Opérations sur les automates
version α 0.1 Sun 10 May 1998 19:26:58
version α 0.2 Sun 17 May 1998 17:00:00

Antoine Petit[*] [†] [‡]

10 mai 1998

Opérations sur les automates

1 Introduction

1.1 Notations, $L_\exists$

Soit $\mathcal{A} = (X, Q, T, I, F)$ un automate fini non déterministe complet (A.F.N.C. dans la suite) et $L = L(\mathcal{A})$ le langage qu'il reconnaît. Dans cette notation, X désigne un alphabet fini, Q l'ensemble fini des états, T l'ensemble des transitions, I l'ensemble des états initiaux et enfin F l'ensemble des états finals.
On note $c = q_0 a_1 q_1 \ldots a_s q_s$ un *chemin réussi* avec $q_0 \in I$, $q_s \in F$ et pour tout i de $[1, s-1]$ l'appartenance $(q_i, a_i, q_{i+1}) \in T$.
L'*étiquette* $\ell(c)$ du chemin c sera le mot $a_1 \cdots a_s$ et l'on remarquera que L peut s'écrire $L_\exists$ au sens où c'est l'ensemble des mots u tels qu'il *existe* un chemin réussi c dont l'étiquette vaut u.

Exemple 1 *Considérons l'A.F.N. de la figure (1.1). Il reconnaît bien entendu le langage $L^1 = A^*(ab)^+A^*$ formé des mots contenant ab comme facteur.*

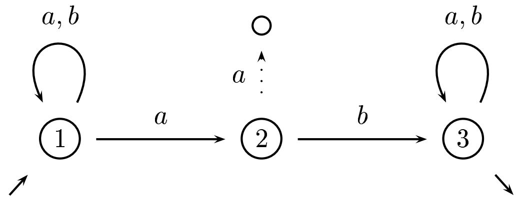

Figure 1.1: Un A.F.N. pour $A^*(ab)^+A^*$

[*] antoine.petit@lsv.ens-cachan.fr
[†] laurent.cheno@hol.fr
[‡] esperet@marie.polytechnique.fr

Déterminisons-le par la méthode des parties (en ne construisant que les états accessibles) ; nous obtenons l'A.F.D. (non minimal) $\mathcal{B}^1_\exists$ de la figure (1.2).

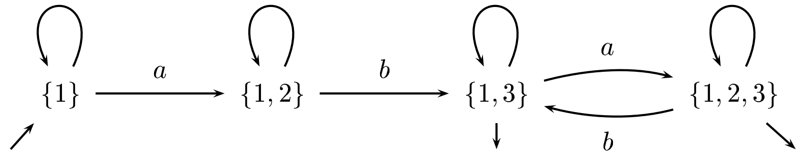

Figure 1.2: Un automate fini déterministe pour $A^*(ab)^+A^*$

Exemple 2 *Considérons maintenant l'A.F.N. de la figure (1.3), qui reconnaît l'ensemble L^2 des mots contenant la lettre a.*

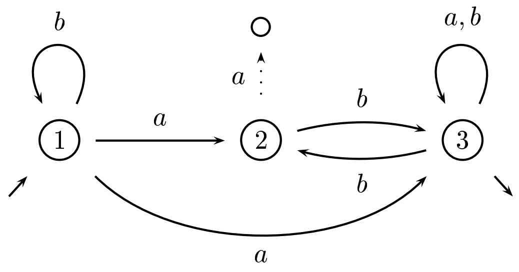

Figure 1.3: Un automate non déterministe pour $A^*(a)^+A^*$

Déterminisons le par la même méthode pour obtenir l'A.F.D. $\mathcal{B}^2_\exists$ de la figure (1.4). L'automate obtenu n'est bien sûr pas minimal, les états $\{2,3\}$ et $\{3\}$ pouvant fusionner.

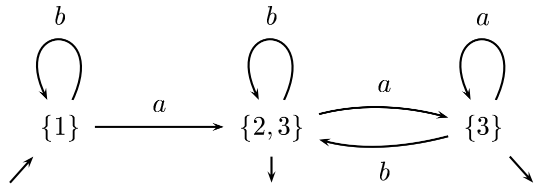

Figure 1.4: Un automate déterministe (non minimal) pour $A^*(a)^+A^*$

1.2 $L_\forall$

On peut alors essayer de changer le *il existe* en *pour tout* dans la définition de $L_\exists$, et noter $L_\forall$ l'ensemble des mots $w = a_1 \dots a_s \in L$ tels que pour tout chemin $c' = q'_0 a_1 q'_1 \cdots a_s q'_s$ (avec $q'_0 \in I$ et $(q'_i, a_i, q'_{i+1} \in T)$ pour tout i de $[1, s-1]$), on ait $q'_s \in F$.
Intuitivement, on demande que tout chemin lisant w aboutisse à un état final, et non plus qu'il existe au moins un tel chemin comme c'est le cas dans le cadre de la théorie usuelle.

REMARQUE : si l'automate de départ est déterministe, il y a bien sûr égalité entre $L_\forall$ et $L_\exists$.

Théorème 1 *Le langage $L_\forall$ est reconnaissable.*

DÉMONSTRATION : nous allons construire un automate fini déterministe (A.F.D. dans la suite) reconnaissant le langage $L_\forall$. Pour cela, déterminisons $\mathcal{A}$ par la méthode classique des parties en n'introduisant cependant que les états accessibles. On sait que cela revient à définir une loi de composition externe $\mathfrak{P}(Q)\times X \mapsto \mathfrak{P}(Q)$, notée « $\cdot$ » avec $P\cdot a = \{q \in Q \mid \exists p \in P$ avec $((p,a,q) \in T\}$. Le loi « $\cdot$ » s'étend naturellement par récurrence en une loi $(P,a) \in \mathfrak{P}(Q) \times X^*$ que l'on note encore « $\cdot$ ».
Le langage L de départ est alors reconnu par $\mathcal{B}_\exists = (X, \mathfrak{P}(Q), \cdot, I, \mathfrak{F})$ où $\mathfrak{F}$ est l'ensemble des parties de Q qui rencontrent F.
Montrons que l'automate $\mathcal{B}_\forall (= X, \mathfrak{P}(Q), \cdot, I, \mathfrak{P}(F) \setminus \emptyset)$ reconnaît $L_\forall$ (on a simplement pris comme états finals non plus toutes les parties rencontrant F mais seulement les parties non vides incluses dans F). Si un mot $w = a_1 \cdots a_s$ est reconnu par $\mathcal{B}_\forall$, alors on a $I \cdot a_1 = E_1, \ldots, E_{s-1} \cdot a_s = E_s \subset F$, et tout chemin étiqueté par w dans l'automate initial aboutit à un état final de cet automate. Réciproquement, si w est $L_\forall$, $I.w$ rencontre de toute façon F mais ne peut contenir un état q'_s qui ne serait pas dans F car il existerait un chemin étiqueté par w qui aboutirait à un état non final. □
REMARQUE : dans l'exemple (1), le langage $L^1_\forall$ est naturellement vide : à tout chemin réussi associé à w, on associe un chemin raté consistant à boucler dans l'automate initial sur l'état initial. Par contre, dans l'exemple (2), on voit qu'un A.F.D. reconnaissant $L^2_\forall$ s'obtient en éliminant la caractère « final » de la partie $\{2,3\}$ sur l'automate de la figure (1.4). On se convainc rapidement que ce nouvel automate est ici minimal.

