Langages formels

Chapitre 01

1 Mots

Définition 1 : alphabet et mot fini

Un **alphabet** est un ensemble fini. Il est classiquement dénoté $\mathcal{A}$ ou Σ (pour « symboles »).
Un **mot** (fini) est une succession (finie) de lettres. Plus formellement : soit $n \in \mathbb{N}$. Un mot de longueur n est un élément de $\mathcal{A}^n$. Un mot fini est un élément de $\bigcup_{n \geqslant 0} \mathcal{A}^n$.

Il existe un (seul) mot de longueur 0, le mot vide : il est généralement noté ε (pour « empty »).
À noter : un mot n'a de sens qu'avec un alphabet fixé. Parfois, cet alphabet n'est donné qu'implicitement.

Sur l'alphabet $\mathcal{A} = \{0, 1\}$, on a les mots $0, 010, 0001000$ ou 0100101001.
Sur l'alphabet ASCII (26 lettres latines), on a les mots « bonjour », « abcd » ou « wdzieczny ».

Notation.

1. Pour un mot m, sa longueur est notée $|m|$.
2. Soit $a \in \mathcal{A}$. On note $|m|_a$ le nombre d'occurrences de la lettre a dans m. On a donc $|m| = \sum_{a \in \mathcal{A}} |m|_a$.
3. Lorsqu'on a un mot m, on note généralement $m = m_0 m_1 \dots m_{|m|-1}$ où les m_i dénotent les lettres composant le mot m. Par exemple, si $m =$ bonjour, alors on a $m_0 =$ b, $m_1 =$ o, $m_2 =$ n, $m_3 =$ j, $m_4 =$ o, ...

Définition 2 : ensembles de mots

Soit $\mathcal{A}$ un alphabet. Pour $n \in \mathbb{N}$, on note $\mathcal{A}^n$ l'ensemble des mots de longueur n.
On note $\mathcal{A}^*$ l'ensemble de tous les mots finis sur $\mathcal{A}$: $\mathcal{A}^* = \bigcup_{n \in \mathbb{N}} \mathcal{A}^n$.

Il existe une opération centrale lors de la manipulation de mots : la concaténation.

Définition 3 : concaténation

Soit $\mathcal{A}$ un alphabet, et soient u et v deux mots sur $\mathcal{A}$. La concaténation de u et de v, notée $u \cdot v$, est le mot de longueur $|u| + |v|$ vérifiant :

pour tout $0 \leqslant i \leqslant |u| - 1$, $(u \cdot v)_i = u_i$;

pour tout $0 \leqslant i \leqslant |v| - 1$, $(u \cdot v)_{|u|+i} = v_i$.

Soit $u =$ apple, $v =$ pen, alors $u \cdot v =$ applepen.

1. En réalité, il n'est pas nécessaire techniquement que u et v soient définis sur le même alphabet. Si u est défini sur $\mathcal{A}$ et v sur $\mathcal{B}$, alors $u \cdot v$ est défini sur $\mathcal{A} \cup \mathcal{B}$.
2. Techniquement, on peut dire qu'un mot est la concaténation de toutes ses lettres.
3. En pratique, tout comme on utilise rarement le « $\times$ », le « $\cdot$ » est généralement implicite, et on note uv.
4. Par analogie avec le produit, on parle de la puissance d'un mot : u^k désigne la concaténation de k copies du mot u. Pour tout mot u, $u^0 = \varepsilon$.

5. Pour u et v deux mots, $|u \cdot v| = |u| + |v|$, et pour toute lettre $a \in \mathcal{A}$, $|u \cdot v|_a = |u|_a + |v|_a$.

Définition 4 : préfixe, suffixe

Soit $\mathcal{A}$ un alphabet, et soient u et v deux mots finis sur $\mathcal{A}$. On dit que :

u est un **préfixe** de v s'il existe w un troisième mot tel que $v = u \cdot w$;

u est un **suffixe** de v s'il existe w un troisième mot tel que $v = w \cdot u$.

fre est un préfixe de $freluquet$ et un suffixe de $chiffre$. aba est à la fois un préfixe et un suffixe de $ababa$.

Propriété 5 : ce sont des relations d'ordre

Soit $\mathcal{A}$ un alphabet. Alors la relation de préfixe, notée $\sqsubseteq_p$, définit une relation d'ordre partielle sur $\mathcal{A}^*$.
De même, la relation de suffixe, notée $\sqsupseteq_s$, définit une relation d'ordre partiel.

Démonstration.

On prouve que la relation de préfixe est une relation d'ordre.

Réflexivité : soit u un mot, alors $u = u \cdot \varepsilon$, donc on a bien $u \sqsubseteq_p u$;

Transitivité : soient u, v et w trois mots tels que $u \sqsubseteq_p v$ et $v \sqsubseteq_p w$: alors il existe x et y deux mots tels que $v = u \cdot x$ et $w = v \cdot y$. Alors :

$$\begin{aligned} w &= v \cdot y \\ &= (u \cdot x) \cdot y = u \cdot (x \cdot y) \end{aligned}$$

Donc on a bien $u \sqsubseteq_p w$.

Antisymétrie : soient u et v deux mots tels que $u \sqsubseteq_p v$ et $v \sqsubseteq_p u$. Alors il existe w et $\tilde{w}$ tels que $v = u \cdot w$ et $u = v \cdot \tilde{w}$. Mais alors :

$$\begin{aligned} u &= v \cdot \tilde{w} \\ &= (u \cdot w) \cdot \tilde{w} \\ &= u \cdot (w \cdot \tilde{w}) \end{aligned}$$

Donc $|u| = |u \cdot (w \cdot \tilde{w})| = |u| + |w| + |\tilde{w}|$: donc en particulier, $|w| = |\tilde{w}| = 0$. Autrement écrit, $w = \tilde{w} = \varepsilon$. Donc $v = u \cdot w = u \cdot \varepsilon = u$.

Partialité : Techniquement, si $\text{Card}(\mathcal{A}) = 1$, la relation de préfixe est totale. Dans le cas (plus classique) où $\text{Card}(\mathcal{A}) \geqslant 2$, soient a et b deux lettres de $\mathcal{A}$: alors $a \not\sqsubseteq_p b$ et $b \not\sqsubseteq_p a$. Pire encore : on peut construire une famille infinie de mots $(m_n)_{n\in\mathbb{N}}$ tels que tous les mots sont deux à deux incomparables. Il suffit de poser $m_n = a^n b$ pour tout $n \in \mathbb{N}$.

L'ordre préfixe est un ordre qu'on dit *bien fondé* : il n'existe pas de famille infinie strictement décroissante.
Toutefois, la famille $(m_n)_{n\in\mathbb{N}}$ exhibée est une famille infinie d'éléments incomparables : on appelle une telle famille une *antichaîne*, et l'existence de cette chaîne fait que l'ordre préfixe n'est pas un *bel ordre*.
Par ailleurs, si $u \sqsubseteq_p v$, alors $|u| \leqslant |v|$.

Définition 6 : facteur et sous-mot

Soit $\mathcal{A}$ un alphabet, et u et v deux mots. On dit que u est un facteur de v s'il existe x et y deux autres mots tels que $v = x \cdot u \cdot y$.
On dit que u est un sous-mot de v s'il existe $\phi : [\![0, |u| - 1]\!] \to [\![0, |v| - 1]\!]$ strictement croissante telle que pour tout $k \in [\![0, |u| - 1]\!]$, $v_{\phi(k)} = u_k$.

$stuc$ est un facteur de $lu\underline{stuc}ru$; uuu est un sous-mot de $l\underline{u}st\underline{u}cr\underline{u}$.

Propriété 7 : ce sont encore des relations d'ordre

Soit $\mathcal{A}$ un alphabet. Alors la relation de facteur, notée $\preceq_f$, est une relation d'ordre partielle sur $\mathcal{A}^*$.
De même, la relation de sous-mot, notée $\preceq_{sm}$, définit une relation d'ordre sur $\mathcal{A}^*$.

Démonstration.

La réflexivité, la transitivité et l'antisymétrie tombent avec des arguments similaires au cas de la relation préfixe. Pour la partialité : encore une fois, si $\text{Card}(\mathcal{A}) = 1$, la relation est totale. Si $\text{Card}(\mathcal{A}) \geqslant 2$, avec a et $b \in \mathcal{A}$, $a \not\preceq_f b$ et $b \not\preceq_f a$; et pire encore, la famille $(m_n)_{n\in\mathbb{N}}$ avec $m_n = ab^n a$ est une antichaîne.

De manière générale, il existe une kyrielle de résultats sur les mots. La difficulté est qu'en combinatoire des mots, beaucoup de résultats sont intuitifs, mais pénibles à prouver.

Lemme 8 : lemme de Levi

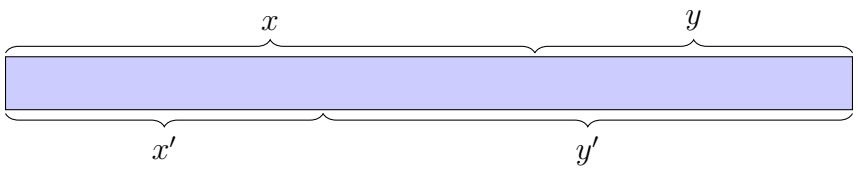
Soit $\mathcal{A}$ un alphabet, et soient $(x, y, x', y') \in (\mathcal{A}^*)^4$. On suppose que $xy = x'y'$. Alors il existe $z \in \mathcal{A}^*$ tel que l'un des deux cas a lieu :

ou bien $x = x'z$ et $y' = zy$ (si $|x| \geqslant |x'|$) ;
ou bien $x' = xz$ et $y = zy'$ (si $|x| \leqslant |x'|$).

Démonstration.

On étudie le cas où $|x| \geqslant |x'|$, l'autre étant symétrique.
Notons $M = xy = x'y'$ le grand mot, et $M = M_0M_1M_2 \dots M_{|M|-1}$. Alors par définition de la concaténation, pour tout $0 \leqslant i \leqslant |x| - 1$, $M_i = x_i$. De même, pour tout $0 \leqslant i \leqslant |x'| - 1$, $M_i = x'_i$. Or, on sait que $|x| \geqslant |x'|$: donc, pour tout $0 \leqslant i \leqslant |x'| - 1$, $x'_i = M_i = x_i$. Notons $z = x_{|x'|}x_{|x'|+1} \dots x_{|x|-1}$. Alors, par définition de la concaténation, $x = x' \cdot z$.
On sait alors que $M = xy = x'zy = x'y'$. Par définition de la concaténation (césure après x'), on en déduit donc que $|zy| = |y'|$ et que pour tout $0 \leqslant i \leqslant |zy|$, $(zy)_i = M_i = y'_i$; autrement écrit, $zy = y'$.

Ce lemme peut être compris par un simple dessin :



Une notion classique autour des mots finis est celle de mot périodique.

Définition 9 : mot périodique

Soit $m \in \mathcal{A}^*$. Pour $p \in \mathbb{N}$, on dit que m est p-périodique si pour tout $j \leqslant |m| - p - 1$, $m_j = m_{j+p}$.

coucou est 2-périodique ; *bamba* est 3-périodique.

Il est classique d'étendre la notion de périodicité au cas où $k \in \mathbb{Q}$. Par exemple, si on considère le mot $bamba$, on observe que bam se répète *presque* : on peut dire alors que $bamba$ est $5/3$-périodique. C'est dans ce contexte qu'existe le théorème de Dejean-Rao.

L'ensemble des périodes d'un mot bénéficie d'une certaine structure. Dans le cas des mots infinis, il s'agit simplement d'un idéal de $\mathbb{N}$. Dans le cas des mots finis, c'est légèrement plus compliqué.

Théorème 10 : théorème de Fine-Wilf

Soit m un mot, et soient p et q deux périodes de m. Si $|m| \geqslant p + q - \text{PGCD}(p, q)$, alors $\text{PGCD}(p, q)$ est aussi une période de m.

Démonstration.

Preuve récupérée chez Michel Rigo.
On commence par un lemme intermédiaire :

Lemme 11

Soit m un mot, et p et q deux périodes de m. On suppose que $m = p+q-1$ et que p et q sont premiers entre eux. Alors m est constant.

Démonstration.

Soit $f : [\![0, p + q - 1]\!] \to [\![0, p + q - 1]\!]$ telle que :

$$f(x) = \begin{cases} x + p & \text{si } x < q \\ x - q & \text{sinon} \end{cases}$$

Alors f est une permutation de $[\![0, p+q-1]\!]$. Soit $j > 0$ tel que $f^j(0) = 0$; alors si on a itéré a fois le premier cas et b fois le second, on a $ap-bq = 0$ et $a+b = j$. Donc $q \mid a, p \mid b$ donc $j \geqslant p+q$. Par propriétés de l'ordre d'une permutation, on en déduit que $j = p + q$; donc f est une permutation composée d'un seul cycle de longueur et d'ordre $p + q$.
On observe maintenant que $m_0 = m_{f(0)} = m_{f^2(0)}$ par p et q-périodicités ; donc m est constant.

Reprenons notre lemme et ses hypothèses. Montrons que s'il est vrai lorsque $|m| = p+q-\text{PGCD}(p, q)$, alors il est vrai pour les mots plus longs. En effet, on saurait alors que le préfixe de m de longueur $p+q-\text{PGCD}(p, q)$ admet $d = \text{PGCD}(p, q)$ comme période ; donc, plus petit encore, le préfixe de m de longueur p admet d comme période. Comme $d \mid p$, ce préfixe s'écrit sous la forme $x^{p/d}$. Mais m est p-périodique : donc m s'écrit sous la forme $(x^{p/d})^{\text{truc}}$; donc finalement, m est d-périodique.
Montrons donc que le théorème est vrai lorsque $|m| = p + q - \text{PGCD}(p, q)$. Considérons les mots $m^{(0)}, m^{(1)}, \ldots, m^{(d-1)}$ où $m^{(i)} = m_i m_{i+d} m_{i+2d} \ldots m_{i+d(p/d+q/d-2)}$; ce sont les mots obtenus en considérant les lettres une fois sur d. Alors chacun de ces mots est de longueur $p/d + q/d - 1$, et sont périodiques de périodes p/d et q/d ; comme $d = \text{PGCD}(p, q)$, p/d et q/d sont premiers entre eux. Donc d'après le lemme, chacun de ces mots est constant.
Donc par construction des $m^{(i)}$, m est d-périodique !

2 Langages

Ce chapitre traite de questions de langages formels (qui ne sont pas des langages de programmation !) :

Définition 12 : langage

Soit $\mathcal{A}$ un alphabet : un langage est un sous-ensemble de $\mathcal{A}^*$.

L'ensemble des langages sur $\mathcal{A}$ est donc $\mathcal{P}(\mathcal{A}^*)$.

Soit $\mathcal{A} = \{0, 1\}$.

