Fonction successeur de langages rationnels

Ce chapitre traite particulièrement de la fonction successeur pour des langages rationnels. Nous avons déjà vu dans les chapitres précédents que la fonction successeur dans un langage rationnel est une fonction rationnelle mais qu'elle n'est pas nécessairement séquentielle ou co-séquentielle, à travers les exemples 56 et 57. Nous allons maintenant montrer que le résultat trouvé pour ces deux exemples est un résultat général :

Théorème 72 ([3])**.** *La fonction successeur d'un langage rationnel est co-séquentielle par morceaux.*

La preuve de ce théorème, que nous donnons dans ce chapitre, construit une union finie de transducteurs séquentiels droits réalisant la fonction successeur d'un langage L à partir d'un automate fini déterministe qui reconnaît L. En fait nous réduisons le problème de la fonction successeur à une fonction successeur restreinte aux mots dont le successeur est de même longueur. Pour retrouver la fonction successeur à partir de cette restriction, il faut imaginer introduire un caractère spécial que l'on insère autant de fois que nécessaire en début de mot.

Afin de montrer la co-séquentialité par morceaux des fonctions successeur à longueur constante, nous présentons tout d'abord des constructions particulières sur des automates et sur des transducteurs pour avoir des transducteurs séquentiels – nous travaillons sur les transducteurs séquentiels sans préciser s'ils sont droits ou gauches –.

8.1 Produit synchronisé d'automates monocyles

Dans cette section nous voulons établir le résultat suivant qui sera utilisé par la suite dans la preuve du Théorème 72 :

Proposition 73. *Si L et K sont deux langages rationnels avec au plus un mot par longueur, alors la fonction α qui associe à tout mot de L le mot de K, de même longueur, si il existe, est une fonction séquentielle et co-séquentielle par morceaux.*

Pour ce faire nous définissons les *langages rayon*, qui sont les langages reconnus par des automates avec un seul cycle, appelés *automates monocycles*. Nous montrons ensuite que le produit synchronisé de deux automates monocycles réalise une fonction séquentielle par morceaux.

8.1.1 Définitions

Langages, automates et transducteurs monocycles

Nous utilisons une terminologie utilisée par Reutenauer dans [38] et nous appelons *langage rayon* un langage L de la forme $L = uv^*w$. L'intérêt de tels langages est que tout langage à croissance bornée est une union finie de langages rayons (folklore, cf. par exemple [43, I.8.4]). En particulier, ce qui va nous intéresser, les langages qui n'ont qu'au plus un mot par longueur sont une union fini de langages rayons.

Un *automate monocycle* est un automate émondé avec un seul état initial, un seul état final et qui est composé d'un unique circuit et de deux chemins qui relient le circuit à l'état initial et l'état final. Un *transducteur monocycle* est un transducteur dont l'automate d'entrée sous-jacent est monocycle.

Un automate monocycle est dit à *boucle préfixe* soit s'il n'a pas de circuit, soit si celui-ci passe par l'état initial. Il est dit à *boucle suffixe* s'il n'a pas de circuit ou bien si celui-ci passe par l'état final.

La proposition suivante traduit les définitions précédentes par la caractérisation des degrés entrants et sortants des états d'un automate monocycle :

Proposition 74. *Les automates monocycles sont les automates émondés avec :*
(i) *un unique état initial,*
(ii) *un unique état final,*
(iii) *le degré entrant de tous les états est au plus* 1 *sauf pour au plus un état pour lequel il vaut* 2*,*

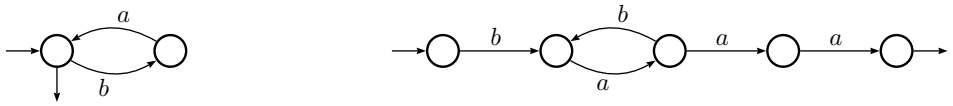
(iv) *le degré sortant de tous les états est au plus* 1 *sauf pour au plus un état pour lequel il vaut* 2.

Les automates monocycles à boucle suffixe (resp. à boucle préfixe) sont les automates émondés avec un unique état initial (resp. état final) dont tous les états ont un degré sortant (resp. entrant) d'au plus 1.

Proposition 75. *Les langages rayons sont exactement les langages reconnus par les automates monocycles.*

Exemple 60. La Figure 8.1 montre deux exemples d'automates monocycles $\mathcal{A}_r$ et $\mathcal{B}_r$. L'automate $\mathcal{A}_r$ reconnaît le langage rayon $(ba)^*$ et $\mathcal{B}_r$ reconnaît $ba(ba)^*aa$.

L'automate $\mathcal{A}_r$ est à la fois à boucle préfixe et à boucle suffixe alors que $\mathcal{B}_r$ n'est ni l'un ni l'autre.



(a) L'automate $\mathcal{A}_r$ (b) L'automate $\mathcal{B}_r$

FIGURE 8.1 – Deux automates monocycles

Proposition 76. *Un automate monocycle $\mathcal{A}$ non déterministe est transformable en un automate monocycle déterministe équivalent.*

Démonstration. La seule source possible de non déterminisme dans un automate $\mathcal{A}$ monocycle est à partir de l'état q dont on sort de la boucle puisque c'est le seul à avoir un degré sortant de 2. Il est possible d'éliminer ce non déterminisme sur q en faisant rouler le circuit vers l'état final, c'est-à-dire :
(i) si q est à la fois l'entrée et la sortie du circuit et qu'il y a une transition de q étiquetée par a à la fois vers q' et q'' alors il faut fusionner q' et q'',
(ii) sinon il faut tout d'abord décaler le cycle d'un état : si q_1 est l'état d'entrée dans le cycle et q_2 sont successeur par la lettre a, alors il faut séparer q_1 en deux états : le premier appartient au cycle mais n'est plus l'état d'entrée du cycle, le deuxième est un état intermédiaire qui n'appartient pas au cycle mais dont la transition sortante va en q_2 (qui devient l'état d'entrée du cycle). Ensuite il faut 'rembobiner' le chemin de sortie : si q a une transition sortante étiqueté par a dans la boucle vers l'état q' et une

transition étiquetée par a vers un état q'' sur le chemin vers l'état final, il est possible de fusionner q' et q''. L'état q' devient alors l'état de sortie de la boucle à la place de q.

