Notes de cours Informatique théorique
(version 1.0 spéciale coronavirus 21 mars 2020)

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.

Ces notes reprennent certains des éléments des notes de cours sur les langages hors-contexte mais ne contiennent pas toutes les définitions. On reprends à partir de la page 5 des notes sur les grammaires hors-contexte. Notre but ici est de donner une preuve plus simple du lemme de pompage pour les langages hors-contexte.

1 notes de cours 2020

Rappellons d'abord que la *profondeur* d'une arborescence $T = (V, E)$ est la longueur (nombre d'arêtes) d'une plus longues chaîne de la racine vers une feuille. Si on définit le niveau L_i de T comme l'ensemble des sommets à distance i de la racine, on voit que la profondeur de T est le plus grand i tel que $L_i \neq \emptyset$. Notons que L_0 contient un seul sommet, la racine.

Une arborescence est k-aire si chaque sommet interne a au plus k enfant (on dit souvent "arbre k-aire" car l'arborescence est sous-entendue). Il est facile de voir que dans un arbre k-aire, $|L_i| \leq k^i$ (prouvez-le!). On peut en déduire (faites-le!) le nombre maximum de feuilles dans un arbre k-aire de profondeur ℓ. Dans le cas de $k = 2$, on parle d'un arbre *binaire* que vous connaissez bien.

Lemme 1 *Soit un arbre binaire $T = (V, E)$ de profondeur ℓ. Alors*

1. *le niveau L_i, $i = 0, 1, \ldots \ell$, contient au plus 2^i sommets; et*
2. *le nombre de feuilles de T est au plus 2^ℓ; et*
3. *le nombre de sommets de T est au plus $2^{\ell+1} - 1$.*

Démonstration. On prouve que $|L_i| \leq 2^i$ par récurrence sur i. Pour $i = 0$, on a $|L_0| = 1$ bien évidemment car un seul sommet est à distance 0 de la racine, celle-ci elle même. Si $|L_i| \leq 2^i$, alors $|L_{i+1}| \leq 2|L_i| \leq 2^{i+1}$ parce que tout sommet de L_i est l'enfant d'un sommet de L_i et chaque sommet de L_i a au plus 2 enfants.
On prouve le deuxième énoncé par récurrence sur ℓ. Si $\ell = 0$, T est un seul sommet est c'est une feuille. Supposons qu'un arbre binaire de profondeur ℓ a au plus 2^ℓ feuille. Soit T un arbre binaire de profondeur $L_{\ell+1}$. Si on enlève les sommets de $L_{\ell+1}$, qui sont tous des feuilles, on obtient un arbre binaire T' de profondeur ℓ avec au plus 2^ℓ feuilles par l'hypothèse de récurrence. Chacun des sommets enlevés était adjacent à un sommet de T' qui est forcément une feuille de T' au niveau ℓ. Donc le nombre de feuilles de T est borné par 2 fois le nombre

de feuilles de T', i.e. par $2 \cdot 2^\ell = 2^{\ell+1}$.
Si T est un arbre binaire de profondeur ℓ, il a $\sum_{i=0}^{\ell} |L_i| \leq \sum_{i=0}^{\ell} 2^i = 2^{\ell+1} - 1$. □

Ceci nous donne un corollaire dont on aura besoin.

Corollaire 1 *Un arbre binaire T avec k feuilles est de profondeur au moins $\lceil \lg k \rceil$.*

Démonstration. On se rappelle que $\lg k = \log_2 k$. Soit ℓ la profondeur de l'abre binaire T. Alors le nombre de feuilles k vérifie

$$k \leq 2^\ell$$

et donc

$$\lg k \leq \ell$$

et ℓ étant un entier

$$\lceil \lg k \rceil \leq \ell.$$

□

Rappellons la définition d'un arbre de dérivation. 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 $F'_1, \ldots, F'_n$, avec $e(F'_k) = Y_k$, $k = 1, \ldots, n$, $Y_j = B \in V$. 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 ℓ 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 F'_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 presque prêts pour le théorème de pompage. Nous allons donner ici une preuve plus simple que celle dans les notes de cours précédentes, mais il nous faut un théorme de plus. On l'énonce mais l'esquisse d'une preuve est dans l'annexe pour ceux qui veulent la voir (l'annexe est pris simplement dans les notes sur les grammaires hors contexte). Rappelons que deux grammaires hors-contexte G et G' sont *équivalentes* si $L(G) = L(G')$.

