Langages artificiels et analyse syntaxique

Jean Colombaud

▶ **To cite this version:**

Jean Colombaud. Langages artificiels et analyse syntaxique. Interface homme-machine [cs.HC]. Université Joseph-Fourier - Grenoble I, 1964. Français. NNT : . tel-00164922

HAL Id: tel-00164922
https://theses.hal.science/tel-00164922

Submitted on 24 Jul 2007

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

N° d'ordre :

THÈSE

présentée à

LA FACULTÉ DES SCIENCES DE L'UNIVERSITÉ DE GRENOBLE

pour obtenir

LE TITRE DE DOCTEUR DE TROISIÈME CYCLE

Mathématiques Appliquées

---

par

Jean COLOMBAUD

Ingènieur I. R. G.

Licencié ès Sciences

LANGAGES ARTIFICIELS EN ANALYSE SYNTAXIQUE

Thèse soutenue le : 1961 *devant la Commission d'examen :*

Monsieur J. KUNTZMANN, Président

Monsieur B. VAUQUOIS, Rapporteur

Monsieur N. GASTINEL, Examinateur

T H E S E

présentée à

LA FACULTE DES SCIENCES DE L'UNIVERSITE .

pour obtenir

LE TITRE DE DOCTEUR DE TROISIEME CYCLE

Mathématiques Appliquées

par

Jean COLOMBAUD

Ingénieur I.R.G.

Licencié ès sciences

L A N G A G E S A R T I F I C I E L S

E N A N A L Y S E S Y N T A X I Q U E

Thèse soutenue le : devant la commission d'dxamen :

Monsieur J. KUNTZMANN, Président

Messieurs B. VAUQUOIS, Examinateurs

N. GASTINEL

Je tiens à exprimer ma profonde reconnaissance

à Monsieur le Profosseur KUNTZMANN, Directeur du Laboratoire do Calcul de l'Université de Grenoble qui a bien voulu me faire l'honneur de présider le Jury

à Monsieur le Professeur VAUQUOIS, sous la direction duquel ce travail a été effectué, pour ses conseils judicieux et son aide bienveillante

à Monsieur GASTINEL, Maître do Conférences, qui a bien voulu faire partie du Jury.

Je remercio égalemont les mombres du Contro d'Etudes pour la Traduction Automatique do Grenoblo, dont l'aido et los conseils me furent précieux.

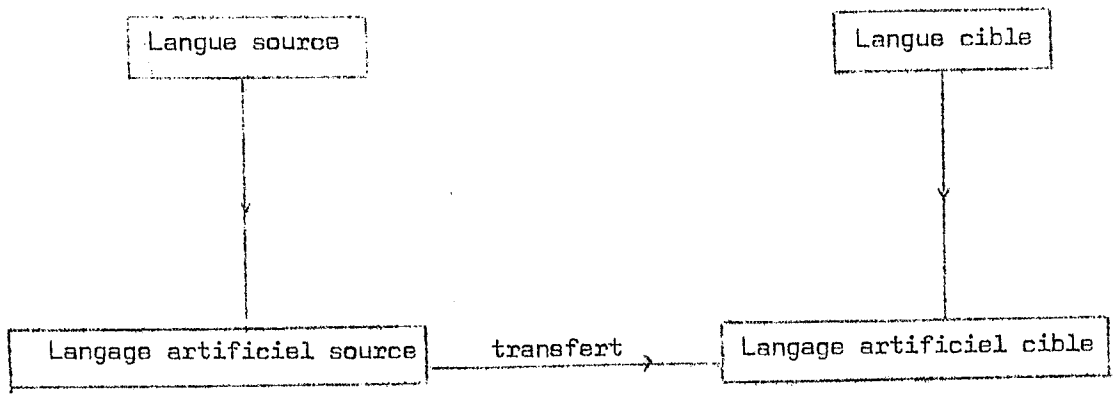
J'exprime toute ma gratitude à Madame PILON qui a assuré la réalisation matérielle de co document.

INTRODUCTION

Toutes recherches en "traduction automatique" doivent évidemment tenir compte des "machines" dont elles disposent. Or, les automates se contentent de reconnaître des caractères et des concaténations de caractères et opèrent donc sur des langages artificiels. Leur rôle se limitera à transférer l'écriture d'un texte d'un langage artificiel source dans un langage artificiel cible.

Les études linguistiques en vue de la traduction automatique ont défini 3 niveaux hiérarchisés : le plus haut, niveau sémantique, est celui de la pensée. Ensuite vient le niveau de l'expression qui est celui de la langue naturelle. Enfin, le niveau de la concaténation de caractères qui est celui des langages artificiels. Compte tenu des possibilités des machines, on étudie le niveau le plus bas, en essayant d'y inclure le plus possible, des éléments des deux niveaux supérieurs.

