Notes de cours Informatique théorique - II
(version 2.5.2, 21.2.2019)

S.V.P. ME SIGNALER TOUTE FAUTE DE FRAPPE, GRAMMAIRE, ORTHOGRAPHE OU LOGIQUE; LES FAUTES DE LOGIQUE VOUS APPORTENT DES POINTS SUPPLÉMENTAIRES POUR LA NOTE FINALE. TOUTE SUGGESTION D'UNE MEILLEURE TRADUCTION D'UN TERME ANGLAIS SERA ÉGALEMENT BIENVENUE.

Définition 1 *Une* grammaire hors-contexte *est un quadruplet* $G = (V, \Sigma, R, S)$ *où* V *est un ensemble fini de* variables, Σ *est un alphabet,* $V \cap \Sigma = \emptyset$, $S \in V$ *est un* symbole de départ *et* $R \subset \{A \longrightarrow \alpha : A \in V, \alpha \in (V \cup \Sigma)^*\}$ *est un ensemble fini de* règles *(aussi appellées* productions*).*

On dit *hors-contexte* parce que la règle $A \longrightarrow \alpha$ permet de remplacer A par α n'importe quand, sans un contexte particulier (contrairement, par exemple, aux règles telles que $aaAbb \longrightarrow aa\alpha bb$ qui ne permettent le remplacement de A par α que s'il est précédé de aa et suivi de bb).

Une grammaire G *génère* un langage $L(G)$: à partir du symbole de départ, on applique des règles et si on aboutit à un mot qui ne contient que des symboles de Σ, on arrête et on a un mot de $L(G)$. Plus précisement, soit $G = (V, \Sigma, R, S)$ une grammaire hors-contexte. Pour $u, v \in (V \cup \Sigma)^*$ on définit u *donne* v *en* k *étapes* (écrit $u \overset{k}{\Longrightarrow} v$) récursivement :

1. $u \overset{0}{\Longrightarrow} u$
2. $u \overset{k}{\Longrightarrow} v$ s'il existe $u_1, u_2, w \in (V \cup \Sigma)^*$, $A \in V$ tels que $u \overset{k-1}{\Longrightarrow} u_1 A u_2$, $A \longrightarrow w \in R$ et $v = u_1 w u_2$.

On dira que u *donne* v, $u \overset{*}{\Longrightarrow} v$, s'il existe un $k \in N$ tel que $u \overset{k}{\Longrightarrow} v$. Noter les cas particuliers $A \overset{*}{\Longrightarrow} w$ et $S \overset{*}{\Longrightarrow} w$. Avec ceci, on définit $L(G) = \{w \in \Sigma^* : S \overset{*}{\Longrightarrow} w\}$. Un langage généré par une grammaire hors-contexte est, lui aussi, *hors-contexte*.

Cette définition nous permet de prouver que le langage généré par une grammaire G est ce que l'on prétend, ou ce que l'on veut, qu'il soit.

Exemple 1 *Soit* $G = (V = \{S\}, \Sigma = \{a, b\}, R, S)$ *une grammaire hors-contexte qui génère les palindromes sur* $\{a, b\}$*, c'est-à-dire, qui génère* $L = \{w \in \{a, b\}^* : w = w^R\}$*. Il suffit de prendre* $R = \{S \longrightarrow aSa \mid bSb \mid a \mid b \mid \epsilon\}$*. Prouvons que* $L(G) = L$*. On doit prouver que* $L \subseteq L(G)$ *et* $L(G) \subseteq L$*. Les deux preuves sont par récurrence.*
Soit $w \in L$*.*
Si $|w| \leq 0$*,* $w = \epsilon$ *et* $S \overset{*}{\Longrightarrow} \epsilon$ *car* $S \longrightarrow \epsilon \in R$*. Donc* $w \in L(G)$*.*
Si $|w| = 1$*,* $w = a$ *ou* $w = b$*. Puisque* $S \longrightarrow a$ *et* $S \longrightarrow b$ *sont des règles de* G*,* $S \overset{*}{\Longrightarrow} w$ *dans les deux cas et* $w \in L(G)$*.*
Si $|w| > 1$*, alors il existe* $u \in L$ *et* $x \in \{a, b\}$ *tel que* $w = xux$*. Evidemment,* $0 \leq |u| = |w| - 2$*.*

Par l'hypothèse de récurrence, $S \stackrel{*}{\Longrightarrow} u$. Mais alors $S \stackrel{1}{\Longrightarrow} xSx \stackrel{*}{\Longrightarrow} xux$, comme on le veut, car $S \longrightarrow xSx$ est une règle de G que x soit a ou b.

Soit maintenant $w \in L(G)$. On veut montrer que $w \in L$. Puisqu'on doit se servir des règles pour le prouver, il faut faire un peu plus pour que la récurrence marche. Prouvons que si $S \stackrel{k}{\Longrightarrow} u$, $u \in (V \cup \Sigma)^*$, alors il existe $v \in \Sigma^*$ et $X \in \{S, a, b, \epsilon\}$ tels que $u = vXv^R$. Ceci prouvera que si $u \in L(G)$, alors $u = vv^R$, ou $u = vbv^R$, ou $u = vav^R$, i.e., $u \in L$.

Si $k = 0$, $S \stackrel{0}{\Longrightarrow} S$ et on peut prendre $v = \epsilon$.

Si $k = 1$, $S \stackrel{1}{\Longrightarrow} \{aSa \mid bSb \mid a \mid b \mid \epsilon\}$ (abus de notation clair). Dans les deux premiers cas, v est a ou b, respectivement, et $X = S$. Dans les trois derniers cas, X est a, b, ϵ, respectivenent, et $v = \epsilon$.

Si $k > 1$, on doit avoir $S \stackrel{1}{\Longrightarrow} xSx \stackrel{k-1}{\Longrightarrow} xux$, $x \in \Sigma$, sinon $k \leq 1$. Mais alors on a, par l'hypothèse de récurrence, que $S \stackrel{k-1}{\Longrightarrow} u$ et $u = vXv^R$. Ceci donne que $w = xvXv^Rx$ et w est de la forme recherchée.

Exercice 1 *Trouvez une grammaire hors-contexte et prouvez un résultat analogue pour un alphabet Σ arbitraire.*

Exercice 2 *Soit Σ un aphabet. Donner une définitions précise d'un palindrome $w \in \Sigma^*$ plus précise que "$w = w^R$".*

Pour les exercices ci-haut, une bonne approche est de chercher une définition récursive: définir des palindromes de longueur $0, 1$ et ensuite les plus longs Faites-le pas en faisant la récurrence sur la longueur mais plutôt sur la forme (inspirez-vous de la définition d'une expression régulière). Avec une telle définition, la preuve que $L(G) = L$ est bien plus facile.

Exercice 3 *Soit $\Sigma = \{(,)\}$ l'alphabet de deux symboles, parenthèse gauche et parenthèse droite. Soit $L = \{w \in \Sigma^* :$ le parenthésage est correct$\}$.*

1. *Qu'est-ce que cela veut dire,* le parenthésage est correct*? Donnez deux définitions* précises *différentes, l'une récursive et l'autre pas, et prouvez qu'elles sont équivalentes.*
2. *Est-ce que L est hors-contexte (voir plus bas)? Prouvez votre réponse (c'est pour ceci que vous avez besoin des réponses à la première question).*
3. *Combien y-a-t-il de "bons parenthésages" de n paires de parenthèses?*

L'exemple 1 se sert implicitement du lemme assez évident, mais que l'on prouve quand même, suivant.

Lemme 1 *Soit G une grammaire hors-contexte, Pour tout $\alpha \in (V \cup \Sigma)^*$, si $w = u\alpha v$, $u, v \in (V \cup \Sigma)^*$ et si $\alpha \stackrel{*}{\Longrightarrow} \beta$, $\beta \in (V \cup \Sigma)^*$, alors $w \stackrel{*}{\Longrightarrow} u\beta v$.*

Démonstration. On fait une récurrence sur la longueur k de la dérivation. Si $\alpha \stackrel{0}{\Longrightarrow} \beta$, $\beta = \alpha$ et $u\alpha v \stackrel{0}{\Longrightarrow} u\alpha v$. Si $\alpha \stackrel{k}{\Longrightarrow} \beta$, $k > 0$, il existe $A \in V$, une règle $A \longrightarrow \gamma$, et $u_1, u_2 \in (V \cup \Sigma)^*$ tels que $\alpha \stackrel{k-1}{\Longrightarrow} u_1 A u_2 \stackrel{1}{\Longrightarrow} u_1\gamma u_2$, par définition, avec $\beta = u_1\gamma u_2$. Alors $w \stackrel{k-1}{\Longrightarrow} uu_1 A u_2 v$, par l'hypothèse de récurrence, et donc $w \stackrel{k}{\Longrightarrow} uu_1\gamma u_2 v$ en applicant la règle $A \longrightarrow \gamma$ et $uu_1\gamma u_2 v = u\beta v$. □

Rappellons quelques idées sur les arborescences. On suppose que tout le monde sait ce que c'est qu'un graphe $G(V, E)$ connexe et ce que c'est qu'un cycle. On se rapelle qu'un arbre est un graphe connexe sans cycle. Dans un arbre, les sommets de degré au plus 1 (il y en a toujours au moins 2 dans un arbre avec au moins 2 sommets) s'appellent *feuilles*, les sommets de degré supérieur à 1 s'appellent *sommets internes*.