Définition 1 *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 $S \longrightarrow \varepsilon$.*

Si un langage hors-contexte L ne contient pas ε, les règles seront de la forme $A \longrightarrow BC$, ou $A \longrightarrow a$, $A, B, C \in V$, $a \in \Sigma$, $B, C \neq S$. Si $\varepsilon \in L$, il faut ajouter $S \longrightarrow \varepsilon$. La variable de départ n'est jamais utilise après sa premiře utilisation, elle soit génère ε, soit commence une dérivation d'un mot non-vide.

Théorème 1 *Soit G une grammaire hors-contexte. Il existe une grammaire hors contexte équivalente en forme normale de Chomsky.*

Pour en aprendre un peu sur la forme normale de Chomsky voir aussi cette vidéo. Lisez l'annexe d'abord (Sipser p.108 donne un aperçu).

Voici, finalement, le lemme du pompage.

Théorème 2 *Soit L un langage hors-contexte sur l'alphabet Σ. Alors il existe un entier positif p tel que pour tout mot $w \in L(G)$ 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. $uv^ixy^iz \in L$ *pour tout* $i \in N$.

Démonstration. [Conseil: faites un dessin en lisant] Si $L = \emptyset$ ou $L = \{\varepsilon\}$, le théorème est vrai trivialement. On peut donc supposer que $\{\varepsilon\} \neq L \neq \emptyset$. Puisque L est hors-contexte, il existe une grammaire $G = (V, \Sigma, R, S)$ en forme normale de Chomsky telle que $L = L(G)$. Soit $w \in L$ un mot non-vide de longuer k. Soit T un arbre de dérivation de w dans G. Puisque la grammaire est en forme normale de Chomsky, l'arbre est binaire avec exactement k feuilles (aucune feuille n'est étiquetée par une variable ni par ε). Il est donc de profondeur $\ell \geq \lceil \lg k \rceil$, ce qui veut dire qu'un plus long chemin de la racine vers une feuille contient exactement ℓ sommets étiquetés par des variables (dans un chemin de longeur ℓ il y a $\ell + 1$ sommets, dont un seul est étiqueté par une lettre de Σ). Si maintenant $\ell > |V|$, il y aura (au moins) deux sommets étiquetés par la même variable. Soit donc $p > 2^{|V|}$. Alors quel que soit $w \in L$, si $|w| \geq p$, son arbre de dérivation dans la grammaire G sera de profondeur $\ell > \lg 2^{|V|} = |V|$. Soit w in tel mot, soit $T_G(S)$ son arbre de dérivation[1] avec racine r et soit C un plus long chemin de la racine vers une feuille. On peut noter cet arbre $T[r, S]$ car la racine est étiquetée par S. Disons que $C = rX_1 \ldots X_\ell$, où les X_i sont des sommets de

[1] Il peut ne pas être unique, il existe des grammaires *ambigues* dans lesquelles tout mot a au moins deux dérivation.

