de F (resp. de G), alors $M \times G$ (resp. $F \times M$) est un assignement de $F \times G$.

Le détail de la démonstration est laissé au lecteur. En supposant que ψ soit un codage $A \rightarrow U$ et que $M \subset F/\psi$, il suffit de vérifier que l'on a $M \times G \subset (F \times G)/\varphi$ pour le codage $\varphi : A \times V \rightarrow U \times V$ tel que $\varphi(a \times v) = \psi(u) \times v$. C'est une simple application de la formule (6.33).

6.3.12 Commentaire

Considérons une machine de Mealy $R[X, Y, Z, S, F]$, de composante séquentielle $S[X, Y]$ et de composante combinatoire $F[X \times Y, Z]$. Supposons qu'on ait trouvé un recouvrement $\sigma : U \rightarrow Y$ compatible avec R , c'est-à-dire compatible avec S et avec F au sens de (5.13). Ce recouvrement peut être utilisé soit pour réduire R (en supposant $|U| \leq |Y|$), soit pour construire un assignement de S décomposable.

6.4 PARTITIONS

6.4.1 Introduction

Tout au long des sections précédentes, on a vu que les assignements décomposables d'une machine séquentielle $S[X, Y]$ peuvent être obtenus au moyen de recouvrements de Y compatibles avec S . Lorsque la table de transition de S est de petite taille, comme dans les exemples précédents ces recouvrements se découvrent facilement sans méthode de recherche spécifique. Pour les machines de plus grande dimension, les recouvrements compatibles deviennent très nombreux. Par exemple si k est un entier tel que $1 \leq k \leq |Y|$, l'ensemble de tous les sous-ensembles à k éléments de Y constitue

toujours un recouvrement compatible avec S . Le nombre des recouvrements compatibles avec S croît donc au moins exponentiellement avec $|Y|$. Il n'est donc pas possible d'envisager une méthode pratique qui les engendre tous.

Par contre l'ensemble des *partitions* de Y compatibles avec S est beaucoup plus limité, et il est possible de les engendrer toutes systématiquement. On désigne ici par le terme de partition l'*ensemble des classes* d'une partition de Y *sans classe vide*, au sens du paragraphe 6.1.13. Considérant par exemple le tableau 6.62, on dira que l'ensemble $\{\{0,1,3\}, \{2,4\}\}$ est une partition de l'ensemble $Y = \{0,1,2,3,4\}$.

Dans toute la présente section, S désigne une machine séquentielle de *type standard*, avec les alphabets $[X, Y]$. Le terme de partition, et les symboles π, π', π'', \dots désignent des partitions de Y au sens ci-dessus.

6.4.2 Définition

On dit que deux éléments y, y' sont *équivalents selon* π et l'on écrit

$$y \equiv y' (\pi) \tag{6.100}$$

lorsque y et y' appartiennent à une même classe de π .

6.4.3 Proposition

Pour trois éléments y, y', y'' quelconques de Y , on a

$$y \equiv y (\pi) \tag{6.101}$$

$$y \equiv y' (\pi) \Rightarrow y' \equiv y (\pi) \tag{6.102}$$

$$y \equiv y' (\pi) \text{ et } y' \equiv y'' (\pi) \Rightarrow y \equiv y'' (\pi). \tag{6.103}$$

Ces relations découlent immédiatement de la définition qui précède.

6.4.4 Définition

On dit de deux partitions π, π' , que π est *plus fine* que π' , et l'on écrit $\pi \leq \pi'$ si chaque classe de π est contenue dans une classe de π' .

Par exemple si $Y = \{0,1,2,3,4\}$, la partition $\pi = \{\{0,4\}, \{1\}, \{2,3\}\}$ est plus fine que $\pi' = \{\{0,1,4\}, \{2,3\}\}$. Pour simplifier l'écriture, on omettra les parenthèses extérieures $\{\}$ ainsi que les virgules entre les classes. On écrira ainsi $\pi = \{0,4\}\{1\}\{2,3\}$.

6.4.5 Propositions

Pour trois partitions quelconques π, π', π'' , on a

$$\pi \leq \pi' \iff (y \equiv y' (\pi) \Rightarrow y \equiv y' (\pi')) \tag{6.104}$$

quels que soient y, y')

$$\pi \leq \pi \tag{6.105}$$

$$\pi \leq \pi' \text{ et } \pi' \leq \pi \Rightarrow \pi = \pi' \tag{6.106}$$

$$\pi \leq \pi' \text{ et } \pi' \leq \pi'' \Rightarrow \pi \leq \pi''. \tag{6.107}$$

Ces relations sont évidentes.

6.4.6 Définition

On appelle *conjonction* de deux partitions π, π' et l'on note $\pi \wedge \pi'$ la partition formée par toutes les intersections non vides d'une classe de π et d'une classe de π' .

