TD Théorie des Langages 1

Sélection d'exercices corrigés

Exercice 1 Soit V un vocabulaire et soit un sous-ensemble $A \subseteq V$. Dans cet exercice on considérera la fonction $|\,.\,|_A : V^* \to \mathbb{N}$ qui à tout mot $w \in V^*$ associe le nombre d'occurrences d'éléments de A présents dans w. Ainsi, si $V = \{a, b, c, d\}$ et $A = \{a, b\}$, alors :

— $|cabdbacc|_A = 4$,
— $|bbaba|_A = 5$,
— $|ccdc|_A = 0$.

On pose $V = \{\wedge, \vee, \neg,), (, \bot, \top\}$, et on considère l'ensemble E des formules logiques défini par induction structurelle de la façon suivante :

Base. $\top \in E$ et $\bot \in E$.

Induction. Si w, w_1 et w_2 sont dans E, alors :
— $(\neg\, w) \in E$,
— $(w_1 \wedge w_2) \in E$,
— $(w_1 \vee w_2) \in E$.

▷ Question 1 Est-ce que les deux mots suivants sont dans E ? On justifiera en quelques mots les réponses.

1. $(\top \wedge \neg)$
2. $(\top \wedge \top \wedge \top)$

▷ Question 2 Montrer que $((\top \wedge \top) \vee (\bot \vee \top)) \in E$.

On définit les ensembles de symboles suivants :

$$U = \{\neg\}, \qquad B = \{\wedge, \vee\},$$
$$S = \{\top, \bot\}, \qquad N = U \cup B \cup S.$$

▷ Question 3 Soit $w_1 = (\top \wedge ((\neg\bot) \vee (\top \wedge (\bot \vee (\neg\top)))))$. Calculer $|w_1|_U$, $|w_1|_B$ et $|w_1|_S$.

▷ Question 4 Démontrer par induction structurelle que pour tout $w \in E$, on a $|w|_S = |w|_B + 1$.

▷ Question 5 **[Avancé]** Soit $w \in E$. Exprimer $|w|$ en fonction de $|w|_B$ et $|w|_U$.

▷ Question 6 **[Avancé]** Utiliser la question 5 pour justifier formellement les réponses à la question 1.

Solution de l'Exercice 1.

▷ Question 1 Le symbole $\neg$ ne peut apparaître qu'après « (» et il manque des parenthèses pour le deuxième mot (pour chaque $\wedge$ il doit y avoir une parenthèse ouvrante).

▷ Question 2 Pour montrer qu'un terme appartient à un ensemble défini par induction, il faut le construire en utilisant les constructeurs et les cas de base. Notons $u = (\top \wedge \top)$ et $v = (\bot \vee \top)$. On a $u \in E$ par application du deuxième constructeur inductif à deux fois le premier cas de base. De même, $v \in E$ par application du troisième constructeur inductif au deuxième et premier cas de base. Au final, le mot $((\top \wedge \top) \vee (\bot \vee \top))$ appartient à E par application du troisième constructeur inductif à u et v.

▷ Question 3 On a $|w_1|_U = 2$, $|w_1|_B = 4$ et $|w_1|_S = 5$.

▷ Question 4 **Rappel :** Pour faire une démonstration par induction structurelle d'une propriété P sur un ensemble E défini inductivement, on doit démontrer P pour les cas de base et démontrer que pour chaque constructeur inductif κ, si on suppose P pour tous les arguments $u_1, \ldots, u_k$ de κ (d'arité k), alors on peut démontrer P sur le mot $\kappa(u_1, \ldots, u_k)$.

Ici, on a deux cas de base et trois constructeurs inductifs, donc cinq cas à considérer.

Montrons la propriété [1] $P[w] = |w|_S = |w|_B + 1$ par induction structrelle sur E :
— Pour $\top$: $|\top|_S = 1 = 0 + 1 = |\top|_B + 1$.
— Pour $\bot$: $|\bot|_S = 1 = 0 + 1 = |\bot|_B + 1$.

1. J'utilise les crochets [_] car les parenthèses font partie du vocabulaire V.

— Pour $(\neg u)$: Soit u dans E et supposons $P[u]$. On a

$$\begin{aligned} |(\neg u)|_S &= |u|_S \\ &\overset{HI}{=} |u|_B + 1 \\ &= |(\neg u)|_B + 1 \end{aligned}$$

Ainsi, on a montré $P[(\neg u)]$.

— Pour $(u \wedge v)$: Soient u, v dans E et supposons $P[u]$ et $P[v]$. On a

$$\begin{aligned} |(u \wedge v)|_S &= |u|_S + |v|_S \\ &\overset{HI}{=} (|u|_B + 1) + (|v|_B + 1) \\ &= (1 + |u|_B + |v|_B) + 1 \\ &= |(u \wedge v)|_B + 1 \end{aligned}$$

Ainsi, on a montré $P[(u \wedge v)]$.

— Pour $(u \vee v)$: Soient u, v dans E et supposons $P[u]$ et $P[v]$. On a

$$\begin{aligned} |(u \vee v)|_S &= |u|_S + |v|_S \\ &\overset{HI}{=} (|u|_B + 1) + (|v|_B + 1) \\ &= (|u|_B + |v|_B + 1) + 1 \\ &= |(u \vee v)|_B + 1 \end{aligned}$$

Ainsi, on a montré $P[(u \vee v)]$.

▷ Question 5 On a $|w| = 4|w|_B + 3|w|_U + 1$. En effet, $|w| = |w|_N + |w|_{\{),(\}} = (2|w|_B + |w|_U + 1) + (2|w|_B + 2|w|_U)$.

▷ Question 6 L'égalité de la question précédente n'est pas vérifiée pour les mots de la question 1, ils ne font donc pas partie de E.

Exercice 6 [A savoir faire] Soit L un langage. Montrer que les propositions suivantes sont équivalentes :

(i) $\varepsilon \in L$

(ii) $\forall i \geq 0$, $\varepsilon \in L^i$

(iii) $\forall i \geq 0$, $L^i \subseteq L^{i+1}$

Solution de l'Exercice 6.

(i) $\Rightarrow$ (ii) Supposons que $\varepsilon \in L$. On prouve par récurrence sur i qu'on a alors pour tout $i \geq 0$, $\varepsilon \in L^i$.

Base $L^0 = \{\varepsilon\}$ donc $\varepsilon \in L^0$

Induction Soit un entier i et supposons que $\varepsilon \in L^i$. Montrons $\varepsilon \in L^{i+1}$.
Comme $\varepsilon \in L$, on a $\varepsilon = \varepsilon.\varepsilon \in L.L^i = L^{i+1}$.

(ii) $\Rightarrow$ (iii) Supposons que $\forall i \geq 0$, $\varepsilon \in L^i$. Alors en particulier, $\varepsilon \in L$. Soit $n \geq 0$, montrons que $L^n \subseteq L^{n+1}$. Par définition, $L^{n+1} = L.L^n$, et comme $\varepsilon \in L$, on a $L^n = \{\varepsilon\}.L^n \subseteq L^{n+1}$.

(iii) $\Rightarrow$ (i) Supposons que $\forall i \geq 0$, $L^i \subseteq L^{i+1}$. Alors en particulier, $L^0 \subseteq L^1$, i.e., $\varepsilon \in L$.

