Chapitre 3

Corrigé des exercices

• Mots et alphabets

Exercice 1

T	O	B	E	O
R	N	O	T	T
O	B	E	A	R
E	G	U	L	A
R	R	E	G	X

Exercice 2

a) Par hypothèse il existe des mots x et y tels que $w = ux = vy$. D'après le lemme de Levi il existe un mot t tel que $u = vt$, $y = tx$ ou $v = ut$, $x = ty$. Dans le premier cas v est préfixe de u ; dans le second cas u est préfixe de v.

b) Raisonnons par récurrence sur $|u|$.

– Si $u = \varepsilon$ le résultat est évident.

– Si $|u| \geqslant 1$, supposons le résultat acquis pour tout mot de longueur inférieure, et appliquons le lemme de Levi : il existe un mot t tel que $u = at$ et $u = tb$. On a donc $at = tb$ et $|t| < |u|$ donc par hypothèse de récurrence $a = b$ et $t \in \{a\}^*$. Mais alors $u = at \in \{a\}^*$, ce qui prouve le résultat souhaité.

c) Posons $t = u^p = v^q$. On a $t^2 = u^{2p} = uu^pu^{p-1} = uv^qv^{q-1}$ donc uv est préfixe de t^2.
De même, $t^2 = v^{2q} = vv^qv^{q-1} = vu^pv^{q-1}$ donc vu est préfixe de t^2. Or uv et vu ont même longueur donc $uv = vu$.
D'après le deuxième théorème issu du lemme de Levi il existe un mot w et deux entiers m et n tels que $u = w^m$ et $v = w^n$.

Exercice 3

a) La relation est *réflexive* : si on pose $x = u$ et $y = \varepsilon$ on a $u = xy = yx$ donc $u\mathscr{R}u$.
La relation est *symétrique* pour des raisons évidentes.
La relation est *transitive* : supposons $u\mathscr{R}v$ et $v\mathscr{R}w$. Il existe donc $x, y, z, t \in \Sigma^*$ tels que $u = xy$, $v = yx$, $v = zt$, $w = tz$.
On a $yx = zt$ donc d'après le lemme de Levi il existe un mot r tel que $y = zr$, $t = rx$ ou alors $z = yr$, $x = rt$. Dans le premier cas on a $u = (xz)r$ et $w = r(xz)$; dans le second cas on a $u = r(ty)$ et $w = (ty)r$. Dans les deux cas on a bien $u\mathscr{R}w$.

b) Si on a $u\mathscr{R}v$ il suffit de poser $w = x$ pour avoir $uw = xyx = wv$.
Réciproquement, s'il existe w tel que $uw = wv$ d'après le premier théorème issu du lemme de Levi il existe deux mots x et y et un entier k tel que $u = xy$, $v = yx$ (et $w = (xy)^kx$) donc on a bien $u\mathscr{R}v$.

c) Si $u\mathscr{R}v$ il existe x et y tel que $u = xy$ et $v = yx$. Alors $u^n = (xy)(xy)^{n-1}$ et $v^n = (yx)^{n-1}(yx) = (xy)^{n-1}(xy)$ donc $u^n\mathscr{R}v^n$.
Réciproquement, on raisonne par récurrence sur $n \in \mathbb{N}^*$.

– Si $n = 1$ le résultat est évident.

– Si $n > 1$, supposons le résultat acquis jusqu'au rang $n-1$, et considérons x et y tels que $u^n = xy$ et $v^n = yx$. Observons déjà que $|x| + |y| = n|u| = n|v|$ donc $|u| = |v|$.
Puisque $n > 1$ l'un des deux mots x ou y est de longueur supérieure à u ; supposons par exemple que ce soit x.

x est préfixe de u^n donc il existe $k \geqslant 1$ et u' préfixe de u tel que $x = u^k u'$. Puisque $|v| = |u|$ il existe v' suffixe de v tel que $x = v'v^k$.
On a $u^k u' = v'v^k$ donc $|v'| = |u'| \leqslant |u|$; v' est donc préfixe de u et par voie de conséquence $v' = u'$. Ainsi, $u^k u' = u'v^k$ ce qui, d'après la question précédente, prouve que $u^k \mathscr{R} v^k$. Par hypothèse de récurrence on en déduit que $u \mathscr{R} v$.