Par exemple si $Y = \{0, \dots, 4\}$, $\pi = \{0,1,2\} \{3,4\}$, et $\pi' = \{0,1\} \{2,3,4\}$, alors $\pi \wedge \pi' = \{0,1\} \{2\} \{3,4\}$.

La conjonction $\pi \wedge \pi'$ peut être définie par la relation

$$y \equiv y'(\pi \wedge \pi') \iff y \equiv y'(\pi) \text{ et } y \equiv y'(\pi'). \tag{6.108}$$

6.4.7 Définitions

Pour un couple de partitions (π, π') , appelons *chaîne d'éléments* de Y selon (π, π') toute suite y_1, \dots, y_n d'éléments tous distincts telle que, si $n > 1$,

$$y_i \equiv y_{i+1}(\pi) \text{ ou } y_i \equiv y_{i+1}(\pi') \quad (i = 1, \dots, n-1). \tag{6.109}$$

Par exemple soient

$$\begin{aligned} Y &= \{1, 2, \dots, 8\} \\ \pi &= \{1, 2\} \{3, 4\} \{5, 6\} \{7, 8\} \\ \pi' &= \{1\} \{2, 3\} \{4\} \{5\} \{6, 7\} \{8\}. \end{aligned}$$

On a $1 \equiv 2(\pi); 2 \equiv 3(\pi'); 3 \equiv 4(\pi)$

donc 1, 2, 3, 4 est une chaîne selon (π, π') .

Une chaîne selon (π, π') est dite maximale, s'il est impossible de l'allonger. Par exemple la chaîne 1, 2, 3, 4 ci-dessus est maximale.

Il est clair que l'ensemble des chaînes maximales selon (π, π') forme une partition de Y . Cette partition est appelée *disjonction* du couple (π, π') et notée $\pi \vee \pi'$. On a donc

$$y \equiv y'(\pi \vee \pi') \iff \text{il existe une chaîne } y_1, \dots, y_n \tag{6.110}$$

selon (π, π') telle que $y_1 = y$ et $y_n = y'$.

Dans l'exemple ci-dessus, $\pi \vee \pi' = \{1, 2, 3, 4\} \{5, 6, 7, 8\}$.

6.4.8 Partitions triviales

L'ensemble des sous-ensembles à un élément de Y est appelée la partition triviale π_\emptyset . La partition dont la seule classe est Y est appelée la partition triviale π_Y .

6.4.9 Proposition

Pour trois partitions π, π', π'' quelconques, on a

$$\left. \begin{aligned} \pi \vee \pi' &= \pi' \vee \pi \\ \pi \wedge \pi' &= \pi' \wedge \pi \end{aligned} \right\} \tag{6.111}$$

$$\left. \begin{aligned} \pi \leq \pi \vee \pi' \text{ et } \pi' \leq \pi \vee \pi' \\ \pi \leq \pi'' \text{ et } \pi' \leq \pi'' \Rightarrow \pi \vee \pi' \leq \pi'' \end{aligned} \right\} \tag{6.112}$$

$$\left. \begin{array}{l} \pi \wedge \pi' \leq \pi \text{ et } \pi \wedge \pi' \leq \pi' \\ \pi'' \leq \pi \text{ et } \pi'' \leq \pi' \Rightarrow \pi'' \leq \pi \wedge \pi' \end{array} \right\} \quad (6.113)$$

$$\left. \begin{array}{l} \pi_{\emptyset} \leq \pi \\ \pi \leq \pi_Y \end{array} \right\} \quad (6.114)$$

Ces formules découlent immédiatement des définitions qui précèdent. Les formules (6.112) et (6.113) s'expriment en disant que $\pi \vee \pi'$ est la borne supérieure et $\pi \wedge \pi'$ la borne inférieure du couple π, π' .

Les formules ci-dessus, jointes aux formules (6.105), (6.106), (6.107) se résument en disant que l'ensemble des partitions de Y constitue un treillis.

On en déduit les formules suivantes :

$$\left. \begin{array}{l} \pi \vee \pi = \pi \\ \pi \wedge \pi = \pi \end{array} \right\} \quad (6.115)$$

$$\left. \begin{array}{l} (\pi \vee \pi') \vee \pi'' = \pi \vee (\pi' \vee \pi'') \\ (\pi \wedge \pi') \wedge \pi'' = \pi \wedge (\pi' \wedge \pi'') \end{array} \right\} \quad (6.116)$$

$$\left. \begin{array}{l} \pi \leq \pi' \iff \pi \vee \pi' = \pi' \\ \pi \leq \pi' \iff \pi \wedge \pi' = \pi \end{array} \right\} \quad (6.117)$$

$$\left. \begin{array}{l} \pi \vee \pi_{\emptyset} = \pi \\ \pi \wedge \pi_Y = \pi \end{array} \right\} \quad (6.118)$$

$$\left. \begin{array}{l} \pi \vee \pi_Y = \pi_Y \\ \pi \wedge \pi_{\emptyset} = \pi_{\emptyset} \end{array} \right\} \quad (6.119)$$