Comme le chemin de sortie est fini, si le non déterminisme subsiste, il suffit de refaire rouler le circuit un nombre fini de fois (dans le pire des cas, l'automate deviendra alors a boucle suffixe). La transformation est montrée sur la Figure 8.2.

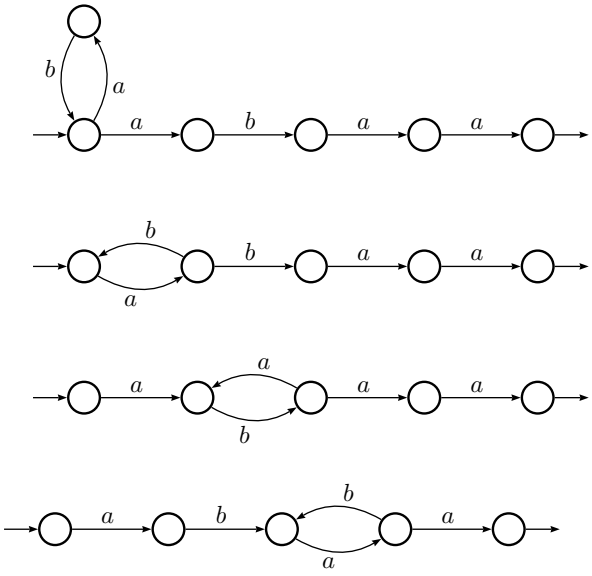


FIGURE 8.2 – Transformation d'un automate monocycle en automate monocycle déterministe

□

La transformation ci-dessus est adaptable de manière duale pour obtenir un automate monocycle co-déterministe en roulant le cycle vers l'état initial.

Produit synchronisé de deux automates

Définition 46. *Soient* $\mathcal{A} = \langle Q, A, E, I, T\rangle$ *et* $\mathcal{B} = \langle R, A, F, J, U\rangle$ *deux automates finis. Le produit synchronisé de* $\mathcal{A}$ *par* $\mathcal{B}$ *est la partie émondée du transducteur :*

$$\mathcal{A} \bowtie \mathcal{B} = \langle Q \times R, A, A, G, I \times J, T \times U\rangle \ ,$$

où l'ensemble des transitions G est défini par :

$$G = \{((p,r),(a,b),(q,s)) \mid (p,a,q) \in E, (r,b,s) \in F\} \ .$$

Si $\mathcal{A}$ et $\mathcal{B}$ sont deux automates finis sur A^* qui reconnaissent respectivement les langages L et K alors le produit synchronisé [1] $\mathcal{A} \bowtie \mathcal{B}$ réalise la relation de A^* dans A^* dont le graphe est :

$$\{(u,v) \mid u \in L, v \in K, |u| = |v|\} = L \times K \cap (A \times A)^* \ .$$

Si $\mathcal{B}$ reconnaît un langage avec au plus un seul mot par longueur alors le transducteur $\mathcal{A} \bowtie \mathcal{B}$ réalise une fonction.

Le produit synchronisé est distributif par rapport à l'union :

$$\left[\bigcup_{i\in I} \mathcal{A}_i\right] \bowtie \left[\bigcup_{j\in J} \mathcal{B}_j\right] = \bigcup_{(i,j)\in I\times J} (\mathcal{A}_i \bowtie \mathcal{B}_j) \ .$$

Le degré sortant d'un état (p,q) de $\mathcal{A} \bowtie \mathcal{B}$ est le produit des degrés sortants des états p de $\mathcal{A}$ et q de $\mathcal{B}$. De la définition du produit synchronisé on tire que, même si $\mathcal{A}$ est déterministe, le transducteur $\mathcal{A} \bowtie \mathcal{B}$ n'est pas séquentiel dès $\mathcal{B}$ a un état de degré sortant au moins 2.

Proposition 77. *Si $\mathcal{A}$ est un automate monocycle déterministe et si $\mathcal{B}$ est un automate monocycle à boucle suffixe (resp. préfixe) alors le produit synchronisé $\mathcal{A} \bowtie \mathcal{B}$ est un transducteur monocycle séquentiel (resp. co-séquentiel).*

Démonstration. Cette proposition est un corollaire direct de la Proposition 74 car $\mathcal{B}$ à boucle suffixe implique que le degré sortant de chaque état est au plus 1 et donc il n'y a qu'une transition créée dans $\mathcal{A} \bowtie \mathcal{B}$ pour chaque transition sortante de $\mathcal{A}$ par état de $\mathcal{A} \bowtie \mathcal{B}$ correspondant.

Comme $\mathcal{A}$ est déterministe, l'automate d'entrée sous-jacent à $\mathcal{A} \bowtie \mathcal{B}$ est déterministe. □

Exemple 61 (*Ex. 60 cont.*)**.** La Figure 61 montre le produit synchronisé de $\mathcal{A}_r$ par $\mathcal{B}_r$ qui n'est ni séquentiel ni co-séquentiel.

1. Le produit synchronisé défini ici diffère un peu de celui défini en [43, Exerc. IV.6.17] qui réalise la relation dont le graphe est $L \times K$.

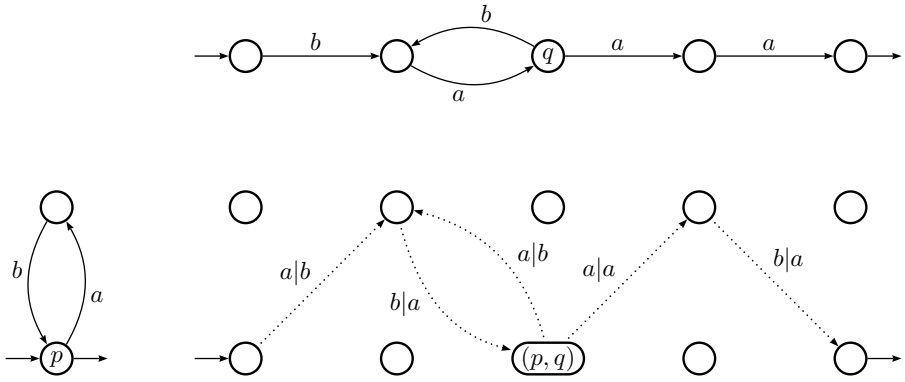


FIGURE 8.3 – Le produit synchronisé $\mathcal{A}_r \bowtie \mathcal{B}_r$

8.1.2 Séquentialité du produit de deux automates monocycles

Dans cette sous-section, nous allons montrer que la fonction réalisée par le produit synchronisé de deux automates monocycles est une fonction séquentielle et co-séquentielle.