1.3 $L_{\exists!}$

On s'intéresse désormais au langage $L_{\exists!}$ formé des mots $w \in L$ tels qu'il existe un chemin c réussi *unique* d'étiquette w. Il ne va pas être aussi simple de trouver un A.F.D. reconnaissant ce langage et ainsi de prouver de prouver :

Théorème 2 *Le langage $L_{\exists!}$ est reconnaissable.*

DÉMONSTRATION : le lecteur est invité à suivre à la main la construction décrite ci-dessous, en partant des automates (1.1) et (1.3) pour aboutir aux figures (1.5) et (1.6).
Considérons l'A.F.D. $\mathcal{A} = (X,Q,T,I,F)$ et construisons une copie $\overline{Q}$ surlignée des états. Si P est un sous ensemble de Q, on note comme dans le paragraphe précédent $P \cdot a$ l'ensemble des q tels qu'il existe p dans P avec $(p,a,q) \in T$, et l'on étend cette loi sur aux mots. Introduisons alors l'automate $(X, \mathfrak{P}(Q \sqcup \overline{Q}), \boxdot, \{I\}, \mathfrak{F})$ avec les définitions suivantes :

– $(P \sqcup \overline{R}) \boxdot a = P_1 \sqcup \overline{R_1}$ où

$$P_1 = \{q \in P \mid \exists! p \in P \text{ avec } (p,a,q) \in T\} \qquad \text{et}$$

$$R_1 = R\cdot a \cup \{q \in P \mid \exists!(p_1,p_2) \text{ distincts dans } P \text{ avec } (p_1,a,q) \in T \text{ et } (p_2,a,q) \in T\}.$$

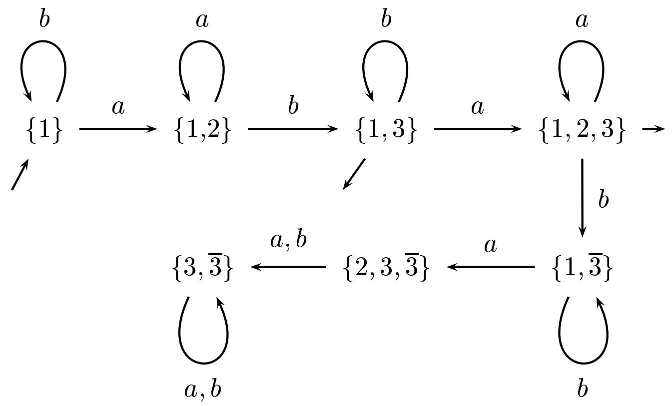

Figure 1.5: $L_{\exists !}$ (exemple 1)

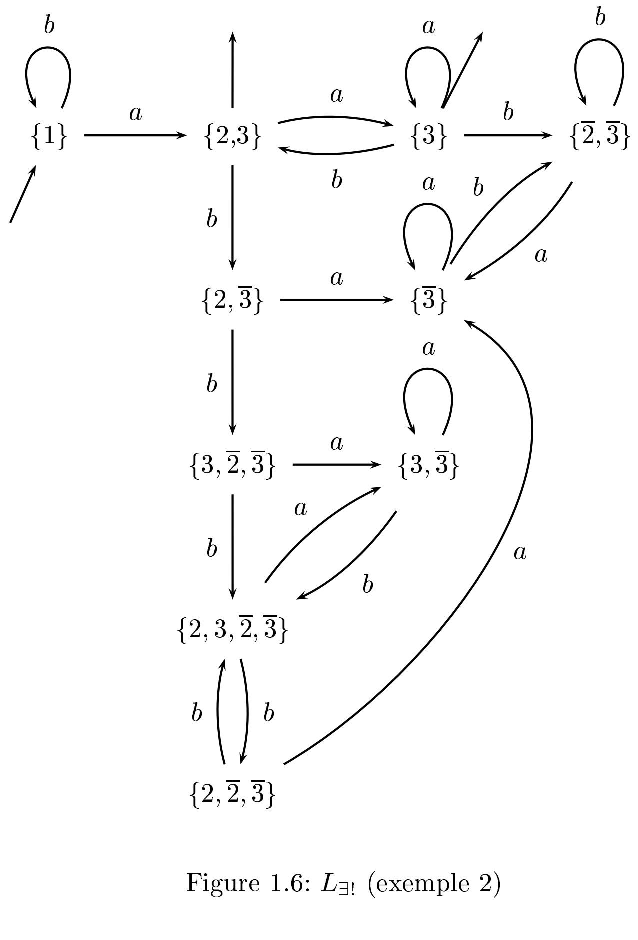

Figure 1.6: $L_{\exists !}$ (exemple 2)

– $\mathfrak{F} = \{P \sqcup \overline{R} \mid |P \cap F| = 1 \text{ et } |R \cap F| = 0\}$.

