Math. Systems Theory 11, 147–155 (1977)

Mathematical
Systems Theory

Le cylindre des langages linéaires.

L. Boasson* and M. Nivat*

U.E.R. de Mathématiques, Université de Picardie, Amiens, France;

U.E.R. de Mathématiques, Université Paris 7, Paris, France

Abstract. Define a cylinder to be a family of languages which is closed under inverse homomorphisms and intersection with regular sets. A number of well-known families of languages are cylinders:

—CFL, the family of context-free languages, is a principal cylinder, i.e. the smallest cylinder containing a language L_0 described in [6].

—the family of deterministic context-free languages is proved to be a nonprincipal cylinder in [7].

—the family of unambiguous context-free languages is a cylinder: to prove that it is not principal seems to be a very hard problem.

In this paper we prove that Lin, the family of linear context-free languages, is a nonprincipal cylinder. This is achieved in the standard way by exhibiting a sequence of languages S_n, $n \in N$, such that Lin is the union of all the principal cylinders generated by these languages and is not the union of any finite number of these cylinders.

This leaves open the problem raised by Sheila Greibach of whether there exists a language L such that every linear context-free language is the image of L in some inverse gsm mapping.

S. Greibach [6] a récemment exhibé un langage algébrique L_0 dont tous les autres sont images dans un morphisme inverse, ce qui implique évidemment que

*Ce travail s'est fait dans le cadre du Laboratoire Associé du C.N.S.R.S. "Informatique Théorique et Programmation".

L_0 soit, dans la famille des langages algébriques, celui dont la reconnaissance exige un maximum de temps ou de mémoire. Au delà de ses implications dans le domaine de la complexité, qui peuvent être résolues différemment (ainsi, dans [9] le problème est-il résolu de ce point de vue pour les langages linéaires), ce résultat conduit naturellement à se poser la question de savoir si cette description "algébrique" est possible pour d'autres familles. En particulier, on peut espérer obtenir de cette façon des résultats concernant les familles de langages que la théorie des cones rationnels (ou des "full AFL's") ne peut appréhender (langages déterministes [7], langages non-ambigus). Nous nous proposons donc de définir ici la notion de cylindre comme étant une famille de langages fermée par morphisme inverse et intersection rationnelle. Cette définition étant posée, on peut énoncer ainsi le résultat de [6]: la famille Alg des langages algébriques est un cylindre principal (i.e. engendré par un seul élément). Le but principal de cet article est d'établir que la famille Lin des langages linéaires est un cylindre non principal. Par des arguments similaires, J. M. Autebert [1] démontre que la famille Oct des langages à un compteur est un cylindre également non-principal.

Outre les notions classiques de grammaire algébrique et de dérivations [4], nous utiliserons dans la suite les notions de:

—facteur itérant [2]: (α,u,β) est un facteur iterant de f dans L si $f=\alpha u\beta$ et $\alpha u^n\beta\in L\ \forall n\ (u\neq 1)$.

—paire itérante stricte [2]: $(\alpha,u,\beta,v,\gamma)$ est une paire iterante de f dans L si $f=\alpha u\beta v\gamma$, $\alpha u^n\beta v^n\gamma\in L$, $\forall n\geqslant 1$ $(uv\neq 1)$. Cette paire est stricte si $\alpha u^n\beta v^m\gamma\in L\Rightarrow n=m$.

Nous notons une grammaire algébrique G comme un triplet $\langle X,V,P\rangle$ où X est l'alphabet terminal, V l'alphabet non terminal et P l'ensemble de ses règles. Le langage engendré par G avec le non terminal A pour axiome est noté $L(G,A)$.

Le présent article est divisé en trois parties. La première vise essentiellement à présenter une construction concernant les grammaires linéaires permettant d'éliminer les règles unilatères; la deuxième présente une famille de langages linéaires constituant un système générateur de cylindre Lin, la dernière partie étant destinée à établir que ce système générateur engendre une hiérarchie croissante de cylindres.

