2
Les langages réguliers et les automates finis.

Ce chapitre est consacré à l'étude des *langages réguliers* (aussi qualifiés de *rationnels*) qui jouent un grand rôle dans la théorie des langages, et à celle des *automates finis* qui permettent de les reconnaître. En compilation, la définition des langages réguliers comme interprétation des *expressions régulières* constitue la "spécification lexicale", les automates finis, quant à eux, sont essentiels pour l'*analyse lexicale* mais ils jouent aussi un rôle dans les méthodes d'*analyse syntaxique* que nous étudierons plus tard.

Dans toute la suite

les alphabets sont supposés finis

et cette hypothèse joue un rôle essentiel.

1 – Les langages réguliers.

On appelle *opérations régulières*, les trois opérations suivantes sur les langages :

- réunion finie,
- concaténation,
- itération.

On peut donner la définition des langages réguliers de la façon suivante :

Les langages réguliers

Un langage est dit régulier ssi on peut le construire, à partir de langages finis, par un nombre fini d'applications d'opérations régulières.

Mais nous allons reformuler cette définition de façon plus formelle avant de donner des exemples.

Techniquement, il est intéressant de considérer un langage régulier comme l'interprétation d'une *expression régulière*; pour écrire ces expressions, nous utilisons les symboles suivants :

$$\emptyset, \text{ les } x \in \mathcal{A}, +, \cdot, ^* \text{ et les parenthèses (et).}$$

Les expressions régulières

$EReg(\mathcal{A})$ est l'ensemble des expressions définies par application des clauses inductives suivantes :

(a) $x \in EReg(\mathcal{A})$ pour tout $x \in \mathcal{A}$,

(b) $\emptyset \in EReg(\mathcal{A})$,

(c) si $\alpha \in EReg(\mathcal{A})$ et $\beta \in EReg(\mathcal{A})$ alors $(\alpha + \beta) \in EReg(\mathcal{A})$,

(d) si $\alpha \in EReg(\mathcal{A})$ et $\beta \in EReg(\mathcal{A})$ alors $(\alpha \cdot \beta) \in EReg(\mathcal{A})$,

(e) si $\alpha \in EReg(\mathcal{A})$ alors $\alpha^* \in EReg(\mathcal{A})$.

Les parenthèses déterminent entièrement la façon dont une expression régulière a été construite à partir du symbole $\emptyset$ et des éléments de $\mathcal{A}$. Ceci signifie qu'une expression régulière admet une *lecture unique*, ce qui permet de raisonner par **induction** sur l'ensemble $EReg(\mathcal{A})$. La première application de ce principe d'induction est la définition d'une interprétation $I : EReg(\mathcal{A}) \to \mathcal{P}(\mathcal{A}^*)$ des expressions régulières par des langages.

I est définie par les clauses inductives suivantes :

(a) $I(x) = x$ pour tout $x \in \mathcal{A}$, (langage dont le seul élément est le mot à une seule lettre x)
(b) $I(\emptyset) = \emptyset$, (langage vide)
(c) $I((\alpha + \beta)) = I(\alpha) + I(\beta)$, (réunion de langages)
(d) $I((\alpha \cdot \beta)) = I(\alpha)I(\beta)$, (concaténation de langages)
(e) $I(\alpha^*) = I(\alpha)^*$. (itération d'un langage)

Les langages réguliers

$L \subseteq \mathcal{A}^*$ est un langage régulier sur $\mathcal{A}$ ssi il existe une expression régulière $\alpha \in EReg(\mathcal{A})$ telle que $L = I(\alpha)$; on dira alors que α est une expression régulière de L.

Les propriétés de la somme, de la concaténation et de l'itération (cf. chapitre 1) montrent qu'en fait, chaque langage régulier admet une infinité s'expressions régulières!

La définition des langages réguliers permet de caractériser l'ensemble qu'ils forment d'une façon plus directe :

Propriété des langages réguliers

L'ensemble $Reg(\mathcal{A})$ des langages réguliers sur $\mathcal{A}$ est le plus petit qui vérifie :

a) $x \in Reg(\mathcal{A})$ pour tout $x \in \mathcal{A}$,
b) $\emptyset \in Reg(\mathcal{A})$,
c) si $L \in Reg(\mathcal{A})$ et $M \in Reg(\mathcal{A})$ alors $L + M \in Reg(\mathcal{A})$,
d) si $L \in Reg(\mathcal{A})$ et $M \in Reg(\mathcal{A})$ alors $LM \in Reg(\mathcal{A})$,
e) si $L \in Reg(\mathcal{A})$ alors $L^* \in Reg(\mathcal{A})$.

Une expression régulière est, littéralement, une présentation formelle d'un langage régulier : elle définit *la forme* (ou *le motif*) qui sert de modèle aux mots du langage en question.

Il y a peu de différence entre une expression régulière et son interprétation; il est courant de faire des inductions, soi–disant sur les langages réguliers : en fait, il s'agit bien d'inductions sur les expressions régulières qui sont représentées par leur interprétation ...
De même, il est courant de négliger certaines parenthèses dans l'écriture d'une expression régulière : celles qui, compte tenu des conventions habituelles pour en réinstaller, ne peuvent pas en modifier l'interprétation.

Exemples de langages réguliers et d'expressions régulières.

- Le langage vide $\emptyset \subseteq \mathcal{A}^*$ est régulier.
- Le langage $\varepsilon \subseteq \mathcal{A}^*$, interprétation de l'expression régulière $\emptyset^*$ est régulier. On utilise souvent ε pour désigner l'expression régulière $\emptyset^*$ elle–même.
- Un langage réduit à un seul mot est régulier. En particulier tout $x \in \mathcal{A}$ est un langage régulier.
- Tout langage fini est la réunion d'une famille finie de langages à un seul mot : c'est pourquoi un langage fini est régulier.
- Rappelons que nous ne considérons que des alphabets $\mathcal{A}$ finis : ceci sert à assurer que $\mathcal{A} \subseteq \mathcal{A}^*$ lui–même est un langage régulier.

• En appliquant la clôture de $Reg(\mathcal{A})$ par itération, on déduit du point précédent que $\mathcal{A}^* \subseteq \mathcal{A}^*$ est régulier.

• Pour toute expression régulière α et tout entier n, on peut définir α^n par récurrence sur n : le résultat est une façon d'écrire une expression régulière.

• On peut évidemment donner une expression régulière de langages plus particuliers que ceux qui précèdent. Par exemple, pour l'alphabet $\mathcal{A} = \{a, b\}$ comportant deux symboles distincts :

- $\emptyset$, ε, a, b, $\mathcal{A} = a + b$ et $\mathcal{A}^* = (a + b)^*$ définissent des langages réguliers déjà décrits,
- $\mathcal{A}^n = (a + b)^n$ est une expression régulière de l'ensemble des mots de longueur n,
- $(\varepsilon + \mathcal{A})^n = (\varepsilon + a + b)^n$ est une expression régulière de l'ensemble des mots de longueur $\leq n$,
- $\mathcal{A}^m(\varepsilon + \mathcal{A})^n = (a + b)^m(\varepsilon + a + b)^n$ est une expression régulière de l'ensemble des mots dont la longueur est comprise entre m et $m + n$,
- enfin, voici un exemple plus spécial : $((\varepsilon + a)b)^*(\varepsilon + a)$ est une expression régulière du langage formé des mots qui ne comportent pas le facteur aa (cf. exercice 1).

Ce dernier exemple est un peu plus délicat à justifier que ceux qui le précèdent : les automates que nous étudierons par la suite permettent de calculer des expressions régulières de façon plus systématique (en l'occurrence, cf. les exercices 8 et 10).

Attention.

• Il existe des langages sur $\mathcal{A}$ qui ne sont pas réguliers, dès que $\mathcal{A} \neq \emptyset$ (cf. l'annexe et l'exercice 5).

On peut construire des langages qui sont réguliers, mais pour lesquels on ne peut pas trouver effectivement d'expression régulière (cf. section 6.2.2).

1.1 – Equations linéaires à coefficients réguliers.

Nous avons vu, au chapitre 1, que la plus petite solution d'un système d'équations linéaires se calcule à partir de ses coefficients, par application d'opérations régulières : or, ces opérations transforment des langages réguliers en des langages réguliers. On a donc :

Propriété

La plus petite solution d'un système d'équations linéaires à coefficients réguliers est constituée de langages réguliers.

De plus, on peut remarquer que le calcul de cette solution (par la "méthode de Gauss") est effectif, lorsque le système est fini, comme dans la section 4.2 du chapitre 1.

Un cas important est celui où les coefficients sont des langages finis : c'est lui qui nous sera utile dans la démonstration du Théorème de Kleene (cf. section 3.2).

2 – Automates déterministes et complets (ADC).

Nous allons d'abord donner une définition relative à un monoïde quelconque, puis, nous la spécialiserons au monoïde $(\mathcal{A}^*, \cdot, \varepsilon)$ des mots sur un alphabet fini $\mathcal{A}$.

Action d'un monoïde

Soient Q un ensemble et $(D, \times, e)$ un monoïde.
Une action de $(D, \times, e)$ *sur* Q est une application $\bullet : Q \times D \to Q$ qui vérifie :

$$q \bullet e = q$$
$$q \bullet (x \times y) = (q \bullet x) \bullet y$$

quels que soient $q \in Q$, $x \in D$ et $y \in D$.

Remarques.

Les éléments de l'ensemble Q s'appellent souvent des *états*. Par ailleurs, il est intéressant de considérer Q comme un alphabet : les états sont alors des symboles. Dans ce qui suit, ces symboles sont souvent des entiers naturels avec lesquels il faut prendre les précautions d'usage (cf la section 1.1 du chapitre 1). Nous appliquerons à Q toutes les conventions faites à propos des alphabets, par exemple, $Q = 0+1+2+3+4$ représente $\{0\}+\{1\}+\{2\}+\{3\}+\{4\} = \{0,1,2,3,4\}$ et non pas la somme, au sens arithmétique du terme!

Exemple.

Lorsque l'on prend $Q = D$, on a une action naturelle $q \bullet x = q \times x$. En effet, il suffit d'appliquer la définition d'un monoïde pour écrire : $q \bullet e = q \times e = q$ et $q \bullet (x \times y) = q \times (x \times y) = (q \times x) \times y = (q \bullet x) \bullet y$.

Nous n'utilisons que les actions du monoïde $(\mathcal{A}^*, \cdot, \varepsilon)$ des mots sur un alphabet fini $\mathcal{A}$.

Propriété principale

Toute application $\bullet : Q \times \mathcal{A} \to Q$ *s'étend de façon unique en une action* $\dot{\bullet} : Q \times \mathcal{A}^* \to Q$.

Cette propriété est une proche parente de la propriété principale de $\mathcal{A}^*$ énoncée dans la section 5.2 du premier chapitre. Contentons–nous de dire que l'action $\dot{\bullet}$ peut se définir de la façon suivante, par récurrence :

1) $q \dot{\bullet} \varepsilon = q$
2) $q \dot{\bullet} ux = (q \dot{\bullet} u) \bullet x$.

Vocabulaire et convention.

Nous dirons que l'application $\dot{\bullet}$ est *l'action définie par* $\bullet$ et nous la noterons simplement $\bullet$: la propriété principale nous y autorise.

Avec cette convention, la seconde condition que doit vérifier $\bullet$ pour être une action est

$$q \bullet (uv) = (q \bullet u) \bullet v$$

pour tout $u, v \in \mathcal{A}^*$. La vérification de cette propriété est un peu longue (voyez la démonstration analogue qui a été faite pour la propriété principale de $\mathcal{A}^*$).

La récurrence définissant l'action correspond à une procédure récursive. Par exemple :

$$\begin{aligned} q \bullet abba &= (q \bullet abb) \bullet a && \text{(par 2)} \\ &= ((q \bullet ab) \bullet b) \bullet a && \text{(par 2)} \\ &= (((q \bullet a) \bullet b) \bullet b) \bullet a && \text{(par 2)} \\ &= ((((q \bullet \varepsilon) \bullet a) \bullet b) \bullet b) \bullet a && \text{(par 2)} \\ &= (((q \bullet a) \bullet b) \bullet b) \bullet a && \text{(par 1)} \end{aligned}$$

Les égalités justifiées par 2) sont des appels récursifs. Dans la dernière expression, tous les $\bullet$ ont enfin leur sens d'origine : il ne reste plus qu'à effectuer les calculs.

On éviterait les appels récursifs en remplaçant 2) par $q \bullet xu = (q \bullet x) \bullet u$ utilisable lorsque l'on construit les mots par adjonction d'occurrences à gauche : ceci est évidemment correct et peut quelquefois se montrer intéressant.

Exemple.

Lorsque l'on fait agir le monoïde $(\mathcal{A}^*, \cdot, \varepsilon)$ sur l'ensemble $\mathcal{A}^*$ lui–même (cf. l'exemple précédent), l'action naturelle est la concaténation $\mathcal{A}^* \times \mathcal{A}^* \to \mathcal{A}^*$ dont la restriction $\mathcal{A}^* \times \mathcal{A} \to \mathcal{A}^*$ est simplement l'opération d'adjonction d'une occurrence à droite!

La propriété principale est importante car elle permet de donner de bonnes représentations d'une action $\bullet$ de $(\mathcal{A}^*, \cdot, \varepsilon)$ sur un ensemble Q :

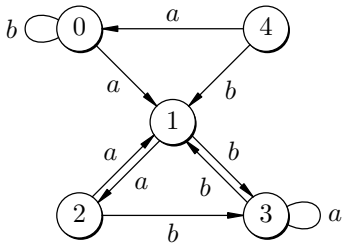
– L'action $\bullet$ peut se définir par *une table* sur laquelle on reporte la valeur de $q \bullet x$ à l'intersection de la ligne $q \in Q$ et de la colonne $x \in \mathcal{A}$: cette table est finie lorsque Q l'est (ce que nous supposerons définitivement un peu plus bas).

– *Le graphe de transition* est une représentation géométrique dont la lecture est plus immédiate (du moins, lorsque Q et $\mathcal{A}$ n'ont pas trop d'éléments). Ce graphe est constitué de nœuds et d'arêtes étiquetés :
  – un nœud (q) pour chaque $q \in Q$;
  – une arête $(q) \xrightarrow{x} (r)$ pour chaque $q \in Q$, chaque $x \in \mathcal{A}$ et $r = q \bullet x$.

Exemple 1.

On prend $\mathcal{A} = a + b$ et $Q = 0 + 1 + 2 + 3 + 4$.

q	$q \bullet a$	$a \bullet b$
0	1	0
1	2	3
2	1	3
3	3	1
4	0	1



Remarquez qu'il n'est pas nécessaire de préciser l'orientation d'une arête dont l'origine et l'extrémité sont confondues. Une autre convention consiste à coller plusieurs étiquettes sur une arête pour en représenter plusieurs de même origine et même extrémité, par exemple $(q) \xrightarrow{x,y} (r)$ représente le couple d'arêtes $(q) \underset{y}{\overset{x}{\rightrightarrows}} (r)$.

Chemins dans un graphe de transition.

Choisissons un état $q \in Q$. Chaque $u \in \mathcal{A}^*$ détermine un chemin unique $ch(q,u) \in Chem(q, q \bullet u)$ (cf. la section 7 du chapitre 1) que l'on peut définir pour tout u par la récurrence suivante :

– $ch(q,\varepsilon) = q$,
– soit $ch(q,u) = \chi r$ avec $\chi \in Q^*$ et $r = q \bullet u$, et soit $s = r \bullet x$ alors

$$ch(q,ux) = ch(q,u) \circ rs = ch(q,u)s.$$

Si, dans l'exemple précédent, on choisit $q = 0$, on a $ch(0,abab) = 01331$: la première lettre du mot sert à la fois à déterminer l'arête à parcourir et à payer ce parcours; le mot se trouve raccourci et on peut ainsi poursuivre son chemin jusqu'à épuisement de ses lettres.

Une façon commode pour sélectionner des mots est, un point de départ étant fixé, de leur demander d'atteindre l'un des points d'arrivée, eux aussi fixés à l'avance : cette épreuve se passe dans un objet du type suivant.

Les ADC

Un automate déterministe et complet (en abrégé ADC) est la donnée d'un 5–uplet $\mathbf{A} = (Q, \mathcal{A}, \bullet, q_0, F)$ où :

– Q est un ensemble, (d'états)
– $\mathcal{A}$ est un alphabet fini,
– $\bullet : Q \times \mathcal{A} \to Q$ est une application, (ce qui définit une action)
– $q_0 \in Q$ est l'entrée, (ou état initial)
– $F \subseteq Q$ est l'ensemble des sorties. (ou états finaux)

Commentaires.

- $q \bullet x$ est une abréviation pour $q \underset{\mathbf{A}}{\bullet} x$; lorsqu'un second automate $\mathbf{B}$ sera en cause dans une même question, on pourra utiliser l'abréviation $q \circ x$ pour $q \underset{\mathbf{B}}{\bullet} x$, mais, lorsque l'action de $\mathbf{B}$ est une extension (ou une restriction) de celle de $\mathbf{A}$ et qu'une confusion n'est pas à craindre, on préférera conserver la notation $q \bullet x$.
- Les qualificatifs "complet" et "déterministe" viennent du fait qu'à chaque mot u correspond un et un seul chemin partant d'un état donné q dans le graphe de transition : ceci sera précisé par la suite.

Exemple 1 (suite).

On peut compléter l'exemple 1 de telle façon à définir un ADC, par exemple en choisissant l'entrée $q_0 = 0$ et la seule sortie $F = 2$. Voici la table de cet ADC : elle comporte une colonne pour les informations d'entrée ($\rightarrow$) et sorties ($\leftarrow$).

e/s	q	$q \bullet a$	$q \bullet b$
$\rightarrow$	0	1	0
	1	2	3
$\leftarrow$	2	1	3
	3	3	1
	4	0	1

Langage reconnu par un ADC

Le langage reconnu par un ADC $\mathbf{A}$ sur $\mathcal{A}$ est $\mathcal{L}(\mathbf{A}) \subseteq \mathcal{A}^*$ défini par : "$u \in \mathcal{L}(\mathbf{A})$ ssi $q_0 \bullet u \in F$".
Deux ADC $\mathbf{A}$ et $\mathbf{A}'$ sont dits *équivalents* ssi $\mathcal{L}(\mathbf{A}) = \mathcal{L}(\mathbf{A}')$.

