Chapitre 8

Automates à pile

8.1. Introduction informelle

Nous avons vu que des langages extrêment simples comme le langage $L = \{a^n b^n : n \in \mathbb{N}\}$ ne sont pas reconnus par des automates. On remarque que pour reconnaître un mot de L il suffit d'ajouter une mémoire à un automate qui permettra de compter le nombre de a du début du mot puis de décompter exactement le même nombre de b. Nous devons ajouter des contraintes sur l'utilisation de cette mémoire par un automate, sinon une telle machine ne correspondrait pas à une modélisation finie et permettrait de reconnaître n'importe quel langage. Il suffirait pour cela de mettre lors de la lecture du mot, le mot dans la mémoire et de placer ennsuite l'automate dans un état acceptant si et seulement si le mot appartient au langage donné. La contrainte que l'on va imposer à la mémoire sera d'être sous la forme d'une pile, c'est-à-dire que l'on ne pourra accéder à chaque étape qu'à l'élément se trouvant au-dessus de la pile.

Pour représenter le fonctionnement d'un automate déterministe pour la reconnaissance d'un mot on peut placer le mot sur un *ruban* et utiliser *une tête de lecture* sur le ruban. L'algorithme se déroule alors ainsi :

1. se placer à l'état initial et placer la tête de lecture en position 0 (c'est-à-dire juste à gauche du mot) ;
2. tant que la tête de lecture n'est pas à la fin du mot,
   - déplacer la tête de lecture sur la lettre suivante ;
   - changer d'état en fonction de l'état où il se trouve et de la lettre lue ;
3. si l'automate se trouve dans un état final le mot est accepté ; sinon il est refusé.

Pour représenter le fonctionnement d'un automate à pile déterministe, on adjoint une pile dont le contenu sera un mot sur un second alphabet, le dessus de la pile étant la première lettre de ce mot. L'algorithme de reconnaissance d'un mot u se déroule de la façon suivante :

1. se placer à l'état initial, placer la tête de lecture en position 0 et initialiser la pile (c'est-à-dire mettre un mot fixé dans la pile, en général réduit à une lettre) ;
2. tant que la tête de lecture n'est pas à la fin du mot u et la pile n'est pas vide,
   - déplacer la tête de lecture sur la lettre suivante de u ;
   - changer d'état en fonction de l'état où il se trouve, la lettre lue et la première lettre de la pile ;
   - supprimer la première lettre de la pile (**dépiler**), et la remplacer par un mot de l'alphabet de pile (**empiler**) dépendant de l'état antérieur, de la lettre lue et de la première lettre antérieure de la pile.
3. le mot est accepté si l'automate se trouve dans un état final et la tête de lecture à la fin du mot u ; sinon il est refusé. (Cette dernière étape peut-être remplacer par : le mot est accepté si la pile est vide et la tête de lecture est à la fin du mot u).

8.2. Définitions

8.2.1. Automates à pile. Un *automate à pile* (non déterministe avec transitions instantanées) est la donnée d'un octuplet $\mathcal{A} = (Q, \Sigma, \Pi, S, \delta, \tau, I, F)$ où :
- Q est un ensemble fini non vide d'*états* ;
- Σ est un alphabet fini appelé *alphabet de ruban* ;
- Π est un alphabet fini appelé *alphabet de pile* ;
- S est un élement de Π appelé *pile initiale* ;
- δ est une application de $Q \times \Pi \times \Sigma$ dans $\mathcal{P}_{\mathrm{finies}}(Q \times \Pi^*)$ (le graphe de δ formant l'ensemble des *transitions par lettre*) ;
- τ est une application de $Q \times \Pi$ dans $\mathcal{P}_{\mathrm{finies}}(Q \times \Pi^*)$ (le graphe de τ formant l'ensemble des *transitions instantanées*) ;
- *L'ensemble des états initiaux* I est une partie de Q ;
- *L'ensemble des états finaux* F est une partie de Q.

Ci-dessus $\mathcal{P}_{\mathrm{finies}}(Q \times \Pi^*)$ désigne l'ensemble des parties finies de $Q \times \Pi^*$.