Exercice 4

a) Soit $n \geqslant 3$. Il est facile d'établir par récurrence que si n est pair, f_4 est suffixe de f_n et que si n est impair, f_3 est suffixe de f_n. Or $f_3 = ba$ et $f_4 = bab$ donc le résultat est vrai pour tout $n \geqslant 3$.

b) Montrons par récurrence sur $n \geqslant 3$ que g_n est un palindrome.

- Si $n = 3$, $g_3 = \varepsilon$ est un palindrome.
- Si $n = 4$, $g_4 = b$ est un palindrome.
- Si $n = 5$, $g_5 = bab$ est un palindrome.
- Si $n \geqslant 6$, supposons le résultat acquis jusqu'au rang $n-1$ et distinguons deux cas selon la parité de n.
  Si n est pair, $f_n = f_{n-1}f_{n-2} = f_{n-2}f_{n-3}f_{n-2} = g_{n-2}abg_{n-3}bag_{n-2}ab$ donc $g_n = g_{n-2}abg_{n-3}bag_{n-2}$. Par hypothèse de récurrence g_{n-2} et g_{n-3} sont des palindromes, donc il en est de même de g_n.
  Si n est impair on obtient $g_n = g_{n-2}bag_{n-3}abg_{n-2}$ et le raisonnement est identique.

Exercice 5

a) On détermine si un mot appartient à $\mathscr{D}$ en calculant sa valuation ainsi que celle de ses préfixes.

```
let dick =
  let rec aux acc = function
    | []   -> acc = 0
    | a::q -> acc >= 0 && aux (acc+1) q
    | b::q -> acc >= 0 && aux (acc-1) q
  in aux 0 ;;
```

b) Soit $m = m_1 m_2 \cdots m_p \in \mathscr{D}$, et $\varphi(1), \varphi(2), \ldots, \varphi(k+1) = p$ les valeurs de j pour lesquelles $\sigma(m_1 m_2 \cdots m_j) = 0$. Alors $(m_1 \cdots m_{\varphi(1)})$, $(m_{\varphi(1)+1} \cdots m_{\varphi(2)})$, …, $(m_{\varphi(k)+1} \cdots m_p)$ sont les facteurs de m. Ainsi, déterminer le nombre de facteurs revient à dénombrer le nombre de préfixes p de m de valuation nulle :

```
let nb_facteurs m =
  let rec aux acc nb = function
    | []              -> nb
    | b::q when acc=1 -> aux 0 (nb+1) q
    | b::q            -> aux (acc-1) nb q
    | a::q            -> aux (acc+1) nb q
  in aux 0 0 m ;;
```

(On part du principe que cette fonction n'est utilisée que sur les mots de $\mathscr{D}$.)
Pour obtenir la fonction d'affichage, on remplace le comptage par des effets de bord dans la fonction précédente :

```
let factorisation m =
  let rec aux acc = function
    | []              -> ()
    | [b]             -> print_char `)`
    | b::q when acc=1 -> print_string ")." ; aux 0 q
    | b::q            -> print_char `)` ; aux (acc-1) q
    | a::q            -> print_char `(` ; aux (acc+1) q
  in aux 0 m ;;
```

Illustration :

```
# factorisation [a;a;b;b;a;b;a;a;b;a;b;b] ;;
(()).().(()()) - : unit = ()
```

Exercice 6

a) Nous avons successivement : $\overline{aababa} = \overline{ababab} = \overline{bababb} = \overline{abaabb} = \overline{bababb} = \overline{a}ababb = bababb$.
Montrons par récurrence sur $|r|$ que $\overline{rs} = \overline{s}\,\overline{r}$.