$\mathcal{L}(\mathbf{A})$ est donc l'ensemble des mots qui définissent des chemins partant de l'entrée q_0 et aboutissant à l'une des sorties.

2.1 – Etats accessibles d'un ADC.

Il est très clair que, dans l'ADC de l'exemple 1, aucun chemin partant de $q_0 = 0$ ne passera par l'état 4 : cette observation conduit naturellement à poser la définition suivante.

Les états accessibles

$q \in Q$ est *accessible* ssi il existe $u \in \mathcal{A}^*$ tel que $q = q_0 \bullet u$.

Désignons par $\mathcal{A}cc$ l'ensemble des états accessible de $\mathbf{A}$. Il est clair que $\mathcal{A}cc$ est le plus petit ensemble qui vérifie :

- $q_0 \in \mathcal{A}cc$,
- pour tout $q \in \mathcal{A}cc$ et tout $x \in \mathcal{A}$: $q \bullet x \in \mathcal{A}cc$.

Cette observation est suffisante pour construire l'ADC $\mathbf{A}' = (Q', \mathcal{A}, \bullet, q_0, F')$ suivant :

- *Etats.* $Q' = \mathcal{A}cc$, (l'ensemble des états accessibles dans $\mathbf{A}$)
- *Entrée.* q_0, (état initial de $\mathbf{A}$)
- *Action.* $\bullet : Q' \times \mathcal{A} \rightarrow Q'$, (la restriction de $\bullet$ à Q')
- *Sortie.* $F' = F \cap Q'$, (sorties accessibles de $\mathbf{A}$).

Par construction, tous les états de $\mathbf{A}'$ sont accessibles à partir de q_0 et, comme tout chemin partant de q_0 ne passe que par des états accessibles, on a $\mathcal{L}(\mathbf{A}') = \mathcal{L}(\mathbf{A})$.

Propriété

Tout ADC est équivalent à un ADC dont tous les états sont accessibles.

L'ADC $\mathbf{A}'$ ainsi construit s'appelle *la partie accessible de* $\mathbf{A}$: c'est la seule partie utile lorsque l'on ne s'intéresse qu'à $\mathcal{L}(\mathbf{A})$.

3 – Automates finis déterministes et complets (AFDC).

Si l'on n'impose aucune condition supplémentaire, tout $L \subseteq \mathcal{A}^*$ est reconnu par un ADC approprié (cf. l'annexe). La condition qui est utile à notre problème est

l'ensemble Q des états est fini

et nous la supposons vérifiée dans toute la suite de ce chapitre : un ADC dont l'ensemble des états est fini est appelé un *automate fini déterministe et complet*, ce que l'on abrègera en AFDC. Rappelons que l'alphabet $\mathcal{A}$, lui-même, est supposé fini, ce qui assure que le nombre des arêtes d'un AFDC est fini.

3.1 – Etats accessibles d'un AFDC.

Les définitions données dans la section précédente sur les ADC s'adaptent au cas fini : langage reconnu par un AFDC, AFDC équivalents, états accessibles. Par exemple, on a la propriété :

Partie accessible d'un AFDC

Tout AFDC est équivalent à un AFDC dont tous les états sont accessibles.

Le calcul de $\mathcal{A}cc$ est maintenant possible en un nombre fini d'étapes!

- $\mathcal{A}cc$ se calcule de proche en proche comme la "limite" de la suite $\mathcal{U}_i$ définie par la récurrence suivante :
  - $\mathcal{U}_0 = q_0$,
  - $\mathcal{U}_{i+1} = \mathcal{U}_i \bullet (\varepsilon + \mathcal{A}) = \mathcal{U}_i + \mathcal{U}_i \bullet \mathcal{A}$.

La seconde clause, où l'on a utilisé l'extension de l'action aux langages, peut aussi s'écrire :

$$q \in \mathcal{U}_{i+1} \text{ ssi "} q \in \mathcal{U}_i \text{ ou il existe } r \in \mathcal{U}_i \text{ et } x \in \mathcal{A} \text{ tels que } q = r \bullet x\text{"}$$

pour tout $q \in Q$.

- La suite $\mathcal{U}_i$, qui est une suite croissante de parties de l'ensemble fini Q, est stationnaire; plus précisément, il existe $N < |\,Q\,|$ tel que $i \geq N$ implique $\mathcal{U}_i = \mathcal{U}_N$: il est clair que $\mathcal{U}_N = \mathcal{A}cc$.

Exemple 1 (suite).

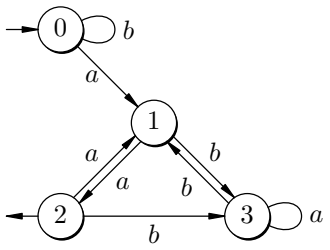
En appliquant la construction par récurrence à l'exemple 1, il vient successivement :

$$\begin{aligned}
&\mathcal{U}_0 = 0\\
&\mathcal{U}_1 = 0 + 0 \bullet (a+b) = 0 + 0 \bullet a + 0 \bullet b = 0 + 1 + 0 = 0 + 1\\
&\mathcal{U}_2 = 0 + 1 + (0+1) \bullet (a+b) = 0 + 1 + 2 + 3\\
&\mathcal{U}_3 = 0 + 1 + 2 + 3 + (0+1+2+3) \bullet (a+b) = \mathcal{U}_2
\end{aligned}$$

On a donc $\mathcal{A}cc = 0 + 1 + 2 + 3$.

L'AFDC $\mathbf{A}'$ obtenu en ne conservant que les états accessibles se présente ainsi :

e/s	q	$q \bullet a$	$q \bullet b$
$\rightarrow$	0	1	0
	1	2	3
$\leftarrow$	2	1	3
	3	3	1



• La construction précédente de l'ensemble est effective. La voici, présentée sous la forme d'un algorithme, qui pourra servir d'exemple à tous les calculs analogues que nous verrons par la suite :

Calcul de $\mathcal{A}cc$

```
Acc := q0
tant qu'il existe q ∈ Acc non marqué faire
    sélectionner q ∈ Acc non marqué
    pour x parcourant A faire
        Acc := Acc + q • x
    fin
    marquer q
fin
```

Dans cet algorithme, on marque les états après les avoir traités, pour éviter un nouveau calcul à leur sujet, qui serait complètement inutile et ne permettrait pas à l'algorithme de terminer!

Remarque.

L'algorithme précédent permet de décider si $\mathcal{L}(\mathbf{A}) = \emptyset$ puisque cette propriété est équivalente à $\mathcal{A}cc \cap F = \emptyset$.

3.2 – Le théorème de Kleene.

Ce théorème fait le lien entre les langages réguliers et les langages reconnus par les AFDC : c'est le résultat essentiel de la théorie et c'est aussi la base des méthodes d'analyse lexicale.

THEOREME (Kleene)

Pour tout $L \subseteq \mathcal{A}^*$ les deux conditions suivantes sont équivalentes :

(a) L est régulier.

(b) Il existe un AFDC $\mathbf{A}$ tel que $L = \mathcal{L}(\mathbf{A})$.

La preuve du fait que (a) implique (b) consiste en la construction d'un AFDC à partir d'une expression régulière et ne sera faite qu'à la section 5.

Vérifions l'autre implication : soit $\mathbf{A} = (Q, \mathcal{A}, \bullet, q_0, F)$ un AFDC, nous allons montrer que $\mathcal{L}(\mathbf{A})$ est l'une des composantes de la solution d'un système d'équations linéaires à coefficients réguliers (plus précisément, finis), satisfaisant la condition d'unicité : c'est donc un langage régulier.

Pour tout $q \in Q$ on considère l'ensemble $Rec(q)$ des mots reconnus par l'AFDC $\mathbf{A}(q) = (Q, \mathcal{A}, \bullet, q, F)$ obtenu en prenant l'état $q \in Q$ comme entrée dans $\mathbf{A}$ au lieu de q_0. Ceci revient à écrire : "$u \in Rec(q)$ ssi $q \bullet u \in F$".

On a les propriétés suivantes : quels que soient $q \in Q$, $x \in \mathcal{A}$ et $u \in \mathcal{A}^*$

- $\varepsilon \in Rec(q)$ ssi $q \in F$;
- $xu \in Rec(q)$ ssi $xu \in xRec(q \bullet x)$.

La première est évidente et la seconde vient des équivalences successives :

$$\begin{array}{lll} xu \in Rec(q) & \text{ssi } q \bullet (xu) \in F & \text{(définition de } Rec(q)\text{)} \\ & \text{ssi } (q \bullet x) \bullet u \in F & \text{(définition d'une action)} \\ & \text{ssi } u \in Rec(q \bullet x) & \text{(définition de } Rec(q \bullet x)\text{)} \\ & \text{ssi } xu \in xRec(q \bullet x) & \end{array}$$

En utilisant le fait que tout élément de $\mathcal{A}^*$ est ou bien ε ou bien de la forme xu pour un $x \in \mathcal{A}$ et un $u \in \mathcal{A}^*$ (adjonction à gauche, pour une fois!), on peut écrire :

$$Rec(q) = \begin{cases} \displaystyle\sum_{x \in \mathcal{A}} xRec(q \bullet x) + \varepsilon & \text{si } q \in F, \\ \displaystyle\sum_{x \in \mathcal{A}} xRec(q \bullet x) & \text{sinon.} \end{cases}$$

L'ensemble des égalités correspondant à tous les $q \in Q$ est un système d'équations linéaires dont les inconnues sont $X_q = Rec(q)$ et dont les coefficients sont réguliers (ε et des sommes finies d'éléments de $\mathcal{A}$); de plus, il n'y a aucune inconnue dont le coefficient contient ε (la condition d'unicité est donc vérifiée) : $\mathcal{L}(\mathbf{A}) = Rec(q_0)$ est un langage régulier d'après le résultat de la section 1.1.

Exemple 1 (suite).

Le système d'équations correspondant à l'AFDC $\mathbf{A}'$ de l'exemple 1 précédent est :

$$\begin{array}{lllllllll} X_0 & = & bX_0 & + & aX_1 & & & & \\ X_1 & = & & & & aX_2 & + & bX_3 & \\ X_2 & = & & & aX_1 & & + & bX_3 & + \quad \varepsilon \\ X_3 & = & & & bX_1 & & + & aX_3 & \end{array}$$

Ce système a été résolu dans le premier chapitre : la seule composante de la solution qui nous intéresse ici est X_0, qui est le langage reconnu par $\mathbf{A}'$:

$$\mathcal{L}(\mathbf{A}') = b^*a(aa + ba^*b + aba^*b)^*a.$$

4 – Automates finis (AF).

L'action d'un AFDC est définie par une application $\bullet : Q \times \mathcal{A} \to Q$ qui, à chaque $q \in Q$ et chaque $x \in \mathcal{A}$ associe donc **un et un seul** état $q \bullet x$. Or, pour démontrer la réciproque du théorème de Kleene, il faut construire un AFDC à partir d'une expression régulière : lorsque l'on tente de faire une telle construction, il est facile d'obtenir un objet, représentable par un graphe de transition, mais il est bien rare que ce soit celui d'un AFDC!

Pour tenir compte de cet echec potentiel, on est conduit à considérer des *actions non nécessairement déterministes et complètes*, c'est–à–dire des applications

$$\bullet : Q \times \mathcal{A} \to \mathcal{P}(Q),$$

Avec une telle action, $q \bullet x$ est un ensemble, pas nécessairement réduit à un élément, c'est–à–dire qu'il peut aussi :

- ou bien être vide : l'AF n'est alors *pas complet*,
- ou bien comporter plusieurs éléments : l'AF n'est alors *pas déterministe*.

Lorsque que l'on accepte qu'un automate ne soit pas déterminsite, il est normal de ne plus se restreindre au cas d'une entrée unique, mais d'accepter aussi un ensemble d'entrées.

L'intérêt des automates finis ainsi obtenus tient surtout au fait qu'ils sont en général plus facile à concevoir que les AFDC; par ailleurs, nous verrons que l'on peut transformer un tel automate en un AFDC équivalent, et ce, de façon effective.

Ceci mène à la définition qui suit :

Les AF

Un automate fini (en abrégé AF) est la donnée d'un 5–uplet $\mathbf{A} = (Q, \mathcal{A}, \bullet, I, F)$ où :

- Q est un ensemble fini, (d'états)
- $\mathcal{A}$ est un alphabet fini,
- $\bullet : Q \times \mathcal{A} \to \mathcal{P}(Q)$ est une application, (une action non nécessairement DC)
- $I \subseteq Q$ est l'ensemble des entrées, (ou états initiaux)
- $F \subseteq Q$ est l'ensemble des sorties. (ou états finaux)

- La table définissant $\bullet : Q \times \mathcal{A} \to \mathcal{P}(Q)$ devra maintenant comporter des parties de l'ensemble des états Q :
  - l'ensemble vide sera évidemment noté $\emptyset$,
  - l'ensemble $\{q\}$ constitué du seul élément $q \in Q$ sera simplement noté q,
  - plus généralement, l'ensemble $\{q_1, \ldots, q_n\}$ comportant $n > 1$ éléments sera noté de la façon qui nous est maintenant habituelle : $q_1 + \cdots + q_n$.
- Les constituants du *graphe de transition* d'un AF sont les suivants :
  - un nœud (q) pour chaque $q \in Q$;
  - une arête $(q) \xrightarrow{x} (r)$ pour chaque $q \in Q$, chaque $x \in \mathcal{A}$ et chaque $r \in q \bullet x$.
- Le fonctionnement d'un AF est essentiellement le même que celui d'un AFDC.

Si $\mathbf{A} = (Q, \mathcal{A}, \bullet, I, F)$ est un AF et $u \in \mathcal{A}^*$ on dira encore que $u \in \mathcal{L}(\mathbf{A})$ lorsque u est capable de définir un chemin qui part d'une entrée $q \in I$ et qui aboutit à l'une des sorties. Cependant, on observe que les tentatives pour construire un tel chemin peuvent être multiples : certaines d'entre elles peuvent échouer, d'autres au contraire peuvent réussir, mais de façons diverses!

Nous allons tout d'abord adapter la notion de cheminement au cas des AF : la notion obtenue est traditionnellement appelée *dérivation*.

4.1 – Dérivation dans un AF.

Toutes les définitions qui suivent sont relatives à un AF $\mathbf{A} = (Q, \mathcal{A}, \bullet, I, F)$.

- *Une configuration* est un couple $(q, u) \in Q \times \mathcal{A}^*$.
- *Une transition* est un changement de configuration en un seul pas : une transition "coûte" (ou "consomme") une lettre.
Une transition se définit, pour tout $q \in Q$ et tout $r \in Q$, par :
$$(q, xu) \vdash_{\mathbf{A}}^{1} (r, u) \text{ ssi } r \in q \bullet x$$
quels que soient $u \in \mathcal{A}^*$ et $x \in \mathcal{A}$.
- *Une dérivation de longueur* i, notée $(q, u) \vdash_{\mathbf{A}}^{i} (r, v)$ est un enchaînement de i transitions successives qui se définit par récurrence sur i :
$$0 : (q, u) \vdash_{\mathbf{A}}^{0} (q, u) \qquad \text{(pas de transition)}$$
$$i \mapsto i+1 : (q, u) \vdash_{\mathbf{A}}^{i} (s, w) \vdash_{\mathbf{A}}^{1} (r, v) \qquad \text{(une transition de plus)}$$
- On définit la relation $\vdash_{\mathbf{A}}^{*}$ entre configurations par :
$$(q, u) \vdash_{\mathbf{A}}^{*} (r, v) \text{ ssi il existe une dérivation } (q, u) \vdash_{\mathbf{A}}^{i} (r, v).$$

- Lorsqu'il n'y a pas d'ambiguïté sur l'AF $\mathbf{A}$ qui est en cause, on note $\vdash^{i}$ et $\vdash^{*}$ au lieu de $\vdash^{i}_{\mathbf{A}}$ et $\vdash^{*}_{\mathbf{A}}$.

Langage reconnu par un AF

Le langage reconnu par un AF $\mathbf{A} = (Q, \mathcal{A}, \bullet, I, F)$ est $\mathcal{L}(\mathbf{A}) \subseteq \mathcal{A}^*$ défini par

$$u \in \mathcal{L}(\mathbf{A}) \text{ ssi il existe } s \in I \text{ et } r \in F \text{ tels que } (s,u) \vdash^{*}_{\mathbf{A}} (r,\varepsilon).$$

Deux AF $\mathbf{A}$ et $\mathbf{A}'$ sont *équivalents* ssi $\mathcal{L}(\mathbf{A}) = \mathcal{L}(\mathbf{A}')$.

On peut maintenant préciser la notion de chemin : la suite des états par lesquels passe une dérivation $(q,u) \vdash^{i} (r,\varepsilon)$ définit un élément de $ch(q,u)$, c'est–à–dire un chemin que l'on peut parcourir en partant de l'état q, en utilisant correctement les caractères de u.

AF particuliers.

AFD : Un AF est dit *déterministe* ssi
- son ensemble d'entrées comporte **au plus** un élément;
- et, quels que soient $q \in Q$ et $x \in \mathcal{A}$, $q \bullet x$ contient **au plus** un élément.

AFC : Un AF est dit *complet* ssi
- son ensemble d'entrées comporte **au moins** un élément;
- et, quels que soient $q \in Q$ et $x \in \mathcal{A}$, $q \bullet x$ contient **au moins** un élément.

Les AF qui sont à la fois déterministes et complets sont donc les **AFDC** que nous avons introduits initialement et que nous avons qualifiés par anticipation; en tout cas, on ne devra pas oublier qu'un AF peut être un AFD et même un AFDC!

Il est facile de vérifier que, pour tout $u \in \mathcal{A}^*$ et tout $q \in Q$:
- $ch(q,u)$ contient **au plus** un élément lorsque $\mathbf{A}$ est déterministe,
- $ch(q,u)$ contient **au moins** un élément lorsque $\mathbf{A}$ est complet.