6.4.10 Définitions

Pour tout sous-ensemble $K \subset Y$, on désigne par π_K la partition la plus fine telle que K soit contenu dans une classe de π . D'où les notations π_{\emptyset} et π_Y pour les partitions triviales. Si K n'est pas un sous-ensemble trivial de Y , les classes de π_K sont K et tous les ensembles $\{y\}$ à un élément $y \notin K$. Si $K = \{y, y', \dots\}$ on écrira $\pi_K = \pi_{y, y', \dots}$. Si $K = \{y_1, \dots, y_k\}$ on écrira $\pi_K = \pi_{1, \dots, k}$ de sorte que si $Y = \{y_1, \dots, y_n\}$, $\pi_{1, \dots, k} = \{y_1, \dots, y_k\} \{y_{k+1}\} \dots \{y_n\}$.

Les partitions de la forme $\pi_{y, y'}$ où y et y' sont deux éléments de Y distincts, sont appelées *partitions minimales*, en ce sens que $\pi < \pi_{y, y'} \Rightarrow \pi = \pi_{\emptyset}$.

6.4.11 Proposition

Quels que soient y, y' on a

$$y \equiv y'(\pi_{y, y'}) \quad (6.120)$$

Pour toute partition π , on a

$$y \equiv y'(\pi) \iff \pi_{y, y'} \leq \pi. \quad (6.121)$$

Pour toute partition $\pi \neq \pi_\emptyset$, on a

$$\pi = \bigvee \pi_{y, y'} \quad (y \equiv y'(\pi) \text{ et } y \neq y'). \quad (6.122)$$

Cette dernière relation signifie que *toute partition $\pi \neq \pi_\emptyset$ est une disjonction de partitions minimales.*

Les relations (6.120), (6.121) sont évidentes. Désignons par π' le second membre de (6.122). En vertu de (6.121), chaque terme $\pi_{y, y'}$ de cette disjonction est $\leq \pi$, donc $\pi' \leq \pi$. Réciproquement on a $\pi \leq \pi'$. En effet supposons que $y \equiv y'(\pi)$. Si $y = y'$ on a trivialement $y \equiv y'(\pi')$. Si $y \neq y'$, on a $y \equiv y'(\pi')$ par (6.120) et (6.112).

6.4.12 Rappel

Rappelons que $S[X, Y]$ étant une machine de type standard, la compatibilité d'une partition π avec S peut s'exprimer comme suit (§ 5.2.4, 5.2.7):

$$\begin{aligned} &\text{Pour toute classe } C \in \pi \text{ et pour tout } x \in X \\ &\text{il existe une classe } C' \in \pi \text{ telle que } C + x \subset C'. \end{aligned} \quad (6.123)$$

6.4.13 Définition

On dira qu'un sous-ensemble $K \subset Y$ est *impliqué* par π (selon S) si K est de la forme $C + x$ ($C \in \pi, x \in X$), et si K n'est pas contenu dans une classe de π . On désignera par $\mathbf{Imp}(\pi)$ l'ensemble des ensembles $K \subset Y$ impliqués par π . On a donc par (6.123):

$$\pi \text{ compatible avec } S \iff \mathbf{Imp}(\pi) = \emptyset. \quad (6.124)$$

6.4.14 Proposition

Supposons que $K \in \mathbf{Imp}(\pi), \pi \leq \pi'$, et π' soit compatible avec S . Alors $\pi \vee \pi_K \leq \pi'$. Par suite

$$\begin{aligned} &\pi \leq \pi' \text{ et } \pi' \text{ compatible avec } S \text{ impliquent} \\ &\pi \vee \left(\bigvee_{K \in \mathbf{Imp}(\pi)} \pi_K \right) \leq \pi'. \end{aligned} \quad (6.125)$$

6.4.15 Démonstration

Soient $C \in \pi$ et $x \in X$ tels que $K = C + x$. Comme $\pi \leq \pi'$, il existe une classe $C' \in \pi'$ telle que $C \subset C'$, donc $K \subset C' + x$. Comme π' est compatible avec S , K est contenu dans une classe de π' , et ceci entraîne évidemment $\pi_K \leq \pi'$, d'où $\pi \vee \pi_K \leq \pi'$ par (6.112).