Définition 2 *Une* arborescence *est un arbre avec un sommet* racine *distingué.*

De manière équivalente on définit un arbre comme un graphe dans lequel entre chaque paire de sommets il existe exactement un chemin (preuve de l'équivalence?) Il s'en suit que dans une arborescence, chaque chemin de la racine vers une feuille peut être orienté de la racine vers la feuille et on obtient ainsi un arbre orienté.

Le *niveau* dans une arborescence est l'ensemble de sommets qui sont à la même distance de la racine. A distance i, on a le niveau $L_i = \{v \in V : d(r, v) = i\}$ ($d(r, v)$ est la distance de r à v, i.e., la longueur – le nombre d'ârretes – de l'unique chemin de r vers v).
On peut également définir les niveaux récursivement. Soit $T = (V, A)$ une arborescence avec la racine r. Le niveau L_i (le i-ème niveau) est définit par :

1. $L_0 = \{r\}$
2. $L_{i+1} = \{v \in V \setminus (\cup_{j=0}^{i} L_j) : \text{il y a un } u \in L_i,\ uv \in E\}$, c'est-à-dire, le niveau L_{i+1} contient tous les voisins qui ne sont pas déjà dans un des L_j, pour $j < i$, des sommets dans L_i.

Tout sommet appartient à un seul niveau que l'on appelle le *niveau de u*. On appelle *enfant de u* tout sommet adjacent à u au niveau supérieur à celui de u, i.e., les enfants de $u \in L_i$ forment l'ensemble $\{v \in L_{i+1} : uv \in E\}$. Si $u \in L_i$, on appelle *parent de u* l'unique sommet v au niveau L_{i-1} adjacent à u et on le note $p(u)$.

Une arborescence est *ordonnée* si les enfants de chaque sommet sont ordonnés par un ordre total $\leq$ (pensez *numérotés*). Ceci permet de définir un ordre total sur tous les sommets de chaque niveau et donc sur tous les sommets de T : pour $u \neq v$, on met $u \preceq v$ si et seulement si soit $p(u) \preceq p(v)$, soit $p(u) = p(v)$ et $u \leq v$. Bien sur, pour que $\preceq$ soit un ordre linéaire, il faut aussi $u \preceq u$ pour tout u. Si T a n sommets, on obtient donc un ordre total $v_1 \preceq v_2 \preceq \ldots \preceq v_n$ des sommets de l'arborescence. On appelle cet ordre *l'ordre lexicographique*.

On peut voir cet ordre d'une autre façon. Puisque dans une arborescence pour tout sommet v il y a exactement un chemin de la racine r vers v, on peut représenter v par

ce chemin (il y a une bijection entre ces chemins et les sommets). Si on note $a_1 \dots a_k$ le chemin unique de la racine $r = a_1$ vers $a_k = u$ et, mutatis mutandis (*en changeant ce qu'il faut changer*), $b_1 \dots b_\ell$, $b_1 = r$, $b_\ell = v$, on a que $u \preceq v$ si et seulement si $a_k \preceq b_\ell$ si $a_1 \dots a_k \preceq b_1 \dots b_l$, ce qui arrive si soit $k < l$, soit $a_i \leq b_i$ avec i le minimum tel que $a_i \neq b_i$.

On dessine une telle arborescence dans l'ordre: d'abord la racine, puis ses enfants dans l'ordre (on les dessine de gauche à droite), puis les enfants du premier enfant de la racine, les enfants du deuxième enfant de la racine, etc. - de façon à ce qu'une fouille en largeur à partir de la racine rencontre les enfants de chaque sommet dans l'ordre.

Si chaque sommet interne de l'arborescence T a exactement b enfants et si T a $k+1$ niveaux (numérotés $0, 1, \dots, k$), alors l'arborescence a exactement b^k feuilles. Si chaque sommet interne de T a *au plus* b enfants, l'arborescence a *au plus* b^k feuilles (ceci est facile à prouver par récurrence sur k). Il en découle qu'une arborescence dans laquelle chaque sommet interne a au plus b enfants et qui a au moins $b^k + 1$ feuilles doit avoir au moins $k + 2$ niveaux. Notons que si une arborescence a ℓ niveau, elle doit contenir un chemin de *longueur* $\ell - 1$ de la racine vers une feuille; on appelle cette longueur la *profondeur* ou encore la *hauteur* de l'arborescence (attention, le nombre de niveaux n'est pas la hauteur – profonduer - se l'arbre car on commence à numéroter à 0).

Un *arbre de dérivation* (ADD), aussi appelé *arbre d'analyse* (AA) associé à une grammaire $G = (V, \Sigma, R, S)$ est une arborescence T ordonnée dont les sommets sont étiquetés par $e : V(T) \longrightarrow (V \cup \Sigma \cup \{\varepsilon\})$, définie récursivement :

1. Un sommet seul étiqueté par une variable est un ADD;
2. Si T_0 est un ADD, F une feuille de T_0 étiquetée par une variable B et $B \longrightarrow X_1 X_2 \dots X_k$ une règle de G, l'arborescence T ordonnée obtenue en ajoutant k nouvelles feuilles $F_1, \dots, F_k$, dans l'ordre naturel ($F_i \leq F_j$ si $i \leq j$), avec F_i étiquetées par X_i, comme enfants de F, est également un ADD.

On note un tel ADD $T_G(A)$ si la racine est étiquetée par la variable A et la plupart du temps on oublie l'indice G qui est clair du contexte. Puisque l'ADD est une arborescence ordonnée, on peut "lire" les étiquettes des feuilles en pré-ordre. Plus précisément, si $F_1, \dots, F_k$ sont les feuilles, en pré-ordre, d'un ADD, le mot $e(F_1)e(F_2)\dots e(F_k)$ est noté par $w(T)$ et on l'appelera le *mot de* T. Pour définir préordre sur les feuilles, on regarde les chemins $ru_1a_2 \dots u_k$ et $rv_1 \dots v_\ell$, $u_k = u$, $v_\ell = v$ de la racine r vers deux sommets u et v de l'arborescence et on met $\text{pre}(u, v) = \min\{i : a_i \neq b_i\}$. Avec ceci on peut définir $u \sqsubseteq v$ if $u_{\text{pre}(u,v)} \preceq v_{\text{pre}(u,v}$. L'ordre $\sqsubseteq$ est le préordre et s'applique, bien sûr, aux feuilles (visuellement, si on dessine l'arborescence pour que l'ordre soit de gauche à droite, les feuilles en pré-ordre vont alors de gauche à droite).

Une meilleure façon de définir un arbre d'analyse $T_G(A)$ et, en même temps, son mot, noté $w(T_G(A))$ est la suivante (en omettant l'indice G car il s'agit toujours de la même grammaire).

On définit un *arbre de dérivation* (ADD), aussi appelé *arbre d'analyse* (AA) $T_G(A)$, associé à une grammaire $G = (V, \Sigma, R, S)$ et une variable $A \in V$, ainsi que son mot $w(T(A))$ (on oublie l'indice G quand il est clair du contexte) récursivement. On définit $T(A)$ comme une arborescence T ordonnée dont les sommets sont étiquetés par $e : V(T) \longrightarrow (V \cup \Sigma \cup \{\varepsilon\})$ et son mot $w(T(A))$ comme la suite des étiquettes de ses feuilles.

1. Un sommet seul étiqueté par A est un ADD $T_0(A)$ et son mot est A, $w(T_0(A)) = A$;
2. Soit $T_i(A)$ un ADD avec les feuilles $L_1, \ldots, L_n$, avec $e(L_k) = Y_k$, $k = 1, \ldots, n$, $Y_j = B \in V$. Soit $w(T_i(A)) = Y_1 Y_2 \ldots Y_{j-1} B Y_{j+1} \ldots Y_n$, $Y_k \in V \cup \Sigma \cup \{\varepsilon\}$ pour $k = 1, \ldots, j-1, j+1 \ldots n$. Soit $B \longrightarrow X_1 X_2 \ldots X_\ell$ une règle de G. L'arborescence ordonnée obtenue en ajoutant k nouvelles feuilles $F_1, \ldots, F_\ell$, dans l'ordre naturel ($F_i \leq F_j$ si $i \leq j$), avec F_i étiquetées par X_i, comme enfants de L_j, est un ADD $T_{i+1}(A)$ et son mot est $w(T_{i+1}(A)) = Y_1 Y_2 \ldots Y_{j-1} X_1, \ldots, X_\ell Y_{j+1} \ldots Y_n$.

