Chapitre 4 : Langages réguliers et automates (2/2)

1 Langages locaux et un sens du théorème de Kleene

1.1 Langages locaux

1.1.1 Définitions et propriétés

Remarque 1.1

Dans cette section on manipule à la fois des ensembles de mots (comme L ou $F(L)$ ci-dessous), et des ensembles de lettres (comme $P(L)$ et $S(L)$ ci-dessous). Quand cela sera utile, on assimilera de manière implicite les lettres à des mots réduits à une lettre de manière à concaténer ces ensembles de lettres comme si c'étaient bien des ensembles de mots.

Définition 1.2

Soit L un langage sur un alphabet Σ.

- *L'**ensemble préfixe** de L, noté $P(L)$, est l'ensemble des initiales des mots de L, i.e. $P(L) = \{a \in \Sigma \mid a \cdot \Sigma^\star \cap L \neq \emptyset\}$*
- *L'**ensemble suffixe** de L, noté $S(L)$, est l'ensemble des lettres finales des mots de L, i.e. $S(L) = \{a \in \Sigma \mid \Sigma^\star \cdot a \cap L \neq \emptyset\}$*
- *L'**ensemble des 2-facteurs** de L, noté $F(L)$, est l'ensemble des 2-facteurs des mots de L, i.e. $F(L) = \{u \in \Sigma^2 \mid \Sigma^\star u \Sigma^\star \cap L \neq \emptyset\}$*
- *L'**ensemble des 2-non-facteurs** de L, noté $N(L)$, est l'ensemble mots de taille 2 qui ne sont facteurs d'aucun mot de L, i.e. $N(L) = \Sigma^2 \setminus F(L)$*
- *L'**ensemble des mots vides** de L, noté $\Lambda(L)$, est $\{\varepsilon\}$ ou $\emptyset$ selon que $\varepsilon \in L$ ou non, i.e. $\Lambda(L) = L \cap \{\varepsilon\}$*

*On définit alors le **langage local engendré** par L, noté $\rho(L)$, comme étant le langage des mots commençant par une lettre de $P(L)$, finissant par une lettre de $S(L)$ et dont tout 2-facteur se trouve dans $F(L)$, plus le mot vide s'il est déjà dans L. Plus précisément on définit $\rho(L)$ comme suit.*

$$\rho(L) = \Lambda(L) \sqcup \Big(\big(P(L)\cdot\Sigma^\star\big) \cap \big(\Sigma^\star \cdot S(L)\big)\Big) \setminus \big(\Sigma^\star \cdot N(L) \cdot \Sigma^\star\big)$$

Exercice de cours 1.3

Pour $L = \{a, aba\}$, donner $\rho(L)$.

Exercice de cours 1.4

Justifier que $\rho : (\mathscr{P}(\Sigma^\star), \subseteq) \to (\mathscr{P}(\Sigma^\star), \subseteq)$ est une fonction croissante.

Définition 1.5

*On appelle **langage local**, un langage L tel que $L = \rho(L)$.*

Remarque 1.6

Les structures (groupe, espace vectoriel, espace affines,...) engendrées par un ensemble S sont habituellement définies comme le plus petit ensemble contenant S ayant cette structure. On en déduit qu'un ensemble a telle ou telle structure si la structure engendrée par cet ensemble est lui-même.
C'est par analogie avec ces structures mathématiques qu'on a décidé d'appeler $\rho(L)$ le langage local engendré par L. En effet, les propriétés suivantes assurent que $\rho(L)$ est bien le plus petit langage local contentant L. Ce qui diffère ici, c'est qu'on a défini un langage local comme un langage égal au langage local qu'il engendre au lieu de commencer par définir la structure de langage local.

Proposition 1.7

Pour tout langage L, $L \subseteq \rho(L)$.

Démonstration : Montrons que $L \subseteq \rho(L)$. Soit $w \in L$, montrons que $w \in \rho(L)$.

- Si $w = \varepsilon$ alors $w \in L \cap \{\varepsilon\} = \Lambda(L) \subseteq \rho(L)$.
- Sinon $w = w_1 w_2 \dots w_n$ pour $n \in \mathbb{N}^*$.
  - $w \in (w_1 \cdot \Sigma^\star) \cap L$, donc $w_1 \in P(L)$, et finalement $w \in P(L) \cdot \Sigma^\star$.
  - De même $w \in \Sigma^\star \cdot S(L)$.
  - Pour tout $i \in [\![1, n-1]\!]$, $w \in (\Sigma^\star w_i w_{i+1} \Sigma^\star) \cap L$ donc $w_i w_{i+1} \in F(L)$. On en déduit que pour tout $i \in [\![1, n-1]\!]$, $w_i w_{i+1} \notin N(L)$, et finalement que $w \notin \Sigma^\star \cdot N(L) \cdot \Sigma^\star$.

  Ainsi $w \in \Big((P(L)\cdot\Sigma^\star) \cap (\Sigma^\star \cdot S(L))\Big) \setminus (\Sigma^\star \cdot N(L) \cdot \Sigma^\star)$

Dans les deux cas $w \in \rho(L)$, d'où $L \subseteq \rho(L)$. □

Proposition 1.8

Soit L un langage sur un alphabet Σ.
S'il existe $P \subseteq \Sigma$, $S \subseteq \Sigma$, $N \subseteq \Sigma^2$ et $\Lambda \subseteq \{\varepsilon\}$ tels que $L = \Lambda \sqcup \Big(\Big(P\cdot\Sigma^\star\Big) \cap \Big(\Sigma^\star \cdot S\Big)\Big) \setminus \Big(\Sigma^\star \cdot N \cdot \Sigma^\star\Big)$
alors $\rho(L) = L$.

Démonstration : Nous avons que $L \subseteq \rho(L)$, il nous suffit donc de montrer que $\rho(L) \subseteq L$, pour cela nous allons montrer que $S(L) \subseteq S$, $P(L) \subseteq P$, $F(L) \subseteq (\Sigma^2 \setminus N)$ et $\Lambda(L) \subseteq \Lambda$.

- Soit $a \in P(L)$, il existe donc $w \in \Sigma^\star$ tel que $w \in (a \cdot \Sigma^\star) \cap L$. En particulier $w \in L \setminus \{\varepsilon\}$ donc $w \in P \cdot \Sigma^\star$, et donc nécessairement $a \in P$, d'où $P(L) \subseteq P$.
- De même on montre que $S(L) \subseteq S$.
- Soit $ab \in F(L)$, il existe donc $w \in \Sigma^\star \cdot ab \cdot \Sigma^\star \cap L$. En particulier $w \in L \setminus \{\varepsilon\}$ donc $w \notin \Sigma^\star N \Sigma^\star$, donc nécessairement $ab \notin N$, soit $ab \in F(L)$ d'où $F(L) \subseteq \Sigma^2 \setminus N$.
- Si $\Lambda(L) = \emptyset$ alors $\Lambda(L) \subseteq \Lambda$. Sinon $\Lambda(L) = \{\varepsilon\}$, donc par définition de Λ, $\varepsilon \in L$, donc $\Lambda(L) \subseteq \Lambda$.

Finalement $S(L) \subseteq S$, $P(L) \subseteq P$, $N \subseteq N(L)$ et $\Lambda(L) \subseteq \Lambda$, aussi $\rho(L) \subseteq \Lambda \cup ((P\cdot\Sigma^\star) \cap (\Sigma^\star \cdot S)) \setminus (\Sigma^\star \cdot N \cdot \Sigma^\star)$, soit $\rho(L) \subseteq L$. □

Corollaire 1.9

Pour tout langage L, $\rho(\rho(L)) = \rho(L)$.

Démonstration : Il suffit d'appliquer la proposition 1.8 au langage $\rho(L)$, en utilisant les ensembles $P = P(L)$, $S = S(L)$, $F = F(L)$ et $\Lambda = \Lambda(L)$ pour établir que l'hypothèse est vérifiée. □

Remarque 1.10

Un langage L sur un alphabet Σ est local si et seulement s'il existe $P \subseteq \Sigma$, $S \subseteq \Sigma$, $N \subseteq \Sigma^2$ et $\Lambda \subseteq \{\varepsilon\}$ tels que $L = \Lambda \cup (P \cdot \Sigma^\star \cap \Sigma^\star \cdot S) \setminus (\Sigma^\star \cdot N \cdot \Sigma^\star)$.
En effet, par définition, si L est local, les ensembles $P(L), S(L), N(L)$ et $\Lambda(L)$ conviennent.
Réciproquement si de tels ensembles existent la proposition **??** assure que $\rho(L) = L$, *i.e.* L est local.
On trouve parfois cette définition de langage local.

Exemple 1.11
On se place sur l'alphabet $\Sigma = \{a, b\}$, alors :

- le langage $\{a\}$ est local ;
- le langage $\{a, b\}$ est local ;
- le langage $\mathcal{L}(a^\star)$ est local ;
- le langage $\mathcal{L}((ab)^\star)$ est local ;
- le langage $\{a, ab\}$ est local ;
- le langage $\{aa\}$ n'est pas local ;
- le langage $\mathcal{L}(a|(ab)^\star)$ n'est pas local ;
- le langage $\mathcal{L}(a \cdot (ab)^\star)$ n'est pas local.

Exercice de cours 1.12

Démontrer les points précédents.

1.1.2 Stabilités des langages locaux

Proposition 1.13

L'intersection de deux langages locaux, définis sur le même alphabet Σ, est un langage local.

Démonstration : Soit deux langages locaux L_1 et L_2 sur l'alphabet Σ.

$$L_1 = \Lambda(L_1) \cup ((P(L_1) \cdot \Sigma^\star) \cap (\Sigma^\star \cdot S(L_1))) \setminus (\Sigma^\star \cdot N(L_1) \cdot \Sigma^\star)$$
$$L_2 = \Lambda(L_2) \cup ((P(L_2) \cdot \Sigma^\star) \cap (\Sigma^\star \cdot S(L_2))) \setminus (\Sigma^\star \cdot N(L_2) \cdot \Sigma^\star)$$

Considérons alors $P = P(L_1) \cap P(L_2)$, $S = S(L_1) \cap S(L_2)$, $N = N(L_1) \cup N(L_2)$ et finalement $\Lambda = \Lambda(L_1) \cap \Lambda(L_2)$. Nous avons alors :

$$\begin{aligned}
L_1 \cap L_2 &= \Big(\Lambda(L_1) \cup ((P(L_1) \cdot \Sigma^\star) \cap (\Sigma^\star \cdot S(L_1))) \setminus (\Sigma^\star \cdot N(L_1) \cdot \Sigma^\star)\Big) \\
&\quad \cap \Big(\Lambda(L_2) \cup ((P(L_2) \cdot \Sigma^\star) \cap (\Sigma^\star \cdot S(L_2))) \setminus (\Sigma^\star \cdot N(L_2) \cdot \Sigma^\star)\Big) \\
&= (\Lambda(L_1) \cap \Lambda(L_2)) && \textit{i.e. } \Lambda \\
&\quad \cup (\Lambda(L_1) \cap (((P(L_2) \cdot \Sigma^\star) \cap (\Sigma^\star \cdot S(L_2))) \setminus \Sigma^\star \cdot N(L_2) \cdot \Sigma^\star)) && \text{vide} \\
&\quad \cup (\Lambda(L_2) \cap (((P(L_1) \cdot \Sigma^\star) \cap (\Sigma^\star \cdot S(L_1))) \setminus \Sigma^\star \cdot N(L_1) \cdot \Sigma^\star)) && \text{vide} \\
&\quad \cup (((P(L_1) \cdot \Sigma^\star \cap \Sigma^\star \cdot S(L_1)) \setminus \Sigma^\star \cdot N(L_1) \cdot \Sigma^\star) \cap ((P(L_2)\Sigma^\star \cap \Sigma^\star \cdot S(L_2)) \setminus \Sigma^\star \cdot N(L_2) \cdot \Sigma^\star)) \\
&= \Lambda \cup ((P(L_1)\Sigma^\star \cap \Sigma^\star S(L_1) \cap P(L_2)\Sigma^\star \cap \Sigma^\star S(L_2)) \setminus (\Sigma^\star N(L_1)\Sigma^\star) \cup \Sigma^\star N(L_2)\Sigma^\star) \\
&= \Lambda \cup (((P(L_1) \cap P(L_2))\Sigma^\star \cap \Sigma^\star(S(L_1) \cap S(L_2))) \setminus (\Sigma^\star(N(L_1) \cup N(L_2))\Sigma^\star)) \\
&= \Lambda \cup (P\Sigma^\star \cap \Sigma^\star S) \setminus \Sigma^\star N \Sigma^\star
\end{aligned}$$

D'après la proposition 1.8, le langage $L_1 \cap L_2$ est donc local. □

Proposition 1.14

Si L est un langage local alors $L^\star$ est un langage local.