I. Preliminaires

Dans toute la suite, S_n désigne le langage symétrique sur n lettres, c'est-à-dire le langage $L(G,S)$ engendré par la grammaire $G=\langle Z_n,\{S\},P\rangle$, $P=\{S\to x_iS\bar{x}_i;\ i=1,\ldots,n\}\cup\{S\to 1\}$ sur l'alphabet $Z_n=X_n\cup\bar{X}_n$ avec $X_n=\{x_1,\ldots,x_n\}$ et $\bar{X}_n=\{\bar{x}_1,\ldots,\bar{x}_n\}$ (1 désigne le mot vide).

Une grammaire linéaire $G=\langle X,V,P\rangle$ est dite

—*littérale* si toutes ses règles sont de la forme $v \to m$ avec

$$m \in XVX \cup XV \cup VX \cup X$$

—*pure* si toutes ses règles sont de la forme $v \to m$ avec

$$m \in X^+VX^+ \cup X^+ \quad \text{(i.e. si } G \text{ est sans règle linéaire unilatère)}$$

Un *langage linéaire* est *pur* s'il existe une grammaire linéaire pure qui l'engendre. Notons que tout *langage* linéaire *propre* (c'est-à-dire ne contenant pas le mot vide) peut être engendré par une grammaire linéaire littérale. Remarquons aussi que certains langages linéaires propres ne sont pas purs: ainsi $L = \{a^n b^n c^p \mid n,p \geqslant 1\}$ n'est pas pur ainsi que l'on peut le voir immédiatement en utilisant deux fois le lemme d'Ogden [8]. On peut cependant énoncer:

Lemme 1. *Etant donné un langage linéaire propre L, il existe un langage linéaire pur L_p et une substitution rationnelle propre σ tels que $L = \sigma(L_p)$.*

Avant d'établir ce lemme, notons qu'il permet de retrouver un résultat de [5]:

Corollaire. *Si L est générateur du cône rationnel des langages linéaires $L \cup \{1\}$ en est un générateur fidèle.*

Preuve du corollaire. Il est immédiat de montrer qu'à toute grammaire linéaire pure G, on peut associer un entier n et un langage rationnel K tels que $L(G,A)$ soit image de $S_n \cap K$ dans un homomorphisme strictement alphabétique (="length-preserving"). Ainsi, tout langage linéaire pur est-il élément du cône rationnel fidèle (="not full") engendré par S_2. Une substitution rationnelle propre étant une transduction rationnelle fidèle et continue (=une transduction n'utilisant que des homomorphismes qui n'effacent pas), il résulte du lemme 1 que tout langage linéaire propre est image de S_2 dans une telle transduction et comme le mot vide est dans S_2, il en va de même de tout langage linéaire. Or S_2 est sans facteur itérant; le corollaire se déduit alors d'un résultat général de [3].

Preuve du lemme 1. Le principe de la preuve consiste à construire une grammaire linéaire pure G_p et une substitution rationnelle σ à partir d'une grammaire linéaire littérale de telle sorte que $L(G,A) = \sigma(L(G_p,A))$. Plutot que de procéder directement (ce qui est possible), nous allons obtenir ce résultat en deux temps: nous construirons d'abord une grammaire $\bar{G}$ déduite de G qui est sans règle unilatère gauche (i.e. sans règle de la forme $v \to m$ avec $m \in VX^+$). Puis, appliquant à $\bar{G}$ une construction analogue, nous obtenons la grammaire G_p cherchée. Nous procédons ainsi parce que nous utiliserons la grammaire $\bar{G}$ dans la seconde partie de cet article.

Soit $G = \langle X, V, P \rangle$ une grammaire linéaire littérale engengrant le langage propre L avec $A \in V$ comme axiome. Les règles unilatères gauches de G et ses règles terminales (i.e. de la forme $v \to m$, $m \in X$) constituent une grammaire

$G_g = \langle X, V, P_g \rangle$ (où donc P_g est un sous ensemble de P). A chaque couple de variables $(v,v') \in V \times V$, on associe le langage $L(v,v') = \{f \in X^+ | v \xrightarrow{+} fv'$ dans $G_g\}$. Par ailleurs, on note $L(v)$ le langage $L(G_g, v)$. Comme G_g est une grammaire linéaire unilatère (littérale), tous les langages ainsi définis sont rationnels propres ou vides.