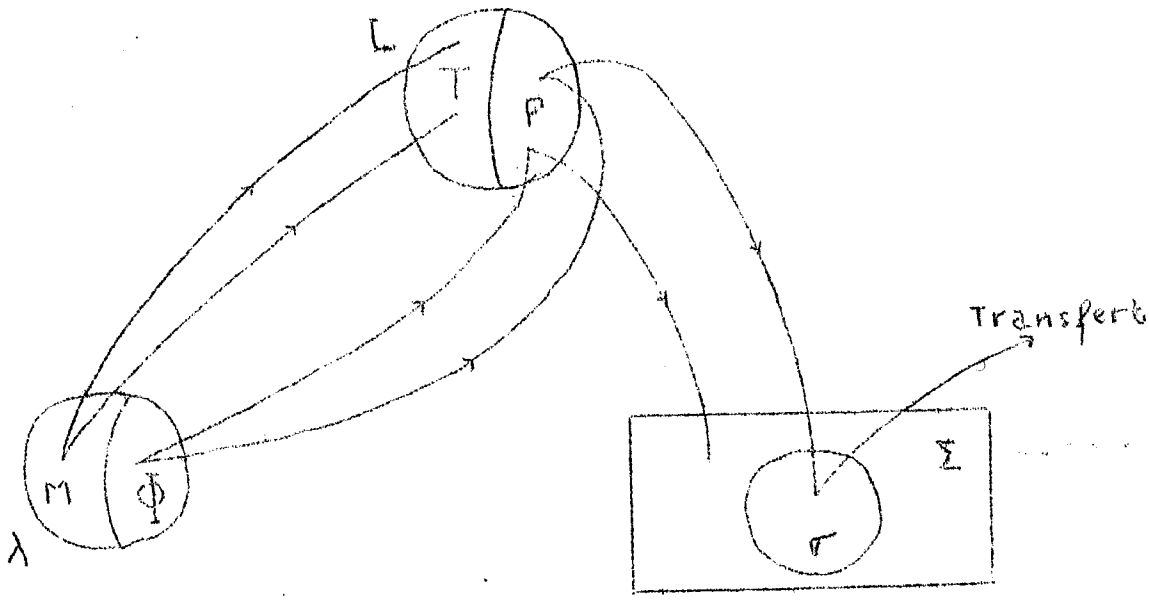
Aussi sommes nous conduits à fabriquer pour chaque langue naturelle un langage artificiel, modèle de cette langue. Le chemin de la traduction automatique sera alors le suivant :



Rappelons brièvement [2] qu'on a l'habitude de distinguer à l'intérieur d'un système formel deux sortes d'éléments : les termes et les formules. Si on établit une correspondance biunivoque entre les termes et une classe d'objets, on a une représentation du système. Si on établit une correspondance biunivoque entre les formules du système et une certaine classe d'énoncés dont les valeurs "vrai" ou "faux" sont déterminées en dehors du système et si, en outre, les formules valables du système et celles qu'on peut déduire correspondent à des énoncés vrais, on a alors défini une interprétation du système formel.

Essayons d'appliquer ces notions aux langages artificiels modèles de langues naturelles. Aux objets et aux énoncés précédents correspondront les mots et les phrases de la langue. Il faut remarquer qu'il y a déjà ici une difficulté puisque les correspondances ne sont plus toujours biunivoques. Au niveau de l'interprétation, on projettera les formules du langage sur un plan sémantique, et là encore les correspondances ne sont plus biunivoques. Seules, les formules se projetant sur la partie du plan sémantique, considérée comme valable, seront retenues pour le transfert.

On peut schématiser ces 2 opérations comme suit :



λ : Langue naturelle { M : ensemble des mots ; Φ : ensemble des phrases }

L : Langage artificiel modèle { T : ensemble des termes ; F : ensemble des formules }

Σ : Plan Sémantique

Σ^{-} : Partie valable du plan sémantique

La première opération qui consiste à fabriquer T au moyen de M est l'analyse morphologique [3] . Chaque terme issu de cette analyse est un syntagme élémentaire conservé dans un dictionnaire fini sous la forme suivante :

$\langle s.e. \rangle = \langle N^{o}\ d'U.L. \rangle \langle K.u.v. \rangle \langle C\Delta \rangle \langle vgp \rangle \langle Vgc \rangle \langle cg \rangle \langle cd \rangle \langle cs \rangle \langle c\Sigma^{-} \rangle$

s.e.	:	syntagme élémentaire
N° d'U.L.	:	numéro d'unité lexicale
Ku.v.	:	catégorie syntaxique
CΔ	:	code de dérivation
vgp	:	variables grammaticales permanentes
vgc	:	variables grammaticales contingentes
cg	:	code de gouvernement
cd	:	code de dépendance
c.s.	:	code syntaxique
c.Σ^{-}	:	code sémantique

Chaque catégorie syntaxique (Ku,v) apparaît en première approximation comme définissant une classe d'équivalence groupant tous les éléments ayant même comportement vis-à-vis de tous les autres éléments.

L'ensemble des s.e. va constituer la base des deux étapes suivantes, l'analyse syntaxique et l'analyse sémantique. Mais, si l'on a pu aisément étudier séparément la morphologie, on peut montrer [1] qu'il est impossible d'obtenir des résultats satisfaisants en dissociant la syntaxe de la sémantique.

Néanmoins, on se limitera dans cette étude à l'obtention de la construction syntaxique.

Il s'agira tout d'abord de choisir parmi les différents types logiques de modèles, celui qui convient le mieux à une description simple de la syntaxe des langues naturelles.

On montrera pourquoi on a été amené à prendre un modèle du type "context free" et quelles en sont les limites. En fait, le modèle est inutilisable tel quel, car il conduit à l'écriture d'un nombre de règles de construction trop élevé (de l'ordre de 25 000) pour qu'on puisse envisager d'écrire toutes ces règles sans erreur.

C'est pourquoi, dans un second chapitre, on cherchera à modifier ce modèle en construisant un nouveau langage capable de décrire les langages "context free", mais conduisant à un nombre de règles de beaucoup inférieur. L'ensemble de ces règles constituera la grammaire G_0 . On montrera ensuite sous quelles formes ce langage pourrait être exploité en machine.

Dans un dernier chapitre, on donnera les bases d'un langage d'indexage, λ , pouvant permettre ultérieurement de décrire les particularités lexicographiques d'une langue naturelle.

CHAPITRE I

POSSIBILITES ET CONTRAINTES D'UNE GRAMMAIRE

I_1.- DIFFERENTS TYPES DE GRAMMAIRES

On commencera par le rappel de quelques définitions [4]

Langage : Un vocabulaire V est un ensemble fini de symboles. Soit V_T, un vocabulaire contenant l'élément neutre pour la concaténation, dit vocabulaire terminal et soit V* l'ensemble des chaînes obtenues par concaténation sur V_T. Un langage L sera une partie de V* et sera dit construit sur V_T. La chaîne vide sera notée I.