- Alors $\mathcal{L}_0 = \{0^k 1 \mid k \in \mathbb{N}\}$ est un langage.
- De même, $\mathcal{L}_1 = \{1^p \mid p \text{ est un nombre premier }\}$ est un langage.
- Remarquons que $\mathcal{L}_2 = \varnothing$ est un langage, appelé le langage vide.
- Aussi, $\mathcal{L}_3 = \{101001\}$ est aussi un langage (qui ne contient qu'un seul mot).
- Par ailleurs, $\mathcal{L}_4 = \{\text{codages binaires des futurs numéros du loto }\}$ est aussi un langage.

De manière générale, on observe qu'il y a des langages plus ou moins complexes à décrire algorithmiquement. Généralement, on analyse cette complexité à l'aide de la hiérarchie dûe à Chomsky, du plus simple au plus complexe : langages réguliers, algébriques, contextuels et récursivement énumérables. Dans ce chapitre, nous nous concentrerons sur l'échelon 1 : les langages réguliers. Ils sont censés représenter le modèle de calcul le plus simple qui existe en informatique.

Définition 13 : opérations rationnelles sur les langages

Soit $\mathcal{A}$ un alphabet, et L et L' deux langages sur $\mathcal{A}$. On définit les langages suivants :

$L + L' = L \cup L' = \{m \mid m \in L \text{ ou } m \in L'\}$;

$L \cdot L' = \{u \cdot v \mid u \in L, v \in L'\}$;

pour $n \in \mathbb{N}$, $L^n = \{m_1 \cdot m_2 \cdot \cdots \cdot m_n \mid \forall 1 \leqslant i \leqslant n, m_i \in L\}$;

$L^* = \bigcup_{n \geqslant 0} L^n$.

Le dernier opérateur est appelé « étoile de Kleene ».

Pour n'importe quel langage L, $L^0 = \{\varepsilon\}$.

Propriété 14 : quelques propriétés des opérations rationnelles

L'addition est commutative, pas la concaténation ;

L'addition et la concaténation sont toutes les deux associatives ;

La concaténation est distributive par rapport à l'addition ;

L'étoile de Kleene est involutive.

Une approche algébrique de la théorie des langages formels souligne que $(\mathcal{A}^*, \cdot)$ est un monoïde, appelée **monoïde libre** ; $(\mathcal{P}(\mathcal{A}^*), \cdot)$ est aussi un monoïde.

Définition 15 : langages rationnels

L'ensemble des langages rationnels est le plus petit ensemble de $\mathcal{P}(\mathcal{A}^*)$ contenant l'ensemble vide et les singletons $\{a\}$ pour $a \in \mathcal{A}$, qui soit stable par union, concaténation et étoile de Kleene. On note cet ensemble $\mathrm{Rat}(\mathcal{A}^*)$.

On se restreint à l'alphabet $\mathcal{A} = \{a, b\}$.

— $\{aabb\}$ est un langage rationnel : en effet, $\{aabb\} = \{a\} \cdot \{a\} \cdot \{b\} \cdot \{b\}$.
— De manière générale, pour tout mot fini non vide $m \in \mathcal{A}^+$, $\{m\} \in \mathrm{Rat}(\mathcal{A}^*)$.
— Plus subtil : $\{\varepsilon\} = \varnothing^*$, donc pour tout mot fini $m \in \mathcal{A}^*$, même vide, $\{m\} \in \mathrm{Rat}(\mathcal{A}^*)$.
— $\mathcal{A}$ est un langage rationnel : $\mathcal{A} = \underset{a\in\mathcal{A}}{+}\{a\}$.
— On en déduit que $\mathcal{A}^*$ est un langage rationnel.
— $\{m \in \mathcal{A}^* \mid m$ est une succession de $a\}$ est un langage rationnel : il s'agit de $\{a\}^*$.
— $\{m \mid$ il n'y a pas deux a d'affilée dans $m\}$ est un langage rationnel.

Définition 16 : expressions régulières

Soit $\mathcal{A}$ un alphabet. Les expressions régulières sont définies de la manière suivante :

$\varnothing$ et ε sont des expressions régulières ;

pour tout $a \in \mathcal{A}$, a est une expression régulière ;

si E et E' sont deux expressions régulières, alors :

$E \mid E'$ est une expression régulière ;
$E \cdot E'$ est une expression régulière ;
(E) est une expression régulière ;
E^* est une expression régulière ;
E^+ est une expression régulière ;
$E?$ est une expression régulière ;
(à rajouter plus tard !!!) $\overline{E}$ est une expression régulière.

Définition 17 : langage régulier

Soit $\mathcal{A}$ un alphabet, et E une expression régulière. Le langage de l'expression E, noté $\mathcal{L}(E)$, est défini par induction structurelle sur E :

$\mathcal{L}(\varnothing) = \varnothing$, $\mathcal{L}(\varepsilon) = \{\varepsilon\}$;
$\mathcal{L}(a) = \{a\}$ pour $a \in \mathcal{A}$;
pour deux expressions régulières E et E' :

$\mathcal{L}(E|E') = \mathcal{L}(E) + \mathcal{L}(E')$;
$\mathcal{L}(E \cdot E') = \mathcal{L}(E) \cdot \mathcal{L}(E')$;
$\mathcal{L}((E)) = \mathcal{L}(E)$;
$\mathcal{L}(E^*) = (\mathcal{L}(E))^*$;
$\mathcal{L}(E^+) = \mathcal{L}(E) \cdot (\mathcal{L}(E))^*$;
$\mathcal{L}(E?) = \{\varepsilon\} + \mathcal{L}(E)$;
$\mathcal{L}(\overline{E}) = \mathcal{A}^* \backslash \mathcal{L}(E)$.

Un langage L est régulier s'il existe une expression régulière E telle que $L = \mathcal{L}(E)$. On note $\mathrm{Reg}(\mathcal{A}^*)$ l'ensemble des langages réguliers sur $\mathcal{A}^*$.

Propriété 18 : TINTINTIN

Soit $\mathcal{A}$ un alphabet : $\mathrm{Rat}(\mathcal{A}^*) = \mathrm{Reg}(\mathcal{A}^*)$.

Démonstration.

$\Rightarrow$: D'abord, $\varnothing \in \mathrm{Reg}(\mathcal{A}^*)$, et pour tout $a \in \mathcal{A}$, $\{a\} \in \mathrm{Reg}(\mathcal{A}^*)$.
Montrons ensuite que les langages réguliers sont stables par les opérations rationnelles. Soient E et E' deux expressions régulières définissant deux langages réguliers $\mathcal{L}(E)$ et $\mathcal{L}(E')$. Alors :

$\mathcal{L}(E) + \mathcal{L}(E') = \mathcal{L}(E|E')$, donc $\mathcal{L}(E) + \mathcal{L}(E')$ est aussi un langage régulier ;

$\mathcal{L}(E) \cdot \mathcal{L}(E') = \mathcal{L}(E \cdot E')$, donc $\mathcal{L}(E) \cdot \mathcal{L}(E')$ est aussi un langage régulier ;

$(\mathcal{L}(E))^* = \mathcal{L}(E^*)$, donc $(\mathcal{L}(E))^*$ est aussi un langage régulier.

Donc $\mathrm{Reg}(\mathcal{A}^*)$ contient $\varnothing$, contient les singletons $\{a\}$ pour $a \in \mathcal{A}$ et est stable par les opérations rationnelles ; or, $\mathrm{Rat}(\mathcal{A}^*)$ est le plus petit ensemble de langages vérifiant ces propriétés. Donc $\mathrm{Rat}(\mathcal{A}^*) \subseteq \mathrm{Reg}(\mathcal{A}^*)$.

$\Leftarrow$: Montrons par induction structurelle que le langage d'une expression régulière est un langage rationnel.

Initialisation :

$\mathcal{L}(\varnothing) = \varnothing$ est rationnel ;

$\mathcal{L}(\varepsilon) = \{\varepsilon\}$ est rationnel ;

pour tout $a \in \mathcal{A}$, $\mathcal{L}(a) = \{a\}$ est rationnel ;

Hérédité : Soient E et E' deux expressions régulières telles que $\mathcal{L}(E)$ et $\mathcal{L}(E')$ sont des langages rationnels. Alors :

$\mathcal{L}(E \mid E') = \mathcal{L}(E) + \mathcal{L}(E')$ est rationnel ;

$\mathcal{L}(E \cdot E') = \mathcal{L}(E) \cdot \mathcal{L}(E')$ est rationnel ;

$\mathcal{L}((E)) = \mathcal{L}(E)$ est rationnel ;

$\mathcal{L}(E^*) = (\mathcal{L}(E))^*$ est rationnel ;

$\mathcal{L}(E^+) = \mathcal{L}(E) \cdot (\mathcal{L}(E))^*$ est rationnel ;

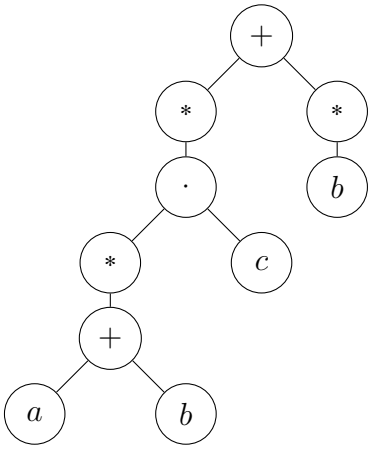
$\mathcal{L}(E?) = \{\varepsilon\} + \mathcal{L}(E)$ est rationnel.

La distinction entre langages réguliers et expressions régulières est une nuance entre sémantique et syntaxe. Les expressions régulières servent à désigner des langages réguliers : mais un seul langage régulier peut être représenté par plusieurs expressions régulières différentes.

Par exemple, considérons $L_{ex} = \{m \mid \text{il n'y a pas deux } a \text{ d'affilée dans } m\}$. Alors on peut écrire $L_{ex} = (b \mid ab)^* \cdot (a \mid \varepsilon)$ ou $L_{ex} = (a \mid \varepsilon) \cdot (b \mid ba)^*$. Dans les deux cas, L_{ex} désigne toujours le même langage : ce langage est désigné par deux expressions régulières distinctes. On dit que ces deux expressions régulières sont équivalentes.

Langages réguliers et induction

Une idée centrale dans la manipulation des langages réguliers est leur structure inductive. Une interprétation de cette structure est sous forme d'arbre. Par exemple, on a ici une représentation du langage $((a + b)^* \cdot c)^* + b^*$.



L'abre associé n'est pas nécessairement binaire, mais il est d'arité au plus 2. On appelle taille d'une expression régulière le nombre de nœuds dans l'arbre qui la représente.

À chaque expression régulière est associé un unique arbre ; mais, pour rappel, plusieurs expressions régulières peuvent représenter le même langage.

Il est possible d'interpréter le problème de l'acceptation à l'aide de cette représentation : si m est un mot et E une expression régulière, est-ce que $m \in L(E)$? L'idée consiste alors à descendre le long de l'arbre.

Définition 19 : taille d'une expression régulière

Soit E une expression régulière. On définit $|E|$ inductivement :

si $E = \varnothing$ ou $E = a$ pour $a \in \mathcal{A}$, $|E| = 1$;

si $E = E'|E''$, $|E| = |E'| + |E''| + 1$;

si $E = E' \cdot E''$, $|E| = |E'| + |E''| + 1$;

si $E = (E')^*$, $|E| = |E'| + 1$;

si $E = (E')$, $|E| = |E'|$.

3 Automates finis : théorie générale

3.1 Définitions

Une autre approche de la théorie des langages formels est une approche graphaire.

Définition 20 : automate fini

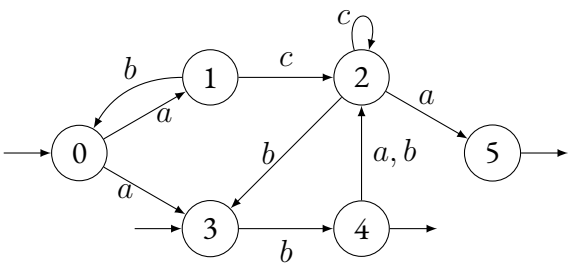
Soit $\mathcal{A}$ un alphabet. Un automate fini $\mathfrak{A}$ est un quintuplet $(\mathcal{A}, Q, I, F, T)$ où :

Q est un ensemble fini, appelé l'ensemble des **états**;

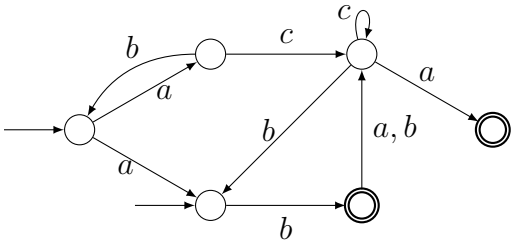
$I \subseteq Q$ et $F \subseteq Q$ sont respectivement l'ensemble des **états initiaux** et des **états finaux**;

$T \subseteq Q \times \mathcal{A} \times Q$ est l'ensemble des **transitions**.

Les automates sont, de manière classique, représentés par des graphes étiquetés, comme dans l'exemple ci-contre. Dans ce dessin, les états sont les sommets du graphe; les transitions en sont les arêtes étiquetées. Les états initiaux sont ceux avec des arêtes entrantes « vides » (ici, les états 0 et 3); les états finaux sont ceux avec des arêtes sortantes vides (ici, les états 4 et 5).



Il existe différentes conventions pour dessiner des automates. D'abord, il est usuel d'omettre le nom des états dans la représentation graphaire d'un automate. Les états finaux peuvent aussi être désignés par un double cerclage. Enfin, une transition (p, a, q) est aussi notée $p \xrightarrow{a} q$.



Définition 21 : acceptation par un automate fini

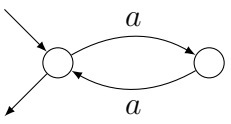
Soit $\mathcal{A}$ un alphabet et $\mathfrak{A} = (\mathcal{A}, Q, I, F, T)$ un automate fini. Soit $m \in \mathcal{A}^*$. Une marche sur $\mathfrak{A}$ étiquetée par m est un tuple $(q_k)_{0 \leqslant k \leqslant |m|} \in Q^{m+1}$ tel que pour tout $k \in [\![0, |m| - 1]\!]$, $q_k \xrightarrow{m_k} q_{k+1} \in T$. On note cette marche $q_0 \xrightarrow{m_0} q_1 \xrightarrow{m_1} q_2 \xrightarrow{m2} \dots \xrightarrow{m_{|m|-1}} q_{|m|}$; $p \xrightarrow{m}{}^* q$ désigne une marche étiquetée par m issue de p et terminant en q.
S'il existe $i \in I$ et $f \in F$ tels qu'il existe une marche $i \xrightarrow{m}{}^* f$, on dit que m est accepté par $\mathfrak{A}$, et une marche $p \xrightarrow{m}{}^* q$ est appelée un calcul acceptant m.