Exercice 20 Le produit de deux automates est défini de la façon suivante : soient $A_1 = (Q_1, V, \delta_1, \{i_1\}, F_1)$ et $A_2 = (Q_2, V, \delta_2, \{i_2\}, F_2)$ deux automates qu'on suppose complets et sans ε-transition. On considère l'automate

$$A_1 \times A_2 \overset{\text{def}}{=} (Q_1 \times Q_2,\ V,\ \delta,\ \{\langle i_1, i_2\rangle\},\ F_1 \times F_2),$$

où δ est défini par : pour tous $\langle q_1, q_2\rangle, \langle q_1', q_2'\rangle \in Q_1 \times Q_2$ et pour tout $a \in V$,

$$(\langle q_1, q_2\rangle, a, \langle q_1', q_2'\rangle) \in \delta \text{ si et seulement si } (q_1, a, q_1') \in \delta_1 \wedge (q_2, a, q_2') \in \delta_2.$$

▷ Question 1 Donner des automates complets et sans ε-transition qui reconnaissent :

1. les mots sur $\{0, 1\}$ contenant un nombre pair de 0 ;
2. les mots sur $\{0, 1\}$ contenant un nombre impair de 1.

▷ Question 2 Calculer l'automate produit des deux automates de la question précédente. Constater qu'il reconnaît les mots sur $\{0, 1\}$ contenant un nombre pair de 0 **et** un nombre impair de 1.

On considère dans la suite deux automates A_1 et A_2 quelconques, complets et sans ε-transition. On souhaite prouver que $\mathcal{L}(A_1 \times A_2) = \mathcal{L}(A_1) \cap \mathcal{L}(A_2)$.

▷ Question 3 **[Avancé]** Montrer par induction sur w l'équivalence entre les deux propositions suivantes :

1. il existe un chemin de trace w de l'origine p_1 à l'extrémité q_1 dans A_1
   et il existe un chemin de trace w de l'origine p_2 à l'extrémité q_2 dans A_2 ;
2. il existe un chemin de trace w de l'origine $\langle p_1, p_2\rangle$ à l'extrémité $\langle q_1, q_2\rangle$ dans $A_1 \times A_2$.

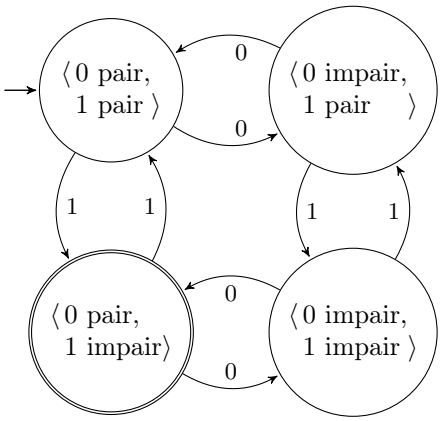
▷ Question 4 En déduire que $\mathcal{L}(A_1 \times A_2) = \mathcal{L}(A_1) \cap \mathcal{L}(A_2)$.

▷ Question 5 Quel langage aurait été reconnu si l'ensemble des états finaux de $A_1 \times A_2$ avait été $(F_1 \times Q_2) \cup (Q_1 \times F_2)$ et non pas $F_1 \times F_2$? Justifier.

Solution de l'Exercice 20.

▷ Question 1 Pour déterminer si le nombre de 0 dans un mot est pair, l'information à maintenir est la parité du nombre de 0, à mettre à jour à chaque nouveau symbole lu. Il faut donc deux états, un état « nombre pair de 0 » et un état « nombre impair de 0 ». De même pour un nombre impair de 1. Ainsi, on a les deux automates suivants, qui reconnaissent respectivement les mots avec un nombre pair de 0 et les mots avec un nombre impair de 1. Remarquez que les états initiaux sont ceux avec un nombre pair, car le mot vide contient un nombre pair de 0 et de 1.

▷ Question 2 L'intuition à garder en tête pour comprendre l'automate produit est d'exécuter en parallèle deux automates. Les états de l'automate produit sont donc des couples, dont la première composante sert à exécuter le premier automate et la seconde composante le second automate. L'automate produit est donc :



▷ Question 3 Le résultat se démontre par induction sur w.

Cas $w = \varepsilon$ **:** Le chemin vide () est un chemin de p_1 à p_1 (donc $q_1 = p_1$ ici) de trace ε dans A_1 et de même, () est un chemin de p_2 à p_2 de trace ε dans A_2 et () est un chemin de $\langle p_1, p_2\rangle$ à $\langle p_1, p_2\rangle$ de trace ε dans $A_1 \times A_2$.

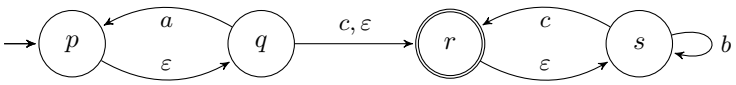
Cas $w = aw'$ **:** Comme A_1 ne contient pas d'ε-transitions, un chemin de trace aw' dans A_1 est de la forme $(p_1, a, r_1)\chi_1$, avec $(p_1, a, r_1) \in \delta_1$ et χ_1 un chemin de r_1 à q_1 de trace w' dans A_1. De même pour A_2 : un chemin de trace aw' est de la forme $(p_2, a, r_2)\chi_2$. L'hypothèse d'induction sur χ_1 et χ_2 donne un chemin χ dans $A_1 \times A_2$ de $\langle r_1, r_2\rangle$ à $\langle q_1, q_2\rangle$ de trace w'. De $(p_1, a, r_1) \in \delta_1$ et $(p_2, a, r_2) \in \delta_2$, la définition de δ nous donne que $(\langle p_1, p_2\rangle, a, \langle r_1, r_2\rangle) \in \delta$ donc $(\langle p_1, p_2\rangle, a, \langle r_1, r_2\rangle)\chi$ est un chemin de $\langle p_1, p_2\rangle$ à $\langle q_1, q_2\rangle$ de trace aw' dans $A_1 \times A_2$.

▷ Question 4 On peut raisonner directement par équivalence ici :

$$
\begin{array}{rcl}
w \in \mathcal{L}(A_1 \times A_2) & \overset{\text{def}}{\iff} & \text{il existe un chemin dans } A_1 \times A_2 \text{ d'origine } \langle i_1, i_2\rangle \text{ et d'extrémité } \langle q_1, q_2\rangle \in F_1 \times F_2 \\
& \overset{\text{quest 3}}{\iff} & \text{il existe un chemin de trace } w \text{ de l'origine } i_1 \text{ à l'extrémité } q_1 \in F_1 \text{ dans } A_1 \\
& & \text{et il existe un chemin de trace } w \text{ de l'origine } i_2 \text{ à l'extrémité } q_2 \in F_2 \text{ dans } A_2 \\
& \overset{\text{def}}{\iff} & w \in \mathcal{L}(A_1) \text{ et } w \in \mathcal{L}(A_2)
\end{array}
$$

▷ Question 5 Si l'ensemble des états finaux de $A_1 \times A_2$ avait été $(F_1 \times Q_2) \cup (Q_1 \times F_2)$, l'automate aurait reconnu $\mathcal{L}(A_1) \cup \mathcal{L}(A_2)$. On peut adapter la preuve de la question précédente pour le montrer proprement. En effet, $(F_1 \times Q_2) \cup (Q_1 \times F_2)$ signifie : la première composante accepte (cas $F_1 \times Q_2$) ou la seconde composante accepte (cas $Q_1 \times F_2$).