Grammaire : Une grammaire d'un langage objet est un langage outil, fini, qui permet de construire ce langage objet, donc de générer les chaînes de ce langage. Une grammaire se présente comme un système formel avec un nombre fini d'axiomes et un nombre fini de règles de dérivation.

Notations : Les symboles de V_T seront notés a, b, c .... (premières lettres de l'alphabet en caractères minuscules). Les chaînes sur V_T seront notées x, y, z .... (dernières lettres de l'alphabet en caractères minuscules).

Grammaires chomskiennes [5]

On adjoint à V_T un nouvel ensemble fini de symboles, V_N, dit vocabulaire non terminal.

V_N contient un élément S (l'axiome) et un élément # (séparateur) et est tel que $V_T \cap V_N$ = l'ensemble vide. Soit $V = V_T \cup V_N$.

Les symboles de V_N seront notés A, B, C ... (premières lettres de l'alphabet en caractères majuscules).

Les chaînes sur V_N seront notées X, Y, Z ... (dernières lettres de l'alphabet en caractères majuscules).

Les chaînes ou symboles sur V seront notés par des lettres grecques.

On définit alors une relation entre 2 chaînes sur V, notée $\longrightarrow$, ($\varphi \longrightarrow \psi$ signifiant que la chaîne φ peut être réécrite comme ψ) ayant les propriétés suivantes :

- $\longrightarrow$ est irréflexive
- $A \in V_N$ si et seulement si, on peut trouver φ, ψ, ω telles que $\varphi A \psi \longrightarrow \varphi \omega \psi$, $\omega \neq I$
- On ne peut pas trouver φ, ψ, ω telles que $\varphi \longrightarrow \psi I \omega$ avec $\psi \neq I$ et $\omega \neq I$
- On peut trouver un ensemble fini de paires (χ_1, ω_1) ..... (χ_n, ω_n) tel que pour toutes φ, ψ, on puisse écrire $\varphi \longrightarrow \psi$ si et seulement si il existe φ_1, φ_2 et $j \leqslant n$ tels que $\varphi = \varphi_1 \chi_j \varphi_2$ et $\psi = \varphi_1 \omega_j \varphi_2$

  En d'autres termes, on peut imaginer une grammaire contenant un nombre fini de règles $\chi_j \longrightarrow \omega_j$, déterminant toutes les dérivations possibles.

Définitions : . Soit une grammaire G et la suite $(\varphi_1 \dots \varphi_n), n \geq 1$

φ est une φ - dérivation dans G si

$\varphi = \varphi_1 \quad \varphi = \varphi_n$ et $\varphi_i \longrightarrow \varphi_{i+1} \quad (1 \leq i \leq n)$

. φ sera une φ- dérivation terminale s'il n'existe aucune chaîne φ_{n+1} pour laquelle on aurait

$$\varphi_n \longrightarrow \varphi_{n+1}$$

. Etant donné V_T, V_N et G, le langage L (G) généré par G est l'ensemble de toutes les chaînes sur V_T qui sont des # S # dérivations terminales dans G.

Remarques : . Une grammaire G ne suffit pas à générer toutes les chaînes de L (G). Il faut lui adjoindre un algorithme qui exploite toutes les règles de G de toutes les manières possibles.

. A chaque chaîne de L (G) on peut associer le graphe montrant la dérivation qui a été effectuée.
En général, la structure associée n'est pas un arbre.

Restrictions : 1) Chaque chaîne de L (G) aura un arbre comme structure associée.
Tout couple (χ_j, ω_j) s'écrit alors :

$\chi_j = \xi A \eta$ et $\omega_j = \xi \omega \eta \qquad \omega \neq I$

ou $A \in V_N$ et ξ, η sont des chaînes sur V pouvant être vides.

La grammaire G sera alors formée de règles de réécriture du type $\xi A \eta \longrightarrow \xi \omega \eta$ et sera dite "context sensitive".

2) On impose à tout couple (χ_j , ω_j) la forme :

$\chi_j = A \qquad \omega_j = \omega \neq I$

La grammaire ainsi formée de règles du type $A \longrightarrow \omega$ est dite "context free".

3) A partir de la restriction précédente, on se limite aux règles $A \longrightarrow a$ ou $A \longrightarrow a\,B$. On retrouve alors les grammaires d'états finis.

I_2.- CHOIX D'UNE GRAMMAIRE POUR L'ANALYSE SYNTAXIQUE

Alors que l'analyse morphologique peut être réalisée au moyen d'un automate d'états finis dans la plupart des langues indo-européennes, l'analyse syntaxique doit faire appel à un système plus puissant.

Or, les langages "context free" ont l'avantage de pouvoir être analysés par des automates connus et semblent être suffisants pour servir de modèles aux phénomènes syntaxiques d'une langue naturelle. On peut objecter a priori que certaines constructions syntaxiques ne peuvent être décrites au moyen d'une telle grammaire. En fait, certaines constructions du type "context sensitive" peuvent être ramenées au type "context free" (cf $II_{1,2,4}$). Quant aux cas irréductibles, ils sont rares et peuvent être étudiés séparément.

On possède donc une grammaire formée de règles de la forme $A \longrightarrow \omega$ où $A \in V_N$ et ω est une chaîne quelconque sur V

Grammaire normale :

Une grammaire normale est une grammaire "context free" où ω ne peut prendre que les deux formes suivantes :

$$\omega = a \qquad\qquad \omega = A\ B$$

On démontre [5] qu'à toute grammaire G, "context free" on peut associer une grammaire normale G_N , telle que $L(G) = L(G_N)$.