6.4.16 Construction

Etant donné une partition π , on lui associe une partition notée π^S , en effectuant l'algorithme suivant (fig. 6.69):

Tant que $\text{Imp}(\pi) \neq \emptyset$ remplacer π par

$$\pi \vee \left(\bigvee_{K \in \text{Imp}(\pi)} \pi_K \right). \quad (6.126)$$

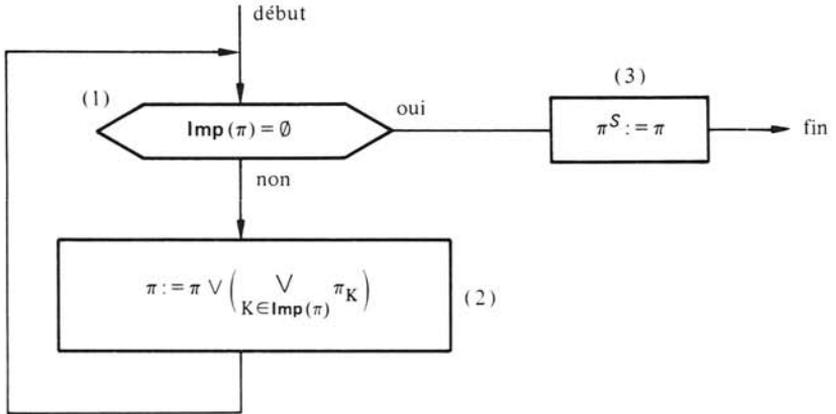


Fig. 6.69

6.4.17 Exemple

Soient S la machine du tableau 6.70, et

$$\pi = \{1,2\} \{0\} \{3\} \{4\}.$$

Déterminons π^S selon l'algorithme de la figure 6.69, dont les instructions sont numérotées (1), (2), (3). Le déroulement du processus est le suivant.

(1) Non. $\text{Imp}(\pi) = \{3,4\}$

(2) $\pi := \pi \vee \pi_{3,4} = \{0\}\{1,2\}\{3,4\}$

(1) Non. $\text{Imp}(\pi) = \{0,4\}$

(2) $\pi := \pi \vee \pi_{0,4} = \{0,3,4\}\{1,2\}$

(1) Oui.

(3) $\pi^S = \{0,3,4\}\{1,2\}$.

	0	1
0	—	1
1	4	—
2	3	4
3	4	1
4	0	2

Tableau 6.70

6.4.18 Proposition

Pour deux partitions π, π' quelconques on a les propriétés suivantes :

$$\pi^S \text{ est compatible avec } S \quad (6.127)$$

$$\pi \leq \pi^S \tag{6.128}$$

$$\pi \leq \pi' \text{ et } \pi' \text{ compatible avec } S \Rightarrow \pi^S \leq \pi' \tag{6.129}$$

$$\pi \text{ compatible avec } S \iff \pi^S = \pi \tag{6.130}$$

$$(\pi^S)^S = \pi^S \tag{6.131}$$

$$\pi \leq \pi' \Rightarrow \pi^S \leq \pi'^S \tag{6.132}$$

$$(\pi \vee \pi')^S = (\pi^S \vee \pi'^S)^S. \tag{6.133}$$

Les trois premières propriétés se résument de la façon suivantes: π^S est la partition π' la plus fine qui soit compatible avec S , et telle que $\pi \leq \pi'$.

6.4.19 Démonstration

Les deux premières relations sont évidentes (fig. 6.69). La troisième découle de (6.125). Il est clair pour la quatrième, que si π est compatible avec S alors $\pi^S = \pi$ car $\text{Imp}(\pi) = \emptyset$ (fig. 6.69). Réciproquement, si $\pi^S = \pi$, alors π est compatible avec S par (6.127). La relation (6.131) découle immédiatement de (6.127) et (6.130). Pour (6.132), supposons que $\pi \leq \pi'$. On a $\pi' \leq \pi'^S$ par (6.128), donc $\pi \leq \pi'^S$. Comme π'^S est compatible avec S , il vient $\pi^S \leq \pi'^S$ par (6.129). Pour (6.133), on peut écrire $\pi \vee \pi' \leq \pi^S \vee \pi'^S$ par (6.128), donc $(\pi \vee \pi')^S \leq (\pi^S \vee \pi'^S)^S$ par (6.132). Inversément on peut écrire $\pi^S \leq (\pi \vee \pi')^S$ et $\pi'^S \leq (\pi \vee \pi')^S$ par (6.132), donc $\pi^S \vee \pi'^S \leq (\pi \vee \pi')^S$ par (6.113), et il vient $(\pi^S \vee \pi'^S)^S \leq (\pi \vee \pi')^S$ par (6.132) et (6.131).

6.4.20 Cas particulier

Supposons que la machine S possède la propriété suivante :

$$y + x \neq \emptyset \iff y' + x \neq \emptyset \text{ quels que soient } x, y, y' \tag{6.134}$$

Cela signifie qu'une colonne quelconque de la table de S ne contient aucun tiret ou ne contient que des tirets. C'est le cas en particulier si S est complètement spécifiée.

On montre alors que *si π et π' sont compatibles avec S , $\pi \vee \pi'$ est compatible avec S .*

En effet soit $C = \{y_1, \dots, y_n\}$ une classe de $\pi \vee \pi'$. On peut supposer que les éléments y_1, \dots, y_n sont ordonnés de façon à former une chaîne selon (π, π') au sens de (6.109). Supposons que $C + x \neq \emptyset$. On a $y_i + x \neq \emptyset$ ($i = 1, \dots, n$) en vertu de (6.134). Comme π et π' sont compatibles avec S , la suite $y_1 + x, \dots, y_n + x$ vérifie aussi (6.109), de sorte que $C + x$ est contenu dans une classe de $\pi \vee \pi'$.