Démonstration : Soit un langage L local. Par définition $L = \Lambda(L) \cup (((P(L)\cdot\Sigma^\star) \cap (\Sigma^\star\cdot S(L))) \setminus \Sigma^\star\cdot N(L)\cdot\Sigma^\star)$.
On pose alors $\Lambda_\star = \{\varepsilon\}$, $P_\star = P(L)$, $S_\star = S(L)$, $F_\star = F(L) \cup S(L)\cdot P(L)$ et $N_\star = \Sigma^2 \setminus F_\star$ et enfin $L_\star = \Lambda_\star \cup (((P_\star\cdot\Sigma^\star) \cap (\Sigma^\star\cdot S_\star)) \setminus \Sigma^\star N_\star \Sigma^\star)$. Montrons que $L^\star = L_\star$ par double inclusion.

$\subseteq$ Soit $w \in L^\star$. Si $w = \varepsilon$, comme $\varepsilon \in \Lambda_\star$ par définition de l'étoile de Klenne : $w \in L_\star$.
Sinon, il existe une famille de **mots** $(u^i)_{i\in[\![1,n]\!]} \in (L\setminus\{\varepsilon\})^n$ avec $n \in \mathbb{N}^*$ telle que $w = u^1\cdot u^2 \dots u^n$.
- En particulier $u^1 \in L\setminus\{\varepsilon\}$, donc $u^1 \in P(L)\cdot\Sigma^\star$. Comme $w \in u^1\cdot\Sigma^\star$, on en déduit que $w \in (P(L)\cdot\Sigma^\star)\cdot\Sigma^\star = P(L)\cdot\Sigma^\star$.
- Mais aussi $u^n \in L\setminus\{\varepsilon\}$, donc $u^n \in \Sigma^\star\cdot S(L)$, on en déduit de même $w \in \Sigma^\star\cdot S(L)$.
- Enfin on montre que $w \notin \Sigma^\star\cdot N_\star\cdot\Sigma^\star$ par l'absurde. Supposons donc que $w \in \Sigma^\star\cdot N_\star\cdot\Sigma^\star$. Deux cas sont alors possibles selon que le facteur de $N_\star$ considéré apparaît dans l'un des u^i ou au contraire à cheval sur deux mots de $(u^i)_{i\in[\![1,n]\!]}$.
  - Soit il existe $i \in [\![1,n]\!]$ tel que $u^i \in \Sigma^\star\cdot N_\star\cdot\Sigma^\star$, or $N_\star \subseteq N(L)$ donc $u^i \in \Sigma^\star\cdot N(L)\cdot\Sigma^\star$. Or comme $u^i \in L\setminus\{\varepsilon\}$, $u^i \notin \Sigma^\star\cdot N(L)\cdot\Sigma^\star$. Absurde.
  - Soit il existe $i \in [\![1,n-1]\!]$ et $(a,b) \in \Sigma\times\Sigma$ tels que $u^i = \underline{u^i}\cdot a$, $u^{i+1} = b\cdot\overline{u^{i+1}}$ et $ab \in N_\star$. Comme $u^i \in L \setminus \{\varepsilon\}$, $u^i \in \Sigma^\star\cdot S(L)$, donc $a \in S(L)$. De même, $u^{i+1} \in L \setminus \{\varepsilon\}$, donc $u^{i+1} \in P(L)\cdot\Sigma^\star$ et $b \in P(L)$. Ainsi $ab \in S(L)\cdot P(L) \subseteq F_\star$, soit $ab \notin N_\star$. Absurde.

  Finalement $w \notin \Sigma^\star\cdot N_\star\cdot\Sigma^\star$.

Finalement $w \in ((P(L)\cdot\Sigma^\star) \cap (\Sigma^\star\cdot S(L))) \setminus (\Sigma^\star\cdot N_\star\cdot\Sigma^\star) \subseteq L_\star$. D'où $L^\star \subseteq L_\star$.

$\supseteq$ Soit $w \in L_\star$ alors $w \in \Lambda_\star \cup ((P_\star\cdot\Sigma^\star) \cap (\Sigma^\star\cdot S_\star)) \setminus (\Sigma^\star\cdot N_\star\cdot\Sigma^\star)$.
Si $w = \varepsilon$ alors $w \in L^\star$ par définition de l'étoile de Kleene. Sinon, $w \in ((P_\star\cdot\Sigma^\star) \cap (\Sigma^\star\cdot S_\star)) \setminus (\Sigma^\star\cdot N_\star\cdot\Sigma^\star)$. Par définition de $L_\star$, les 2-facteurs de w sont dans $F_\star = F(L) \cup (S(L)\cdot P(L))$. Pour se ramener à des mots dont les 2-facteurs sont dans $F(L)$ seulement, on va couper w au niveau de chaque 2-facteur de $S(L)\cdot P(L)$. Plus précisément, si w contient deux lettres consécutives w_j et w_{j+1} telles que $w_j \in S(L)$ et $w_{j+1} \in P(L)$, on décompose w en deux facteurs : $w = w_{|[\![0,j]\!]}\cdot w_{|[\![j+1,|w|-1]\!]}$. On remarque que ces deux facteurs, comme w avant eux, commencent par une lettre de $P(L)$ et finissent par une de $S(L)$. On itère ce processus sur les deux facteurs obtenus tant qu'ils contiennent des 2-facteurs de $S(L)\cdot P(L)$. On obtient finalement une factorisation de w en $u^1\cdot u^2 \dots u^n$ telle que $\forall i \in [\![1,n]\!], u^i \in P(L)\cdot\Sigma^\star$, $u^i \in \Sigma^\star S(L)$ et $u^i \notin \Sigma^\star(S(L)\cdot P(L))\cdot\Sigma^\star$, dont on déduit $u^i \notin \Sigma^\star\cdot F(L)\cdot\Sigma^\star$ puisque w n'avait pas de 2-facteurs dans $F_\star(L)$. Autrement dit, pour chaque $i \in [\![1,n]\!]$, $u^i \in L$, donc $w \in L^n \subseteq L^\star$, d'où $L_\star \subseteq L^\star$.

□

Proposition 1.15

Soient L_1 un langage local sur un alphabet Σ_1 et L_2 un langage local sur un alphabet Σ_2 tels que $\Sigma_1 \cap \Sigma_2 = \emptyset$, alors $L_1 \cup L_2$ est un langage local sur l'alphabet $\Sigma_1 \sqcup \Sigma_2$.

Démonstration : Soient deux langages locaux L_1 et L_2 sur des alphabets Σ_1 et Σ_2 disjoints.

$$L_1 = \Lambda(L_1) \cup \Big(((P(L_1)\cdot\Sigma_1^\star \cap \Sigma_1^\star\cdot S(L_1)) \setminus (\Sigma_1^\star\cdot N(L_1)\cdot\Sigma_1^\star)\Big)$$
$$L_2 = \Lambda(L_2) \cup \Big(((P(L_2)\cdot\Sigma_2^\star \cap \Sigma_2^\star\cdot S(L_2)) \setminus (\Sigma_2^\star\cdot N(L_2)\cdot\Sigma_2^\star)\Big)$$

On pose alors : $\Sigma = \Sigma_1 \sqcup \Sigma_2$, $\Lambda_\cup = \Lambda(L_1) \cup \Lambda(L_2)$, $P_\cup = P(L_1) \sqcup P(L_2)$, $S_\cup = S(L_1) \sqcup S(L_2)$, $F_\cup = F(L_1) \sqcup F(L_2)$, $N_\cup = (\Sigma^2 \setminus F_\cup)$, et enfin $L_\cup = \Lambda_\cup \cup (P_\cup\cdot\Sigma^\star \cap \Sigma^\star\cdot S_\cup) \setminus (\Sigma^\star\cdot N_\cup\cdot\Sigma^\star)$. Montrons que $L_\cup = L_1 \sqcup L_2$.

$\subseteq$ Soit $w \in L_\cup$.
- Si $w = \varepsilon$, alors nécessairement $w \in \Lambda_\cup$, or $\Lambda_\cup = \Lambda(L_1) \cup \Lambda(L_2)$, donc $\varepsilon \in \Lambda(L_1) \subseteq L_1$ ou $\varepsilon \in \Lambda(L_2) \subseteq L_2$, d'où $w = \varepsilon \in L_1 \cup L_2$.
- Sinon, $w \in (P_\cup\cdot\Sigma^\star \cap \Sigma^\star\cdot S_\cup) \setminus (\Sigma^\star\cdot N_\cup\cdot\Sigma^\star)$. Notons alors $w_1w_2 \dots w_n$ les lettres de w et distinguons deux cas selon w_1.
  - Si $w_1 \in \Sigma_1$. Puisque $w \in P_\cup\cdot\Sigma^\star$, $w_1 \in P_\cup$, or $P_\cup \cap \Sigma_1 = P(L_1)$ car $P(L_2) \cap \Sigma_1 \subseteq \Sigma_2 \cap \Sigma_1 = \emptyset$. Donc $w_1 \in P(L_1)$. Puisque $w \notin \Sigma^\star\cdot N_\cup\cdot\Sigma^\star$, $w_1w_2 \in \Sigma^2 \setminus N_\cup$ soit $w_1w_2 \in F(L_\cup) = F(L_1) \cup F(L_2)$, et $F(L_2) \cap \Sigma_1 = \emptyset$ donc $w_1w_2 \in F(L_1)$, et donc $w_2 \in \Sigma_1$. Par une récurrence triviale, $\forall i \in [\![1,n]\!]$, $w_i \in$

Σ_1 et $\forall i \in [\![1, n-1]\!]$, $w_i w_{i+1} \in F(L_1)$. En particulier $w_n \in \Sigma_1$, or $w_n \in S_\cup = S(L_1) \cup S(L_2)$, donc $w_n \in S(L_1)$. Finalement $w \in L_1$.

- Si $w_1 \in \Sigma_2$ on fait de même et on conclut que $w \in L_2$.

Ainsi dans tous les cas $w \in L_1 \cup L_2$, d'où $L_\cup \subseteq L_1 \cup L_2$.

$\supseteq$ Par construction $\Lambda_\cup \supseteq \Lambda_{L_1}$, $P_\cup \supseteq P_{L_1}$, $S_\cup \supseteq S_{L_1}$, $N_\cup \subseteq N_{L_1}$, et $\Sigma \subseteq \Sigma_1$. On en déduit que $L_\cup \supseteq L_1$. De même $L_\cup \supseteq L_2$, d'où $L_1 \cup L_2 \subseteq L_\cup$.

□

Exercice de cours 1.16

Donner des exemples de deux langages locaux L_1 et L_2 tels que $L_1 \cup L_2$ n'est pas un langage local.

Proposition 1.17

Soient L_1 un langage local sur un alphabet Σ_1 et L_2 un langage local sur un alphabet Σ_2 ***tels que*** *$\Sigma_1 \cap \Sigma_2 = \emptyset$, alors $L_1 \cdot L_2$ est un langage local sur l'alphabet $\Sigma_1 \sqcup \Sigma_2$.*

Démonstration : Soit deux langages locaux L_1 et L_2 sur des alphabets Σ_1 et Σ_2 disjoints.

$$L_1 = \Lambda(L_1) \cup \Big(((P(L_1)\cdot\Sigma_1^\star \cap \Sigma_1^\star \cdot S(L_1)) \setminus (\Sigma_1^\star \cdot N(L_1) \cdot \Sigma_1^\star)\Big)$$
$$L_2 = \Lambda(L_2) \cup \Big(((P(L_2)\cdot\Sigma_2^\star \cap \Sigma_2^\star \cdot S(L_2)) \setminus (\Sigma_2^\star \cdot N(L_2) \cdot \Sigma_2^\star)\Big)$$

On pose alors $\Sigma = \Sigma_1 \sqcup \Sigma_2$, $\Lambda_c = \Lambda(L_1) \cap \Lambda(L_2)$, $F_c = F(L_1) \cup F(L_2) \cup S(L_1)\cdot P(L_2)$, $N_c = (\Sigma^2 \setminus F_c)$,

$$P_c = \begin{cases} P(L_1) & \text{si } \Lambda(L_1) = \emptyset \\ P(L_1) \cup P(L_2) & \text{si } \Lambda(L_1) = \{\varepsilon\} \end{cases}$$

et enfin $L_c = \Lambda_c \cup (P_c\Sigma^\star \cap \Sigma^\star S_c) \setminus (\Sigma^\star \cdot N_c \cdot \Sigma^\star)$. Montrons que $L^c = L_1 \cdot L_2$.

$\subseteq$ Soit $w \in L_c$.