On construit alors la grammaire linéaire $\bar{G} = \langle \bar{X}, \bar{V}, \bar{P} \rangle$ sur l'alphabet terminal $\bar{X} = X \cup \{\langle x,v,v' \rangle | x \in X \cup \{1\}, v,v' \in V\} \cup \{\langle v \rangle | v \in V\}$ (où donc les symboles $\langle x,v,v' \rangle$ et $\langle v \rangle$ sont de nouvelles lettres terminales). L'alphabet non terminal $\bar{V}$ est $V \cup \{\bar{A}\}$ où $\bar{A}$ est un nouveau symbole (en fait, une copie de l'axiome A de G). Les règles de $\bar{G}$ sont alors définies comme suit:

(1) A chaque non-terminal v de V, on associe la règle $v \to \langle v \rangle$ (qui est terminale puisque $\langle v \rangle \in \bar{X}$). En outre, on ajoute la règle $\bar{A} \to \langle A \rangle$.

(2) A chaque règle $w \to xvy$ de G où x est non vide, (i.e. à chaque règle de G qui n'est pas dans G_g), on associe les règles suivantes de $\bar{P}$:

- $w \to xvy$
- $w \to xv' \langle y,v,v' \rangle$ pour tout v' de V
- $\bar{A} \to xv' \langle y,A,v' \rangle$ pour tout v' de V

L'idée est simple; on supprime dans G les règles unilatères gauches et on remplace par de nouveaux symboles toute chaîne de dérivation utilisant une suite de telles règles; si ce morceau de dérivation se trouve utilisé en début de dérivation, on utilise les règles $\bar{A} \to xv' \langle y,A,v' \rangle$; si elles sont utilisées pour terminer la dérivation, ce sont les règles $v \to \langle v \rangle$ qui les simuleront; dans les autres cas, les règles $w \to xv' \langle y,v,v' \rangle$ rendront compte de la situation. Plus précisement, la grammaire $\bar{G}$ est construite de telle façon que le langage $L(\bar{G}, \bar{A})$ vérifie la propriété suivante: si dans $L(\bar{G}, \bar{A})$, on substitue à chaque lettre $\langle y,v,v' \rangle$ le langage $yL(v,v')$ et à chaque lettre $\langle v \rangle$ le langage $L(v)$, on retrouve le langage $L(G,A)$. La substitution ainsi définie est évidemment une substitution rationnelle propre puisque les langages $L(v,v')$ et $L(v)$ sont tous rationnels propres. En outre, la grammaire $\bar{G}$ est clairement linéaire littérale et sans règle unilatère gauche.

Il suffit de noter que cette construction ne crée aucune règle unilatère droite pour s'assurer que le même processus appliqué aux règles unilatères droites conduit à la grammaire G_p annoncée.

II. Les langages $\hat{S}_n$

Nous allons maintenant définir une famille de langages linéaires qui ne sont rien d'autre que des "langages symétriques non déterministes". Ceux-ci sont légérement différents de ceux définis dans [9].

L'intérêt majeur en est qu'ils constituent une famille génératrice de cylindre des langages linéaires. Le but de cette deuxième partie est donc de définir d'abord ces langages et d'établir ensuite leur caractère de famille génératrice.