Exercice 24 [A savoir faire] Construire un automate sans ε-transition équivalent à celui ci-dessous :



Solution de l'Exercice 24. On calcule les ensembles d'états ε-accessibles, soit directement, soit par itération selon la méthode vue en cours :

$\mathrm{Acc}_\varepsilon(\_)$	p	q	r	s
$k=0$	p	q	r	s
$k=1$	p,q	q,r	r,s	s
$k=2$	p,q,r	q,r,s	r,s	
$k=3$	p,q,r,s	q,r,s		
$k=4$	p,q,r,s			

ce qui nous donne

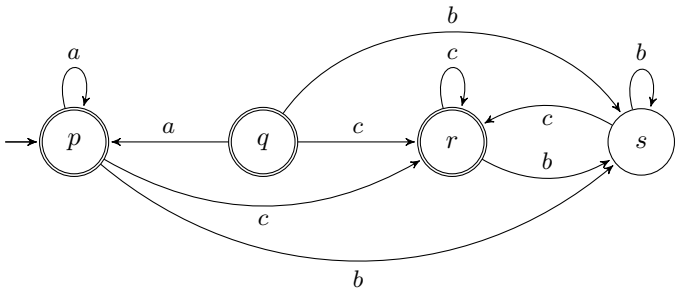
Q	$\mathrm{Acc}_\varepsilon(q)$
p	$\{p,q,r,s\}$
q	$\{q,r,s\}$
r	$\{r,s\}$
s	$\{s\}$

Notez que dans le tableau de calcul, on arrête chaque colonne lorsqu'elle a stabilisée.

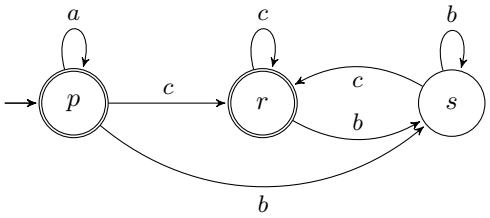
On en déduit la relation de transition de l'automate :

δ	a	b	c	initial/final
p	p	s	r	F
q	p	s	r	F
r	$\times$	s	r	F
s	$\times$	s	r	

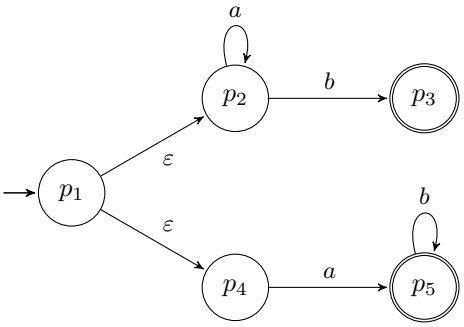
L'automate obtenu est donc :



On remarque que l'état q n'est plus accessible donc peut être supprimé.

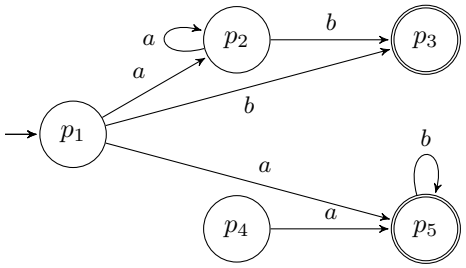


Exercice 28 [A savoir faire] Déterminiser l'automate suivant :



Solution de l'Exercice 28. L'ensemble des états ε-accessibles depuis p_1 est $\{p_1, p_2, p_4\}$; les autres ensembles sont des singletons. On en déduit la relation de transition suivante pour l'automate obtenu après suppression des ε-transitions :

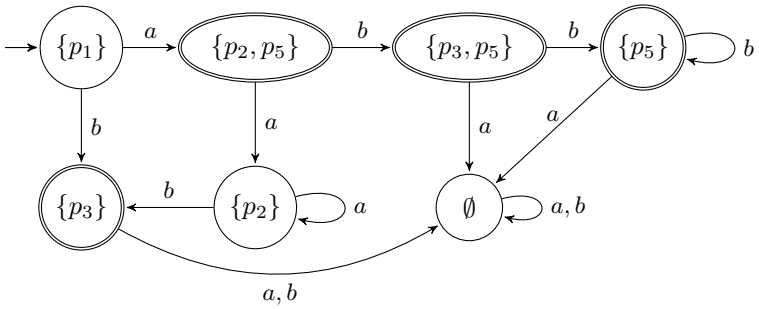
δ	a	b
p_1	p_2, p_5	p_3
p_2	p_2	p_3
p_3	-	-
p_4	p_5	-
p_5	-	p_5



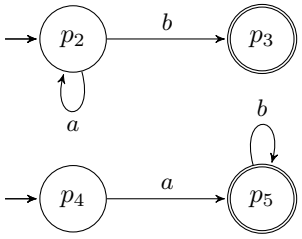
On remarque que p_4 n'est plus accessible et peut donc être supprimé, ce que la déterminisation va faire pour nous.

Déterminisation :

δ	a	b	I/F
p_1	p_2, p_5	p_3	I
p_2, p_5	p_2	p_3, p_5	F
p_3	$\emptyset$	$\emptyset$	F
p_2	p_2	p_3	
p_3, p_5	$\emptyset$	p_5	F
p_5	$\emptyset$	p_5	F
$\emptyset$	$\emptyset$	$\emptyset$	



Un des intérêts de l'exo : il est facile de décrire en français le langage reconnu sur l'automate d'origine (« autant de a qu'on veut puis un b, ou alors un a puis autant de b qu'on veut »), mais c'est plus dur sur l'AFD... Sur l'automate d'origine on voit bien l'union (avec les ε) ; on la verrait aussi bien avec

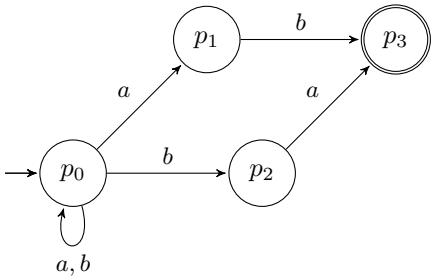


Exercice 29 [A savoir faire] Construire des automates déterministes reconnaissant les langages sur $\{a, b\}$ suivants :

1. L'ensemble des mots terminés par ab ou bien par ba.
2. L'ensemble des mots contenant au moins deux fois la séquence ab.
3. Le langage $\{aab\}^*\{b\}$.
4. Le langage $\{a\}^*\{aba\}^*$.

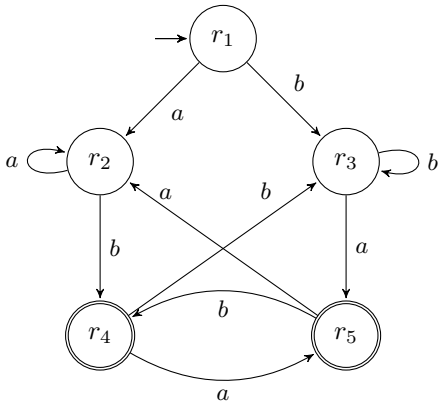
Solution de l'Exercice 29. Ici, on peut avoir tendance à chercher directement un automatate déterministe. Sur les questions 2 et 4, il est facile de se tromper...