D'après cette définition, le mot vide est accepté ssi il existe un état à la fois final et initial.

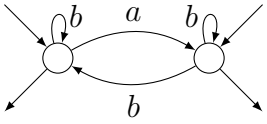
Définition 22 : langage d'un automate fini et langage reconnaissable

Soit $\mathcal{A}$ un alphabet, et $\mathfrak{A}$ un automate fini. Le langage de $\mathfrak{A}$ est $\mathcal{L}(\mathfrak{A}) = \{m \in \mathcal{A}^* \mid m \text{ est accepté par } \mathfrak{A}\}$. On dit que $\mathcal{L}(\mathfrak{A})$ est reconnu par $\mathfrak{A}$, et que $\mathcal{L}(\mathfrak{A})$ est reconnaissable. On note $\mathrm{Rec}(\mathcal{A}^*)$ l'ensemble des langages reconnaissables sur $\mathcal{A}$.

1. $\mathcal{L}(\mathfrak{A}_0) = \{a^{2k} \mid k \in \mathbb{N}\}$;
2. $\mathcal{L}(\mathfrak{A}_1) = \{m \in \{a, b\}^* \mid m \text{ n'a pas deux } a \text{ d'affilée}\}$.

$\mathfrak{A}_0 =$ 

On a $\mathcal{L}(\mathfrak{A}_0) = \mathcal{L}((aa)^*)$ et $\mathcal{L}(\mathfrak{A}_1) = \mathcal{L}((b+ab)^*(a+\varepsilon))$.

$\mathfrak{A}_1 =$ 

3.2 Théorème de Kleene

On a montré précédemment que les langages rationnels et réguliers ne faisaient qu'un. Cette preuve était plutôt rapide et naturelle : il se trouve que c'est aussi le cas des langages reconnaissables.

Théorème 23 : théorème de Kleene

Soit $\mathcal{A}$ un alphabet, alors $\mathrm{Rat}(\mathcal{A}^*) = \mathrm{Rec}(\mathcal{A}^*)$.

3.2.1 Automates finis avec ε-transitions

Pour prouver ce résultat, nous allons explorer des constructions classiques sur les automates, et en particulier une extension classique du modèle des automates finis : celui des automates finis avec ϵ-transitions.

Définition 24 : automate fini avec ε-transitions

Soit $\mathcal{A}$ un alphabet. Un automate fini $\mathfrak{A}$ avec ε-transitions est un quintuplet $(\mathcal{A}, Q, I, F, T)$ où :

Q est l'ensemble des états ;

$I \subseteq Q$ et $F \subseteq Q$ sont respectivement l'ensemble des états initiaux et des états finaux ;

$T \subseteq Q \times (\mathcal{A} \times \varepsilon) \times Q$ est l'ensemble des transitions.

On dit que m est accepté par $\mathfrak{A}$ s'il existe $(q_k)_{0 \leqslant k \leqslant M} \in Q^{M+1}$ tel que :

$q_0 \in I$ et $q_M \in F$;

il existe une fonction strictement croissante $\phi : [\![0, |m| - 1]\!] \subseteq [\![0, M-1]\!]$ telle que :

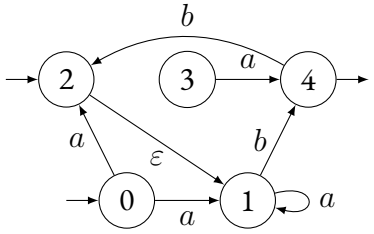
pour tout $k \in [\![0, |m| - 1]\!]$, $q_{\phi(k)} \xrightarrow{m_k} q_{\phi(k)+1} \in T$;

pour tout $j \in [\![0, M-1]\!] \backslash \phi([\![0, |m| - 1]\!])$, $q_j \xrightarrow{\varepsilon} q_{j+1} \in T$.

Le langage d'un automate fini avec ε-transitions est l'ensemble des mots acceptés.

En pratique, le modèle des automates finis avec ε-transitions est moins compliqué qu'il n'y paraît.

Dans l'automate ci-contre, la transition au milieu est une ε-transition. Cela signifie que lors d'un calcul, on peut passer de l'état 2 à l'état 1 sans lire de lettre : par conséquent, $b \in \mathcal{L}(\mathfrak{A})$.



Propriété 25 : les ε-transitions n'apportent rien de plus au modèle

Soit $\mathfrak{A}$ un automate fini avec ε-transitions ; alors il existe un automate $\mathfrak{B}$, sans ε-transitions, tel que $\mathcal{L}(\mathfrak{A}) = \mathcal{L}(\mathfrak{B})$.

Démonstration.

Soit $\mathfrak{A} = (\mathcal{A}, Q, I, F, T)$. On partitionne T en $T = T' \cup T_\varepsilon$, tel T_ε contient exactement les ε-transitions de $\mathfrak{A}$. On pose alors $\mathfrak{B} = (\mathcal{A}, Q, I, F \cup F', T' \cup T'')$ avec $T'' = \{(p, a, q) | \exists t \in Q,$ il existe $p \xrightarrow{\varepsilon}{}^* t$ et $t \xrightarrow{a} q \in T'\}$ et $F' = \{q \in Q \mid \exists f \in F,$ il existe $q \xrightarrow{\varepsilon}{}^* f\}$. Montrons que $\mathcal{L}(\mathfrak{A}) = \mathcal{L}(\mathfrak{B})$.

$\subseteq$: soit $m \in \mathcal{L}(\mathfrak{A})$. Soit $(q_k)_{0 \leqslant k \leqslant M} \in Q^{M+1}$ correspondant à un calcul acceptant m dans $\mathfrak{A}$, et soit $\phi : [\![0, |m| - 1]\!] \subseteq [\![0, M-1]\!]$ la fonction comme dans la définition.

Remarquons alors que $q_{\phi(|m|-1)+1} \in F \cup F'$: si ce n'est pas dans F, alors les transitions suivantes seront nécessairement des ε-transitions, et donc cet état appartient à F'.

Puis découpons le calcul dans $\mathfrak{A}$ en $q_{\phi(i-1)+1} \xrightarrow{\varepsilon}{}^* q_{\phi(i)} \xrightarrow{m_i} q_{\phi(i)+1}$ pour $i \in [\![0, |m| - 1]\!]$, avec la convention $\phi(-1) = -1$. Il s'agit bien d'un découpage du calcul de q_0 à $q_{\phi(|m|-1)+1}$. Il y a alors deux possibilités :

$\phi(i-1)+1 = \phi(i)$ (ie on ne passe pas par des ε-transitions), et alors $q_{\phi(i-1)+1} \xrightarrow{m_i} q_{\phi(i)+1} \in T'$;

$\phi(i-1)+1 < \phi(i)$ (ie on passe par des ε-transitions), et alors $q_{\phi(i-1)+1} \xrightarrow{m_i} q_{\phi(i)+1} \in T''$.

Donc $q_0 = q_{\phi(-1)+1} \xrightarrow{m_0} q_{\phi(0)+1} \xrightarrow{m_1} q_{\phi(2)+1} \xrightarrow{m_2} \dots \xrightarrow{m_{|m|-1}} q_{\phi(|m|-1)+1} \in F \cup F'$: donc $m \in \mathcal{L}(\mathfrak{B})$.

$\supseteq$: soit $m \in \mathcal{L}(\mathfrak{B})$, et soit $q_0 \xrightarrow{m_0} q_1 \xrightarrow{m_1} \dots \xrightarrow{m_{|m|-1}} q_{|m|}$ un calcul acceptant m dans $\mathfrak{B}$. Alors :

$q_0 \in I$;

si $q_{|m|} \in F'$, alors il existe $f \in F$ tel que $q_{|m|} \xrightarrow{\varepsilon}{}^* f$, et sinon $q_{|m|} \in F$;

une transition $q_i \xrightarrow{m_i} q_{i+1}$ dans $\mathfrak{B}$ peut se réécrire $q_i \xrightarrow{\varepsilon}{}^* t_i \xrightarrow{m_i} q_{i+1}$ pour un certain $t_i \in Q$.

En conclusion, on obtient un calcul $q_0 \xrightarrow{\varepsilon}{}^* t_0 \xrightarrow{m_0} q_1 \xrightarrow{\varepsilon}{}^* t_1 \xrightarrow{m_1} q_2 \xrightarrow{\varepsilon}{}^* \dots \xrightarrow{m_{|m|-1}} q_{|m|} \xrightarrow{\varepsilon}{}^* f$ de m acceptant dans $\mathfrak{A}$: donc $m \in \mathcal{L}(\mathfrak{A})$.

La construction donnée est appelée *fermeture arrière* des ε-transitions ; il existe une version symétrique appelée *fermeture avant*.

3.2.2 Inclusion directe

Avec la notion d'automate fini avec ε-transitions, il devient « simple » de montrer que les langages reconnaissables sont stables par les opérations rationnelles :

Propriété 26 : stabilité par addition

Soit $L, L' \in \mathrm{Rec}(\mathcal{A}^*)$. Alors $L + L' \in \mathrm{Rec}(\mathcal{A}^*)$.

Démonstration.

Soit $\mathfrak{A}_0 = (\mathcal{A}, Q_0, I_0, F_0, T_0)$ et $\mathfrak{A}_1 = (\mathcal{A}, Q_1, I_1, F_1, T_1)$ deux automates finis. Quitte à renommer les états et les transitions, on suppose que $Q_0 \cap Q_1 = \varnothing$ et $T_0 \cap T_1 = \varnothing$.
Posons alors $\mathfrak{A}_2 = (\mathcal{A}, Q_0 \cup Q_1, I_0 \cup I_1, F_0 \cup F_1, T_0 \cup T_1)$, et montrons que $\mathcal{L}(\mathfrak{A}_0) + \mathcal{L}(\mathfrak{A}_1) = \mathcal{L}(\mathfrak{A}_2)$.
$\subseteq$: soit $m \in \mathcal{L}(\mathfrak{A}_0)$. Alors il existe $q_0^{(0)} \xrightarrow{m_0} q_1^{(0)} \xrightarrow{m_1} q_2^{(1)} \xrightarrow{m_2} \dots \xrightarrow{m_{|m|-1}} q_{|m|}^{(0)}$ un calcul acceptant m dans $\mathfrak{A}_0$. Alors par définition de $\mathfrak{A}_2$, les états de ce calcul sont des états de $\mathfrak{A}_2$, les transitions de ce calcul sont des transitions de $\mathfrak{A}_2$, $q_0^{(0)}$ est un état initial de $\mathfrak{A}_2$ et $q_{|m|}^{(0)}$ est un état final de $\mathfrak{A}_2$. Donc $m \in \mathcal{L}(\mathfrak{A}_2)$. Symétriquement, pour tout $m \in \mathcal{L}(\mathfrak{A}_1)$, on a $m \in \mathcal{L}(\mathfrak{A}_2)$: finalement, on a bien montré que $\mathcal{L}(\mathfrak{A}_0) + \mathcal{L}(\mathfrak{A}_1) \subseteq \mathcal{L}(\mathfrak{A}_2)$.
$\supseteq$: soit $m \in \mathcal{L}(\mathfrak{A}_2)$: alors il existe $q_0 \xrightarrow{m_0} q_1 \xrightarrow{m_1} q_2 \xrightarrow{m_2} \dots \xrightarrow{m_{|m|-1}} q_{|m|}$ un calcul acceptant m dans $\mathfrak{A}_2$. Supposons que $q_0 \in Q_0$. Montrons alors par récurrence que pour tout $k \in [\![0, |m|]\!]$, $q_k \in Q_0$.
Initialisation : $k = 0$: notre hypothèse initiale nous donne $q_0 \in Q_0$.
Hérédité : supposons que $q_k \in Q_0$, et que $k < |m|$. Considérons alors la transition $q_k \xrightarrow{m_k} q_{k+1} \in T_0 \cup T_1$. Si on avait $q_k \xrightarrow{m_k} q_{k+1} \in T_1$, on aurait $q_k \in Q_1$: absurde ! Donc $q_k \xrightarrow{m_k} q_{k+1} \in T_0 \subseteq Q_0 \times \mathcal{A} \times Q_0$. Donc $q_{k+1} \in Q_0$.
Donc finalement, $q_0 \xrightarrow{m_0} q_1 \xrightarrow{m_1} \dots \xrightarrow{m_{|m|-1}} q_{|m|}$ est aussi un calcul acceptant dans $\mathfrak{A}_0$: donc $m \in \mathcal{L}(\mathfrak{A}_0)$.
Symétriquement, si $q_0 \in Q_1$, on a $m \in \mathcal{L}(\mathfrak{A}_1)$. En conclusion, $\mathcal{L}(\mathfrak{A}_2) \subseteq \mathcal{L}(\mathfrak{A}_0) + \mathcal{L}(\mathfrak{A}_1)$.

Propriété 27 : stabilité par concaténation

Soit $L, L' \in \mathrm{Rec}(\mathcal{A}^*)$. Alors $L \cdot L' \in \mathrm{Rec}(\mathcal{A}^*)$.

Démonstration.