Pour les applications pratiques, en particulier pour l'analyse syntaxique de langages particuliers, les automates à pile déterministes sont plus intéressants. Mais contrairement à la classe des langages rationnels, nous verrons plus tard qu'un langage algébrique (langage reconnu par un automate à pile) n'est pas nécessairement reconnu par un automate à pile déterministe.

8.2.2. Automates à pile déterministes. Un *automate à pile déterministe* est un automate à pile qui n'a qu'un seul état initial et tel que
- pour tout $q \in Q$, tout $x \in \Pi$ et tout $a \in \Sigma$, $\delta(q, x, a)$ contient au plus un élément ;
- pour tout $q \in Q$ et tout $x \in \Pi$, $\tau(q, x)$ contient au plus un élément et dans le cas où $\tau(q, x)$ est non vide alors pour tout $a \in \Sigma$, $\delta(q, x, a)$ est vide.

8.3. Langages reconnus

Soient a une lettre de Σ, p et q deux états et u, v deux mots de pile. Si on a $u = xu'$ et $v = v_0u'$ tels que $(q, v_0) \in \delta(p, x, a)$, cela signifie qu'il y a une transition de (p, u) vers (q, v) étiquetée par a, que nous noterons $(p, u) \xrightarrow{a} (q, v)$. Si on a $(q, v_0) \in \tau(p, x)$, cela signifie qu'il y a une transition instantanée (p, u) vers (q, v) que nous noterons $(p, u) \xrightarrow{\varepsilon} (q, v)$.

Pour un mot $w = a_1...a_n$, nous dirons qu'il y a un chemin de (p, u) vers (q, v) étiqueté par w, noté $(p, u) \xrightarrow{w} (q, v)$, s'il existe une suite de transitions adéquates étiquetées par les a_i et de transitions instantanées (voir définition analogue en 2.4.3).

La description précédente du fonctionnement d'un automate à pile suggère deux définitions possibles pour l'acceptation d'un mot :

1. Un mot w est *accepté par état final* s'il existe un chemin $(p, S) \xrightarrow{w} (q, v)$ tel que p est initial et q final.
2. Un mot w est *accepté par pile vide* s'il existe un chemin $(p, S) \xrightarrow{w} (q, v)$ tel que p est initial et v est vide (c'est-à-dire $v = \varepsilon$).

On appelle *langage reconnu par état final*, l'ensemble des mots acceptés par état final, et *langage reconnu par pile vide*, l'ensemble des mots acceptés par pile vide. Pour un automate donné, il n'y a aucune raison que les deux langages reconnus soient les mêmes. Par contre, on va vérifier plus loin qu'un langage est reconnaissable par état final si et seulement il est reconnaissable par pile vide. On peut remarquer qu'un langage reconnu par un automate à pile sans transitions instantanées (que ce soit par état final ou par pile vide) est décidable : pour tout mot il n'y a qu'un nombre borné de chemins possibles (en fonction de la longueur du mot), donc il suffit d'écrire un algorithme d'essai et erreurs. Il est moins évident de vérifier qu'un langage reconnu par un automate à pile contenant des transitions instantanées est aussi décidable (on peut avoir des chemins infinis de transitions instantanées). Nous verrons dans le chapitre 10 que les langages reconnaissables par automates à pile sont exactement les langages algébriques, c'est-à-dire ceux engendrés par des grammaires algébriques (ou hors-contextes). Ceci nous permettra de vérifier que ces langages sont tous décidables. Mais contrairement aux langages rationnels, nous verrons alors qu'il n'est pas toujours possible de "déterminiser" un automate à pile : il existe des langages algébriques qui ne sont reconnaissables par aucun automate à pile déterministe.

8.4. Exemples

Pour représenter un automate à pile, on peut simplement donner sa table de transition de la façon suivante :

	a	b
$\rightarrow 0, S$	$1, S$	
$1, S$	$1, TS$	$0, \varepsilon$
$1, T$	$1, TT$	$2, \varepsilon$
$2, S$		$0, \varepsilon$
$2, T$		$2, \varepsilon$

FIG. 8.1 – Automate reconnaissant $\{a^n b^n : n \in \mathbb{N}\}$ par état final

On peut représenter ce même automate par un graphe où les flèches représentent les transitions : au début de la flèche on met l'étiquette correspondant au couple formé de la lettre de Σ et de la première lettre de la pile ; on milieu de la flèche le mot de Π qui remplace la première lettre antérieure :