Pour cela, nous allons modifier légèrement la définition des automates pour y ajouter des transitions spontanées et des fonctions initiale et finale :

- les transitions d'un automate sont soit étiquetées par une lettre a de A, soit par le mot vide 1_{A^*} ;
- les ensembles initiaux et finaux I et T sont remplacés par des fonctions de Q dans A^* que nous notons également I et T et dont les valeurs pour un état q seront notées I_q et T_q.

Quand ce sera nécessaire, dans cette sous-section, nous appellons *automates classiques* les automates sans transitions spontanées et sans fonctions initiales et finales. Il est connu que les automates – avec la définition étendue – reconnaissent exactement les mêmes langages que les automates classiques et que les automates classiques sont vus comme des automates en définissant une fonction initiale (resp. finale) ayant pour valeur 1_{A^*} sur les états initiaux (resp. finaux) – et $\emptyset$ sur les autres –.

Si c est un calcul réussi d'un automate $\mathcal{A}$, alors nous notons $\ell(c)$ la *longueur* de c, c'est-à-dire le nombre de transitions de c.

Cette définition élargie de la notion d'automate nous permet de montrer la proposition suivante :

Proposition 78. *Tout automate monocycle classique $\mathcal{B}$ peut être transformé en un automate monocycle $\breve{\mathcal{B}}$ tel que :*
(i) *$\breve{\mathcal{B}}$ est équivalent à $\mathcal{B}$;*
(ii) *$\breve{\mathcal{B}}$ est à boucle suffixe ;*
(iii) *si les calculs c et c' respectivement de $\mathcal{B}$ et $\breve{\mathcal{B}}$ sont étiquetés par le même mot, alors $\ell(c) = \ell(c')$.*

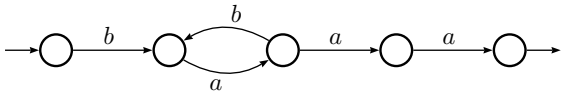
Démonstration. L'automate $\breve{\mathcal{B}}$ est obtenu à partir de l'automate monocycle $\mathcal{B}$ par les opérations suivantes :

(a) Si q est l'état de $\mathcal{B}$ par lequel on peut sortir du circuit et si w est l'étiquette du chemin qui va de q à l'état final de $\mathcal{B}$ alors on remplace ce chemin par une valeur pour la fonction finale en q de $\breve{\mathcal{B}}$ égale à w.

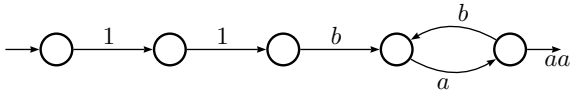
(b) Ajout avant l'état initial de $\mathcal{B}$ de $|w|$ transitions spontanées.

Par (a), l'automate $\breve{\mathcal{B}}$ est a boucle suffixe et est équivalent à $\mathcal{B}$; l'égalité des longueurs des calculs vient de l'opération (b). □

Exemple 62 (*Ex. 60 cont.*)**.** La construction appliquée à l'automate $\mathcal{B}_r$ est montrée sur la Figure 8.4.



(a) L'automate classique $\mathcal{B}_r$



(b) L'automate $\breve{\mathcal{B}}_r$

FIGURE 8.4 – Transformation d'un automate monocycle en un automate monocycle à boucle suffixe

Il existe une transformation duale qui transforme un automate monocycle en un automate monocycle à boucle préfixe.

La définition du produit synchronisé est également étendue à la nouvelle classe d'automates :

$$\mathcal{A} \bowtie \mathcal{B} = \langle Q \times R, A, A, G, I \times J, T \times U \rangle$$

où : $[I \times J]_{(p,q)} = (I_p, J_q)$, $[T \times U]_{(p,q)} = (T_p, U_q)$ et :

$$G = \{((p,r),(x,y),(q,s)) \mid (p,x,q) \in E, (r,y,s) \in F\} \ .$$

Exemple 63 (*Ex. 60 cont.*)**.** La Figure 8.5 montre le produit synchronisé de $\mathcal{A}_r$ par $\breve{\mathcal{B}}_r$.

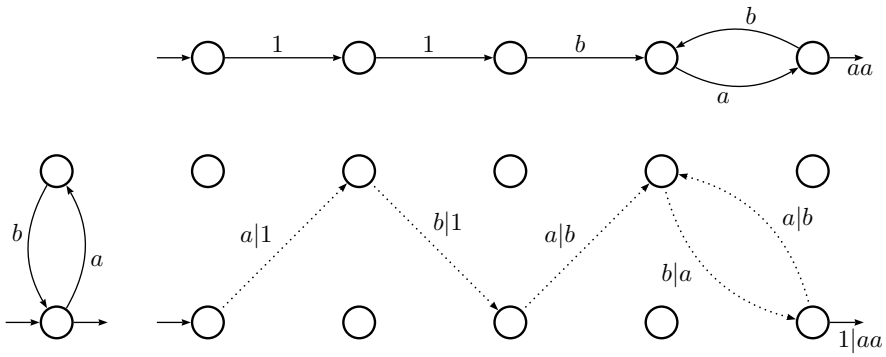


FIGURE 8.5 – Le produit synchronisé $\mathcal{A}_r \bowtie \breve{\mathcal{B}}_r$

Le graphe de la fonction réalisée par le produit synchronisé de deux automates $\mathcal{A}$ et $\mathcal{B}$ qui reconnaissent L et K n'est plus l'ensemble des couples (u, v) où u est dans L et v dans K tels que $|u| = |v|$ mais tels qu'il existe un calcul de u dans $\mathcal{A}$ de longueur égale à un calcul de v dans $\mathcal{B}$.

D'après la définition de $\breve{\mathcal{B}}$ on a donc :

Proposition 79. *Soit $\mathcal{A}$ un automate et $\mathcal{B}$ un automate monocycle. Le produit synchronisé $\mathcal{A} \bowtie \mathcal{B}$ est équivalent au produit synchronisé $\mathcal{A} \bowtie \breve{\mathcal{B}}$.*

Comme l'automate $\breve{\mathcal{B}}$ a un degré sortant d'au plus A, il apparaît que le produit synchronisé $\mathcal{A} \bowtie \breve{\mathcal{B}}$ est séquentiel lorsque $\mathcal{A}$ est déterministe.