Soit $\mathfrak{A}_0 = (\mathcal{A}, Q_0, I_0, F_0, T_0)$ et $\mathfrak{A}_1 = (\mathcal{A}, Q_1, I_1, F_1, T_1)$ deux automates finis. De nouveau, on suppose que les états et les transitions des deux automates sont distincts. Posons alors $\mathfrak{A}_2 = (\mathcal{A}, Q_0 \cup Q_1, I_0, F_1, T_0 \cup T_1 \cup$

T_ε) avec $T_\varepsilon = \{f_0 \xrightarrow{\varepsilon} i_1 \mid f_0 \in F_0 \text{ et } i_1 \in I_1\}$ (donc $\mathfrak{A}_2$ est un automate fini avec ε-transitions). Montrons :

$$\mathcal{L}(\mathfrak{A}_0) \cdot \mathcal{L}(\mathfrak{A}_1) = \mathcal{L}(\mathfrak{A}_2)$$

$\subseteq$: Soit $m \in \mathcal{L}(\mathfrak{A}_0)$ et $\tilde{m} \in \mathcal{L}(\mathfrak{A}_1)$; on a donc $q_0 \xrightarrow{m_0} q_1 \xrightarrow{m_1} q_2 \xrightarrow{m_2} \dots \xrightarrow{m_{|m|-1}} q_{|m|}$ un calcul dans $\mathfrak{A}_0$ acceptant m, et $\tilde{q}_0 \xrightarrow{\tilde{m}_0} \tilde{q}_1 \xrightarrow{\tilde{m}_1} \tilde{q}_2 \xrightarrow{\tilde{m}_2} \dots \xrightarrow{\tilde{m}_{|\tilde{m}|-1}} \tilde{q}_{|\tilde{m}|}$ un calcul dans $\mathfrak{A}_1$ acceptant $\tilde{m}$. En particulier, $q_{|m|} \in F_0$ et $\tilde{q}_0 \in I_1$. Donc $q_0 \xrightarrow{m_0} q_1 \xrightarrow{m_1} \dots \xrightarrow{m_{|m|-1}} q_{|m|} \xrightarrow{\varepsilon} \tilde{q}_0 \xrightarrow{\tilde{m}_0} \tilde{q}_1 \xrightarrow{\tilde{m}_1} \dots \xrightarrow{\tilde{m}_{|\tilde{m}|-1}} \tilde{q}_{|\tilde{m}|}$ est un calcul acceptant $m \cdot \tilde{m}$ dans $\mathfrak{A}_2$.
$\supseteq$: Soit $m \in \mathcal{L}(\mathfrak{A}_2)$, alors il existe $q_0 \xrightarrow{m_0} q_1 \xrightarrow{m_1} q_2 \xrightarrow{m_2} \dots \xrightarrow{m_{|m|-1}} q_{|m|}$ un calcul acceptant m dans $\mathcal{L}(\mathfrak{A}_2)$. On sait que $q_0 \in I_0$. Supposons qu'on ne passe par aucune transition de T_ε : alors, par récurrence, on montre que tout état du calcul appartient à Q_0.
Initialisation : $q_0 \in I_0 \subseteq Q_0$.
Hérédité : si $q_k \in Q_0$, alors la transition suivante est $q_k \xrightarrow{m_k} q_{k+1}$: comme $q_k \in Q_0$, la transition ne peut appartenir à T_1 ; par hypothèse, on a aussi supposé que la transition n'appartenait pas à T_ε. Donc la transition appartient à T_0 : donc $q_{k+1} \in Q_0$.
Donc on aurait $q_{|m|-1} \in Q_0$, pourtant, le calcul étant acceptant, $q_{|m|-1} \in F_1$: absurde! Donc l'une des transitions appartient à T_ε. Le calcul a alors la forme $q_0 \xrightarrow{\tilde{m}}{}^* f_0 \xrightarrow{\varepsilon} i_1 \xrightarrow{\overline{m}}{}^* q_{|m|-1}$. Par minimalité, on peut supposer que notre fameuse ε-transition est la première, et donc que toutes les transitions du calcul $q_0 \xrightarrow{\tilde{m}}{}^* f_0$ utilisent des transitions de T_0 : comme $q_0 \in I_0$ et $f_0 \in F_0$, on en déduit que $\tilde{m} \in \mathcal{L}(\mathfrak{A}_0)$.
Par ailleurs, par une autre récurrence immédiate, on peut montrer qu'aucune transition de $i_1 \xrightarrow{\overline{m}}{}^* q_{|m|-1}$ ne peut appartenir à T_0 ou à T_ε : ce sont toutes des transitions de T_1. Donc comme $i_1 \in I_1$ et $q_{|m|-1} \in F_1$, on en déduit que $\overline{m} \in \mathcal{L}(\mathfrak{A}_1)$. Donc $m = \tilde{m} \cdot \overline{m}$ avec $\tilde{m} \in \mathcal{L}(\mathfrak{A}_0)$ et $\overline{m} \in \mathcal{L}(\mathfrak{A}_1)$: donc $m \in \mathcal{L}(\mathfrak{A}_0) \cdot \mathcal{L}(\mathfrak{A}_1)$.

Propriété 28 : stabilité par étoile

Soit $L \in \mathrm{Rec}(\mathcal{A}^*)$. Alors $L^* \in \mathrm{Rec}(\mathcal{A}^*)$.

Démonstration.

Soit $\mathfrak{A} = (\mathcal{A}, Q, I, F, T)$ un automate fini. Alors on pose $\mathfrak{B} = (\mathcal{A}, Q, I, F, T \cup T_\varepsilon)$ où $T_\varepsilon = \{f \xrightarrow{\varepsilon} i \mid f \in F \text{ et } i \in I\}$. Montrons alors que $\mathcal{L}(\mathfrak{A}))^+ = \mathcal{L}(\mathfrak{B})$.

$\subseteq$: soit $n \in \mathbb{N}^*$ et $m^{(1)}, \dots, m^{(n)} \in \mathcal{L}(\mathfrak{A})$. Pour chacun de ces mots, on a un calcul $i^{(j)} \xrightarrow{m^{(j)}}{}^* f^{(j)}$ acceptant $m^{(j)}$ dans $\mathfrak{A}$. Remarquons alors que pour tout $j \in [\![1, n-1]\!]$, $f^{(j)} \xrightarrow{\varepsilon} i^{(j+1)} \in T_\varepsilon$.
Donc $i^{(1)} \xrightarrow{m^{(1)}}{}^* f^{(1)} \xrightarrow{\varepsilon} i^{(2)} \xrightarrow{m^{(2)}}{}^* f^{(2)} \xrightarrow{\varepsilon} \dots \xrightarrow{\varepsilon} i^n \xrightarrow{m^{(n)}}{}^* f^{(n)}$ est un calcul acceptant $m = m^{(1)} \cdot \ldots \cdot m^{(n)}$ dans $\mathfrak{B}$. Donc $(\mathcal{L}(\mathfrak{A}))^+ \subseteq \mathcal{L}(\mathfrak{B})$.
$\supseteq$: soit $m \in \mathcal{L}(\mathfrak{B})$, on a donc $i \xrightarrow{m}{}^* f$ un calcul acceptant m dans $\mathfrak{B}$. Découpons ce calcul selon les transitions de T_ε : on obtient $i = q_0 \xrightarrow{\varepsilon}{}^* p_1 \xrightarrow{m^{(1)}}{}^* q_1 \xrightarrow{\varepsilon}{}^* p_2 \xrightarrow{m^{(2)}}{}^* q_2 \xrightarrow{\varepsilon}{}^* \dots \xrightarrow{\varepsilon}{}^* p_n \xrightarrow{m^{(n)}}{}^* q_n \xrightarrow{\varepsilon}{}^* p_{n+1} = f$, avec l'hypothèse que pour tout $i \in [\![0, n]\!]$, $q_i \xrightarrow{\varepsilon}{}^* p_{n+1}$ est constitué exclusivement de transitions de T_ε. On obtient donc que pour tout $j \in [\![0, n]\!]$, $q_j \in F$ et $p_{j+1} \in I$; donc pour tout $j \in [\![1, n]\!]$, $p_j \xrightarrow{m^{(j)}}{}^* q_j$ commence dans I, termine dans F et n'est constitué que de transitions de T : donc $m^{(j)} \in \mathcal{L}(\mathfrak{A})$. Comme $m = m^{(1)} \cdot \ldots \cdot m^{(n)}$, on a donc $m \in (\mathcal{L}(\mathfrak{A}))^+$.
Donc si un langage L est reconnaissable, L^* est aussi reconnaissable. Pour conclure, on remarque que pour tout langage L, $L^* = L^+ + \{\varepsilon\}$; par stabilité par addition, et comme $\{\varepsilon\}$ est reconnaissable, on en déduit que si L est reconnaissable, alors L^* est aussi reconnaissable.

Corollaire 29

$\mathrm{Rat}(\mathcal{A}^*) \subseteq \mathrm{Rec}(\mathcal{A}^*)$.

Démonstration.

Comme $\varnothing \in \mathrm{Rec}(\mathcal{A}^*)$, $\{a\} \in \mathrm{Rec}(\mathcal{A}^*)$ et $\mathrm{Rec}(\mathcal{A}^*)$ est stable par addition, concaténation et étoile, $\mathrm{Rat}(\mathcal{A}^*)$ étant le plus petit ensemble vérifiant ces propriétés, $\mathrm{Rat}(\mathcal{A}^*) \subseteq \mathrm{Rec}(\mathcal{A}^*)$.

3.2.3 Inclusion indirecte

Pour l'inclusion indirecte, nous allons explorer, de nouveau, une construction légèrement augmentée des automates finis : ceux dont les étiquettes sont des langages réguliers.

Définition 30 : automates finis généralisés

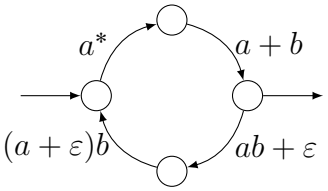
Soit $\mathcal{A}$ un alphabet. Un automate fini généralisé $\mathfrak{A}$ est un quintuplet $(\mathcal{A}, Q, I, F, T)$ où :

Q est un ensemble fini d'états ;

$I \subseteq Q$ et $F \subseteq Q$ sont les états initiaux et finaux ;

$T \subseteq Q \times \mathrm{Reg}(\mathcal{A}^*) \times Q$ est l'ensemble des transitions.

Un mot est accepté par un automate fini généralisé s'il est possible de le découper en facteurs de sorte à ce qu'en lisant chaque facteur l'un après l'autre, on puisse partir d'un état initial et arriver à un état final.
Par exemple, ici, prenons $m = aaaababaaa$, alors on peut l'écrire comme $m = aaaa \cdot b \cdot \varepsilon \cdot ab \cdot aa \cdot a$: avec ce découpage, on remarque que le mot est accepté par l'automate ci-contre.
En revanche, le mot $m' = bb$, lui, n'est pas accepté.



Propriété 31 : l'extension n'apporte rien de plus au modèle

Soit $\mathfrak{A}$ un automate fini généralisé : alors il existe un automate fini $\mathfrak{B}$ (non généralisé) tel que $\mathcal{L}(\mathfrak{A}) = \mathcal{L}(\mathfrak{B})$.

Démonstration.

Soit $\mathfrak{A} = (\mathcal{A}, Q, I, F, T)$: considérons une transition $q \xrightarrow{E} r$, où E est une expression régulière.
On a prouvé précédemment que $\mathrm{Reg}(\mathcal{A}^*) = \mathrm{Rat}(\mathcal{A}^*) \subseteq \mathrm{Rec}(\mathcal{A}^*)$: on dispose donc d'un automate fini $\mathfrak{A}_E$ reconnaissant le langage de E. On va modifier légèrement $\mathfrak{A}_E$ pour en faire un automate à ε-transitions, de sorte à construire un automate $\mathfrak{A}'_E$ reconnaissant le même langage, mais n'admettant qu'un seul état initial et un seul état final. Cela se fait en ajoutant à $\mathfrak{A}_E$ un état i et un état f et en ajoutant pour tout état initial q_i de $\mathfrak{A}_E$ une ε-transition $i \xrightarrow{\varepsilon} q_i$, de même en ajoutant $q_f \xrightarrow{\varepsilon} f$ pour tout état final q_f de $\mathfrak{A}_E$; i et f sont alors les nouveaux état inial et final de $\mathfrak{A}'_E$.
L'idée est alors de remplacer la transition $q \xrightarrow{E} r$ dans $\mathfrak{A}$ par l'automate $\mathfrak{A}'_E$, où i prend la place de q et f de r. En procédant à cette substitution, on a remplacé une transition étiquetée par un langage régulier par tout un automate fini avec ε-transitions, et le langage de l'automate global n'a pas changé ; on procède ainsi pour toutes les transitions de $\mathfrak{A}$. On a alors construit $\mathfrak{A}'$ un automate à ε-transitions tel que $\mathcal{L}(\mathfrak{A}) = \mathcal{L}(\mathfrak{A}')$; comme on a déjà montré que les ε-transitions n'apportaient rien au modèle des automates finis, il existe un automate fini $\mathfrak{B}$ tel que $\mathcal{L}(\mathfrak{A}) = \mathcal{L}(\mathfrak{B})$.

Propriété 32

Soit $L \in \mathrm{Rec}(\mathfrak{A}^*)$. Alors il existe une expression régulière E telle que $\mathcal{L}(E) = L$.

Démonstration.

Soit $\mathfrak{A} = (\mathcal{A}, Q, I, F, T)$ un automate fini. Montrons qu'il existe une expression régulière E telle que $\mathcal{L}(E) = \mathcal{L}(\mathfrak{A})$. On va pour cela procéder algorithmiquement : cette technique s'appelle **méthode de Brzozowski** ou

algorithme d'élimination des états.
On commence déjà par rajouter deux états, le premier nommé α, l'autre ω, et des ε-transitions reliant α à tous les états initiaux de $\mathfrak{A}$, et tous les états finaux de $\mathfrak{A}$ à ω.
On fait en sorte que toutes les boucles soient sous forme d'un seul langage régulier : il suffit de faire un plus. On pourra alors parler *du* langage régulier d'une boucle.
Puis, considérons un état différent de α et ω, noté q.
S'il n'existe pas de boucle $q \xrightarrow{E} q$, pour tous états p et r et expressions régulières E et E' tels que $p \xrightarrow{E} q$ et $q \xrightarrow{E'} r$, on ajoute à $\mathfrak{A}$ la transition $p \xrightarrow{E \cdot E'} r$.
S'il existe une boucle $q \xrightarrow{G} q$. pour tous états p et r et expressions régulières E et E' tels que $p \xrightarrow{E} q$ et $q \xrightarrow{E'} r$, on ajoute à $\mathfrak{A}$ la transition $p \xrightarrow{E \cdot G^* \cdot E'} r$.
Si, désormais, un état possède plusieurs boucles, on les unifie en un seul langage régulier à l'aide d'un plus.
Enfin, on retire à $\mathfrak{A}$ l'état q, ainsi que toutes les transitions qui en partent et y arrivent.
Notons $\mathfrak{A}'$ le nouvel automate obtenu. On affirme alors que $\mathcal{L}(\mathfrak{A}) = \mathcal{L}(\mathfrak{A}')$.
Si un mot m est accepté par $\mathfrak{A}$, considérons un calcul acceptant $\alpha \xrightarrow{w_0} q_1 \xrightarrow{w_1} \dots \xrightarrow{w_k} \omega$ (ici, les w_i sont des mots, et non plus les lettres, car on manipule un automate fini généralisé) (on a donc $m = w_0 \cdot \dots w_k$). S'il y a une transition de la forme $p \xrightarrow{w_i} q \xrightarrow{w_{i+1}} r$ ($p, r \neq q$), alors par définition de $\mathfrak{A}'$, il y aura dans $\mathfrak{A}'$ une transition de la forme $p \xrightarrow{E} r$ où $w_i \cdot w_{i+1} \in \mathcal{L}(E)$. S'il y a une transition de la forme $p \xrightarrow{w_i} q \xrightarrow{w_{i+1}} q$, alors on peut concaténer ces transitions jusqu'à avoir $p \xrightarrow{w_i} q \xrightarrow{w_{i+1}} q \xrightarrow{w_{i+2}} \dots \xrightarrow{w_j} q \xrightarrow{w_{j+1}} r$, avec p et r différents de q (c'est toujours possible car $q \neq \alpha$ et $\neq \omega$); alors $w_{i+1} \cdot w_{i+2} \cdot \dots \cdot w_j \in E^*$ où E est le langage de la boucle autour de q. Donc, dans $\mathfrak{A}'$, on a une transition $p \xrightarrow{F \cdot E^* \cdot F'} r$ où $w_i \cdot w_{i+1} \cdot \dots \cdot w_j \cdot w_{j+1} \in \mathcal{L}(F \cdot E^* \cdot F')$.
Donc on peut transformer tout calcul acceptant dans $\mathfrak{A}$ en un calcul acceptant dans $\mathfrak{A}'$: $\mathcal{L}(\mathfrak{A}) \subseteq \mathcal{L}(\mathfrak{A}')$.
Réciproquement, si un mot est accepté par $\mathfrak{A}'$, en considérant un calcul acceptant, il suffira de considérer les transitions qui ont été ajoutées au passage de $\mathfrak{A}$ à $\mathfrak{A}'$; ces transitions peuvent être redécoupées en transitions de $\mathfrak{A}$, donc $\mathcal{L}(\mathfrak{A}') \subseteq \mathcal{L}(\mathfrak{A})$.
On vient de prouver que notre algorithme disposait d'un invariant de boucle (le langage de l'automate); on remarque alors qu'il dispose aussi d'un variant de boucles (le nombre d'états différents de α et ω).
Une remarque importante est que, dans notre algorithme, on ne créera pas de transition pointant vers α, ni de transition partant de ω ; en effet, tout état dont part une nouvelle transition était un état dont partait déjà une transition. Autrement écrit, un passage de boucle ne transforme pas d'état de « état sans départ » à « état avec départ ».