Supposons que $\{I\} \boxdot w = P \sqcup \overline{R}$ avec $P \cap F = \{f\} \in F$ et $\overline{R} \cap \overline{F} = \emptyset$. En développant w en $a_1 \cdots a_s$, on a $((P_0 = I, \emptyset), a_1, (P_1, \overline{R_1}), \dots, a_s, (P_s, \overline{R_s}))$ avec $P_s = P$ et $R_s = R$. On choisit pour q_{s-1} l'antécédent de $f \in F$ sous la lettre a_s, puis de proche en proche l'antécédent $q_{i-1} \in P_{i-1}$ de q_i sous la lettre a_i $(i = 1, \dots, s)$. Ainsi, le mot w est bien reconnu par $\mathcal{A}$. Montrons maintenant qu'il l'est de manière unique. Supposons que $c' = q'_0, a_1, q'_1, \dots, a_s, q'_s$ soit un chemin réussi dans $\mathcal{A}$ et distinct de $c = q_0, a_1, q_1, \dots, a_s, q_s$. Il n'est pas possible que pour tout i l'état q'_i soit dans P_i, et l'on peut ainsi introduire le plus petit indice $j \geq 0$ tel que $q'_j \in P_j$ mais $q'_{j+1} \notin P_{j+1}$. Alors q'_{j+1} devient surligné, et ses successeurs dans c' également. Alors $q'_s \in R_s$ et cela est absurde car q'_s est final dans l'automate de départ.
Réciproquement, supposons que $w = a_1 \cdots a_s$ soit lu de manière unique par $\mathcal{A}$, on vérifie qu'à cause de l'unicité le chemin se fait dans les P_i, puis que l'état final doit-être tel que $|P_s \cap F| = 1$ et que $R_s \cap F = \emptyset$.
REMARQUE : sur les figures (1.5) et (1.6), seule compte la première ligne : le reste pourrait sur chacun des automates être fusionné en un puits. Dans les deux cas, $\{\overline{3}\}$ est contenu dans son image sous a ou sous b, une fois cet état rencontré on sait que tous ses successeurs seront parasites.
On vérifie sur la figure (1.5) que le langage $L^1_{\exists!}$ est décrit par $b^*a^+b^+a^*$, et il est facile de montrer que c'est la réponse attendue pour le problème ed lecture unique associé à $\mathcal{A}^1$. On fait la même remarque à propos de $L^2_{\exists!}$ sur la figure (1.6) ; l'expression régulière associée est $b^*a(a^+b)^*$.

1.4 Vers une généralisation

Les problèmes $L_\forall$ et $L_{\exists!}$ ont été résolus par la construction d'automates *ad hoc*. Des questions du même type, comme « imposer au nombre de chemins réussis d'être congru à 3 modulo 7 » sont pour le moment sans réponse. Nous allons voir au chapître suivant une manière systématique de fondre ces problèmes dans le même creuset.

2 Comptage des chemins

2.1 Introduction

Soit $\mathcal{A}$ un A.F.N., p et q deux états et w un mot. On note $m_{p,q}(w)$ le nombre de chemins allant de p à q en lisant w et l'on identifie Q à $[1, n]$. On a tout de suite la relation $(u \in X^*, v \in X^*)$:

$$m_{p,q}(uv) = \sum_{r \in Q} m_{p,r}(u)\, m_{r,q}(v). \tag{1}$$

Dans la formule ci-dessus, les matrices $m(x)$ $(x \in X)$ sont connues avec l'A.F.N. $\mathcal{A}$ et l'on a de cette façon par récurrence toutes les matrices $m(w)$ $(w \in X^*)$.
On définit ainsi, pour chaque A.F.N. $\mathcal{A}$, un morphisme de monoïdes μ de X^* dans $(\mathcal{M}_n(\mathbb{N}), \times)$ par son action sur les lettres : $\mu(x)$ est la matrice des $m_{p,q}(x)$ définie ci-dessus. On pourra remplacer d'ailleurs $\mathbb{N}$ par n'importe quel anneau commutatif, les booléens ou les entiers modulaires par exemple, et même généraliser la structure encore davantage.

Théorème 3 *Si l'on note I le vecteur $(i_1, \dots, i_n)$ des états initiaux ($i_\alpha = 1 \iff \alpha \in I$) et F le vecteur $(f_1, \dots, f_n)$ ($f_\alpha = 1 \iff \alpha \in F$) des* *états finals, le nombre de chemins réussis d'étiquette w est $\gamma(w) = {}^tI\,\mu(w)\,F$.*

DÉMONSTRATION : découle directement de la formule 1. □

2.2 $L_\exists$, $L_\forall$

Si l'on prend les matrices à valeurs dans `{true=1,false=0}`, on obtient $L = L_\exists$ comme l'ensemble des mots tels que $\gamma(w) = 1$. Comme il y a un nombre fini de vecteur booléens ($2^{|Q|}$), on obtient un automate déterministe fini qui reconnaît $L_\exists$ en prenant comme états ces $2^{|Q|}$ vecteurs booléens, comme transitions sous la lettre x les multiplications par la matrice $\mu(x)$, comme états initiaux (resp. finals) les vecteurs de la base canoniques $(0, 0, \ldots, 1, 0, \ldots, 0)$ avec un « un » en position i quand $i \in I$ (resp. $i \in F$).
Cependant, si l'on veut généraliser à $L_\forall$, il vaut mieux garder les informations cumulées contenues dans les matrices et prendre lesdites matrices m pour les états d'un nouvel automate sur l'alphabet X. L'état initial est alors Id (matrice identité de taille $|Q|$), et les transitions sous la lettre x se font en multipliant la matrice m par $\mu(x)$ (à droite). Reste à choisir les états finals.
Pour le langage $L_\exists$, on garde les matrices m telles qu'il existe i dans I et f dans F avec $m_{i,f} = 1$. Pour le langage $L_\forall$, on adopte les matrices qui sont telles qu'il existe $(i_0, f_0) \in I \times F$ avec $m_{i_0,f_0} = 1$, mais que tout i dans I et tout $g \in F^c$, on ait $m_{i,g} = 0$.