Rappelons d'abord que le langage symétrique S_n est défini sur l'alphabet $Z_n = X_n \cup \bar{X}_n$ avec $X_n = \{x_1, x_2, \ldots, x_n\}$ et $\bar{X}_n = \{\bar{x}_1, \ldots, \bar{x}_n\}$ comme les mots du langage de Dyck $D_n'^*$ n'ayant qu'un pic, soit $S_n = X_n^* \bar{X}_n^* \cap D_n'^*$. Sur le même alphabet Z_n, on définit le langage rationnel $R_n = \{x_i \bar{x}_i | 1 \leqslant i \leqslant n\}^+$ et l'on pose $S_n'' = \theta_n(S_n)$ où θ_n est la substitution rationnelle propre qui à chaque occurence de $x \in X_n$ (resp $\bar{x} \in \bar{X}_n$) substitue le langage xR_n (resp $\bar{x}R_n \cup \bar{x}$). Notons que S_n'' est donc encore un langage linéaire et que $S_n' = R_n$. S_n l'est aussi. Chaque mot du langage S_n' peut être représenté comme une montée suivie d'une descente, où, cette fois, la montée comme la descente comporte des paliers horizontaux constitués de dents de scie de hauteur 1.

Nous allons maintenant décrire un langage S_n' "non déterministe" de la même façon que l'on trouve un langage de Dyck "non déterministe" dans [6].

Pour ce faire, nous ajoutons à l'alphabet Z_n deux nouveaux symboles c et d et on définit le langage linéaire:

$$\hat{S}_n' = \{ du_1 c v_1 c w_1 d \cdots du_p c v_p c w_p | p \geqslant 0,\ u_i, w_i \in (Z_n \cup \{c\})^*,\ v_i \in Z_n^*,$$

$$i = 1, \ldots, p \text{ et } v_1 v_2 \cdots v_p \in S_n' \}$$

Définissant le langage rationnel K_n par

$$K_n = d\left(\{d,c\} \bar{X}_n \cdot (X_n^2 \cup X_n \cup \{1\})\right)^*$$

on pose

$$\hat{S}_n = \{ f | f \in K_n \text{ et } dx_1 f \in \hat{S}_n' \} \cup \{1\}.$$

Le langage $\hat{S}_n$ est donc encore linéaire; on peut se le représenter comme un langage S_n'' non déterministe où chaque élément figurant entre deux c est constitué d'un "motif" pris parmi trois possibles: une descente d'une unité, cette descente suivie d'une montée d'une unité (réalisant un "palier") ou cette descente suivie d'une montée de deux unités (réalisant une montée d'une unité). L'intérêt de ces langages provient du

Lemme 2. *Le langage L sur l'alphabet X est linéaire si et seulement si il existe un entier n et un homomorphisme h de X^* dans $(Z_n \cup \{c,d\})^*$ tel que $L = h^{-1}\hat{S}_n$.*

Preuve. Nous allons procéder d'une façon très similaire à celle utilisée dans [6] par S. Greibach, la seule difficulté consistant à traduire le caractère linéaire d'une grammaire lors de sa mise sous forme normale de Greibach.

Commençons par remarquer qu'il nous suffit d'établir le lemme pour un langage linéaire propre puisque nous avons adjoint le mot vide à $\hat{S}_n$.

Considérons donc une grammaire linéaire littérale $G = \langle X, V, P \rangle$ engendrant le langage linéaire propre L. Nous construisons d'abord la grammaire $\bar{G} = \langle \bar{X}, \bar{V}, \bar{P} \rangle$ obtenue dans la preuve du lemme 1. Mais au lieu de considérer les symboles $\langle y, v, v' \rangle$ et $\langle v \rangle$ comme des lettres terminales, nous les considérons comme des axiomes de grammaires linéaires unilatères droites engendrant les

langages rationnels $yL(v,v')$ et $L(v)$ respectivement. Nous obtenons ainsi une grammaire $\hat{G}=\langle X,\hat{V},\hat{P}\rangle$ sous forme normale de Greibach dont les non-terminaux peuvent être partitionnés en deux classes (c'est ce qui traduit la linéarité de la grammaire G de départ): ceux qui ne sont membres gauches que de règles unilatères droites V_d et ceux qui ne sont membres gauches d'aucune règle de cette sorte qui sont ceux de V.