- Si $w = \varepsilon$ alors $w \in \Lambda_c = \Lambda(L_1) \cap \Lambda(L_2)$, donc $\varepsilon \in \Lambda(L_1)$ et $\varepsilon \in \Lambda(L_2)$, et donc $w = \varepsilon = \varepsilon\cdot\varepsilon \in L_1 \cdot L_2$.
- Sinon,
  - Supposons que la première lettre de w est dans Σ_1 et la dernière lettre de w est dans Σ_2, alors w se décompose en $w = u \cdot v$ de sorte que toutes les lettres de u soient dans Σ_1 et la première lettre de v soit dans Σ_2 (en effet il suffit de couper juste avant la première lettre de w qui est dans Σ_2). En notant $u = u_1u_2\dots u_n$ et $v = v_1v_2\dots v_m$ les décompositions en lettres de ces facteurs, alors $\forall i \in [\![1, n]\!]$, $u_i \in \Sigma_1$, $v_1 \in \Sigma_2$ et $v_m = w_{|w|} \in \Sigma_2$. De plus puisque $w \notin \Sigma^\star \cdot N_c \cdot \Sigma^\star$, tous les 2-facteurs de u et v sont dans $F(c)$.
    Puisque $v_1 \in \Sigma_2$ et $v_1v_2 \in F_c = F(L_1) \cup F(L_2) \cup S(L_1)\cdot P(L_2)$, nécessairement $v_1v_2 \in F(L_2)$ et donc $v_2 \in \Sigma_2$. Par une récurrence triviale on montrerait de même que $\forall i \in [\![1, m]\!]$, $v_i \in \Sigma_2$ et $\forall i \in [\![1, m-1]\!]$, $v_iv_{i+1} \in F(L_2)$. De plus, pour tout $i \in [\![1, n-1]\!]$ $u_i \in \Sigma_1$ et $u_iu_{i+1} \in F_c$ donc de même $u_iu_{i+1} \in F(L_1)$.
    Par ailleurs $v_m = w_{|w|} \in S_c = S(L_2) \cup \Lambda(L_2)\cdot S(L_1)$, et comme $v_m \in \Sigma_2$, nécessairement $v_m \in S(L_2)$. Enfin $u_nv_1 \in F_c$ et comme $u_n \in \Sigma_1$ et $v_0 \in \Sigma_2$, nécessairement $u_nv_1 \in S(L_1)\cdot P(L_2)$, soit $u_n \in S(L_1)$ et $v_1 \in P(L_2)$.
    On déduit des ces remarques que $u \in L_1$ et $v \in L_2$, et donc que $w = u\cdot v \in L_1 \cdot L_2$.
  - Si la première et la dernière lettre de w sont dans Σ_1, alors on peut montrer par l'absurde que toutes les lettres de w sont dans Σ_1. En effet si w a une lettre de Σ_2, on peut montrer comme précédemment ♣ que toutes les lettres qui suivent sont aussi dans Σ_2, et en particulier la dernière. Absurde.
    Avec $n = |w|$, $\forall i \in [\![1, n]\!]$, $w_i \in \Sigma_1$ et donc $\forall i \in [\![0, n-1]\!], w_iw_{i+1} \in F(c) \cap (\Sigma_1\cdot\Sigma_1) = F(L_1)$. En particulier $w_n \in \Sigma_1$ donc, par disjonction des alphabets, $w_n \notin \Sigma_2$. Par ailleurs $w \in \Sigma^\star \cdot S(c)$, donc

♣. par récurrence, parce que les 2-facteurs de w commençant par une lettre de Σ_2 ont leur seconde lettre dans Σ_2.

$w_n \in S(c)$ or $S(c) = S(L_2) \cup \Lambda(L_2)\cdot S(L_1)$ et $S(L_2) \subseteq \Sigma_2$, donc $w_n \in \Lambda(L_2)\cdot S(L_1)$. Cela assure en particulier que $\Lambda(L_2) \neq \emptyset$, donc nécessairement $\Lambda(L_2) = \{\varepsilon\}$ et alors $w_n \in \{\varepsilon\}\cdot S(L_1) = S(L_1)$. Finalement $w \in L_1$ et $\varepsilon \in L_2$ donc $w = w\cdot\varepsilon \in L_1\cdot L_2$.
- De même si la première et la dernière lettre de w sont dans Σ_2, on montre que $w \in L_2$ et $\varepsilon \in L_1$ donc $w = \varepsilon\cdot w \in L_1\cdot L_2$.
- Le dernier cas est absurde car $F_c \cap \Sigma_2\cdot\Sigma_1 = \emptyset$, autrement dit aucun des deux facteurs possibles pour w ne permet de passer de lettre de Σ_2 à Σ_1.

$\supseteq$ Soit $w \in L_1\cdot L_2$. Par définition de $\cdot$ il existe alors $u = u_1u_2\dots u_n \in L_1$ et $v_1v_2\dots v_m \in L_2$ tels que $w = u\cdot v$.
- Si $n \neq 0$ et $m \neq 0$ alors :
  - $u_1 \in P(L_1)$ et donc $w_1 = u_1 \in P$.
  - $v_m \in S(L_2)$ et donc $w_{|w|} = v_m \in S$.
  - $\forall i \in [\![1, n-1]\!], u_iu_{i+1} \in F(L_1)$ et donc $\forall i \in [\![1, n-1]\!], w_iw_{i+1} = u_iu_{i+1} \in F$.
  - $\forall i \in [\![1, m-1]\!], v_iv_{i+1} \in F(L_2)$ et donc $\forall i \in [\![n+1, n+m-1]\!], w_iw_{i+1} = v_{i-n}v_{i+1-n} \in F$.
  - $u_n \in S(L_1)$ et $v_1 \in P(L_2)$ donc $u_nv_1 \in S(L_1)P(L_2)$ et donc $w_nw_{n+1} = u_nv_1 \in F$.

  Ainsi $w \in L$.
- Si $n \neq 0$ et $m = 0$ alors $\varepsilon = v \in L_2$ donc $\Lambda(L_2) = \{\varepsilon\}$. De plus :
  - $u_1 \in P(L_1)$ et donc $w_1 = u_1 \in P$.
  - $u_n \in S(L_1) = \{\varepsilon\}\cdot S(L_1)$ et donc $w_{|w|} = u_n \in \Lambda(L_2)\cdot S(L_1) \subseteq S$.
  - $\forall i \in [\![1, n-1]\!], u_iu_{i+1} \in F(L_1)$ et donc $\forall i \in [\![1, n-1]\!], w_iw_{i+1} = u_iu_{i+1} \in F$.

  Ainsi $uv \in .$
- Les autres cas sont identiques

□

Exercice de cours 1.18

Donner des exemples de deux langages locaux L_1 et L_2 tels que $L_1\cdot L_2$ n'est pas un langage local.

1.2 Expressions régulières linéaires

1.2.1 Définition

Définition 1.19

Une expression régulière sur un alphabet Σ est dite ***linéaire*** *si chaque lettre de Σ apparaît au plus une fois dans l'expression régulière. On peut donner une définition inductive du prédicat linéaire(e) testant si une expression régulière e est linéaire :*

$$\begin{aligned}
\textit{linéaire}(\emptyset) &= \textit{vrai}\\
\textit{linéaire}(\varepsilon) &= \textit{vrai}\\
\forall a \in \Sigma,\ \textit{linéaire}(a) &= \textit{vrai}\\
\forall (e, e') \in \mathscr{E}_{reg}(\Sigma)^2,\ \textit{linéaire}(e \mid e') &= \textit{linéaire}(e)\ \textit{et}\ \textit{linéaire}(e')\ \textit{et}\ \Big(\textit{vars}(e) \cap \textit{vars}(e') = \emptyset\Big)\\
\textit{linéaire}(e\cdot e') &= \textit{linéaire}(e)\ \textit{et}\ \textit{linéaire}(e')\ \textit{et}\ \Big(\textit{vars}(e) \cap \textit{vars}(e') = \emptyset\Big)\\
\textit{linéaire}(e^\star) &= \textit{linéaire}(e)
\end{aligned}$$

Exercice de cours 1.20

Définir une fonction OCaml `lineaire : regexp -> bool` permettant de tester si une expression régulière est linéaire. On choisira le type `string` pour la représentation des variables. Quelle est la complexité de votre fonction ? Assurez vous d'obtenir une complexité en $\mathcal{O}(n\log(n))$ où n est la taille de l'expression régulière.

1.2.2 Localité

Corollaire 1.21

Le langage associé à une expression régulière linéaire est local.

Démonstration : Cette proposition est un corollaire des propositions de stabilité qui précèdent (stabilité par étoile, par union et concaténation pourvu que les alphabets soient disjoints). Il nous suffit en fait de montrer la partie la plus simple de la proposition, c'est-à-dire les cas de base.

- Le langage vide est local : $\Lambda = \emptyset$, $P = \emptyset$, $S = \emptyset$, $F = \emptyset$
- Le langage réduit au mot vide est local : $\Lambda = \{\varepsilon\}$, $P = \emptyset$, $S = \emptyset$, $F = \emptyset$
- Le langage réduit à un mot lui-même réduit à une lettre $a \in \Sigma$: $\Lambda = \emptyset$, $P = \{a\}$, $S = \{a\}$, $F = \emptyset$

□

Proposition 1.22

*Si e est une expression régulière **linéaire**, on définit par induction sur e les ensembles P, S, F, Λ de la manière suivante.*

e	$P(e)$	$S(e)$	$F(e)$	$\Lambda(e)$
$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
ε	$\emptyset$	$\emptyset$	$\emptyset$	$\{\varepsilon\}$
$a \in \Sigma$	$\{a\}$	$\{a\}$	$\emptyset$	$\emptyset$
$e \mid e'$	$P(e) \cup P(e')$	$S(e) \cup S(e')$	$F(e) \cup F(e')$	$\Lambda(e) \cup \Lambda(e')$
$e \cdot e'$	$P(e) \cup \Lambda(e) \cdot P(e')$	$S(e') \cup \Lambda(e') \cdot S(e)$	$F(e) \cup F(e') \cup S(e) \cdot P(e')$	$\Lambda(e) \cap \Lambda(e')$
$e^\star$	$P(e)$	$S(e)$	$F(e) \cup S(e) \cdot P(e)$	$\{\varepsilon\}$

Alors $\mathcal{L}(e) = \Lambda(e) \cup \Big(\big(P(e) \cdot \Sigma^\star \cap \Sigma^\star \cdot S(e) \big) \setminus \big(\Sigma^\star . N(e) \cdot \Sigma^\star \big) \Big)$

Démonstration : Corollaire des preuves qui précèdent. □

Exercice de cours 1.23

1. Définir une fonction OCaml `lpsf : regexp -> bool * string list * string list * string list` prenant en argument une expression régulière linéaire e et calculant le quadruplet $\Lambda(e), P(e), S(e), F(e)$.
2. Calculer $\Lambda(e)$, $P(e)$, $S(e)$, $F(e)$ pour e l'expression régulière $(a \cdot b | c \cdot \emptyset)^\star \cdot (d \mid \varepsilon)$.

1.2.3 Linéarisation d'expressions régulières

Si Σ_2 et Σ_2 sont deux alphabets, et $\varphi : \Sigma_1 \to \Sigma_2$ une fonction de l'un dans l'autre, on définit $\widetilde{\varphi} : \Sigma_1^\star \to \Sigma_2^\star$ son extension aux mots par $\forall w = w_1 w_2 \dots w_n \in \Sigma_1^\star, \widetilde{\varphi}(w) = \varphi(w_1)\varphi(w_2)\dots\varphi(w_n)$. Si L est un langage sur Σ_1, $\widetilde{\varphi}(L)$ désigne alors l'ensemble image de L par $\widetilde{\varphi}$ *i.e.* $\widetilde{\varphi}(L) = \{\widetilde{\varphi}(w) \mid w \in L\}$.

Remarque 1.24

Si F et G sont deux langages sur Σ_1, alors $\widetilde{\varphi}(F \cup G) = \widetilde{\varphi}(F) \cup \widetilde{\varphi}(G)$ et $\widetilde{\varphi}(F \cdot G) = \widetilde{\varphi}(F) \cdot \widetilde{\varphi}(G)$. Le premier point découle de la définition de l'ensemble image tandis que le second est démontré par les équivalences suivantes.

$$\begin{aligned} w \in \widetilde{\varphi}(F \cdot G) &\Leftrightarrow \exists u \in F \cdot G,\ w = \widetilde{\varphi}(u) \\ &\Leftrightarrow \exists f \in F, \exists g \in G,\ w = \widetilde{\varphi}(f \cdot g) \\ &\Leftrightarrow \exists f \in F, \exists g \in G,\ w = \widetilde{\varphi}(f) \cdot \widetilde{\varphi}(g) \\ &\Leftrightarrow \exists f' \in \widetilde{\varphi}(F), \exists g' \in \widetilde{\varphi}(G),\ w = f' \cdot g' \\ &\Leftrightarrow w \in \widetilde{\varphi}(F) \cdot \widetilde{\varphi} \end{aligned}$$

Définition 1.25

Soient Σ et Σ' deux alphabets. Soit $\varphi : \Sigma \to \Sigma'$.
*La **transformation selon φ** d'une expression régulière e, notée e_φ, est définie inductivement sur $\mathscr{E}_{reg}(\Sigma)$ comme suit.*

$$\begin{aligned} \emptyset_\varphi &= \emptyset \\ \varepsilon_\varphi &= \varepsilon \\ \forall a \in \Sigma,\ a_\varphi &= \varphi(a) \\ \forall (f,g) \in \mathscr{E}_{reg}(\Sigma)^2,\ (f \mid g)_\varphi &= f_\varphi \mid g_\varphi \\ (f \cdot g)_\varphi &= f_\varphi \cdot g_\varphi \\ (f^\star)_\varphi &= (f_\varphi)^\star \end{aligned}$$

Proposition 1.26 (*Linéarisation d'expression régulière*)

Pour toute expression régulière e sur un alphabet Σ, il existe une expression régulière linéaire l sur un alphabet $\overline{\Sigma}$ et une fonction $\varphi : \overline{\Sigma} \to \Sigma$ telles que $e = l_\varphi$.