Or, une telle grammaire est très intéressante pour l'étude des constructions syntaxiques : les règles $A \longrightarrow a$ font passer des syntagmes non terminaux aux syntagmes terminaux. L'ensemble de ces règles peut constituer une phase séparée de la phase principale qui ne comprend plus alors que des règles de la forme $A \longrightarrow BC$ et ne traite donc plus que des syntagmes non terminaux, selon une stratégie dichotomique.

Grammaire de reconnaissance :

Les grammaires dont on a parlé jusqu'ici étaient des opérateurs de génération. En traduction automatique, on veut reconnaître toutes les structures, permettant à partir des syntagmes élémentaires (éléments terminaux issus de la morphologie) de remonter à la phrase ($\neq$ S $\neq$) . On utilisera donc une grammaire normale, de reconnaissance, comprenant des règles de deux types :

$$BC \longrightarrow A \qquad \text{et} \qquad a \longrightarrow A$$

Notations : Dans la suite de cette étude on adoptera une notation différente. La règle élémentaire permettant de remonter au syntagme C à partir des syntagmes A et B, placés séquentiel-

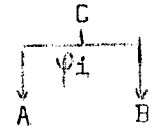
lement dans cet ordre s'écrira :

$$A \mid B \Longrightarrow C \qquad (\varphi i)$$

où φi est le symbole de la règle.

Si la grammaire comporte les deux règles $A \mid B \Longrightarrow C \quad (\varphi i)$ et $B \mid A \Longrightarrow C \quad (\varphi j)$, on pourra grouper ces deux règles sous la forme unique : $A * B \Longrightarrow C \quad (\varphi k)$.

La structure associée à la règle $A \mid B \Longrightarrow C \quad (\varphi i)$ pourra être représentée ainsi :



Les syntagmes tels que A et B seront appelés constituants de la règle et ceux tels que C, résultats de la règle. On pourra préciser que A est le premier constituant et B le second.

Ce qui est à gauche du signe $\Longrightarrow$ sera appelé partie gauche de la règle.

Une règle de la phase, faisant passer d'un syntagme terminal A à un syntagme non terminal B sera noté :

$$A \Longrightarrow B \qquad (\varphi i)$$

Cette phase, en reconnaissance, s'appellera : présyntaxe.

CHAPITRE II

LANGAGE D'ECRITURE DE LA GRAMMAIRE

L'objet de ce chapitre est de déterminer un métalangage qui décrit les langages "context free" au moyen d'un nombre de règles réduit par rapport à l'écriture de Chomsky. On se propose d'obtenir une grammaire G_0 ne contenant que quelques centaines de règles.

II_1.- VARIABLES GRAMMATICALES

On peut tout d'abord grouper dans un même ensemble tous les syntagmes appartenant à une même catégorie syntaxique * . Chaque ensemble prendra alors le nom de la catégorie qu'il représente. A chaque ensemble sera affectée la liste des variables grammaticales de la catégorie et l'on pourra différencier les syntagmes d'un même ensemble par les valeurs qu'ils donnent à ces variables.

La règle de grammaire $A|B \Longrightarrow C \quad (\forall i)$ signifie maintenant qu'un syntagme appartenant à la catégorie A peut être lié à un syntagme appartenant à la catégorie B pour donner un syntagme appartenant à la catégorie C.

A, B, C n'étant plus des syntagmes, on doit réécrire les règles en tenant compte des variables grammaticales, pour les rendre applicables aux syntagmes. A cette fin, on utilise un nouveau langage de description.

* On trouvera en annexe une liste de catégories terminales avec variables et valeurs de variables pour le Russe et l'Allemand.

II$_{1,1}$ - Langage de description des règles de grammaire

- Ω est un ensemble fini d'éléments V_1 , ... V_I ..., V_N appelés variables.
- Chaque élémont V_I est lui-même un ensemble fini d'élémonts, v_{I1} , ... v_{Ij} ... v_{IP_I} , appelés valeurs de la variable V_I .
- Il existe dans Ω un ordre de rangement des variables et dans chaque variable un ordre de rangement des valeurs.

Groupement de variables de Ω

- $E_\alpha(\Omega)$, est un quelconque ensemble de α variables de Ω, $\alpha \leq N$. Il existe C_N^α ensembles $E_\alpha(\Omega)$ et si l'on veut particulariser l'un d'entre eux, on le notera $E_\alpha^k(\Omega)$. $(1 \leq k \leq C_N^\alpha)$

Groupe sur $E_\alpha^k(\Omega)$

Un groupe sur $E_\alpha^k(\Omega)$ est un α - uplet, élément du produit cartésien des ensembles V_I appartenant à $E_\alpha(\Omega)$, ordonnés suivant le critère de rangement de Ω. L'élément séparateur est le blanc. (noté # pour la clarté des exemples).

Exemple : Ω : V_1 , V_2 ..... V_N

$E_3^k(\Omega)$: V_1 , V_3 , V_N

groupe sur $E_3^k(\Omega)$: $v_{1i} \# v_{3j} \# v_{NC}$

L'ensemble des groupes définis sur $E_\alpha^k(\Omega)$ est appelé : produit associé à $E_\alpha^k(\Omega)$.

Partie de $\overset{k}{E}_{\alpha}(\Omega)$

Une partie de $\overset{k}{E}_{\alpha}(\Omega)$ est un ensemble quelconque de variables V_I appartenant à $\overset{k}{E}_{\alpha}(\Omega)$. Par extension $\overset{k}{E}_{\alpha}(\Omega)$ est une partie de lui-même.

Suite sur $\overset{\ell}{E}_{\beta}(\Omega)$

Une suite sur $\overset{\ell}{E}_{\beta}(\Omega)$ est une concaténation de groupes définis chacun sur une quelconque partie de $\overset{\ell}{E}_{\beta}(\Omega)$, mais dont au moins un groupe est défini sur $\overset{\ell}{E}_{\beta}(\Omega)$ lui-même.