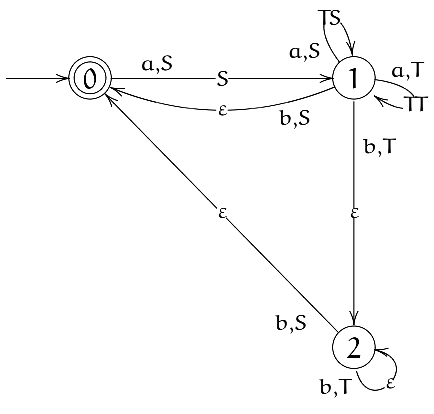

FIG. 8.2 – Graphe de l'automate précédent

Notons que cette automate est déterministe. Il reconnait le langage $\{a^n b^n : n \in \mathbb{N}\}$ par état final et le langage $\{a^n b^n : n \in \mathbb{N}^*\}$ par pile vide.

	a	b	ε
$\rightarrow 0, S$	$0, SS$		$1, \varepsilon$
$1, S$		$1, \varepsilon$	

FIG. 8.3 – Automate reconnaissant $\{a^n b^n : n \in \mathbb{N}\}$ par pile vide

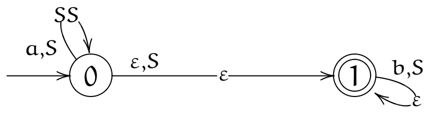

FIG. 8.4 – Graphe de l'automate ci-dessus

Notons qu'ici le langage reconnu par état final est $\{a^p b^q : q \leq p\}$.

8.5. EQUIVALENCE DES LANGAGES RECONNUS PAR PILE VIDE ET DES LANGAGES RECONNUS PAR ÉTAT FINAL

8.5.1 Proposition. *Si* L *est le langage reconnu par un automate à pile par état final alors il existe un automate à pile reconnaissant* L *par pile vide.*

Démonstration. Soit $\mathcal{A} = (Q, \Sigma, \Pi, S, \delta, \tau, I, F)$ un automate à pile. On va simuler l'automate $\mathcal{A}$ par un automate $\mathcal{B}$ qui pourra dépiler la pile quand $\mathcal{A}$ se trouve dans un état final. Pour cela nous allons ajouter un nouvel état q_ε tel que pour tout état final q de $\mathcal{A}$ on ait dans le nouvel automate les transitions représentées par :

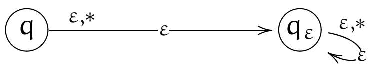

FIG. 8.5 – Transitions d'un ancien état final q vers q_ε

Par contre le nouvel automate ne doit pas a priori reconnaître un mot qui conduit l'automate $\mathcal{A}$ à une pile vide dans un état final sans être à la fin de la lecture de ce mot. Pour celà on va utiliser une nouvelle lettre S' représentant la pile initiale et un nouvel état q_0 qui sera le nouvel état inial. Dans le nouvel automate $\mathcal{B}$, on ajoute pour chaque ancien état initila q la transition représentée par :

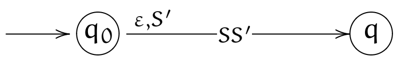

FIG. 8.6 – Transition de q_0 vers un ancien état initial.

Formellement cela donne l'automate $\mathcal{B} = (Q \cup \{q_0, q_\varepsilon\}, \Sigma, \Pi \cup \{S'\}, S', \delta, \tau', \{q_0\}, \emptyset)$ tel que :

1. $\tau'(q_0, S') = \{(q, SS') : q \in I\}$;
2. Pour tout $q \in F$ et tout $x \in \Pi$, $\tau'(q, x) = \tau(q, x) \cup \{(q_\varepsilon, \varepsilon)\}$;
3. Pour tout $q \in F$, $\tau'(q, S') = \{(q_\varepsilon, \varepsilon)\}$;

4. Pour tout $q \in Q \setminus F$ et tout $x \in \Pi$, $\tau'(q, x) = \tau(q, x)$;
5. Pour tout $x \in \Pi \cup \{S'\}$, $\tau'(q_\varepsilon, x) = \{(q_\varepsilon, \varepsilon)\}$;
6. Dans tous les autres cas, τ' a pour valeur $\emptyset$.