1. Automate non-déterministe :

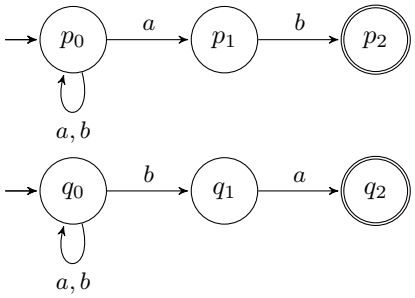


Déterminisation :

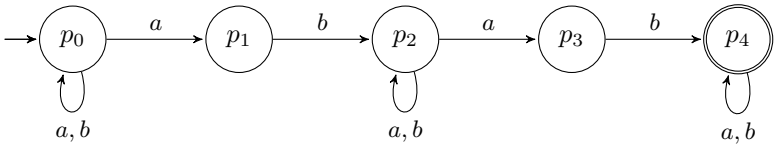
Nom	I/F	δ	a	b
r_1	I	p_0	p_0, p_1	p_0, p_2
r_2		p_0, p_1	p_0, p_1	p_0, p_2, p_3
r_3		p_0, p_2	p_0, p_1, p_3	p_0, p_2
r_4	F	p_0, p_2, p_3	p_0, p_1, p_3	p_0, p_2
r_5	F	p_0, p_1, p_3	p_0, p_1	p_0, p_2, p_3



On peut aussi partir de

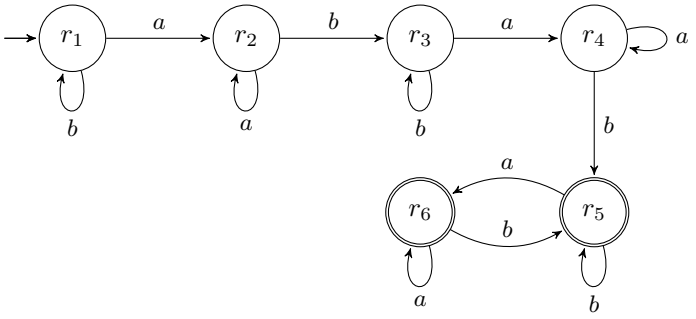


2. Automate non-déterministe :



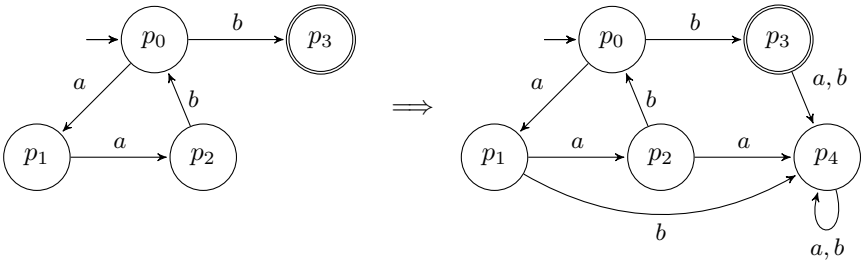
Déterminisation :

Nom	I/F	δ	a	b
r_1	I	p_0	p_0, p_1	p_0
r_2		p_0, p_1	p_0, p_1	p_0, p_2
r_3		p_0, p_2	p_0, p_1, p_2, p_3	p_0, p_2
r_4		p_0, p_1, p_2, p_3	p_0, p_1, p_2, p_3	p_0, p_2, p_4
r_5	F	p_0, p_2, p_4	p_0, p_1, p_2, p_3, p_4	p_0, p_2, p_4
r_6	F	p_0, p_1, p_2, p_3, p_4	p_0, p_1, p_2, p_3, p_4	p_0, p_2, p_4

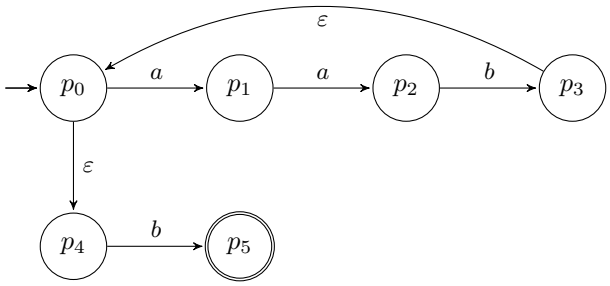


En question subsidiaire, on peut demander « exactement 2 fois la séquence ab ».

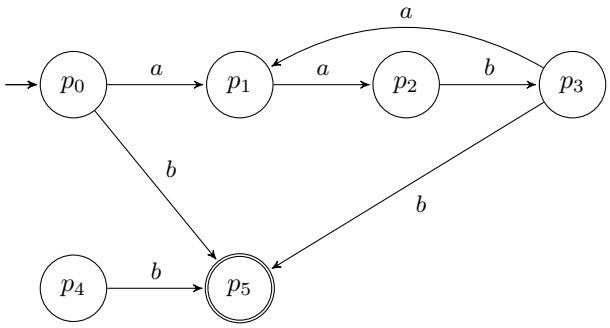
3. On peut directement construire un automate déterministe non complet, puis le compléter :



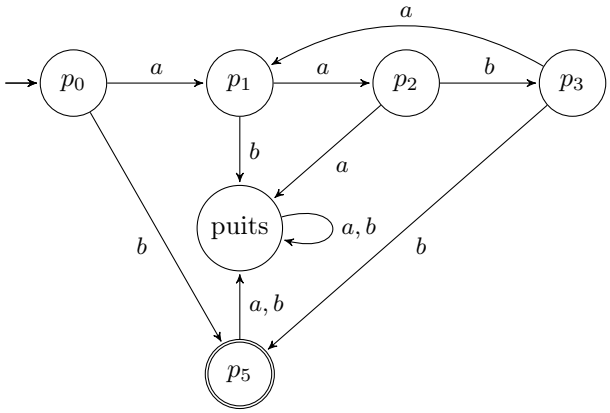
Ou bien passer par un automate non-déterministe, qu'on déterminise (et dont on supprime d'abord les ε-transitions au besoin) :



Après suppression des ε-transitions (non détaillée) :

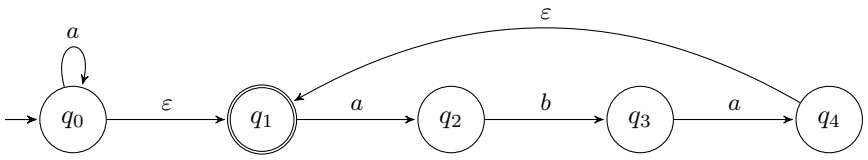


L'automate est déjà déterministe mais ni complet ni initialement connecté. On peut corriger cela en supprimant l'état p_4 pour devenir initialement connecté et en ajoutant un état puits pour devenir complet :



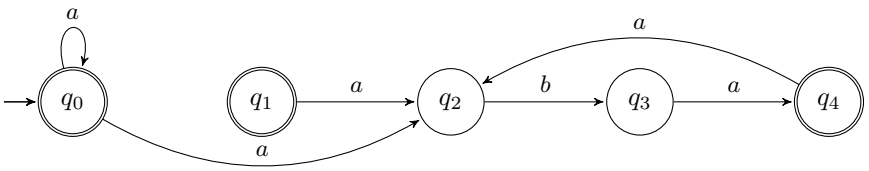
Remarquez que les automates déterministes obtenus par les deux méthodes n'ont pas le même nombre d'états mais sont équivalents.