T^2.On a choisi w assez long pour que $\ell > |V|$ et par conséquent - piogonnier - il y aura deux sommets parmi les X_i étiqutés par la même variable. Soit $j \in \{1, 2, \ldots, \ell - 1\}$ le plus grand index tel qu'il existe un $i \in \{j+1, \ldots, \ell - 1\}$ avec $e(X_i) = e(X_j)$[3]. Soit cette étiquette la variable A. Regardons maintenant les sous-arbres $T[X_j, A]$ et $T[X_i, A]$. Le mot w est lu, dans l'ordre, sur les feuilles de $T = T[r, S]$, i.e. $w = w(T[r, S])$. Il peut être décomposé en $uvxyz$ un utilisant les mots $w(T[X_j, A])$ et $w(T[X_i, A])$ (encore une fois, faites un dessin!). Si on met $x = w(T[X_i, A])$, on peut mettre $w(T[X_j, A]) = vxy$ où v et y sont des parties qui restent de $w(T[X_i, A])$ en enlevant x. De même, $w = w(T[r, S]) = uw(T[X_j, A])z$ avec u et z le partie qui restent en enlevant $w(T[X_j, A])$. En mettant tout cela ensemble, on a bien que $w = uvxyz$.
Il nous reste a montrer que $|vxy| \leq p$, $|vy| > 0$ et $uv^ixy^iz \in L$ pour tout $i \in N$. Le premier découle du fait que la partie $X_jX_{j+1} \ldots X_\ell$ de C contient au plus $|V|$ variables comme étiquettes, par le choix de j (dont une répétée). La profondeur de $T[X_j, A]$ est donc au plus $|V|$ et la longeur de $w([T[X_j, A]) = vxy$ est au plus $2^{|V|} < p$. Puisqu'aucune règle ne donne ε, $vy \neq \varepsilon$. Finalement, à partir de $T = T_1$ on peut obtenir l'arbre T_0 en enlevant $T[X_j, A]$ et en le remplaçant par $T[X_i, A]$ de façon à obtenir $w(T_0) = uxz$ et, récursivement, T_i à partir de T_{i-1}, pour $i \geq 2$ en remplaçant $T[X_i, A]$ par une copie de $T[X_j, A]$ (ceci pourrait être fait plus précis au prix de cacher ce qui se passe). □

2 Annexe

Ici on reprend les notes de cours sur les langages hors-contexte originales.

Théorème 3 *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. $uv^ixy^iz \in L$ *pour tout* $i \in N$.

Démonstration. Si $L = \emptyset$ ou $L = \{\varepsilon\}$, le théorème est vrai trivialement. On peut donc supposer que $\{\varepsilon\} \neq L \neq \emptyset$. Puisque L est hors-contexte, il existe une grammaire $G = (V, \Sigma, R, S)$ telle que $L = L(G)$. Par théorème 6, 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

[2] Il faut bien distinguer entre les *sommets* de T et les *étiquettes.*

[3] C'est-à-dire, on prends comme X_j le premier sommet en remontant de la feuille de C vers la racine qui est étiqueté par la même variable qu'un X_i, $i > j$.

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 rgles 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 $1 \leq j \leq k, j \neq i$ 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 $T[Y, A]$ et le remplaçant par $T[X, A]$ 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). $\square$

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 4 *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 $L = L(G)$.

Théorème 5 *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. *$uv^ixy^iz \in L$ 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 $T[X, A]$. $\square$

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 $S \overset{*}{\Longrightarrow} w$ dans G si et seulement si $S \overset{*}{\Longrightarrow} w$ dans la nouvelle grammaire. Essayons de faire mieux: ce n'est pas les $A \longrightarrow \varepsilon$ qu'il faut remplacer, mais $A \overset{*}{\Longrightarrow} \varepsilon$.

Soit donc $G = (V, \Sigma, R, S)$. Soit $E = \{A \in V : A \overset{*}{\Longrightarrow} \varepsilon\}$ (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 $A \overset{*}{\Longrightarrow} \varepsilon$.
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 $A \in V$, $A \overset{*}{\Longrightarrow} w$ dans G si et seulement si $A \overset{*}{\Longrightarrow} w$ dans G'. Dans les deux directions, on procède par récurrence sur la longueur de la dérivation. On observe d'abord que pour aucune variable $A \in V$, $A \overset{*}{\Longrightarrow} \varepsilon$ dans G', car il n'y a pas de règles de la forme $B \longrightarrow \varepsilon$ dans R'.