Remarque.

Soit $\mathbf{A}$ un AF sur $\mathcal{A}$ et soit $\mathcal{B}$ un alphabet tel que $\mathcal{A} \subseteq \mathcal{B}$, alors il est facile de considérer $\mathbf{A}$ comme un AF sur $\mathcal{B}$. Mais, si $\mathbf{A}$ est un AFC (en particulier un AFDC) et si $\mathcal{B} \neq \mathcal{A}$ alors l'AF sur $\mathcal{B}$ dont il vient d'être question n'est plus complet, car $q \bullet x$ est vide pour $x \in \mathcal{B} - \mathcal{A}$ (il reste déterministe s'il l'était).

Exemple 1 (suite).

Dans un AFDC, $r \in q \bullet x$ équivaut à $r = q \bullet x$, quels que soient $q \in Q$, $r \in Q$ et $x \in \mathcal{A}$. Dans cette situation, on voit que pour tout $q \in Q$ et tout $u \in \mathcal{A}^*$, il existe une dérivation unique $(q,u) \vdash^{i} (q \bullet u, \varepsilon)$, et que $i = |u|$. Par ce qui précède, il est clair que lorsqu'un AF $\mathbf{A}$ est déterministe et complet, la définition de $\mathcal{L}(\mathbf{A})$ est bien équivalente à celle qui a été donnée pour les AFDC.

Dans l'exemple 1 précédent, on peut écrire la dérivation suivante

$$(0,abab) \vdash^{1} (0 \bullet a, bab) = (1,bab) \vdash^{1} (1 \bullet b, ab) = (3,ab) \vdash^{1} (3 \bullet a, b) = (3,b) \vdash^{1} (3 \bullet b, \varepsilon) = (1,\varepsilon)$$

Exemple 2.

Soient $\mathcal{A} = a + b$ et $Q = 0 + 1 + 2 + 3$ et considérons l'AF $\mathbf{A}$ tel que $I = 0$ et $F = 3$ présenté par sa table et représenté par son graphe de transition.

- $ch(0,aa) = \emptyset$.

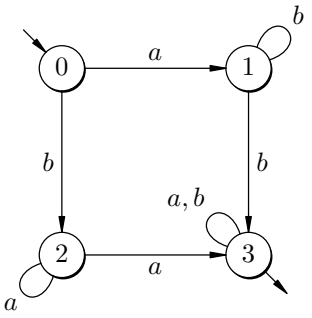
Ceci signifie que l'on n'a $(0,aa) \vdash^{*} (q,\varepsilon)$ pour aucun état q : en effet on peut effectuer la transition $(0,aa) \vdash^{1} (1,a)$, mais il n'est pas possible d'aller plus loin puisque $1 \bullet a = \emptyset$.

- $ch(0,abba) = 01133 + 01333$.

Voici la dérivation définissant le chemin 01133 :

$$(0,abba) \vdash^{1} (1,bba) \vdash^{1} (1,ba) \vdash^{1} (3,a) \vdash^{1} (3,\varepsilon)$$

e/s	q	$q \bullet a$	$q \bullet b$
$\rightarrow$	0	1	2
	1	$\emptyset$	$1+3$
	2	$2+3$	$\emptyset$
$\leftarrow$	3	3	3



AF de l'exemple 2.

- Toutes les tentatives pour construire un chemin ne sont pas nécessairement fructueuses, par exemple $(0, abba) \vdash^1 (1, bba) \vdash^1 (1, ba) \vdash^1 (1, a)$ tombe sur une impasse puisque $1 \bullet a = \emptyset$.

4.2 – Détermination.

On peut simuler toutes les tentatives pour construire un chemin de façon plus systématique en considérant l'extension

$$\bullet : \mathcal{P}(Q) \times \mathcal{A} \to \mathcal{P}(Q)$$

aux ensembles d'états de l'application $\bullet : Q \times \mathcal{A} \to \mathcal{P}(Q)$, obtenue par préservation des sommes, c'est–à–dire, définie par

$$S \bullet x = \sum_{s \in S} s \bullet x$$

pour toute $S \subseteq Q$ et tout $x \in \mathcal{A}$.

Cette application s'étend à son tour de façon unique en une action $\bullet : \mathcal{P}(Q) \times \mathcal{A}^* \to \mathcal{P}(Q)$ de $\mathcal{A}^*$ sur l'ensemble $\mathcal{P}(Q)$, par la récurrence

- $S \bullet \varepsilon = S$,
- $S \bullet ux = (S \bullet u) \bullet x$,

pour toute $S \subseteq Q$, tout $u \in \mathcal{A}^*$ et tout $x \in \mathcal{A}$.

Propriétés.

Soit $\mathbf{A} = (Q, \mathcal{A}, \bullet, I, F)$ *un AF alors, pour tout* $u \in \mathcal{A}^*$ *:*

1) *pour toute* $S \subseteq Q$ *et tout* $r \in Q$ *:*

$$r \in S \bullet u \text{ ssi il existe } s \in S \text{ tel que } (s, u) \vdash^* (r, \varepsilon),$$

2) $u \in \mathcal{L}(\mathbf{A})$ *ssi* $(I \bullet u) \cap F \neq \emptyset$.

La preuve de 1) se fait par récurrence (voir, ci–dessous, la figure illustrant l'exemple 2).

- Dans un sens, faisons une récurrence sur les mots :
  - $r \in S \bullet \varepsilon$ signifie simplement $r \in S$: en prenant $s = r$, on a bien une dérivation $(s, \varepsilon) \vdash^0 (r, \varepsilon)$!
  - Supposons la propriété vraie pour $u \in \mathcal{A}^*$ (quel que soit S!) et soit $x \in \mathcal{A}$:
    si $r \in S \bullet (ux)$ alors $r \in (S \bullet u) \bullet x$, c'est–à–dire $r \in T \bullet x$ pour $T = S \bullet u$: on a donc un $t \in S \bullet u$ tel que $r \in t \bullet x$. La dernière remarque signifie que l'on a une dérivation $(t, x) \vdash^1 (r, \varepsilon)$; par ailleurs, l'hypothèse de récurrence appliquée à $t \in S \bullet u$ nous donne une dérivation $(s, u) \vdash^i (t, \varepsilon)$. On peut conclure en considérant la dérivation composée : $(s, ux) \vdash^i (t, x) \vdash^1 (r, \varepsilon)$.
- L'autre implication se montre par récurrence sur la longueur des dérivations . . .

La propriété 2) est une conséquence simple de 1) et des définitions.

En fait, nous venons de démontrer que les AF savent reconnaître les mêmes langages que les AFDC :

Propriété de détermination

Tout AF est équivalent à un AFDC.

Plus précisément l'AFDC $DC(\mathbf{A})$ défini ci–dessous, est équivalent à l'AF $\mathbf{A} = (Q, \mathcal{A}, \bullet, I, F)$: l'opération DC s'appelle la *détermination* (ou *déterminisation*) de $\mathbf{A}$.

$DC(\mathbf{A})$

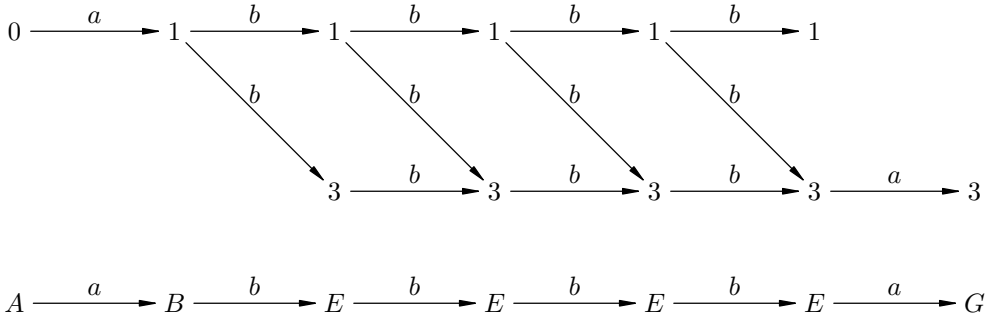
$DC(\mathbf{A})$ est la partie accessible de l'AFDC $(Q', \mathcal{A}, \bullet, q'_0, F')$ suivant :
- *Etats.* $S \in Q'$ ssi $S \subseteq Q$,
- *Action.* $\bullet : Q' \times \mathcal{A} \to Q'$, l'extension de $\bullet$ aux ensembles d'états,
- *Entrée.* $q'_0 = I$,
- *Sorties.* $S \in F'$ ssi $S \cap F \neq \emptyset$.

Le fait que $DC(\mathbf{A})$ est équivalent à $\mathbf{A}$ est exprimé par les propriétés ci–dessus.

La construction de $DC(\mathbf{A})$ est effective. Les états accessibles sont les seuls qui nous intéressent : leur calcul se fait de proche en proche à partir de I, en suivant les lignes de l'algorithme de la section 3.1.

Exemple 2 (suite).

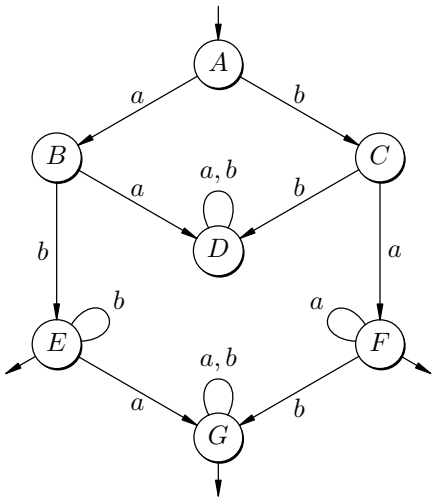
La figure montre toutes les tentatives pour construire un chemin défini par un mot u à partir de l'état 0, dans l'AF de l'exemple 2, et la même tentative dans l'AFDC correspondant : c'est une bonne illustration pour suivre la démonstration de la propriété 1) ci–dessus.



Exemple 2 avec $u = abbbba$.

La construction de $DC(\mathbf{A})$ est présentée dans la figure ci–dessous.

e/s	S	$S \bullet a$	$S \bullet b$
$\rightarrow$	$A = 0$	B	C
	$B = 1$	D	E
	$C = 2$	F	D
	$D = \emptyset$	D	D
$\leftarrow$	$E = 1 + 3$	G	E
$\leftarrow$	$F = 2 + 3$	F	G
$\leftarrow$	$G = 3$	G	G



4.3 – Les états productifs d'un AF.

La notion d'état accessible dans un AFDC (section 3.1) s'adapte sans problème au cas des AF : il n'est pas nécessaire d'y revenir ici.

Une notion duale est celle d'*état productif* : un état est productif s'il existe un chemin qui en part et qui mène à une sortie.

Les états productifs

$q \in Q$ est *productif* ssi il existe $u \in \mathcal{A}^*$ tel que $(q \bullet u) \cap F \neq \emptyset$.

$(q \bullet u) \cap F \neq \emptyset$ signifie évidemment qu'il existe $r \in F$ tel que $(q, u) \vdash^* (r, \varepsilon)$.

Il est clair que l'ensemble $\mathcal{P}rod$ des états productifs de $\mathbf{A}$ est le plus petit ensemble qui vérifie :

- $F \subseteq \mathcal{P}rod$,
- pour tout $q \in Q$, s'il existe $x \in \mathcal{A}$ tel que $(q \bullet x) \cap \mathcal{P}rod \neq \emptyset$ alors $q \in \mathcal{P}rod$.

Cette observation est suffisante pour construire l'AF $\mathbf{A}' = (Q', \circ, I', F)$ suivant :

- *Etats.* $Q' = \mathcal{P}rod$, (l'ensemble des états productifs de $\mathbf{A}$)
- *Entrées.* $I' = I \cap \mathcal{P}rod$, (l'ensemble des entrées productives)
- *Action.* $\circ : Q' \times \mathcal{A} \rightarrow Q'$, ($q \circ x = (q \bullet x) \cap \mathcal{P}rod$)
- *Sorties.* F.

Par construction, tous les états de $\mathbf{A}'$ sont productifs et, comme tout chemin aboutissant en F part d'un état productif, on a $\mathcal{L}(\mathbf{A}') = \mathcal{L}(\mathbf{A})$.

Propriété

Tout AF est équivalent à un AF dont tous les états sont productifs.

Il est facile de vérifier que $\mathcal{P}rod$ est la limite de la suite croissante stationnaire $\mathcal{U}_i$ définie par la récurrence suivante :

- $\mathcal{U}_0 = F$,
- $\mathcal{U}_{i+1} = \mathcal{U}_i \bullet (\varepsilon + \mathcal{A})^{-1} = \mathcal{U}_i + \mathcal{U}_i \bullet \mathcal{A}^{-1}$.

Dans la seconde clause, on a utilisé l'extension aux langages de l'*action inverse* d'un élément de $\mathcal{A}$ sur un état, $q \bullet x^{-1} \subseteq Q$, définie par

$$r \in q \bullet x^{-1} \text{ ssi } q \in r \bullet x.$$

Cette construction permet de calculer effectivement $\mathcal{P}rod$ et d'imaginer un algorithme qui décide si $\mathcal{L}(\mathbf{A}) = \emptyset$, car cette condition équivaut à $I \cap \mathcal{P}rod = \emptyset$.

Attention.

Lorsque $\mathbf{A}$ est un AFDC, l'AF équivalent, dont tous les états sont productifs, **n'est** généralement **pas complet**. En effet :

- on peut avoir $q_0 \notin \mathcal{P}rod$,
- il peut y avoir des états accessibles qui ne sont pas productifs : par exemple, dans un AFDC obtenu par détermination d'un AF, l'état $\emptyset$ (l'état D dans l'exemple 2 précédent) peut être accessible, mais il n'est jamais productif.

5 – Des AF encore moins déterministes.

Il est fort utile d'aller encore plus loin dans le non déterminisme en admettant des ε–transitions, c'est–à–dire des transitions gratuites! On obtient ainsi les ε–AF, qui permettent de faire aisément les constructions utiles pour la démonstration de la fin du théorème de Kleene et qui, de plus, interviendront de façon essentielle dans un algorithme d'analyse syntaxique que nous étudierons par la suite. En voici la définition :

Les ε–AF

Un ε–automate fini (en abrégé ε–AF) est la donnée d'un 5–uplet $\mathbf{A} = (Q, \mathcal{A}, \delta, I, F)$ où Q, $\mathcal{A}$, I et F sont comme précédemment, mais où $\delta : Q \times (\varepsilon + \mathcal{A}) \to \mathcal{P}(Q)$.

- Un AF est un ε–AF tel que $\delta(q, \varepsilon) = \emptyset$ pour tout q.
- En dehors de ce cas, la table définissant δ doit maintenant comporter une colonne pour les valeurs des $\delta(q, \varepsilon)$.
- Les constituants du *graphe de transition* d'un ε–AF sont les suivants :
  - un nœud (q) pour chaque $q \in Q$;
  - une arête $(q) \xrightarrow{x} (r)$ pour chaque $q \in Q$, chaque $x \in \varepsilon + \mathcal{A}$ et chaque $r \in \delta(q, x)$.

Nous allons les étudier succintement, montrer qu'ils sont capables de reconnaître les mêmes langages que les AF.

5.1 – Dérivation dans un ε–AF.

Toutes les définitions qui suivent sont relatives à un ε–AF $\mathbf{A} = (Q, \mathcal{A}, \delta, q_0, F)$: elles sont les adaptations naturelles des définitions relatives aux AF.

- *Une configuration* est un couple $(q, u) \in Q \times \mathcal{A}^*$.
- *Une transition* est un changement de configuration en un seul pas : une transition "coûte" (ou "consomme") au plus une lettre.

Une transition se définit, pour $q \in Q$, $r \in Q$, $x \in \varepsilon + \mathcal{A}$ et $u \in \mathcal{A}^*$, par :

$$(q, xu) \vdash^{1}_{\mathbf{A}} (r, u) \text{ ssi } r \in \delta(q, x).$$

On parle d'*ε–transition* lorsque $x = \varepsilon$ et de *transition sur* x lorsque $x \in \mathcal{A}$.

- *Une dérivation de longueur* i, notée $(q, u) \vdash^{i}_{\mathbf{A}} (r, v)$ est un enchaînement de i transitions successives qui se définit par récurrence sur i :

$$0 : (q, u) \vdash^{0}_{\mathbf{A}} (q, u) \qquad \text{(pas de transition)}$$

$$i \mapsto i+1 : (q, u) \vdash^{i}_{\mathbf{A}} (s, w) \vdash^{1}_{\mathbf{A}} (r, v) \qquad \text{(une transition de plus)}$$

- *Une ε–dérivation* est une dérivation de la forme $(q, u) \vdash^{i}_{\mathbf{A}} (r, u)$: c'est donc ou bien une dérivation de longueur nulle ou bien une dérivation constituée d'ε–transitions.
- On définit la relation $\vdash^{*}_{\mathbf{A}}$ entre configurations par :

$$(q, u) \vdash^{*}_{\mathbf{A}} (r, v) \text{ ssi il existe une dérivation } (q, u) \vdash^{i}_{\mathbf{A}} (r, v).$$

- S'il n'y a pas d'ambiguïté sur l'ε–AF $\mathbf{A}$ qui est en cause, on note $\vdash^{i}$ et $\vdash^{*}$ au lieu de $\vdash^{i}_{\mathbf{A}}$ et $\vdash^{*}_{\mathbf{A}}$.

Précisons la notion de chemin : la suite des états par lesquels passe une dérivation $(q,u) \vdash^{i} (r,\varepsilon)$ définit un élément de $ch_q(u)$, c'est–à–dire un chemin que l'on peut parcourir en partant de l'état q, en utilisant correctement les caractères de u. Une définition par récurrence peut se faire en suivant celle de la dérivation.

Langage reconnu par un ε–AF

Le langage reconnu par un ε–AF $\mathbf{A} = (Q, \mathcal{A}, \delta, I, F)$ noté $\mathcal{L}(\mathbf{A}) \subseteq \mathcal{A}^*$, est défini par

$$u \in \mathcal{L}(\mathbf{A}) \text{ ssi il existe } s \in I \text{ et } r \in F \text{ tels que } (s,u) \vdash^{*}_{\mathbf{A}} (r,\varepsilon).$$

Deux ε–AF $\mathbf{A}$ et $\mathbf{A}'$ sont *équivalents* ssi $\mathcal{L}(\mathbf{A}) = \mathcal{L}(\mathbf{A}')$.