Il est facile de voir (mais demande une preuve si on veut être rigoureux, voir Hopcroft, Motwani, Ullman, *Introduction to automata theory, languages, and computation*, pp.181 – 191), que dans une grammaire G, $A \stackrel{*}{\Longrightarrow} w$ si et seulement si il existe un ADD T dont la racine est étiquetée par A tel que $w(T) = w$ (ici $w \in (V \cup \Sigma)^*$).

Si G est une grammaire, $A \in V$ et $T = T_G(A)$ un ADD, on note $T[X, B]$ la sous-arborescence de T dont la racine est le sommet X étiqueté par la variable B. Si X est un sommet interne de T différent de la racine de T, il existe $u, z \in (V \cup \Sigma)^*$ tel que $w(T) = uw(T[X, B])z$.

Nous sommes prêts pour le théorème.

Théorème 1 *Soit L un langage hors-contexte sur l'alphabet Σ. Alors il existe un entier positif p tel que pour tout mot $w \in L(G)$ de longueur au moins p il existe $u, v, x, y, z \in \Sigma^*$ vérifiants :*

1. *$w = uvxyz$;*
2. *$|vxy| \leq p$;*
3. *$|vy| > 0$;*
4. *$uv^i xy^i z \in L$ pour tout $i \in N$.*

Démonstration. Si $L = \emptyset$ ou $L = \{\varepsilon\}$, le théorème est vrai trivialement. On peut donc supposer que $\varepsilon \notin L \neq \emptyset$. Puisque L est hors-contexte, il existe une grammaire $G = (V, \Sigma, R, S)$ telle que $L = L(G)$. Par théorème 3, on peut supposer qu'aucune règle n'est de la forme $A \longrightarrow \varepsilon$ ou $A \longrightarrow B$, $A, B \in V$. Soit $b = max\{|\alpha| : A \longrightarrow \alpha \in R\}$. Si $b \leq 1$, $L(G) \subseteq \Sigma \cup \{\varepsilon\}$ (exercice facile) et le théorème est vrai. Supposons donc que $b \geq 2$. On va montrer que $p = b^{|V|+1}$ peut être la constante du théorème.
Soit $w \in L$, $|w| \geq p$. Soit $T_G(S) = T$ un arbre d'analyse pour w, i.e. tel que $w(T) = w$.

Puisque $|w| \geq b^{|V|+1}$, T a au moins $|V|+2$ niveaux. Soit C un chemin le plus long de la racine de T vers une feuille; C contient au moins $|V|+2$ sommets dont le dernier est une feuille. Les $|V|+1$ étiquettes des sommets internes de C ne peuvent pas toutes être distinctes car il n'y en a que $|V|$. En parcourant C à partir de la feuille qui le termine, soit A la première étiquette (variable) qui apparaît pour la deuxième fois. Soit X le sommet de T étiqueté par cette deuxième occurrence de A et soit Y le sommet de T étiqueté par la première occurrence de A. Il existe $u, v, y, z \in \Sigma^*$ tels que $w = w(T) = uw(T[X, A])z = uvw(T[Y, A])yz$. En mettant $x = w(T[Y, A])$ on obtient la décomposition désirée. Il reste à vérifier les conditions. On vient de voir que $w = uvxyz$. La partie de C contenue dans $T[X, A]$ contient au plus $|V|+2$ sommets car les étiquettes des sommets autres que X et la feuille sont toutes des variables distinctes. Donc $|vxy| = |w(T[X, A])| \leq b^{|V|+1} \leq p$. Si $|vy| = 0$, il y a deux possibilités (pas exclusives). La première, les sommets entre X et Y n'ont qu'un seul enfant chaque et donc G contient des règles de la forme $A \longrightarrow B$. La deuxième, la règle utilisé à X est $A \longrightarrow X_1 ... X_k$ telle que pour un $1 \leq i \leq k$ on a $X_i \stackrel{*}{\Longrightarrow} x$ et $X_j \stackrel{*}{\Longrightarrow} \varepsilon$ pour $1 \leq j \leq k,] \neq i$ et dans ce cas G contient des règles de la forme $A \longrightarrow \varepsilon$. Les deux cas sont impossible par le choix de G. Finalement, à partir de $T = T_1$ on peut obtenir l'arbre T_0 en enlevant $T[Y, A]$ de façon à obtenir $w(T_0) = uxz$ et, récursivement, T_i à partir de T_{i-1}, pour $i \geq 2$ en remplaçant $T[Y, A]$ par une copie de $T[X, A]$ (ceci pourrait être fait plus précis au prix de cacher ce qui se passe). □

Le théorème peut être renforcé. Pour le faire, nous introduisons encore un peu de notation. Etant donné un mot $w \in \Sigma^*$, on voudrait distinguer certaines des positions de ce mot. Par exemple, si $w = aabbcc$, on voudrait "marquer" le premier a, le deuxième b et deuxième c. Une façon de le faire est de définir, pour $u = a_i a_{i+1} \dots a_j$, $I(u) = \{i, i+1, \dots, j\}$ (donc $|u| = j - i + 1$) et $D \subseteq I(u)$. En particulier, si $w = a_1 a_2 \dots a_n$, $|w| = n$ et $D(w) \subseteq \{1, \dots, n\}$. Dans notre exemple, $n = 6$ et $D(w) = \{1, 4, 6\}$. Si $w = a_1 a_2 \dots a_n$ est fixé et si $u = a_i a_{i+1} \dots a_j$, on définit $D(u) = I(u) \cap D(w)$. Le résultat à prouver est analogue au théorème précédent (et on va simplement copier des parties de la preuve).

Théorème 2 *Soit L un langage hors-contexte sur l'alphabet Σ. Alors il existe un entier positif p tel que pour tout mot $w = a_1 \dots a_n \in L(G)$ avec $|D(w)| \geq p$ il existe $u, v, x, y, z \in \Sigma^*$ vérifiants :*

1. *$w = uvxyz$;*
2. *$|D(vxy)| \leq p$;*
3. *$|D(vy)| > 0$;*
4. *$uv^i xy^i z \in L$ pour tout $i \in N$.*

Démonstration. Puisque L est hors-contexte, il existe une grammaire $G = (V, \Sigma, R, S)$ telle que $L = L(G)$. Soit $b = max\{|\alpha| : A \longrightarrow \alpha \in R\}$. Si $b \leq 1$, $L(G) \subseteq \Sigma \cup \{\varepsilon\}$ (exercice facile) et le théorème est vrai. Supposons donc que $b \geq 2$. On va montrer que $p = b^{|V|+1} + 1$ peut être la constante du théorème.

Soit $w \in L$, $w = a_1 \dots a_n$ et $|D(w)| \geq p$. Soit $T_G(S) = T$ un arbre d'analyse pour w, i.e. tel que $w(T) = w$. Soit $D(T)$ l'ensemble de sommets de T ayant au moins deux enfants Y, Z tels que $I(w(T[Y, e(Y)] \cap D(w) \geq 1$ et $I(w(T[Z, e(Z)] \cap D(w) \geq 1$, c'est-à-dire, $D(T)$ est l'ensemble de sommets ayant au moins deux enfants étiqutés par des variables qui donnent chacune au moins un a_i, $i \in D$, i.e. qui donne chacune au moins une position marquée. Puisque $|D(w)| \geq b^{|V|+1}+1$, T contient un chemin de la racine vers une feuille avec au moins $|V|+1$ sommets de $D(T)$. Soit C un tel chemin. Les $|V|+1$ étiquettes des sommets de C qui sont dans $D(T)$ ne peuvent pas toutes être distincts car il n'y en a que $|V|$. En parcourant C à partir de la feuille qui le termine, soit A la première étiquette (variable) qui apparaît pour la deuxième fois sur un sommet de $D(T)$. Soit X le sommet de T étiqueté par cette deuxième occurrence de A et soit Y le sommet de $D(T)$ sur C étiqueté par la première occurrence de A. Il existe $u, v, y, z \in \Sigma^*$ tels que $w = w(T) = uw(T[X, A])z = uvw(T[Y, A])yz$. En mettant $x = w(T[Y, A])$ on obtient la décomposition désirée. Il reste à vérifier les conditions.
On vient de voir que $w = uvxyz$. La partie de C contenue dans $T[X, A]$ contient au plus $|V|+1$ sommets de $D(T)$, donc $D(vxy)| \leq b^{|V|+1} \leq p$. De plus, $|D(vy)| > 0$ car de $X \in D(T)$. Finalement, à partir de $T = T_1$ on peut obtenir l'arbre T_0 en enlevant $T[Y, A]$ de façon à obtenir $w(T_0) = uxz$ et, récursivement, T_i à partir de T_{i-1}, pour $i \geq 2$ en remplaçant $T[Y, A]$ par une copie de $T[X, A]$. □

En analogie avec ce qu'on a vu pour les automates finis, on dit que deux grammaires G et G' sont *équivalentes* si $L(G) = L(G')$.