Démonstration : Puisque dans une expression linéaire les lettres doivent être deux à deux distinctes, on se donne un alphabet qui compte une *super*-lettre pour chaque occurrence de lettre dans e. L'expression régulière étant finie, c'est-à-dire ayant un nombre fini de lettres même comptées avec multiplicité, ce nouvel alphabet sera bien fini. Concrètement ces *super*-lettres peuvent être des couples (lettre, n° d'occurrence), formant ainsi $\overline{\Sigma} \subseteq \Sigma \times [\![0, N]\!]$ où N est la taille de e. Il suffit alors de considérer φ la projection sur la première composante *i.e.* $\varphi : \overline{\Sigma} \to \Sigma, (a, \bullet) \mapsto a$. □

Exemple 1.27

Considérons par exemple l'expression régulière $e = c^\star((aa)|\varepsilon)(b((a|c|\varepsilon)^\star))^\star baa^\star$. On lui associe l'expression régulière linéaire $l = c_1^\star((a_1a_2)|\varepsilon)(b_1((a_3 + c_2 + \varepsilon)^\star))^\star b_2a_4a_5^\star$ et le morphisme $\varphi = (a_1 \mapsto a, a_2 \mapsto a, a_3 \mapsto a, a_4 \mapsto a, a_5 \mapsto a, b_1 \mapsto b, b_2 \mapsto b, c_1 \mapsto c, c_2 \mapsto c)$ qui est tel que : $l_\varphi = e$.

Exercice de cours 1.28

Linéariser l'expression $(a \cdot a | b \cdot \emptyset)^\star \cdot (b \mid \varepsilon)$.

Proposition 1.29

Sous les notations de la définition précédente, $\mathcal{L}(e_\varphi) = \widetilde{\varphi}(\mathcal{L}(e))$.

Démonstration : On le démontre par induction sur $e \in \mathscr{E}_{reg}(\Sigma)$.

- cas $e = \emptyset$: $\mathcal{L}(\emptyset_\varphi) = \mathcal{L}(\emptyset) = \emptyset = \widetilde{\varphi}(\emptyset) = \widetilde{\varphi}(\mathcal{L}(\emptyset))$

- cas $e = \varepsilon$: $\mathcal{L}(\varepsilon_\varphi) = \mathcal{L}(\varepsilon) = \{\varepsilon\} = \widetilde{\varphi}(\{\varepsilon\}) = \widetilde{\varphi}(\mathcal{L}(\varepsilon))$
- cas $e = a \in \Sigma$: $\mathcal{L}(a_\varphi) = \mathcal{L}(\varphi(a)) = \{\varphi(a)\} = \widetilde{\varphi}(\{a\}) = \widetilde{\varphi}(\mathcal{L}(a))$
- cas $e = f|g$:

$$\begin{aligned}\mathcal{L}((f|g)_\varphi) &= \mathcal{L}(f_\varphi|g_\varphi)\\ &= \mathcal{L}(f_\varphi) \cup \mathcal{L}(g_\varphi)\\ &= \widetilde{\varphi}(\mathcal{L}(f)) \cup \widetilde{\varphi}(\mathcal{L}(g))\\ &= \widetilde{\varphi}(\mathcal{L}(f) \cup \mathcal{L}(g))\\ &= \widetilde{\varphi}(\mathcal{L}(f|g))\end{aligned}$$

- cas $e = (f^\star)_\varphi$:

$$\begin{aligned}\mathcal{L}((f^\star)_\varphi) &= \mathcal{L}((f_\varphi)^\star)\\ &= \{w_1w_2\dots w_n \mid \forall i \in [\![1,n]\!], w_i \in \mathcal{L}(f_\varphi)\}\\ &= \{w_1w_2\dots w_n \mid \forall i \in [\![1,n]\!], w_i \in \widetilde{\varphi}(\mathcal{L}(f))\}\\ &= \{\widetilde{\varphi}(u_1)\widetilde{\varphi}(u_2)\dots\widetilde{\varphi}(u_n) \mid \forall i \in [\![1,n]\!], u_i \in \mathcal{L}(f)\}\\ &= \widetilde{\varphi}(\{u_1u_2\dots u_n \mid \forall i \in [\![1,n]\!], u_i \in \mathcal{L}(f)\})\\ &= \widetilde{\varphi}(\mathcal{L}(f^\star))\end{aligned}$$

- cas $e = f\cdot g \in \Sigma$:

$$\begin{aligned}\mathcal{L}((f\cdot g)_\varphi) &= \mathcal{L}(f_\varphi\cdot g_\varphi)\\ &= \mathcal{L}(f_\varphi)\cdot\mathcal{L}(g_\varphi)\\ &= \widetilde{\varphi}(\mathcal{L}(f))\cdot\widetilde{\varphi}(\mathcal{L}(g))\\ &= \widetilde{\varphi}(\mathcal{L}(f)\cdot\mathcal{L}(g))\\ &= \widetilde{\varphi}(\mathcal{L}(f\cdot g))\end{aligned}$$

□

Proposition 1.30

Soit $\mathcal{A} = (\Sigma, Q, I, F, \delta)$ un automate (sans ε-transition). Soit $\varphi : \Sigma \to \Sigma'$.
L'automate $\mathcal{A}_\varphi = (\Sigma', Q, I, F, \delta_\varphi)$ où $\delta_\varphi = \{(q, \varphi(a), q') \mid (q, a, q') \in \delta\}$ reconnaît $\widetilde{\varphi}\Big(\mathcal{L}(\mathcal{A})\Big)$.

Démonstration : Pour tout $n \in \mathbb{N}$ et tout $w \in (\Sigma')^n$:

$$\begin{aligned}w = w_1w_2\dots w_n \in \mathcal{L}(\mathcal{A}_\varphi) &\Leftrightarrow \exists (q_i)_{i\in[\![0,n]\!]} \in Q^n, \text{ tel que } q_0 \xrightarrow{w_1} q_1 \xrightarrow{w_2} q_2 \dots \xrightarrow{w_n} q_n \text{ une exéc. acceptante de } \mathcal{A}_\varphi\\ &\Leftrightarrow \exists (q_i)_{i\in[\![0,n]\!]} \in Q^n, \exists (u_i)_{i\in[\![1,n]\!]} \in \Sigma^n \text{ tels que } \forall i \in [\![1,n]\!],\ w_i = \varphi(u_i)\\ &\quad \text{et } q_0 \xrightarrow{u_1} q_1 \xrightarrow{u_2} q_2 \dots \xrightarrow{u_n} q_n \text{ est une exéc. acceptante de } \mathcal{A}\\ &\Leftrightarrow \exists (u_i)_{i\in[\![1,n]\!]} \in \Sigma^n \text{ tel que } \forall i \in [\![1,n]\!],\ w_i = \varphi(u_i)\\ &\quad \text{et } u_1u_2\dots u_n \in \mathcal{L}(\mathcal{A})\\ &\Leftrightarrow \exists u \in \Sigma^n \text{ tel que } w = \widetilde{\varphi}(u) \text{ et } u \in \mathcal{L}(\mathcal{A})\\ &\Leftrightarrow w \in \widetilde{\varphi}(\mathcal{L}(\mathcal{A}))\end{aligned}$$

□

1.3 Automates locaux

Définition 1.31

*Un automate déterministe $\mathcal{A} = (\Sigma, Q, \{q_0\}, F, \delta)$ est dit **local** lorsque :*

$$\forall a \in \Sigma, \mathrm{card}\{q' \in Q \mid \exists q \in Q, (q, a, q') \in \delta\} \leqslant 1$$

*Il est de plus dit **standard** lorsque $\forall (q, a, q') \in \delta, q' \neq q_0$.*

Ainsi un automate est dit local lorsque l'état dans lequel il se trouve après la lecture d'une lettre dépend uniquement de la lettre lue et non pas de l'état dans lequel il était. Un automate local est dit standard si aucune transition ne conduit à l'état initial.

Exercice de cours 1.32

Donner un automate local standard reconnaissant le langage des mots w sur l'alphabet $\Sigma = \{a, b, c\}$ tels que $\forall i \in [\![0, |w| - 1]\!], (w_i = b \Rightarrow w_{i+1} = c) \wedge (w_i = c \Rightarrow w_{i+1} = a)$.

Proposition 1.33 (*"proto théorème de Kleene"*)

Un langage est local si et seulement s'il est reconnu par un automate standard local.

Démonstration : Montrons les deux sens de l'implication.

$\Rightarrow$ Soit L un langage local sur un alphabet Σ. On considère l'automate $\mathcal{A} = (\Sigma, Q, I, F, \delta)$ où

$$Q = \{\varepsilon\} \cup \Sigma \quad I = \{\varepsilon\} \quad F = \Lambda(L) \cup S(L)$$

$$\delta = \{(q, a, a) \,|\, a \in \Sigma, q \in \Sigma,\ qa \in F(L)\} \sqcup \{(\varepsilon, a, a) \,|\, a \in P(L)\}$$

Par construction $\mathcal{A}$ est un automate déterministe local standard.
Montrons qu'il reconnaît L, *i.e.* que $\mathcal{L}(\mathcal{A}) = L$, par double inclusion.

$\subseteq$ Soit $q_0 \xrightarrow{w_1} q_1 \xrightarrow{w_2} q_2 \dots \xrightarrow{w_n} q_n$ une exécution acceptante de $\mathcal{A}$.
Par définition $q_0 \in I$, soit ici $q_0 = \varepsilon$, et $q_n \in F$, soit ici $q_n \in \Lambda(L) \cup S(L)$.
- Si $n = 0$, alors $\varepsilon = q_0 = q_n \in \Lambda(L) \cup S(L)$, or $S(L) \subseteq \Sigma^1$, donc $\varepsilon \notin S(L)$, donc $\varepsilon \in \Lambda(L) \subseteq L$, soit $w \in L$.
- Sinon :
  - $(q_0, w_1, q_1) = (\varepsilon, w_1, q_1) \in \delta$, donc par définition de δ, $w_1 \in P(L)$.
  - $(q_{n-1}, w_{n-1}, q_n) \in \delta$, donc par définition de δ, $w_{n-1} = q_n \neq \varepsilon$, or $q_n \in \Lambda(L) \cup S(L)$, donc $w_{n-1} \in S(L)$.
  - Pour tout $i \in [\![1, n-1]\!]$ $(q_{i-1}, w_i, q_i) \in \delta$, donc par définition de δ, $w_i = q_i \neq \varepsilon$ (et $q_{i-1} w_i \in F(L)$). De plus, puisque $(q_i, w_{i+1}, q_{i+1}) \in \delta$ avec $q_i \neq \varepsilon$, par définition de δ, $w_{i+1} = q_{i+1}$ et $q_i w_{i+1} \in F(L)$, soit $w_i w_{i+1} \in F(L)$.

  Ces trois points montrent que w commence par une lettre de $P(L)$, finit par une lettre de $S(L)$, et n'admet que des 2-facteurs appartenant à $F(L)$, donc $w \in \rho(L) = L$ (car L est local).

Dans les deux cas $w \in L$. D'où $\mathcal{L}(\mathcal{A}) \subseteq L$.

$\supseteq$ Réciproquement, soit $w_1 w_2 \dots w_n \in L$.
- Si $n = 0$, alors $\varepsilon = w \in L$, donc par définition de $\Lambda(L)$ $\Lambda(L) = \{\varepsilon\}$. On en déduit en particulier que $\varepsilon \in F$, donc ε est un état à la fois initial et final, donc $\varepsilon \in \mathcal{L}(\mathcal{A})$, soit $w \in \mathcal{L}(\mathcal{A})$.
- Sinon, par définitions de $P(L), S(L)$ et $F(L)$:
  - $w_1 \in P(L)$, donc $(\varepsilon, w_1, w_1) \in \delta$.
  - $w_n \in S(L)$ donc $w_n \in F$
  - pour tout $i \in [\![1, n-1]\!], w_i w_{i+1} \in F(L)$, donc $(w_i, w_{i+1}, w_{i+1}) \in \delta$

  Ces trois points montrent que $\varepsilon \xrightarrow{w_1} w_1 \xrightarrow{w_2} w_2 \dots \xrightarrow{w_n} w_n$ est une exécution acceptante de $\mathcal{A}$, et comme elle est d'étiquette w, $w \in \mathcal{L}(\mathcal{A})$.