Exemple 3.

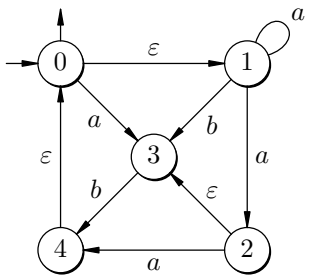
On prend $\mathcal{A} = a + b$ et $Q = 0 + 1 + 2 + 3 + 4$.
L'ε–AF $\mathbf{A}$ tel que $q_0 = 0$ et $F = 0$ est présenté par sa table et représenté par son graphe de transition : $\mathbf{A}$ comporte des ε–transitions. La notion de chemin est encore moins déterministe que dans l'exemple précédent car les ε–transitions ne consomment aucun caractère. Par exemple

$$ch(0,ab) = 034 + 0340 + 03401 + 0113 + 01234 + 012340 + 0123401$$

est constitué de chemins qui n'ont pas tous la même longueur. Voici une dérivation définissant le chemin 012340 :

$$(0,ab) \vdash^{1} (1,ab) \vdash^{1} (2,b) \vdash^{1} (3,b) \vdash^{1} (4,\varepsilon) \vdash^{1} (0,\varepsilon)$$

e/s	q	$\delta(q,\varepsilon)$	$\delta(q,a)$	$\delta(q,b)$
$\leftrightarrow$	0	1	3	$\emptyset$
	1	$\emptyset$	$1+2$	3
	2	3	4	$\emptyset$
	3	$\emptyset$	$\emptyset$	4
	4	0	$\emptyset$	$\emptyset$



5.2 – Détermination.

Les ε–AF savent reconnaître les mêmes langages que les AFDC.

Propriété de détermination

Tout ε–AF est équivalent à un AFDC.

Nous allons construire un AFDC $DC(\mathbf{A})$ équivalent à un ε–AF $\mathbf{A} = (Q, \mathcal{A}, \delta, I, F)$ donné : cette opération s'appelle la *détermination* (ou *déterminisation*) de $\mathbf{A}$.

Les états de $DC(\mathbf{A})$ sont des parties de Q qui, dans le cas où il existe effectivement des ε–transitions, ne peuvent pas être quelconques. En effet, si $\bullet$ désigne l'opération servant à définir l'action de $DC(\mathbf{A})$ et si $S \subseteq Q$ est un état de $DC(\mathbf{A})$, on devra avoir $S \bullet \varepsilon = S$, c'est–à–dire $\delta(s,\varepsilon) \subseteq S$ pour tout $s \in S$: ceci signifie que S doit être close par ε–transition.

Précisons cette notion de clôture dans $\mathbf{A}$

- La clôture de $q \in Q$ est l'ensemble $cl(q) \subseteq Q$ défini par :

$$r \in cl(q) \text{ ssi } (q,\varepsilon) \vdash^{*} (r,\varepsilon).$$

- cl s'étend à tout $S \subseteq Q$, par préservation des sommes : $cl(S) = \sum_{s \in S} cl(s)$.
- De même, δ s'étend à toute $S \subseteq Q$, par préservation des sommes : $\delta(S, x) = \sum_{s \in S} \delta(s, x)$

Définition. $S \subseteq Q$ est dite *close* ssi $cl(S) = S$.

On peut remarquer que :

- L'existence de la dérivation triviale $(q, \varepsilon) \overset{0}{\vdash\!\!\!-} (q, \varepsilon)$ signifie que l'on a $q \in cl(q)$;
- $\emptyset$ est clos;
- pour tout $q \in Q$ et toute $S \subseteq Q$ close : $q \in S$ ssi $cl(q) \subseteq S$;
- $cl(S)$ est close pour toute $S \subseteq Q$.

Nous pouvons maintenant définir $DC(\mathbf{A})$:

$DC(\mathbf{A})$

$DC(\mathbf{A})$ est la partie accessible de l'AFDC $(Q', \mathcal{A}, \bullet, q'_0, F')$ suivant :

- *Etats.* $S \in Q'$ ssi $S \subseteq Q$ et S est close;
- *Action.* $\bullet : Q' \times \mathcal{A} \to Q'$ est définie par $S \bullet x = cl(\delta(S, x))$;
- *Entrée.* $q'_0 = cl(I)$;
- *Sorties.* $S \in F'$ ssi $S \cap F \neq \emptyset$.

Exemple 3 (suite).

Reprenons notre précédent exemple 3.
Pour calculer la table de $DC(\mathbf{A})$, il est commode de remplacer la colonne des valeurs de $\delta(q, \varepsilon)$ par celle des $cl(q)$ dans la table de $\mathbf{A}$.

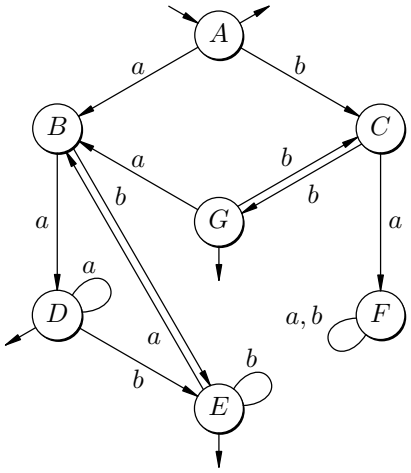
e/s	q	$cl(q)$	$\delta(q,a)$	$\delta(q,b)$
$\leftrightarrow$	0	$0+1$	3	$\emptyset$
	1	1	$1+2$	3
	2	$2+3$	4	$\emptyset$
	3	3	$\emptyset$	4
	4	$0+1+4$	$\emptyset$	$\emptyset$

Voici les actions sur $B = 1 + 2 + 3$:

$B \bullet a = cl(1+2) + cl(4) + \emptyset = 0 + 1 + 2 + 3 + 4;$
$B \bullet b = cl(3) + \emptyset + cl(4) = 0 + 1 + 3 + 4.$

Le calcul de $DC(\mathbf{A})$ se fait en partant de $A = cl(0)$: ceci donne la table et le graphe de transition suivants :

e/s	S	$S \bullet a$	$S \bullet b$
$\leftrightarrow$	$A = 0 + 1$	B	C
	$B = 1 + 2 + 3$	D	E
	$C = 3$	F	G
$\leftarrow$	$D = 0 + 1 + 2 + 3 + 4$	D	E
$\leftarrow$	$E = 0 + 1 + 3 + 4$	B	E
	$F = \emptyset$	F	F
$\leftarrow$	$G = 0 + 1 + 4$	B	C



5.3 – Démonstration de la propriété de détermination.

La propriété $\mathcal{L}(DC(\mathbf{A})) = \mathcal{L}(\mathbf{A})$ n'est pas parfaitement évidente bien qu'elle soit très analogue à celle du cas des AF.

Propriétés.

Soient $\mathbf{A} = (Q, \mathcal{A}, \delta, I, F)$ un ε–AF et Q' l'ensemble des parties closes de Q alors, pour tout $u \in \mathcal{A}^*$:

1) quels que soient $S \in Q'$ et $r \in Q$:

$$r \in S \bullet u \text{ ssi il existe } s \in S \text{ tel que } (s,u) \vdash^* (r,\varepsilon),$$

2) $u \in \mathcal{L}(\mathbf{A})$ ssi $(cl(I) \bullet u) \cap F \neq \emptyset$.

Démonstration de 1).

Désignons cette propriété par $P(u)$.

Démonstration de $P(\varepsilon)$:

- d'un côté, il est clair que si $r \in S$ alors on a $(s,\varepsilon) \vdash^* (r,\varepsilon)$ avec $s = r$!
- de l'autre, $(s,\varepsilon) \vdash^* (r,\varepsilon)$ signifie $r \in cl(s)$, ce qui implique $r \in S$ puisque S est close.

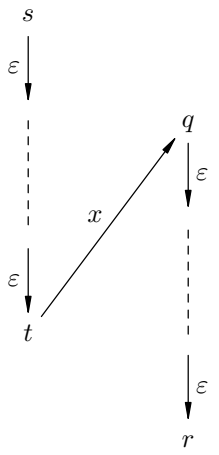
Démonstration de $P(x)$ pour $x \in \mathcal{A}$: les définitions permettent d'écrire les équivalences successives suivantes

$r \in S \bullet x$ ssi $r \in cl(\delta(S,x))$
ssi il existe $q \in \delta(S,x)$ tel que $r \in cl(q)$
ssi il existe $t \in S$ et $q \in \delta(t,x)$ tels que $r \in cl(q)$
ssi il existe $s \in S$ tel que $(s,x) \vdash^* (r,\varepsilon)$

La dernière équivalence mérite une petite explication :

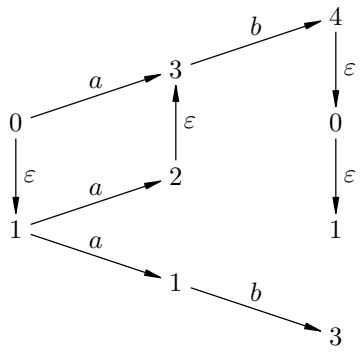
- il est facile de descendre, en composant $(t,x) \vdash^1 (q,\varepsilon)$ et une dérivation $(q,\varepsilon) \vdash^i (r,\varepsilon)$: il suffit alors de prendre $s = t$ pour conclure;
- réciproquement, décomposons une dérivation $(s,x) \vdash^i (r,\varepsilon)$ pour mettre en évidence la transition sur x :
  1) $(s,x) \vdash^k (t,x)$,
  2) $(t,x) \vdash^1 (q,\varepsilon)$,
  3) $(q,\varepsilon) \vdash^l (r,\varepsilon)$,

  L'existence de la dérivation 1) implique $t \in S$ lorsque $s \in S$, puisque S est clos, celle de 2) implique $q \in \delta(t,x)$ et enfin, celle de la dérivation 3) implique $r \in cl(q)$.



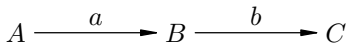
Il est maintenant facile de montrer que $P(u)$ implique $P(ux)$.

$$S \xrightarrow{\ x\ } S \bullet x$$



Exemple 3 (suite).

La figure ci–contre montre toutes les tentatives pour construire un chemin défini par le mot $u = ab$ à partir de l'état 0 dans l'ε–AF de l'exemple 3, et, les mêmes tentatives dans l'AFDC équivalent.



Démonstration de 2).

Rappelons que $u \in \mathcal{L}(\mathbf{A})$ ssi il existe $s \in I$ et $r \in F$ tels que $(s,u) \overset{*}{\vdash} (r,\varepsilon)$.

Nous allons vérifier successivement les deux implications :

- Soient $s \in I \subseteq cl(I)$ et $r \in F$ tels que $(s,u) \overset{*}{\vdash} (r,\varepsilon)$. $P(u)$ implique alors que $r \in cl(I) \bullet u$ et donc que $r \in (cl(I) \bullet u) \cap F$: cet ensemble n'est donc pas vide!
- Réciproquement, soit $r \in (cl(I) \bullet u) \cap F$:
  - $P(u)$ appliquée à cette situation signifie l'existence d'un $t \in cl(I)$ tel que $(t,u) \overset{*}{\vdash} (r,\varepsilon)$.
  - $t \in cl(I)$ signifie l'existence d'un $s \in I$ vérifiant $(s,\varepsilon) \overset{*}{\vdash} (t,\varepsilon)$, et donc $(s,u) \overset{*}{\vdash} (t,u)$.

Il ne reste plus qu'à composer ces dérivations pour conclure.

Ceci termine la démonstration de la propriété de détermination car la propriété 2) signifie exactement que $\mathcal{L}(DC(\mathbf{A})) = \mathcal{L}(\mathbf{A})$.

5.4 – Fin de la démonstration du théorème de Kleene.

Il nous reste à démontrer que (a) implique (b) dans le théorème :

THEOREME (Kleene).
Pour tout $L \subseteq \mathcal{A}^*$ les deux conditions suivantes sont équivalentes :

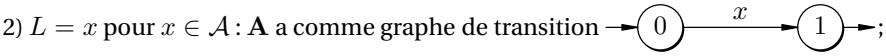
(a) L est régulier;
(b) il existe un AFDC $\mathbf{A}$ tel que $L = \mathcal{L}(\mathbf{A})$.

Pour cela il faut construire un AFDC reconnaissant L pour chaque langage régulier L : en fait, nous construisons un ε–AF (à une seule entrée) $\mathbf{A} = (Q, \mathcal{A}, \delta, q_0, F)$ reconnaissant L et la propriété de détermination fera le reste.

Cette construction se fait par induction sur la définition des langages réguliers (pour être rigoureux, il faudrait utiliser des expressions régulières).

1) $L = \emptyset$: il suffit que $F = \emptyset$ pour que $\mathcal{L}(\mathbf{A}) = \emptyset$, par exemple 

;

2) $L = x$ pour $x \in \mathcal{A}$: $\mathbf{A}$ a comme graphe de transition 

;

3) $L = L_1 + L_2$ où L_1 est reconnu par $\mathbf{A}_1 = (Q_1, \mathcal{A}, \delta_1, q_1, F_1)$ et L_2 par $\mathbf{A}_2 = (Q_2, \mathcal{A}, \delta_2, q_2, F_2)$, on peut supposer sans perte de généralité que $Q_1 \cap Q_2 = \emptyset$. $\mathbf{A}$ est défini par :

- $Q = q_0 + Q_1 + Q_2$ où q_0 est un nouvel état;
- q_0 est le nouvel état qui vient d'être introduit;
- δ prolonge δ_1 et δ_2 par $\delta(q_0,\varepsilon) = q_1 + q_2$;

– $F = F_1 + F_2$.

4) $L = L_1 L_2$ où L_1 et L_2 sont comme ci–dessus. $\mathbf{A}$ est défini par :

– $Q = Q_1 + Q_2$;
– $q_0 = q_1$;
– δ prolonge δ_1 et δ_2 par $\delta(q, \varepsilon) = \delta_1(q, \varepsilon) + q_2$ pour tout $q \in F_1$;
– $F = F_2$.

5) $L = L_1^*$ où L_1 est comme ci–dessus. $\mathbf{A}$ est défini par :

– $Q = q_0 + Q_1$ où q_0 est un nouvel état;
– q_0 est le nouvel état qui vient d'être introduit;
– δ prolonge δ_1 par $\delta(q_0, \varepsilon) = q_1$ et $\delta(q, \varepsilon) = \delta_1(q, \varepsilon) + q_0$ pour tout $q \in F_1$;
– $F = q_0$.

On vérifie sans difficulté que ces constructions sont correctes, c'est–à–dire que l'on a bien $\mathcal{L}(\mathbf{A}) = L$ dans chaque cas.

Remarques.

• Les ε–transitions jouent un rôle essentiel dans le cas de la concaténation : lorsqu'il a pénétré dans $\mathbf{A}_2$ un chemin ne doit pas pouvoir entrer à nouveau dans $\mathbf{A}_1$!

• On peut faire une remarque analogue à propos de l'itération : on ne peut pas conserver q_1 comme entrée de $\mathbf{A}$ car celle–ci est aussi une sortie, or la présence éventuelle d'une boucle autour de q_1, permettrait alors de reconnaître des mots qui ne sont pas dans L_1^*.

• La construction relative à $L_1 + L_2$ aurait été simplifiée si l'on ne s'était pas limité à une seule entrée (il suffisait alors de juxtaposer les deux ε–AF) mais celles qui sont relatives à $L_1 L_2$ et L_1^* auraient été notablement compliquées.

Des ε–AF reconnaissant respectivement L_1 et L_2 dans les constructions ci–dessus, ne sont pas supposés avoir d'autres propriétés! Ceci est intéressant dans la pratique de cette méthode, car celle-ci a la fâcheuse tendance à introduire de nombreuses ε–transitions, rendant la détermination du résultat final très périlleuse. On aura intérêt, au cours d'une construction, à simplifier des résultats intermédiaires, par exemple par détermination. D'autres constructions, intéressantes à des titres divers, sont proposées dans les exercices 18 et 19.

Effectivité du théorème de Kleene.

La démonstration du théorème de Kleene est basée sur des algorithmes. On peut donc préciser l'énoncé de ce théorème en signalant que :

il existe une correspondance effective entre les expressions régulières et les AFDC.

Il n'est pas possible d'étendre ce résultat aux langages réguliers eux–mêmes, comme on le verra dans la section 6.2.2.

5.5 – ε–AF et systèmes d'équations linéaires.

Le système d'équations linéaires, analogue à celui qui nous a servi à faire la preuve du fait que (a) implique (b) dans le théorème de Kleene, peut encore s'écrire dans le cas plus général des ε–AF. C'en est une transposition facile mais qui pose des problèmes techniques plus délicats (cf. exercice 20) : nous nous contenterons ici d'énoncer les résultats.

Soit $\mathbf{A} = (Q, \mathcal{A}, \delta, I, F)$ un ε–AF. Pour tout $q \in Q$ on considère l'ensemble $Rec(q)$ des mots reconnus par l'ε–AF $\mathbf{A}(q) = (Q, \mathcal{A}, \delta, q, F)$ obtenu en prenant un état $q \in Q$ comme unique entrée dans $\mathbf{A}$. On a alors, en étendant Rec aux ensembles d'états :

$$Rec(q) = \begin{cases} \sum_{x \in \varepsilon + \mathcal{A}} xRec(\delta(q,x)) + \varepsilon & \text{si } q \in F, \\ \sum_{x \in \varepsilon + \mathcal{A}} xRec((\delta(q,x)) & \text{sinon.} \end{cases}$$

L'ensemble des égalités correspondant à tous les $q \in Q$ est un système d'équations linéaires dont les inconnues sont $X_q = Rec(q)$: nous admettrons que les $Rec(q)$ constituent la plus petite solution de ce système (qui ne vérifie plus la condition d'unicité dès qu'il y a effectivement des ε–transitions!). Par ailleurs, $\mathcal{L}(\mathbf{A})$ est $Rec(I)$, calculé dans la plus petite solution du système ci–dessus (cf. exercice 20).