En outre, toute règle non terminale est de la forme:

si $v\in V$, $v\to xww'$ avec $w'\in V_d$ et $w\in V$

si $v\in V_d$, $v\to xw$ avec $w\in V_d$

Considérant alors l'ensemble fini:

$$H_x=\{\bar{v}w'w|(v\to xww')\in\hat{P}\}\cup\{\bar{v}w|(v\to xw)\in\hat{P}\}\cup\{\bar{v}|(v\to x)\in\hat{P}\}$$

on pose: $h(x)=du_1cu_2c\cdots cu_p$ où $H_x=\{u_1,u_2,\ldots,u_p\}$.

Il est alors immédiat de prouver que $L=h^{-1}(\hat{S}_n)$ où n est le nombre de non-terminaux de $\hat{G}$ et où le symbole x_1 de Z_n représente l'axiome de $\hat{G}$. (La preuve est identique à celle de [6]). La réciproque est elle immédiate puisque $\hat{S}_n$ est linéaire [4].

III. Le cylindre lin. n'est pas principal

Du lemme 2, il résulte que si Lin est un cylindre principal il existe un entier n tel que $\hat{S}_n$ en soit générateur; en effet, si A est un générateur du cylindre Lin, on sait associer à A un entier n et un homomorphisme h tel que $A=h^{-1}(\hat{S}_n)$ puisque A est linéaire. Il en résulte que le cylindre engendré par $\hat{S}_n$ contient celui engendré par A et donc est Lin tout entier. Il nous suffit donc d'établir que les langages $\hat{S}_n$ constituent une hiérarchie croissante; intuitivement tel est le cas car on ne peut pas coder $\hat{S}_{n+p}$ dans $\hat{S}_n$ sans perdre la linéarité: il faut remplacer les "paliers horizontaux" réalisés par R_n par des dents de scie de plus en plus hautes, ce qui, en fait, conduit à remplacer les langages rationnels R_n par des langages linéaires S_n dans la définition de $\hat{S}_n$!!

Nous n'allons pas établir directement que les langages $\hat{S}_n$ forment une hiérarchie croissante. Mais nous allons montrer que, quelque soit n, il existe un langage linéaire au moins qui n'est pas dans le cylindre engendré par $\hat{S}_n$. L'idée directrice est ici que l'entier n croit avec le nombre de règles unilatères gauches des grammaires considérées car ce sont elles qui obligent à créer de nouveaux non-terminaux dans la mise sous forme normale de Greibach réalisée dans le lemme 2. Nous allons donc construire des langages linéaires ayant nécessairement un grand nombre de telles règles. En outre, pour nous simplifier la preuve, nous allons tenter aussi de nous placer dans une situation telle que les seules opérations de cylindre à considérer soient les homomorphismes inverses (et donc à ne pas avoir à faire intervenir d'intersection rationnelle, ou du moins à la réduire au strict minimum).

Sur l'alphabet $X_p \cup \{z,t\}$, nous considerons les langages suivants:

—les rationnels $R = x_1^+ x_2^+ \cdots x_p^+$ et $\bar{R} = X_p^* \setminus R$
—les langages linéaires

$$A_p = \{z^n x_1^{n_1} \cdots x_p^{n_p} t x_p^{n_p} \cdots x_1^{n_1} z^n | n \geqslant 1,\ n_i \geqslant 1,\ i = 1,\ldots,p\}$$

$$A'_p = \{z^n x_1^{n_1} \cdots x_p^{n_p} t x_p^{n_p} \cdots x_1^{n_1} z^m | n,m \geqslant 1,\ n_i \geqslant 1,\ i = 1,\ldots,p\ n \neq m\}$$

$$L'_p = A_p R \qquad \text{et} \qquad L''_p = A'_p \bar{R}$$

$$L_p = L'_p \cup L''_p$$

Le langage L_p est donc constitué de l'union disjointe de deux langages linéaires, celui qui va nous intéresser vraiment étant L'_p, le second L''_p étant là pour nous assurer du

Lemme 3. *Quelque soit le langage S, s'il existe un homomorphisme h et un langage rationnel K tels que $L_p = h^{-1}S \cap K$, on peut supposer que $K = z^+ Rt\tilde{R}z^+ X_p^*$*

($\tilde{R}$ désigne l'image miroire de R).