- C'est évident si $r = \varepsilon$.
- Si $|r| > 1$, on suppose le résultat acquis pour tout mot de longueur inférieure, et on considère deux cas :
  - si $r = ar'$, alors $\overline{rs} = \overline{r's}b = \overline{s}\overline{r'}b = \overline{s}\overline{ar'} = \overline{s}\,\overline{r}$;
  - si $r = br'$, alors $\overline{rs} = \overline{r's}a = \overline{s}\overline{r'}a = \overline{s}\overline{br'} = \overline{s}\,\overline{r}$.

b) Concrètement, il faut remplacer des a par des b (et réciproquement) puis prendre l'image miroir du mot.
On utilise un fold left :

```
let invert = function
  | a -> b
  | b -> a ;;

let barre = it_list (fun q t -> (invert t)::q) [] ;;
```

Illustration :

```
# barre [a;a;b;a;b;a] ;;
- : alphabet list = [b; a; b; a; b; b]
```

c) On observe que $u_{n+1} = u_n a \overline{u_n}$, ce qui conduit à la définition :

```
let rec pliage = function
  | 0 -> []
  | n -> let u = pliage (n-1) in u@(a::(barre u)) ;;
```

d) On peut commencer par observer que $|u_{n+1}| = 2|u_n| + 1$, donc $|u_n| = 2^n - 1$.
Montrons par récurrence sur n que les lettres d'indice $4p+1$ de u_n sont des a et les lettres d'indices $4p+3$ des b :

- c'est clair lorsque $n \leqslant 2$ car $u_0 = \varepsilon$, $u_1 = a$ et $u_2 = aab$;
- si $n \geqslant 2$, supposons que u_n s'écrive $u_n = aw_2bw_4aw_6\cdots aw_{2k}b$ (il se termine par un b car $2^n - 1 \equiv 3 \mod 4$). Alors :
$$u_{n+1} = aw_2bw_4aw_6\cdots aw_{2k}baa\overline{w}_{2k}b\overline{w}_{2k-2}a\cdots a\overline{w}_2b$$
  ce qui établit le résultat au rang $n+1$.

On peut en outre noter que si on considère le mot v_n formé des lettres d'indices pairs de u_n, à savoir $v_n = w_2w_4\cdots w_{2k}$ alors $v_{n+1} = v_n a \overline{v}_n$. Sachant que $v_1 = \varepsilon$, on en déduit par récurrence que $v_n = u_{n-1}$ pour $n \geqslant 1$, et donc que pour tout $n \in \mathbb{N}$, $w_{2n} = w_n$.
Tout ceci nous permet d'obtenir la fonction de calcul de la n^{e} lettre suivante :

```
let rec lettre = function
  | n when n mod 4 = 1 -> a
  | n when n mod 4 = 3 -> b
  | n                  -> lettre (n/2) ;;
```

Par exemple, $w_{2000} = w_{1000} = w_{500} = w_{250} = w_{125} = a$ car $125 = 4 \times 31 + 1$.

Exercice 7

a) Tout mot de plus de trois lettres sur l'alphabet a, b et sans facteur carré possède l'un des deux préfixes suivants : aba ou bab (les six autres facteurs possibles comportent tous un carré). S'il possède au moins quatre lettres, le préfixe suivant sera l'un des quatre mots : $abaa$, $abab$, $baba$, $babb$, qui tous possèdent un facteur carré.

b) Montrons par récurrence sur $n \in \mathbb{N}$ que $\sigma^n(a)$ est préfixe de $\sigma^{n+1}(a)$:

- c'est le cas lorsque $n = 0$ puisque $\sigma^0(a) = a$ et $\sigma(a) = ab$;
- si $n \geqslant 1$, supposons que $\sigma^{n-1}(a)$ soit préfixe de $\sigma^n(a)$: on peut écrire $\sigma^n(a) = \sigma^{n-1}(a)u$. Alors $\sigma^{n+1}(a) = \sigma^n(a)\sigma(u)$, donc $\sigma^n(a)$ est bien préfixe de $\sigma^{n+1}(a)$.

c) Il est évident que tous les mots de Σ_1^* comportent le même nombre de a que de b. Si c'est le cas du mot s, ce ne peut être le cas des mots asa et bsb.
Montrons maintenant par récurrence sur $n \in \mathbb{N}^*$ que $\sigma^n(a) \in \Sigma_1^*$.