Dans ce cas particulier, la formule (6.133) devient

$$(\pi \vee \pi')^S = \pi^S \vee \pi'^S \tag{6.135}$$

en vertu de (6.127) et (6.130).

6.4.21 Méthode

Il est clair que pour toute partition $\pi' \neq \pi_\emptyset$ compatible avec S il existe une partition $\pi \neq \pi_\emptyset$ telle que $\pi' = \pi^S$ (par exemple la partition $\pi = \pi'$ en vertu de (6.130)). Comme toute partition $\pi \neq \pi_\emptyset$ est une disjonction de partitions minimales, toute par-

tition $\pi' \neq \pi_\emptyset$ compatible avec S est de la forme

$$(\bigvee (\pi_{y,y'})^S)^S \tag{6.136}$$

dans le cas général (6.133), et de la forme

$$\bigvee (\pi_{y,y'})^S \tag{6.137}$$

dans le cas particulier du paragraphe précédent (6.135).

L'expression (6.136) (resp. (6.37)) indique la façon de construire toutes les partitions compatibles avec S . Si $|Y| = n > 1$ il y a $n(n-1)/2$ partitions minimales donc

$$N = 2^{n(n-1)/2} - 1 \tag{6.138}$$

expressions de la forme (6.136) (resp. 6.137)). Cependant la plupart de ces expressions représentent la même partition. Il arrive souvent même que les seules partitions non triviales soient de la forme $(\pi_{y,y'})^S$.

Le calcul des partitions $(\pi_{y,y'})^S$ s'effectue en principe selon l'algorithme de la figure 6.69. Cependant, au fur et à mesure du calcul de ces partitions, le travail se simplifie grâce à la formule suivante

$$K \in \mathbf{Imp}(\pi) \Rightarrow \pi \vee (\pi_K)^S \leq \pi^S. \tag{6.139}$$

En effet si $K \in \mathbf{Imp}(\pi)$, on a $\pi \vee \pi_K \leq \pi^S$ par (6.125), donc $\pi_K \leq \pi^S$, et $(\pi_K)^S \leq \pi^S$ par (6.131), (6.132).

Ainsi dans l'opération (2) de l'algorithme, on peut remplacer π_K par $(\pi_K)^S$ lorsqu'on connaît déjà $(\pi_K)^S$.

6.4.22 Exemple

Considérons la machine S du tableau 6.71. On détermine selon l'algorithme (fig. 6.69) la partition

$$(\pi_{1,3})^S = \{1,3\}\{2,4,6\}\{5\}. \tag{6.140}$$

Soit à déterminer $(\pi_{2,4})^S$. On observe que $\{1,3\} \in \mathbf{Imp}(\pi_{2,4})$. On a

$$\pi_{2,4} \vee (\pi_{1,3})^S = (\pi_{1,3})^S \tag{6.140}$$

donc

$$(\pi_{2,4})^S = (\pi_{1,3})^S.$$

	0	1	2
1	6	—	2
2	5	4	1
3	2	5	4
4	—	6	3
5	4	1	6
6	3	2	5

Tableau 6.71

6.4.23 Exercice

Poursuivre la détermination de toutes les partitions $(\pi_{y,y'})^S$ (tab. 6.71) et vérifier qu'il n'y a pas d'autres partitions non triviales compatibles avec S .

6.4.24 Exemple

Le codage des états secondaires dans la figure 3.19 a été construit au moyen d'une partition de $Y = \{1, \dots, 10\}$ compatible avec la composante séquentielle S de cette machine. La symétrie du graphe suggère de calculer $(\pi_{y,y'})^S$ pour deux états y, y' symétriques, et par exemple

$$(\pi_{1,3})^S = \{1,3\} \{2,4\} \{5,8\} \{6,9\} \{7,10\} = \sigma$$

On cherche à assembler deux des classes de cette partition pour avoir une partition à quatre classes, et le graphe suggère immédiatement

$$\pi = \{1,3,2,4\} \{5,8\} \{6,9\} \{7,10\}.$$

On vérifie que cette partition est bien compatible avec S , et l'on établit le tableau de codage 6.72 conformément aux principes exposés dans la section 6.2. Ce tableau permet de construire un assignement ayant la structure de la figure 6.73. Les équations (3.85) sont conformes à ce schéma.

y_1	y_2	y_3	y_4	$\varphi(y_1 y_2 y_3 y_4)$
0	0	0	0	1
0	0	0	1	3
0	0	1	0	2
0	0	1	1	4
0	1	0	0	5
0	1	0	1	8
1	1	0	0	6
1	1	0	1	9
1	0	0	0	7
1	0	0	1	10