Exemple 3 (suite).

Le système correspondant à l'ε–AF de l'exemple 3 est le suivant :

$$\begin{aligned} X_0 &= X_1 + aX_3 + \varepsilon \\ X_1 &= a(X_1 + X_2) + bX_3 \\ X_2 &= X_3 + aX_4 \\ X_3 &= bX_4 \\ X_4 &= X_0 \end{aligned}$$

Sa plus petite solution se calcule par la méthode exposée dans la section 4.2 du premier chapitre.

6 – Constructions sur les AFDC.

Toute construction sur les AFDC, est transposable aux langages réguliers, grâce au théorème de Kleene. Nous allons voir les plus importantes dans cette section.

6.1 – Rôle des sorties d'un AFDC.

Dans la section précédente, on a considéré tous les choix possibles de l'entrée dans un AFDC. Dans la présente section, on va agir de même à l'égard des sorties.

$(Q, \mathcal{A}, \bullet, q_0)$ étant fixé correctement, considérons, pour tout $F \subseteq Q$ l'AFDC $\mathbf{A}(F) = (Q, \mathcal{A}, \bullet, q_0, F)$, alors :

Propriétés.

1) $\mathcal{L}(\mathbf{A}(\emptyset)) = \emptyset$,
2) $\mathcal{L}(\mathbf{A}(Q)) = \mathcal{A}^*$,

et, pour tout $F \subseteq Q$ et tout $G \subseteq Q$:

3) $\mathcal{L}(\mathbf{A}(F+G)) = \mathcal{L}(\mathbf{A}(F)) + \mathcal{L}(\mathbf{A}(G))$,
4) $\mathcal{L}(\mathbf{A}(F \cap G)) = \mathcal{L}(\mathbf{A}(F)) \cap \mathcal{L}(\mathbf{A}(G))$,
5) $\mathcal{L}(\mathbf{A}(\overline{F})) = \overline{\mathcal{L}(\mathbf{A}(F))}$,

où $\overline{F}$ est le complémentaire de F par rapport à Q et $\overline{\mathcal{L}(\mathbf{A}(F))}$ celui de $\mathcal{L}(\mathbf{A}(F))$ par rapport à $\mathcal{A}^*$.

1) et 2) sont parfaitement évidentes. 3) est un application directe des définitions; en effet, pour tout $u \in \mathcal{A}^*$, on a successivement :

$$\begin{aligned} u \in \mathcal{L}(\mathbf{A}(F+G)) \text{ ssi } & q_0 \bullet u \in F + G \\ \text{ssi } & q_0 \bullet u \in F \text{ ou } q_0 \bullet u \in G \\ \text{ssi } & u \in \mathcal{L}(\mathbf{A}(F)) \text{ ou } u \in \mathcal{L}(\mathbf{A}(G)) \\ \text{ssi } & u \in \mathcal{L}(\mathbf{A}(F)) + \mathcal{L}(\mathbf{A}(G)) \end{aligned}$$

et de même pour 4) en remplaçant les disjonctions par des conjonctions. En remarquant que, pour $F \subseteq Q$ et $F' \subseteq Q$, la propriété $F' = \overline{F}$ équivaut à la conjonction "$F + F' = Q$ et $F \cap F' = \emptyset$" et en faisant une remarque analogue pour les parties de $\mathcal{A}^*$, on déduit la propriété 5) des quatre précédentes (cette propriété peut aussi se démontrer directement!).

En appliquant le théorème de Kleene, on obtient deux propriétés importantes des langages réguliers :

Propriétés des langages réguliers

Soient L et M deux langages réguliers sur $\mathcal{A}$, alors :

- *le complémentaire $\overline{L}$ de L est un langage régulier,*
- *l'intersection $L \cap M$ est un langage régulier.*

En effet, il suffit d'appliquer la propriété 5) qui précède à un AFDC reconnaissant L pour vérifier la première propriété. Pour la seconde : si L et M sont réguliers, leurs complémentaires $\overline{L}$ et $\overline{M}$ le sont aussi, ainsi que $N = \overline{L} + \overline{M}$. Le complémentaire de N est encore régulier or, on sait que $\overline{N} = L \cap M$!

6.2 – Produit d'AFDC.

La construction du produit de deux AFDC est importante et simple.

Le produit $\mathbf{A} \times \mathbf{B} = (Q, \mathcal{A}, \circ, q_0, F)$ des AFDC $\mathbf{A} = (Q^{\mathbf{A}}, \mathcal{A}, \underset{\mathbf{A}}{\bullet}, q_0^{\mathbf{A}}, F^{\mathbf{A}})$ et $\mathbf{B} = (Q^{\mathbf{B}}, \mathcal{A}, \underset{\mathbf{B}}{\bullet}, q_0^{\mathbf{B}}, F^{\mathbf{B}})$ est l'AFDC défini de la façon suivante :

Etats. $Q = Q^{\mathbf{A}} \times Q^{\mathbf{B}}$,

Action. $(p,q) \circ x = (p \underset{\mathbf{A}}{\bullet} x, q \underset{\mathbf{B}}{\bullet} x)$ pour tout $p \in Q^{\mathbf{A}}$ et tout $q \in Q^{\mathbf{B}}$,

Entrée. $q_0 = (q_0^{\mathbf{A}}, q_0^{\mathbf{B}})$,

Sorties. $F = F^{\mathbf{A}} \times F^{\mathbf{B}}$.

Le calcul des états de $\mathbf{A} \times \mathbf{B}$, effectué de proche en proche à partir de l'entrée, donne la partie accessible de cet AFDC.

La propriété la plus évidente de cette opération est $\mathcal{L}(\mathbf{A} \times \mathbf{B}) = \mathcal{L}(\mathbf{A}) \cap \mathcal{L}(\mathbf{B})$.

En effet, pour chaque $u \in \mathcal{A}^*$:

$$\begin{aligned} u \in \mathcal{L}(\mathbf{A} \times \mathbf{B}) \text{ ssi } & q_0 \circ u \in F \\ \text{ssi } & (q_0^{\mathbf{A}}, q_0^{\mathbf{B}}) \circ u \in F^{\mathbf{A}} \times F^{\mathbf{B}} \\ \text{ssi } & (q_0^{\mathbf{A}} \underset{\mathbf{A}}{\bullet} u, q_0^{\mathbf{B}} \underset{\mathbf{B}}{\bullet} u) \in F^{\mathbf{A}} \times F^{\mathbf{B}} \\ \text{ssi } & q_0^{\mathbf{A}} \underset{\mathbf{A}}{\bullet} u \in F^{\mathbf{A}} \text{ et } q_0^{\mathbf{B}} \underset{\mathbf{B}}{\bullet} u \in F^{\mathbf{B}} \\ \text{ssi } & u \in \mathcal{L}(\mathbf{A}) \text{ et } u \in \mathcal{L}(\mathbf{B}) \\ \text{ssi } & u \in \mathcal{L}(\mathbf{A}) \cap \mathcal{L}(\mathbf{B}) \end{aligned}$$

Ceci redonne le résultat précédent sur l'intersection de langages réguliers, mais d'une façon qui est certainement plus intéressante. Voyons une application importante de cette construction.

6.2.1 – Algorithme de Moore.

Soient $\mathbf{A}$ et $\mathbf{B}$ deux AFDC sur $\mathcal{A}$ et $\mathbf{A} \times \mathbf{B}$ leur produit.

Propriété.

L'égalité $\mathcal{L}(\mathbf{A}) = \mathcal{L}(\mathbf{B})$ *équivaut à la propriété :*

(†) *pour tout état* (q,r) *accessible de* $\mathbf{A} \times \mathbf{B}$ *on a* $q \in F^{\mathbf{A}}$ *ssi* $r \in F^{\mathbf{B}}$.

Un algorithme de comparaison issu de cette propriété est facile à imaginer : tester l'équivalence sur chaque nouvel état obtenu lors du calcul de la partie accessible de $\mathbf{A} \times \mathbf{B}$.

La propriété est facile à vérifier car (†) équivaut au fait que, pour tout $u \in \mathcal{A}^*$, on a $q_0^{\mathbf{A}} \underset{\mathbf{A}}{\bullet} u \in F^{\mathbf{A}}$ ssi $q_0^{\mathbf{B}} \underset{\mathbf{B}}{\bullet} u \in F^{\mathbf{B}}$, c'est–à–dire, $u \in \mathcal{L}(\mathbf{A})$ ssi $u \in \mathcal{L}(\mathbf{B})$.

6.2.2 – Action d'un langage.

L'action d'un AFDC s'étend aux langages : pour tout $q \in Q^{\mathbf{A}}$ et tout $M \subseteq \mathcal{A}^*$, $q \underset{\mathbf{A}}{\bullet} M \subseteq Q^{\mathbf{A}}$ est défini par :

$$s \in q \underset{\mathbf{A}}{\bullet} M \text{ ssi il existe } v \in M \text{ tel que } s = q \underset{\mathbf{A}}{\bullet} v.$$

Propriété.

Si $M = \mathcal{L}(\mathbf{B})$ alors :

$$s \in q \underset{\mathbf{A}}{\bullet} M \text{ ssi il existe } r \in F^{\mathbf{B}} \text{ tel que } (s,r) \text{ soit accessible dans } \mathbf{A} \times \mathbf{B} \text{ à partir de } (q, q_0^{\mathbf{B}})$$

pour tout $s \in Q^{\mathbf{A}}$.

En effet, l'existence d'un $r \in F^{\mathbf{B}}$ vérifiant la condition équivaut à celle d'un $v \in \mathcal{A}^*$ tel que $q_0^{\mathbf{B}} \underset{\mathbf{B}}{\bullet} v \in F^{\mathbf{B}}$ et $s = q \underset{\mathbf{A}}{\bullet} v$, ce qui est bien équivalent à $s \in q \underset{\mathbf{A}}{\bullet} M$.

Remarque.

Lorsque M est quelconque, $q \underset{\mathbf{A}}{\bullet} M$ reste bien défini, mais il se peut qu'on n'ait aucun moyen pour le calculer effectivement!

Application : Quotient d'un langage par un autre.

Pour tout $L \subseteq \mathcal{A}^*$ et tout $M \subseteq \mathcal{A}^*$ on définit $M^{-1}L \subseteq \mathcal{A}^*$ par

$$w \in M^{-1}L \text{ ssi il existe } v \in M \text{ tel que } vw \in L$$

pour tout $w \in \mathcal{A}^*$ (cf. exercice 1.12).

Propriété.

Si L est régulier alors $M^{-1}L$ l'est aussi.

Pour vérifier cela, considérons un AFDC $\mathbf{A} = (Q, \mathcal{A}, \bullet, q_0, F)$ tel que $L = \mathcal{L}(\mathbf{A})$, et posons $I = q_0 \bullet M$, alors l'AF

$$M^{-1}\mathbf{A} = (Q, \mathcal{A}, \bullet, I, F)$$

reconnaît le langage $M^{-1}L$. En effet, on a les équivalences suivantes :

$w \in \mathcal{L}(M^{-1}\mathbf{A})$	ssi	(def. de $\mathcal{L}(M^{-1}\mathbf{A})$)
	il existe $s \in q_0 \bullet M$ tel que $s \bullet w \in F$	
	ssi	(def. de $q_0 \bullet M$ et $s = q_0 \bullet v$)
	il existe $v \in M$ tel que $(q_0 \bullet v) \bullet w \in F$	
	ssi	(def de $\mathbf{A}$)
	$vw \in L$	

Remarque.

Lorsque M est régulier et que l'on dispose d'un AFDC le reconnaissant alors on peut effectivement calculer l'AF $M^{-1}\mathbf{A}$.

Si M est quelconque, $M^{-1}L$ est encore régulier, mais il se peut qu'on n'ait aucun moyen pour calculer effectivement $M^{-1}\mathbf{A}$, ni même aucun automate reconnaissant $M^{-1}L$, bien qu'on sache qu'il en existe!

7 – Minimisation des AFDC.

Parmi les AFDC reconnaissant un langage régulier donné L, il en existe dont le nombre d'états $|Q|$ est le plus petit possible (car un ensemble d'entiers, non vide, a un plus petit élément). Nous montrerons dans l'annexe qu'il n'en existe qu'un seul, au renommage près des états, que l'on appelle l'AFDC minimal de L.

Dans la présente section, nous décrivons la construction de l'AFDC minimal $Min(\mathbf{A}) = (Q', \mathcal{A}, \circ, q_0', F')$ équivalent à un AFDC $\mathbf{A} = (Q, \mathcal{A}, \bullet, q_0, F)$ donné : la justification de cette construction est donnée en annexe. Notons qu'elle ne s'applique que lorsque **tous les états de $\mathbf{A}$ sont accessibles** (l'élimination des états inaccessibles est déjà une étape vers la minimisation!). Il nous faut tout d'abord rappeler une notion qui joue un grand rôle ici.

Partition d'un ensemble.

• Une *partition* Π d'un ensemble $Q \neq \emptyset$ est un découpage de Q, c'est–à–dire un ensemble de parties $\Pi \subseteq \mathcal{P}(Q)$ qui vérifie :

- pour toute $S \in \Pi$: $S \neq \emptyset$;
- pour toute $S \in \Pi$ et toute $T \in \Pi$: $S = T$ ou $S \cap T = \emptyset$;
- pour tout $q \in Q$, il existe $S \in \Pi$ telle que $q \in S$.

Un élément d'une partition s'appelle souvent une *classe*.

• Soit Π est une partition de Q alors, tout $q \in Q$ appartient à un élément de Π et à un seul que l'on notera $[q]$ (la classe de q). Pour toute $S \in \Pi$ et tout $q \in Q$ on a donc la propriété :

$$[q] = S \text{ ssi } q \in S.$$

7.1 – Construction de Q'.

Q' est la partition Π de Q calculée par l'algorithme suivant.

Dans celui–ci, on utilise l'action réciproque (dans $\mathbf{A}$) $q \bullet x^{-1} \subseteq Q$ de $x \in \mathcal{A}$ sur $q \in Q$ qui est définie par :

$$r \in q \bullet x^{-1} \text{ ssi } r \bullet x = q$$

et son extension à toute $S \subseteq Q$, qui vérifie évidemment la propriété :

$$r \in S \bullet x^{-1} \text{ ssi } r \bullet x \in S.$$

Calcul d'une partition de Q

• *La valeur initiale de* Π *est* $\Pi = \begin{cases} \{Q\} & \text{si } F = Q \text{ ou } F = \emptyset, \\ \{F, Q - F\} & \text{sinon.} \end{cases}$

• *tant qu'il existe* $S \in \Pi$, $T \in \Pi$ *et* $x \in \mathcal{A}$ *tels que*

$$S_1 = S \cap (T \bullet x^{-1}) \neq \emptyset \text{ et } S_2 = S - (T \bullet x^{-1}) \neq \emptyset$$

on raffine Π *en remplaçant* S *par les deux classes* S_1 *et* S_2.

L'algorithme se termine puisque l'ensemble Q est fini; ceci se produit dès que la condition d'entrée dans la boucle n'est plus satisfaite, c'est–à–dire lorsque l'on a

$$(\dagger) \qquad S \cap (T \bullet x^{-1}) = \emptyset \text{ ou } S \subseteq T \bullet x^{-1}$$

pour toute $S \in \Pi$, tout $T \in \Pi$ et tout $x \in \mathcal{A}$.

7.2 – Construction de l'AFDC minimal équivalent à A.

L'AFDC minimal $Min(\mathbf{A})$ équivalent à l'AFDC $\mathbf{A} = (Q, \mathcal{A}, \bullet, q_0, F)$ **dont tous les états sont accessibles** est défini par :

$Min(\mathbf{A})$

$Min(\mathbf{A})$ est l'AFDC $(Q', \mathcal{A}, \circ, q'_0, F')$ suivant :

- *Etats.* Q' est la partition Π calculée par l'algorithme ci–dessus.
- *Action.* $[q] \circ x = [q \bullet x]$.
- *Entrée.* $q'_0 = [q_0]$.
- *Sorties.* $[q] \in F'$ ssi $q \in F$.

Commentaires.

• Pour que cette définition soit celle d'un AFDC, il faut évidemment que l'action soit bien déterminée, c'est–à–dire que $[q \bullet x] = [r \bullet x]$ lorsque $[q] = [r]$: ceci est une conséquence de la propriété (†) de la partition Π. En effet, posons $S = [q] = [r]$ et $T = [q \bullet x]$: S et T sont des éléments de Π qui vérifient

1) $q \in S$,
2) $r \in S$,
3) $q \bullet x \in T$ (ce qui équivaut à $q \in T \bullet x^{-1}$).

Il faut montrer que $[r \bullet x] = T$.

1) et 3) impliquent $q \in S \cap T \bullet x^{-1}$: cette intersection n'est donc pas vide et, pour satisfaire (†) il est nécessaire que $S \subseteq T \bullet x^{-1}$, d'où $r \in T \bullet x^{-1}$ grâce à 2), ce qui est bien équivalent à $[r \bullet x] = T$.

• La définition de F' pourrait encore s'écrire : $[q] \in F'$ ssi $[q] \subseteq F$. En effet, $q \in F$ équivaut à $[q] \subseteq F$ puisque la partition Q' est un raffinement de la partition initiale : les éléments d'une classe sont ou bien tous dans F, ou bien tous en dehors de F.

Exemple.

Nous allons "minimiser" l'AFDC dont le graphe de transition est présenté ci–dessous.