4. Automate non déterministe :

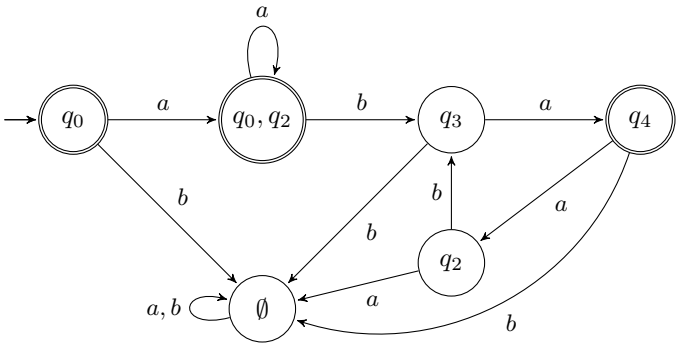


À comparer avec la question précédente : ici si on ne met pas d'ε-transition entre q_0 et q_1 ça ne marche pas... mais on peut faire directement aba en triangle (sans q_4 et son ε-transition vers q_1).

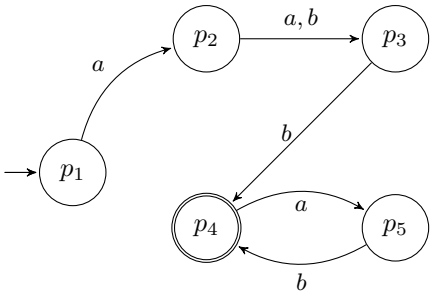
Automate non déterministe sans ε-transition :

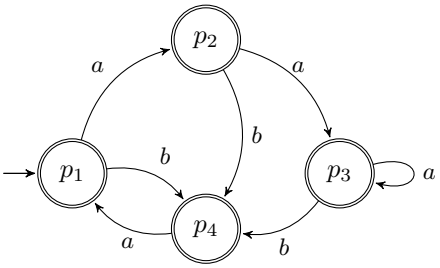


Automate déterminisé :



Exercice 35 Minimiser les automates suivants :



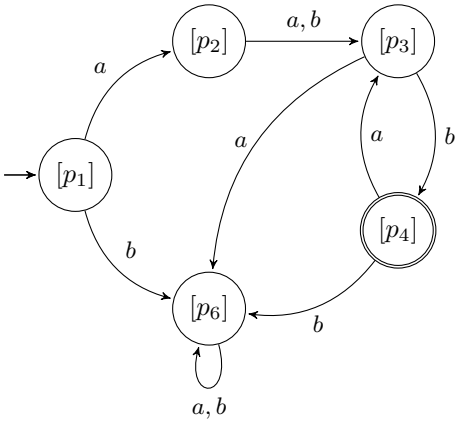


Solution de l'Exercice 35. Les automates sont déterministes mais non complets, il faut penser à ajouter un état puits.

Les premières classes (celles de $\equiv_0$) sont $Q \setminus F$ et F. Dans la suite, on détermine la classe $\equiv_{k+1}$ en fonction de la classe $\equiv_k$ et de la « signature » de chaque état, c.-à-d. des classes vers lesquelles on arrive après une transition par chacun des symboles du vocabulaire. Par exemple, la signature CD signifie qu'en lisant un a, on arrive dans la classe C et qu'en lisant un b, on arrive dans la classe D. On rappelle qu'il est inutile de traiter les classes singletons car elles ne peuvent pas diminuer d'avantage.

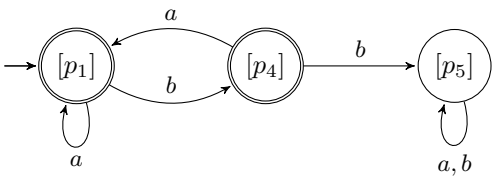
Premier automate : L'état p_6 est l'état puits qu'on a rajouté pour compléter l'automate.

$\equiv_0$	$\{p_1,$	$p_2,$	$p_3,$	p_5	$p_6\}$	$\{p_4\}$
noms des classes			A			B
« signatures »	AA	AA	AB	AB	AA	
$\equiv_1$	$\{p_1,$	$p_2,$	$p_6\}$	$\{p_3,$	$p_5\}$	$\{p_4\}$
noms des classes		C			D	B
« signatures »	CC	DD	CC	CB	CB	
$\equiv_2$	$\{p_1,$	$p_6\}$	$\{p_2\}$	$\{p_3,$	$p_5\}$	$\{p_4\}$
noms des classes	E		G		D	B
« signatures »	GE	EE		EB	EB	
$\equiv_3$	$\{p_1\}$	$\{p_6\}$	$\{p_2\}$	$\{p_3,$	$p_5\}$	$\{p_4\}$
noms des classes	H	I	G		D	B
« signatures »				IB	IB	
$\equiv_4$	$\equiv_3$					



Second automate : L'état p_5 est l'état puits, rajouté pour compléter l'automate.

$\equiv_0$	$\{p_5\}$	$\{p_1,$	$p_2,$	$p_3,$	$p_4\}$
noms des classes	A			B	
« signatures »		BB	BB	BB	BA
$\equiv_1$	$\{p_5\}$	$\{p_1,$	$p_2,$	$p_3\}$	$\{p_4\}$
noms des classes	A		C		D
« signatures »		CD	CD	CD	
$\equiv_2$	$\equiv_1$				



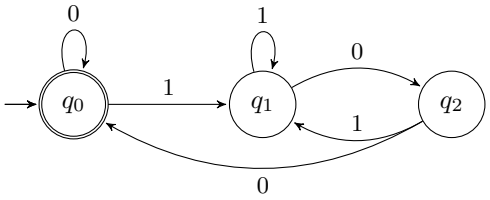
Exercice 38 [A savoir faire] Soit E une expression régulière. Simplifier les expressions suivantes :

1. $E.E^* + \epsilon$
2. $\epsilon.E$
3. $\emptyset^*$
4. ϵ^*
5. $\emptyset.E$
6. $\emptyset + E$

Solution de l'Exercice 38.

1. $E.E^* + \epsilon = E^*$
2. $\epsilon.E = E$
3. $\emptyset^* = \epsilon$
4. $\epsilon^* = \epsilon$
5. $\emptyset.E = \emptyset$
6. $\emptyset + E = E$

Exercice 40 Calculer l'expression régulière correspondant à l'automate ci-dessous, en résolvant le système d'équations obtenu dans deux ordres différents, puis en utilisant la méthode par suppression d'états dans les mêmes ordres. Constater que les systèmes associés aux automates avec certains états supprimés correspondent aux différentes étapes de résolution du système initial.