Comme dans les automates, on peut avoir beaucoup de redondance dans une grammaire. En effet, non seulement peut-il y avoir des variables inutilisées, mais aussi des chaînes de règles comme $A \longrightarrow B$, $B \longrightarrow C$, $C \longrightarrow D$, etc. Mais on peut simplifier. Dans la suite, une grammaire est toujours hors contexte.

Lemme 2 *Pour toute grammaire G tel que $\varepsilon \notin L(G)$ il existe une grammaire G' équivalente sans règles de la forme $A \longrightarrow \varepsilon$, $A \in V$.*

Démonstration. La première idée est de choisir une règle $A \longrightarrow \varepsilon$, l'enlever de R, et de la remplacer par toutes les variantes de toute règle ayant A à droite ($B \longrightarrow X_1, \dots A \dots X_s$). Les variantes comprennent tous les remplacements possibles des A par ε (c'est-à-dire, on enlève simplement les A) pour chaque ensemble d'occurrences de A dans $X_1 \dots X_s$. Si on fait cela pour chaque règle $A \longrightarrow \varepsilon$, on devrait pouvoir garder le langage de la grammaire obtenue inchangé.
L'idée est bonne, mais a plusieurs problèmes telle quelle. D'abord, si $X_1 = X_2 = \dots X_s = A$, on ajouterait $B \longrightarrow \varepsilon$. En plus , de tels remplacements ne sont pas suffisants pour prouver que que $S \stackrel{*}{\Longrightarrow} w$ dans G si et seulement si $S \stackrel{*}{\Longrightarrow} w$ dans la nouvelle grammaire. Essayons de faire mieux: ce n'est pas les $A \longrightarrow \varepsilon$ qu'il faut remplacer, mais $A \stackrel{*}{\Longrightarrow} \varepsilon$.

Soit donc $G = (V, \Sigma, R, S)$. Soit $E = \{A \in V : A \stackrel{*}{\Longrightarrow} \varepsilon\}$ (voir la remarque suivant la preuve). Toute règle r de R est de la forme $A \longrightarrow X_1 \dots X_s$, $X_i \in V \cup \Sigma$. Mettons $|r| = s$ et définissons $O_r = \{j \in \{1, \dots, |r|\} : X_j \in E\}$, c'est-à-dire, O_r est l'ensemble de positions des variables qui peuvent devenir ε. Pour chaque T, $\{1, \dots, |r|\} \neq T \subseteq O_r$, soit r_T la règle

$A \longrightarrow Y_1 \dots Y_{|r|}$ avec $Y_i = \varepsilon$ si $i \in T$ et $Y_i = X_i$ sinon. Notons la raison pour exiger que $\{1, \dots, |r|\} \neq T$: on ne veut pas réintroduire une règle $A \stackrel{*}{\Longrightarrow} \varepsilon$.
Soit maintenant $G' = (V, \Sigma, R', S)$ définie par $R' = \{r_T : r \in R, \{1, \dots, |r|\} \neq T \subseteq O_r\}$. Notons que $R \setminus \{A \longrightarrow \varepsilon\} \subseteq R'$ car toute règle de la forme $A \longrightarrow \alpha,\ \alpha \neq \varepsilon$, est dans R' pour $T = \emptyset$. Il reste à prouver que $L(G) = L(G')$. Ceci est fait en prouvant que pour tout $w \in \Sigma^*$, $w \neq \varepsilon$, et tout $A \in V$, $A \stackrel{*}{\Longrightarrow} w$ dans G si et seulement si $A \stackrel{*}{\Longrightarrow} w$ dans G'. Dans les deux directions, on procède par récurrence sur la longueur de la dérivation. On observe d'abord que pour aucune variable $A \in V$, $A \stackrel{*}{\Longrightarrow} \varepsilon$ dans G', car il n'y a pas de règles de la forme $B \longrightarrow \varepsilon$ dans R'.

Supposons que $A \stackrel{k}{\Longrightarrow} w$ dans G. Si $k = 1$, la règle $A \longrightarrow w$ est dans R, et, par une observation plus haut, elle est dans R'. Donc $A \stackrel{*}{\Longrightarrow} w$ dans G'. Pour $k > 1$, si $A \stackrel{k}{\Longrightarrow} w$ dans G, alors il existe une règle $A \longrightarrow X_1 \dots X_s$ dans R telle que $A \stackrel{1}{\Longrightarrow} X_1 \dots X_s \stackrel{k-1}{\Longrightarrow} w = w_1 \dots w_s$, avec $X_i \stackrel{k_i}{\Longrightarrow} w_i$ pour $i = 1, \dots s$ et $k_i < k$. Appelons cette règle r. Par l'hypothèse de récurrence, $X_i \stackrel{*}{\Longrightarrow} w_i$ dans G' pour chaque i tel que $w_i \neq \varepsilon$. Soit $T = \{j \in \{1, \dots, s\} : w_j = \varepsilon\}$. Alors dans $r_T \in R'$ et on a, dans G', une dérivation de w par $A \longrightarrow Y_1 \dots Y_s \stackrel{*}{\Longrightarrow} w$, avec $Y_i = \varepsilon$ si $i \in T$ et $Y_i = X_i$ sinon.
Supposons que $A \stackrel{k}{\Longrightarrow} w$ dans G'. Si $k = 1$, la règle $A \longrightarrow w$ est dans R', et alors soit elle est dans R, soit il existe une règle $A \longrightarrow X_1 \dots X_s$, appelons la r, dans R et un $T \subseteq \{1, \dots, s\} \neq T$ tels que $A \longrightarrow w$ soit r_T. Dans les deux cas, $A \stackrel{*}{\Longrightarrow} w$ dans G: directement dans le premier, et par $A \stackrel{1}{\Longrightarrow} X_1 \dots X_s \stackrel{*}{\Longrightarrow} w$ dans le second. Pour $k > 1$, si $A \stackrel{k}{\Longrightarrow} w$ dans G', alors il existe une règle $A \longrightarrow X_1 \dots X_s$ dans R' telle que $A \stackrel{1}{\Longrightarrow} X_1 \dots X_s \stackrel{k-1}{\Longrightarrow} w = w_1 \dots w_s$, avec $X_i \stackrel{k_i}{\Longrightarrow} w_i \neq \varepsilon$ pour $i = 1, \dots s$ et $k_i < k$. Pour être dans R', il doit y avoir une règle $r \in R$ telle que $A \stackrel{1}{\Longrightarrow} X_1 \dots X_s$ soit r_T pour un $T \subseteq \{1, \dots, |r|\},\ T \neq \{1, \dots, |r|\}$. Soit r la règle $A \longrightarrow Y_1 \dots Y_{|r|}$. Par l'hypothèse de récurrence, pour $i = 1, \dots, s$, $X_i \stackrel{*}{\Longrightarrow} w_i$ dans G. On a donc une dérivation, dans G, de w $A \stackrel{1}{\Longrightarrow} Y_1 \dots Y_{|r|} \stackrel{*}{\Longrightarrow} X_1 \dots X_s \stackrel{*}{\Longrightarrow} w_1 \dots w_s = w$. □

Notons que l'ensemble E peut être obtenu itérativement: si $E_1 = \{A \in V : A \longrightarrow \varepsilon \in R\}$ et si pour $i > 1$ on met $E_i = E_{i-1} \cup \{A \in V : A \longrightarrow X_1 \dots X_s,\ X_j \in E_{i-1},\ j = 1, \dots, s\}$, alors on peut mettre $E = E_{|V|}$ car à chaque itération on peut ajouter une variable, donc après $|V|$ on ne pourra plus changer E.

De manière semblable, on peut prouver les lemmes suivants (*les preuves seront ajoutées plus tard*).

Lemme 3 *Pour toute grammaire hors contexte $G = (V, \Sigma, R, S)$ il existe une grammaire $G' = (V', \Sigma, R', S)$ équivalente sans règles de la forme $A \longrightarrow B$, $A, B \in V'$.*

Démonstration. (*Esquisse*) On peut procéder par récurrence sur le nombre de règles de la forme $A \longrightarrow B$. S'il n'y en a pas dans G, $G' = G$ et on a terminé. Supposons que $G = G_0$ contient k règles de la forme $A \longrightarrow B$, on va trouver une grammaire G_1 équivalente qui a $k - 1$ règles de cette forme (ce qui prouve que G_k en a zéro). En effet, soit $A \longrightarrow B$ une règle de G. Alors $G_1 = (V, \Sigma, R_1, S)$ aura $R_1 = (R \setminus \{A \longrightarrow B)) \cup \{A \longrightarrow \alpha : B \longrightarrow \alpha \in R\}$.