Alors il est facile de vérifier par récurrence que pour tous $p, q \in Q$, tous $u, v \in \Pi^*$ et tout mot $w \in \Sigma^*$, on a

$$(p, u) \xrightarrow[\mathcal{A}]{w} (q, v) \text{ ssi } (p, u) \xrightarrow[\mathcal{B}]{w} (q, v) \text{ ssi } (p, uS') \xrightarrow[\mathcal{B}]{w} (q, vS') .$$

Il suit qu'un mot w est accepté par $\mathcal{A}$ par état final si et seulement s'il existe un état final q de $\mathcal{A}$ et $v \in \Pi^*$ tel que $(q_0, S') \xrightarrow[\mathcal{B}]{w} (q, vS')$ si et seulement si $(q_0, S') \xrightarrow[\mathcal{B}]{w} (q_\varepsilon, \varepsilon)$ si et seulement si w est accepté par $\mathcal{B}$ par pile vide. □

8.5.2 Proposition. *Si* L *est le langage reconnu par un automate à pile par pile vide alors il existe un automate à pile reconnaissant* L *par état final.*

Démonstration. Soit L un lanagage reconnu par pile vide par un automate $\mathcal{A} = (Q, \Sigma, \Pi, S, \delta, \tau, I, F)$. Il s'agit alors de simuler $\mathcal{A}$ par un automate qui envoie par une transition instantanée dans un nouvel état final dès que $\mathcal{A}$ se trouve dans une configuration où la pile est vide. Pour celà on définit un automate $\mathcal{B}$ à partir de l'automate $\mathcal{A}$ où l'on ajoute deux états, l'état q_0 qui est l'unique état initial de $\mathcal{B}$, et l'état q_ε qui est l'unique état final. On considère une nouvelle lettre S' qui représente la pile initiale de $\mathcal{B}$ et on ajoute les transitions instantanées suivantes :

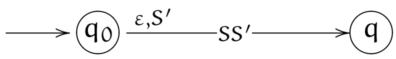

FIG. 8.7 – Nouvelle transition de q_0 vers un ancien état initial $q \in I$.

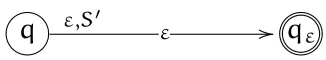

FIG. 8.8 – Nouvelle transition d'un ancien état $q \in Q$ vers q_ε

Il alors facile de vérifer que L est reconnu par $\mathcal{B}$ par état final. □

8.6. AUTOMATES À PILE DÉTERMINISTES

Revenons maintenant aux automates à pile déterministes. Remarquons que la définition d'un automate à pile déterministe n'interdit pas qu'à partir d'une certaine configuration, l'automate fasse un chemin infini de transitions instantanées. Un tel automate pourrait donc boucler à l'infini (problème de l'arrêt).

8.6.1. Configurations. A partir de maintenant on appelle *configuration* d'une machine à un instant donné, la donnée de l'état complet où elle se trouve quand elle est en fonctionnement. Par exemple pour un automate à pile, il s'agit de la donnée de son état, du mot qu'il y a à cet instant sur son ruban à partir de la tête de lecture, et du mot de pile. Un configuration d'un automate à pile $\mathcal{A} = (Q, \Sigma, \Pi, S, \delta, \tau, I, F)$ sera donc représentée par un triplet (q, w, u) où $q \in Q$, $w \in \Sigma^*$ et $u \in \Pi^*$.

Nous utiliserons les notations $\longmapsto$ (respectivement $\overset{n}{\longmapsto}$ et $\overset{*}{\longmapsto}$) pour le passage d'une configuration à une autre en une étape (respectivement n-étapes et en un nombre non précisé d'étapes).

8.6.2 Définition. On dira qu'une configuration (q, w, u) d'un automate à pile déterministe *boucle* s'il existe une suite infinie de configuration (p_i, w, u_i) telle que $p_0 = q$, $u_0 = u$, $|u_i| \geq |u|$ pour tout i et

$$(q, w, u) \longmapsto (p_1, w, u_1) \longmapsto \cdots \longmapsto (p_i, w, u_i) \longmapsto (p_{i+1}, w, u_{i+1}) \longmapsto \cdots$$

8.6.3. Remarque. Notons que si un automate boucle à l'infini à partir d'une configuration (q, w, u) alors en considérant un instant où la pile est de taille minimale dans cette boucle, on en déduit qu'il existe une configuration (q', w, u') qui boucle dans le sens ci-dessus. De plus comme u' est de taille minimale, en considérant la première lettre x de u', il suit que la configuration (q', ε, x) boucle également.