Solution de l'Exercice 40. Système d'équations associé :

$$\begin{cases} x_0 &= 0x_0 + 1x_1 + \varepsilon \\ x_1 &= 0x_2 + 1x_1 \\ x_2 &= 0x_0 + 1x_1 \end{cases}$$

— Élimination de x_2 puis x_1 :

$$\begin{cases} x_0 &= 0x_0 + 1x_1 + \varepsilon \\ x_1 &= 00x_0 + (1+01)x_1 \\ x_2 &= 0x_0 + 1x_1 \end{cases} \qquad \begin{cases} x_0 &= 0x_0 + 1(1+01)^*00x_0 + \varepsilon \\ x_1 &= (1+01)^*00x_0 \\ x_2 &= 0x_0 + 1x_1 \end{cases}$$

D'où

$$[0 + 1(1+01)^*00]^* .$$

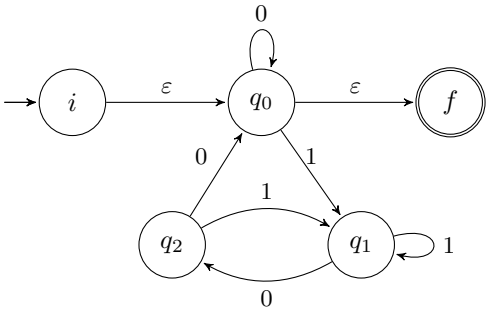
— Élimination de x_1 puis x_2 :

$$\begin{cases} x_0 &= 0x_0 + 1^+0x_2 + \varepsilon \\ x_1 &= 1^*0x_2 \\ x_2 &= 0x_0 + 1^+0x_2 \end{cases} \qquad \begin{cases} x_0 &= 0x_0 + (1^+0)^+0x_0 + \varepsilon \\ x_1 &= 1^*0x_2 \\ x_2 &= (1^+0)^*0x_0 \end{cases}$$

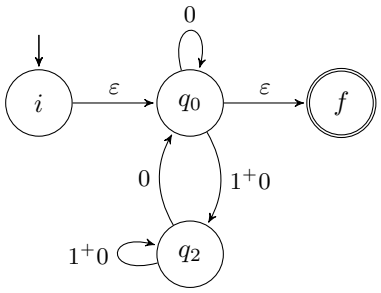
D'où

$$\left[0 + (1^+0)^+0\right]^* .$$

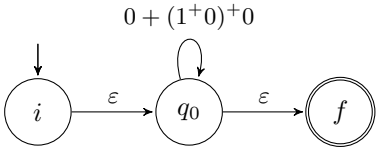
— Méthode graphique par élimination d'états, dans l'ordre q_1, puis q_2, puis q_0. On commence par modifier l'automate pour se ramener à un unique état initial sans transition entrante et un unique état acceptant sans transition sortante.



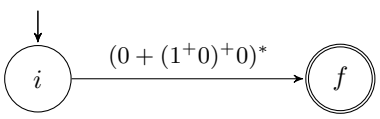
Supprimons q_1 :



Puis, supprimons q_2 :



Enfin, supprimons q_0 :

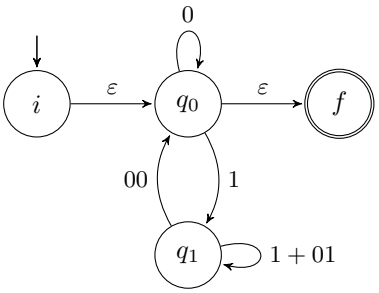


Les systèmes associés sont :

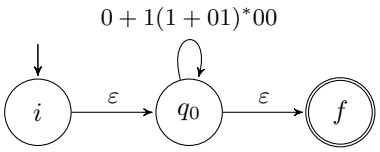
$$\begin{cases} x_i &= x_0 \\ x_0 &= 0x_0 + 1^+0x_2 + x_f \\ x_2 &= 0x_0 + 1^+0x_2 \\ x_f &= \varepsilon \end{cases} \quad \text{et} \quad \begin{cases} x_i &= x_0 \\ x_0 &= (0 + (1^+0)^+0x_2 + x_f \\ x_f &= \varepsilon \end{cases}$$

— Méthode graphique par élimination d'états, dans l'ordre q_2, puis q_1, puis q_0. Comme précéemment, on part de l'automate avec les états i et f ajoutés.

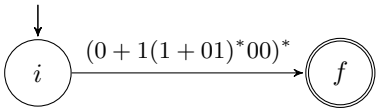
Supprimons q_2 :



Puis, supprimons q_1 :



Enfin, supprimons q_0 :



Les systèmes associés sont :

$$\begin{cases} x_i &= x_0 \\ x_0 &= 0x_0 + 1x_1 + x_f \\ x_1 &= 00x_0 + (1 + 01)x_1 \\ x_f &= \varepsilon \end{cases} \quad \text{et} \quad \begin{cases} x_i &= x_0 \\ x_0 &= (0 + 1(1 + 01)^*00)x_0 + x_f \\ x_f &= \varepsilon \end{cases}$$

On remarque que les équations de x_0, x_1 et x_2 dans les automates après suppression de certains états correspondent bien aux différentes étapes de résolution.

Exercice 41 [A savoir faire] Caractériser (par une phrase en français) les langages représentés par les expressions régulières suivantes :

1. $0^*(10^*10^*10^*)^*$
2. $(1 + 01 + 001)^*(\epsilon + 0 + 00)$
3. $1^*(0 + \epsilon)1^*$

Solution de l'Exercice 41.

1. Les mots $w \in \{0,1\}^*$ contenant un nombre de 1 multiple de 3 (c.-a-d., tels que $|w|_1 = 3k$, où $k \in \mathbb{N}$).
2. Les mots $w \in \{0,1\}^*$ qui ont au plus deux 0 consécutifs (jamais trois).
3. Les mots $w \in \{0,1\}^*$ contenant au plus un 0.

Exercice 46 On admet que $M = \{a^n b^n \mid n \geq 0\}$ n'est pas régulier. Montrer que les langages suivants ne sont pas réguliers non plus **sans se servir du lemme de l'étoile** :

1. $L_1 = \{w \in \{a,b\}^* \mid w \text{ a autant de } a \text{ que de } b\}$
2. $L_2 = \{a^i b^j c^k \mid i + j = k \geq 0\}$
3. $L_3 = \{(ab)^{2n}(cd)^{2n} \mid n \geq 0\}$

Solution de l'Exercice 46.

1. On a $M = L_1 \cap a^*b^*$, donc L_1 n'est pas régulier.
2. Soit h la fonction définie par :

$$h : \begin{cases} a &\mapsto a \\ b &\mapsto a \\ c &\mapsto b \end{cases}$$

C'est un homomorphisme, et on a $h(L_2) = M$ donc L_2 n'est pas régulier.

3. **Remarque :** On ne voit plus en détail la fermeture par homomorphisme inverse en cours.
   Soit h la fonction définie par :
$$h : \begin{cases} a & \mapsto & a \\ b & \mapsto & \varepsilon \\ c & \mapsto & b \\ d & \mapsto & \varepsilon \end{cases}$$
   On a $h(L_3) = \{a^{2n}b^{2n} \mid n \geq 0\}$; ce langage est régulier si L_3 est régulier. Donc, si L_3 est régulier, alors $h(L_3) \cup \{a\}.h(L_3).\{b\} = M$ est également régulier, une contradiction.