- C'est le cas lorsque $n = 1$ puisque $\sigma(a) = ab$.
- Si $n > 1$, supposons que $\sigma^{n-1}(a)$ appartienne à Σ_1. On peut donc écrire $\sigma^{n-1}(a) = u_1 u_2 \cdots u_p$ avec $u_i \in \Sigma_1$, et $\sigma^n(a) = \sigma(u_1)\sigma(u_2)\cdots\sigma(u_p)$. Or $\sigma(ab) = abba \in \Sigma_1^*$ et $\sigma(ba) = baab \in \Sigma_1^*$ donc $\sigma^n(a) \in \Sigma_1^*$.

d) Supposons que m possède un facteur de la forme r^2x, où x est la première lettre de r, et choisissons ce facteur de longueur minimale. Notons $r = xs$; ainsi, $xsxsx$ est facteur de m et donc de $\sigma^n(a)$ pour un certain entier $n \geqslant 1$.
Puisque $\sigma^n(a)$ appartient à Σ_1^*, et suivant la position du facteur $xsxsx$ dans le mot m, l'un des mots $xsxsxy$ ou $yxsxsx$ (y désignant la lettre qui succède ou qui précède ce facteur dans m) appartient à Σ_1^* et possède un antécédent par σ, lui-même facteur de m.
Distinguons alors plusieurs cas suivant la parité de $|s|$:

- si $xsxsxy \in \Sigma_1^*$ et si $|s|$ est pair, $(xsx)s(xy)$ appartient à Σ_1^* donc xsx et s appartiennent à Σ_1^*, ce qui ne se peut d'après la question précédente ;
- si $xsxsxy \in \Sigma_1^*$ et si $|s|$ est impair, $(xs)(xs)xy = \sigma(s's'x)$, et $s's'x$ est un facteur de m qui contredit le caractère minimal de $|r|$ car la première lettre de s' est nécessairement un x (puisque $\sigma(s') = xs$) ;
- si $yxsxsx \in \Sigma_1^*$ et si $|s|$ est pair, $(yx)s(xsx)$ appartient à Σ_1^* donc s et xsx appartiennent à Σ_1^*, ce qui ne se peut d'après la question précédente ;
- si $yxsxsx \in \Sigma_1^*$ et si $|s|$ est impair, $yx(sx)(sx) = \sigma(ys's')$, y étant la dernière lettre de s'. En posant $s' = s''y$ on obtient un facteur $ys''ys''y = (ys'')(ys'')y$ de m qui contredit le caractère minimal de $|r|$.

e) D'après ce qui précède, bbb n'est pas facteur de m, donc l'alphabet $\{0, 1, 2\}$ suffit à écrire le mot μ.
Supposons maintenant que μ possède deux facteurs consécutifs égaux. Ces derniers ont pour origine un facteur de m de la forme $asasa$, autrement dit de la forme r^2x, où x est la première lettre de r. Cette situation est impossible ; on en déduit que μ ne possède pas deux facteurs consécutifs égaux.

f) On définit le type :

```
type alphabet = a| b ;;
```

puis les fonctions :

```
let rec sigma = function
  | []   -> []
  | a::q -> a::b::(sigma q)
  | b::q -> b::a::(sigma q) ;;

let rec itere_sigma = function
  | 0 -> [a]
  | n -> sigma (itere_sigma (n-1)) ;;
```

Il nous reste à écrire une fonction qui calcule le nombre de b entre deux a consécutifs d'un mot de $\{a, b\}^*$:

```
let rec filtre = function
  | a::a::q       -> 0::(filtre (a::q))
  | a::b::a::q    -> 1::(filtre (a::q))
  | a::b::b::a::q -> 2::(filtre (a::q))
  | _             -> [] ;;
```