Preuve. Supposons donc que $L_p = h^{-1}S \cap K$. On établit facilement en utilisant le lemme de l'étoile qu'il existe un entier r tel que

$$(z^r)^+ (x_1^r)^+ \cdots (x_p^r)^+ t (x_p^r)^+ \cdots (x_1^r)^+ (z^r)^+ X_p^* \subset R.$$

Il s'agit du même argument que celui permettant de montrer que si un langage rationnel contient $\{a^n b^n | n \geqslant 1\}$ il existe un entier r tel qu'il contienne aussi $(a^r)^+ (b^r)^+$.

Si l'on pose alors $g(t) = t$ et $g(x) = x^r$ pour tout $x \in X_p \cup \{z\}$, on a $L_p = g^{-1}(h^{-1}(S)) \cap z^+ Rt\tilde{R}z^+ X_p^*$.

Lemme 4. *Pour $p > n^2$, il n'existe pas d'homomorphisme h et de langage rationnel K tels que $L_p = h^{-1}(\hat{S}_n) \cap K$.*

Preuve. En vertu du lemme 3, on peut supposer que, si tel était le cas

$$K = z^+ x_1^+ \cdots x_p^+ t x_p^+ \cdots x_1^+ z^+ X_p^*$$

L'idée intuitive est alors la suivante: chaque x_i a une image par h qui *doit* apporter une contribution non vide à la suite des choix réalisés dans $\hat{S}_n$ puisque chacun est impliqué dans une égalité (celle assurée par A_p). Il en résulte que lorsque ce même x_i apparait à droite du dernier z, il en va de même, mais qu'alors cette contribution se réduit à un "palier". Or, on a au plus n^2 "paliers" différent dans $\hat{S}_n$. Ainsi deux occurences de x_i et x_j contribueront de la même façon à la formation d'un mot de $\hat{S}_n$ et pourront s'échanger ce qui nous fera sortir du langage L_p.

Plus précisement, considérons les mots f_r de L_p définis par:

$$f_r = z^r x_1^r \cdots x_p^r t x_p^r \cdots x_1^r z^r x_1^r \cdots x_p^r \in L_p \ \forall r \geqslant 1$$

Nous allons établir:

(1) $\forall i \leqslant p | h(x_i)|_d \neq 0$ (où $|f|_d$ est le nombre d'occurences de d dans f). Supposons d'abord que $h(x_i) = 1$, alors si dans f_r on remplace la première occurence du facteur x_i^r par x_i^{2r}, on obtient un mot f_r' qui n'est pas dans L_p et qui pourtant vérifie $f_r' \in K$ et $h(f_r') = h(f_r)$; ce qui est impossible. Ainsi $h(x_i)$ n'est pas vide. Supposons alors qu'il ne contienne aucune occurence de d; il en contient au moins une de la lettre c puisque dans $\hat{S}_n$ la longueur des facteurs dans Z_n^* est bornée par 3 et que pour tout entier r, $h(x_i^r)$ est facteur d'un mot de $\hat{S}_n$.

On peut donc écrire $h(x_i) = u_i c v_i$. Si l'on considère alors $h(x_i^2) = u_i c v_i u_i c v_i$, on trouve un facteur d'un mot de $\hat{S}_n$ ne contenant aucune occurence de la lettre d. Ayant trois facteurs figurant entre deux occurences explicites de la lettre c, on sait que *l'un au plus* de ces facteurs contribue effectivement à la formation d'un mot de S_n' à partir du mot considéré dans $\hat{S}_n$. Ainsi si $h(f x_i^2 g) \in \hat{S}_n$ on a aussi $h(f x_i^{2+k} g) \in \hat{S}_n$ pour tout k. On peut donc, à nouveau dans les mots f_r, remplacer la première occurence de x_i^2 par x_i^{2+r}, ce qui devrait nous donner un nouveau mot de L_p, ce qui n'est pas et (1) est établi.