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

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 1 *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 6 *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 2 *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 7 *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 6 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 6. 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 3 *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 $(q, u, \alpha) \overset{k}{\longmapsto} (p, v, \beta)$, 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 $(p, \sigma) \in \delta(q, a, A)$, $\sigma \in \Gamma^*$;
- $k > 1$, $(q, u, \alpha) \overset{k-1}{\longmapsto} (p', v, \beta')$ et $(p', v, \beta') \overset{1}{\longmapsto} (p, v, \beta)$.

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

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 $(q, u, \alpha) \overset{k}{\longmapsto} (p, v, \beta)$ est défini comme ci-haut pour $k = 1$, et si, pour $k > 1$, existe

- $q_0, \ldots, q_k \in Q$, $q_0 = q, q_k = p$;

- $A_1, \dots A_k \in \Gamma_\varepsilon$;
- $\alpha_1, \dots, \alpha_k \in \Gamma^*$;
- $a_1, \dots, a_k \in (\Sigma \cup \{\varepsilon\})$

tels que

- $\alpha = A_1\alpha_1$, $u = a_1 \dots a_k v$;
- pour $i = 1, \dots k-2$, $(q_i, a_{i+1}a_{i+2} \dots a_k v, A_{i+1}\alpha_{i+1}) \overset{1}{\longmapsto} (q_{i+1}, a_{i+2} \dots a_k v, A_{i+2}\alpha_{i+2})$;
- $(q_{k-1}, a_k v, A_k\alpha_k) \overset{1}{\longmapsto} (q_k, v, \beta)$.

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

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 $x \in V^*$, $S \stackrel{*}{\Longrightarrow} ux$ si et seulement si $(q, u, S\$) \overset{*}{\longmapsto} (q, \varepsilon, x\$)$. Il est - devrait être - clair que ceci implique que $S \stackrel{*}{\Longrightarrow} w$ si et seulement si $(q_0, w, \varepsilon) \overset{*}{\longmapsto} (f, \varepsilon, \varepsilon)$. 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 $S \stackrel{k}{\Longrightarrow} ux$ et prouvons que $(q, u, S\$) \overset{*}{\longmapsto} (q, \varepsilon, x\$)$. Pour $k = 1$, $S \stackrel{1}{\Longrightarrow} ux$ uniquement si soit $u = \varepsilon = x$, soit $u = \varepsilon$ et $x = BC$, $B, C \in V$. Mais alors $(q, \varepsilon, S\$) \overset{1}{\longmapsto} (q, \varepsilon, \$)$ dans le premier cas et $(q, \varepsilon, S\$) \overset{1}{\longmapsto} (q_B, \varepsilon, C\$) \overset{1}{\longmapsto} (q, \varepsilon, BC\$)$ dans le deuxième car δ comprend les transitions nécessaires. Pour $k > 1$, si $S \stackrel{k}{\Longrightarrow} ux$, alors soit $S \stackrel{k-1}{\Longrightarrow} uAy$, $x = BCy$, $A, B, C \in V$, $A \longrightarrow BC \in R$, soit $S \stackrel{k-1}{\Longrightarrow} vAx$, $u = va$, $A \in V, A \longrightarrow a \in R$. Dans le premier cas,$(q, u, S\$) \overset{*}{\longmapsto} (q, \varepsilon, Ay\$)$ par l'hypothèse de récurrence, et alors

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

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

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

ce qui prouve la première partie.