La partition initiale est l'ensemble des classes définies par $\overset{0}{\equiv}$:

$$\Pi = \{0+5, 1+2+3+4\}$$

On a $(0+5) \bullet a^{-1} = 0+5$, qui ne permet pas de raffiner Π, puis

$$(0+5) \bullet b^{-1} = 2+3\,,$$

qui permet de raffiner Π en :

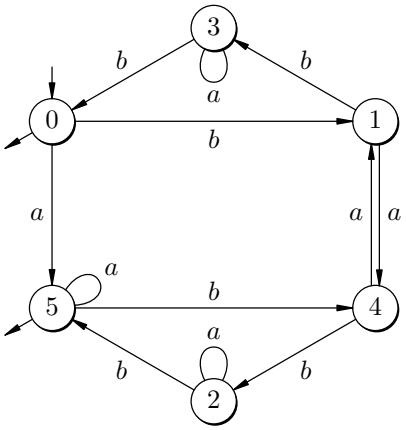
$$\Pi = \{0+5, 1+4, 2+3\}.$$

Enfin

$$(1+4) \bullet a^{-1} = 1+4, (1+4) \bullet b^{-1} = 0+5,$$
$$(2+3) \bullet a^{-1} = 2+3, (2+3) \bullet b^{-1} = 1+4$$

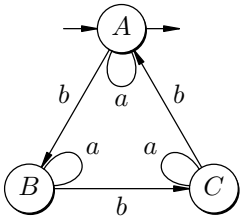
ne permettent plus de raffiner la partition, on a donc

$$Q' = \Pi = \{0+5, 1+4, 2+3\}.$$



La table de $Min(\mathbf{A})$ se calcule alors facilement :

e/s	S	$S \circ a$	$S \circ b$
$\leftrightarrow$	$A = 0+5$	A	B
	$B = 1+4$	B	C
	$C = 2+3$	C	A



Remarque.

Le calcul de $Min(\mathbf{A})$ est effectif. Ceci permet de concevoir un algorithme capable de décider si $\mathcal{L}(\mathbf{A}) = \mathcal{L}(\mathbf{B})$, basé sur la comparaison de $Min(\mathbf{A})$ et $Min(\mathbf{B})$.

8 – Annexe : les ADC et l'algorithme de minimisation.

Les automates déterministes et complets (ADC) ont été définis à la section **2** : dans tout le début de cette annexe, nous ne supposerons pas que l'ensemble de ses états est fini. Par contre, nous supposerons que **tous les états sont accessibles** à partir de l'entrée, dans tous les ADC que nous considérerons dans cette annexe.

8.1 – Comparaison des ADC.

Soient $\mathbf{A} = (Q, \mathcal{A}, \bullet, q_0, F)$ et $\mathbf{B} = (R, \mathcal{A}, \circ, r_0, G)$ deux ADC, sur le même alphabet fini $\mathcal{A}$, et dont tous les états sont accessibles, alors :

Un morphisme $f : \mathbf{A} \to \mathbf{B}$ est une application $f : Q \to R$ qui vérifie les conditions suivantes :

1) $f(q_0) = r_0$,
2) pour tout $q \in Q$ et tout $x \in \mathcal{A}$: $f(q \bullet x) = f(q) \circ x$,
3) pour tout $q \in Q$: $q \in F$ ssi $f(q) \in G$.

Les propriétés d'un morphisme d'ADC qui nous intéressent sont les suivantes :

a) $f(q \bullet u) = f(q) \circ u$ pour tout $q \in Q$ et tout $u \in \mathcal{A}^*$,
b) $\mathcal{L}(\mathbf{A}) = \mathcal{L}(\mathbf{B})$,
c) f est une application surjective.
d) Pour chaque couple $\mathbf{A}$, $\mathbf{B}$ d'ADC comme ci–dessus, il existe au plus un morphisme $\mathbf{A} \to \mathbf{B}$.

Vérifions ces propriétés.

a) découle directement des définitions et de 2).
Les équivalences suivantes font la preuve de b)

$$\begin{aligned} u \in \mathcal{L}(\mathbf{A}) &\text{ ssi } q_0 \bullet u \in F && \text{(par définition)} \\ &\text{ssi } f(q_0 \bullet u) \in G && \text{(par 3))} \\ &\text{ssi } r_0 \circ u \in G && \text{(par a) et 1))} \\ &\text{ssi } u \in \mathcal{L}(\mathbf{B}) && \text{(par définition)} \end{aligned}$$

c) Tout $r \in R$ étant accessible, il existe $u \in \mathcal{A}^*$ tel que $r = r_0 \circ u$: on a alors $f(q) = r$ pour $q = q_0 \bullet u$, ce qui signifie bien que f est surjective.
Regardons en détail la propriété d'unicité d).
A cet effet, considérons l'application $\beta : \mathcal{A}^* \to R$ définie par $\beta(u) = r_0 \circ u$.
Si $f : \mathbf{A} \to \mathbf{B}$ est un morphisme, les conditions 1) et 2) et donc aussi a) impliquent que l'on a

$$\text{si } q = q_0 \bullet u \text{ alors } f(q) = \beta(u).$$

Cette dernière égalité définit f de façon unique, car tout état de $\mathbf{A}$ est accessible.

Réciproquement, l'application β ne définit une application f par la propriété précédente que dans la mesure où $\beta(u)$ ne dépend que de $q = q_0 \bullet u$ et non pas de u lui–même, c'est–à–dire que lorsqu'elle vérifie la condition

$$\text{(@)} \qquad \text{si } q_0 \bullet u = q_0 \bullet v \text{ alors } \beta(u) = \beta(v)$$

quels que soient u et $v \in \mathcal{A}^*$.

Cette remarque est la clef des constructions qui vont suivre.

Rappels sur les cardinaux.

La notion de cardinal généralise celle de "nombre d'éléments" au cas des ensembles quelconques.

- Deux ensembles ont le même cardinal ssi il existe une bijection de l'un vers l'autre.

- On peut choisir, parmi les ensembles ayant le même cardinal, un ensemble particulier que l'on appelle "le cardinal" de ces ensembles : le cardinal d'un ensemble X est noté $|\,X\,|$. Par exemple, tous les ensembles finis ayant n éléments ont pour cardinal l'ensemble des entiers naturels $< n$, ensemble que l'on confondra avec n lui–même.

- Les cardinaux sont comparables entre eux par la relation : $|\,X\,| \geq |\,Y\,|$ ssi il existe une surjection $X \to Y$. Supposons que $|\,X\,| \geq |\,Y\,|$ alors $|\,Y\,|$ est fini lorsque $|\,X\,|$ l'est, ce qui, par contraposition, signifie aussi que $|\,X\,|$ est infini lorsque $|\,Y\,|$ l'est.

Si l'on considère qu'un ADC est d'autant plus simple que l'ensemble de ses états a un cardinal plus petit, on est conduit à poser la définition suivante.

Définition de la relation $\mathbf{A} \geq \mathbf{B}$

Soient $\mathbf{A}$ et $\mathbf{B}$ deux ADC sur l'alphabet $\mathcal{A}$ dont tous les états sont accessibles, alors

$\mathbf{A} \geq \mathbf{B}$ ssi il existe un morphisme $\mathbf{A} \to \mathbf{B}$.

On doit remarquer que $\mathbf{A} \geq \mathbf{B}$ implique $|Q| \geq |R|$.

Nous dirons que $\mathbf{A}$ et $\mathbf{B}$ sont *isomorphes* lorsqu'il existe un morphisme $f : \mathbf{A} \to \mathbf{B}$ où f est une bijection (l'application réciproque de f définit elle aussi un morphisme). On peut dire que deux ADC isomorphes sont "identiques au renommage près des états".

Nous pouvons maintenant aborder l'objet principal de la présente annexe, à savoir, la démonstration de la propriété suivante :

Propriété de comparaison.

Soit $L \subseteq \mathcal{A}^*$. Alors, l'ensemble des ADC reconnaissant L et dont tous les états sont accessibles possède un élément maximal $\mathbf{M}(L)$ et un élément minimal $\mathbf{m}(L)$ (relativement à la relation $\geq$). Pour tout ADC $\mathbf{A}$ reconnaissant L et dont tous les états sont accessibles on a donc :

$$\mathbf{M}(L) \geq \mathbf{A} \geq \mathbf{m}(L).$$

La construction de $\mathbf{m}(L)$, qui nous intéresse le plus, peut se faire effectivement à partir d'un ADC reconnaissant L :

Minimisation des ADC.

A partir de tout ADC $\mathbf{A}$ reconnaissant L, dont tous les états sont accessibles, il est possible de construire effectivement un ADC $Min(\mathbf{A}) \leq \mathbf{A}$ qui est isomorphe à $\mathbf{m}(L)$ et donc minimal.

L'annexe se termine par la justification de l'algorithme de minimisation qui a été donné dans la section précédente.

8.2 – Propriété de comparaison.

Soit $L \subseteq \mathcal{A}^*$ un langage quelconque.

8.2.1 – Construction de $\mathbf{M} = \mathbf{M}(L)$.

Cet ADC est défini de la façon la plus banale possible : $\mathbf{M} = (\mathcal{A}^*, \mathcal{A}, \cdot, \varepsilon, L)$, où $\cdot : \mathcal{A}^* \times \mathcal{A} \to \mathcal{A}^*$ est l'opération d'adjonction d'un caractère à droite, qui définit l'action de concaténation.
Tous les états de $\mathbf{M}$ sont accessibles, à partir de l'entrée ε, puisque $u = \varepsilon \cdot u$!
Enfin, $\mathcal{L}(\mathbf{M}) = L$ puisque $\varepsilon \cdot u \in L$ ssi $u \in L$!

Soit $\mathbf{A} = (Q, \mathcal{A}, \bullet, q_0, F)$ un ADC reconnaissant L : il faut montrer que $\mathbf{M} \geq \mathbf{A}$. Or, la condition (@) est trivialement vraie et, si $f : \mathcal{A}^* \to Q$ désigne l'application ainsi définie, on a bien $u \in L$ ssi $f(u) = q_0 \bullet u \in L$ pour tout $u \in \mathcal{A}^*$, donc f vérifie 3) et est bien le morphisme $\mathbf{M} \to \mathbf{A}$ cherché.

8.2.2 – Construction de $\mathbf{m} = \mathbf{m}(L)$.

Considérons la relation définie sur $\mathcal{A}^*$ par

$$u \sim v \text{ ssi } \forall w \in \mathcal{A}^*(uw \in L \Leftrightarrow vw \in L).$$

- Les propriétés suivantes sont des cas particuliers de la définition :
  (I) $u \sim v$ implique $(u \in L \Leftrightarrow v \in L)$.
  (II) $u \sim v$ implique $ux \sim vx$ pout tout $x \in \mathcal{A}$.
- $\sim$ est une relation d'équivalence, c'est–à–dire que
  - $u \sim u$,
  - $u \sim v$ implique $v \sim u$,
  - $u \sim v$ et $v \sim w$ implique $u \sim w$,

pour tout u, tout v et tout $w \in \mathcal{A}^*$.

• Rappelons qu'une *partition* P d'un ensemble $X \neq \emptyset$ est un découpage de X, c'est–à–dire un ensemble de parties $P \subseteq \mathcal{P}(X)$ qui vérifie :

- pour toute $A \in P$: $A \neq \emptyset$
- pour toute $A \in P$ et toute $B \in P$: $A = B$ ou $A \cap B = \emptyset$
- pour tout $x \in X$, il existe $A \in P$ telle que $x \in A$

La classe d'un $u \in \mathcal{A}^*$ pour $\sim$ est la partie $\overline{u} \subseteq \mathcal{A}^*$ définie par $v \in \overline{u}$ ssi $u \sim v$. On sait que ces classes définissent une partition de $\mathcal{A}^*$, de plus $\overline{u} = \overline{v}$ équivaut à $u \sim v$.

Nous sommes maintenant en mesure de définir $\mathbf{m} = (R, \mathcal{A}, \circ, r_0, G)$:

- *Etats.* R est l'ensemble des $\overline{u}$ (des mots équivalents se trouvent confinés dans un même état)
- *Action.* $\overline{u} \circ x = \overline{ux}$ (ceci définit une application $d : R \times \mathcal{A} \to E$ grâce à (II))
- *Entrée.* $r_0 = \overline{\varepsilon}$
- *Sorties.* $G \subseteq R$ est défini par $\overline{u} \in G$ ssi $\overline{u} \subseteq L$.

Ceci définit bien un ADC dont tous les états sont accessibles à partir de $r_0 = \overline{\varepsilon}$ puisque $\overline{u} = \overline{\varepsilon u} = \overline{\varepsilon} \circ u$ pour tout $u \in \mathcal{A}^*$.

Soit $\mathbf{A}$ un ADC reconnaissant L et dont tous les états sont accessibles : nous devons montrer que $\mathbf{A} \geq \mathbf{m}$.

• Avec les données actuelles, la condition (@) s'écrit : $q_0 \bullet u = q_0 \bullet v$ implique $\overline{u} = \overline{v}$, pour tout $u \in \mathcal{A}^*$ et tout $v \in \mathcal{A}^*$, ce qui est vrai puisque

$$\begin{aligned} q_0 \bullet u = q_0 \bullet v \text{ implique } & \forall w \in \mathcal{A}^* (q_0 \bullet uw = q_0 \bullet vw) \\ \text{implique } & \forall w \in \mathcal{A}^* (uw \in L \Leftrightarrow vw \in L) \\ \text{implique } & u \sim v \end{aligned}$$

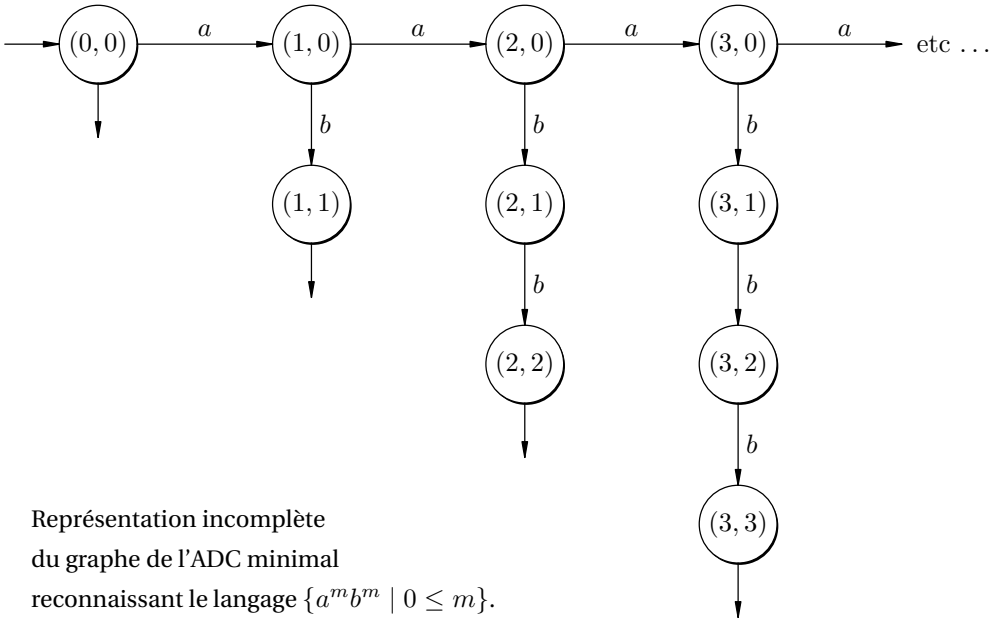
L'application $f : Q \to R$ ainsi définie est $f(q_0 \bullet u) = \overline{u}$. Cette application est un morphisme car on a $\overline{u} \subseteq L$ ssi $u \in L$ grâce à (I), donc :

$$q_0 \bullet u \in F \text{ ssi } u \in L \text{ ssi } \overline{u} \in G$$

pour tout $u \in \mathcal{A}^*$, ce qui signifie bien que f vérifie la condition 3).

Exemple.

Calculons l'ADC $\mathbf{m} = \mathbf{m}(L)$ reconnaissant le langage $L = \{a^m b^m \mid m \geq 0\}$.



Représentation incomplète
du graphe de l'ADC minimal
reconnaissant le langage $\{a^m b^m \mid 0 \leq m\}$.

- Calcul des états de $\mathbf{m}$. Soit $u \in (a+b)^*$, il est facile d'observer que :
  - si $u \notin fg(L)$ alors $v \sim u$ pour tout $v \notin fg(L)$. $\mathcal{A}^* - fg(L)$ constitue donc une classe d'équivalence, que nous désignerons par Φ,
  - si $u \in fg(L)$ alors $v \sim u$ ssi $v = u$, chaque élément de $fg(L)$ constitue donc une classe à lui seul. Les éléments de $fg(L)$ sont les mots $a^i b^j$ tels que $j \leq i$: la classe d'un tel mot $a^i b^j$ sera désignée par (i, j).
- Les transitions de $\mathbf{m}$ se calculent sans problème :

$$(i,j) \circ a = \begin{cases} (i+1,0) & \text{si } j = 0 \\ \Phi & \text{sinon} \end{cases} \qquad \text{et} \qquad \Phi \circ a = \Phi,$$

$$(i,j) \circ b = \begin{cases} (i,j+1) & \text{si } j < i \\ \Phi & \text{sinon} \end{cases} \qquad \text{et} \qquad \Phi \circ b = \Phi.$$

- L'entrée de $\mathbf{m}$ est $(0,0)$.
- Les sorties de $\mathbf{m}$ sont les états (m,m).

Pour simplifier la figure représentant le "début" de l'ADC $\mathbf{m}$, nous n'y avons fait figurer ni l'état Φ ni les transitions qui y aboutissent.

Remarque. $\mathbf{m}(L)$ n'est donc pas fini pour le langage $L = \{a^m b^m \mid m \geq 0\}$: ceci a pour conséquence qu'il n'existe pas d'ADC reconnaissant L qui soit fini, ce qui, par le théorème de Kleene, prouve que L **n'est pas régulier**.

8.3 – Minimisation des ADC.

Soit $\mathbf{A} = (Q, \mathcal{A}, \bullet, q_0, F)$ un ADC dont tous les états sont accessibles.

8.3.1 – Construction de $Min(\mathbf{A})$.

La relation $\equiv$ définie sur Q par :

$$q \equiv r \text{ ssi } f(q) = f(r)$$

est évidemment une relation d'équivalence.

En notant $[q]$ la classe de q pour cette relation, on peut définir l'ADC $Min(\mathbf{A}) = (Q', \mathcal{A}, \circ, q'_0, F')$ de la façon suivante :

- *Etats.* Q' est l'ensemble de ces classes.
- *Action.* $[q] \circ x = [q \bullet x]$.
- *Entrée.* $q'_0 = [q_0]$.
- *Sorties.* $[q] \in F'$ ssi $q \in F$.