En conclusion, à la fin de notre algorithme, l'automate obtenu a une forme très simple : notre invariant de boucle nous permet de conclure que $\mathcal{L}(E) = \mathcal{L}(\mathfrak{A})$.

Corollaire 33

$\mathrm{Rec}(\mathcal{A}^*) \subseteq \mathrm{Reg}(\mathcal{A}^*)$.

4 Acceptation d'un mot par un automate fini

4.1 Problème de l'acceptation d'un mot par un automate fini

Définition 34 : problème de l'acceptation

Le problème de l'acceptation est le problème informatique suivant :

Donnée : un automate fini $\mathfrak{A}$, un mot m

Question : $m \in \mathcal{L}(\mathfrak{A})$?

Algorithme 35 : acceptation par parcours en profondeur

Données : un automate fini $\mathfrak{A} = (\mathcal{A}, Q, I, F, T)$ et un mot m
$\mathcal{F} \leftarrow$ la file des couples (i, m) pour $i \in I$
tant que *P n'est pas vide, ou qu'il n'y a pas eu d'interruption précédente* **:**
  défiler (q, w) le premier élément de $\mathcal{F}$
  si $w = \varepsilon$ *et* $q \in F$ **:**
    retourner *vrai*
  si $w \neq \varepsilon$ **:**
    soit $w = ax$
    enfiler à $\mathcal{F}$ les couples (r, x) tels que $q \xrightarrow{a} r \in T$
retourner *faux*

Dans le pire cas, cet algorithme nécessite $O(|Q|^{|m|})$.

En général, on note $|\mathfrak{A}| = \max(|Q|, |T|)$.

Démonstration.

Dans le pire cas, pour chaque couple (q, w), on rajoute tous les couples (r, x) avec $r \in Q$, ce qui donne bien $O(|Q|^{|m|})$ étapes.

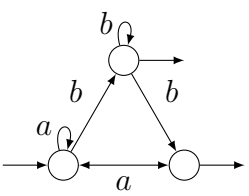
Il existe un autre algorithme par parcours en largeur du même accabit.

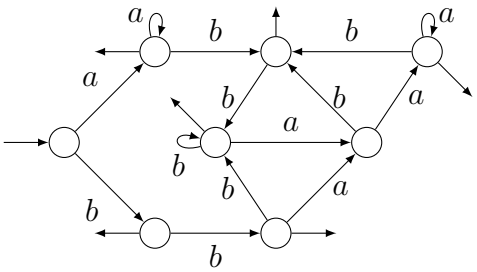
4.2 Automate fini déterministe

Définition 36 : automate fini déterministe

Un automate fini $\mathfrak{A} = (\mathcal{A}, Q, I, F, T)$ est dit **déterministe** si $\text{Card}(I) = 1$ et si pour tout état $q \in Q$, pour toute lettre $a \in \mathcal{A}$, il existe au plus un état $r \in Q$ tel que $q \xrightarrow{a} r \in T$.
On peut aussi définir un automate fini déterministe comme un uplet $\mathfrak{A} = (\mathcal{A}, Q, i, F, \delta)$ où $\mathcal{A}$, Q, et F sont définis comme d'habitude, $i \in Q$ est l'état initial, et $\delta : Q \times \mathcal{A} \to Q$ est une fonction partielle, qui, étant donné un état q et une lettre a, renvoie, s'il existe, le seul état atteint en partant de q en lisant a.

Ci-contre, l'automate de gauche n'est pas déterministe : l'état le plus à gauche a deux arêtes sortantes étiquetées par a. L'automate de droite, quant à lui, est déterministe. Remarquons que les deux automates ont le même langage.





Algorithme 37 : acceptation pour un automate déterministe

Données : un aut. fini déterministe $\mathfrak{A} = (\mathcal{A}, Q, i, F, \delta)$ et un mot m
$s \leftarrow i$
pour *toute lettre* a *de* m **:**
  si $\delta(s, a)$ *n'est pas défini* **:**
    retourner *faux*
  $s \leftarrow \delta(s, a)$
si $s \in F$ **:**
  retourner *vrai*
sinon
  retourner *faux*

Dans le pire cas, cet algorithme nécessite $O(|m|)$ appels à la fonction δ.

Selon l'implémentation de l'automate, la complexité change. Une idée peut être d'implémenter δ comme un tableau de tableaux : appeler une valeur de δ est alors un $O(1)$.

Une notation usuelle importante est la généralisation de la fonction δ (définie sur $Q \times \mathcal{A}$) aux mots :

Définition 38 : fonction de transition généralisée

Soit $\mathfrak{A} = (\mathcal{A}, Q, i, F, \delta)$ un automate fini déterministe. On appelle fonction de transition généralisée, notée usuellement δ^*, la fonction partielle $\delta^* : Q \times \mathcal{A}^*$ définie récursivement par :

$\delta^*(q, \varepsilon) = q$;

$\delta^*(q, ma) = \delta(\delta^*(q, m), a)$.

De manière équivalente, on a aussi $\delta^*(q, am) = \delta^*(\delta(q, a), m)$.

L'algorithme d'acceptation pour un automate déterministe revient à calculer $\delta^*(i, m)$; de manière générale, pour tout état q et mot m, le calcul de $\delta^*(q, m)$ est généralement considéré comme linéaire en $|m|$.

Les automates finis déterministes sont donc utiles d'un point de vue pratique, car ils permettent d'efficacement pouvoir résoudre le problème de l'acceptation. Ce serait formidable si tous les automates finis pouvaient être rendus déterministes !

4.3 Déterminisation des automates

Propriété 39 : tout automate fini a un équivalent déterministe

Soit $\mathfrak{A}$ un automate fini. Alors il existe un automate fini déterministe $\mathfrak{B}$ tel que $\mathcal{L}(\mathfrak{A}) = \mathcal{L}(\mathfrak{B})$.

Démonstration.

Soit $\mathfrak{A} = (\mathcal{A}, Q, I, F, T)$ un automate fini. Quitte à le compléter, on peut supposer $\mathfrak{A}$ complet.
On définit alors $\mathfrak{A}_\mathcal{P} = (\mathcal{A}, \mathcal{P}(Q), \{I\}, F_\mathcal{P}, \delta)$ l'automate fini déterministe défini par :

$F_\mathcal{P} = \{E \in \mathcal{P}(Q) \mid E \cap F \neq \varnothing\}$;

$\delta(E, a) = \bigcup_{e \in E} \{q \in Q \mid e \xrightarrow{a} q \in T\}$.

Cet automate est appelé **automate des parties de** $\mathfrak{A}$. Par définition, $\mathfrak{A}_\mathcal{P}$ est un automate fini déterministe. Reste à montrer que $\mathcal{L}(\mathfrak{A}) = \mathcal{L}(\mathfrak{A}_\mathcal{P})$.

$\subseteq$: Soit $m \in \mathcal{L}(\mathfrak{A})$, et $q_0 \xrightarrow{m_0} q_1 \xrightarrow{m_1} \dots \xrightarrow{m_{|m|-1}} q_{|m|}$ un calcul acceptant m. Montrons par récurrence que pour tout $i \in [\![0, |m|]\!]$, il existe un état E_i de $\mathfrak{A}_\mathcal{P}$ tel que $q_i \in E_i$ et $E_0 \xrightarrow{m_0} E_1 \xrightarrow{m_1} \dots \xrightarrow{m_{i-1}} E_i$.

Initialisation : on pose $E_0 = I$. Comme le calcul dans $\mathfrak{A}$ est acceptant, cela signifie que $q_0 \in I$: donc on a bien $q_0 \in E_0$.
Hérédité : on suppose que pour $i \in [\![0, |m| - 1]\!]$, il existe $E_i \in \mathcal{P}(Q)$ tel que $q_i \in E_i$ et $E_0 \xrightarrow{m_0} E_1 \xrightarrow{m_1} \dots \xrightarrow{m_{i-1}} E_i$. Pour continuer le calcul dans $\mathfrak{A}_\mathcal{P}$, on n'a pas vraiment le choix : posons $E_{i+1} = \delta(E_i, m_i)$. Alors, comme $q_i \in E_i$ et $q_i \xrightarrow{m_i} q_{i+1} \in T$, par définition de δ, $q_{i+1} \in E_{i+1}$.
Donc on a bien un chemin $E_0 \xrightarrow{m_0} E_1 \xrightarrow{m_1} \dots \xrightarrow{m_{|m|-1}} E_{|m|}$ dans $\mathfrak{A}_\mathcal{P}$. On a $E_0 = \{I\}$; notre récurrence nous montre que $q_{|m|} \in E_{|m|}$. Or, comme le premier calcul était acceptant dans $\mathfrak{A}$, alors $q_{|m|} \in F$: donc $E_{|m|}$ contient un état final, et donc $E_{|m|} \in F_\mathcal{P}$. Donc le calcul $E_0 \xrightarrow{m_0} E_1 \xrightarrow{m_1} \dots \xrightarrow{m_{|m|-1}} E_{|m|}$ est bien acceptant dans $\mathfrak{A}_\mathcal{P}$: donc $m \in \mathcal{L}(\mathfrak{A}_\mathcal{P})$.
$\supseteq$: soit $m \in \mathcal{L}(\mathfrak{A}_\mathcal{P})$, et soit $E_0 \xrightarrow{m_0} E_1 \xrightarrow{m_1} \dots \xrightarrow{m_{|m|-1}} E_{|m|}$ le calcul acceptant associé (on a donc $E_0 = I$). On va construire un calcul dans $\mathfrak{A}$ à reculons : montrons que pour tout $i \in [\![0, |m|]\!]$, il existe $q_i \in E_i$ tel que $q_i \xrightarrow{m_i} q_{i+1} \xrightarrow{m_{i+1}} \dots \xrightarrow{m_{|m|-1}} q_{|m|}$ est un chemin terminant par un état final dans $\mathfrak{A}$.
Initialisation : On sait que $E_{|m|} \in F_\mathcal{P}$; donc il existe $f \in F$ tel que $f \in E_{|m|}$. On pose alors $q_{|m|} = f$.
Hérédité : supposons que pour $i \in [\![1, |m|]\!]$, on a construit un chemin $q_i \xrightarrow{m_i} q_{i+1} \xrightarrow{m_{i+1}} \dots \xrightarrow{m_{|m|-1}} q_{|m|}$ dans $\mathfrak{A}$ terminant par un état final avec $q_i \in E_i$. Donc $q_i \in E_i = \delta(E_{i-1}, m_{i-1})$. Alors il existe $e \in E_{i-1}$ tel que $e \xrightarrow{m_{i-1}} q_i \in T$: posons donc $q_{i-1} = e$. Alors $q_{i-1} \xrightarrow{m_{i-1}} q_i \xrightarrow{m_i} q_{i+1} \xrightarrow{m_{i+1}} \dots \xrightarrow{m_{|m|-1}} q_{|m|}$ est un chemin dans $\mathfrak{A}$ terminant par un état final avec $q_{i-1} \in E_{i-1}$.
On dispose donc dans $\mathfrak{A}$ d'un chemin de la forme $q_0 \xrightarrow{m_0} q_1 \xrightarrow{m_1} \dots \xrightarrow{m_{|m|-1}} q_{|m|}$ terminant par un état final avec $q_0 \in E_0$; mais $E_0 = I$, donc $q_0 \in I$. Finalement, notre chemin est bien un calcul acceptant m dans $\mathfrak{A}$: $m \in \mathcal{L}(\mathfrak{A})$.

4.4 Automate fini complet

Définition 40 : automate fini complet

Un automate fini $\mathfrak{A} = (\mathcal{A}, Q, F, I, T)$ est complet si pour tout état $q \in Q$ et toute lettre $a \in \mathcal{A}$, il existe un état r tel que $q \xrightarrow{a} r \in T$.

Propriété 41 : Complétion des automates

Soit $\mathfrak{A}$ un automate fini. Alors il existe $\mathfrak{B}$ un automate fini complet tel que $\mathcal{L}(\mathfrak{A}) = \mathcal{L}(\mathfrak{B})$.

Démonstration.

Soit $\mathfrak{A} = (\mathcal{A}, Q, I, F, T)$, et soit $p \notin Q$. On pose $\mathfrak{B} = (\mathcal{A}, Q \cup \{p\}, I, F, T \cup T_c \cup T_p)$ où $T_c = \{q \xrightarrow{a} p \mid \text{il n'existe pas d'autre état } r \in Q \text{ tel que } q \xrightarrow{a} r \in T\}$ et $T_p = \{p \xrightarrow{a} p \mid a \in \mathcal{A}\}$. Montrons que $\mathcal{L}(\mathfrak{A}) = \mathcal{L}(\mathfrak{B})$.
$\subseteq$: Tout calcul acceptant dans $\mathfrak{A}$ est aussi un calcul acceptant dans $\mathfrak{B}$.
$\supseteq$: Supposons qu'un calcul acceptant dans $\mathfrak{B}$ passe par l'état p. Toute transition sortant de p arrive dans p. Donc ce calcul acceptant serait ultimement constant en p, et terminerait en l'état p : mais cet état n'est pas final. Absurde !
Donc tout calcul acceptant dans $\mathfrak{B}$ ne passe pas par l'état p : c'est donc aussi un calcul acceptant dans $\mathfrak{A}$.
Par ailleurs, il faut justifier la complétude de $\mathfrak{B}$: soit q un état de $\mathfrak{B}$ et $a \in \mathcal{A}$.