Exercice 47 En utilisant le lemme de l'étoile, montrer que les langages suivants ne sont pas réguliers :

1. $L_1 = \{wb^n \mid n \in \mathbb{N}, w \in \{a,b\}^n\}$
2. $L_2 = \{w \in \{a,b\}^* \mid w \text{ est un palindrome}\}$

Solution de l'Exercice 47. D'après le lemme de l'étoile, si L est un langage régulier, alors il existe un entier n tel que si $z \in L$ est de longueur au moins n, alors z est de la forme uvw, où $|uv| \leq n$, $|v| \geq 1$ et pour tout $i \geq 0$, $uv^iw \in L$.

1. Soit n l'entier donné par le lemme de l'étoile appliqué à L_1. Prenons le mot $z = a^nb^n$; alors z est de la forme uvw, et comme $1 \leq |uv| \leq n$, on a $uv \in a^+$. Donc $v \in a^+$, et on devrait avoir $uv^2w = a^{n+|v|}b^n \in L_1$, ce qui est impossible.
2. Soit n l'entier donné par le lemme de l'étoile appliqué à L_2. Soit $z = a^nba^n$. Ce mot est un palindrome de longueur au moins n, il est donc de la forme uvw, et nécessairement, $v \in a^+$. Mais on a $uv^0w = a^{n-|v|}ba^n$ qui n'est pas un palindrome, une contradiction.

Exercice 58 Grammaire ambiguë

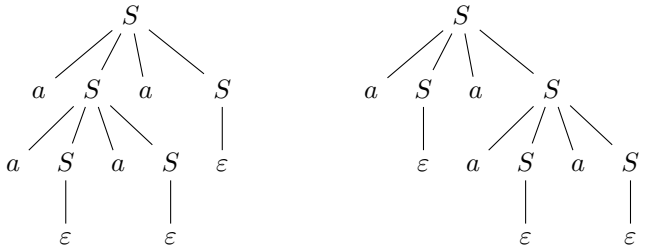
▷ Question 1 Montrer que les grammaires suivantes sont ambiguës et proposer des grammaires équivalentes non ambiguës :

1. $S \to aSaS \qquad S \to \varepsilon$
2. $S \to aSb \qquad S \to aS \qquad S \to \varepsilon$

▷ Question 2 Donner une grammaire non ambiguë pour le langage $\{a^nb^p \mid 2p \geq n \geq p\}$. Montrer que votre grammaire est non-ambiguë en appliquant les conditions suffisantes vues en cours.

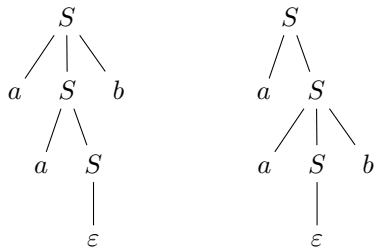
Solution de l'Exercice 58.

▷ Question 1 1. Pour $S \to aSaS \mid \varepsilon$:
Le mot $aaaa$ peut être généré par deux arbres de dérivation différents :



Le langage engendré par cette grammaire est $(aa)^*$ donc on peut proposer $S \to aaS \mid \varepsilon$ ou $S \to aSa \mid \varepsilon$. Si on veut une grammaire régulière : $S \to aX \mid \varepsilon \qquad X \to aS$.

2. Pour $S \to aSb \qquad S \to aS \qquad s \to \varepsilon$:
Le mot aab possède deux arbres de dérivation :



Pour obtenir un grammaire équivalente non-ambiguë, il faut ordonner les règles. Deux solutions :
— $S \to aS \mid A \qquad A \to aAb \mid \varepsilon$
— $S \to aSb \mid A \qquad A \to aA \mid \varepsilon$

▷ Question 2 On commence par une grammaire ambiguë : $S \to aaSb \mid aSb \mid \varepsilon$. Pour lever l'ambiguïté, il faut ordonner les règles qui posent problème, donc choisir si on commence par deux a pour un b ou un seul. Cela revient à décomposer le langage comme suit : $\{a^{2p}a^qb^qb^p \mid p,q \geq 0\}$ ou $\{a^pa^{2q}b^qb^p \mid p,q \geq 0\}$. Voici les grammaires non-ambiguës qui correspondent :

$$\begin{array}{l} S \to aaSb \mid X \\ X \to aXb \mid \varepsilon \end{array} \qquad \begin{array}{l} S \to aSb \mid X \\ X \to aaXb \mid \varepsilon \end{array}$$

Pour montrer que ces grammaires sont non-ambiguës, on utilise les deux conditons suffisantes vues en cours :
— pour tout couple de règles $(A \to \alpha, A \to \beta)$ de G tel que $\alpha \neq \beta$, $\mathcal{L}(\alpha) \cap \mathcal{L}(\beta) = \emptyset$;
Autrement dit, à un point donné d'une dérivation, on ne peut remplacer une règle par une autre et obtenir le même mot (cf. cas 2 de la question 1 de l'exercice 58).
— pour toute règle de la forme $A \to X_1X_2...X_n$, où $X_i \in V_T \cup V_N$, $\forall w \in V_T^*$ tel que $X_1X_2...X_n \Longrightarrow^* w$, $\exists!(w_1, w_2, ...w_n)$ tel que $w_i \in V_T^*$, $w = w_1w_2...w_n$ et $\forall i$, $X_i \Longrightarrow^* w_i$.
Autrement dit, lorsqu'on a appliqué une règle, les mots qui doivent être générés par chacun des non-terminaux sont uniques (cf. cas 1 de la question 1 de l'exercice 58).

Utilisons ces conditions sur la première grammaire.

Première condition : Pour les règles $X \to AXb$ et $X \to \varepsilon$, la longueur permet de les distinguer : $\mathcal{L}(\varepsilon) = \varepsilon$ qui est de longueur 0 alors tout $w \in \mathcal{L}(aSb)$ est de longueur ≥ 2.
Pour les règles $S \to aaSb$ et $S \to X$, il faut remarquer que tout mot issu de X contient autant de a que de b (on a même $\mathcal{L}(X) = \{a^nb^n \mid n \in \mathbb{N}\}$) alors que ce n'est pas le cas d'un mot issu de $aaSb$ car un mot $w \in \mathcal{L}(S)$ vérifie $|w|_a \geq |w|_b$.

Seconde condition : Comme les parties droites des règles ne contiennent qu'un unique symbole non-terminal, les preuves sont triviales. En effet, tout symbole terminal ne peut dériver qu'en lui-même donc le reste doit l'être par le symbole non-terminal.

Exercice 59 Preuves sur les grammaires

On définit deux langages L_1 et L_2 sur le vocabulaire $V = \{a, b\}$.

Soit $P_1(w)$ la propriété $|w|_a = |w|_b$ et $P_2(w)$ la propriété $\forall u \in Prefixe(w), |u|_a \geq |u|_b$. Rappelons que la notation $|w|_x$ représente le nombre d'occurrences du symbole x dans w et u est un préfixe de w si et seulement il existe un mot $v \in V^*$ tel que $uv = w$.

— **Définition de L_1 par compréhension :** $L_1 = \{w \mid P_1(w) \wedge P_2(w)\}$
— **Définition de L_2 par grammaire :** $S \to \varepsilon$, $S \to SS$ et $S \to aSb$ avec $L_2 = \mathcal{L}(S)$.