L'application $f' : Q' \to R$ définie par $f'([q]) = f(q)$ est un morphisme d'ADC, de plus, elle est bijective "par construction", c'est donc bien un isomorphisme :

$$Min(\mathbf{A}) \text{ est isomorphe à } \mathbf{m}(L)$$

pour $L = \mathcal{L}(\mathbf{A})$.

Etude de la relation $\equiv$.

Soient $q = q_0 \bullet u$ et $r = q_0 \bullet v$ deux éléments de Q, on a les équivalences successives :

$$\begin{aligned} q \equiv r \text{ ssi } & f(q) = f(r) \\ \text{ssi } & \overline{u} = \overline{v} \\ \text{ssi } & u \sim v \\ \text{ssi } & \forall w \in \mathcal{A}^*(uw \in L \Leftrightarrow vw \in L) \\ \text{ssi } & \forall w \in \mathcal{A}^*(q_0 \bullet (uw) \in F \Leftrightarrow q_0 \bullet (vw) \in F) \\ \text{ssi } & \forall w \in \mathcal{A}^*(q \bullet w \in F \Leftrightarrow r \bullet w \in F). \end{aligned}$$

En particulier, on a la propriété :

(¶) $q \equiv r$ implique $q \bullet x \equiv r \bullet x$ pour tout $x \in \mathcal{A}$.

Définissons pour chaque entier naturel i :

$$q \overset{i}{\equiv} r \text{ ssi } \forall w \in (\varepsilon + \mathcal{A})^i (q \bullet w \in F \Leftrightarrow r \bullet w \in F).$$

($w \in (\varepsilon + \mathcal{A})^i$ signifie que $w \in \mathcal{A}^*$ et $|w| \leq i$)

Il est clair que chaque $\overset{i}{\equiv}$ est une relation d'équivalence et que

$$q \equiv r \text{ ssi } q \overset{i}{\equiv} r \text{ pour tout } i.$$

De plus

(I) $q \overset{0}{\equiv} r$ ssi $(q \in F \Leftrightarrow r \in F)$

(II) $q \overset{i+1}{\equiv} r$ ssi $q \overset{i}{\equiv} r$ et $\forall x \in \mathcal{A}(q \bullet x \overset{i}{\equiv} r \bullet x)$.

(II) exprime en particulier que $q \overset{i+1}{\equiv} r$ implique $q \overset{i}{\equiv} r$: si l'on désigne par $[q]_i$ la classe de q pour $\overset{i}{\equiv}$, cette implication signifie que $[q]_{i+1} \subseteq [q]_i$.

8.4 – L'algorithme de minimisation.

Nous revenons maintenant au cas des langages réguliers : nous supposerons donc que Q **est fini**, c'est–à–dire que $\mathbf{A}$ est un A**F**DC, jusqu'à la fin de cette annexe.

Sous cette condition, la suite des classes $[q]_i$ pour q fixé ne peut décroître indéfiniment! Il existe donc n tel que $i \geq n$ implique $q \overset{i}{\equiv} r \Leftrightarrow q \overset{n}{\equiv} r$ pour tout $q \in Q$ et tout $r \in Q$: il est clair que la relation $\overset{n}{\equiv}$ est exactement $\equiv$ (on peut vérifier que $n < |Q|$). Nous sommes maintenant en mesure de calculer la partition de Q en classes d'équivalence pour $\equiv$: l'application de (I) et (II) fournit un tel calcul, nous allons l'améliorer un peu.

Rappelons que l'action inverse $q \bullet x^{-1} \subseteq Q$ de $x \in \mathcal{A}$ sur $q \in Q$ est définie par

$$r \in q \bullet x^{-1} \text{ ssi } r \bullet x = q,$$

et que cette action inverse s'étend à toute $S \subseteq Q$ en posant, comme d'habitude :

$$r \in S \bullet x^{-1} \text{ ssi } r \bullet x \in S,$$

quel que soit $r \in Q$.

Calcul des états de l'AFDC $Min(\mathbf{A})$

- *La valeur initiale de* Π *est* $\Pi = \begin{cases} \{Q\} & \text{si } F = Q \text{ ou } F = \emptyset, \\ \{F, Q - F\} & \text{sinon.} \end{cases}$
- *tant qu'il existe* $S \in \Pi$, $T \in \Pi$ *et* $x \in \mathcal{A}$ *tels que*

$$S_1 = S \cap (T \bullet x^{-1}) \neq \emptyset \text{ et } S_2 = S - (T \bullet x^{-1}) \neq \emptyset$$

on raffine Π *en remplaçant* S *par les deux classes* S_1 *et* S_2.

L'algorithme se termine puisque l'ensemble Q est fini; ceci se produit dès que la condition d'entrée dans la boucle n'est plus satisfaite, c'est–à–dire lorsque l'on a

$$(\dagger) \qquad S \cap (T \bullet x^{-1}) = \emptyset \text{ ou } S \subseteq T \bullet x^{-1}$$

pour toute $S \in \Pi$, tout $T \in \Pi$ et tout $x \in \mathcal{A}$.

Propriété. *La valeur finale de* Π *est l'ensemble* Q' *des classes pour la relation* $\equiv$.

La démonstration se fait en deux temps :

1) Toute classe est incluse dans un élément de Π.
2) Tout élément de Π est inclus dans une classe.

1) Montrons que toute classe est incluse dans un élément de Π : pour cela il suffit de montrer que c'est vrai à chaque étape de l'exécution de l'algorithme. Plus précisément, il suffit de montrer qu'à chaque étape

$$q \equiv r \text{ implique } (q \in S \Leftrightarrow r \in S)$$

quels que soient $S \in \Pi$, $q \in Q$ et $r \in Q$.

- C'est vrai initialement car la partition est alors exactement celle que définit $\overset{0}{\equiv}$, or $q \equiv r$ implique $q \overset{0}{\equiv} r$.
- Supposons que S et T vérifient la propriété et que l'on ait x tel que

  $$S_1 = S \cap (T \bullet x^{-1}) \neq \emptyset \text{ et } S_2 = S - (T \bullet x^{-1}) \neq \emptyset$$

  Il suffit alors d'appliquer (¶) pour en déduire que S_1 et S_2 la vérifient aussi.

2) Réciproquement, considérons maintenant la valeur finale de Π : il faut montrer que pour tout entier i et tout $S \in \Pi$:

$$q \in S \text{ et } r \in S \text{ implique } q \overset{i}{\equiv} r.$$

La preuve se fait par récurrence sur i.

- Chaque $S \in \Pi$ est contenu dans un élément de la partition initiale, c'est–à–dire dans une classe pour $\overset{0}{\equiv}$: la propriété est donc vraie pour 0.
- Supposons la propriété vérifiée pour i et soient $S \in \Pi$, $q \in S$, $r \in S$ et $x \in \mathcal{A}$:

  α) L'hypothèse de récurrence appliquée à S implique que $q \overset{i}{\equiv} r$.

  β) Tout élément de Q est contenu dans un élément de Π car Π est une partition de Q : ainsi, existe–t–il $T \in \Pi$ tel que $q \bullet x \in T$, c'est–à–dire $q \in T \bullet x^{-1}$. On a donc $q \in S \cap T \bullet x^{-1} \neq \emptyset$, ce qui, par $(\dagger)$, entraine que $S \subseteq T \bullet x^{-1}$ et donc que $r \in T \bullet x^{-1}$, c'est–à–dire $r \bullet x \in T$.
  L'hypothèse de récurrence appliquée à T implique que $q \bullet x \overset{i}{\equiv} r \bullet x$.

  Grâce à (II), α) et β) permettent de conclure que $q \overset{i+1}{\equiv} r$.

Ceci termine la démonstration de la propriété de minimisation.

EXERCICES.

$\mathcal{A}$ désigne un alphabet fini.

Exercice 1.

Soit $\mathcal{A} = \{a, b\}$ un alphabet à deux lettres.

a) Donner une expression régulière de chacun des langages sur $\mathcal{A}$ suivants :

- ensemble des mots se terminant par aa;
- ensemble des mots comportant le facteur aaa;
- ensemble des mots commençant par ab et se terminant par bb;
- ensemble des mots ayant un nombre pair d'occurrences de a;
- ensemble des mots ne comportant pas le facteur aa.
- ensemble des mots ne comportant pas le facteur aaa.

b) Montrer que $(a+bb^*a^2)^*b^*(\varepsilon+a)$ est une expression régulière de l'ensemble des mots ne comportant pas le facteur bab (ε est une abréviation pour $\emptyset^*$).

Exercice 2. Conjugaison (cf. exercice 1.3.b).

Montrer que $Conj(L)$ est régulier pour tout langage régulier L.

Indication. Soit $\mathbf{A} = (Q, \mathcal{A}, \bullet, q_0, F)$ un AFDC et soit $L = \mathcal{L}(\mathbf{A})$. Pour chaque $q \in Q$ on considère les deux AFDC suivants :

$$\mathbf{B}(q) = (Q, \mathcal{A}, \bullet, q_0, q) \qquad \text{(la seule sortie est } q\text{)}$$
$$\mathbf{C}(q) = (Q, \mathcal{A}, \bullet, q, F) \qquad \text{(l'entrée est } q\text{)}$$

et on pose $M(q) = \mathcal{L}(\mathbf{B}(q))$ et $N(q) = \mathcal{L}(\mathbf{C}(q))$.

Montrer que l'on a $L = \sum_{q\in Q} M(q)N(q)$ et $Conj(L) = \sum_{q\in Q} N(q)M(q)$.

Exercice 3.

Soit $\mathcal{A} = a$ un alphabet à un seul symbole.

Pour chaque $i \in \{0, 1, 2\}$ on définit $L_i \subseteq a^*$ par : $u \in L_i$ ssi $|u| \; mod\, 3 = i$ pour tout $u \in a^*$.

Trouver un ensemble d'états Q, un état initial $q_0 \in Q$, une application $\bullet : Q \times \mathcal{A} \to Q$ et, pour chaque $i \in \{0, 1, 2\}$ un ensemble $F_i \subseteq Q$ de sorties, de telle façon que l'AFDC $\mathbf{A}_i = (Q, \mathcal{A}, \bullet, q_0, F_i)$ reconnaisse le langage L_i.

Exercice 4.

Soit a un symbole.

Montrer que pour tout langage régulier $L \subseteq a^*$, il existe deux langages **finis** A et B et un mot $\gamma \in a^*$ tels que $L = A + B\gamma^*$ (Réciproquement, un langage de cette forme est régulier!).

Indication. Trouver la forme générale des AFDC sur un alphabet à une seule lettre : le théorème de Kleene fera le reste.

Exercice 5. Lemme d'itération pour les langages réguliers.

a) Démontrer *le lemme d'itération* suivant :

si $L \subseteq \mathcal{A}^*$ est un langage régulier alors il existe un entier $N > 0$ tel que pour tout $u \in L$ vérifiant $|u| \geq N$, on peut trouver trois mots u_1, v et u_2 sur $\mathcal{A}$ tels que l'on ait les propriétés

1) $u = u_1 v u_2$,

2) $|v| > 0$ et $|v| \leq N$,
3) pour tout entier naturel k, $u_1 v^k u_2 \in L$.

Indication. Observer attentivement les *longs chemins* que l'on peut parcourir dans un AFDC reconnaissant L.

b) En déduire que les langages suivants ne sont pas réguliers :

- $L_1 = \{a^m b^m \mid m \geq 0\}$
- $L_2 = \{a^{m^2} \mid m \geq 0\}$

Indication. Montrer que ces langages ne vérifient pas la conclusion du lemme d'itération : pour un langage L, la négation du lemme d'itération peut s'exprimer de la façon suivante :

pour tout entier $N > 0$, il existe $u \in L$ vérifiant $|u| \geq N$, tel que quels que soient u_1, v et u_2 vérifiant

1) $u = u_1 v u_2$,
2) $|v| > 0$ et $|v| \leq N$,

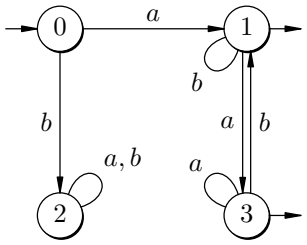
il existe un entier naturel k tel que $u_1 v^k u_2 \notin L$.

c) On considère les langages $L = (a+b)^* ba(a+b)^*$ et $M = \{a^m b^m \mid m \geq 0\}$.

Montrer que le langage $L + M$ vérifie la conclusion du lemme d'itération ci–dessus mais qu'**il n'est pas régulier**.

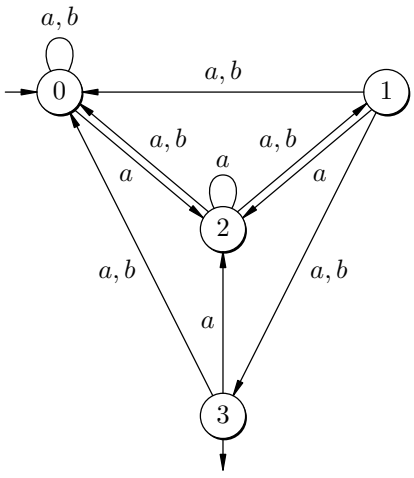
Exercice 6. (cf. exercices 1.4, 1.5 et 1.6)

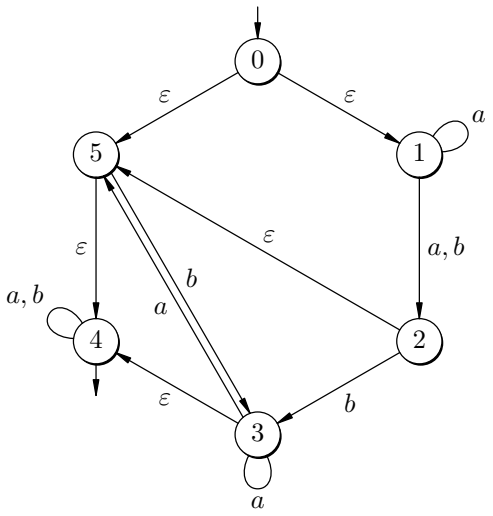
Soit f l'une des applications fg, fd, $fact$.
a) Montrer que f transforme un langage régulier en un langage régulier, en utilisant les propriétés démontrées dans les exercices cités.
b) Construire un AF reconnaissant $f(L)$ à partir d'un AFDC reconnaissant L. Appliquer cette construction à l'AFDC ci–contre.



Exercice 7.

Déterminer l'AF et l'ε–AF suivants :





Exercice 8.

Construire des AFDC qui reconnaissent respectivement chacun des langages décrits dans l'exercice 1.a).

Indication. On pourra commencer par construire un ε–AF reconnaissant "visiblement" ledit langage ou son complémentaire.

Exercice 9.

Soit $\mathcal{A} = a + b$ un alphabet à deux symboles.

a) Reprendre l'exercice 3 ci–dessus, mais cette fois pour les $L_i \subseteq \mathcal{A}^*$ définis par : $u \in L_i$ ssi $|u|_a \; mod \, 3 = i$ pour tout $u \in \mathcal{A}^*$.

b) Construire un AFDC sur $\mathcal{A}$ qui reconnaît l'ensemble des mots $u \in \mathcal{A}^*$ ne comportant pas le facteur bab et tels que $|u|_a \bmod 3 = 2$.

Exercice 10.

a) Soit $\mathbf{A}$ un AFDC sur un alphabet $\mathcal{A}$ et posons $L = \mathcal{L}(\mathbf{A})$.

Construire un ε–AF qui reconnaît l'ensemble des mots sur $\mathcal{A}$ dont aucun facteur n'est un élément de L.

b) Appliquer a) pour construire un AFDC reconnaissant l'ensemble des mots sur $\{a, b\}$ ne comportant pas le facteur bab, puis utiliser cet AFDC pour redémontrer le résultat de l'exercice 1.b) ci–dessus.

Exercice 11. Image par une substitution.

Soit $f : \mathcal{A} \to \mathcal{P}(\mathcal{B}^*)$ une substitution telle que $f(x)$ est régulier pour tout $x \in \mathcal{A}$.

a) Montrer que $f(L)$ est régulier pour tout $L \subseteq \mathcal{A}^*$ régulier, sans utiliser aucun AF.

b) Soit $\mathbf{A}$ un AFDC reconnaissant L et soit, pour chaque $x \in \mathcal{A}$, $\mathbf{A}(x)$ un AFDC reconnaisant $f(x)$: construire un ε–AF reconnaissant $f(L)$.

Appliquer cette construction à la substitution sm (cf. exercice 1.17)

Exercice 12. Image inverse par une substitution.

Soit $f : \mathcal{A} \to \mathcal{P}(\mathcal{B}^*)$ une substitution et soit $f^{-1} : \mathcal{B}^* \to \mathcal{P}(\mathcal{A}^*)$ son application inverse (cf. exercice 1.15), soit enfin $\mathbf{B} = (Q, \mathcal{B}, \underset{\mathbf{B}}{\bullet}, q_0, F)$ un AFDC : on considère alors l'ε–AF $\mathbf{A} = (Q, \mathcal{A}, \underset{\mathbf{A}}{\bullet}, q_0, F)$ dont l'action est définie par $q \underset{\mathbf{A}}{\bullet} x = q \underset{\mathbf{B}}{\bullet} f(x)$ (cf. section 6.2.2).

a) Montrer que si $M = \mathcal{L}(\mathbf{B})$ alors $\mathcal{L}(\mathbf{A}) = f^{-1}(M)$.

b) Etudier le cas où $f(x)$ est un langage régulier pour tout $x \in \mathcal{A}$.

Exercice 13.

Montrer, en utilisant les résultats de l'exercice 1.14, que l'image inverse d'un langage régulier par une substitution alphabétique est un langage régulier.

Exercice 14.

Appliquer les exercices 11 et 12 ci–dessus pour montrer que chacune des applications tc et usd de l'exercice 1.15 transforme un langage régulier en un langage régulier.

Exercice 15.

Il est possible de démontrer les résultats de l'exercice précédent par la construction d'AF.