Comme nous avons montré que la transformation $\breve{\mathcal{B}}$ est possible pour tout automate monocycle :

Théorème 80. *Le produit synchronisé $\mathcal{A} \bowtie \mathcal{B}$ d'un automate $\mathcal{A}$ déterministe avec un automate $\mathcal{B}$ monocycle réalise une fonction séquentielle.*

De la Proposition 77 nous déduisons également :

Proposition 81. *Soient $\mathcal{A}$ et $\mathcal{B}$ deux automates monocycles. Si $\mathcal{A}$ est déterministe, alors le produit synchronisé $\mathcal{A} \bowtie \breve{\mathcal{B}}$ est un transducteur monocycle séquentiel.*

Comme nous avons vu par la Proposition 76 qu'un automate monocycle $\mathcal{A}$ pouvait être transformé en un automate monocycle déterministe équivalent et comme des transformations duales existent, nous avons :

Théorème 82. *Le produit synchronisé $\mathcal{A} \bowtie \mathcal{B}$ de deux automates monocycles réalise une fonction séquentielle et co-séquentielle.*

Il faut noter que si le produit synchronisé de deux automates monocycles réalise une fonction à la fois séquentielle et co-séquentielle, c'est parce qu'il existe – et nous les construisons – un tranducteur séquentiel et un transducteur co-séquentiel qui la réalisent. Il n'y a cependant pas nécessairement de transducteur à la fois séquentiel et co-séquentiel pour réaliser cette fonction.

8.1.3 Les langages avec un mot par longueur au plus

Nous pouvons maintenant donner la preuve de la Proposition 73 :

Démonstration. Les langages L et K sont des unions finies de langages rayon et donc respectivement reconnus par des unions $\mathcal{A}_1, \ldots, \mathcal{A}_n$ et $\mathcal{B}_1, \ldots, \mathcal{B}_m$ d'automates monocycles. La fonction α est réalisée par l'union des produits synchronisés $\mathcal{A}_i \bowtie \mathcal{B}_j$.

Nous avons montré que $\breve{\mathcal{B}}_i$ est équivalent à $\mathcal{B}_i$ et donc K est également reconnu par l'union des automates monocycles $\breve{\mathcal{B}}_1, \ldots, \breve{\mathcal{B}}_m$ (Proposition 78).

Pour tout i et j, par la Proposition 81, le transducteur $\mathcal{A}_i \bowtie \breve{\mathcal{B}}_j$ est un transducteur séquentiel et donc α est une fonction séquentielle par morceaux.

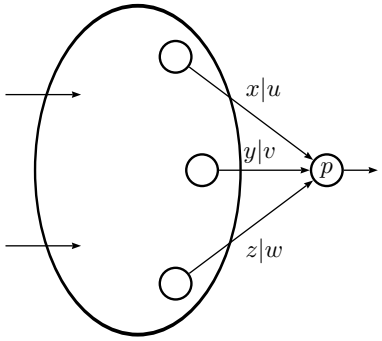
La co-séquentialité se prouve directement de manière duale. □

8.2 Concaténation de transducteurs co-séquentiels

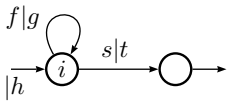
Dans cette section nous revenons à la définition classique des automates et nous allons donner des conditions sur des transducteurs co-séquentiels afin de réaliser la concaténation par un transducteur co-séquentiel. Plus précisément, et comme c'est le cas qui nous intéressera dans le développement de la preuve, nous étudions la concaténation d'un transducteur co-séquentiel avec un transducteur monocycle co-séquentiel.

Comme nous l'avons déjà montré, la concaténation de deux fonction séquentielles – ou co-séquentielles – n'est pas nécessairement séquentielle, ce n'est même pas nécessairement une fonction.

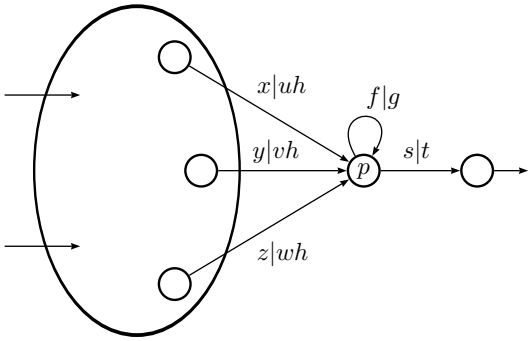
Sur la Figure 8.6 nous montrons la construction qui nous intéresse particulièrement : la concaténation d'un transducteur co-séquentiel et co-standard κ avec un transducteur monocycle co-séquentiel τ. Cette concaténation est notée $\kappa||\tau$.



(a) Le transducteur co-standard κ



(b) Le transducteur monocycle τ



(c) La concaténation $\kappa||\tau$

FIGURE 8.6 – La concaténation de deux transducteurs

Proposition 83. *Soit κ un transducteur co-séquentiel et co-standard. Si τ est un transducteur monocycle co-séquentiel, alors la concaténation $\kappa||\tau$ est un transducteur co-séquentiel si, et seulement si, on a l'une des conditions suivantes :*

(i) *τ n'a pas de circuit ;*

(ii) *τ n'est pas à boucle préfixe ;*

(iii) *τ est à boucle préfixe mais l'étiquette de l'unique transition entrante dans l'état initial de l'automate sous-jacent d'entrée de τ est différente de*

l'étiquette des transitions arrivant dans p, état final de l'automate sous-jacent de κ.

Pour comprendre cette propriété, il faut remarquer que lors de la concaténation de deux transducteurs co-séquentiels, la seule non co-séquentialité peut apparaître lors de la fusion de l'état initial i de τ et de l'état final p de κ. Il faut donc regarder les transitions entrantes dans l'état initial i de τ.

Si τ n'est pas à boucle préfixe, i a un degré entrant nul et l'affaire est entendue. Si i est sur le circuit de τ, la troisième condition assure que les étiquettes de la transition qui arrive en i et de celles qui arrivent en p sont différentes. La concaténation est alors assurée d'être co-séquentielle.