Dans les deux cas $w \in \mathcal{L}(\mathcal{A})$. D'où $L \subseteq \mathcal{L}(\mathcal{A})$.

$\Leftarrow$ Soit $\mathcal{A} = (\Sigma, Q, \{q_0\}, F, \delta)$ un automate local standard. Notons $\mathcal{L}_\mathcal{A} = \mathcal{L}(\mathcal{A})$ et montrons que $\mathcal{L}_\mathcal{A}$ est local, *i.e.* que $\rho(\mathcal{L}_\mathcal{A}) = \mathcal{L}_\mathcal{A}$.
D'après la propriété 1.7, $\mathcal{L}_\mathcal{A} \subseteq \rho(\mathcal{L}_\mathcal{A})$ car le langage local engendré contient toujours le langage lui-même. Montrons l'inclusion réciproque, *i.e.* $\rho(\mathcal{L}_\mathcal{A}) \subseteq \mathcal{L}_\mathcal{A}$.
Soit $w \in \rho(\mathcal{L}_\mathcal{A}) = \Lambda(\mathcal{L}_\mathcal{A}) \cup \left((P(\mathcal{L}_\mathcal{A}) \cdot \Sigma^\star \cap \Sigma^\star \cdot S \cdot (\mathcal{L}_\mathcal{A})) \setminus (\Sigma^\star \cdot N(\mathcal{L}_\mathcal{A}) \cdot \Sigma^\star\right)$

- Si $w=\varepsilon$, alors $w\in\Lambda(\mathcal{L}_\mathcal{A})$, donc par définition de Λ, $\varepsilon\in\mathcal{L}_\mathcal{A}$, soit $w\in\mathcal{L}_\mathcal{A}$.
- Sinon, on note $w_1w_2\dots w_n$ les lettres de w. Par définition de $\rho(\mathcal{L}_\mathcal{A})$ $w_1\in P(\mathcal{L}_\mathcal{A})$, $w_n\in S(\mathcal{L}_\mathcal{A})$, et $\forall i\in[\![1,n-1]\!]$, $w_iw_{i+1}\in F(\mathcal{L}_\mathcal{A})$. Montrons par récurrence finie sur $p\in[\![1,n]\!]$ qu'il existe une exécution d'étiquette $w_1w_2\dots w_p$ dans $\mathcal{A}$ (H_p)
  - Puisque $w_1\in P(\mathcal{L}_\mathcal{A})$, il existe un mot $u=w_1\overline{u}\in\mathcal{L}_\mathcal{A}$. Il existe donc $q_0\xrightarrow{w_1}q_1\overset{\overline{u}}{\dashrightarrow}\dots$ une exécution acceptante dans $\mathcal{A}$, en particulier il existe une exécution $q_0\xrightarrow{w_1}q_1$ dans $\mathcal{A}$, donc (H_1) est vraie.
  - Soit $p\in[\![1,n-1]\!]$. Supposons (H_p) vraie. Montrons que (H_{p+1}) l'est aussi. Par hypothèse, il existe une exécution $q_0\xrightarrow{w_1}q_1\xrightarrow{w_2}q_2\dots\xrightarrow{w_p}q_p$ dans $\mathcal{A}$. On cherche à la prolonger par une transition étiquetée par w_{p+1}, or on sait que $w_pw_{p+1}\in F(\mathcal{L}_\mathcal{A})$, ce qui implique que $\mathcal{A}$ reconnaît un mot u de la forme $u=\underline{u}w_pw_{p+1}\overline{u}$, et donc qu'il existe une transition étiquetée par w_{p+1} sortant de q_p. En effet, il existe une exécution dans $\mathcal{A}$ de la forme $q_0\overset{\underline{u}}{\dashrightarrow}r\xrightarrow{w_p}s\xrightarrow{w_{p+1}}t\overset{\overline{u}}{\dashrightarrow}\dots$, et comme $\mathcal{A}$ est local, il existe un seul état où peuvent aboutir les transitions étiquetées par w_p : q_p. Donc $s=q_p$, et en notant $q_{p+1}=t$, la transition $(q_p,w_{p+1},q_{p+1})\in\delta$. Ainsi $q_0\xrightarrow{w_1}q_1\xrightarrow{w_2}q_2\dots\xrightarrow{w_p}q_p\xrightarrow{w_{p+1}}q_{p+1}$ est une exécution de $\mathcal{A}$ et (H_{p+1}) est vraie.

  On en déduit que (H_n) est vraie, autrement dit il existe dans $\mathcal{A}$ une exécution $q_0\xrightarrow{w_1}q_1\xrightarrow{w_2}q_2\dots\xrightarrow{w_n}q_n$, reste à montrer qu'elle est acceptante. Comme $q_n\in S(\mathcal{L}_\mathcal{A})$, on sait que $\mathcal{A}$ reconnaît un mot de la forme $\underline{u}w_n$. Il existe donc dans $\mathcal{A}$ une exécution acceptante de la forme $q_0\overset{\underline{u}}{\dashrightarrow}s\xrightarrow{w_n}t$. Pusque cette exécution est acceptante $t\in F$, et puisque $\mathcal{A}$ est local, nécessairement $t=q_n$ (le seul état où $\mathcal{A}$ peut aboutir après la lecture de w_n). Finalement $q_0\xrightarrow{w_1}q_1\xrightarrow{w_2}q_2\dots\xrightarrow{w_n}q_n$ est exécution est acceptante de $\mathcal{A}$, donc son étiquette w est reconnue, *i.e.* $w\in\mathcal{L}_\mathcal{A}$. D'où $\rho(\mathcal{L}_\mathcal{A})\subseteq\mathcal{L}_\mathcal{A}$.

□

1.4 Les langages réguliers sont reconnaissables

1.4.1 Algorithme de Berry-Sethi

Théorème 1.34

Tout langage régulier est reconnaissable.
De plus, étant donné une expression régulière e, l'algorithme de Berry-Sethi permet de construire un automate reconnaissant $\mathcal{L}(e)$ dont le nombre d'états est un plus le nombre de lettres de e comptées avec multiplicité.

Démonstration : Soit e une expression régulière. D'après la propriété 1.26, on peut linéariser e, c'est-à-dire qu'il existe une expression régulière linéaire l sur un alphabet $\overline{\Sigma}$ et une fonction $\varphi:\overline{\Sigma}\to\Sigma$ telles que $e=l_\varphi$. Puisque l est linéaire, $\mathcal{L}(l)$ est un langage local, d'après la propriété 1.33, il existe donc $\mathcal{A}$ un automate local standard reconnaissant $\mathcal{L}(l)$. Finalement $\mathcal{L}(\mathcal{A}_\varphi)=\widetilde{\varphi}(\mathcal{L}(\mathcal{A}))=\widetilde{\varphi}(\mathcal{L}(l))=\mathcal{L}(l_\varphi)=\mathcal{L}(e)$.

La borne sur la taille de l'automate découle de deux constructions :

- celle de l'expression linéarisée fournie dans la preuve de la proposition 1.26, qui prend un nouvel alphabet en bijection avec les occurrences des lettres dans l'expression initiale, donc un alphabet de cardinal le nombre de lettres de l'expression initiale comptées avec multiplicité ;
- celle de l'automate fournie dans la preuve de la proposition 1.33, qui prend pour ensemble d'états l'alphabet et ajoute un état initial ε.

□

Algorithme 1 : Algorithme de Berry-Sethi

Entrée : Une expression régulière e sur un alphabet Σ
Sortie : Un automate sur Σ reconnaissant $\mathcal{L}(e)$

1 On linéarise e comme expliqué dans la propriété 1.26, on obtient ainsi une expression régulière linéaire l sur un alphabet $\overline{\Sigma}$ et $\varphi : \overline{\Sigma} \to \Sigma$ telles que $l_\varphi = e$;
2 On calcule inductivement $\Lambda(l), P(l), S(l), F(l)$ comme expliqué dans la propriété 1.22 ;
3 On construit $\mathcal{A}^l = (\overline{\Sigma}, Q, I, F, \delta)$ où $Q = \overline{\Sigma} \cup \{\varepsilon\}$, $I = \{\varepsilon\}$, $F = S(l) \cup \Lambda(l)$ et $\delta = \left\{(q, a, a) \,\middle|\, a \in \overline{\Sigma}, q \in \overline{\Sigma},\ qa \in F(l)\right\} \sqcup \{(\varepsilon, a, a) \mid a \in P(l)\}$;
4 On calcule $\mathcal{A}^e = (\mathcal{A}^l)_\varphi$ en appliquant φ à toutes les étiquettes des transitions de $\mathcal{A}^l$;
5 **retourner** $(\mathcal{A}^e)$

Exemple 1.35
Reprenons l'exemple de l'expression $e = c^\star((aa)|\varepsilon)(b((a|c|\varepsilon)^\star))^\star baa^\star$, linéarisée en l qu'on peut ré-écrire comme suit♣ :

$$l = \left(c_1^\star \cdot (a_1 a_2 | \varepsilon)\right) \cdot \left(\left(\left((b_1 \cdot (a_3 | (c_2 | \varepsilon)))^\star\right)^\star \cdot b_2\right) \cdot (a_4 \cdot a_5^\star)\right)$$

Afin de calculer plus facilement les ensembles $\Lambda(l), P(l), S(l), F(l)$ définis par induction, on met au jour la structure inductive de l'expression l en dessinant son arbre de syntaxe (*Cf.* figure 1).

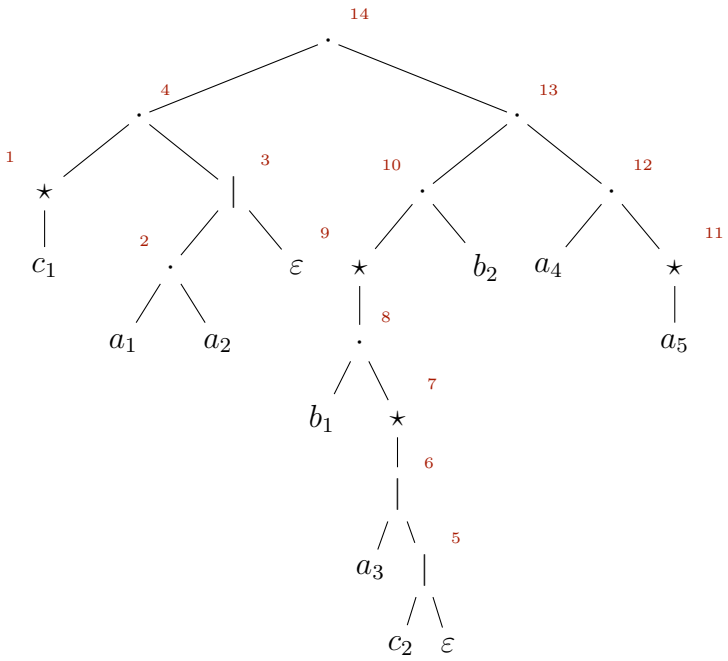


FIGURE 1 – Arbre de syntaxe de l'expression linéaire l

♣. On a ajouté des parenthèses pour lever les ambiguïtés syntaxiques, qui n'impactaient pas la sémantique, mais cassaient l'unicité de l'arbre de syntaxe.