Dans l'autre sens, supposons que $(q, u, S\$) \overset{k}{\longmapsto} (q, \varepsilon, x\$)$. 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 $S \stackrel{*}{\Longrightarrow} \varepsilon$. Si $k > 1$, alors soit $u = va$ et $(q, u, S\$) \overset{k-1}{\longmapsto} (q, a, Ax\$) \overset{1}{\longmapsto} (q, \varepsilon, x\$)$, soit $x = BCy$ et $(q, u, S\$) \overset{k-2}{\longmapsto} (q, \varepsilon, Ay\$) \overset{1}{\longmapsto} (q_B, \varepsilon, Cy\$) \overset{1}{\longmapsto} (q, \varepsilon, x\$)$, et, dans le premier cas $S \stackrel{*}{\Longrightarrow} vAx \stackrel{1}{\Longrightarrow} vax = ux$, dans le deuxième $S \stackrel{*}{\Longrightarrow} uAy \stackrel{1}{\Longrightarrow} uBCy = ux$, par l'hypothèse de récurrence et les règles de la grammaire présentes. $\square$

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

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 $(q, u, \alpha) \overset{k-2}{\vdash} (q', au, \beta) \overset{1}{\vdash} (p, v, \beta$. $\square$

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, $(q, w, \varepsilon) \overset{k}{\longmapsto} (p, \varepsilon, \varepsilon)$ 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 $i = 0, \ldots, k$, $(q, w, \varepsilon) = (r_0, w, \alpha_0) \overset{i}{\longmapsto} (r_i, w_i, \alpha_i)$. 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 $(q, aub, \varepsilon) \overset{1}{\longmapsto} (r_1, ub, X) \overset{k-1}{\longmapsto} (r_{k-1}, b, X) \overset{1}{\longmapsto} (p, \varepsilon, \varepsilon)$. 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 $(p, w, \varepsilon) \overset{*}{\longmapsto} (q, \varepsilon, \varepsilon)$. 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 $(p, \varepsilon, \varepsilon) \overset{0}{\longmapsto} (p, \varepsilon, \varepsilon)$. 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 $(p, w1, \varepsilon) \overset{*}{\longmapsto} (r, \varepsilon, \varepsilon)$ et $(r, w_2, \varepsilon) \overset{*}{\longmapsto} (q, \varepsilon, \varepsilon)$. Donc $(p, w, \varepsilon) = (p, w_1w_2, \varepsilon) \overset{*}{\longmapsto} (r, w_2, \varepsilon) \overset{*}{\longmapsto} (q, \varepsilon, \varepsilon)$.

Dans le deuxième cas, on a $(r, u, \varepsilon) \overset{*}{\longmapsto} (s, \varepsilon, \varepsilon)$, 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 $(q, aub, \varepsilon) \overset{1}{\longmapsto} (r, ub, X) \overset{*}{\longmapsto} (s, b, X) \overset{1}{\longmapsto} (q, \varepsilon, \varepsilon)$.

Supposons maintenant que $(p, w, \varepsilon) \overset{k}{\longmapsto} (q, \varepsilon, \varepsilon)$. Si $k = 0$, la seule possibilité est que $(p, \varepsilon, \varepsilon) \overset{0}{\longmapsto} (p, \varepsilon, \varepsilon)$ 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 $(p, w, \varepsilon) \overset{1}{\longmapsto} (r_1, w_1, X) \overset{k-2}{\longmapsto} (r_{k-1}, w_{k-1}, X) \overset{1}{\longmapsto} (q, \varepsilon, \varepsilon)$. Soit $r = r_1,\ s = r_{k-1},\ w = aw_1 = aub,\ a, b \in \Sigma \cup \{\varepsilon\}, u \in \Sigma^*$. Donc $(r, ub, X) \overset{k-2}{\longmapsto} (s, b, X)$, i.e. $(r, u, \varepsilon) \overset{k-1}{\longmapsto} (s, b, \varepsilon)$ 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 $(p, w_1w_2, \varepsilon) \overset{l}{\longmapsto} (r, w_2, \varepsilon) \overset{l'}{\longmapsto} (q, \varepsilon, \varepsilon)$. 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. □