En composant les deux dernières fonctions on obtient des préfixes de longueur arbitraire de mot μ. Par exemple :

```
# do_list print_int (filtre (itere_sigma 6)) ;;
2102012101202102012021012102012- : unit = ()
```

Ainsi, $\mu = 2102012101202102012021012102012101202101210201202102012101202 10\cdots$

• Expressions et langages rationnels

Exercice 8

- $(\varepsilon+\Sigma)(\varepsilon+\Sigma)$ dénote l'ensemble des mots ayant au plus deux lettres.
- $(\Sigma^2)^*$ dénote le langage des mots ayant un nombre pair de lettres.
- $(b+ab)^*(a+\varepsilon)$ dénote le langage des mots sur l'alphabet $\{a,b\}$ dans lesquels il n'existe pas deux a consécutifs.
- $(ab^*a+b)^*$ dénote le langage de tous les mots sur l'alphabet $\{a,b\}$ qui contiennent un nombre pair de a.

Exercice 9

- $(a+b)^*a(a+b)^*$ dénote le langage des mots qui contiennent au moins un a ;
- $b^*(a+\varepsilon)b^* \equiv b^*ab^*+b^*$ dénote le langage des mots qui contiennent au plus un a ;
- $(aa+b)^*$ dénote le langage des mots dans lequel toute série de a est de longueur paire ;
- $\Sigma(\Sigma^3)^*+\Sigma^2(\Sigma^3)^*$ dénote le langage des mots dont la longueur n'est pas un multiple de 3 ;
- $(b+\varepsilon)(ab)^*(a+\varepsilon) \equiv (ab)^*+(ba)^*+a(ba)^*+b(ab)^*$ dénote le langage des mots sur $\{a,b\}$ ne contenant pas deux lettres consécutives ;
- enfin, le langage des mots sur $\{a,b,c\}$ qui ne contient pas deux lettres consécutives peut être dénoté par :

$$(c+\varepsilon)\Big(\big((ab)^+ + (ba)^+\big)c\Big)^*\Big((ab)^*+(ba)^*\Big)$$

Exercice 10 En identifiant une expression rationnelle avec le langage qu'elle dénote on dispose des inclusions suivantes :

- $(e+f)^* \subset e^*(e+f)^*$ et $e^* \subset (e+f)^* \Longrightarrow e^*(e+f)^* \subset (e+f)^*$, ce qui prouve l'équivalence $(e+f)^* \equiv e^*(e+f)^*$;
- $e+f \subset e^*+f \Longrightarrow (e+f)^* \subset (e^*+f)^*$ et $e^*+f \subset (e+f)^* \Longrightarrow (e^*+f)^* \subset (e+f)^*$ ce qui prouve l'équivalence $(e+f)^* \equiv (e^*+f)^*$;
- $e+f \subset e^*f^* \Longrightarrow (e+f)^* \subset (e^*f^*)^*$ et $e^*f^* \subset (e+f)^* \Longrightarrow (e^*f^*)^* \subset (e+f)^*$ ce qui prouve l'équivalence $(e+f)^* \equiv (e^*f^*)^*$;
- $e^*f \subset (e+f)^* \Longrightarrow (e^*f)^*e^* \subset (e+f)^*e^* = (e+f)^*$ et $e+f \subset (e^*f)^*e^* \Longrightarrow (e+f)^* \subset ((e^*f)^*e^*)^* = (e^*f)^*e^*$, ce qui prouve l'équivalence $(e+f)^* \equiv (e^*f)^*e^*$.