Il y a une et une seule transition entrante sur i si et seulement si i est sur le circuit de τ. Il convient alors de vérifier si le transducteur $\kappa||\tau$ est co-séquentielle en analysant les automates d'entrée sous-jacents.

8.3 La fonction successeur uniforme

Dans cette section nous définissons la fonction successeur uniforme et nous montrons que le problème de la co-séquentialité de la fonction successeur se réduit à la co-séquentialité de cette fonction uniforme.

Nous introduisons tout d'abord les langages des mots minimaux et des mots maximaux – pour chaque longueur – d'un langage L :

$$\begin{aligned}\mathsf{Min}(L) &= \{u \in L \mid \forall v \in L, |u| = |v| \Rightarrow u \preccurlyeq v\} \ , \\ \mathsf{Max}(L) &= \{u \in L \mid \forall v \in L, |u| = |v| \Rightarrow v \preccurlyeq u\} \ .\end{aligned}$$

On peut voir dans [42, 46], par exemple, que si L est rationnel alors les langages $\mathsf{Min}(L)$ et $\mathsf{Max}(L)$ sont également rationnels. Leur définition implique également que ce sont des langages avec au plus un mot par longueur.

Ces langages apparaissent utiles pour prouver la co-séquentialité de la fonction successeur car lorsque un mot u et son successeur v ont la même longueur, alors il existe $a < b$ deux lettres et u', v' deux mots tels que, si w est le plus long commun préfixe de u et v alors :

$$u = wau' \qquad \text{et} \qquad v = wbv' \ . \tag{8.1}$$

Et u est le mot maximal de longueur $|u|$ de L commençant par wa et v est le mot minimal de longueur $|u|$ de L commençant par wb. En fait v est le mot minimal de longueur $|u|$ dans L commençant par wc pour toute lettre c telle que $a < c$.

C'est pour cela que le bloc principal de la construction de notre preuve est le produit synchronisé de langages avec au plus un mot par longueur. Comme le produit synchronisé est une fonction qui préserve la longueur, et que la fonction successeur ne le fait pas, nous allons tout d' abord utiliser une restriction de la fonction successeur préservant la longueur :

Définition 47. *La fonction successeur uniforme d'un langage rationnel L, notée ULSucc_L, est la restriction de la fonction Succ_L à $(A \times A)^*$:*

$$\forall u \in L, \quad \mathsf{ULSucc}_L(u) = \mathsf{Succ}_L(u) \Leftrightarrow |u| = |\mathsf{Succ}_L(u)| \ ,$$

et si $|\mathsf{Succ}_L(u)| > |u|$ alors ULSucc_L n'est pas définie sur u.

Les mots de L pour lesquels la fonction ULSucc_L n'est pas définie sont exactement les mots dans $\mathsf{Max}(L)$ – c'est-à-dire dont le successeur est dans $\mathsf{Min}(L)$–.

De plus, si $\mathsf{Succ}_L(u) = v$, pour $u \in \mathsf{Max}(L)$, alors il n'y a aucun mot dans L de longueur l telle que $|u| < l < |v|$.

L'idée, pour réduire le problème de la fonction successeur à sa restriction uniforme, est d'introduire dans l'alphabet un nouveau caractère spécial – que nous notons \$ dans la suite –, qui est plus petit que toutes les autres lettres, et qui s'ajoute en tête de chaque mot du langage pour former le langage $K = \$^*L$. Ce langage est choisi car tout mot u de L peut être représenté par un mot de K de longueur $l \geqslant |u|$: $\$^{l-|u|}u$. En particulier le calcul de la fonction successeur uniforme sur K pour tout mot de K 'contient' le calcul de la fonction successeur sur L.

Exemple 64. La Figure 8.7 représente le langage $K_1 = \$^*L_1$ où L_1 est le langage des mots contenant un nombre pair d'occurrences de b.

Plus particulièrement, si u est dans $\mathsf{Max}(L)$, et donc son successeur v est de longueur plus grande, alors le mot $\$^{|v|-|u|}u$ de K a pour successeur v dans K et $\$^{|v|-|u|}u$ et v ont la même longueur. Cela permet de montrer que même si la fonction successeur ne préserve pas la longueur, la fonction successeur uniforme permet de retrouver tous les successeurs. En particulier, le résultat qui nous intéresse est :

Proposition 84. *Si $\mathsf{ULSucc}_{\$^*L}$ est co-séquentielle par morceaux, alors Succ_L l'est également.*

Démonstration. La fonction qui "efface" toutes les occurrences du caractère spécial \$ en début de mot est une fonction séquentielle et co-séquentielle réalisée par le transducteur de la Figure 8.8 pour un alphabet $\{a_1, \ldots, a_n\}$.

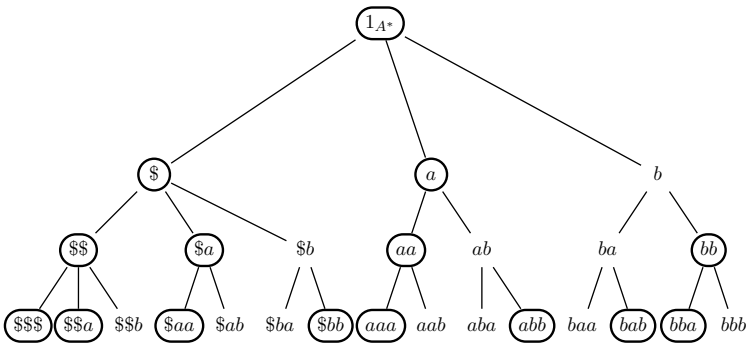


FIGURE 8.7 – L'arbre représentant K_1

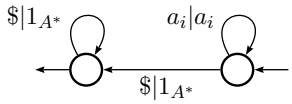


FIGURE 8.8 – Transducteur co-séquentiel qui efface les \$

Si la fonction ULSucc_L est co-séquentielle par morceaux, alors elle peut être réalisée par une cascade de transducteurs co-séquentiels de hauteur h. La cascade de transducteurs de hauteur $h+1$, formée par la cascade précédente et dont tous les transducteurs de hauteur $h+1$ sont le transducteur de la Figure 8.8, réalise la fonction successeur sur L.