$f \in \mathscr{E}_{reg}(\Sigma)$	$\Lambda(f)$	$P(f)$	$S(f)$	$F(f)$
c_1	$\emptyset$	$\{c_1\}$	$\{c_1\}$	$\emptyset$
$1 = c_1^\star$	$\{\varepsilon\}$	$\{c_1\}$	$\{c_1\}$	$\{c_1c_1\}$
a_1	$\emptyset$	$\{a_1\}$	$\{a_1\}$	$\emptyset$
a_2	$\emptyset$	$\{a_2\}$	$\{a_2\}$	$\emptyset$
$2 = a_1 \cdot a_2$	$\emptyset$	$\{a_1\}$	$\{a_2\}$	$\{a_1a_2\}$
ε	$\{\varepsilon\}$	$\emptyset$	$\emptyset$	$\emptyset$
$3 = 2 \mid \varepsilon$	$\{\varepsilon\}$	$\{a_1\}$	$\{a_2\}$	$\{a_1a_2\}$
$4 = 1 \cdot 3$	$\{\varepsilon\}$	$\{c_1, a_1\}$	$\{a_2, c_1\}$	$\{a_1a_2, c_1c_1, c_1a_1\}$
b_1	$\emptyset$	$\{b_1\}$	$\{b_1\}$	$\emptyset$
a_3	$\emptyset$	$\{a_3\}$	$\{a_3\}$	$\emptyset$
c_2	$\emptyset$	$\{c_2\}$	$\{c_2\}$	$\emptyset$
$5 = c_2 \mid \varepsilon$	$\{\varepsilon\}$	$\{c_2\}$	$\{c_2\}$	$\emptyset$
$6 = a_3 \mid 5$	$\{\varepsilon\}$	$\{a_3, c_2\}$	$\{a_3, c_2\}$	$\emptyset$
$7 = 6^\star$	$\{\varepsilon\}$	$\{a_3, c_2\}$	$\{a_3, c_2\}$	$\{c_2c_2, c_2a_3, a_3c_2, a_3a_3\}$
$8 = b_1 \cdot 7$	$\emptyset$	$\{b_1\}$	$\{c_2, a_3, b_1\}$	$F(7) \cup \{b_1a_3, b_1c_2\} = \{c_2c_2,\ c_2a_3,\ a_3c_2,\ a_3a_3,\ b_1a_3,\ b_1c_2\}$
$9 = 8^\star$	$\{\varepsilon\}$	$\{b_1\}$	$\{c_2, a_3, b_1\}$	$F(8) \cup \{c_2b_1, a_3b_1, b_1b_1\}$ $= \{c_2c_2,\ c_2a_3,\ a_3c_2,\ a_3a_3,\ b_1a_3,\ b_1c_2,\ c_2b_1,\ a_3b_1,\ b_1b_1\}$
b_2	$\emptyset$	$\{b_2\}$	$\{b_2\}$	$\emptyset$
$10 = 9 \cdot b_2$	$\emptyset$	$\{b_1, b_2\}$	$\{b_2\}$	$F(9) \cup \{c_2b_2, a_3b_2, b_1b_2\}$ $= \{c_2c_2, c_2a_3, a_3c_2, a_3a_3, b_1a_3, b_1c_2, c_2b_1, a_3b_1, b_1b_1, c_2b_2,$ $a_3b_2,\ b_1b_2\}$
a_4	$\emptyset$	$\{a_4\}$	$\{a_4\}$	$\emptyset$
a_5	$\emptyset$	$\{a_5\}$	$\{a_5\}$	$\emptyset$
$11 = a_5^\star$	$\{\varepsilon\}$	$\{a_5\}$	$\{a_5\}$	$\{a_5a_5\}$
$12 = a_4 \cdot 11$	$\emptyset$	$\{a_4\}$	$\{a_5, a_4\}$	$\{a_5a_5\} \cup \{a_4a_5\}$
$13 = 10 \cdot 12$	$\emptyset$	$\{b_1, b_2\}$	$\{a_5, a_4\}$	$F(10) \cup F(12) \cup \{b_2a_4\}$ $= \{c_2c_2, c_2a_3, a_3c_2, a_3a_3, b_1a_3, b_1c_2, c_2b_1, a_3b_1, b_1b_1, c_2b_2,$ $a_3b_2,\ b_1b_2,\ a_5a_5,\ a_4a_5,\ b_2a_4\}$
$14 = 4 \cdot 13$	$\emptyset$	$\{c_1, a_1, b_1, b_2\}$	$\{a_5, a_4\}$	$F(4) \cup F(13) \cup \{a_2b_1, a_2b_2, c_1b_1, c_1b_2\}$ $= \{a_1a_2,\ c_1c_1,\ c_1a_1,\ c_2c_2,\ c_2a_3,\ a_3c_2,\ a_3a_3,\ b_1a_3,$ $b_1c_2,\ c_2b_1,\ a_3b_1,\ b_1b_1,\ c_2b_2,\ a_3b_2,\ b_1b_2,\ a_5a_5,\ a_4a_5,$ $b_2a_4, a_2b_1, a_2b_2, c_1b_1, c_1b_2\}$

D'après la dernière ligne du tableau $\Lambda(l) = \emptyset$, $P(l) = \{c_1, a_1, b_1, b_2\}$, $S(l) = \{a_4, a_5\}$ et

$$F(l) = \{\ a_1a_2,\ a_2b_1,\ a_2b_2,\ a_3a_3,\ a_3b_1,\ a_3b_2,\ a_3c_2,\ a_4a_5,\ a_5a_5,\ b_1a_3,\ b_1b_1,\ b_1b_2,\ b_1c_2,\ b_2a_4,\ c_1a_1,\ c_1b_1,\ c_1c_1,\ c_1b_2,\ c_2a_3,\ c_2b_1,\ c_2b_2,\ c_2c_2\ \}$$

Ces ensembles permettent de construire l'automate $\mathscr{A}^l$ représenté sur la figure 2 qui reconnaît $\mathscr{L}(l)$, et en appliquant φ à toutes les étiquettes on obtient finalement l'automate $\mathscr{A}^e = (\mathscr{A}^l)_\varphi$ représenté sur la figure 3 qui reconnaît $\mathscr{L}(e)$.

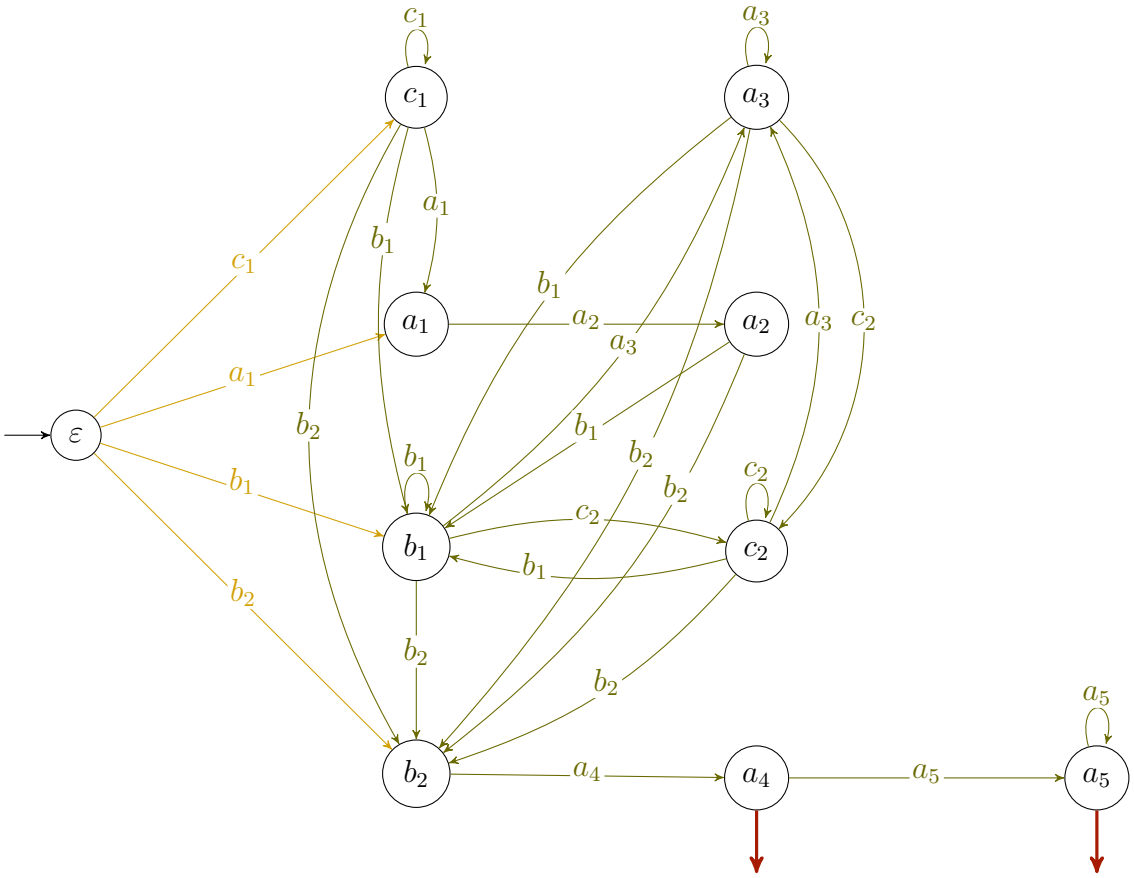


FIGURE 2 – Automate $\mathcal{A}^l$ qui reconnaît $\mathcal{L}(l)$

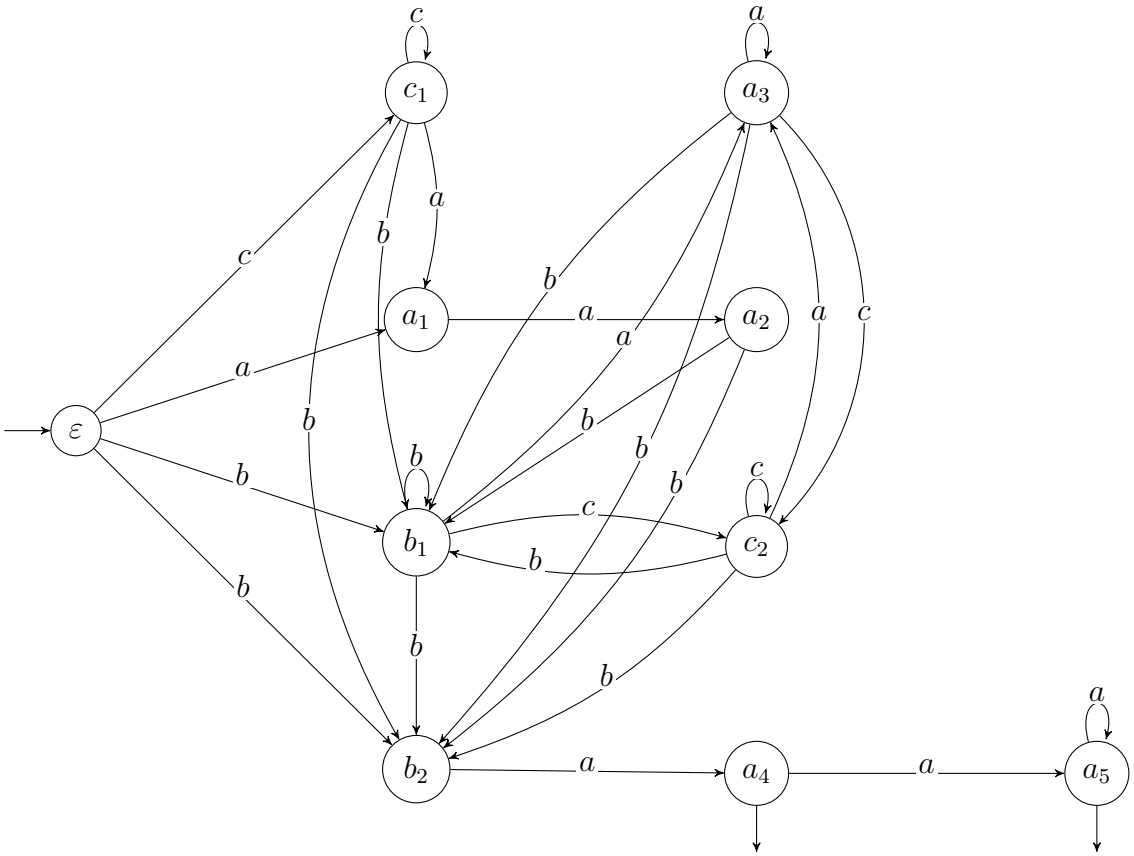


FIGURE 3 – Automate $\mathcal{A}^e$ qui reconnaît $\mathcal{L}(e)$

Exercice de cours 1.36

Donner une expression régulière pour laquelle l'algorithme de Berry-Sethi produit un automate non déterministe.
Donner une expression régulière pour laquelle l'algorithme de Berry-Sethi produit un automate déterministe.

Exercice de cours 1.37

En utilisant l'algorithme du cours, donner un automate dont le langage est $(ab|a)^\star(\varepsilon|caaa)$.

2 La classe des langages reconnaissables

Dans la section précédente, nous avons établi que la classe des langages reconnaissables est exactement celle des langages réguliers, ce qui implique qu'elle est close par union, concaténation et étoile. Cependant nous n'avons pas répondu à toutes les questions : est-elle close par passage au complémentaire ? Par intersection ? Et au delà, on peut aussi se demander si elle contient tous les langages. Ces questions sont l'objet de cette dernière section.

2.1 Stabilités de la classe des langages reconnaissables

Proposition 2.1

Soit $\mathcal{A} = (\Sigma, Q, I, F, \delta)$ un automate.
Il existe un automate qui reconnaît le langage complémentaire, i.e. $\Sigma^\star \setminus \mathcal{L}(\mathcal{A})$.

Démonstration : Si $\mathcal{A}$ est **complet et déterministe**, l'automate $\overline{\mathcal{A}} = (\Sigma, Q, I, Q \setminus F, \delta)$ reconnaît $\Sigma^\star \setminus \mathcal{L}(\mathcal{A})$. Si $\mathcal{A}$ n'est pas complet on a vu comment le compléter, et s'il n'est pas déterministe, on a vu comment le déterminiser. □

Exercice de cours 2.2

Dans la preuve précédente, démontrer que $\overline{\mathcal{A}} = (\Sigma, Q, I, Q \setminus F, \delta)$ reconnaît $\Sigma^\star \setminus \mathcal{L}(\mathcal{A})$.