Tableau 6.72

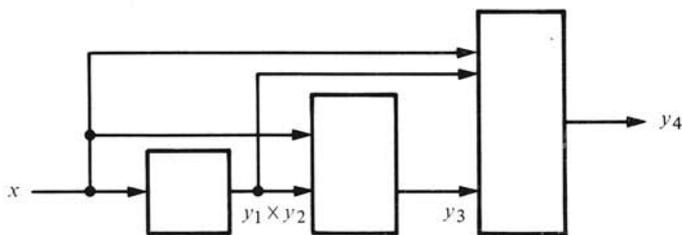


Fig. 6.73

6.4.25 Conclusion et références

Le paragraphe essentiel de ce chapitre est le paragraphe 6.2.32, car il contient le principe fondamental de construction des assignements décomposables. Une fois que l'on connaît ce principe, on sait qu'il s'agit de trouver des recouvrements adéquats, et

on les trouve, avec ou sans méthode. Nous n'accordons sur ce sujet qu'une importance secondaire aux méthodes systématiques. Par contre, les notions de compatibilité et de machine quotient, omniprésentes dans ce chapitre et le précédent sont parmi les plus importantes de la théorie des machines séquentielles.

La théorie de la décomposition est due pour l'essentiel à Hartmanis et Stearns qui lui ont consacré tout un livre [3]. On en trouve un bon résumé pratique en français dans [10].

BIBLIOGRAPHIE

- [1] P.R. HALMOS, *Introduction à la théorie des ensembles*, Gauthier-Villars, Paris 1970.
- [2] T.L. BOOTH, *Sequential Machines and Automata Theory*, Wiley, New York, 1967.
- [3] J. HARTMANIS, R.E. STEARNS, *Algebraic Structure Theory of Sequential Machines*, Prentice-Hall, Englewood Cliffs, 1966.
- [4] S. GINSBURG, *An Introduction to Mathematical Machine Theory*, Addison-Wesley, Reading, 1962.
- [5] A. GILL, *Introduction to the Theory of Finite-State Machines*, McGraw-Hill, New York, 1962.
- [6] A. GINZBURG, *Algebraic Theory of Automata*, Academic Press, New York, 1968.
- [7] E.F. MOORE (ed.), *Sequential Machines (Selected Papers)*, Addison-Wesley, Reading, 1964.
- [8] E.F. MOORE, Gedanken-Experiments on Sequential Machines, in *Automata Studies*, C.E. Shannon and J. McCarthy (eds.), Princeton University Press, Princeton, 1956.
- [9] G.H. MEALY, A Method for Synthesizing Sequential Circuits, *Bell Syst. Tech. Journal*, vol. 34, 1955, pp. 1045-1079.
- [10] J.P. PERRIN, M. DENOUE, E. DACLIN, *Systèmes Logiques*, Dunod, Paris, 1967.
- [11] K. JENSEN, N. WIRTH, *PASCAL User Manual and Report*, Second edition, Springer-Verlag, New York, 1978.
- [12] N.E. KOBRINSKII, B.A. TRAKHTENBROT, *Introduction to the Theory of Finite Automata*, North-Holland, Amsterdam, 1965.
- [13] J. FLORINE, *Automatismes à séquences et commandes numériques*, Dunod, Paris, 1969.
- [14] N. WIRTH, *Systematic Programming: An Introduction*, Prentice-Hall, Englewood Cliffs, 1973.
- [15] J.M. DUMAS, F. PRUNET, A Method for Local and Reduced Studies in Parallel Process Models, *Digital Processes*, vol. 2, 1976, pp. 99-118.
- [16] M. BLANCHARD, Rapport de la "Commission de normalisation de la représentation du cahier des charges d'un automatisme logique" (Groupe de travail AFCET "Systèmes logiques"), *Automatisme*, vol. 23, No 3-4, Mars-Avril 1978, pp. 66-83.
- [17] E. DACLIN, M. BLANCHARD, *Synthèses des systèmes logiques*, CEPADUES-EDITIONS, Toulouse, 1976.