▷ Question 1 Justifier en quoi la chaîne $aabbab$ appartient à la fois à L_1 et L_2.

▷ Question 2 On veut montrer que $L_2 \subseteq L_1$. On rappelle que ceci revient à montrer que tout mot dérivable à partir de S vérifie les propriétés $P1$ et $P2$.

1. ajouter une règle de grammaire qui violerait la propriété $P2$.
2. en considérant la grammaire initiale prouvez $L_2 \subseteq L_1$.

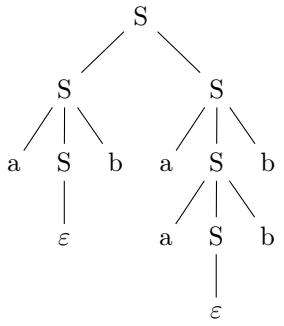
▷ Question 3 On veut maintenant montrer que $L_1 \subseteq L_2$. Pour cela on doit montrer que tout mot vérifiant les propriétés $P1$ et $P2$ peut être dérivé de l'axiome.

1. On vous propose de faire l'analyse par cas suivante : (1) $w = \varepsilon$, (2) $w = abu$, (3) $w = uab$ (4) $w = aub$. Justifier en quoi cette décomposition n'est pas complète, i.e. qu'il existe des mots de L_1 qui ne peuvent être produits comme une combinaison de ces différents cas.
2. Prouvez $L_1 \subseteq L_2$.

Solution de l'Exercice 59.

▷ Question 1 Pour L_1 :
— (P_1) $|abaabb|_a = 3 = |abaabb|_b$.
— (P_2) Ses sept préfixes ont au moins autant de a que de b. Donc $abaabb \in L_1$.
Pour L_2, deux options : écrire une dérivation ou un arbre de dérivation

$$
\begin{aligned}
S &\xrightarrow{S\to SS} SS\\
&\xrightarrow{S\to aSb} aSbS\\
&\xrightarrow{S\to\varepsilon} abS\\
&\xrightarrow{S\to aSb} abaSb\\
&\xrightarrow{S\to aSb} abaaSbb\\
&\xrightarrow{S\to\varepsilon} abaabb
\end{aligned}
$$



▷ Question 2

1. Plusieurs possibilités :
   - $S \to b$ permet de produire b qui viole à la fois P_1 et P_2 ;
   - $S \to ba$ permet de produire ba qui satisfait P_1 mais pas P_2 ;
   - $S \to a$ permet de produire a qui satisfait P_2 mais pas P_1.
2. Il faut montrer que si $S \Longrightarrow^* w$ alors w satisfait P_1 et P_2.
   De $w \in V_T^*$, on déduit $w \neq S$ donc qu'il existe $n > 0$ tel que $S \Longrightarrow^n w$, d'où $S \Longrightarrow \alpha \Longrightarrow^{n-1} w$ avec $S \to \alpha \in R$.
   On procède donc par récurrence sur n, la longueur de la dérivation.

   $n = 1$: Une possibilité : $S \Longrightarrow \varepsilon\ (= w)$
   ε satisfait P_1 et P_2 (il est son seul préfixe) donc $\varepsilon \in L_1$

   $n > 1$: Deux possibilités :
   - $S \Longrightarrow aSb \Longrightarrow^{n-1} aw_1b\ (= w)$ avec $S \Longrightarrow^{n-1} w_1$
   - $S \Longrightarrow SS \Longrightarrow^{n-1} w_1w_2\ (= w)$ avec $S \Longrightarrow^p w_1$ et $S \Longrightarrow^q w_2$ et $p + q = n - 1$

   L'hypothèse d'induction (HI) à utiliser est donc : $\forall k < n,\ S \Longrightarrow^k w \Rightarrow w \in L_1$.

   - $S \Longrightarrow aSb \Longrightarrow^{n-1} aw_1b\ (= w)$ avec $S \Longrightarrow^{n-1} w_1$
     Par HI, $w_1 \in L_1$ et donc, pour $w = aw_1b$:
     (a) $|w|_a = |w_1|_a + 1 = |w_1|_b + 1 = |w|_b$
     (b) u préfixe de w est, soit ε, soit aw_1b, soit au_1 avec u_1 préfixe de w_1. On conclut facilement en analysant ces trois cas (car $|u_1|_a \geq |u_1|_b$).
   - $S \Longrightarrow SS \Longrightarrow^{n-1} w_1w_2\ (= w)$ avec $S \Longrightarrow^p w_1$, $S \Longrightarrow^q w_2$, $p + q = n - 1$ donc $p < n$, $q < n$
     Par HI, w_1 et w_2 sont dans L_1 et donc, pour $w = w_1w_2$:
     (a) $|w|_a = |w_1|_a + |w_2|_a = |w_1|_b + |w_2|_b = |w|_b$
     (b) pour u préfixe de w :
     - si u préfixe de w_1 : OK
     - si $u = w_1u_2$ avec u_2 préfixe de w_2 : OK (car $|w_1|_a = |w_1|_b$)

▷ Question 3

1. Voici un exemple de mot de L_1 qui n'est pas couvert par la décomposition proposée : $aabbaabb$.
2. Soit $w \in L_1$. Notons $n = |w|$ et supposons que pour tout mot $x \in L_1$ de longueur $< n$, on a $x \in L_2$. Montrons que $w \in L_2$ (i.e. $S \Longrightarrow^* w$).
   Pour cela distinguons trois cas :
   (a) $w = \varepsilon$ (qui $\in L_1$) : La règle $S \to \varepsilon$ assure que $w \in L_2$.
   (b) Il existe $(x_1, x_2) \in {L_1}^2$ tel que $w = x_1x_2$, $x_1 \neq \varepsilon$ et $x_2 \neq \varepsilon$. On a $|x_1| < n$ et $|x_2| < n$ donc par HI $S \Longrightarrow^* x_1$ et $S \Longrightarrow^* x_2$. La règle $S \to SS$ finit de démontrer que $w \in L_2$: $S \Longrightarrow SS \Longrightarrow^* x_1S \Longrightarrow^* x_1x_2 = w$.
   (c) Sinon : on n'est dans aucun des deux cas précédents, on a donc forcément $\forall x_1 \in Prefixe(w) \setminus \{\varepsilon, w\}, |x_1|_a > |x_1|_b$, sans quoi un tel x_1 (avec donc $|x_1|_a = |x_1|_b$) donnerait la décomposition $w = x_1x_2$ du second cas.
   Du coup, on a forcément $w = axb$, avec $|x|_a = |x|_b$ (car $|w|_a = |w|_b$) et par ailleurs $\forall u \in Prefixe(x), au \in Prefixe(w) \setminus \{\varepsilon, w\}$, donc $|au|_a > |au|_b$ et donc $|u|_a \geq |u|_b$. Par conséquent $x \in L_1$, et donc par HI, $S \Longrightarrow^* x$.
   La règle $S \to aSb$ finit de démontrer que $w \in L_2$: $S \Longrightarrow aSb \Longrightarrow^* axb = w$.

On a ainsi prouvé que $L_1 \subseteq L_2$, ce qui donne au final $L_1 = L_2$.