Ceci permet d'avoir, pour $w \in (V \cup \Sigma)^*$, $S \stackrel{*}{\Longrightarrow} w$ dans G si et seulement si $S \stackrel{*}{\Longrightarrow} w$ dans G'. □

Exercice 4 *Si vous avez compris l'esquisse de la preuve du lemme 3, expliquer pourquoi elle n'est pas bonne. Comment peut-on la réparer?*

Soit $G = (V, \Sigma, R, S)$ une grammaire hors-contexte. Soit $\alpha = X_1 X_2 \dots X_k \in (V \cup \Sigma)^*$. Définissons $v(\alpha) = \{X_i : 1 \leq i \leq k \text{ et } X_i \in V\}$ (i.e. $v(\alpha)$ est l'ensemble des variables qui apparaîssent dans α.

Lemme 4 *Pour toute grammaire hors contexte $G = (V, \Sigma, R, S)$ il existe une grammaire $G' = (V', \Sigma, R', S)$ équivalente telle que pour tout $A \in V'$ il existe $w \in \Sigma^*$ et $\alpha, \beta \in (V \cup \Sigma)^*$ tels que $A \stackrel{*}{\Longrightarrow} w$ et $S \stackrel{*}{\Longrightarrow} \alpha A \beta$.*

Démonstration. (*Esquisse*) Voici un algorithme qui construit une telle grammaire. Soit $R_0 = \{A \longrightarrow \alpha \in R : \alpha \in \Sigma^*\}$ et soit $V_0 = \{A \in V : A \longrightarrow \alpha \in R_0\}$. Pour $i > 0$, soit $R_i = R_{i-1} \cup \{A \longrightarrow \alpha : v(\alpha) \subseteq V_{i-1}\}$ et soit $V_i = \{A \in V : A \longrightarrow \alpha \in R_i\}$ (i.e., V_i contient toutes les variables qui apparaîssent à droite dans une règle de R_i). Après au plus $|R|$ étapes, $R_i = R_{i+1}$ et $V_i = V_{i+1}$ (pourquoi?). On peut donc définir $R'' = R_{|R|}$ et $V'' = V_{|R|}$ (on n'a pas fini). Si $S \notin V'$, alors $L(G) = L(G') = \emptyset$ et $G' = (\{S\}, \Sigma, \emptyset, S)$. Sinon, $R' = R'', V' = V''$. Il reste à prouver que $L(G) = L(G')$. Ceci se fait facilement : pour $w \in \Sigma^*$, $S \stackrel{*}{\Longrightarrow} w$ dans G si est seulement si toutes les variables utilisée dans la dérivation de w mènent aux mots sur Σ^*, ce qui arrive si et seulement si $S \stackrel{*}{\Longrightarrow} w$ in G'. Pour le prouver en détail, il faut montrer que pour $w \in \Sigma^*$ et $A \in V$, $A \stackrel{*}{\Longrightarrow} w$, in G si et seulement si $A \in V'$ si et seulement si $A \stackrel{*}{\Longrightarrow} w$ in G'. □

Ce dernier lemme dit que l'on peut obtenir une grammaire dans laquelle chaque variable sert dans une dérivation d'un mot de $L(G')$.

Les lemmes ensemble prouvent le théorème suivant.

Théorème 3 *Soit L un langage hors-contexte sur l'alphabet Σ. Alors $L \setminus \{\varepsilon\}$ peut être généré par une grammaire hors-contexte $G = (V, \Sigma, R, S)$ telle que*

1. *pour tout $A \in V$ il existe $w \in \Sigma^*$ avec $A \stackrel{*}{\Longrightarrow} w$ et $\alpha, \beta \in (V \cup \Sigma)^*$ tels que $S \stackrel{*}{\Longrightarrow} \alpha A \beta$;*
2. *aucune règle n'est de la forme $A \longrightarrow B$, $A, B \in V$;*
3. *aucune règle n'est de la forme $A \longrightarrow \varepsilon$, $A \in V$.*

Si $\varepsilon \in L$, alors il est généré par la grammaire $G_\varepsilon = (V \cup \{S'\}, \Sigma, R', S')$ où $R' = R \cup \{S' \longrightarrow \alpha : S \longrightarrow \alpha \in R\} \cup \{S' \longrightarrow \varepsilon\}$.

Mais l'utilité principale des lemmes est l'obtention d'une forme *normale* d'une grammaire hors contexte.

Définition 3 *Soit $G = (V, \Sigma, R, S)$ une grammaire hors-contexte. On dit que G est en* forme normale de Chomsky *si toute règle est soit de la forme $A \longrightarrow BC$, soit de la forme $A \longrightarrow a$, $A, B, C \in V$, $a \in \Sigma$, $B, C \neq S$, soit $S \longrightarrow \varepsilon$.*

Remarque 1 *On peut inclure dans la définition que toutes les variables sont utile – bien évidemment. les lemme permettent de prouver cette version plus forte.*

Théorème 4 *Pour tout langage hors-contexte L il existe une grammaire hors-contexte en forme normale de Chomsky qui le génère.*

Démonstration. (*Esquisse*) Avec le théorème 3 ceci est simple. Soit $G_0 = (V_0, R_0, \Sigma, S_0)$ une grammaire hors contexte telle que $L = L(G)$. On s'assure d'abord que le symbole de départ n'apparaît dans aucune règle à droite, i.e., si $A \longrightarrow X_1 \dots X_k$ alors $X_i \in ((V \setminus \{S\}) \cup \Sigma))$. Soit $G_1 = (V_1, \Sigma, R_1, S_1)$ où $V_1 = V_0 \cup \{S_1\}$, $R_1 = R_0 \cup \{S_1 \longrightarrow S_0\}$ et $S_1 \notin V_0$. Soit $G_2 = (V_2, \Sigma, R_2, S_2)$ une grammaire dont l'existence est garantie par le théorème 3. A part la règle éventuelle $S_2 \longrightarrow \varepsilon$, toute règle est de la forme $A \longrightarrow X_1 \dots X_k$, $k \geq 2$, $X_i \in V \cup \Sigma$ pour $1 \leq i \leq k$. Pour $a \in \Sigma$, soit A_a une nouvelle variable (i.e. $A_a \notin V$ et $A_a \neq A_b$ si $a \neq b$). On procède en plusieurs étapes.

1. On remplace chaque règle r qui est $A \longrightarrow X_1 \dots X_k$, par $A \longrightarrow Y_1 \dots Y_k$, mettant $Y_i = A_a$ si $X_i = a \in \Sigma$ et $Y_i = X_i$ si $X_i \in V$;
2. On remplace $A \longrightarrow Y_1 \dots Y_k$ par $A \longrightarrow Y_1 Z_2^r$, $Z_2^r \longrightarrow Y_2 Z_3^r, \dots, Z_i^r \longrightarrow Y_i Z_{i+1}^r, \dots Z_{k-1}^r \longrightarrow Y_{k-1} Y_k$ où les Z_i^r sont des nouvelles variables propres à la règle r.
3. On ajoute la règle $A_a \longrightarrow a$ pour tout $a \in \Sigma$.

On obtient ainsi la grammaire $G_3 = (V_3, \Sigma, R_3, S_3)$ avec $V_3 = V_2 \cup \{A_a : a \in \Sigma\} \cup \{Z_i^r : r \in R_2 \text{ est } A \longrightarrow X_1 \dots X_{|r|},\ i = 2, \dots, |r| - 1\}$ ($A_a \notin V_2$, et $|r|$ défini comme dans la preuve du lemme 2), $S_3 = S_2$, et R_3 défini dans les étapes décrites ci-haut. On laisse comme exercice la preuve par récurrence que la nouvelle grammaire génère exactement le même langage L que G. □

Soit X un alphabet. Définissons $X_\varepsilon = X \cup \{\varepsilon\}$.

Définition 4 *Un* automate à pile $M = (Q, \Sigma, \Gamma, \delta, q_0, F)$ *est défini par*

- *un ensemble fini Q d'états ;*
- *un alphabet du rubanΣ ;*
- *un alphabet de pile Γ ;*
- *un état initial q_0 ;*
- *un ensemble d'états acceptants $F \subseteq Q$;*

- *une fonction de transition* $\delta : Q \times \Sigma_\varepsilon \times \Gamma_\varepsilon \longrightarrow 2^{Q \times \Gamma_\varepsilon}$.

C'est-à-dire, quand M est dans l'état q en lisant $a \in \Sigma_\varepsilon$ sur le ruban et avec $A \in \Gamma_\varepsilon$ en haut de la pile, la fonction δ nous donne un choix d'états et un choix de remplacements de A. Si $\delta(q, a, A) = \{(q_1, A_1), \ldots, (q_k, A_k)\}$, M pourra se mettre dans un état q_i et remplacer A en haut de la pile par A_i. Notons que la machine est non déterministe également en permettant de simplement remplacer le haut de la pile sans rien lire ($a = \varepsilon$), d'ajouter un symbole sur la pile $A = \varepsilon$, ou d'en retirer un $A_i = \varepsilon$. *D'autre définitions sont possibles, mais toutes sont équivalentes. Par contre, il n'y pas toujours un automate à pile déterministe équivalent.*

On remarque que le non-déterminisme peut être vu comme modélisant le fait que dans une grammaire hors-contexte on peut remplacer une variable de plusieurs façons.