L'opérateur de concaténation est la virgule et a le sens du symbole logique "ou".

Les groupes d'une suite ne sont pas ordonnés, mais le premier groupe doit être l'un de ceux qui sont définis sur $\overset{\ell}{E}_{\beta}(\Omega)$ lui-même.

Une suite sur $\overset{\ell}{E}_{\beta}(\Omega)$ commence donc par un groupe de β éléments qui permet de déterminer à lui seul l'ensemble $\overset{\ell}{E}_{\beta}(\Omega)$ sur lequel la suite est définie.

Exemple de suite sur $\overset{k}{E}_{3}(\Omega)$:

$$V_{1i} \# V_{3j} \# V_{1\ell} \; , \quad V_{32} \# V_{Ni} \; , \quad V_{1j} \# V_{N\ell} \; , \quad V_{N1}$$

Compartiment sur Ω :

Un compartiment sur Ω est une concaténation de suites sur des ensembles $\overset{k}{E}_{\alpha}(\Omega)$ disjoints. L'opérateur de concaténation est la barre oblique et a le sens du symbole logique "et".

Indices

On utilise trois indices (1) (2) et (1.2).

Variable indicée :

Une variable indicée désigne une ou plusieurs valeurs de la variable qui est indicée. Elle se représente par le symbole de la variable suivi d'un indice. Lorsque l'indice est (1) [ou (2)] , la variable indicée désigne la ou les valeurs prises par le premier [ou le deuxième] constituant. Lorsque l'indice est (1.2), la variable indicée désigne la ou les valeurs communes aux deux constituants.

Exemple :

- Soit V_I , comprenant v_{I1} , v_{I2} , v_{I3} , v_{I4}
- Si le premier constituant prend les valeurs v_{I1} et v_{I3} , v_I (1) désigne ces deux valeurs.

Couple sur (Ω , Ω') :

Soient 2 ensembles de variables Ω et Ω' .
Un couple sur (Ω , Ω') est la concaténation de deux variables indicées, représentée comme suit :

V_I (1) # V_J (2) où $V_I \in \Omega$ et $V_J \in \Omega'$

Simplification :

Un couple de la forme V_I (1) # V_I (2) , $V_I \in \Omega$ et $V_I \in \Omega'$, se notera simplement V_I .

Identité sur (Ω, Ω')

Une identité sur (Ω, Ω') est une concaténation de couples sur (Ω, Ω'), l'opérateur de concaténation étant la barre oblique avec le sens du symbole logique "et".

$II_{1,2}$.- Ecriture de règles

Etant donné trois catégories syntaxiques A, B, C non nécessairement distinctes, la règle de grammaire la plus générale s'écrira :

A // VVA | B // VVB - NVI ⟹ C/VC* - RESV

VVA : valeurs de variables de A
VVB : valeurs de variables de B
NVI : noms de variables identiques
VC* : valeurs de variables de C
RESV : réserve

$II_{1,2,1}$.- VVA et VVB sont des compartiments définis respectivement sur A et B. (Ensembles des variables affectées aux catégories dont ils portent le nom (voir II_1))

VVA (respectivement VVB) donne les conditions auxquelles doivent satisfaire les valeurs de variables d'un syntagme appartenant à la catégorie A (respectivement B) pour qu'on puisse lui appliquer la règle.

Exemples : (# est remplacé par le blanc)

A // NOM SIN , ACC PLU / FEM

Cet exemple impose les conditions suivantes : être soit au Nominatif Singulier, soit à l'Accusatif Pluriel et être au Féminin.

A // NOM MAS SIN , ACC PLU / ANI / POS , COM

Dans ce deuxième exemple, la première suite est définie sur un groupement comprenant les variables CAS , GNR , NBR d'après la forme du premier groupe. Le groupe ACC PLU contient alors implicitement les 3 genres : il remplace ACC MAS PLU , ACC FEM PLU , ACC NEU PLU.

$II_{..1,2,2}$:- NVI

Pour que l'on puisse appliquer une règle à 2 syntagmes a et b, $a \in A$, $b \in B$, il faut que dans chaque couple, les deux variables indicées aient au moins une valeur commune.

Exemples : NVI : CAS (1) CAG (2) / NBR

(CAG désigne le cas gouverné)

Le cas de a et le CAG de b doivent avoir au moins une valeur commune. En outre, les deux syntagmes doivent avoir au moins une valeur de la variable NBR en commun.

$II_{..1,2,3}$:- VC*

C'est un compartiment sur C* - C* est un ensemble contenant les mêmes variables que C, chaque variable comprenant, outre ses valeurs dans C, les valeurs des variables A et B indicées. VC* donne les valeurs des variables du syntagme résultant.

Exemple : NOM SIN, CAS (1) PLU / GNR (1.2)

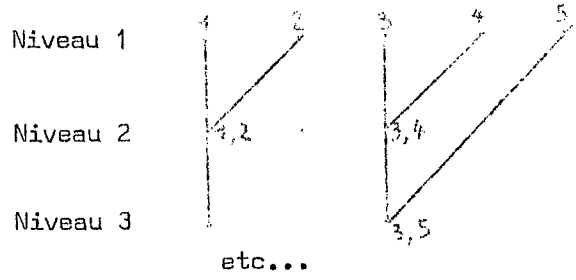
Le syntagme résultant est, soit au nominatif singulier, soit au pluriel avec les différents cas du promier constituant. Il prend en outre les genres communs aux deux constituants.

Remarque importante :

Les compartiments de partie gauche d'une règle peuvent comporter des vides. En effet, lorsqu'une variable n'y a pas de valeur précisée, cela signifie qu'on n'impose aucune restriction aux syntagmes constituants. Par contre, la partie droite, symbolisant la construction d'un syntagme, doit comprendre au moins une valeur de chaque variable affectée à la catégorie résultante. Si l'on veut indiquer que ce syntagme prend toutes les valeurs d'une variable, on doit toutes les noter.