1er cas : $q \in Q$

1er sous-cas : il existe un état $r \in Q$ tel que $q \xrightarrow{a} r \in T$.

2nd sous-cas : il n'existe pas d'état $r \in Q$ tel que $q \xrightarrow{a} r \in T$. Alors par définition de T_c, $q \xrightarrow{a} p \in T_c$.

2nd cas : $q = p$. Alors pour tout $a \in \mathcal{A}$, $p \xrightarrow{a} p \in T_p$.

Propriété 42

Soit $\mathfrak{A}$ un automate fini. Alors il existe un automate fini **déterministe et complet** $\mathfrak{B}$ tel que $\mathcal{L}(\mathfrak{A}) = \mathcal{L}(\mathfrak{B})$.

Démonstration.

Il suffit de relire la preuve de la déterminisation d'un automate fini : l'automate des parties qui y est construit est déterministe et complet.

4.5 Conséquence théorique

Propriété 43 : stabilité par complémentaire

Soit $L \in \mathrm{Rec}(\mathcal{A}^*)$. Alors $\mathcal{A}^* \backslash L \in \mathrm{Rec}(\mathcal{A}^*)$.

Démonstration.

Soit $\mathfrak{A}$ un automate fini. Quitte à déterminiser et compléter, on peut supposer que $\mathfrak{A}$ est déterministe et complet : soit $\mathfrak{A} = (\mathcal{A}, Q, i, F, \delta)$. On pose alors $\overline{\mathfrak{A}} = (\mathcal{A}, Q, i, Q \backslash F, \delta)$. Remarquons alors que $\overline{(\mathfrak{A})}$ est lui-même déterministe et complet. Montrons que $\mathcal{L}(\overline{\mathfrak{A}}) = \mathcal{A}^* \backslash \mathcal{L}(\mathfrak{A})$.

$\subseteq$: Soit $m \in \mathcal{L}(\overline{\mathfrak{A}})$. Considérons alors son calcul acceptant $q_0 \xrightarrow{m_0} q_1 \xrightarrow{m_1} \dots \xrightarrow{m_{|m|-1}} q_{|m|}$: par déterminisme de $\overline{\mathfrak{A}}$, ce calcul est unique. Donc dans $\mathfrak{A}$, il existe un seul chemin partant de i étiqueté par m, et il s'agit aussi de $q_0 \xrightarrow{m_0} q_1 \xrightarrow{m_1} \dots \xrightarrow{m_{|m|-1}} q_{|m|}$. Seulement, si le calcul est acceptant dans $\overline{\mathfrak{A}}$, cela signifie que $q_{|m|} \in Q \backslash F$; donc le calcul en question est rejeté par $\mathfrak{A}$. Donc $m \notin \mathcal{L}(\mathfrak{A})$: autrement écrit, $m \in \mathcal{A}^* \backslash \mathcal{L}(\mathfrak{A})$.

$\supseteq$: Le raisonnement symétrique tient.

Le théorème de Kleene étend cette stabilité aux langages rationnels.

Propriété 44 : stabilité par intersection

Soit $L, L' \in \mathrm{Rec}(\mathcal{A}^*)$. Alors $L \cap L' \in \mathrm{Rec}(\mathcal{A}^*)$.

Démonstration.

Il suffit de remarquer que $L \cap L' = \mathcal{A}^* \backslash ((\mathcal{A}^* \backslash L) + (\mathcal{A}^* \backslash L'))$.

En fait, il n'y a pas vraiment besoin de déterminisme ou de complétude.

Soit $\mathfrak{A}_0 = (\mathcal{A}, Q_0, I_0, F_0, T_0)$ et $\mathfrak{A}_1 = (\mathcal{A}, Q_1, I_1, F_1, T_1)$ deux automates finis. On définit l'**automate produit** $\mathfrak{A}_2 = (\mathcal{A}, Q_0 \times Q_1, I_0 \times I_1, F_0 \times F_1, T')$ avec $T' = \{(q^{(0)}, q^{(1)}) \xrightarrow{a} (r^{(0)}, r^{(1)}) \mid q^{(0)} \xrightarrow{a} r^{(0)} \in T_0 \text{ et } q^{(1)} \xrightarrow{a} r^{(1)} \in T_1\}$. Montrons que $\mathcal{L}(\mathfrak{A}_0) \cap \mathcal{L}(\mathfrak{A}_1) = \mathcal{L}(\mathfrak{A}_2)$.

$\subseteq$: soit $m \in \mathcal{L}(\mathfrak{A}_0) \cap \mathcal{L}(\mathfrak{A}_1)$. Alors il existe deux calculs, respectivement $q_0^{(0)} \xrightarrow{m_0} q_1^{(0)} \xrightarrow{m_1} \dots \xrightarrow{m_{|m|-1}} q_{|m|}^{(0)}$ et $q_0^{(1)} \xrightarrow{m_0} q_1^{(1)} \xrightarrow{m_1} \dots \xrightarrow{m_{|m|-1}} q_{|m|}^{(0)}$, le premier sur $\mathfrak{A}_0$, le second sur $\mathfrak{A}_1$, acceptant tous les deux m. Alors $(q_0^{(0)}, q_0^{(1)}) \xrightarrow{m_0} (q_1^{(0)}, q_1^{(1)}) \xrightarrow{m_1} \dots \xrightarrow{m_{|m|-1}} (q_{|m|}^{(0)}, q_{|m|}^{(1)})$ est un calcul acceptant de $\mathfrak{A}_2$.

$\supseteq$: soit $m \in \mathcal{L}(\mathfrak{A}_2)$. Alors son calcul est de la forme $(q_0^{(0)}, q_0^{(1)}) \xrightarrow{m_0} (q_1^{(0)}, q_1^{(1)}) \xrightarrow{m_1} \dots \xrightarrow{m_{|m|-1}} (q_{|m|}^{(0)}, q_{|m|}^{(1)})$. Alors $q_0^{(0)} \xrightarrow{m_0} q_1^{(0)} \xrightarrow{m_1} \dots \xrightarrow{m_{|m|-1}} q_{|m|}^{(0)}$ est un calcul acceptant m sur $\mathfrak{A}_0$, et $q_0^{(1)} \xrightarrow{m_0} q_1^{(1)} \xrightarrow{m_1} \dots \xrightarrow{m_{|m|-1}} q_{|m|}^{(0)}$ sur $\mathfrak{A}_1$. Donc $m \in \mathcal{L}(\mathfrak{A}_0) \cap \mathcal{L}(\mathfrak{A}_1)$.

4.6 Un dernier algorithme d'acceptation

Algorithme 45 : acceptation par « déterminisation à la volée »

Données : un automate fini $\mathfrak{A} = (\mathcal{A}, Q, I, F, T)$ et un mot m
$S \leftarrow I$ **pour** *toute lettre* a *de* m **:**
  $S \leftarrow$ états accessibles depuis les états de S en lisant a
si s *contient un état final* **:**
  retourner *vrai*
sinon
  retourner *faux*

Dans le pire cas, cet algorithme nécessite $O(|m| \times |Q|^2)$ (il y a peut-être mieux ?).

Démonstration.

La difficulté consiste en la mise à jour de S. Pour tout état q de S, on accède à ses successeurs de $O(1)$ (en supposant que l'automate soit stocké sous forme d'un dictionnaire, par exemple). On procède à la fusion successive des listes : en supposant les listes triées, chaque fusion coûte $O(|Q|)$. Donc la mise à jour coûte $O(|Q|^2)$.

4.7 Quitte à parler stabilité …

Définition 46 : morphisme du monoïde libre

Soit $\mathcal{A}$ un alphabet, une fonction $\phi : \mathcal{A}^* \to \mathcal{A}^*$ est un morphisme de monoïde si pour tous $u, v \in \mathcal{A}^*$, $\phi(u \cdot v) = \phi(u) \cdot \phi(v)$.

Pour définir un morphisme du monoîde libre, il suffit alors de définir l'image des lettres de l'alphabet.

Propriété 47 : définir un morphisme du monoïde libre

Soit $\mathcal{A}$ un alphabet, et ϕ et ψ deux morphismes de monoïde sur $\mathcal{A}^*$.
Alors $\phi = \psi$ ssi pour toute lettre $a \in \mathcal{A}$, $\phi(a) = \psi(a)$.

Démonstration.

Le sens direct est trivial, le sens réciproque est évident (récurrence).

Propriété 48 : stabilité par morphisme

Soit $L \in \text{Rec}(\mathcal{A}^*)$ et $\phi : \mathcal{A}^* \to \mathcal{A}^*$ un morphisme. Alors $\phi(L) \in \text{Rec}(\mathcal{A}^*)$.

Démonstration.

L'idée consiste à découper toute transition $p \xrightarrow{a} q$ en une succession de transitions $p \xrightarrow{\phi(a)}{}^* q$ en rajoutant des états intermédiaires.
Formellement, c'est affreux à formaliser.
En particulier, il faut prendre en compte le fait que le morphisme est peut-être effaçant, donc faire attention en introduisant les états et transitions supplémentaires.

Propriété 49 : stabilité par morphisme inverse

Soit $L \in \text{Rec}(\mathcal{A}^*)$ et $\phi : \mathcal{A}^* \to \mathcal{A}^*$ un morphisme. Alors $\phi^{-1}(L) = \{m \in \mathcal{A}^* \mid \phi(m) \in L\} \in \text{Rec}(\mathcal{A}^*)$.

Démonstration.

Soit $\mathfrak{A} = (\mathcal{A}, Q, I, F, T)$ un automate fini. On pose alors $\mathfrak{B} = (\mathcal{A}, Q, I, F, U)$ où $U = \{p \xrightarrow{a} q \mid$ il existe $p \xrightarrow{\phi(a)}{}^* q$ dans $\mathfrak{A}\}$.

Soit $m \in \phi^{-1}(\mathcal{L}(\mathfrak{A}))$. Alors il existe $w \in \mathcal{L}(\mathfrak{A})$ tel que $\phi(m) = w$; on peut alors considérer le calcul de w dans $\mathfrak{A}$. Alors on peut découper ce calcul selon les différents mots $\phi(m_j)$ pour $j \in [\![0, |m| - 1]\!]$, pour avoir le calcul $q_0 \xrightarrow{\phi(m_0)}{}^* q_1 \xrightarrow{\phi(m_1)}{}^* q_2 \xrightarrow{\phi(m_2)}{}^* \dots \xrightarrow{\phi(m_{|m|-1})}{}^* q_{|m|}$: comme c'est un calcul acceptant w dans $\mathfrak{A}$, $q_0 \in I$ et $q_{|m|} \in F$. Par définition de U, il existe toutes les transitions $q_j \xrightarrow{m_j} q_{j+1}$ pour $j \in [\![0, |m| - 1]\!]$. Donc $q_0 \xrightarrow{m_0} q_1 \xrightarrow{m_1} q_2 \xrightarrow{m_2} \dots \xrightarrow{m_{|m|-1}} q_{|m|}$ est un calcul acceptant m dans $\mathfrak{B}$.

Soit $m \in \mathcal{L}(\mathfrak{B})$, et un calcul acceptant $q_0 \xrightarrow{m_0} q_1 \xrightarrow{m_1} \dots \xrightarrow{m_{|m|-1}} q_{|m|}$ dans $\mathfrak{B}$. Alors par défiition de U, on peut réécrire ce calcul en un calcul dans $\mathfrak{A}$ de la forme $q_0 \xrightarrow{\phi(m_0)}{}^* q_1 \xrightarrow{\phi(m_1)}{}^* \dots \xrightarrow{\phi(m_{|m|-1})}{}^* q_{|m|}$, avec $q_0 \in I$ et $q_{|m|} \in F$. Donc $\phi(m_0) \cdot \phi(m_1) \cdot \ldots \cdot \phi(m_{|m|-1}) = \phi(m)$ est accepté par $\mathfrak{A}$; donc $m \in \phi^{-1}(\mathcal{L}(\mathfrak{A}))$.

Les deux résultats tiennent même si le morphisme est effaçant, ie s'il existe $a \in \mathcal{A}, \phi(a) = \varepsilon$.

5 Algorithmique des automates finis

5.1 Émondage des automates

Définition 50 : état accessible, co-accessible

Soit $\mathfrak{A} = (\mathcal{A}, Q, I, F, T)$ un automate fini.
Un état q est accessible s'il existe $i \in I$ et $m \in \mathcal{A}^*$ tel que $i \xrightarrow{m}{}^* q$ dans $\mathfrak{A}$.
Un état q est co-accessible s'il existe $f \in F$ et $m \in \mathcal{A}^*$ tel que $q \xrightarrow{m}{}^* f$ dans $\mathfrak{A}$.

Définition 51 : automate émondé

Soit $\mathfrak{A}$ un automate fini. On dit que $\mathfrak{A}$ est émondé si tous ses états sont accessibles et co-accessibles.

Propriété 52 : Émondage

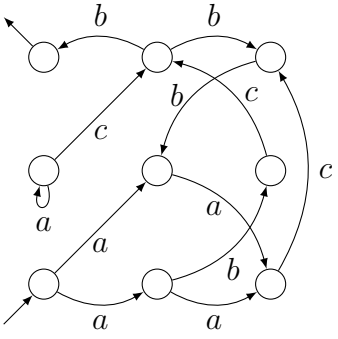
Soit $\mathfrak{A}$ un automate fini. Alors il existe $\mathfrak{A}$ un automate fini émondé tel que $\mathcal{L}(\mathfrak{A}) = \mathcal{L}(\mathfrak{B})$.
La construction de $\mathfrak{B}$ se fait en $O(|T|)$.

Démonstration.