Exercice de cours 2.3

Donner un automate sur l'alphabet $\Sigma = \{a, b\}$, ayant deux états et trois transitions et reconnaissant l'ensemble des mots terminant par un b. **En utilisant la proposition précédente**, donner un automate reconnaissant le langage des mots sur $\Sigma = \{a, b\}$ ne terminant pas par un b.

Corollaire 2.4

Soient $\mathcal{A}_1$ et $\mathcal{A}_2$ deux automates sur un alphabet Σ.
- *Il existe un automate qui reconnaît $\mathcal{L}(\mathcal{A}_1) \cap \mathcal{L}(\mathcal{A}_2)$.*
- *Il existe un automate qui reconnaît $\mathcal{L}(\mathcal{A}_1) \setminus \mathcal{L}(\mathcal{A}_2)$.*
- *Il existe un automate qui reconnaît $\mathcal{L}(\mathcal{A}_1) \Delta \mathcal{L}(\mathcal{A}_2)$.*

Autrement dit la classe des langages reconnaissables est stable par intersection, différence et différence symétrique.

Démonstration : Il suffit de rappeler quelques égalités générales sur les ensembles pour montrer que ces trois opérations s'obtiennent successivement à partir de l'union et du passage au complémentaire, opérations pour lesquelles on sait déjà construire un automate reconnaissant le résultat à partir des automates reconnaissant les opérandes. □

Exercice de cours 2.5

Donner les égalités ensemblistes justifiant les stabilités ci-dessus.

Remarque 2.6

La classe des langages reconnaissables est stable par de nombreuses opérations : miroir, racine carré, interclassement... Se reporter au TD.

Proposition 2.7

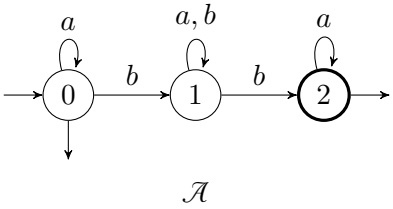
Il existe un algorithme permettant de tester, étant donnés deux automates $\mathcal{A}$ et $\mathcal{B}$:
- *si le langage de $\mathcal{A}$ est inclus dans le langage de $\mathcal{B}$;*
- *si le langage de $\mathcal{A}$ est égal au langage de $\mathcal{B}$.*

Démonstration : Remarquons que pour tous langages L et L', $L \subseteq L' \Leftrightarrow L \setminus L' = \emptyset$. Ainsi pour tester si $\mathcal{L}(\mathcal{A}) \subseteq \mathcal{L}(\mathcal{B})$ il suffit de construire un automate $\mathcal{C}$ dont le langage est $\mathcal{L}(\mathcal{A}) \setminus \mathcal{L}(\mathcal{B})$ (en utilisant le Corollaire 2.4) puis de tester la vacuité du langage de l'automate $\mathcal{C}$.

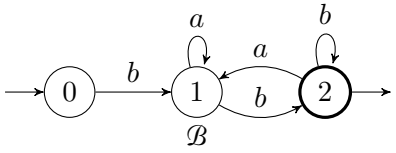
Pour tester si les langages de deux automates sont égaux il suffit de tester les deux inclusions. □

Exercice de cours 2.8

Considérons les deux automates ci-dessous. Donner les langages de ces deux automates. En appliquant l'algorithme de la Propriété 2.7, justifier que $\mathcal{L}(\mathcal{B}) \subseteq \mathcal{L}(\mathcal{A})$



$\mathcal{A}$



$\mathcal{B}$

2.2 Lemme de l'étoile

Au vu de ces nombreuses stabilités, on peut se demander si la classe des langages reconnaissables n'est pas en fait l'ensemble de tous les langages.

Remarque 2.9

On peut répondre très rapidement par la négative à cette question pour des raisons de cardinalité. Sur un alphabet $\Sigma = \{0, 1\}$ l'ensemble $\Sigma^\star$ des mots sur Σ est en bijection avec $\mathbb{N}$ (puisqu'on peut encoder chaque entier par son écriture en binaire de taille minimale). L'ensemble des langages, qui est l'ensemble des parties de $\Sigma^\star$, est alors en bijection avec $\mathcal{P}(\mathbb{N})$, qui lui-même est en bijection avec $\mathbb{R}$ et donc non dénombrable (on utilise le fait qu'il n'existe pas de bijection entre $\mathbb{N}$ et $\mathbb{R}$). D'un autre coté l'ensemble des langages reconnaissables s'injecte dans l'ensemble des automates (en associant un automate le reconnaissant à chaque langage). Or, pour tout $n \in \mathbb{N}$, il y a un nombre fini d'automates à n états sur l'alphabet $\Sigma = \{0, 1\}$ et donc, en tant qu'union dénombrable d'ensembles finis, l'ensemble de tous les automates est dénombrable. Aussi l'ensemble des langages reconnaissables s'injecte dans un ensemble dénombrable, il ne peut donc pas être égal à l'ensemble de tous les langages qui n'est pas dénombrable.

Un langage reconnaissable est un langage reconnu par un automate, cet automate a une "mémoire" finie : il ne dispose que d'un nombre fini d'états. Aussi, il est nécessaire que la lecture de nombreux mots conduise au même état, et de là le comportement de l'automate est le même peu importe quel mot l'a mené dans cet état, autrement dit l'automate n'a pas la mémoire de ce qu'il a lu depuis le début. C'est l'idée du lemme de l'étoile énoncé ci-dessous.

Théorème 2.10 (*Lemme de l'étoile*)

Soit L le langage reconnu par un automate sur Σ à n états. Pour tout $u \in L$ tel que $|u| \geqslant n$, il existe des mots x, y, z sur Σ tels que :

- $u = xyz$,
- $|xy| \leqslant n$,
- $y \neq \varepsilon$,
- $\mathcal{L}(x \cdot y^\star \cdot z) \subseteq L$.

Démonstration : Comme suggéré en introduction, ce résultat découle de la finitude de l'automate. Soit L le langage reconnu par un automate $\mathcal{A} = (\Sigma, Q, I, F, \delta)$ à n états (*i.e.* $|Q| = n$). Soit $u \in L$ tel que $|u| \geqslant n$. Notons alors le découpage de u en lettres : $u = u_1 u_2 \dots u_m$ avec $m \geqslant n$. Puisque $u \in \mathcal{L}(\mathcal{A})$, il existe $(q_i)_{i \in [\![0,n]\!]} \in Q^{n+1}$ telle que $q_0 \xrightarrow{u_1} q_1 \xrightarrow{u_2} q_2 \dots \xrightarrow{u_m} q_m$ est une exécution acceptante dans $\mathcal{A}$. En particulier la lecture des n premières lettres de u conduit à une exécution de la forme $q_0 \xrightarrow{u_1} q_1 \xrightarrow{u_2} q_2 \dots \xrightarrow{u_n} q_n$. Étant donné que l'automate $\mathcal{A}$ a n états, par le principe des tiroirs, il existe nécessairement $(i, j) \in [\![0, n]\!]^2$ avec $i < j$ tels que $q_i = q_j \overset{\text{déf}}{=} q$. On fixe alors

- $x = u_1 u_2 \dots u_i$ (avec potentiellement $i = 0$ et donc $x = \varepsilon$) ;
- $y = u_{i+1} u_{i+2} \dots u_j$ (avec $i + 1 \leqslant j$ et donc $y \neq \varepsilon$) ;
- $z = u_{j+1} u_{j+2} \dots u_m$ (avec potentiellement $j = n = m$ et donc $z = \varepsilon$).

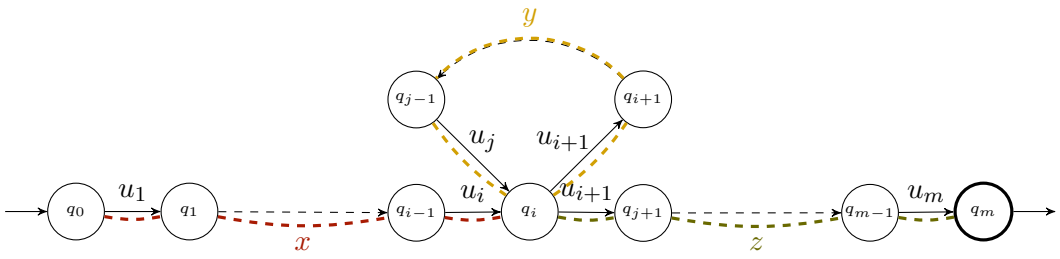
Par construction :

- $u = x \cdot y \cdot z$;
- $|x \cdot y| = j \leqslant n$;
- $y \neq \varepsilon$ car $y = j - i > 0$.

Il reste à montrer que $x \cdot y^\star \cdot z \subseteq L$, autrement dit que $\forall p \in \mathbb{N}, x \cdot y^p \cdot z \in L$. Montrons alors par induction sur $p \in \mathbb{N}$ qu'il existe dans l'automate $\mathcal{A}$ une exécution d'étiquette $x \cdot y^p$ conduisant à l'état q.

- Base : cas $p = 0$. Par construction de x est l'étiquette de l'exécution $q_0 \xrightarrow{u_1} q_1 \xrightarrow{u_2} q_2 \dots \xrightarrow{u_i} q_i = q_0$.
- Supposons le résultat vrai pour un certain $p \in \mathbb{N}$ et montrons le pour $p+1$. Il existe alors une exécution d'étiquette $x \cdot y^p$ menant à q, ce qu'on note $q_0 \overset{xy^p}{\dashrightarrow} q$. Or $q = q_i \xrightarrow{u_{i+1}} q_{i+1} \xrightarrow{u_{i+2}} q_{i+2} \dots \xrightarrow{u_j} q_j = q$ est une suite de transitions dans $\mathcal{A}$. Ainsi $q_0 \overset{xy^p}{\dashrightarrow} q \xrightarrow{u_{i+1}} q_{i+1} \xrightarrow{u_{i+2}} q_{i+2} \dots \xrightarrow{u_j} q_j = q$ est une exécution dans $\mathcal{A}$, étiquetée par $(x \cdot y^p) \cdot y = x \cdot y^{p+1}$ et menant à q, donc la propriété est vérifiée au rang $p+1$.

Par principe d'induction on en déduit que pour tout $p \in \mathbb{N}$, il existe une suite d'exécution d'étiquette xy^p menant à l'état q. En remarquant que $q = q_j \xrightarrow{u_{j+1}} q_{j+1} \xrightarrow{u_{j+2}} q_{j+2} \dots \xrightarrow{u_m} q_m$ est une suite de transitions de l'automate d'étiquette z conduisant à q_m qui est un état final (car $u \in \mathcal{L}(\mathcal{A})$, on en déduit que pour tout $p \in \mathbb{N}$, il existe une exécution acceptante d'étiquette $x \cdot y^p \cdot z$ dans $\mathcal{A}$. On a finalement démontré les 4 points annoncés pour les mots x, y, z que l'on a choisis. Le schéma ci-dessous illustre les différents éléments de la preuve.



□

Remarque 2.11

Ce lemme de l'étoile impose une condition sur les langages réguliers/reconnaissables infinis : des facteurs du début des mots de ce langage peuvent être supprimés ou au contraire répétés sans sortir du langage.

Exercice de cours 2.12

Démontrer le lemme de l'étoile suivant. Soit L le langage reconnu par un automate à n états sur Σ. Pour tout $u \in L$ tel que $|u| \geqslant n$, il existe des mots x, y, z sur Σ tels que :
- $u = xyz$,
- $|yz| \leqslant n$,
- $y \neq \varepsilon$,
- $\mathcal{L}(x \cdot y^\star \cdot z) \subseteq L$.

2.3 Prouver que des langages sont non reconnaissables/non réguliers

Dans cette section nous présentons comment utiliser le lemme de l'étoile pour prouver la non reconnaissabilité d'un langage, en l'occurrence le langage $L = \{a^n b^n \mid n \in \mathbb{N}\}$. Intuitivement ce langage ne peut être reconnaissable pour la raison suivante : quand on commence à lire des a, il faut garder en mémoire un entier indiquant combien de a ont été lus, sans quoi on ne pourrait pas reconnaître qu'il y a exactement autant de b. La mémoire finie d'un automate ne permet pas une telle mémorisation. En effet une information de type entier peut prendre une infinité de valeurs, et il faudrait un état pour indiquer chacune de ces valeurs possibles.
L'intérêt de cette sous-partie n'est pas tant le lemme suivant que la technique de preuve employée dans sa démonstration.

Lemme 2.13

Le langage $L = \{a^n b^n \mid n \in \mathbb{N}\}$ n'est pas reconnaissable.