II$_{1,2,4}$ - Réserve (RESV)

L'utilité de ce compartiment apparait lors du traitement de constructions du type "context sensitive" se ramenant au type "context free". Rappelons que les constructions d'une phrase peuvent être schématisées comme ci-dessous :
[7] et [8]

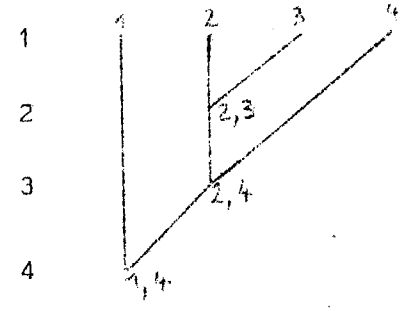


etc...

Au niveau 1, (Niveau des syntagmes élémentaires), chaque élément reçoit un numéro séquentiel. Aux niveaux supérieurs, chaque élément reçoit 2 numéros qui sont les numéros séquentiels des deux syntagmes élémentaires extrêmes qui ont servi à sa construction. On les note ν_d et ν_f . Dans le cas de "context free", deux syntagmes si et sj ne pourront être liés par une règle que si $\nu_{f_i} = \nu_{d_j} - 1$ et si en outre, la pos-

sibilité d'appliquer la règle, ainsi que la construction du syntagme s_k résultant, ne dépendent que des caractéristiques de si et sj. Soit maintenant, une construction où on a bien toujours la relation $\nu_{p_i} = \nu_{s_j} - 1$, mais où la possibilité d'application de la règle et la construction du syntagme s_k résultant, dépendent d'un autre syntagme s_e qui entre dans la construction de s_k à un niveau inférieur et dont on a perdu les caractéristiques au moment de son emploi dans la construction antérieure. Ces cas se traitent d'une manière élégante par des moyens "context sensitive"

Exemple



Si l'on suppose que la construction de $o_{1,4}$ dépend des caractéristiques de s_2 , perdues lors de la construction de $s_{2,3}$ cet exemple est bien un cas de "context sensitive".

On se ramène au "context free" de la manière suivante :

Au moyen d'une réserve (RESV) on pourra garder en mémoire certains renseignements du syntagme s_e , mis de côté au moment de son emploi. Avant la construction de s_k , l'un de ses composants (ou les deux) pourra être transformé, en tenant compte de la réserve, au moyen d'une règle de transformation du type :

$$A \mid RESV - n \Longrightarrow A$$

S_e aura été employé dans une règle φ_n dont la partie droite comprendra - RESV-n / NVR où RESV - n est une nouvelle catégorie et n le numéro de la règle.

NVR est un compartiment sur C* .

Pour s'assurer que s_e est bien constituant de s_k à un niveau inférieur, il faudra vérifier que l'intervalle $\nu_{d_k} - \nu_{f_k}$ contient l'intervalle $\nu_{d_e} - \nu_{f_e}$ c'est-à-dire que $\nu_{d_k} < \nu_{d_e}$ et $\nu_{f_k} > \nu_{f_e}$

Il faudra donc conserver également ν_{d_e} et ν_{f_e} en réserve.

Une même règle, munie d'un RESV peut être appliquée plusieurs fois successivement, les renseignements en RESV étant inscrits les uns à la suite des autres.

On peut montrer l'équivalence au langage "context free" . Soit φ_e la règle qui utilise s_e comme constituant et φ_k celle qui permet de construire s_k et soit $\{\varphi_{ek}\}$ l'ensemble des règles qui ont permis la construction des syntagmes intermédiaires. Il suffit, pour obtenir l'équivalence de créer des nouvelles variables, affectées aux catégories auxquelles appartiennent tous les syntagmes construits par φ_e et $\{\varphi_{ek}\}$

A ces nouvelles variables, on donnera les valeurs de s_e que l'on doit conserver. Les règles de $\{\varphi_{ek}\}$ devront imposer ces valeurs aux syntagmes résultats Enfin, on ne permettra la construction de s_k que si ces valeurs satisfont aux contraintes d'accord.

Une telle règle de transformation ne construit pas un nouveau syntagme, mais modifie seulement les caractéristiques d'un syntagme donné. Une fois cette règle employée, on efface la partie RESV qui a été utilisée. Cette précaution permet de construire plusieurs cas semblables en cascade.

Remarque :

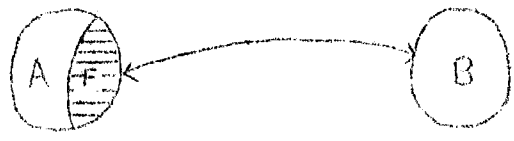
Le langage d'écriture ainsi défini s'applique aux règles de la présyntaxe, en omettant tout ce qui a trait au deuxième constituant.

II_2.- INVALIDATION

$II_{2,1}$.- Soit la règle $A \mid B \Longrightarrow C \qquad (\varphi i)$

(Dans ce paragraphe, les variables grammaticales n'entrant pas en jeu, on écrira les règles sous leur forme globale et on confondra les symboles des syntagmes et des catégories).

On peut trouver des syntagmes, appartenant à une partie F de A, ne pouvant se lier par φ i à aucun élément de B.