- [18] J.E. HOPCROFT, J.D. ULLMAN, *Formal Languages and Their Relation to Automata*, Addison-Wesley, Reading, 1969.
- [19] S.C. KLEENE, Representation of Events in Nerve Nets and Finite Automata, in *Automata Studies*, C.E. Shannon and J. McCarthy (eds.), Princeton University Press, Princeton, 1956.
- [20] S. EILENBERG, *Automata, Languages and Machines*, Volume A, Academic Press, New York, 1974.
- [21] S.H. UNGER, *Asynchronous Sequential Switching Circuits*, Wiley, New York, 1969.
- [22] D.A. HUFFMAN, The Synthesis of Sequential Switching Circuits, *Journal of the Franklin Institute*, vol. 257, 1954, p. 161-190, 275-303.
- [23] W.S. MEISEL, A Note on Internal State Minimization in Incompletely Specified Sequential Networks, *IEEE Transactions on Electronic Computers*, vol. EC-16, 1967, pp. 508-509.
- [24] F.J. HILL, G.R. PETERSON, *Introduction to Switching Theory and Logical Design*, Wiley, New York, 1968.
- [25] M.C. PAULL, S.H. UNGER, Minimizing the Number of States in Incompletely Specified Switching Functions, *IRE Transactions on Electronic Computers*, vol. EC-8, 1959, pp. 356-367.
- [26] A. GRASELLI, F. LUCCIO, A Method for Minimizing the Number of Internal States in Incompletely Specified Sequential Networks, *IEEE Transactions on Electronic Computers*, vol. EC-14, 1965, pp. 350-359.
- [27] I. ALEKSANDER, F.K. HANNA, *Automata Theory: An Engineering Approach*, Crane Russak, New York, 1975.

INDEX ANALYTIQUE

Les références sont celles des pages

- Algèbre de Boole, 3
- Algèbre des expressions booléennes, 65
- Alphabet, 12
 - d'entrée, 19, 21
 - de sortie, 19, 21
- Arbre des recouvrements, 208
- Arête, 15
 - de première espèce, 56
 - de seconde espèce, 56
- Assignement, 222, 231
- Automate fini, 145
 - associé à un diagramme régulier, 147
 - complet, 152
- Bijection, 6
- Boole, 3

- Chaîne d'éléments, 253
- Chemin d'un graphe, 16
- Classe
 - de compatibilité, 189, 190
 - d'incompatibilité, 200
 - d'un recouvrement, 178, 223
 - impliquée, 201
 - première, 210
 - maximale, 210
- Codage, 225
- Compatibilité, 178, 180, 181, 228
- Complémentation, 61
- Complément d'un ensemble, 3
- Comportement asynchrone, 115
- Composition série, 235
- Condition
 - de couverture, 189
 - de fermeture, 189
 - de franchissement, 100
 - initiale, 43
- Conjonction, 61, 253
- Connexion, 27, 30
- Conservation de la longueur, 20
- Constante booléenne, 62
- Construction des sous-ensembles, 151, 155

- Correspondance, 8, 11
 - composée, 224
 - déterministe, 10
 - de transition, 38
 - fonctionnelle, 10
 - plus fine, 11
 - réciproque, 10, 224
- Couple, 6, 7
- Critères d'élimination, 192

- Décomposition
 - d'une séquence, 124
 - parallèle d'une machine séquentielle, 248
 - série d'une machine séquentielle, 235
- Délai, 49
- Déterministe, 150
- Diagramme
 - de compatibilité, 196
 - de réceptivité, 113
 - d'incompatibilité, 201
 - régulier, 135
 - régulier associé à un automate, 146
- Diminution, 187
- Disjonction, 61, 253
- Division, 236
 - exacte, 236
- Dualité, 4

- Egalité
 - de correspondances, 8
 - de fonctions, 5
 - d'ensembles, 1
 - de séquences, 12
- Élément, 1
- Élément de mémoire $\bar{\sigma}$, 24
- Ensemble, 1
 - d'arrivée, 8
 - de départ, 8
 - des parties, 3
 - fini, 2

- vide, 2
- Ensembles disjoints, 3
- Equations booléennes, 69
 - équivalentes, 70
- Equations de récurrence booléennes, 74
- Equations régulières linéaires, 131
- Equivalence, 252
- Etat
 - d'entrée, 74
 - de sortie, 74
 - primaire, 74
 - secondaire, 74
 - secondaire stable, 96
 - secondaire totalement stable, 96
 - secondaire unitaire, 107
 - superflu, 188
- Etiquette, 15, 16
- Exclusion, 210
- Expression booléenne, 61, 62
- Expressions booléennes équivalentes, 64
- Expressions régulières, 132
- Extension d'une relation, 2
- Extrémité, 15, 16

- Famille d'automates, 153
 - à graphe et ensemble initial commun, 154
- Fermé, 205
- Fermeture, 202
 - minimale, 202
- Fin de réponse, 159
- Fin de séquence, 158
- Fonction, 5
 - bijective, 6
 - booléenne, 68
 - booléenne nulle en ..., 95
 - caractéristique, 93
 - injective, 6
 - partielle, 10
 - réciproque, 6
 - représentée par une expression booléenne, 68
 - surjective, 6
- Fonctionnement, 20, 21
- Format, 21
- Franchissement, 100
- Fusionnement, 215

- Ginsburg, 219
- Graphe, 15
- Graphe de récurrence booléen, 92, 98, 100
 - réceptif, 110, 112
 - réceptif associé, 112
- Graphe de transition, 38, 45, 56, 157

- Grasselli, 219
- Hartmanis, 260
- Huffman, 219

- Implication, 194, 201, 255
- Inclusion, 1
- Inégalité booléenne, 66
- Injection, 6
- Intersection, 3
- Itération, 125