2.3 $L_{\exists!}$

Commençons par une définition :

Définition 1 *s On appelle emi-anneau un ensemble A muni de deux lois de composition interne, $+$ et $*$, et deux éléments privilégiés 0 et 1, avec $(A, +, 0)$ monoïde commutatif, $(A, *, 1)$ monoïde, de telle sorte que $*$ soit distributif par rapport à $+$ et que $0 * a = a * 0 = 0$ pour tout a.*

On crée maintenant un semi-anneau à trois éléments $\{0, 1, \infty\}$ muni des deux lois (on utilise ici le caractère symétrique des deux tables) :

$+$	0	1	∞
0	0	1	∞
1	$-$	∞	∞
∞	$-$	$-$	∞

$*$	0	1	∞
0	0	0	0
1	$-$	1	∞
∞	$-$	$-$	∞

On choisit les états finals de telle sorte qu'il existe $(i_0, f_0) \in I \times F$ avec $m_{i_0,f_0} = 1$ mais $m_{i,f} = 0$ pour les couples de $I \times F$ distincts de (i_0, f_0).
Le code `Caml` qui gérerait ces deux lois pourrait être :

```
type surbool=
| Un
| Zero
| Infinity;;

(*
+/ : surbool -> surbool -> surbool = <fun>
*/ : surbool -> surbool -> surbool = <fun>
print_surbool : surbool -> int = <fun>
*)
```

```
let (prefix +/)=fun
| Zero x    -> x
| x    Zero -> x
| _    _    -> Infinity;;

let (prefix */)=fun
| Zero x    -> Zero
| x    Zero -> Zero
| Un   x    -> x
| x    Un   -> x
| _    _    -> Infinity;;
```

2.4 Problème de type $L_{|c| \bmod 7=3}$

On note $L_{|c| \bmod 7=3}$ l'ensemble des mots w de L tels que le nombre de chemins réussis reconnaissant w soit congru à 3 modulo 7.
Le calcul matriciel dans l'anneau $\mathcal{M}_{|Q|}(\mathbb{Z}/7\mathbb{Z})$ selon les même techniques que ci-dessus démontrerait que $L_{|c| \bmod 7=3}$ est reconnaissable.

2.5 Exemple

Reprenons l'automate $\mathcal{A}^2$ de la figure (1.3), pour lequel on a :

$$\mu(a) = \begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad \text{et} \qquad \mu(b) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}.$$

La « tête » de l'automate associé à $L_{\exists!}$ (il a en réalité plus de 30 états) apparaît sur la figure (2.1), sur laquelle on a surligné les états finals.

3 Reconnaissabilité par morphisme

3.1 Introduction, théorème

Définition 2 *Soit S un monoïde fini, et η un morphisme de X^* dans S. On dit que $L \subset X^*$ est un langage reconnaissable s'il existe $F \subset S$ avec $L = \eta^{-1}(F)$.*

On peut d'ailleurs se passer de F et dire que L est reconnaissable si et seulement si $L = \eta^{-1}\eta L$ (indépendamment du caractère homomorphe de η, on a toujours $\eta^{-1}\eta\eta^{-1}L = L$, et il suffit donc de prendre $F = \eta L$).

Théorème 4 *Un langage L est reconnaissable au sens précédent si et seulement si il est reconnaissable au sens usuel,* ie. *si et seulement s'il existe un A.F.N.C. $\mathcal{A} = (X, Q, \cdot, I, F)$ qui le reconnaît.*

DÉMONSTRATION : si L est reconnaissable au sens précédent, on construit l'A.F.N.C. $\mathcal{B} = (X, S, \cdot, e, \eta L)$ où e est l'élément neutre de S et où la loi « $\cdot$ » est définie par $s \cdot a = a * \eta a$. Comme η est un morphisme, on montre par récurrence que $s \cdot w = s * \eta(w)$ pour tout mot $w \in X^*$.
Soit maintenant w un mot de L, $e \cdot w = e * \eta w = \eta w \in \eta L$ et donc $w \in L_{\mathcal{B}}$. Réciproquement, si $w \in L_{\mathcal{B}}$ on a $e \cdot w = \eta w \in \eta L$, donc $w \in \eta^{-1}\eta L$; mais l'on a déjà remarqué que cet ensemble était en fait L.
Réciproquement, notons $L_{\mathcal{A}}$ le langage reconnu par l'automate complet $\mathcal{A}$, dont on note les états $0, 1, \ldots n-1$ ($n = |Q|$). Construisons le monoïde $S = \mathcal{M}_n(\{0,1\}, \times)$ (on identifie 1 à `true` et 0 à `false`). Il est fini, et l'on prolonge