Il faut alors scinder A en deux sous catégories $A_1 = A - F$ et $A_2 = F$. La règle φ i devient ainsi $A_1 \mid B \Longrightarrow C$. Mais ce procédé oblige à doubler les r - 1 autres règles ayant A en partie gauche,

$$A \mid D \Longrightarrow E \quad (\varphi j) \quad \text{devenant} \quad \begin{cases} A_1 \mid D \Longrightarrow E & (\varphi_{j1}) \\ A_2 \mid D \Longrightarrow E & (\varphi_{j2}) \end{cases}$$

On évite ce dédoublement en écrivant des règles de la forme :

$$A \mid B \Longrightarrow C \;//\; \rho_c \quad (\varphi_i)$$

où ρ_c est une liste de symboles de règles appartenant à l'ensemble R_c des règles ayant C en partie gauche. Chaque symbole de règle peut être suivi d'un indice (u), u prenant les valeurs 1 ou 2.

Exemple : ρ_c : $\varphi_1\,(1)$, $\varphi_{15}\,(2)$, $\varphi_{27}\,(1)$, φ_{31} ,...

Cette écriture signifie que le syntagme résultant de φ_i ne peut être utilisé comme uième constituant d'une règle indicée (u), écrite dans ρ_c .

On dit alors qu'une telle règle est invalidée (u) pour le syntagme résultant.

Lorsqu'il n'y a pas d'ambiguïté sur la position d'un syntagme invalidant une règle, on omet l'indice (u) .

Exemple :

$A \mid B \Longrightarrow C \;//\; \varphi_3 , \varphi_{12}\,(1) , \varphi_{15}\,(2)$ $\quad (\varphi_i)$

$C \mid D \Longrightarrow E$ $\quad (\varphi_3)$

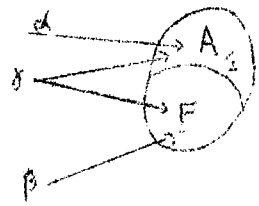
$C * F \Longrightarrow G$ $\quad (\varphi_{12})$

$C * C \Longrightarrow C$ $\quad (\varphi_{15})$

Soit Co un syntagme construit par φ_i . Toutes les constructions de la forme CoD, CoF, CCo sont impossibles. Par contre on peut faire les constructions de la forme FCo et CoC .

. Soit P_A , l'ensemble des règles permettant de construire les syntagmes appartenant à A (on dira que P_A est l'ensemble des règles conduisant à A) et R_A l'ensemble des r règles ayant A en partie gauche. Montrons l'amélioration que l'on peut attendre du nouveau procédé.

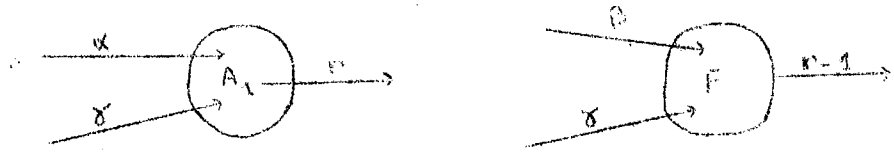
F est un sous ensemble de A dont les syntagmes invalident la règle $\varphi_k \in R_A$. P_A est formé de α règles conduisant à $A - F = A_1$, de β règles conduisant à F, et de γ règles conduisant à la fois à F et A_1.



$$\varphi = \alpha + \beta + \gamma$$

On supposera que l'on peut différencier pour chacune des γ règles, les syntagmes constituants qui entraînent une construction conduisant à A_1 de ceux entraînant une construction conduisant à F (point de vue lexicographique).

Si l'on sépare A en deux catégories A_1 et F, on devra écrire $\alpha + \gamma$ règles conduisant à A_1, $\beta + \gamma$ règles conduisant à F, r règles ayant A_1 en partie gauche et r - 1 règles ayant F en partie gauche, soit : $\varphi + \gamma + 2r - 1$ règles.

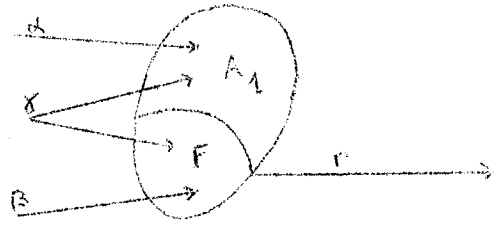


Si l'on utilise l'invalidation, on écrira pour conduire à $A = A_1 + F$:

α	règles du type	$B \mid C \Longrightarrow A$
β	" "	$D \mid E \Longrightarrow A \;//\; \varphi_k$
γ	doublets du type	$F \mid G \Longrightarrow A$
		$F \mid G \Longrightarrow A \;//\; \varphi_k$

soit $\varphi + \delta$ règles comme précédemment. Ce résultat était évident et on ne raisonnera dans la suite que sur les règles ayant A ou F en partie gauche.

En revanche, on écrira seulement r règles ayant A en partie gauche, et on aura donc réalisé un gain de r - 1 règles.



Remarque :

Le décompte précédent suppose que R_A contient des règles ne possédant A qu'une seule fois en partie gauche. Dans le cas contraire, l'amélioration est encore plus nette car de telles règles doivent être comptées deux fois dans le processus sans invalidation.

Si les éléments de F invalident m règles parmi les r de R_A, on montre que le gain est de r - m règles.

Soient maintenant deux sous-ensembles F et G de A, disjoints. Les éléments de F invalident une règle φ_k de R_A et ceux de G une autre règle φ_ℓ - $k \neq \ell$

Si l'on scinde A en 3 catégories $A_1 = A - (F + G)$, F, G on devra écrire r règles ayant A en partie gauche, r - 1 ayant F et r - 1 ayant G en partie gauche, soient 3r - 2 règles.

Au moyen de l'invalidation, on écrira seulement r règles, ce qui permet de gagner 2r - 2 règles.

- Si les éléments de F invalident m règles ($m < r$) et ceux de G, n règles ($n < r$), on a alors un gain de 2r - (m + n) règles.
- Si enfin F et G ont une partie commune, on trouve avec les mêmes hypothèses que dans les 2 cas précédents, des gains de 3r - 4 et 3r - 2 (m + n), respectivement.