Schéma de preuve

→ On suppose par l'absurde que le langage est reconnaissable, par un automate ayant $n \in \mathbb{N}$ états.
→ Le langage L étant infini, il contient nécessairement des mots de longueur strictement supérieure à n, candidats à l'application du lemme de l'étoile. Cependant, pour pouvoir utiliser le résultat il faut choisir un mot u dont le début (*i.e.* les n premières lettres) est bien connu, et bien choisi.
→ D'après le lemme de l'étoile, il existe un facteur de ce début qu'on peut supprimer ou répéter tout en restant dans L.
→ Connaissant la forme qu'a nécessairement ce facteur, on exhibe un mot où il est supprimé ou répété qui n'appartient pas à L, et de ce fait une absurdité.
→ On en déduit que le L n'est reconnu par aucun automate, donc pas reconnaissable.

Attention : On peut choisir le mot u, mais pas son facteur y.

Démonstration : Supposons que L est reconnaissable par un automate fini à N états. Le langage L ne permet pas de répéter des a sans ajouter autant de b, aussi nous appliquons le lemme de l'étoile au mot $u = a^N b^N$, dont le début, les N premières lettres, est constitué exclusivement de a. D'après le lemme de l'étoile il existe une factorisation de u en xyz telle que :

- $u = xyz$
- $|xy| \leqslant N$
- $y \neq \varepsilon$,
- $\mathcal{L}(xy^\star z) \subseteq L$

Puisque $|xy| \leqslant N$, x et y sont uniquement composés de a, il existe donc $(p, q) \in \mathbb{N}^2$ tels que $x = a^p$ et $y = a^q$ avec $p + q = |xy| \leqslant N$, et donc $z = a^{N-p-q} b^N$. De plus comme $y \neq \varepsilon$, $q = |y| > 0$. Enfin le dernier point assure que les mots suivants de $\mathcal{L}(xy^\star z)$ sont dans L :

- $xy^0 z = a^{N-q} b^N \in L \rightarrow$ Absurde car $N - q \neq N$ puisque $q \neq 0$
- $xy^2 z = a^{N+q} b^N \in L \rightarrow$ Absurde car $N + q \neq N$ puisque $q \neq 0$
- $xy^3 z = a^{N+2q} b^N \in L \rightarrow$ Absurde car $N + 2q \neq N$ puisque $q \neq 0$
- ...

Un seul de ces mots suffisait à établir une contradiction. Le langage L n'est donc pas reconnaissable. □

Exercice de cours 2.14

Démontrer que le langage $\{a^n b^m c^p \mid n+m=p\}$ n'est pas régulier.

Proposition 2.15

Il existe des langages non reconnaissables.

Démonstration : C'est un corollaire du lemme précédent. □

A Prouver la correction d'un automate

Dans cette annexe on propose une technique de preuve de correction d'un automate. Étant donné un automate $\mathcal{A} = (\Sigma, Q, I, F, \delta)$, il s'agit de démontrer que le langage reconnu $\mathcal{L}(\mathcal{A})$ est bien le langage attendu qu'on note L. Pour cela on peut suivre la méthodologie suivante.

→ pour chaque état $q \in Q$, on énonce $C_q(w)$ une condition nécessaire et suffisante sur $w \in \Sigma^\star$ pour qu'il existe une exécution d'étiquette w menant à l'état q dans $\mathcal{A}$ (ce que l'on note $I \overset{w}{\dashrightarrow} q$)

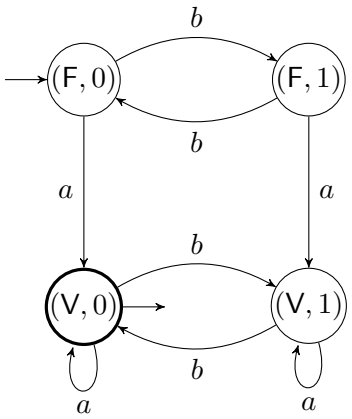
→ on démontre par récurrence sur la taille de w que $\forall q \in Q, I \overset{w}{\dashrightarrow} q \Leftrightarrow C_q(w)$

→ on en déduit que qu'un mot $w \in \Sigma^\star$ est accepté par l'automate ssi $\exists q_f \in F,\ C_{q_f}(w)$.

→ il suffit donc de démontrer $\forall w \in \Sigma^\star,\ w \in L \Leftrightarrow \bigvee_{q_f \in F} C_{q_f}(w)$

Exemple A.1

Pour $\Sigma = \{a, b\}$, on construit l'automate $\mathcal{A}$ représenté ci-dessous ♣ pour reconnaître les mots contenant au moins un a et un nombre pair de b, *i.e.* la langage $L = \{w \in \Sigma^\star \mid |w|_a \geqslant 1 \text{ et } |w|_b \equiv 0[2]\}$.



Montrons, grâce à la méthode proposée ci-avant, que $\mathcal{A}$ est correct, c'est-à-dire que $\mathcal{L}(\mathcal{A}) = L$.

Démonstration : Pour tout $(v, r) \in \mathbb{B} \times \{0, 1\}$, nous énonçons la condition suivante sur $w \in \Sigma^\star$:

$$C_{(v,r)}(w) : (|w|_a \geqslant 1 \Leftrightarrow v) \text{ et } (|w|_b \equiv r[2])$$

Montrons alors par induction sur $w \in \Sigma^\star$ la propriété $\mathcal{P}_w : \forall q \in Q, I \overset{w}{\dashrightarrow} q \Leftrightarrow C_{(v,r)}(w)$.

• Montrons $\mathcal{P}_w$ pour $w = \varepsilon$. Soit $q = (v, r) \in \mathbb{B} \times \{0, 1\} = Q$. $I \overset{\varepsilon}{\dashrightarrow} q$ ssi $q \in I$,*i.e.* ssi $q = (\mathsf{F}, 0)$. Par ailleurs $C_q(\varepsilon) \Leftrightarrow C_{(v,r)}(\varepsilon) \Leftrightarrow (0 \equiv r[2])$ et $(0 \geqslant 1 \Leftrightarrow v) \Leftrightarrow r = 0$ et $v = \mathsf{V} \Leftrightarrow q = (\mathsf{F}, 0)$ Donc $C_q(\varepsilon) \Leftrightarrow I \overset{w}{\dashrightarrow} q$, d'où $\mathcal{P}_\varepsilon$.

• Soit $w = u \cdot z$ avec $z \in \Sigma^\star$ tel que $\mathcal{P}_u$ et $a \in \Sigma$. Soit $q = (v, r) \in \mathbb{B} \times \{0, 1\} = Q$.

♣. la formalisation de ce petit automate comme quintuplet $\mathcal{A} = (\Sigma, Q, I, F, \delta)$ est laissée en exercice, mais est utilisée dans la preuve qui suit.

$\Rightarrow$ Supposons que $I \xrightarrow{w} q$, alors il existe $q' = (v', r') \in Q$ tel que $I \xrightarrow{u} q \xrightarrow{a} q'$. D'après $\mathscr{P}_u$, $C_{(v',r')}(u)$. Montrons par disjonction de cas que $C_{(v,r)}(w)$.

- Si $z = b$, $|w|_a = |u|_a + |z|_a = |u|_a$ ($\spadesuit_a$) et $|w|_b = |u|_b + |z|_b = |u|_b + 1$ ($\spadesuit_b$). On distingue cependant deux sous-cas.
  - Si $q = (x, 0)$, nécessairement $q' = (x, 1)$. *(En effet, dans $\mathscr{A}$, les seules transitions étiquetées par b aboutissant sur un état de la forme $(x, 0)$ proviennent de l'état $(x, 1)$.)* Donc d'après $C_{q'}(u) = C_{(x,1)}(u)$ $|u|_a \geqslant 1 \Leftrightarrow x$ et $|u|_b \equiv 1[2]$, d'après ($\spadesuit_a$) on en déduit que $|w|_a \geqslant 1 \Leftrightarrow x$ et $|w|_b \equiv 0[2]$, soit $C_{(x,0)}(w)$ ou encore $C_q(w)$.
  - Si $q = (x, 1)$ et $z = b$, nécessairement $q' = (x, 0)$, et de même on montre qu'alors $C_{(x,0)}(w)$ soit $C_q(w)$.
- Si $z = a$, $|w|_a = |u|_a + |z|_a = |u|_a + 1$, donc $|w|_a \geqslant 1$ soit $|w|_a \geqslant 1 \Leftrightarrow \mathsf{V}$ ($\clubsuit_a$) et $|w|_b = |u|_b + |z|_b = |u|_b$ ($\clubsuit_b$). De plus les transitions étiquetées par a aboutissant toutes dans un état de la forme $(\mathsf{V}, \bullet)$ on a à traiter seulement les cas $q = (\mathsf{V}, 0)$ et $q = (\mathsf{V}, 1)$.
  - Si $q = (\mathsf{V}, 0)$, nécessairement $q' = (\mathsf{V}, 0)$ ou $q' = (\mathsf{F}, 0)$. Donc d'après $C_{q'}(u)$, en particulier $|u|_b \equiv 0[2]$, donc d'après ($\clubsuit_b$), $|w|_b \equiv 0[2]$. Avec ($\clubsuit_a$) on en déduit $C_{(\mathsf{V},0)}(w)$, soit $C_q(w)$.
  - Si $q' = (\mathsf{V}, 1)$, nécessairement $q' = (\mathsf{V}, 1)$ ou $q' = (\mathsf{F}, 1)$. Donc d'après $C_{q'}(u)$ en particulier $|u|_b \equiv 1[2]$, donc d'après ($\clubsuit_b$), $|w|_b \equiv 1[2]$. Avec ($\clubsuit_a$) on en déduit $C_{(\mathsf{V},1)}(w)$, soit $C_q(w)$.

Dans tous les cas $C_q(w)$.

$\Leftarrow$ Réciproquement supposons que $C_{(v,r)}(w)$ et montrons que $I \xrightarrow{w} q$. En particulier $|w|_b \equiv r[2]$.
On pose $v' = \mathsf{V}$ si $|u|_a \geqslant 1$ et $v' = \mathsf{F}$ sinon, ainsi par construction $|u|_a \geqslant 1 \Leftrightarrow v'$ ($\diamondsuit_a$)

- Si $z = a$, alors $|w|_a \geqslant 1$, et comme $|w|_a \geqslant\!\!\!\Leftrightarrow v$ par $C_{(v,r)}(w)$, nécessairement $v = \mathsf{V}$. De plus $|w|_b = |u|_b$, donc $|u|_b \equiv r[2]$, et avec ($\diamondsuit_a$) on en déduit $C_{(v',r)}(u)$. D'après $\mathscr{P}_u$: $I \xrightarrow{u} (v', r)$, or par construction de $\mathscr{A}$ il existe une transition de (v, r) vers (V, r) d'étiquette a donc $I \xrightarrow{ua} (\mathsf{V}, r)$ soit $I \xrightarrow{w} (\mathsf{V}, r) = (v, r)$.

- Si $z = b$, $|w|_a = |u|_a$ donc par définition de v' et puisque $|w|_a \geqslant 1 \Leftrightarrow v$ par $C_{(v,r)}(w)$, $v' = v$. De plus $|w|_b = |u|_b + 1$, donc en posant $r' = 1 - r$, $|u|_b \equiv r'[2]$, et finalement $C_{(v',r')}(u)$. D'après $\mathscr{P}_u$: $I \xrightarrow{u} (v', r')$, or par construction de $\mathscr{A}$ il existe une transition de (v', r') vers $(v', 1 - r')$ d'étiquette b, donc $I \xrightarrow{ua} (v', 1 - r')$ soit $I \xrightarrow{w} (v', 1 - r') = (v, r)$.

Dans tous les cas $I \xrightarrow{w} (v, r) = q$. On en déduit que $\forall q = (v, r) \in Q, I \xrightarrow{w} (v, r) = q \Leftrightarrow C_{(v,r)}(w)$, autrement dit $\mathscr{P}(w)$ est vraie.

• Par induction on en déduit que $\forall w \in \Sigma^\star, \forall q \in Q, C_q(w) \Leftrightarrow I \xrightarrow{w} q$. Pour tout $w \in \Sigma^\star$:

$w \in \mathcal{L}(\mathscr{A})$	ssi	$\exists q_f \in F,\ I \xrightarrow{w} q_f$	*par définition du langage accepté*
	ssi	$I \xrightarrow{w} (\mathsf{V}, 0)$	*car $F = \{(V, 0)\}$*
	ssi	$C_{(\mathsf{V},0)}(w)$	*d'après l'équivalence qu'on vient de démonter*
	ssi	$(\lvert w\rvert_a \geqslant 1 \Leftrightarrow \mathsf{V})$ et $(\lvert w\rvert_b \equiv 0[2])$	*par définition de la condition C*
	ssi	$\lvert w\rvert_a \geqslant 1$ et $(\lvert w\rvert_b \equiv 0[2])$	
	ssi	$w \in L$	*d'après la définition du langage L*

D'où $\mathcal{L}(\mathscr{A}) = L$. □