# TD n°7

## Encore Rationnalité et non-Rationnalité – Encore de l'Automate à l'E.R.

### Caractérisation des langages rationnels sur une seule lettre

**Exercice 1** *On travaille sur l'alphabet* $\Sigma = \{a\}$.

*On appelle* suite arithmétique *tout langage de la forme* $\{a^{m+kr} | k \in \mathbb{N}\}$ *où* $m$ *et* $r$ *sont deux entiers positifs fixés.*

*En raisonnant sur la forme possible d'un automate fini sur l'alphabet* $\Sigma = \{a\}$*, démontrez que les seuls langages de* $Rat(\Sigma)$ *sont :*

- *les langages finis ;*
- *les suites arithmétiques ;*
- *ou les langages obtenus comme union de langages finis et de suites arithmétiques.*

### Démontrer la non-rationalité d'un langage en utilisant les propriété de fermeture de $Rat = Rec$.

Le lemme d'itération n'est pas le seul outil pour montrer qu'un langage n'est pas rationnel.
Pour cela on peut aussi utiliser les propriété de fermeture de la famille $Rat = Rec$ à condition de connaître déjà quelques langages non-rationnels.
Vous connaissez déjà beaucoup de ces propriétés dont certaines dérivent de la définition même de $Rat = Rec$ alors que d'autres ont été démontrées ou indiqués en cours ou en TD.

On sait notamment que $Rat = Rec$ est fermée relativement à : $\cup, \cdot, ^*, \cap, \mathcal{C}$ (complémentaire), $\setminus$ (différence d'ensembles), $\Delta$ (différence symétrique), $\sim$ (miroir), par préfixes, ...

Vous connaissez déjà également certains langages non rationnels vus dans les TD précédents dont vous pourrez vous servir, par exemple le langage $L_0 = \{a^n b^n | n \in \mathbb{N}\}$. On a déjà montré que ce langage n'est pas rationnel (par exemple par le Lemme de l'étoile).

**Exemple d'application.**
On veut montrer que $L_1 = \{w \in \{a, b\}^* || w|_a = w|_b\}$ n'est pas rationnel.

On a : $L_0 = L_1 \cap a^*b^*$.
Si $L_1$ était rationnel alors son intersection avec un autre rationnel (dans ce cas $a^*b^*$) le serait aussi car $Rat = Rec$ est fermée relativement à $\cap$. Mais on sait que cette intersection est $L_0$, qui n'est pas rationnel donc $L_1$ ne peut pas l'être.

**Note.** On remarque que pour appliquer cette technique il faut exprimer un langage dont on a déjà montré la non-rationnalité (dans l'exemple $L_0$) en fonction du langage dont on veut montrer la non-rationnalité (dans l'exemple $L_1$), de langages rationnels (dans l'exemple $a^*b^*$) et d'opérations relativement auxquelles $Rat = Rec$ est fermée (dans l'exemple $\cap$, mais des fois on a besoin d'utiliser plusieurs opérations).

**Exercice 2** *Reprenez ces langages des exercices des TD précédents et essayez de trouver une preuve du fait qu'ils ne sont pas rationnels en utilisant les propriétés de fermeture.*

1. $\{a^p : p \text{ non premier}\}$
2. $\{a^m b^n : m+n \text{ est un carré}\}$
3. $\{a^m b^n : m \neq n\}$
4. $\{u \in \{a,b,c\}^* : |u|_a = |u|_b\}$
5. $\{a^m b^n c^{m+n} : m,n \in \mathbb{N}\}$
6. $\{a^{n+2} b^n : n \in \mathbb{N}\}$
7. $\{uav : u,v \in \{a,b\}^*, |u| = |v|\}$
8. $\{u\widetilde{u} : u \in \{a,b\}^*\}$
9. $\{u^2 : u \in \{a,b\}^*\}$
10. $\{a^m b^n : m \geq n\}$
11. $\{a^m b^n : m < n\}$

**De l'automate à l'expression rationnelle - La méthode de Brzozowski et McCluskey**

La méthode de Brzozowski et McCluskey (ou méthode d'élimination des états) est un méthode alternative au Lemme d'Arden pour obtenir l'expression rationnelle à partir de l'automate.

On part d'un automate (non nécessairement déterministe).

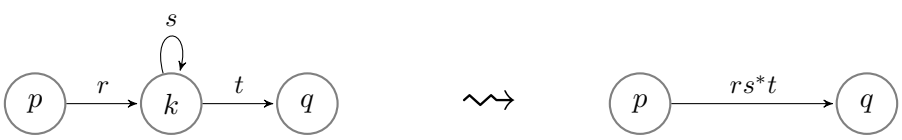
On commence par transformer $\mathcal{A}$ en ajoutant deux nouveaux états $i$ et $t$ et des transitions étiquetées par $\varepsilon$ de $i$ vers les états initiaux de $\mathcal{A}$ ainsi que des transitions étiquetées par $\varepsilon$ des états terminaux de $\mathcal{A}$ vers $t$. On obtient ainsi un automate avec un seul état initial qui n'a pas de transitions entrantes et un seul état terminal qui n'a pas de transitions sortantes, dit *normalisé*.