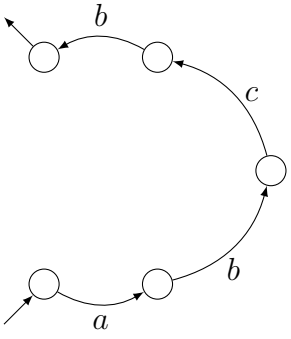
Soit $\mathfrak{A} = (\mathcal{A}, Q, I, F, T)$; on pose $Q' = \{q \in Q \mid q \text{ n'est pas accessible ou co-accessible}\}$, on pose alors $\mathfrak{B} = (\mathcal{A}, Q\backslash Q', I\backslash Q', F\backslash Q', T\backslash(Q' \times \mathcal{A} \times Q \cup Q \times \mathcal{A} \times Q'))$: moins formellement, il s'agit de l'automate $\mathfrak{A}$ dont on a retiré les états Q' et les transitions associées.
Soit $m \in \mathcal{L}(\mathfrak{A})$, alors il existe $q_0 \xrightarrow{m_0} q_1 \xrightarrow{m_1} dots \xrightarrow{m_{|m|-1}} q_{|m|}$ un calcul acceptant m dans $\mathfrak{A}$. Alors, pour tout $j \in [\![0, |m|]\!]$, $q_0 \xrightarrow{?}{}^* q_j$ et $q_j \xrightarrow{?}{}^* q_{|m|}$ sont des chemins possibles dans $\mathfrak{A}$: donc ils sont tous accessibles et co-accessibles, donc $q_j \in Q\backslash Q'$. Donc, en particulier, toutes les transitions $q_j \xrightarrow{m_j} q_{j+1}$ sont dans $T\backslash(Q' \times \mathcal{A} \times Q \cup Q \times \mathcal{A} \times Q')$ (elles relient deux états accessibles et co-accessibles).
Réciproquement, si $m \in \mathcal{L}(\mathfrak{B})$, comme tous les états et transitions de $\mathfrak{B}$ sont aussi des états et transitions de $\mathfrak{A}$, le calcul acceptant m dans $\mathfrak{B}$ accepte aussi m dans $\mathfrak{A}$.
Pour déterminer Q', il suffit de faire un parcours en largeur de $\mathfrak{A}$ à partir des états initiaux pour obtenir les états accessibles; puis à faire un parcours en largeur « à l'envers » (en renversant les transitions) à partir des états finaux pour obtenir les états co-accessibles. Ces parcours se font en $O(|T|)$. En stockant ces résultats dans deux tableaux de booléens, on peut faire l'intersection en $O(|Q|)$. Une réponse paresseuse pour déterminer les transitions consiste à ne pas changer les transitions; sinon, comme les états de Q' sont stockés dans un tableau de booléens, c'est facile à établir.

L'émondation et la complétion sont, généralement, des opérations « opposées » : ajouter un état puits consiste à créer un état non co-accessibles.
L'intérêt de l'émondation est un gain en mémoire; l'intérêt de la complétion est plus théorique.

Considérons l'automate fini suivant, et émondons-le.



L'émondation consiste principalement à se demander : quels états sont vraiment utiles ?Un état à partir duquel on ne peut atteindre un état final est un état virtuellement mort : l'atteindre signifie que l'automate rejetera systématiquement le mot en cours de lecture.



5.2 Des expressions régulières vers les automates

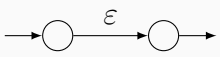
Dans cette section, on s'intéresse à partir d'une expression régulière et à construire un automate équivalent.

Algorithme 53 : algorithme de Thompson

Données : une expression régulière E

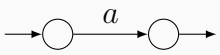
si $E = \varepsilon$ **:**

retourner



si $E = a$ *pour une lettre* $a \in \mathcal{A}$ **:**

retourner

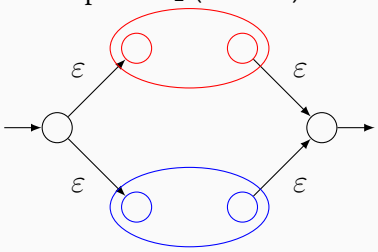


si $E = E_1 + E_2$ **:**

calculer un automate pour E_1 (en rouge)

calculer un automate pour E_2 (en bleu)

retourner

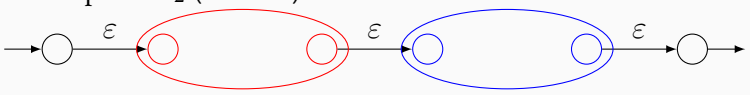


si $E = E_1 \cdot E_2$ **:**

calculer un automate pour E_1 (en rouge)

calculer un automate pour E_2 (en bleu)

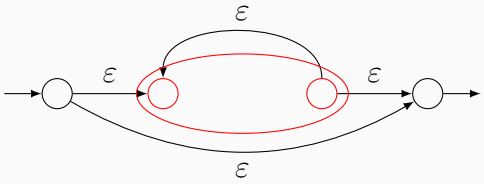
retourner



si $E = (E_1)^*$ **:**

calculer un automate pour E_1 (en rouge)

retourner



— L'automate construit contient de nombreuses ε-transitions : il faut les retirer à la fin si besoin est.

— L'automate construit a exactement un état initial et un état final ; on ne peut pas retourner à l'état initial ; on ne peut pas partir de l'état final. On peut alors remarquer que c'est le cas à chaque étape, et il est alors possible de légèrement améliorer les trois constructions inductives pour utiliser quelques états en moins.

— Les propriétés précédentes (un seul état initial, un seul final, transitions dans un sens) devraient caractériser ce qu'on appelle un **automate normalisé** : mais il faut aussi qu'il n'y ait pas d'ε-transitions pour un tel automate.

— On remarque qu'à chaque étape de construction de l'automate de Thompson, on ajoute deux états ; on en déduit par induction que pour une expression régulière E, l'automate de Thompson associé possède $2 \times |E|$ états. Par ailleurs, chaque état possède au plus deux arêtes sortantes, donc l'automate de Thompson a au plus $4 \times |E|$ transitions.

Pour l'algorithme suivant, nous devons introduire quelques notions intermédiaires.

Définition 54 : expression régulière linéaire

On dit qu'une expression régulière est **linéaire** si, dans son expression, chaque lettre n'apparaît au plus qu'une seule fois.

Pour $\mathcal{A} = \{a, b, c\}$, $a^* \cdot (b + c)$ est une expression linéaire, tandis que $(aa)^*$ ne l'est pas.

Toute expression régulière n'est pas « linéarisable » en tant que tel. Par exemple, pour $\mathcal{A} = \{a\}$, il n'existe que 5 expressions linéaires non équivalentes : $\varnothing$, ε, a, $a + \varepsilon$, a^*.

Définition 55 : langage local

Soit $L \subseteq \mathcal{A}^*$ un langage. On dit que L est un langage local s'il existe $P, D \subseteq \mathcal{A}$ et $F \subseteq \mathcal{A} \times \mathcal{A}$ tels que :

tout mot non vide de L commence par une lettre de P ;

tout mot non vide de L termine par une lettre de D ;

aucun facteur de longueur 2 de tous les mots non vides de L n'est dans F.

On dit que F est l'ensemble des facteurs interdits de L.

Un langage local est rationnel : on a en effet $L\backslash\{\varepsilon\} = (P\mathcal{A}^* \cap \mathcal{A}^*D)\backslash\mathcal{A}^*F\mathcal{A}^*$ (on rajoute ε selon).

Pour tout langage non vide L, on peut définir $P(L)$ l'ensemble des premières lettres de L, $D(L)$ et $F(L)$ l'ensemble des mots de deux lettres n'apparaissant pas comme facteurs de mots de L. Alors $L\backslash\{\varepsilon\} \subseteq (P\mathcal{A}^* \cap \mathcal{A}^*D)\backslash\mathcal{A}^*F\mathcal{A}^*$. L'inclusion n'est pas toujours réciproque : pour $L = \{abb, bba\}$, on a $P(L) = D(L) = \{a, b\}$ et $F(L) = \{aa, ba\}$; donc $bbb \in (P\mathcal{A}^* \cap \mathcal{A}^*D)\backslash\mathcal{A}^*F\mathcal{A}^*$, mais pourtant $bbb \notin L$. Un exemple infini : a^*ba^*.

Propriété 56 : expression régulière linéaire et langages locaux

Soit E une expression régulière linéaire. Alors $L(E)$ est un langage local.

Démonstration.

On démontre ce résultat par induction sur la forme de l'expression linéaire. On notera L le langage associé à E. Pour P, D et F trois ensembles convenables, on note $Loc(P, D, F)$ le langage local associé.

Initialisation : si $E = \varnothing$, $P(L) = \varnothing$ et $F(L) = \mathcal{A}^2$; donc $Loc(P, D, F)$ ne contient que des mots de longueur 1 sans lettre initiale : $Loc(P, D, F) = \varnothing = L$. Le même raisonnement tient pour $E = \varepsilon$.

Si $E = \{a\}$ pour une lettre $a \in \mathcal{A}$, alors $P(L) = D(L) = \{a\}$ et $F(L) = \mathcal{A}^2$; L est bien un langage local.

Induction : supposons que E se décompose en expression régulières plus petites. Alors ces sous-expressions sont aussi linéaires ; notre hypothèse d'induction est donc que ces expressions représentent des langages locaux. Il est aussi important de noter que ces expressions régulières s'écrivent sur des alphabets disjoints. Montrons alors que leur combinaison donne un langage local.

$\underline{E = E_1 + E_2}$: notons P_i, D_i et F_i les ensembles correspondant respectivement à E_1 et E_2, et $\mathcal{A}_1$ et $\mathcal{A}_2$ leurs alphabets. On pose alors $P = P_1 \cup P_2$, $F = D_1 \cup D_2$ et $F = F_1 \cup F_2 \cup (\mathcal{A}_1 \times \mathcal{A}_2)$. Soit $m \in L$ non vide : alors si $m \in L(E_1)$, m s'écrit sur $\mathcal{A}_1$; donc m ne peut contenir aucun facteur de $F_2 \subseteq \mathcal{A}_2^2$, ni de facteur de la forme $\mathcal{A}_1 \times \mathcal{A}_2$, donc $m \in Loc(P, D, F)$. Le même raisonnement symétrique tient pour $m \in L(E_2)$. Donc $L \subseteq Loc(P, D, F)$.

Soit $m \in Loc(P, D, F)$. Alors si m commence par une lettre de $\mathcal{A}_1$, comme $\mathcal{A}_1 \times \mathcal{A}_2 \subseteq F$, par récurrence immédiate, m ne contient que des lettres de $\mathcal{A}_1$. Donc sa dernière lettre appartient à D_1, et ses facteurs pas

à F_1 ; donc $m \in L(E_1)$. Le même raisonnement symétrique tient si la première lettre est dans $\mathcal{A}_2$. Donc $Loc(P, D, F) \in L$.
$\underline{E = E_1 \cdot E_2}$: Il y a techniquement 4 cas à traiter, selon que ε appartienne ou non à $L(E_1)$ et $L(E_2)$. On traite le cas où $\varepsilon \notin L(E_1)$ et $\varepsilon \in L(E_2)$, histoire de montrer les idées générales.
On pose $P = P_1$, $D = D_1 \cup D_2$, $F = F_1 \cup F_2 \cup (\mathcal{A}_2 \times \mathcal{A}_1) \cup ((\mathcal{A}_1 \times \mathcal{A}_2) \backslash (D_1 \times P_2))$. Soit $m \in L$ non vide : alors $m = m^{(1)} \cdot m^{(2)}$. On sait que $m^{(1)}$ est non vide : donc la première lettre est dans $P = P_1$. Il se pourrait que $m^{(2)}$ soit vide : donc la dernière lettre appartient à $D_1 \cup D_2 = D$. Pour les facteurs, ceux de $m^{(1)}$ ne peuvent être dans F_1, ni dans F_2 ou $\mathcal{A}_1 \times \mathcal{A}_2$ à cause des alphabets ; symétriquement pour $m^{(2)}$. La liaison entre les deux est, elle un élément de $D_1 \times P_2$, donc pas interdite. Donc $m \in Loc(P, D, F)$.
Soit $m \in Loc(P, D, F)$. Alors m commence par une lettre de P_1. Parce que $\mathcal{A}_2 \times \mathcal{A}_1 \subseteq F$, m contient d'abord des lettres de $\mathcal{A}_1$, puis des lettres de $\mathcal{A}_2$: notons ce découpage $m = m^{(1)} \cdot m^{(2)}$, où $m^{(1)}$ ne peut être vide.
Si $m^{(2)}$ est vide, alors m ne contient que des lettres de $\mathcal{A}_1$, les facteurs ne sont pas dans F_1 et le mot termine par une lettre de D_1, donc $m \in L(E_1) \subseteq L(E_1) \cdot L(E_2)$.
Si $m^{(2)}$ n'est pas vide, alors parce que $(\mathcal{A}_1 \times \mathcal{A}_2) \backslash (D_1 \times P_2) \subseteq F$, la transition entre $m^{(1)}$ et $m^{(2)}$ dans m est de la forme $D_1 \times P_2$. Donc $m^{(1)}$ commence par une lettre de P_1, termine par une lettre de D_1, et ne contient aucun facteur de F_1 ; de même pour $m^{(2)}$. Donc $m \in \subseteq L(E_1) \cdot L(E_2)$.
Donc $Loc(P, D, F) \subseteq L$.
$\underline{E = (E_1)^*}$: Posons $P = P_1$, $D = D_1$ et $F = F_1 \backslash (D_1 \times P_1)$.
Soit $m \in L$ non vide : alors $m = m^{(1)} \cdot \ldots m^{(n)}$ (pour un $n \geqslant 1$). Alors m commence par une lettre de P, termine par une lettre de D, et ses facteurs ne sont pas dans F (les transitions internes ne sont pas dans F_1, et les transitions externes sont dans $D_1 \times P_1$).
Soit $m \in Loc(P, D, F)$ non vide : alors m commence par une lettre de P_1 et termine par une lettre de D_1. Considérons successivement les facteurs de longueurs 2 de m : soit ils sont légaux pour E_1 (c'est-à-dire dans $\mathcal{A}^2 \backslash F_1$), soit ils sont dans $D_1 \times P_1$. On en déduit un découpage de m en $m^{(1)} \cdot \ldots \cdot m^{(n)}$ où chaque mot commence par une lettre de P_1 et termine par une lettre de D_1. Faisons ce découpage de sorte à ce qu'aucune transition interne n'ait la forme $D_1 \times P_1$. Alors aucune des transitions internes n'est dans F_1 : donc chaque $m^{(i)}$ est dans $L(E_1)$. Donc $m \in L(E_1)^*$: $Loc(P, D, F) \subseteq L$.

La réciproque est fausse : a^+ est un langage local, mais n'admet pas d'expression régulière linéaire.

Définition 57 : automate déterministe local

Soit $\mathfrak{A} = (\mathcal{A}, Q, i, F, \delta)$ un automate fini déterministe. On dit que $\mathfrak{A}$ est local si pour tout $a \in \mathcal{A}$, il existe un état $q_a \in Q$ tel que pour tout $q \in Q$, $\delta(q, a) = q_a$ (si tant est que $\delta(q, a)$ soit défini.)

Propriété 58 : langage locaux et automates locaux

Soit L un langage local. Alors il existe un automate déterministe local $\mathfrak{A}$ tel que $\mathcal{L}(\mathfrak{A}) = L$.

Démonstration.

Soit P, D et $\mathcal{F}$ définissant L (oups, collision de notations). On construit alors $\mathfrak{A}$ de la façon suivante :

$Q = \{q_a \mid a \in \mathcal{A}\} \cup \{i\}$;

l'état initial est i ;

$F = \{q_d \mid d \in D\}$;

pour $(a, b) \in \mathcal{A}^2$, $\delta(q_a, b)$ est défini ssi $ab \notin \mathcal{F}$, et vaut $\delta(q_a, b) = a$.

Il est alors trivial que $\mathcal{L}(\mathfrak{A}) = L$ (récurrence sur δ^*).