On peut donc supposer que $h(x_i) = u_i d v_i$ et si l'on considère à nouveau $h(x_i)^3 = u_i d v_i u_i d v_i u_i d v_i$, on voit apparaître deux facteurs $v_i u_i$ effectivement encadrés par des occurences de la lettre d et l'on sait donc que *chacun* doit apporter une contribution au mot de S_n' construit à partir du mot $h(f_r) \in \hat{S}_n$.

Considérant maintenant le fait que $(1, z^r, f_r', z^r, f_r'')$ constitue une paire itérante stricte de f_r dans L_p, on sait [2] que $(1, h(z^r), h(f_r'), h(z^r), h(f_r''))$, en constitue une de $h(f_r)$ dans $\hat{S}_n$. Il en résulte que tout facteur itérant contenant une occurence de d dans $h(f_r'')$ provient d'une suite de choix de facteurs de $h(f_r)$ formant un élément de $\bar{X}_n R_n X_n$ (puisque ce facteur itérant est dans la "descente" du mot associé de S_n').

Or chaque x_i^r constitue bien un tel facteur iterant; et donc chaque $h(x_i^r)$ apporte une contribution à la formation d'un mot de S_n' que l'on peut décrire comme $\bar{S}_i w_i T_i$ avec $w_i \in \{X_j \bar{X}_j | j = 1, \ldots, n\}^*$ et S_i, $T_i \in X_n$. Comme nous supposons $p > n^2$, on peut trouver deux valeurs i et j distinctes telles que $(S_i, T_i) = (S_j, T_j)$. Il en résulte que l'on peut, dans $\hat{S}_n$, échanger les occurences des facteurs correspondants et donc dans $h^{-1}\hat{S}_n$ échanger des occurences de x_i et de x_j. On obtient alors un mot:

$$g = z^r f_r' z^r \hat{f}_r'' \in h^{-1}\hat{S}_n \cap K \qquad \text{avec} \qquad \hat{f}_r'' \in \bar{R}.$$

Or $g \notin L_p$ et donc il est impossible que le langage L_p soit un élément du cylindre engendré par $\hat{S}_n$ (si $p > n^2$).

On en déduit alors le

Théorème. *Le cylindre des langages linéaires n'est pas principal.*

Pour conclure, il convient de remarquer que le problème reste ouvert d'établir qu'il n'existe pas de langage linéaire A dont tout langage linéaire soit

image dans un "gsm mapping" inverse. En effet, la preuve proposée ici ne s'étend apparemment pas aux "gsm". Cependant, le présent article réduit le problème à établir que les langages $\hat{S}_n$ forment une hiérarchie croissante vis à vis de cette opération.

References

1. J. M. AUTEBERT, Le Cylindre des Langages à Compteur, *à paraitre*.
2. L. BOASSON, Langages algébriques, paires itérantes et transductions rationnelles, *T.C.S.* **2** (1976), 209–223.
3. L. BOASSON ET M. NIVAT, Sur diverses familles de langages fermées par transductions rationnelles, *Acta Informatica* **2** (1973), 180–188.
4. S. GINSBURG, *The Mathematical Theory of Context-Free Languages*, McGraw-Hill, 1966.
5. S. GINSBURG, J. GOLDSTINE AND S. GREIBACH, Uniformly erasable AFL, *J.C.S.S.* **10** (1975), 165–182.
6. S. GREIBACH, The hardest context-free language, *SIAM J. Computing* **2** (1973), 304–310.
7. S. GREIBACH, Jump Pda's and hierarchies of deterministic context-free languages, *SIAM J. Computing* **3** (1974), 111–127.
8. W. OGDEN, A helpful result for proving inherent ambiguity, *Math. System Theory* **2** (1967), 191–194.
9. I. H. SUDBOROUGH, A note on tape-bounded complexity classes and linear context-free languages, *Journal of A.C.M.* **22** (1975), 499–500.

Received December 24, 1975, and in revised form: February 15, 1977 and June 15, 1977.