# Module Langages Formels
# TD 2 : Résiduels et automates finis

**Exercice 1** `Résiduels`

**1.1.** Calculer le résiduel de $L$ par rapport à tout mot $u$ sur $\Sigma = \{a, b\}$ dans les exemples suivants :

$$L = a^*b^* \qquad\qquad L' = \{a^n b^n \mid n \geq 0\}$$

**1.2.** Si $x$ est une lettre de $\Sigma$, que valent $x^{-1}(L_1 \cup L_2)$, $x^{-1}(L_1L_2)$ et $x^{-1}L_1^*$ où $L_1$ et $L_2$ sont deux langages sur $\Sigma$ ?

**Exercice 2** `Codes et quotients`

On étend la définition des résiduels à gauche à des langages sur $\Sigma^*$ : le **quotient à gauche** d'un langage $L_1$ par un langage $L_2$ est défini par $L_2^{-1}L_1 = \bigcup_{u \in L_2} u^{-1}L_1$.
Soit $X$ un sous-ensemble de $\Sigma^+$. On définit la suite $(U_i)$ de langages

$$\left\{ \begin{array}{rcl} U_1 & = & X^{-1}X \setminus \{\varepsilon\} \\ U_{n+1} & = & X^{-1}U_n \cup U_n^{-1}X \text{ pour } n \geq 1 \end{array} \right.$$

**2.1.** Montrer que pour tout $n \geq 1$, et pour tout $k \in [1, n]$,

$$\varepsilon \in U_n \iff \exists u \in U_k, \exists i, j \geq 0 \text{ tels que } uX^i \cap X^j \neq \emptyset \text{ avec } i + j + k = n$$

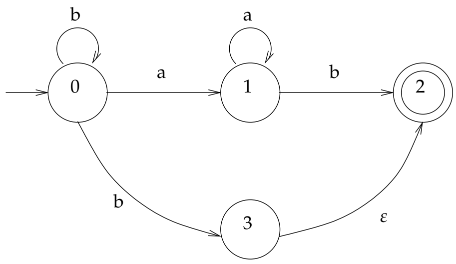
On appelle **code** sur un alphabet $\Sigma$ tout langage $X$ sur $\Sigma$ tel que pour toutes familles $(x_i) \in X^{[\![1,p]\!]}$ et $(y_i) \in X^{[\![1,q]\!]}$, $x_1x_2 \ldots x_p = y_1y_2 \ldots y_q$ entraine $p = q$ et $x_i = y_i$ pour tout $i$. Dire que $X$ est un code revient donc à dire que tout élément de $X^*$ se factorise de manière unique sur $X$.

**2.2.** En déduire que $X$ est un code si et seulement si aucun des $U_i$ ne contient $\varepsilon$.

**Exercice 3**
Écrire un automate déterministe qui reconnaît les entiers écrits en base 2 qui sont congrus à 1 modulo 3.

**Exercice 4** `La méthode de Thompson`
On décide d'ajouter aux automates non déterministes la possibilité d'utiliser des $\varepsilon$-transitions. $\varepsilon$ est une étiquette de transition qui correspond au mot vide.

Par exemple, $b \in \mathcal{L}(A)$, avec $A$ l'automate ci-dessous.




**4.1.** Proposer un algorithme de déterminisation des automates finis à $\varepsilon$-transitions et l'appliquer sur l'exemple ci-dessus.

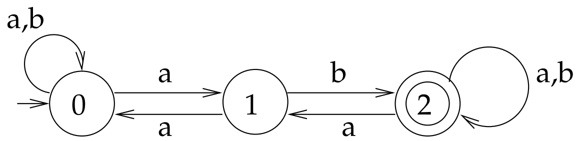
**4.2.** Montrer que tout automate fini non déterministe est équivalent à un automate fini non déterministe ayant un unique état initial et un unique état final.

**4.3.** Soient $A$ et $B$ deux automates finis. Construire des automates reconnaissant

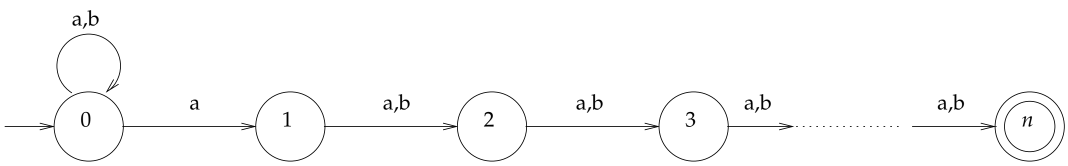
$$\mathcal{L}(A).\mathcal{L}(B) \qquad\qquad \mathcal{L}(A) \cup \mathcal{L}(B) \qquad\qquad \mathcal{L}(A)^*$$

## Exercice 5 `Déterminisation`

**5.1.** Déterminiser l'automate suivant :




**5.2.** Nous allons maintenant calculer la complexité au pire de la déterminisation d'un automate en fonction de son nombre d'états. On a vu en cours que le déterminisé d'un automate à $Q$ états a au plus $2^{|Q|}$ états, nous allons détailler un exemple. Considérons l'automate $A$ suivant, qui reconnaît l'ensemble des mots de $\{a,b\}^*$ dont la $n^e$ lettre en partant de la fin est un $a$ (on suppose $n > 0$) :




Soit $B = (Q, \Sigma = \{a,b\}, q_0, F, \delta)$ un automate fini déterministe reconnaissant le même langage que $A$.

**5.2. 1.** Montrer que $B$ est complet (i.e. si $u \in \Sigma^*$, alors $\delta(q_0, u)$ existe).

**5.2. 2.** Prouver que la fonction $\varphi : \begin{array}{l} \Sigma^{n-1} \to Q \\ u \mapsto q_u = \delta(q_0, u) \end{array}$ est injective.

En conclure que la déterminisation de $A$ est au pire exponentielle en nombre d'états.