Soit $M = (Q, \Sigma, \Gamma, \delta, q_0, F)$ un automate à pile (non déterministe). Une *description instantanée* ou un *état instantané* de M pendant son exécution sur un mot $w \in \Sigma^*$ est un triplet (q, u, α) où q est l'état de M au moment où la tête est sur (lit) le premier symbole de $u \in \Sigma^*$, avec $\alpha \in \Gamma^*$ sur la pile. On dit que l'état instantané (q, u, α) *mène à* ou *donne* l'état instantané (p, v, β) en k étapes, et on écrit $(q, u, \alpha) \overset{k}{\vdash} (p, v, \beta)$, si

- $k = 0, q = p, u = v, \alpha = \beta$;
- $k = 1$, $u = av$, $a \in \Sigma \cup \{\varepsilon\}$, $\alpha = A\gamma$, $A \in \Gamma$, $\gamma \in \Gamma^*$, $\beta = \sigma\gamma$ et $(p, \sigma) \in \delta(q, a, A)$, $\sigma \in \Gamma^*$;
- $k > 1$, $(q, u, \alpha) \overset{k-1}{\vdash} (p', v, \beta')$ et $(p', v, \beta') \overset{1}{\vdash} (p, v, \beta)$.

On écrit souvent $\vdash$ pour $\overset{1}{\vdash}$.

Notons que cette définition est bien plus simple et plus courte que la définition équivalente (preuve?) suivante :

On dit que l'état instantané q, u, α *mène à* ou *donne* l'état instantané (p, v, β) en k étapes, si $(q, u, \alpha) \overset{k}{\vdash} (p, v, \beta)$ est défini comme ci-haut pour $k = 1$, et si, pour $k > 1$, existe

- $q_0, \ldots, q_k \in Q$, $q_0 = q, q_k = p$;
- $A_1, \ldots A_k \in \Gamma_\varepsilon$;
- $\alpha_1, \ldots, \alpha_k \in \Gamma^*$;
- $a_1, \ldots, a_k \in (\Sigma \cup \{\varepsilon\})$

tels que

- $\alpha = A_1\alpha_1$, $u = a_1 \ldots a_k v$;
- pour $i = 1, \ldots k-2$, $(q_i, a_{i+1}a_{i+2} \ldots a_k v, A_{i+1}\alpha_{i+1}) \overset{1}{\vdash} (q_{i+1}, a_{i+2} \ldots a_k v, A_{i+2}\alpha_{i+2})$;

- $(q_{k-1}, a_k v, A_k \alpha_k) \overset{1}{\vdash\!\!-} (q_k, v, \beta)$.

On dit que (q, u, α) donne (mène à) (p, v, β), et on écrit $(q, u, \alpha) \overset{*}{\vdash\!\!-} (p, v, \beta)$, s'il y a un k tel que $(q, u, \alpha) \overset{k}{\vdash\!\!-} (p, v, \beta)$.
Avec ceci on peut définir le langage reconnu par M comme $L(M) = \{w \in \Sigma^* : \text{il existe } p \in F,\ \alpha \in \Gamma^* \text{ tels que } (q_0, w, \varepsilon) \overset{*}{\vdash\!\!-} (p, \varepsilon, \alpha)\}$.

Comme dans les cas précédents, on dit que deux automates à piles M et M' sont *équivalents* si $L(M) = L(M')$. On dit également qu'un automate à pile M et une grammaire hors-contexte G sont équivalents si $L(M) = L(G)$.

Proposition 1 *Pour tout langage hors-contexte L il existe un automate à pile (non déterministe) qui le reconnait.*

Démonstration. Soit L un langage hors-contexte. Puisque tout langage hors-contexte est généré par une grammaire hors-contexte en forme normale de Chomsky, on peut supposer que l'on ait une telle grammaire $G = (V, \Sigma, R, S)$ pour L. Soit $M = (Q, \Sigma, \Gamma, \delta, q_o, F)$ l'automate avec l'alphabet Σ défini par

- $Q = \{q_0, q, f\} \cup \{q_A : A \in V\}$, $q_A \neq q_0, q, f$;
- $\Gamma = \{\$\} \cup V$;
- $F = \{f\}$

avec l'état initial q_0 et δ donnés par
$\delta(q_0, \varepsilon, \varepsilon) = \{(q_S, \$)\}$
$\delta(q, a, A) = \{(q, \varepsilon)\}$ pour tout $A \in V, a \in \Sigma$ tels que$A \longrightarrow a \in R$
$\delta(q, \varepsilon, A) = \{(q_B, C)\}$ pour tout $A, B, C \in V$ tels que $A \longrightarrow BC \in R$
$\delta(q, \varepsilon, S) = \{(q, \varepsilon)\}$ si $S \longrightarrow \varepsilon \in R$
$\delta(q_B, \varepsilon, \varepsilon) = \{(q, B)\}$ pour tout $B \in V$
$\delta(q, \varepsilon, \$) = \{(f, \varepsilon)\}$.

Il faut maintenant prouver que l'automate décide exactement le langage généré par la grammaire. On le fait en prouvant que pour tout $u \in \Sigma^*$ et $x \in V^*$, $S \overset{*}{\Longrightarrow} ux$ si et seulement si $(q, u, S\$) \overset{*}{\vdash\!\!-} (q, \varepsilon, x\$)$. Il est - devrait être - clair que ceci implique que $S \overset{*}{\Longrightarrow} w$ si et seulement si $(q_0, w, \varepsilon) \overset{*}{\vdash\!\!-} (f, \varepsilon, \varepsilon)$. On prouve chacune des deux implications par récurrence sur la longueur de la dérivation ou de l'exécution, suivant le cas. On considère, dans G, que les dérivations sont faites en appliquant, à chaque étape, une règle à la première variable de la gauche (il est "évident" que l'on peut faire ceci, mais voir aussi la remarque après la preuve). Soit donc $u \in \Sigma^*$ et $x \in V^*$.
Supposons d'abord que $S \overset{k}{\Longrightarrow} ux$ et prouvons que $(q, u, S\$) \overset{*}{\vdash\!\!-} (q, \varepsilon, x\$)$. Pour $k = 1$, $S \overset{1}{\Longrightarrow} ux$ uniquement si soit $u = \varepsilon = x$, soit $u = \varepsilon$ et $x = BC$, $B, C \in V$. Mais alors

$(q, \varepsilon, S\$) \overset{1}{\vdash} (q, \varepsilon, \$)$ dans le premier cas et $(q, \varepsilon, S\$) \overset{1}{\vdash} (q_B, \varepsilon, C\$) \overset{1}{\vdash} (q, \varepsilon, BC\$)$ dans le deuxième car δ comprend les transitions nécessaires. Pour $k > 1$, si $S \overset{k}{\Longrightarrow} ux$, alors soit $S \overset{k-1}{\Longrightarrow} uAy$, $x = BCy,\ A, B, C \in V,\ A \longrightarrow BC \in R$, soit $S \overset{k-1}{\Longrightarrow} vAx,\ u = va,\ A \in V, A \longrightarrow a \in R$. Dans le premier cas,$(q, u, S\$) \overset{*}{\vdash} (q, \varepsilon, Ay\$)$ par l'hypothèse de récurrence, et alors

$$(q, u, S\$) \overset{*}{\vdash} (q, \varepsilon, Ay\$) \overset{1}{\vdash} (q_B, \varepsilon, Cy\$) \overset{1}{\vdash} (q, \varepsilon, BCy\$) = (q, \varepsilon, x\$).$$

Dans le deuxième cas, $(q, va, S\$) \overset{*}{\vdash} (q, \varepsilon, Ax\$)$ par l'hypothèse de récurrence, et alors

$$(q, u, S\$) \overset{*}{\vdash} (q, a, Ax\$) \overset{1}{\vdash} (q, \varepsilon, x\$)$$

ce qui prouve la première partie.