Exercice 11 $(b^*a^2b^*)^*$ dénote le langage des mots sur l'alphabet $\{a^2,b\}$ qui ne contiennent pas que des b et $(a^*b^2a^*)^*$ le langage des mots sur l'alphabet $\{a,b^2\}$ qui ne contiennent pas que des a. Leur intersection est donc le langage des mots sur l'alphabet $\{a^2,b^2\}$ qui contiennent au moins un a^2 et un b^2 (plus le mot vide). Il est dénoté par $\big((b^2)^*a^2(b^2)^*+(a^2)^*b^2(a^2)^*\big)^*$.

Exercice 12 Notons que $\mathrm{L}=a^*b$ est solution puisque $a(a^*b)+b=aa^*b+b=(aa^*+\varepsilon)b=a^*b$.
Si L est une solution alors $\mathrm{L}=a(a\mathrm{L}+b)+b=a^2\mathrm{L}+ab+b$ et en réitérant ce procédé on prouve par récurrence sur n que $\mathrm{L}=a^{n+1}\mathrm{L}+\sum_{k=0}^{n}a^kb$. En particulier on a $a^nb\in\mathrm{L}$ pour tout $n\in\mathbb{N}$ donc $a^*b\subset\mathrm{L}$.
Réciproquement, si $m\in\mathrm{L}$ posons $n=|m|$. Les mots de $a^{n+1}\mathrm{L}$ sont de longueurs au moins égales à $n+1$ donc d'après l'égalité précédente $m\in\sum_{k=0}^{n}a^kb$ et nécessairement $m=a^{n-1}b\in a^*b$.
La preuve s'étend sans modification majeure pour résoudre $\mathrm{L}=\mathrm{AL}+\mathrm{B}$: on établit par récurrence que $\mathrm{L}=\mathrm{A}^{n+1}\mathrm{L}+\sum_{k=0}^{n}\mathrm{A}^k\mathrm{B}$, ce qui prouve que pour tout $n\in\mathbb{N}$, $\mathrm{A}^n\mathrm{B}\subset\mathrm{L}$ et donc que $\mathrm{A}^*\mathrm{B}\subset\mathrm{L}$, et réciproquement puisque $\varepsilon\notin\mathrm{A}$ tout mot de $\mathrm{A}^{n+1}\mathrm{L}$ est au moins de longueur $n+1$ donc un mot de L de longueur n appartient forcément à $\sum_{k=0}^{n}\mathrm{A}^k\mathrm{B}\subset\mathrm{A}^*\mathrm{B}$.

Application. On a $\mathrm{L}_1=a\mathrm{L}_1+b\mathrm{L}_2+\varepsilon$ et $\mathrm{L}_2=a\mathrm{L}_2+b\mathrm{L}_1$. Puisque $\varepsilon\notin\{a\}$ on en déduit d'après le lemme d'Arden que $\mathrm{L}_2=a^*b\mathrm{L}_1$ donc la première équation s'écrit aussi $\mathrm{L}_1=a\mathrm{L}_1+ba^*b\mathrm{L}_1+\varepsilon=(a+ba^*b)\mathrm{L}_1+\varepsilon$ et le lemme d'Arden donne cette fois $\mathrm{L}_1=(a+ba^*b)^*$. On a donc aussi $\mathrm{L}_2=a^*b(a+ba^*b)^*$.

• Langages locaux

Exercice 13 Notons Σ' l'ensemble des lettres qui composent les mots de L. Alors $P(L') = S(L') = \Sigma'$ et $F(L') = F(L)$. Considérons alors un mot u de $(\Sigma'\Sigma^* \cap \Sigma^*\Sigma') \setminus N(L)$ et décomposons-le en lettres : $u = a_1 \cdots a_p$. Nous savons déjà que $a_k a_{k+1} \in F(L)$.
Puisque a_1 est une lettre d'au moins un mot de L, il existe des lettres $b_1, \ldots, b_i$ telles que $b_1 \cdots b_i a_1$ soit préfixe d'un mot de L (avec éventuellement $i = 0$).
De même, il existe des lettres $c_1, \ldots, c_j$ telles que $a_p c_1 \cdots c_j$ soit suffixe d'un mot de L (avec éventuellement $j = 0$). Nous avons alors :

- $b_1 \in P(L)$ (ou a_1 si $i = 0$) ;
- $c_j \in S(L)$ (ou a_p si $j = p$) ;
- $b_k b_{k+1}, \ldots, b_i a_1, \ldots a_k a_{k+1}, \ldots a_p c_1, \ldots c_k c_{k+1} \in F(L)$.