Si on ajoute le transducteur de la Figure 8.8, nous obtenons une cascade de hauteur supérieure qui réalise la fonction Succ_L. La fonction Succ_L est donc co-séquentielle par morceaux. □

Remarque 14. Comme nous avons déjà montré qu'il est possible de transformer une union finie de transducteurs co-séquentiels en une cascade de transducteurs co-séquentiels de hauteur 2, la preuve ci-dessus donne une construction de la cascade réalisant Succ_L à partir de l'union finie réalisant ULSucc_L et que nous allons détailler ci-après.

Finalement, pour prouver le Théorème 72 il ne reste à montrer que le théorème suivant :

Théorème 85. *La fonction successeur uniforme* ULSucc_L *d'un langage rationnel* L *est une fonction co-séquentielle par morceaux.*

Démonstration. Nous donnons la construction d'une union finie de transducteurs co-séquentiels qui réalise ULSucc_L.

L'idée sous-jacente est que si $\mathsf{ULSucc}_L(u) = v$ alors il existe $a < b$ deux lettres de $A \cup \{\$\}$ et u', v' deux mots tels que $u = wau'$ et $v = wbv'$ où $w = u \wedge v$ et u est le mot maximal de longueur n de L commençant par wa ; v est le mot minimal de longueur n de L commençant par wc pour toute lettre c telle que $a < c$ dans L.

Soient $\mathcal{A} = \langle Q, A, \delta, i, T\rangle$, déterministe, qui reconnaît L et $q = i \cdot w$ dans $\mathcal{A}$, alors :

(i) au' est le mot maximal de longueur $|au'|$ que l'on peut lire depuis l'état q dans $\mathcal{A}$ qui commence par la lettre a,

(ii) bv' est le mot minimal de longueur $|bv'| = |au'|$ que l'on peut lire depuis l'état q dans $\mathcal{A}$ qui commence par un lettre plus grande que a.

Il s'agit donc, dans un premier temps, de réaliser la fonction qui associe bv' à au' pour chaque état q atteint. Cette fonction est réalisée par un transducteur monocycle co-séquentiel car est un produit synchronisé de langages avec au plus un mot par longueur.

Posons $L_q = \{w \in A^* \mid q \cdot w \in T\}$ le langage des mots qui sont lus à partir d'un état q dans $\mathcal{A}$ jusqu'à un état final. De même $L'_q = \{w \in A^* \mid i \cdot w = q\}$ sont les mots qui sont lus de l'état initial i jusqu'à q. Les langages L_q et L'_q sont rationnels.

Définissons maintenant $K_{q,a}$ (respectivement $H_{q,a}$) l'ensemble des mots maximaux (resp. minimaux) de L_q qui commencent par a (resp. une lettre plus grande que a) :

$$K_{q,a} = \mathsf{Max}(a(a^{-1}L_q)) \qquad \text{et} \qquad H_{q,a} = \mathsf{Min}(\bigcup_{c>a} c(c^{-1}L_q)) \ .$$

Les langages $K_{q,a}$ et $H_{q,a}$ sont rationnels et ont au plus un mot par longueur. La fonction $\alpha_{q,a}$ qui associe à chaque mot de $K_{q,a}$ le mot de $H_{q,a}$ de même longueur est donc, d'après la Proposition 73 réalisée par une union finie de transducteurs co-séquentiels monocycles.

Lemme 86. *Si* $\mathsf{ULSucc}_L(wau') = wbv'$*, avec* $a < b$ *et que* $w \in L'_q$*, alors la fonction qui associe* au' *à* bv' *est exactement la fonction* $\alpha_{q,a}$*.*

Démonstration. Nous avons déjà montré que au' est dans $K_{q,a}$ et bv' dans $H_{q,a}$ et ils ont la même longueur (par définition de la fonction successeur uniforme).

Il reste donc à montrer que pour tout $s \in K_{q,a}$ et $t \in H_{q,a}$ tels que $|s| = |t|$, tout mot $w \in L'_q$ est tel que $wt = \mathsf{ULSucc}_L(ws)$.

Comme $K_{q,a}$ et $H_{q,a}$ sont des sous-ensembles rationnels de L_q, tout mot $w \in L'_q$ est tel que ws et wt sont dans L. De plus comme s commence par a et t par une lettre plus grande que a, et comme $|ws| = |wt|$ alors $ws \prec wt$. Comme s et t ne commencent pas par la même lettre, et comme s est dans $K_{q,a}$, w est nécessairement le plus long commun préfixe de ws et de son successeur. Finalement son successeur est donc wt car t, dans $H_{q,a}$, est le mot le plus petit, de longueur $|t|$, qui soit plus grand que s et tel que wt soit dans L. □

Par la Proposition 73, la fonction $\alpha_{q,a}$ est une fonction réalisée par une union finie de transducteurs co-séquentiels monocycles que l'on peut construire. Notons $\tau_{q,1}, \ldots, \tau_{q,k}$ de tels transducteurs monocycles. Avec nos notations précédentes, cette union associe bv' à au'.

Pour réaliser la fonction qui associe wbv' à wau', if suffit de concaténer un transducteur réalisant la restriction de l'identité sur L'_q à chaque transducteur $\tau_{q,i}$.

Comme nous l'avons vu précédemment, la concaténation n'est pas nécessairement co-séquentielle. Nous allons donc essayer de réunir les conditions de la Proposition 83. Pour cela, prenons κ_q, un transducteur co-standard et co-séquentiel réalisant l'identité sur L'_q – c'est-à-dire un transducteur dont les entrées sont égales aux sorties et dont l'automate d'entrée sous-jacent est un automate co-déterministe reconnaissant L'_q –.