Dans l'autre sens, supposons que $(q, u, S\$) \overset{k}{\vdash} (q, \varepsilon, x\$)$. Si $k = 1$, la seule transition qui permet d'enlever S de la pile est $\delta(q, \varepsilon, S\$) = \{(q, \varepsilon)\}$, qui est présente uniquement si $S \longrightarrow \varepsilon \in R$. Donc $u = \varepsilon$ et $S \overset{*}{\Longrightarrow} \varepsilon$. Si $k > 1$, alors soit $u = va$ et $(q, u, S\$) \overset{k-1}{\vdash} (q, a, Ax\$) \overset{1}{\vdash} (q, \varepsilon, x\$)$, soit $x = BCy$ et $(q, u, S\$) \overset{k-2}{\vdash} (q, \varepsilon, Ay\$) \overset{1}{\vdash} (q_B, \varepsilon, Cy\$) \overset{1}{\vdash} (q, \varepsilon, x\$)$, et, dans le premier cas $S \overset{*}{\Longrightarrow} vAx \overset{1}{\Longrightarrow} vax = ux$, dans le deuxième $S \overset{*}{\Longrightarrow} uAy \overset{1}{\Longrightarrow} uBCy = ux$, par l'hypothèse de récurrence et les règles de la grammaire présentes. □

Remarque 2 *Encore une fois, on voit que ce qui est évident l'est nettement moins si on essaye de donner une preuve détaillée. En effet, cette preuve aurait été bien plus facile si on avait montré que pour chaque grammaire hors-contexte il en existe une équivalente en* forme normale de Greibach *dont les règles sont de la forme $A \longrightarrow ax,\ a \in \Sigma,\ x \in V^*$.*

Exercice 5 *Prouvez en détails que si pour tout $u \in \Sigma^*$ et $x \in V^*$, $S \overset{*}{\Longrightarrow} ux$ si et seulement si $(q, u, S\$) \overset{*}{\vdash} (q, \varepsilon, x\$)$ alors $S \overset{*}{\Longrightarrow} w$ si et seulement si $(q_0, w, \varepsilon) \overset{*}{\vdash} (f, \varepsilon, \varepsilon)$.*

On veut maintenant la converse : pour tout automate à pile il existe une grammaire hors-contexte équivalente. La preuve rappelle celle la construction d'une expression régulière à partir d'un automate fini. Pour prouver la proposition, il nous faut trois petits lemmes qui permettent de commencer par un automate à pile dans une forme spéciale. Les deux premiers sont tellement simples que leur preuve devient un exercice.

Lemme 5 *Pour tout automate à pile $M = (Q, \Sigma, \Gamma, \delta, s, F)$ il existe un automate à pile équivalent $M' = (Q', \Sigma, \Gamma', \delta', s', F')$ tel que $|F'| = 1$, i.e., M' a un état acceptant unique.*

Démonstration. Exercice. □

Lemme 6 *Pour tout automate à pile $M = (Q, \Sigma, \Gamma, \delta, s, F)$ il existe un automate à pile équivalent $M' = (Q', \Sigma, \Gamma', \delta', s', F')$ avec un état acceptant unique et tel que M' n'accepte aucun mot si sa pile n'est pas vide (i.e. M' vide sa pile avant d'accepter).*

Démonstration. Exercice. □

Lemme 7 *Pour tout automate à pile $M = (Q, \Sigma, \Gamma, \delta, s, F)$ il existe un automate à pile équivalent $M' = (Q', \Sigma, \Gamma', \delta', s', F')$ tel que chaque transition soit ajoute un symbole sur la pile, soit en enlève un.*

Démonstration. La différence entre M et M' est dans les transitions qui remplacent le symbole du haut de la pile par un autre. Il suffit donc de remplacer

1. $(p, B) \in \delta(q, a, A)$ (la transition qui remplace A par B sur la pile) par deux transitions : $(p_B, \varepsilon) \in \delta'(q, a, A)$ (une transition qui enlève A de la pile et qui se souvient de ce qu'il faut y mettre, B et dans quel état il faut arriver, p) et $\{(p, B)\} = \delta'(p_B, \varepsilon, \varepsilon)$ (qui met B sur la pile et met M' dans l'état p);

2. $(p, \varepsilon) \in \delta(q, a, \varepsilon)$ (une transition que ne change rien à la pile) par $(p_X, X) \in \delta'(q, a, \varepsilon)$ (on met un symbole X bidon sur la pile) et $\{(p, \varepsilon)\} = \delta'(p_X, \varepsilon, X)$ (on enlève X et on met M' dans l'état p).

Plus précisément, on met

- $\Gamma' = \Gamma \cup \{X\},\ X \notin \Gamma$;
- $s' = s$;
- $F' = F$;
- $Q' = Q \cup \{p_B : B \in \Gamma \cup \{X\}, p \in Q\},\ p_B \notin Q$;
- pour $a \in \Sigma \cup \{\varepsilon\}$ on définit $\delta'(q, a, A)$ par

  si $q \in Q$ alors

  $\delta'(q, a, A) = \{(p, B) : (p, B) \in \delta(q, a, A)$ et exactement un de A, B est $\varepsilon\} \cup \{(p_B, \varepsilon) : (p, B) \in \delta(q, a, A)$ et $A, B \in \Gamma\} \cup \{(p_X, X) : (p, \varepsilon) \in \delta(q, a, A)$ et $A = \varepsilon\}$

  si $q = p_B$ pour un $p \in Q,\ B \in \Gamma \cup \{X\}$ alors

  $\delta'(p_B, \varepsilon, \varepsilon) = \{(p, B)\}$ pour $B \neq X$
  $\delta'(p_X, \varepsilon, X) = \{(p, \varepsilon)\}$

Pour voir l'équivalence des deux automates, on observe que pour tout $q \in Q, u \in \Sigma^*, \alpha \in \Gamma^*$, $(q,u,\alpha) \overset{*}{\vdash\!\!\!-} (p,v,\beta)$ dans M si et seulement si $q \in Q, u \in \Sigma^*, \alpha \in \Gamma^*$, $(q,u,\alpha) \overset{*}{\vdash\!\!\!-} (p,v,\beta)$ dans M'. Ceci se fait par récurrence sur la longueur de la dérivation. Pour $k=0$, il n'y a rien à prouver. Supposons que $(q,u,\alpha) \overset{k}{\vdash\!\!\!-} (p,v,\beta)$ dans M. Alors il existe $a \in \Sigma \cup \{\varepsilon\}, A, B \in \Gamma \cup \{\varepsilon\}, u' \in \Sigma^*, \alpha', \beta' \in \Gamma^*, \beta = B\beta'$ tels que $(q,u,\alpha) \overset{k-1}{\vdash\!\!\!-} (q', au', A\alpha') \overset{1}{\vdash\!\!\!-} (p,v,B\beta')$ et, par l'hypothèse de récurrence, $(q,u,\alpha) \overset{k-1}{\vdash\!\!\!-} (q',au',A\alpha')$ dans M'. Le passage de $(q',au',A\alpha')$ à $(p,v,B\beta')$ est fait par une transition $(p,B) \in \delta(q',a,A)$. Alors soit exactement un de A, B est ε et $(p,B) \in \delta'(q',a,A)$, soit $A, B \in \Gamma$ et cette transition peut être remplacée par $(p_B, \varepsilon) \in \delta'(q',a,A)$ suivie de $(p,B) \in \delta'(p_B, \varepsilon, \varepsilon)$, soit $A = B = \varepsilon$ et la transition peut être remplacée par $(p_X, X) \in \delta'(q',a,A)$ suivie de $(p,\varepsilon) \in \delta'(p_X, \varepsilon, \varepsilon)$ pour obtenir $(q',au',A\alpha') \overset{*}{\vdash\!\!\!-} (p,v,\beta)$ dans M'. Dans l'autre sens, si $(q,u,\alpha) \overset{k}{\vdash\!\!\!-} (p,v,\beta)$ dans M', on a trois cas. Dans le premier, $(q,u,\alpha) \overset{k-1}{\vdash\!\!\!-} (q',a\alpha',A\beta') \overset{1}{\vdash\!\!\!-} (p,v,\beta) = (p,v,B\beta')$ par une application de $(p,B) \in \delta'(q',a,A)$, et la même dérivation est valable dans M. Dans le deuxième, $A, B \in \Gamma$ et $(q,u,\alpha) \overset{k-2}{\vdash\!\!\!-} (q',au',A\alpha') \overset{1}{\vdash\!\!\!-} (p_B,v,alpha') \overset{1}{\vdash\!\!\!-} (p,v,B\beta') = (p,v,\beta)$. Donc, dans M, $(q,u,\alpha) \overset{k-1}{\vdash\!\!\!-} (p,v,\beta)$ car les deux dernières transitions peuvent être faites par $(p,B) \in \delta(q',a,A)$. Dans le dernier cas, $(q,u,\alpha) \overset{k-2}{\vdash\!\!\!-} (q',au',\beta) \overset{1}{\vdash\!\!\!-} (p_X,v,X\beta) \overset{1}{\vdash\!\!\!-} (p_X,v,\beta)$. $A = \varepsilon = B$. Dans M on aura $(q,u,\alpha) \overset{k-2}{\vdash\!\!\!-} (q',au,\beta) \overset{1}{\vdash\!\!\!-} (p,v,\beta.$ □

Avec ceci on peut prouver la proposition.