À partir de cet automate normalisé, on calcule une suite d'automates normalisés qui reconnaissant le même langage et ayant toujours moins de transitions ou d'états, mais dont les transitions seront étiquetés par des expressions rationnelles plutôt que par des simples lettres.

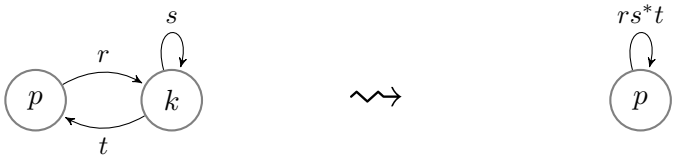
À la fin, il ne reste que les deux états $i$ et $t$ et une transition de $i$ à $t$ dont l'étiquette est une expression régulière dénotant le langage reconnu.

Les transformations sont des réductions des états et de transitions.

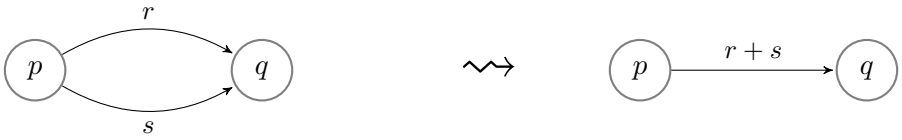
On choisit un état $k$ quelconque avec $k \notin \{i,t\}$. On détermine tous les chemins $(p,r,k)(k,t,q)$ de longueur 2 qui ont $k$ au milieu et on les remplace par la transition $(p, rs^*t, q)$ (où $s$ est l'étiquette d'une éventuelle boucle $(k,s,k)$) :



Attention à ne pas oublier les cas particuliers avec $p = q$, c'est-à-dire les boucles de longueur 2 contenant $k$ et un autre état :

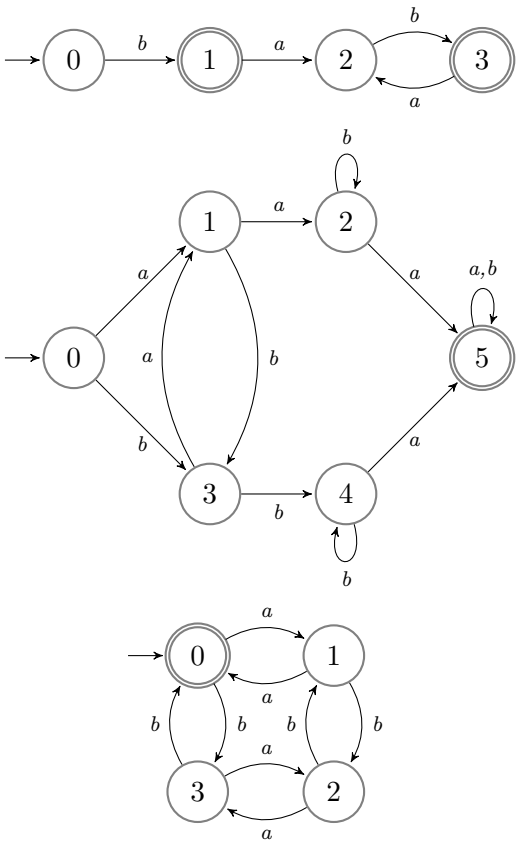


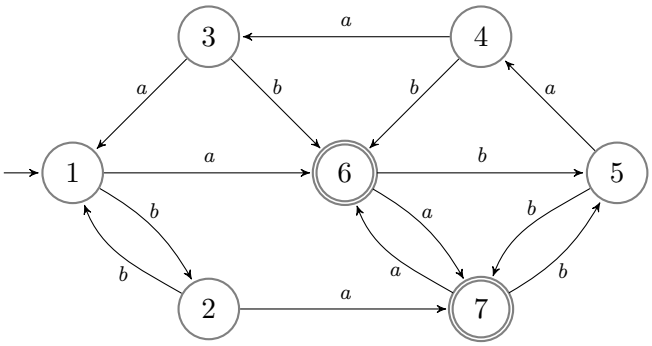

De plus, à chaque fois qu'on se retrouve avec deux transitions multiples $(p, r, q)$ et $(p, s, q)$ on les remplace par la transition $(p, r + s, q)$.




**Astuce :** dans l'exécution à la main, il convient de commencer toujours par les états ayant la plus petite valeur $d^+ \times d^-$ où $d^+$ est le nombre de transitions entrantes et $d^-$ est le nombre de transitions sortantes.

**Exercice 3** *Calculer une expression rationnelle de $\mathcal{L}(\mathcal{A})$ pour les automates ci-dessous, en appliquant la méthode de Brzozowski–McCluskey.*




3
4
1
6
5
2
7
a
a
b
b
b
a
a
b
b
a
b
b
a