La Proposition 83 donne des conditions sur les $\tau_{q,i}$ pour que $\kappa_q || \tau_{q,i}$ soit co-séquentiel. Nous allons maintenant montrer que ces conditions sont soit réalisées, soit que l'on peut transformer κ_q et $\tau_{q,i}$ en des transducteurs qui réalisent les conditions :

Lemme 87. *Le transducteur $\theta_{q,i} = \kappa_q || \tau_{q,i}$ réalise une fonction co-séquentielle par morceaux.*

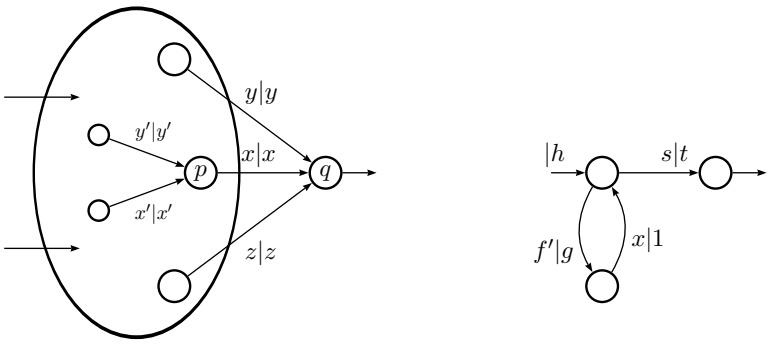
Démonstration. Le transducteur monocycle et co-séquentiel $\tau_{q,i}$ remplit l'une des conditions suivantes :

1. Soit $\tau_{q,i}$ n'a pas de circuit ; soit $\tau_{q,i}$ n'est pas à boucle préfixe ; soit $\tau_{q,i}$ est à boucle préfixe mais l'étiquette de l'unique transition entrante dans l'état initial de l'automate sous-jacent d'entrée de τ est différente de l'entrée des transitions arrivant dans p, état final de κ_q.
2. Ou bien $\tau_{q,i}$ est à boucle initial et l'étiquette d'entrée x de la transition entrante dans l'état initial est égale à l'entrée d'une transition qui va

d'un état p à l'état final q de κ_q. Nous notons $f|g$ l'étiquette du circuit de $\tau_{q,i}$ et donc il existe f tel que $f = f'x$.

Dans le premier cas, correspondant aux conditions de la Proposition 83, la concaténation $\theta_{q,i}$ est co-séquentielle.

Une représentation de $\tau_{q,i}$ et κ_q pour le deuxième cas est montrée sur la Figure 8.9. Pour que la figure soit complètement générale, il faudrait considérer le cas où f' est le mot vide. Cependant le raisonnement qui va suivre est toujours valide avec cette hypothèse.



(a) κ_q (b) $\tau_{q,i}$

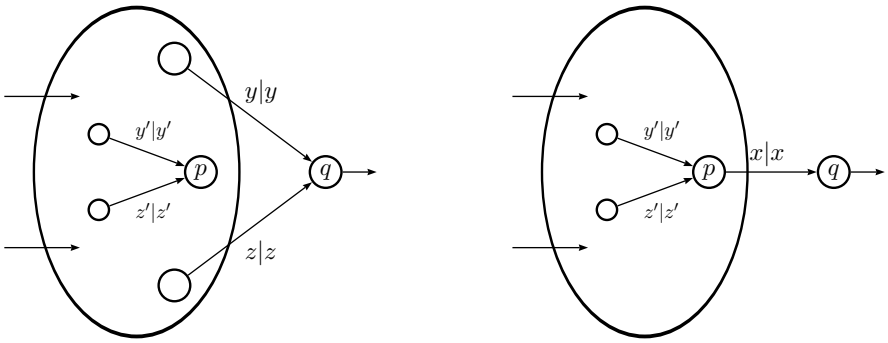
FIGURE 8.9 – κ_q and $\tau_{q,i}$

Dans ce deuxième cas, $\theta_{q,i}$ n'est pas co-séquentiel. Nous allons transformer les transducteurs κ_q et $\tau_{q,i}$ afin de construire une union finie de transducteurs co-séquentiels.

La première transformation consiste à dupliquer κ_q en deux copies $\kappa_{q,1}$ et $\kappa_{q,2}$. Dans $\kappa_{q,1}$ nous enlevons la transition entre p et q étiquetée par x. Dans le deuxièmes nous enlevons toutes les autres transitions entrantes en q – l'état final de κ_q. Cette "séparation" de κ_q en deux transducteurs co-séquentiels est montrée sur la Figure 8.10.

L'union de $\kappa_{q,1}$ et $\kappa_{q,2}$ réalise bien la même fonction identité que κ_q car ce dernier est co-déterministe et co-standard et donc chaque calcul réussi emprunte une unique transition arrivant en l'état final q. Le transducteur $\kappa_{q,1}||\tau_{q,i}$ est dans le cas 1) défini plus haut et donc co-séquentiel.

Maintenant il reste à traiter la concaténation de $\kappa_{q,2}$ avec $\tau_{q,i}$. Rappelons que la seule transition arrivant en l'état final q de $\kappa_{q,2}$ est la transition entre p et q étiquetée par x en entrée.



(a) $\kappa_{q,1}$ (b) $\kappa_{q,2}$

FIGURE 8.10 – Séparation de κ_q en deux composantes

Nous transformons $\kappa_{q,2}$ à nouveau en un transducteur κ'_q en effectuant les opération suivantes :
(i) toutes les transitions d'un état r de $\kappa_{q,2}$ à p, étiquetées par $y|y$ sont dupliquées entre r et q ;
(ii) on concatène x à la sortie de ces nouvelles transitions – l'état de cette transformation peut être vu sur la Figure 8.11 – ;
(iii) on supprime la transition de p à q.

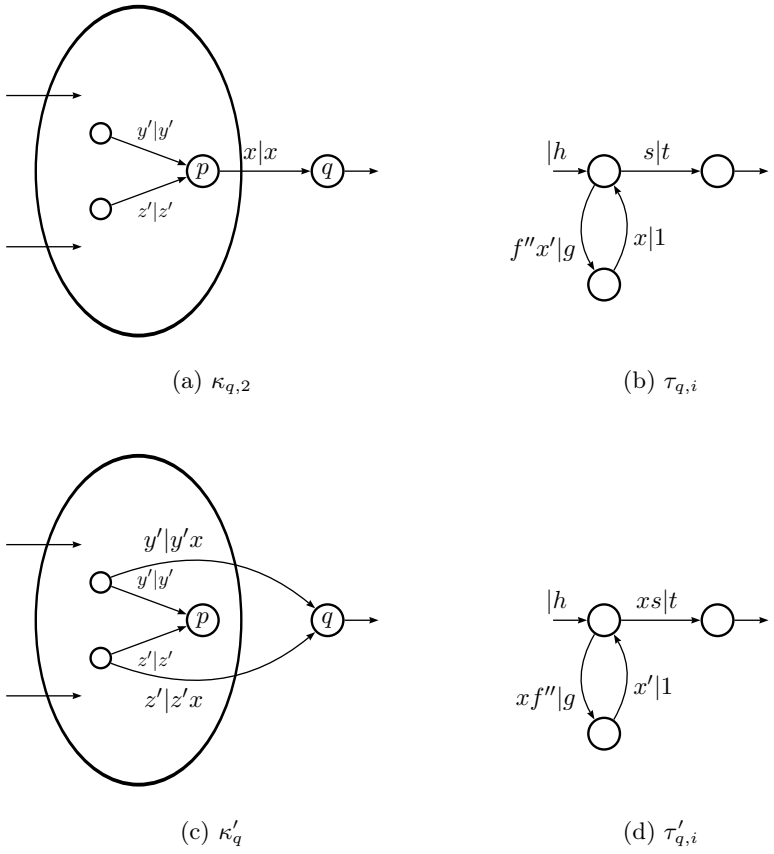
Nous transformons également $\tau_{q,i}$ en $\tau'_{q,i}$ en décalant la boucle de l'état initial. C'est à dire que l'étiquette de la boucle devient $xf'|g$ et nous ajoutons une transition $x|1_{A^*}$ entre la boucle et son chemin de sortie – voir Figure 8.11.