On dira qu'un automate à pile déterministe *est sans boucles* s'il n'a aucune configuration qui boucle.

8.6.4. Algorithme de détection des boucles.

Fixons un automate à pile déterministe $\mathcal{A} = (Q, \Sigma, \Pi, S, \delta, \tau, \{q_0\}, F)$. Alors il existe un entier m tel que pour tout $q \in Q$ et $x \in \Pi$, la configuration (q, ε, x) boucle si et seulement si $(q, \varepsilon, x) \overset{m}{\longmapsto} (p, \varepsilon, u)$ (c'est-à-dire s'il y a un chemin de transitions instantanées de longueur m à partir de (q, x).

Démonstration. Soient $n_1 = |Q|$, le nombre d'états, $n_2 = |\Pi|$, le cardinal de l'alphabet de pile, et l le maximum des longueurs des mots que l'on empilent.

Montrons tout d'abord que s'il existe $r \in Q$ et $v \in \Pi^*$ tels que $|v| > n_1 n_2 l$ et $(q, \varepsilon, x) \overset{*}{\longmapsto} (r, \varepsilon, v)$ alors la configuration (q, ε, x) boucle. Pour chaque $j \in \{1, 2, ..., n_1 n_2 l + 1\}$, soit C_j la dernière configuration entre (q, ε, x) et (r, ε, v) où la pile est de longueur au plus égale à j. Comme $|v| > n_1 n_2 l$, il y au moins $n_1 n_2 + 1$ configurations distinctes parmi les C_j. Comme $|Q \times \Pi| = n_1 n_2$, il existe deux configurations distinctes qui sont dans le même état et tels que le dessus la pile est le même. On en déduit qu'il existe un état $q' \in Q$, une lettre $y \in \Pi$ et deux mots $u_1, u_2 \in \Pi^*$, u_2 étant non vide tels que

$$(q, \varepsilon, u) \overset{*}{\longmapsto} (q', \varepsilon, y u_1) \overset{k}{\longmapsto} (q', \varepsilon, y u_2 u_1) \overset{*}{\longmapsto} (r, \varepsilon, v) \ .$$

De plus, par définition des configurations C_j, on a $(q', \varepsilon, y) \overset{k}{\longmapsto} (q', \varepsilon, y u_2)$ et donc pour tout i, $(q', \varepsilon, y) \overset{ik}{\longmapsto} (q', \varepsilon, y u_2^i)$. Il suit que (q, ε, x) boucle.

Soit maintenant m le produit de n_1 avec le nombre de mots de longueurs inférieures ou égales à $n_1 n_2 l$ sur l'alphabet Π. Supposons que $(q, \varepsilon, x) \overset{m}{\longmapsto} (p, \varepsilon, u)$. Alors ou bien il existe $r \in Q$ et $v \in \Pi^*$ tels que $|v| > n_1 n_2 l$ et $(q, \varepsilon, x) \overset{*}{\longmapsto} (r, \varepsilon, v)$, et alors la configuration (q, ε, x) boucle. Ou bien aucune configuration intermédiaire entre (q, ε, x) et (p, ε, u) ne vérifie cette propriété. Mais par le choix de m, il suit que deux configurations intermédiaires sont identiques. On a donc également dans ce cas une boucle.

□

On remarque dans la preuve ci-dessus que si une configuration boucle alors l'ensemble des états de l'automate parcouru par cette boucle est égal à l'ensemble des états parcouru en m-étapes (pour le m qui a été choisi). Ceci vient du fait que la boucle se produit au plus tard après m étapes. Il suit que l'on peut compléter l'algorithme de détection des boucles afin de séparer celles qui passent par un état final de celles qui ne passent jamais par un état final.