Puisque L est un langage local on en déduit que $b_1 \cdots b_i a_1 \cdots a_p c_1 \cdots c_j \in L$, et donc que $a_1 \cdots a_p$ est facteur d'un mot de L donc un élément de L′. Ceci prouve que L′ est un langage local.

Exercice 14 Supposons que L soit un langage local et considérons $u, v, u', v' \in \Sigma^*$ et $a \in \Sigma$ tel que $uav \in L$ et $u'av' \in L$.
La première lettre de u est dans P(L) (ou a si $u = \varepsilon$), la dernière lettre de v' est dans S(L) (ou a si $v' = \varepsilon$), et tout facteur de longueur 2 de ua et de av' est dans F(L). Puisque L est un langage local, $uav' \in L$.
Réciproquement, supposons la propriété vérifiée et montrons que L est un langage local. Pour ce faire, on considère un mot $m = a_1 \cdots a_p$ tel que $a_1 \in P(L)$, $a_p \in S(L)$ et $a_k a_{k+1} \in F(L)$.
$a_1 \in P(L)$ donc il existe v_1 tel que $\varepsilon a_1 v_1 \in L$.
$a_1 a_2 \in F(L)$ donc il existe u_1, v_2 tels que $u_1 a_1 a_2 v_2 \in L$. On en déduit que $\varepsilon a_1 a_2 v_2 \in L$.
$a_2 a_3 \in F(L)$ donc il existe u_2, v_3 tel que $u_2 a_2 a_3 v_3 \in L$. On en déduit que $\varepsilon a_1 a_2 a_3 v_3 \in L$.
De proche en proche on construit un mot v_p tel que $\varepsilon a_1 a_2 \cdots a_p v_p \in L$.
Enfin, puisque $a_p \in S(L)$ il existe u_p tel que $u_p a_p \varepsilon \in L$, ce qui prouve que $\varepsilon a_1 a_2 \cdots a_p \varepsilon = a_1 \cdots a_p \in L$. L est bien un langage local.

• Exercices divers

Exercice 15

a) $L = \{a, ba, bba, bbba\}$ est un code de quatre mots sur $\Sigma = \{a, b\}$.

b) L_1 n'est pas un code car $a.ba = ab.a$. L_2 est un code en raisonnant par l'absurde : s'il existait deux factorisations $u_1 \cdots u_p$ et $v_1 \cdots v_q$ du même mot, on peut supposer p minimal et considérer le dernier mot u_p : si $u_p = a$ nécessairement $v_p = a$; si $u_p = ab$ nécessairement $v_q = ab$. Dans les deux cas $u_p = v_q$ et donc $u_1 \cdots u_{p-1} = v_1 \cdots v_{q-1}$ ce qui contredit le caractère minimal de p.

c) L'égalité $u_1 \cdots u_p = v_1 \cdots v_q$ implique que u_1 est préfixe de v_1 ou v_1 préfixe de u_1 ; si tous les mots de L ont même longueur on a donc $u_1 = v_1$ puis par récurrence $p = q$ et $u_i = v_i$. L est un code.

d) Si $uv = vu$ alors $\{u, v\}$ n'est pas un code.
Réciproquement, on suppose que $uv \neq vu$ et on prouve par récurrence sur $|uv|$ que $\{u, v\}$ est un code.