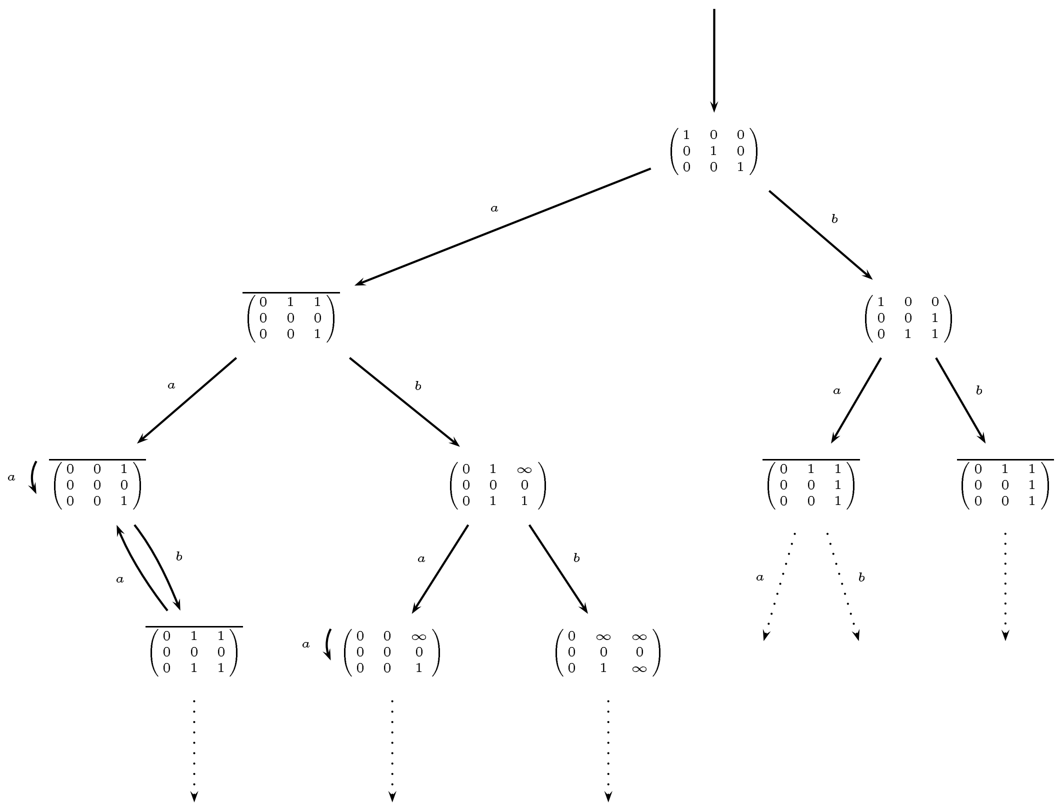

Figure 2.1: Automate et matrices

à X^* l'application définie sur les lettres de X par $\eta x = m \in \mathcal{M}_n(\{0,1\})$ avec $m.(i).(j) = 1$ si et seulement si $i.x = j$. Dans le contexte déterministe complet, chaque matrice $\eta(x)$ a sur chaque ligne un et un seul élément non nul, et cette propriété passe bien sûr à la multiplication ; alors $(\eta w)_{i,j} = 1$ si et seulement si $i \cdot w = j$ dans l'automate $\mathcal{A}$.
Maintenant $L_{\mathcal{A}} = \eta^{-1}F$, où F est l'ensemble des matrices qui ont sur la ligne i_0 (le numéro de l'état initial) leur « un » dans une position correspondant à un état final de $L_{\mathcal{A}}$. □

REMARQUE : le principal intérêt de la notion de reconnaissabilité est de s'étendre à des monoïdes M plus généraux que les X^* (voir le chapître 4 ci-dessous).

3.2 Clôture

Le théorème (4) ci-dessus montre que les langages reconnaissables sont stables par union, complément, concaténation et étoile : c'est le théorème de KLEENE. Il serait intéressant de donner des démonstrations directes de ces propriétés.
Pour le complément, c'est immédiat car $\eta^{-1}(F^c) = (\eta^{-1}F)^c$ (où A^c désigne le complémentaire de l'ensemble A).
Pour l'union, on part de $L_i = \eta_i^{-1}(F_i)$ avec S_1 et S_2 deux monoïdes finis et $F_i \subset S_i$ $(i = 1, 2)$. On définit alors η à valeur dans le monoïde produit $S_1 \times S_2$ par $\eta x = (\eta_1 x, \eta_2 x)$. Il est clair qu'alors $L_1 = \eta^{-1}(F_1 \times S_2)$ et $L_2 = \eta^{-1}(S_1 \times F_2)$. On conclut ainsi que $L_1 \cup L_2$ est bien reconnaissable.
Par contre, pour la concaténation ou l'étoile il ne semble pas y avoir de démonstration directe simple.

3.3 Automate minimal

On peut essayer de trouver dans cette théorie un analogue à l'automate minimal. On va définir sur X^* la relation compatible $\sim_L$ définie par $u \sim_L v \iff ((\forall x, y),\ xuy \in L \iff xvy \in L)$ et poser $M_0 = X^*/\sim_L$. On notera s_0 la surjection canonique de X^* sur M_0.

Définition 3 *$M_0 = X^*/\sim_L$ s'appelle monoïde syntactique.*

Théorème 5 *Si L est reconnu par (η, S, F), il existe un morphisme surjectif d'un sous monoïde S_1 de S sur le monoïde syntactique M_0.*

$$\begin{tikzcd}
X^* \arrow[rr, "\eta"] \arrow[dr, "s_0"'] & & S_1 \arrow[r, "i"] \arrow[dl, dotted, "\alpha"] & S \\
& X^*/\sim_L & &
\end{tikzcd}$$

Figure 3.1: Monoïde syntactique

DÉMONSTRATION : Supposons que L soit reconnaissable *via* (η, S) avec $L = \eta^{-1}(F)$ et construisons un morphisme α surjectif d'un sous-momoïde S_1 de S sur M_0 (voir la figure (3.1), sur laquelle i désigne l'injection canonique de S_1 dans S).

Pour cela, posons $S_1 = \eta(X^*)$ et tentons de définir $\alpha(\eta u) = s_0 u$ pour le plus général des élément de S_1. Cette démonstration ne fonctionnera que si l'on montre qu'elle est indépendante du u choisi pour représenter ηu. Mais si elle l'est, il sera immédiat de vérifier qu'elle construit en fait un morphisme surjectif. Supposons donc $\eta u = \eta u$ et $xuy \in L$. Alors $\eta xuy = \eta xvy$. Le premier est dans F, et l'on en déduit que $xvy \in \eta^{-1}F = L$, puis que $u \sim_L v$, ou encore que $s_0 u = s_0 v$. □

REMARQUE : on peut montrer la réciproque suivante : s'il existe une surjection d'un sous-monoïde S_1 de S sur M_0, alors on peut construire η et trouver F tels que (η, S, P) reconnaissent L (voir la figure (3.2)).

On construit η sur les lettres de la façon suivante : ηx est l'un quelconque des mots de $\alpha^{-1}(s_0 x)$, soit θ_x. On a alors $L = \eta^{-1}(\alpha^{-1}(s_0 L))$. Une inclusion est triviale, et réciproquement si $\alpha(\eta z) = s_0 w$ pour un certain w de L, alors en posant $z = a_1 \cdots a_s$, on a successivement $\eta z = \theta_{a_1} \cdots \theta_{a_s}$, puis $\alpha \eta z = s_0(a_1 \cdots a_s) = s_0 w$. Alors w et z sont conjugués, w est dans L donc z l'est également (prendre $x = y = \varepsilon$). □

$$\begin{tikzcd}
X^* \arrow[rr, dotted, "\eta"] \arrow[dr, "s_0"'] & & S_1 \arrow[r, "i"] \arrow[dl, "\alpha"] & S \\
& X^*/\sim_L & &
\end{tikzcd}$$

Figure 3.2: Monoïde syntactique (réciproque)

4 Rationnel *versus* reconnaissable

4.1 Langages reconnaissables sur un monoïde quelconque

On peut se demander ce qui va se passer si l'on remplace A^* par un monoïde M quelconque.

On définit les langages reconnaissables par morphismes : L est reconnaissable s'il existe un morphisme η du monoïde M dans un monoïde fini S et une partie F de S tels que $L = \eta^{-1}(F)$.

On note encore $\mathrm{Rec}(M)$ l'ensemble des langages de M reconnaissables (par morphisme).

Il n'y a pas de raison de changer la définition des langages rationnels : $\mathrm{Rat}(M)$ est la plus petite partie de $\mathcal{P}(M)$ contenant les singletons et stable par les opérations d'union, de concaténation et d'étoile.