Montrons tout d'abord que la concaténation $\theta'_{q,i} = \kappa'_q||\tau'_{q,i}$ réalise la même fonction que $\kappa_{q,2}||\tau_{q,i}$: d'un coté, nous avons $\mathsf{Dom}\,\kappa'_q = [\mathsf{Dom}\,\kappa_{q,2}]\,x^{-1}$ et $\kappa'_q(u) = ux$, de l'autre coté : $\mathsf{Dom}\,\tau'_{q,i} = x\mathsf{Dom}\,\tau_{q,i}$ et $\tau'_{p,i}(xv) = \tau_{p,i}(v)$.

Nous avons donc finalement :

$$\begin{aligned}\left[\kappa'_q||\tau'_{q,i}\right](uxv) &= \kappa'_q(u)\,\tau'_{q,i}(xv)\\ &= \kappa_{q,2}(ux)\,\tau_{q,i}(v) = \left[\kappa_{q,2}||\tau_{q,i}\right](uxv)\quad .\end{aligned}$$

Nous pouvons remarquer que κ'_q n'a pas exactement les mêmes propriétés que κ_q puisque ses transitions qui arrivent en q ne sont plus étiquetées par une identité. Cependant cela ne joue aucun rôle dans notre construction et nous nous autorisons à faire une récurrence. Nous répétons donc les opérations et les transformations réalisées avec κ_q et $\tau_{q,i}$ avec les nouveaux transducteurs κ'_q et $\tau'_{q,i}$.



(a) $\kappa_{q,2}$ (b) $\tau_{q,i}$

(c) κ'_q (d) $\tau'_{q,i}$

FIGURE 8.11 – Transformation de $\kappa_{q,2}$ et $\tau_{q,i}$ en κ'_q and $\tau'_{q,i}$

Ce genre de transformation pourrait très bien ne pas s'arrêter (cf. Remarque 15 pour la construction d'automates κ et τ pour lesquels ces transformations seraient sans fin). Cependant dans notre cas précis, et de par la nature de $\tau_{q,i}$, cela est impossible et c'est ce que nous établissons maintenant.

Soit n le nombre d'états de $\mathcal{A}$ et $l = |f|$ la longueur de la boucle préfixe de $\tau_{q,i}$. Supposons que nous avons fait nl fois les transformations précédentes en retombant à chaque fois sur le cas 2). Nous aurions alors que f^n est un suffixe d'un mot de L'_q.

Il existe alors nécessairement une boucle dans $\mathcal{A}$ étiquetée par une puissance $k < n$ de f car $\mathcal{A}$ n'a que n états et on va de cette boucle en q en lisant une autre puissance de f. Or comme $\mathcal{A}$ est co-déterministe, cela implique que

$q \cdot f^k = q$.

Comme $\tau_{q,i}$ est une restriction de $K_{q,a} \bowtie H_{q,a}$, les mots $f^k s$ et $hg^k t$ sont des mots de L_q de même longueur. De plus la première lettre de f est a et la première lettre de h est b avec $b > a$. On déduit avec la boucle sur q que f^{2k} et $f^k hg^k t$ sont dans L_q également. Comme h commence par un b on a $f^{2k} \prec f^k hg^k t$. Or donc f^{2k} n'est pas un mot maximal commençant par a et ne devrait donc pas être dans le domaine de $\tau_{q,i}$. Nous en déduisons donc qu'il suffit de faire un nombre fini – inférieur à nl – de fois les transformations pour retomber uniquement sur le cas 1). □

Finalement, il reste à montrer que l'union des $\theta_{q,i}$ réalise exactement une restriction de ULSucc_L aux mots dont le plus grand commun préfixe avec son successeur est dans L'_q. Ceci est direct puisque κ_q réalise exactement l'identité sur L'_q et l'union des $\tau_{q,i}$ exactement la fonction dont l'image de ce que nous avions noté au' est bv'.

□

Remarque 15. Il est possible de construire des transducteurs κ et τ simples, tels que κ soit co-séquentiel et co-standard et τ soit un transducteur monocycle, pour lesquels la construction définie dans la preuve du Lemme 87 est sans fin.

En effet, si l'on prend les transducteurs de la Figure 8.12, la construction est sans fin puisque κ est invariant et que la boucle de τ reste étiquetée par $a|a$.

(a) tranducteur κ (b) transducteur τ

FIGURE 8.12 – Transducteurs pour lesquels la construction est sans fin

Remarque 16. Enfin il faut préciser que toutes les constructions pour la preuve sont symétriques sauf une. En particulier les constructions de produits synchronisés et de concaténations sont compatible avec le passage à la dualité. On pourrait donc s'attendre à pouvoir également construire de manière duale une fonction séquentielle par morceaux. Ce n'est pas le cas car, alors qu'il est possible de choisir un automate déterministe et standard – co-déterministe et co-standard dans notre preuve – reconnaissant un langage L,

il n'est pas possible, en général, de construire un automate déterministe et co-standard – ce qui serait nécessaire pour adapter cette preuve et montrer la séquentialité par morceaux.

En outre, nous avons déjà montré dans le chapitre précédent que la fonction successeur d'un langage rationnel n'est pas, en général, séquentielle par morceaux.