8.6.5. Automates à pile déterministes continus. On dit qu'un automate à pile déterministe $\mathcal{A} = (Q, \Sigma, \Pi, S, \delta, \tau, \{q_0\}, F)$ est continu si pour tout mot $w \in \Sigma^*$, il existe $p \in Q$ et $u \in \Pi^*$ tel que $(q_0, w, S) \overset{*}{\longmapsto} (p, \varepsilon, u)$, c'est-à-dire si $\mathcal{A}$ lit complètement les mots.

8.6.6 Proposition. *Si* L *est un langage reconnu par état final par un automate à pile déterministe alors* L *est reconnu par état final par un automate à pile déterministe continu et sans boucles.*

Démonstration. Tout d'abord il est facile de modifier l'automate afin que lors de la lecture d'un mot, il y ait toujours une étape suivante du moment que la lecture n'est pas terminée (c'est-à-dire tel qu'il n'y ait pas d'arrêt pour cause de pile vide ou d'absence de transitions). Pour cela on ajoute un fond de pile et si nécessaire d'éventuelles transitions vers un état rebut. Ensuite, afin d'éliminer les boucles, on ajoute deux états $\{p_1, p_2\}$, le premier final, l'autre non. Alors pour tout état q et tout lettre $x \in \Pi$ tels que la configuration (q, ε, x) boucle on modifie les transitions de la façon suivante :

– si la boucle passe par un état final, on met une transition instantanée de (q, x) vers (p_1, x),

$$(q, x) \xrightarrow{\varepsilon} (p_1, x) ;$$

– si la boucle ne passe pas par un état final, on met une transition instantanée de (q, x) vers (p_2, x),

$$(q, x) \xrightarrow{\varepsilon} (p_2, x) .$$

Pour toute lettre $a \in \Sigma$, on complète par les transitions :

$$(p_1, x) \xrightarrow{a} (p_2, x) \text{ et } (p_2, x) \xrightarrow{a} (p_2, x) .$$

On vérifie facilement que l'automate obtenu reconnaît le même langage et est continu sans boucle. □

8.6.7 Corollaire. *Si un langage est reconnu par état final par un automate à pile déterministe alors son complémentaire également.*

Démonstration. Considérons pour cela un automate $\mathcal{A}$ continu et sans boucles. Pour reconnaître le complémentaire de $L(\mathcal{A})$, il faut alors simuler $\mathcal{A}$ en retenant en mémoire si $\mathcal{A}$ se trouve à l'état q après un chemin de transitions instantanées passant par un état final ou non. Enfin une fois que l'on est à la fin de la lecture du mot et à la fin du chemin éventuel de transitions instantanées, il faut se placer par une nouvelle transition instantanée dans un état final si $\mathcal{A}$ n'est pas passé par un état final depuis la lecture de la dernière lettre. Pour construire un tel automate, l'on peut tripler chaque état. Les détails sont laissés au lecteur. □

EXERCICES

8.1 Exercice. On considère L l'ensemble des palindromes sur l'alphabet $\{a, b\}$. (Un palindrome est un mot qui se lit de la même facçon de gauche à droite ou de droite à gauche. Le mot $abbba$ est un palindrome).

1. Montrer que L n'est pas rationnel.
2. Montrer que L est reconnaissable par un automate à pile.

8.2 Exercice. On definit le langage L des parenthèses bien formées sur l'alphabet $\{(,)\}$ comme étant l'ensemble des mots u tel que :

- Le nombre $|u|_($ d'occurrences de (dans u est égal au nombre $|u|_)$ d'occurrences de) dans u.
- Pour tout préfixe v de u, $|v|_(\geq |v|_)$.

1. Montrer que L n'est pas rationnel.
2. Montrer que L est reconnaissable par un automate à pile.

8.3 Exercice.

1. Montrer qu'un langage reconnu par un automate à pile déterministe par pile vide est reconnaissable par un automate à pile déterministe par état final. Vérifier que la réciproque est fausse.
2. Montrer qu'un langage L reconnu par un automate à pile déterministe par pile vide est sans préfixe : pour tout mot u de L, il n'existe par de préfixe de u dans L (c'est-à-dire de mots $v \in L$ tels que $u = vw$ avec w non vide).
3. Montrer qu'un langage sans préfixe reconnu par un automate à pile déterministe par état final est reconnaissable par un automate à pile déterministe.