La différence majeure avec la situation habituelle ($M = A^*$) est qu'on n'a plus $\mathrm{Rat}(M) = \mathrm{Rec}(M)$, même si McNight a démontré, dans le cas où M est finiment engendré, que $\mathrm{Rec}(M) \subset \mathrm{Rat}(M)$.

4.2 L'exemple des monoïdes de trace

Les automates finis sur l'alphabet A peuvent décrire des processus, chaque caractère codant une action.

Nous voulons ici modéliser des processus concurrents. A est encore l'alphabet des actions, et on introduit une relation $I \subset A \times A$ d'indépendance, qui est naturellement irréflexive et symétrique.
Dire, par exemple, que $(a, b) \in I$, c'est dire que les actions correspondantes peuvent être exécutées dans n'importe quel ordre.
Notons $\sim$ la relation sur $A^\star$ définie par $uabv \sim ubav$ pour tous mots u et v et tout couple $(a, b) \in I$ d'actions indépendantes, et notons $\sim_I$ la relation obtenue par clôture transitive.
Par exemple, si $A = a, b, c, d$ et si $I = (a, b), (b, a), (a, d), (d, a)$, on aura $abbbccbdcad \sim_I bbabccbdcda$.
Le monoïde quotient $M = A^\star / \sim_I$ est appelé *monoïde de trace* engendré par A et I.
Notons π la surjection canonique $\pi : A^\star \to M$. Un langage $L \subset A^\star$ sera dit I-clos si $L = \pi^{-1}\pi L$.
Nous sommes ici dans le cas où $\mathrm{Rec}(M) \neq \mathrm{Rat}(M)$. Nous laissons au lecteur le soin de démontrer cette assertion, par exemple en reprenant le sujet *Mathématiques de l'informatique* de l'agrégation1998, ou, tout simplement, en réfléchissant au cas de deux lettres indépendantes $a \sim b$: $L' = (\pi(a)\pi(b))^\star \subset M$ est rationnel et pourtant $\pi^{-1}(L')$ n'est pas reconnaissable, puisqu'il s'agit de l'ensemble des mots sur A ayant autant de a que de b.

5 Et si on changeait de semi-anneau?

5.1 Valuation des transitions

semi-anneau Nous appelons ici *semi-anneau* une structure $(K, \oplus, \otimes)$ pour laquelle $(K, \oplus)$ est un monoïde commutatif d'élément neutre noté 0, $(K, \otimes)$ est un monoïde (non nécessairement commutatif) d'élément neutre noté 1, avec la distributivité habituelle, et où enfin 0 est absorbant pour la loi $\otimes$, à savoir : $0 \otimes x = x \otimes 0 = 0$.

poids d'une transition, d'un chemin Nous associons maintenant une valeur dans K à chaque transition d'un automate $\mathcal{A} = (Q, T, I, F)$.
Plus précisément, on introduit trois applications : $\alpha : Q \to K$ (poids d'entrée des états) ; $\beta : Q \to K$ (poids de sortie des états) ; $\gamma : T \to K$ (poids des transitions).
On associe alors à tout chemin $c = q_0 a_1 q_1 \ldots a_n q_n$ (où, comme d'habitude, on a pour $1 \leqslant 2i \leqslant 2n$ $(q_{i-1} a_i q_i) \in T$) un élément de K qu'on appellera son poids, défini par :

$$\omega(c) = \alpha(q_0) \otimes \gamma(q_0, a_1, q_1) \otimes \gamma(q_1, a_2, q_2) \otimes \cdots \otimes \gamma(q_{n-1} a_n q_n) \otimes \beta(q_n).$$

En pratique, on posera généralement $\alpha(q) = 0$ si $q \notin I$ et $\alpha(q) = 1$ si $q \in I$, et, de façon analogue : $\beta(q) = 0$ si $q \notin F$ et $\beta(q) = 1$ si $q \in F$. Ceci permettra de ne considérer que les chemins réussis.

poids d'un mot On étend la notation ω en notant de la même façon pour tout mot $u \in A^\star$ $\omega(u)$ la somme des poids des chemins étiquetés par u :

$$\omega(u) = \bigoplus_{\ell(c) = u} \omega(c).$$

5.2 Quelques exemples

retour à $L_\exists$ On pourra décider de l'existence d'un chemin en choisissant $(K, \oplus, \otimes) = (0, 1, \varepsilon, \wedge)$ et en utilisant les conventions indiquées ci-dessus pour α et β, γ restant constante égale à 1.

compter les chemins On comptera les chemins réussis en choisissant $(K, \oplus, \otimes) = (\mathbb{N}, +, \times)$ et en posant $\gamma(p, a, q) = 1$ pour toute transition de l'automate.

coût d'une exécution Les automates peuvent permettre l'étude des processus. On pourra mesurer le coût maximal de l'exécution d'un processus donné en utilisant $(K, \oplus, \otimes) = (\mathbb{R} \cup -\infty, \max, +)$, la fonction γ associant à chaque transition le coût de l'action correspondante.
Une excellente référence sur ce sujet est `http://www.amadeux.inria.fr/ăgaubert/`.

5.3 L'exemple des algèbres « semi-tropicales »[1]

Considérons le semi-anneau $(K, \oplus, \otimes) = (\mathbb{N} \cup + \leqslant 0, \min, +)$. Différentes définitions de la fonction γ permettront alors d'étudier différents problèmes.
Décrivons rapidement deux applications de ces algèbres semi-tropicales.

taux de non-déterminisme d'un automate On décide ici, pour toute transition $(p, a, q) \in T$, de poser $\gamma(p, a, q) = 1$ s'il existe $q'ot = q$ tel qu'on ait aussi $(p, a, q') \in T$ et $\gamma(p, a, q) = 0$ sinon.
$\omega(u)$ mesure alors le nombre **minimal** de choix à faire — dans un automate non déterministe — pour accepter le mot u. On lira avec profit à ce sujet le livre *Mots*, de Lothaire, chez Hermès.

le problème de la puissance finie Soit $L \subset A^\star$ un langage reconnaissable. On sait qu'on note $L^\star = \bigcup_{n \leqslant \infty} L^n$.
On dira qu'un langage L a la propriété *de la puissance finie* s'il existe un entier p tel que $L^\star = \bigcup_{n=0}^{p} L^n$.
Hashigushi et Simon ont prouvé indépendamment que le problème de la puissance finie est **décidable**.
Nous nous contenterons ici de reformuler le problème en utilisant nos algèbres semi-tropicales.
Soit en effet $\mathcal{A}$ un automate déterministe reconnaissant le langage étudié, et $\mathcal{A}'$ l'automate obtenu à partir de $\mathcal{A}$ en ajoutant systématiquement une ϵ-transition des états finals de $\mathcal{A}$ vers les états initiaux. Il suffit de définir γ sur les transitions de $\mathcal{A}'$ de la façon suivante: on associe 0 aux transitions de $\mathcal{A}$ et 1 aux seules transitions nouvellement ajoutées à $\mathcal{A}'$.
Il s'agit maintenant de décider si l'ensemble $\omega(u)$ est majoré, toujours bien sûr dans le cadre de notre semi-anneau $(K, \oplus, \otimes) = (\mathbb{N} \cup + \leqslant 0, \min, +)$.