Proposition 2 *Pour tout automate à pile $M = (Q, \Sigma, \Gamma, \delta, s, F)$ il existe une grammaire hors-contexte $G = (V, \Sigma, R, S)$ équivalente.*

Démonstration. On construit cette grammaire en mettant dans V les variables A_{pq} telles que $A \overset{*}{\Longrightarrow} w$ si et seulement si $(p,w,\varepsilon) \overset{*}{\vdash\!\!\!-} (q,\varepsilon,\varepsilon)$, i.e. w est généré par A si et seulement si on peut passer de l'état p et la pile vide à l'état q et la pile vide en lisant w. Pour le faire, on a besoin des lemmes vus plus haut.

Sans perte de généralité, on peut supposer, par les lemmes 6, 5, 7, que M n'a qu'un état acceptant f, qu'il vide sa pile avant d'accepter, et que chacune de ses transitions soit empile soit dépile un symbole. Soit alors $V = \{A_{pq} : p,q \in Q\}$, $S = A_{sf}$ et définissons R. Les règles contiennent trois parties.

1. $R_\varepsilon = \{A_{pp} \longrightarrow \varepsilon : p \in Q\}$;
2. $R_{pqr} = \{A_{pq} \longrightarrow A_{pr}A_{rq} : p,q,r \in Q\}$;
3. $R_\delta = \{A_{pq} \longrightarrow aA_{rs}b : p,q,r,s \in Q,\ a,b \in \Sigma \cup \{\varepsilon\},\ X \in \Gamma,\ (r,X) \in \delta(p,a,\varepsilon), (q,\varepsilon) \in \delta(s,b,X)\}$

L'idée est de simuler le comportement de M par G. Puisque M commence et termine par une pile vide, $(q, w, \varepsilon) \overset{k}{\vdash} (p, \varepsilon, \varepsilon)$ de deux manières possibles pour chaque $k \in \mathbb{N}$. Soit $q = r_0, \ldots, r_k = p$ les états de l'exécution de M sur w qui commence dans q et se termine en p et soit α_i le contenu de la pile et w_i la partie de w qui reste à lire après i étapes de cette exécution. C'est-à-dire, pour $i = 0, \ldots, k$, $(q, w, \varepsilon) = (r_0, w, \alpha_0) \overset{i}{\vdash} (r_i, w_i, \alpha_i)$. On a alors deux possibilités : soit $\alpha_i \neq \varepsilon$ sauf pour $i = 0$ et $i = k$, soit $\alpha_j = \varepsilon$ pour un $j \neq 0, k$. On simule le deuxième cas en s'assurant que $A_{pq} \overset{1}{\Longrightarrow} A_{pr}A_{rq} \overset{*}{\Longrightarrow} w'w''$ avec $r = r_j$, $w'' = w_j$ et w' tel que $w = w'w''$ – c'est la partie R_{pqr} des règles.
Le premier cas est simulé en observant que si la pile n'est vide qu'au début et à la fin de l'exécution, alors le symbole de Γ empilé par la première transition est exactement celui dépilé par la dernière. Il en découle que si $w = aub,\ u \in \Sigma^*,\ a, b \in \Sigma \cup \{\varepsilon\}$, alors $(q, aub, \varepsilon) \overset{1}{\vdash} (r_1, ub, X) \overset{k-1}{\vdash} (r_{k-1}, b, X) \overset{1}{\vdash} (p, \varepsilon, \varepsilon)$. Ceci arrive exactement quand $(r_1, X) \in \delta(q, a, \varepsilon)$ et $(p, \varepsilon) \in (r_{k-1}, b, X)$, ce qui est simulé par R_δ.
Naturellement, on passe de p à p sans rien faire et la partie R_ε simule cette dernière possibilité.

Il faut maintenant prouver que $A_{pq} \overset{*}{\Longrightarrow} w$ si et seulement si $(p, w, \varepsilon) \overset{*}{\vdash} (q, \varepsilon, \varepsilon)$. On fait les deux directions par récurrence sur la longueur de la dérivation.

Supposons que $A_{pq} \overset{k}{\Longrightarrow} w$. Si $k = 0$, alors la seule possibilité est que $p = q$ et $w = \varepsilon$. Mais on a que $(p, \varepsilon, \varepsilon) \overset{0}{\vdash} (p, \varepsilon, \varepsilon)$. Si $k > 0$, alors $A_{pq} \overset{k}{\Longrightarrow} w$ si et seulement si

- soit $A_{pq} \overset{1}{\Longrightarrow} A_{pr}A_{rq} \overset{k-1}{\Longrightarrow} w_1w_2 = w$ avec $A_{pr} \overset{l}{\Longrightarrow} w_1$, $A_{rq} \overset{l'}{\Longrightarrow} w_2$, $l, l' < k$.
- soit $A_{pq} \overset{1}{\Longrightarrow} aA_{rs}b \overset{k-1}{\Longrightarrow} aub$ avec $aub = w$ et $A_{rs} \overset{k-1}{\Longrightarrow} u$.

Dans le premier cas, l'hypothèse de récurrence nous dit que $(p, w1, \varepsilon) \overset{*}{\vdash} (r, \varepsilon, \varepsilon)$ et $(r, w_2, \varepsilon) \overset{*}{\vdash} (q, \varepsilon, \varepsilon)$. Donc $(p, w, \varepsilon) = (p, w_1w_2, \varepsilon) \overset{*}{\vdash} (r, w_2, \varepsilon) \overset{*}{\vdash} (q, \varepsilon, \varepsilon)$.
Dans le deuxième cas, on a $(r, u, \varepsilon) \overset{*}{\vdash} (s, \varepsilon, \varepsilon)$, par l'hypothèse de récurrence. On a aussi que $A_{pq} \overset{1}{\Longrightarrow} aA_{rs}b$, donc $A_{pq} \longrightarrow aA_{rs}b \in R_\delta$, c'est-à-dire, il existe un $X \in \Gamma$ tel que $(r, X) \in \delta(p, a, \varepsilon)$ et $(q, \varepsilon) \in \delta(s, b, X)\}$. Donc $(q, aub, \varepsilon) \overset{1}{\vdash} (r, ub, X) \overset{*}{\vdash} (s, b, X) \overset{1}{\vdash} (q, \varepsilon, \varepsilon)$.

Supposons maintenant que $(p, w, \varepsilon) \overset{k}{\vdash} (q, \varepsilon, \varepsilon)$. Si $k = 0$, la seule possibilité est que $(p, \varepsilon, \varepsilon) \overset{0}{\vdash} (p, \varepsilon, \varepsilon)$ et la règle $A_{pp} \longrightarrow \varepsilon$ montre que $A_{pp} \overset{*}{\Longrightarrow} \varepsilon$. Si $k > 0$, il y a encore deux cas. Soit $r_{i\,i=0}^{k}$ l'exécution de M sur w avec $r_0 = p,\ r_k = q$, et soit w_i et α_i définis comme ci-haut. Alors

- soit pour tout $i \neq 0, k$, $\alpha_i \neq \varepsilon$;
- soit il existe $j \neq 0, k$ tel que $\alpha_i = \varepsilon$.

Dans le premier cas on a $(p, w, \varepsilon) \overset{1}{\vdash} (r_1, w_1, X) \overset{k-2}{\vdash} (r_{k-1}, w_{k-1}, X) \overset{1}{\vdash} (q, \varepsilon, \varepsilon)$. Soit $r = r_1,\ s = r_{k-1},\ w = aw_1 = aub,\ a, b \in \Sigma \cup \{\varepsilon\}, u \in \Sigma^*$. Donc $(r, ub, X) \overset{k-2}{\vdash} (s, b, X)$, i.e. $(r, u, \varepsilon) \overset{k-1}{\vdash} (s, b, \varepsilon)$ et, par l'hypothèse de récurrence, $A_{rs} \overset{*}{\Longrightarrow} u$. Mais la première et la dernière étape de la dérivation impliquent que $A_{pq} \longrightarrow aA_{rs}b \in R_\delta$, donc $A_{pq} \overset{1}{\Longrightarrow} aA_{rs}b \overset{*}{\Longrightarrow} auv = w$.
Dans le deuxième cas, soit $r = r_j$. Il existe alors $w_1, w_2 \in SST$ tels que $w = w_1w_2$ et $(p, w_1w_2, \varepsilon) \overset{l}{\vdash} (r, w_2, \varepsilon) \overset{l'}{\vdash} (q, \varepsilon, \varepsilon)$. Par l'hypothèse de récurrence, on obtient que $A_{pr} \overset{*}{\Longrightarrow} w_1$ et $A_{rq} \overset{*}{\Longrightarrow} w_2$. On a donc que $A_{pq} \overset{1}{\Longrightarrow} A_{pr}A_{rq} \overset{*}{\Longrightarrow} w_1w_2 = w$ car $A_{pq} \longrightarrow A_{pr}A_{rq} \in R_{pqr}$.

Ceci termine la preuve. □