- Juxtaposition, 27

- Kleene, 173

- Langage, 124
 - élémentaire, 131
 - régulier, 131
 - régulièrement engendré, 127
 - représenté par un automate, 145
 - représenté par un diagramme régulier, 138
- Longueur d'une séquence, 11
- Luccio, 219

- Machine, 19, 21
 - binaire, 21
 - composée, 26
 - de déclenchement, 117
 - dégénérée, 54
 - d'enclenchement, 117
 - générale, 223
 - JK, 23
 - OU, 22
 - réduite, 178
 - sans entrée, 30
- Machine combinatoire, 33, 34
 - complètement spécifiée, 35
- Machine de Mealy, 51, 53
 - asynchrone, 217
 - asynchrone associée, 217
 - complètement spécifiée, 55
- Machine de Moore, 51, 215
 - complètement spécifiée, 52
- Machine quotient, 178, 182, 229
- Machine séquentielle, 37, 38
 - complètement spécifiée, 46
 - décomposable, 236
 - décomposable en parallèle, 249
 - de délai unité, 49
 - de type cartésien, 241
 - de type standard, 45
 - exactement décomposable, 236
 - linéaire au sens booléen, 106
 - uniforme, 107

- Marque, 137, 142
- Mealy, 59
- Meisel, 219
- Minterme, 67
- Moore, 59
- Mot, 11

- Norme, 109
- Noyau, 68
- N-uple, 8

- Opérateur fin de séquence, 158
- Opérations
 - de composition de machines, 26
 - régulières, 123, 126
 - *, 128
- Ordre d'un système d'équations de récurrence, 75
- Origine, 15, 16

- Parcours
 - d'un automate, 145
 - d'un diagramme régulier, 137
- Partie, 1
- Partitions, 226, 251
 - canoniques d'un produit cartésien, 227
 - minimales, 254
 - plus fines, 252
- Paull, 219
- Permutation, 28, 29, 71
- Petri, 122
- Phase, 100
- Places, 136, 137
- Préfixe, 124
- Produit
 - cartésien d'ensembles, 6, 7
 - cartésien de séquences, 14
 - de langages, 125
 - de séquences, 12
 - logique, 61
- Projection, 31

- Réalisation, 48
 - canonique d'une machine séquentielle, 50
- Réceptivité, 110
- Réciproque, 6, 10, 224
- Recouvrement, 178, 201, 223
 - irrédundant, 207
 - redondant, 207
 - trivial, 190
- Réduction, 175, 176
 - asynchrone, 218
 - de format, 26
 - minimale, 176

- Règles d'activité, 100, 114
- Relation d'appartenance, 1
- Relation d'inclusion, 1
- Réunion, 3

- Schéma de composition, 29
- Score, 208
- Séquence, 11
 - applicable, 234
 - d'entrée, 19, 21
 - des déclenchements, 117
 - des enclenchements, 117
 - de sortie, 19, 21
- Simulation, 175, 176
- Somme logique, 61
- Sommet, 15
 - initial, 145
 - terminal, 145
- Sorties indépendantes, 36
- Sous-ensemble, 1, 3
 - de type standard, 197
 - représenté par une expression booléenne, 63
 - trivial, 197
- Stable, 96, 127
- Stearns, 260
- Substitution, 72
- Surjection, 6
- Syllabe, 124
- Synthèse des machines, 156, 161
- Systèmes d'équations booléennes, 93
 - équivalents, 94

- Tableau de couverture, 208
- Table de transition, 38
- Temps réel, 84
- Terminaison, 124
- Tiret, 45
- Totalement stable, 96
- Trajet, 142
- Transformé, 8, 41
- Type standard, 45, 197, 198

- Unger, 199, 219

- Valeur de vérité, 66
- Variable, 62
 - booléenne, 62
 - d'entrée, 19, 21
 - de sortie, 19, 21
 - primaire, 52, 52
 - secondaire, 52, 53
 - tertiaire, 52, 53
- Zahnd, loi de, 269

Le Traité d'Electricité est l'œuvre collective
des membres du Département d'Electricité de l'EPFL,
assistés par quelques collaborateurs externes.
A ce volume ont collaboré plus particulièrement :

Marc Davio : critique du manuscrit
Claire-Lise Delacrausaz : secrétariat de rédaction,
composition du texte et des formules
Kurt Hofer : dessins et photographie
Allen Kilner : mise en page et montage
Daniel Mange : critique du manuscrit
Bernard Moser : critique du manuscrit
Jacques Neiryck : direction du Traité
Sylviane Pilot : montage des corrections
Renée Pittet : composition du texte et
des formules
André Stauffer : rédaction du paragraphe 3.2.20
Claude Vinckenbosch : critique du manuscrit
Ida Wegmüller : montage du lettrage
Jacques Zahnd : rédaction du volume