- si $|uv| = 2$, u et v sont deux lettres et d'après la question précédente $\{u, v\}$ est un code.
- Supposons maintenant le résultat acquis pour tout couple de mots (u', v') tel que $u'v' \neq v'u'$ et $|u'v'| < |uv|$. Considérons l'égalité : $u_1 \cdots u_p = v_1 \cdots v_q$. Quitte à simplifier par un préfixe commun on peut supposer $u_1 \neq v_1$ puis par exemple $u_1 = u$, $v_1 = v$. Supposons en outre $|u| \leqslant |v|$. u est alors préfixe de v donc on peut poser $v = uv'$. La relation $uv \neq vu$ se simplifie en $uv' \neq v'u$ et l'égalité $u_1 \cdots u_p = v_1 \cdots v_q$ en $u_2 \cdots u_p = v'v_2 \cdots v_q$. Mais ces deux mots appartiennent à $\{u, v'\}^*$ donc par hypothèse de récurrence ne peuvent se factoriser que d'une unique manière sur $\{u, v'\}^*$. On en déduit que v' est un préfixe de u_2. Mais u_2 est aussi un mot de $\{u, v\}^* = \{u, uv'\}^*$ donc u est préfixe de u_2. Par unicité de la factorisation on doit avoir $u = v'$ ce qui est absurde puisque $uv' \neq v'u$.

e) Si $u_1 \cdots u_p = v_1 \cdots v_q$ on peut sans perte de généralité supposer quitte à simplifier par un préfixe commun que $u_1 \neq v_1$ et $|u_1| \leqslant |v_1|$. Mais alors u_1 est un préfixe propre de v_1, ce qui est absurde.

f) Considérons deux codes préfixes L_1 et L_2, et $L = L_1 L_2$. Supposons que L contienne deux mots u et v tel que u soit préfixe propre de v. On peut écrire $u = u_1 u_2$ et $v = v_1 v_2$ avec $u_1, v_1 \in L_1$ et $u_2, v_2 \in L_2$. u_1 est donc aussi préfixe de $v = v_1 v_2$. Envisageons alors deux cas :

– si u_1 est préfixe de v_1, il ne peut en être un préfixe propre donc $u_1 = v_1$ mais dans ce cad u_2 est préfixe propre de v_2, ce qui est absurde ;
– mais dans le cas contraire v_1 est préfixe propre de u_1 ce qui là encore est absurde.

On en déduit que L_1L_2 est encore un code préfixe.

g) Pour L_1 on obtient la suite $S_0 = \{ba\}$, $S_1 = \{a\}$, $S_2 = \{b, bba, abaa\}$, $S_3 = \{aa\}$, $S_4 = \{baa\}$, $S_5 = \{\varepsilon\}$ donc L_1 n'est pas un code (en effet, $abba.ab.aabaa = ab.baa.baa.baa$).

Pour L_2 on obtient $S_0 = \{ab\}$, $S_1 = \{a, b, aab\}$, $S_2 = \{aa, ba, bb, baab\}$, $S_3 = \{a\}$, $S_4 = \{aa, ba, bb, baab\} = S_2$ donc L_2 est un code.

Pour L_3 on obtient $S_0 = \{bba\}$, $S_1 = \emptyset$, $S_2 = \emptyset = S_1$ donc L_3 est un code.

Pour L_4 on obtient $S_0 = \{a\}$, $S_1 = \{a, b\}$, $S_2 = \{\varepsilon, a, b\}$ donc L_4 n'est pas un code (en effet, $ba.ab = b.aa.b$).

Pour L_5 on obtient $S_0 = \{aababa\}$, $S_1 = \{baba\}$, $S_2 = \emptyset$, $S_3 = \emptyset = S_2$ donc L_5 est un code.

Pour L_6 on obtient $S_0 = \{b\}$, $S_1 = \{ba, bab, bbb\}$, $S_2 = \{b\} = S_0$ donc L_6 est un code.

h) On peut observer que pour tout $i \in \mathbb{N}$, S_i est inclus dans l'ensemble des suffixes des mots de L. Puisque L est fini la suite $(S_i)_{i\in\mathbb{N}}$ ne peut prendre qu'un nombre fini de valeurs et est donc périodique à partir d'un certain rang. Ceci assure la terminaison de l'algorithme.