1. La dénomination provient de l'origine — brésilienne — de leur créateur, Imre Simon. On n'hésitera d'ailleurs pas à appeler *exotiques* toutes ces algèbres un peu inhabituelles.

6 Automates et séries formelles

6.1 Notations

On conserve les notations précédentes.
On a associé à tout automate $\mathcal{A}$ une fonction de poids $\omega : A^\star \to K$. Rappelons rapidement que si c est un chemin qui s'écrit $q_0a_1q_1\dots a_nq_n$, on a posé $\omega(c) = \alpha(q_0)\otimes\bigotimes_{i=1}^n \gamma(q_{i-1}a_iq_i)\otimes\beta(q_n)$, et pour tout mot $u\in A^\star$, on a écrit $\omega(u) = \bigoplus_{\ell(c)=u} \omega(c)$.
On peut considérer la *série formelle* $\sum\omega(u)u$. La série formelle S associée à l'automate $\mathcal{A}$ pourra être notée $S=\tilde{\mathcal{A}}$, on dira qu'il s'agit d'une *série formelle reconnaissable.*
Nous userons des notations suivantes : si $S=\sum\omega(u)u$, nous écrirons $(S\mid u) = \omega(u)$ et nous disposons donc de l'écriture habituelle $S=\sum(S\mid u)u$.

6.2 Opérations

L'ensemble des séries formelles à coefficients dans K et à variable dans A^* est noté $K\langle\langle A\rangle\rangle$.
Soit S et T deux séries formelles, éléments de $K\langle\langle A\rangle\rangle$ et $k\in K$.
On définit un produit externe à gauche et à droite en posant $(kS\mid u) = k\otimes(S\mid u)$ et $(Sk\mid u) = (S\mid u)\otimes k$.
L'addition interne est clairement définie par $(S\oplus T\mid u) = (S\mid u)\oplus(T\mid u)$.
En poussant l'analogie avec les séries formelles classiques, on est amené à définir le produit interne en posant

$$(S\otimes T\mid u) = \bigoplus_{u=v.w}(S\mid v)\otimes(T\mid w),$$

la somme étant écrite pour toutes les décompositions du mot u en concaténé de deux facteurs v et w.
La série formelle S est dite *propre* si $(S\mid\epsilon)=0$. On dispose pour une série formelle propre S de la définition suivante de $S^\star$: $(S^\star\mid u) = \bigoplus_{n\leqslant\infty}(S^n\mid u)$, puisque, pour chaque u, la somme écrite est alors finie.

6.3 Le théorème de Kleene pour les séries formelles

Une série formelle *reconnaissable* est une série formelle $\tilde{\mathcal{A}}$ associée à un automate fini $\mathcal{A}$. On notera $\mathrm{Rec}\,K\langle\langle A\rangle\rangle$ leur ensemble.
L'ensemble $\mathrm{Rat}\,K\langle\langle A\rangle\rangle$ des séries formelles *rationnelles* est défini, comme on l'avait fait pour les langages rationnels, par clôture transitive : on part de l'ensemble des séries formelles S associées à un mot unique u (c'est-à-dire $S=u$, ou, si l'on préfère, $(S\mid v)=0$ si $vot = u$ et $(S\mid u)=1$. On accepte alors les opérations définies sur $K\langle\langle A\rangle\rangle$: les produits externes à droite et à gauche, la somme et le produit interne, et l'étoile : $S\mapsto S^\star$. La clôture transitive obtenue définit $\mathrm{Rat}\,K\langle\langle A\rangle\rangle$.
Schützenberger a montré l'analogue du théorème de Kleene pour les séries formelles : les séries formelles reconnaissables sont les séries rationnelles, ou encore $\mathrm{Rat}\,K\langle\langle A\rangle\rangle = \mathrm{Rec}\,K\langle\langle A\rangle\rangle$.
On peut donc dire que les séries formelles sont un bon outil, et s'amuser par exemple à retrouver les stabilités habituelles de l'ensemble des langages rationnels par complémentaire, morphismes, intersection, etc. directement sur les sé-

ries formelles... On se reportera avec profit à l'ouvrage *Les séries formelles et leurs langages*, de Berstel et Reutenauer, chez Masson.

7 Automates et morphismes

7.1 Quelques stabilités intéressantes

shuffle Si K et L sont deux langages, leur mélange, noté $K \sqcup\!\sqcup L$, est défini par

$$K \sqcup\!\sqcup L = u_1 v_1 u_2 v_2 \dots u_n v_n, u_1 \dots u_n \in K, v_1 \dots v_n \in L,$$

les facteurs u_i et v_i pouvant être égaux au mot videăϵ.
Montrons que si K et L sont reconnaissables, alors $K \sqcup\!\sqcup L$ aussi.
Soit en effet $\mathcal{A} = (Q, T, I, F)$ (*resp.* $\mathcal{B} = (Q', T', I', F')$) un automate qui reconnaît K (*resp.* qui reconnaît L).
On construit un automate $\mathcal{C}$ qui reconnaît $K \sqcup\!\sqcup L$ en posant $\mathcal{C} = (Q \times Q', \tilde{T}, I \times I', F \times F')$ où l'ensemble des transitions $\tilde{T}$ s'obtient en posant :

$$\tilde{T} = ((p, q'), a, (q, q')), (p, a, q) \in T, q' \in Q' \bigcup ((q, p'), a, (q, q')), q \in Q, (p', a, q') \in T'.$$

Ainsi, de chaque état (p, q) on choisit, à la lecture d'un caractère a, d'utiliser la transition correspondante dans l'automate $\mathcal{A}$ ou bien dans l'automate $\mathcal{B}$.