Soient $\mathbf{A} = (Q, \mathcal{A}, \bullet, q_0, F)$ un AFDC et $L = \mathcal{L}(\mathbf{A})$.

a) Considérons l'AFDC $\mathbf{B} = (Q + (Q \times \mathcal{A}), \mathcal{A}, \circ, q_0, F)$ dont l'action est définie par :

$q \circ x = (q, x) \in Q \times \mathcal{A}$ pour tout $q \in Q$ et tout $x \in \mathcal{A}$;
$(q, x) \circ y = q \bullet yx$ pour tout $q \in Q$, tout x et tout $y \in \mathcal{A}$.

Montrer que $\mathcal{L}(\mathbf{B}) = tc(L)$.

b) Construire un AFDC $\mathbf{B}$ analogue au précédent et pour lequel on a $\mathcal{L}(\mathbf{B}) = usd(L)$.

Exercice 16.

Pour tout $L \subseteq \mathcal{A}^*$ on définit $R(L) \subseteq \mathcal{A}^*$ par

$$u \in R(L) \text{ ssi il existe } v \in \mathcal{A}^* \text{ tel que } |v| = |u| \text{ et } uv \in L$$

pour tout $u \in \mathcal{A}^*$.
Le but de cet exercice est de montrer que $R(L)$ est régulier pour tout langage régulier L.

a) Appliquer directement la définition précédente pour calculer $R(\varepsilon + aba^*)$.

b) Soit $\mathbf{A} = (Q, \mathcal{A}, \bullet, q_0, F)$ un AFDC, on considère l'AF $\mathbf{R}$ qui est la partie accessible (on a aussi intérêt à ne conserver que les états productifs) de l'AF $(Q \times Q, \mathcal{A}, \circ, I, G)$ défini de la façon suivante :

- $(s,t) \in (q,r) \circ x$ ssi $q \bullet x = s$ et $t \in r \bullet \mathcal{A}^{-1}$ *(action)*
- $I = \{(q_0, r) \mid r \in F\}$ *(ensemble des entrées)*
- $G = \{(q,q) \mid q \in Q\}$ *(ensemble des sorties)*

Appliquer cette construction à un AFDC reconnaissant le langage $\varepsilon + aba^*$.

Montrer que pour tout $u \in \mathcal{A}^*$, on a :

$$(s,t) \in (q,r) \circ u \text{ ssi } q \bullet u = s \text{ et il existe } v \in \mathcal{A}^* \text{ tel que } |v| = |u| \text{ et } t \bullet v = r$$

quels que soient q, r, s et $t \in Q$.

En déduire que si $L = \mathcal{L}(\mathbf{A})$ alors $\mathcal{L}(\mathbf{R}) = R(L)$.

Utiliser ce résultat pour vérifier le calcul précédent de $R(\varepsilon + aba^*)$.

Exercice 17. Mélange de langages (cf. exercice 1.16).

Soient $\mathbf{A} = (Q^{\mathbf{A}}, \mathcal{A}, \underset{\mathbf{A}}{\bullet}, q_0^{\mathbf{A}}, F^{\mathbf{A}})$ et $\mathbf{B} = (Q^{\mathbf{B}}, \mathcal{A}, \underset{\mathbf{B}}{\bullet}, q_0^{\mathbf{B}}, F^{\mathbf{B}})$ deux AFD.

On définit l'AF $\mathbf{C} = (Q, \mathcal{A}, \circ, q_0, F)$ à une seule entrée de la façon suivante :

- *Etats.* $Q = Q^{\mathbf{A}} \times Q^{\mathbf{B}}$,
- *Action.* $(q,r) \circ x = (q \underset{\mathbf{A}}{\bullet} x, r) + (q, r \underset{\mathbf{B}}{\bullet} x)$ pour tout $q \in Q^{\mathbf{A}}$ et tout $r \in Q^{\mathbf{B}}$,
- *Entrée.* $q_0 = (q_0^{\mathbf{A}}, q_0^{\mathbf{B}})$,
- *Sorties.* $F = F^{\mathbf{A}} \times F^{\mathbf{B}}$.

a) Montrer que si $L = \mathcal{L}(\mathbf{A})$ et $M = \mathcal{L}(\mathbf{B})$ alors $\mathcal{L}(\mathbf{C}) = mel(L, M)$.

b) Applications.

- Vérifier les calculs de la question **b)** de l'exercice 1.16 en utilisant la construction précédente.
- Construire un AF qui reconnaît l'ensemble des mots $u \in (a+b)^*$ tels que $(|u|_a - |u|_b) \bmod 3 = 1$ (cf. exercice 9.a).

Exercice 18. Langages locaux.

Un langage M sur un alphabet $\mathcal{B}$ est dit local ssi il existe $X \subseteq \mathcal{B}$, $Y \subseteq \mathcal{B}$ et $Z \subseteq \mathcal{B}^2$ tels que

$$M - \varepsilon = (X\mathcal{B}^* \cap \mathcal{B}^*Y) - \mathcal{B}^*Z\mathcal{B}^*.$$

Pour décider si un mot $u \in \mathcal{B}^*$ appartient à un langage local, il suffit de tester les facteurs de u dont la longueur est au plus 2, aussi les appelle–t–on souvent *langages 2–reconnaissables*.

Un langage local est évidemment régulier.
L'objet de cet exercice est de donner une réciproque à cette propriété :

tout langage régulier est l'image d'un langage local par une substitution strictement alphabétique.

Soit L un langage régulier ne contenant pas ε et soit $\mathbf{A} = (Q, \mathcal{A}, \bullet, q_0, F)$ un AFDC reconnaissant L. On considère un alphabet $\mathcal{B}$ dont les symboles sont des éléments de $Q \times \mathcal{A} \times Q$:

$$\mathcal{B} = \{[q,x,r] \mid q \bullet x = r\}.$$

On considère alors les langages $X \subseteq \mathcal{B}$, $Y \subseteq \mathcal{B}$ et $Z \subseteq \mathcal{B}^2$ définis par :

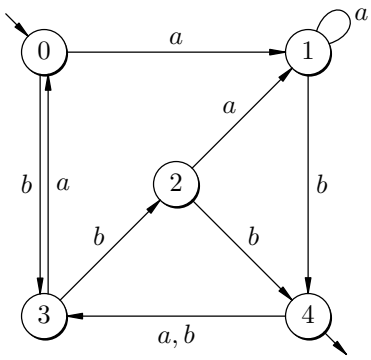
$[q,x,r] \in X$ ssi $q = q_0$,
$[q,x,r] \in Y$ ssi $r \in F$,
$[q,x,r][s,y,t] \in Z$ ssi $r \neq s$,

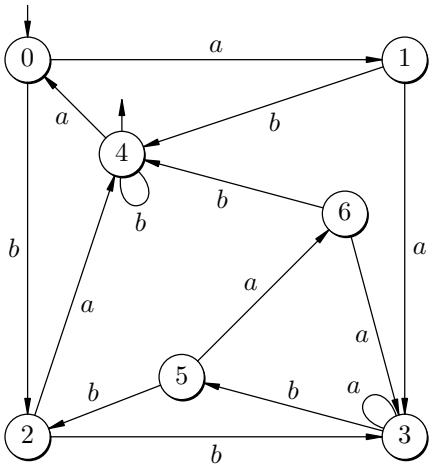
et la substitution $f : \mathcal{B} \to \mathcal{A}$ définie par $f([q,x,r]) = x$.

a) Montrer que $L = f((X\mathcal{B}^* \cap \mathcal{B}^*Y) - \mathcal{B}^*Z\mathcal{B}^*)$.
b) Appliquer la construction précédente à $L = b^2(a+b)^*a$.

Exercice 19.

Minimiser les AFDC suivants :





Exercice 20. AF spéciaux (AFS).

Un AFS sur $\mathcal{A}$ est un couple $(\mathbf{A}, \mathcal{E}ps)$ où :

- $\mathbf{A} = (Q, \mathcal{A}, \bullet, I, f)$ est un AF à une seule sortie, vérifiant les conditions suivantes :
  1) $f \notin I$,
  2) $f \bullet x = \emptyset$ pour tout $x \in \mathcal{A}$,
  3) pour tout $q \in Q - f$, il existe un unique $x \in \mathcal{A}$ tel que $q \bullet x \neq \emptyset$;
- $\mathcal{E}ps \subseteq \varepsilon$ (on a donc $\mathcal{E}ps = \emptyset$ ou $\mathcal{E}ps = \varepsilon$).

Le langage $\mathcal{L}(\mathbf{A}, \mathcal{E}ps) \subseteq \mathcal{A}^*$ reconnu par un AFS $(\mathbf{A}, \mathcal{E}ps)$ est défini par :

$$\mathcal{L}(\mathbf{A}, \mathcal{E}ps) = \mathcal{L}(\mathbf{A}) + \mathcal{E}ps$$

Remarques.

- Le fait que $\mathbf{A}$ a une seule sortie n'est pas très important et ne sert qu'à simplifier la construction **4)** ci–dessous.
- La condition 1) implique évidemment que $\varepsilon \notin \mathcal{L}(\mathbf{A})$: $\mathcal{E}ps$ sert à pallier cet éventuel défaut.
- La condition 2) signifie que l'on ne peut que sortir lorsqu'on est parvenu en f.
- La condition 3) implique que la table de $\mathbf{A}$ est très lacunaire : la ligne de f est vide et la ligne de chaque $q \neq f$ comporte une seule case non vide, ce qui est un avantage évident lorsqu'il s'agit de calculer un AFDC équivalent. On peut aussi profiter de cette condition pour simplifier les représentations de $\mathbf{A}$ (comment?).

L'objet de l'exercice est de vérifier que tout langage régulier est reconnaissable par un AFS.

a) Adapter l'algorithme de détermination des AF au cas des AFS ($\mathcal{E}ps$ doit intervenir!) et appliquer l'algorithme ainsi trouvé à des exemples simples d'AFS.

b) Les constructions suivantes permettent de construire un AFS reconnaissant un langage régulier $L \subseteq \mathcal{A}^*$: vérifier qu'elles sont correctes.

1) $L = \emptyset$ est reconnu par l'AFS $((f, \mathcal{A}, \bullet, \emptyset, f), \emptyset)$ à un seul état.

2) Pour tout $x \in \mathcal{A}$, $L = x$ est reconnu par l'AFS $((q_0 + f, \mathcal{A}, \bullet, q_0, f), \emptyset)$ à deux états, où $q_0 \bullet x = f$.

Pour la suite, on considère deux AFS $((Q_1, \mathcal{A}, \underset{1}{\bullet}, I_1, f_1), \mathcal{E}ps_1)$ et $((Q_2, \mathcal{A}, \underset{2}{\bullet}, I_2, f_2), \mathcal{E}ps_2)$, tels que $Q_1 \cap Q_2 = \emptyset$, et on appelle L_1 et L_2 les langages qu'ils reconnaissent respectivement.

3) $L = L_1 + L_2$ est reconnu par l'AFS construit de la façon suivante :

- $Q = (Q_1 - f_1) + (Q_2 - f_2) + f$ où $f \notin Q_1 + Q_2$ est un nouvel état,
- $I = I_1 + I_2$,
- f est le nouvel état introduit ci–dessus,
- pour chaque $i \in \{1,2\}$, si $q \in Q_i$ et $x \in \mathcal{A}$ sont tels que $f_i \in q \underset{i}{\bullet} x$
  alors $q \bullet x = (q \underset{i}{\bullet} x - f_i) + f$
  sinon $q \bullet x = q \underset{i}{\bullet} x$,

  (f_1 et f_2 ont été identifiées en f),
- $\mathcal{E}ps = \mathcal{E}ps_1 + \mathcal{E}ps_2$.

4) $L = L_1 L_2$ est reconnu par l'AFS construit de la façon suivante* :

- $Q = (Q_1 - f_1) + Q_2$,
- $I = I_1 + \mathcal{E}ps_1 I_2$,
- $f = f_2$,
- soient $q \in Q$ et $x \in \mathcal{A}$, alors :
  1) pour $q \in Q_1$: $q \bullet x = \begin{cases} (q \underset{1}{\bullet} x - f_1) + I_2 + f\mathcal{E}ps_2 & \text{si } f_1 \in q \underset{1}{\bullet} x, \\ q \underset{1}{\bullet} x & \text{sinon,} \end{cases}$
  2) pour $q \in Q_2$: $q \bullet x = q \underset{2}{\bullet} x$,

  (f_1 s'est multipliée pour s'indentifier respectivement à chacun des éléments de I_2),
- $\mathcal{E}ps = \mathcal{E}ps_1 \mathcal{E}ps_2$.

5) $L = L_1^*$ est reconnu par l'AFS construit de la façon suivante :

- $Q = Q_1$,
- $I = I_1$,
- $f = f_1$,
- soient $q \in Q_1$ et $x \in \mathcal{A}$, alors $q \bullet x = \begin{cases} q \underset{1}{\bullet} x + I_1 & \text{si } f_1 \in q \underset{1}{\bullet} x, \\ q \underset{1}{\bullet} x & \text{sinon,} \end{cases}$

  (les antécédents de la sortie sont branchés sur les entrées),
- $\mathcal{E}ps = \varepsilon$.

c) Appliquer la méthode précédente pour calculer un AFS, puis un AFDC sur $\{a,b\}$, reconnaissant le langage $b^* + (a+b)^* a^* a$.

Exercice 21. AF généralisés (AFG).

Voici une méthode "géométrique", pour construire un ε–AF reconnaissant un langage régulier : elle est assez intuitive mais nécessite la considération d'un type très général d'automates finis, dont nous décrirons seulement les graphes de transition.

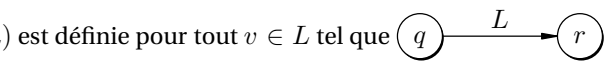
Le graphe de transition d'un AFG $\mathbf{A}$ sur l'alphabet $\mathcal{A}$ est la donnée des éléments suivants :

- Un graphe de transition proprement dit, constitué de nœuds et d'arêtes étiquetés :
  - un nœud (q) pour chaque élément q d'un ensemble fini Q,
  - un ensemble fini d'arêtes $(q) \xrightarrow{L} (r)$ où $q \in Q, r \in Q$ et $L \subseteq \mathcal{A}^*$

    (il est inutile de faire figurer les arêtes vides $(q) \xrightarrow{\emptyset} (r)$),
- un ensemble d'entrées $I \subseteq Q$,
- un ensemble de sorties $F \subseteq Q$.

Le langage reconnu par un tel AFG se définit en adaptant la notion habituelle de dérivation :

* Dans cette construction, tous les ensembles sont traités comme des alphabets.

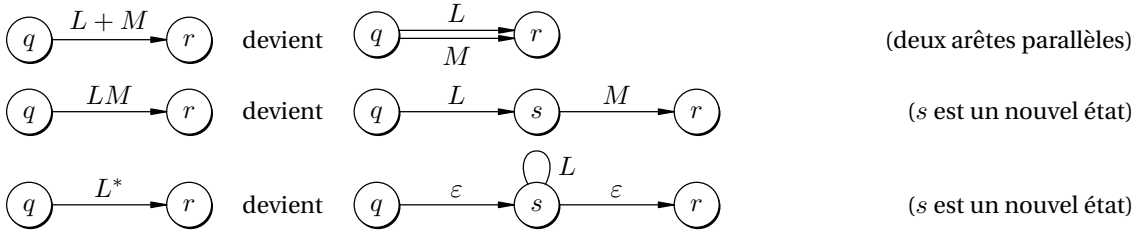
Une transition $(q, vu) \overset{1}{\vdash} (r, u)$ est définie pour tout $v \in L$ tel que  soit une arête du graphe de $\mathbf{A}$. *Une dérivation* est un enchaînement de transitions.

Le langage $\mathcal{L}(\mathbf{A}) \subseteq \mathcal{A}^*$ reconnu par un AFG $\mathbf{A}$ est défini par :

$$u \in \mathcal{L}(\mathbf{A}) \text{ ssi il existe } s \in I \text{ et } r \in F \text{ tels que } (s, u) \overset{*}{\vdash} (r, \varepsilon).$$

Deux AFG sont équivalents s'ils reconnaissent le même langage.

a) Vérifier que l'application de chacune des opérations suivantes (où q et r sont des états qui peuvent être égaux) transforme un AFG en un AFG équivalent.



(Cette dernière transformation ne s'applique utilement que lorsque $L \neq \emptyset$!)

b) En déduire une construction d'un ε–AF reconnaissant un langage régulier défini par une expression régulière.

Exercice 22. Système d'équations associé à un ε–AF. (cf. exercice 1.22)

On associe à tout ε–AF $\mathbf{A} = (Q, \mathcal{A}, \delta, I, F)$ le GL $A : (Q + \varrho)^2 \to \mathcal{P}(\mathcal{A}^*)$ (où $\varrho \notin Q$ est un nouveau symbole) qui est défini par :

$$A(qr) = \{x \in \varepsilon + \mathcal{A} \mid r \in \delta(q, x)\} \text{ pour tout } qr \in Q^2,$$
$$A(q\varrho) = \begin{cases} \varepsilon & \text{pour tout } q \in F \\ \emptyset & \text{pour tout } q \in Q - F, \end{cases}$$
$$A(\varrho q) = \emptyset \text{ pour tout } q \in Q + \varrho.$$

a) Vérifier que le GL ainsi défini est bien celui qui est associé au système d'équations correspondant à l'ε–AF $\mathbf{A}$ (cf. section 5.5 et exercice 1.22).

b) Pour vérifier les affirmations de la section 5.5 au sujet du calcul du langage reconnu par un ε–AF :

1) montrer que quels que soient $q \in Q, r \in Q$ et $u \in \mathcal{A}^*$ on a
$$(q, u) \overset{*}{\vdash} (r, \varepsilon) \text{ ssi } u \in A(Chem(q, r)),$$
2) en déduire que $\mathcal{L}(\mathbf{A}) = A(IQ^*\varrho)$.

Remarque. Après l'exercice 1.22, ceci signifie que le langage engendré par un ε–AF se calcule, comme pour les AFDC, en utilisant la plus petite solution du système qui lui est associé : on doit, bien entendu, tenir compte du fait qu'un ε–AF peut avoir plusieurs entrées.