Algorithme 59 : algorithme de Berry-Sethi

Données : une expression régulière E
linéariser E
déterminer P, D et F définissant le langage local $\mathcal{L}(E)$
construire l'automate local associé
délinéariser l'automate obtenu

Dans le pire cas, cet algorithme nécessite $O(|E|^2)$ opérations élémentaires.

Démonstration.

La linéarisation de E se fait en $O(|E|)$: il suffit de parcourir l'arbre inductif de E en retenant les lettres vues dans un dictionnaire.
Le calcul de P, D et F se fait de manière récursive, en suivant les formules données dans la preuve précédente. On observe que dans le pire des cas, calculer P ou D revient à calculer une union et une intersection (pour vérifier si ε fait partie d'un langage lors de la concaténation) : donc le calcul de P et de D est en $O(|E|)$. En revanche, le calcul de F demande plus de travail : il faut en particulier faire des produits cartésiens. Un autre argument consiste alors à remarquer que dans le pire des cas, $F = \mathcal{A}_E^2$, où $\mathcal{A}_E$ est l'alphabet de E après linéarisation ; que $|\mathcal{A}_E| \leqslant |E|$; et donc $|F| = O(|E|^2)$.
L'automate local associé possède $O(|\mathcal{A}_E|)$ donc $O(|E|)$ états, $O(|E|^2)$ transitions ; le construire demande simplement à vérifier F, donc en $O(|E|^2)$ opérations élémentaires.
La délinérisation se fait en $O(|E|)$ opérations.

L'automate obtenu à la fin de l'algorithme de Berry-Sethi est appelé **automate de Glushkov de** E.

5.3 Déterminisation d'un automate fini

Passer d'un automate fini non-déterministe à un automate équivalent déterministe est une opération algorithmique vitale en théorie des langages formels. Rappelons-en l'algorithme le plus simple, qui construit l'**automate des parties** :

Algorithme 60 : construction de l'automate des parties

Données : un automate fini $\mathfrak{A} = (\mathcal{A}, Q, I, F, T)$
$NQ \leftarrow \mathcal{P}(Q)$
$ni \leftarrow I$
$NF \leftarrow \{E \subseteq Q \mid F \cap E \neq \varnothing\}$
$NT \leftarrow \varnothing$
pour *tout état* $E \in NQ$ *et lettre* $a \in \mathcal{A}$ **:**
  $E' \leftarrow \{q \in Q \mid \exists p \in E, p \xrightarrow{a} q \in T\}$
  ajouter à NT la transition $E \xrightarrow{a} E'$
retourner *l'automate* $\mathfrak{B} = (\mathcal{A}, NQ, ni, NF, NT)$

L'automate déterministe obtenu contient $2^{|Q|}$ états.

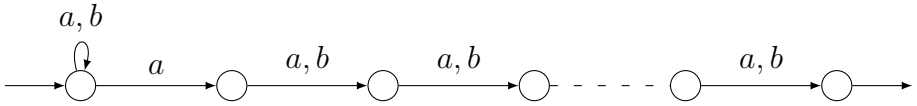
L'automate $\mathfrak{A}$ initial n'a pas besoin d'être complet ou émondé ; l'automate obtenu $\mathfrak{B}$ est complet (mais pas forcément émondé).

Propriété 61 : la borne est mauvaise, mais on n'a pas le choix

Pour tout $n \in \mathbb{N}^*$, il existe un automate $\mathfrak{A}_n$ non-déterministe à $n+1$ états tel que tout automate déterministe $\mathfrak{B}_n$ équivalent admet au moins 2^n états.

Démonstration.

Soit $n \in \mathbb{N}^*$, considérons le langage $L_n = (a+b)^* a (a+b)^{n-1}$. Alors L_n est reconnu par l'automate fini non-déterministe $\mathfrak{A}_n$ suivant :



Cet automate contient $n+1$ états. Il n'est pas déterministe, car le premier état admet deux arêtes étiquetées par a.
Soit $\mathfrak{B} = (\mathcal{A}, Q, i, F, \delta)$ un automate déterministe acceptant L_n. Nous allons montrer que $\mathfrak{B}$ possède au moins 2^n états. On commence par décrire le raisonnement sur un cas simple avant de présenter le « vrai » raisonnement.
Supposons que $\delta(i,a) = \delta(i,b)$. Alors on aurait :

$$\begin{aligned}\delta^*(i, ba^{n-1}) &= \delta^*(\delta(i,b), a^{n-1}) \\ &= \delta^*(\delta(i,a), a^{n-1}) = \delta^*(i, a^n)\end{aligned}$$

Or, ce dernier mot est accepté ; donc $\mathfrak{B}$ accepterait aussi ba^{n-1}, ce qui est absurde.
De manière générale : soient x et $y \in \mathcal{A}^n$, et supposons que $\delta^*(i,x) = \delta^*(i,y)$. Soit s le plus long suffixe commun de x et de y (comme $x \neq y, s \neq x$) : on a $x = x's$ et $y = y's$. Quitte à symétriser, on peut supposer même que $x = x''as$ et $y = y''bs$. On sait que $|s| < |x| = n$, donc $|s| \leqslant n-1$. Donc $x \cdot a^{n-1-|s|} \in L_n$, mais $y \cdot a^{n-1-|s|} \notin L_n$; mais si $\delta^*(i,x) = \delta^*(i,y)$, alors $\delta^*(\delta^*(i,x), a^{n-1-|s|}) = \delta^*(\delta^*(i,y), a^{n-1-|s|})$, ce qui est absurde.
Donc tous les $\delta^*(i,x)$ sont distincts pour tout $x \in \mathcal{A}^n$. Or, $|\mathcal{A}^n| = 2^n$; donc $\mathfrak{B}$ possède au moins 2^n états.

La déterminisation d'un automate n'est pas une opération à faire à la légère. Par exemple, si l'objectif est de résoudre un problème d'acceptation du mot, déterminiser avant d'utiliser l'algorithme linéaire est généralement plus coûteux que la déterminisation à la volée.

5.4 Problème du non-vide et conséquences

Propriété 62

Le problème du non-vide est décidable.
Donnée : un automate fini $\mathfrak{A}$
Question : est-ce que $\mathcal{L}(\mathfrak{A}) \neq \varnothing$?

Démonstration.

Il suffit de faire un parcours à partir des états initiaux de $\mathfrak{A}$: si au moins un des états finaux est accessible, $\mathfrak{A}$ accepte au moins un mot.

Propriété 63 : un arsenal de problèmes à résoudre

Tous les problèmes suivants sont décidables :

problème du vide :
Donnée : un automate fini $\mathfrak{A}$
Question : est-ce que $\mathcal{L}(\mathfrak{A}) = \varnothing$?

problème de l'inclusion :
Donnée : deux automates finis $\mathfrak{A}$ et $\mathfrak{B}$
Question : est-ce que $\mathcal{L}(\mathfrak{A}) \subseteq \mathcal{L}(\mathfrak{B})$?

problème de l'égalité :
Donnée : deux automates finis $\mathfrak{A}$ et $\mathfrak{B}$
Question : est-ce que $\mathcal{L}(\mathfrak{A}) = \mathcal{L}(\mathfrak{B})$?

problème de l'universalité :
Donnée : un automate fini $\mathfrak{A}$
Question : est-ce que $\mathcal{L}(\mathfrak{A}) = \mathcal{A}^*$?

Démonstration.

Le problème du vide est le contraire du problème du non-vide, qui est décidable.

Il suffit de remarquer que $\mathcal{L}(\mathfrak{A}) \subseteq \mathcal{L}(\mathfrak{B})$ ssi $\mathcal{L}(\mathfrak{A}) \cap \overline{\mathcal{L}(\mathfrak{B})} = \varnothing$: il est possible, par les constructions vues précédemment, de construire un automate fini reconnaissant le complémentaire et l'intersection. On applique ensuite le problème du vide.

Il suffit de remarquer que $\mathcal{L}(\mathfrak{A}) = \mathcal{L}(\mathfrak{B})$ ssi $\mathcal{L}(\mathfrak{A}) \subseteq \mathcal{L}(\mathfrak{B})$ et $\mathcal{L}(\mathfrak{B}) \subseteq \mathcal{L}(\mathfrak{A})$.

Pour le problème de l'universalité : on peut se ramener au problème de l'égalité (c'est théoriquement vrai, en pratique très peu efficace).

6 Lemme de l'étoile

Le modèle des langages rationnels (ou réguliers ou reconnaissables), à la suite de ce cours, peut paraître comme un modèle magique : il est stable par toute opération raisonnable, connaît de nombreuses extensions (automate fini généralisé, ε-transitions, ...) : il est pourtant nécessaire de comprendre qu'il s'agit d'un modèle *limité*. Tout langage n'est pas forcément rationnel.

Théorème 64

$\text{Rat}(\mathcal{A}^*) \neq \mathcal{P}(\mathcal{A}^*)$.

Advient alors la question : comment faire pour distinguer les langages rationnels de ceux qui ne le sont pas ? La réponse précise est un peu compliquée, mais il existe un critère tout de même très utile.

Lemme 65 : lemme de l'étoile

Soit $L \subseteq \mathcal{A}^*$ un langage. Si L est rationnel, alors il existe une constante $N \in \mathbb{N}^*$ telle que pour tout mot $m \in L$, il existe un découpage $m = xyz$ vérifiant les conditions suivantes :

$y \neq \varepsilon$;
$|xy| \leqslant N$;
pour tout $k \in \mathbb{N}$ (y compris 0), $xy^k z \in L$.

Soit $L = \mathcal{L}(ab((c+ab)bb)^*a)$ et j'affirme qu'on pose $N = 6$. Soit $m = abcbba$. Alors on remarque qu'en posant $x = ab$, $y = cbb$ et $z = a$, on a :

— $y \neq \varepsilon$;
— $|xy| \leqslant 6$;
— pour tout $k \in \mathbb{N}$, $ab(cbb)^k a \in L$.

Démonstration.

Soit L un langage rationnel, alors il existe un automate fini $\mathfrak{A}$ tel que $\mathcal{L}(\mathfrak{A}) = L$, avec $\mathfrak{A} = (\mathcal{A}, Q, I, F, T)$. Posons alors $N = |Q|$.
Soit $m \in L$ tel que $|m| \geqslant N$: considérons alors un calcul $q_0 \xrightarrow{m_0} q_1 \xrightarrow{m_1} q_2 \xrightarrow{m_2} \dots \xrightarrow{m_{|m|-1}} q_{|m|}$ acceptant m dans $\mathfrak{A}$. Alors on remarque que dans ce calcul, il y a $|m|+1 \geqslant |Q|+1$ états ; par principe des tiroirs, il existe donc un état q apparaissant deux fois dans ce calcul acceptant. Plus encore : en considérant les $N+1$ premiers états du calcul, on a le même raisonnement, donc on peut supposer que cet état apparaissant deux fois apparaît dans les $N+1$ premiers états du calcul.
Découpons le calcul selon l'état q : on a alors $q_0 \xrightarrow{x}{}^* q \xrightarrow{y}{}^* q \xrightarrow{z}{}^* q_{|m|}$. On a alors que $y \neq \varepsilon$, que $|xy| \leqslant N$.
On en déduit enfin que $q_0 \xrightarrow{x}{}^* q \xrightarrow{z}{}^* q_{|m|}$ est un calcul acceptant dans $\mathfrak{A}$, ainsi que, pour tout $k \geqslant 2$, $q_0 \xrightarrow{x}{}^* q \xrightarrow{y^k}{}^* q \xrightarrow{z}{}^* q_{|m|}$.

Ce lemme, qui peut sembler accessoire, est le principal outil utilisé pour montrer qu'un langage n'est pas rationnel.

Propriété 66

$\{a^n b^n \mid n \in \mathbb{N}\} \notin \text{Rat}(\{a,b\}^*)$.

Démonstration.

Soit $L = \{a^n b^n \mid n \in \mathbb{N}\}$; supposons, par l'absurde que L est rationnel. D'après le lemme de l'étoile, il existe $N \in \mathbb{N}^*$ tel que pour tout $m \in L$, on a les bonnes propriétés.
Considérons alors $m = a^N b^N$. Le lemme de l'étoile nous donne alors un découpage $m = xyz$ avec $|xy| \leqslant N$;

donc $xy \sqsubseteq_p a^N$ (pour des questions de longueur). Donc, comme y est non vide, on a $y = a^k$ pour un certain $k \geqslant 1$. D'après le lemme de l'étoile, on a $xy^2z \in L$; mais pourtant, $xy^2z = a^{N+k}b^N \notin L$. Absurde !

Attention, le lemme de l'étoile est une condition nécessaire, mais pas suffisante : il existe des langages non-rationnels qui vérifient les conditions du lemme de l'étoile.

Propriété 67 : le lemme de l'étoile n'est pas une condition suffisante

Il existe des langages $L \subseteq \mathcal{A}^*$ tels que $L \notin \mathrm{Rat}(\mathcal{A}^*)$, mais vérifiant les conditions du lemme de l'étoile.

Démonstration.

Contre-exemple récupéré par Valia Mitsou.

Soit $\mathcal{A} = \{a, b, c\}$, et considérons les deux langages suivants :

$L_0 = \{a^i b^j c^k \mid i, j, k \geqslant 0 \text{ et si } i = 1, \text{ alors } j = k\}$;

$L_1 = \{ab^j c^j \mid j \in \mathbb{N}\}$;

$L_2 = \mathcal{L}(ab^*c^*)$

Alors on remarque que :

L_0 n'est pas rationnel : en effet, s'il l'était, alors comme $L_1 = L_0 \cap L_2$, l'intersection de rationnels étant rationnelle, L_1 serait rationnel, ce qu'il n'est pas (on peut le prouver par lemme de l'étoile). Donc L_0 n'est pas rationnel.

L_0 peut être pompé : posons $N = 2$, et soit $m \in L_0$, avec $m = a^i b^j c^k$ de longueur suffisante.

si $i = 0$ et $j = 0$, on pose $x = \varepsilon$, $y = c$ et $z = c^{k-1}$; alors $xy^K z = c^{k-1+K} \in L_0$.

si $i = 0$ et $j \geqslant 1$, on pose $x = \varepsilon$, $y = b$ et z le reste ; alors $xy^K z = b^{j-1+K}c^k \in L_0$.

si $i = 1$, alors $j \geqslant 1$ (sinon le mot n'est pas assez long), on pose $x = \varepsilon$, $y = a$ et z le reste du mot. Alors $xy^K z = a^K b^j c^j \in L_0$.

si $i = 2$, on pose $x = \varepsilon$, $y = aa$ et z le reste ; alors $(aa)^K b^j c^k \in L_0$.

si $i \geqslant 3$, on pose $x = \varepsilon$, $y = a$ et z le reste du mot. Alors $a^{i-1+K}b^j c^k \in L_0$.