morphismes Si ϕ est un morphisme du monoïde $A^\star$ dans le monoïde $B^\star$ (A et B étant bien entendu deux alphabets finis), et si L est un langage reconnaissable de $A^\star$, alors $\phi(L)$ est un langage reconnaissable de $B^\star$.
Il suffira pour construire un automate qui reconnaisse $\phi(L)$ de partir d'un automate qui reconnaît L, et de remplacer chaque transition (p, a, q) par une chaîne de transitions dans le nouvel automate. Plus précisément, si $\phi(a)$ est le mot $b = b_1 \dots b_k$, on crée $k - 1$ nouveaux états $p_1, \dots, p_{k-1}$ et les transitions (p, b_1, p_1), (p_1, b_2, p_2), $\dots$, (p_{k-1}, b_k, q). On fera simplement très attention à ne pas réutiliser ces états nouvellement créés pour d'autres transitions.

morphismes inverses Avec les mêmes notations, on montre — plus facilement d'ailleurs — que si K est un langage reconnaissable sur l'alphabet B, alors $\phi^{-1}(K)$ est aussi un langage reconnaissable (sur l'alphabet A).
En effet si le langage K est reconnu par l'automate $\mathcal{A} = (Q, T, I, F)$, il suffit de considérer, pour reconnaître le langage $\phi^{-1}(K)$, l'automate (Q, T', I, F) où $(p, a, q) \in T'$ si il existe dans $\mathcal{A}$ un chemin de p à q d'étiquette $\phi(a)$.
Ce qui nous intéressera ici, c'est qu'on a là une méthode **pratique** pour prouver la reconnaissabilité dans de nombreux cas classiques dont nous listons maintenant quelques uns.

7.2 Application : reconnaissabilité de certains langages

Étant donné un langage reconnaissable L et un langage M obtenu à l'aide de certaines opérations à partir de L, nous montrerons que M est également reconnaissable en exhibant deux morphismes ϕ et ψ et un langage reconnaissable K tels que $M = \phi\big(\psi^{-1}(L) \cap K\big)$.

$$\begin{array}{ccc} & \xrightarrow{\cap K} & \\ \psi \downarrow & & \downarrow \phi \\ L \subset X^\star & & M \subset Y^\star \end{array}$$

sous-mots Rappelons qu'un *sous-mot* d'un mot $u = u_1 \dots u_n$ est un mot $u_{i_1} u_{i_2} \dots u_{i_k}$ où on a choisi les indices i_j de sorte que $1 \leqslant 2i_1 < i_2 < \dots < i_k \leqslant 2n$.
Soit L un langage reconnaissable sur l'alphabet $A = a, b$ (on s'est limité à un alphabet à deux lettres pour nos exemples).
Pour montrer que l'ensemble des sous-mots de L est un langage reconnaissable, nous posons $\psi(\bar{b}) = \psi(b) = b$, $\psi(\bar{a}) = \psi(a) = a$, K est l'ensemble de tous les mots sur l'alphabet $a, \bar{a}, b, \bar{b}$, et $\phi(\bar{a}) = \phi(\bar{b}) = \epsilon$, $\phi(a) = a$, $\phi(b) = b$: nous avons sur-barré par ψ certains caractères, que nous effaçons par ϕ.

facteurs Rappelons qu'un *facteur* d'un mot u est un mot v tel qu'il existe deux mots x et y vérifiant $u = xvy$.
Ainsi abb est un sous-mot de $abab$, mais pas un facteur !
On conserve les morphismes précédents, mais on intersecte avec $K = \bar{A}^\star . A^\star . \bar{A}^\star$: on aura effacé des lettres en tête et en queue.

facteurs gauches De la même façon, on trouvera les facteurs gauches, c'est-à-dire les préfixes, en effaçant à droite : $K = A^\star . \bar{A}^\star$.

facteurs droits De la même façon, on trouvera les facteurs droits, c'est-à-dire les suffixes, en effaçant à gauche : $K = \bar{A}^\star . A^\star$.

suppression d'un mot Inversement, montrons que $M = uv, uabv \in L$ est aussi reconnaissable.
Il suffira de sur-barrer des lettres : $\psi(\bar{a}) = \psi(a) = a$, $\psi(\bar{b}) = \psi(b) = b$; d'intersecter avec $K = A^\star . \bar{a}\bar{b} . A^\star$; et de supprimer les lettres barrées : $\phi(\bar{a}) = \phi(\bar{b}) = \epsilon$, $\phi(a) = a$, $\phi(b) = b$.

suppression de toutes les occurrences d'un mot Pour supprimer toutes les occurrences du mot ab on peut penser à procéder comme précédemment en posant simplement cette fois $K = (A^\star . \bar{a}\bar{b})^\star . A^\star$.
Mais voilà, c'est incorrect : parmi les antécédents de $abbab$ par ψ, on trouve $\bar{a}\bar{b}bab$ dont l'image par ϕ est bab qui contient encore une occurrence du ab honni.
La bonne solution consiste à préciser le calcul en posant :

$$K = ((A^\star \setminus A^\star . ab . A^\star) . \bar{a}\bar{b})^\star . (A^\star \setminus A^\star . ab . A^\star).$$

Notons toutefois qu'on n'a pas réalisé la suppression **récursive** de toutes les occurrences de ab : on vérifiera qu'à partir du mot **a**ab**b**ab on obtient **ab**.

insertion d'un mot Montrons par exemple que le langage $M = uabv, uv \in L$ obtenu en insérant le mot ab à l'intérieur des mots du langage reconnaissable L est encore reconnaissable.
On pose cette fois $\psi(a) = a$, $\psi(\bar{a}) = \epsilon$, $\psi(b) = b$, $\psi(\bar{b}) = \epsilon$, afin d'insérer des $\bar{a}$ et des $\bar{b}$ un peu partout.
On intersecte alors avec $K = A^\star . \bar{a}\bar{b} . A^\star$, afin d'obtenir le mot souhaité ab.
Il ne reste plus qu'à supprimer les barres en posant $\phi(\bar{a}) = \phi(a) = a$ et $\phi(\bar{b}) = \phi(b) = b$.

Table des matières

2. La dénomination provient de l'origine — brésilienne — de leur créateur, Imre Simon. On n'hésitera d'ailleurs pas à appeler *exotiques* toutes ces algèbres un peu inhabituelles.