On remarquera que les éléments de la partie commune ne pouvant invalider plus de r règles, on a nécessairement $m + n \leqslant r$.

II$_{2,2}$.- Le procédé d'invalidation qui vient d'être défini, peut être utilisé dans deux autres cas.

II$_{2,2,1}$.- Règles récursives

Le syntagme, résultat d'une construction, peut être dans certains cas, analogue à l'un des constituants. Avec certaines réserves, on pourra confondre les 2 catégories de ces syntagmes.

On sera donc amené à écrire certaines règles récursives du type $A \mid B \Longrightarrow A \quad (\varphi_i)$.

Or, écrite telle quelle, cette règle permet la construction de syntagmes de la forme ABBB ...

Dans certains cas, on veut se limiter à la forme AB .
(En français, l'article défini, suivi d'un substantif, est un élémont qui se comporte comme un substantif vis-à-vis des éléments des autres catégories, mais qu'on ne peut faire précéder d'un article).

On écrira alors la règle :

$$A \mid B \Longrightarrow A \;//\; \varphi_i \qquad (\varphi_i)$$

qui sera dite saturée pour le syntagme en question.

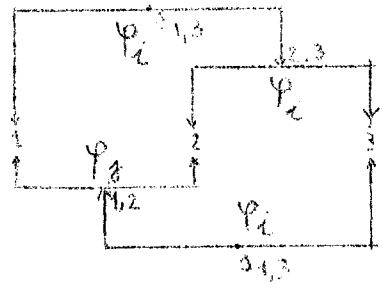
Exemple : ARTICLE | SUBSTANTIF $\Longrightarrow$ SUBSTANTIF // φ_i (φ_i)

II$_{2,2,2}$ - Constructions parasites

La technique d'invalidation peut aussi être utilisée afin d'éviter certaines constructions intermédiaires parasites. Soit en français, la suite de syntagmes élémentaires :

Jolie	Petite	Maison
1	2	3

Le syntagme non terminal s_{13} qui représente l'expression peut être obtenu des 2 façons suivantes :



D'autre part, la grammaire G_o comprend nécessairement les 2 règles :

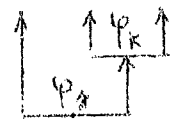
φ_i : ADJQ * SUBC - GNR/NBR $\Longrightarrow$ SUBC// GNR (1,2) / NBR (1,2)

φ_j : ADJQ * ADJQ - GNR/NBR $\Longrightarrow$ ADJQ// GNR (1,2) / NBR (1,2)

La règle φ_j donnant la construction de suite d'adjectifs coordonnés en liaison avec la règle φ_k :

COCO | ADJQ $\Longrightarrow$ ADJQ //GNR (2) / NBR (2) (φ_k)

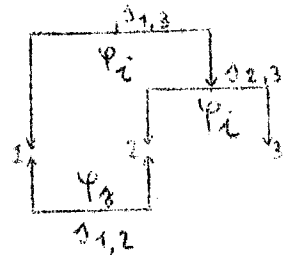
Exemple : Elle est brune et jolie



On ne peut donc éviter de construire à la fois s_{12} et s_{23} .
Mais, on peut s'abstenir de construire s_{13} à partir de s_{12} .
Il suffit d'écrire φ_j sous la forme :

φ_j : ADJQ * ADJQ - GNR/NBR $\Longrightarrow$ ADJQ// GNR(1.2)/NBR(1.2)//φ_i

Ainsi, s_{12} formé par φ_j ne peut être constituant de φ_i .
On ne garde donc que les constructions suivantes :



II_3.- EXEMPLES D'ECRITURE DE REGLES

$II_{3,1}$.- Accord adjectif-substantif en français

Règle SUBG | ADJQ - GNR/NBR $\Longrightarrow$ SUBG/GNR (1.2)/NBR (1.2)

Maison	jaune	Maison	Jaune
SUBG/FEM/SIN	ADJQ/FEM/SIN	SUBG/FEM/SIN	

$II_{3,2}$.- Complément de nom en allemand

Règle	SUBG	SUBG//GEN ⟹	SUBG/CAS(1)/GNR(1)NBR(1)
	Das Buch	des Lehrers	Das Buch des Lehrers
	SUBG/NOM,ACC/NEU/SIN	SUBG/GEN/MAS/SIN	SUBG/NOM,ACC/NEU/SIN

$II_{3,3}$.- Accord cardinal-substantif en russe

Règle	CARD//NOM,ACI	SUBG//GEN/GNR ⟹	SUBG//CAS(1)/NEU SIN,PLU, GNR(2) NBR(2)
Exemple 1	ДВЕ	НЕДЕЛИ	ДВЕ НЕДЕЛИ
	deux	semaines	
	CARD/NOM/FEM	SUBG/GEN/FEM/SIN/INA	SUBG/NOM/NEU SIN,PLU,FEM SIN
Exemple 2	ПЯТЬ	АТЛАСОВ	ПЯТЬ АТЛАСОВ
	cinq	atlas	
	CARD/ACI/MAS,FEM,NEU	SUBG/GEN/MAS/PLU/INA	SUBG/ACI/NEU SIN,PLU,MAS PLU

Le syntagme résultant s'écrira finalement SUBG/ACI/NEU SIN, PLU car MAS PLU est contenu dans PLU, puisqu'on ne différencie pas en russe les genres au pluriel.

A partir du premier exemple on peut trouver :

1)	ДВЕ НЕДЕЛИ	ПРОШЛИ	VERB/PLU/PAS
	deux semaines	ont passé	
2)	"	ПРОШЛО	VERB/NEU/SIN/PAS
		a passé	

Dans 1), c'est la solution SUBG/NOM/PLU du syntagme résultant qui permettra l'accord SUBG | VERB

Dans 2) ce sera SUBG/NOM/NEU SIN

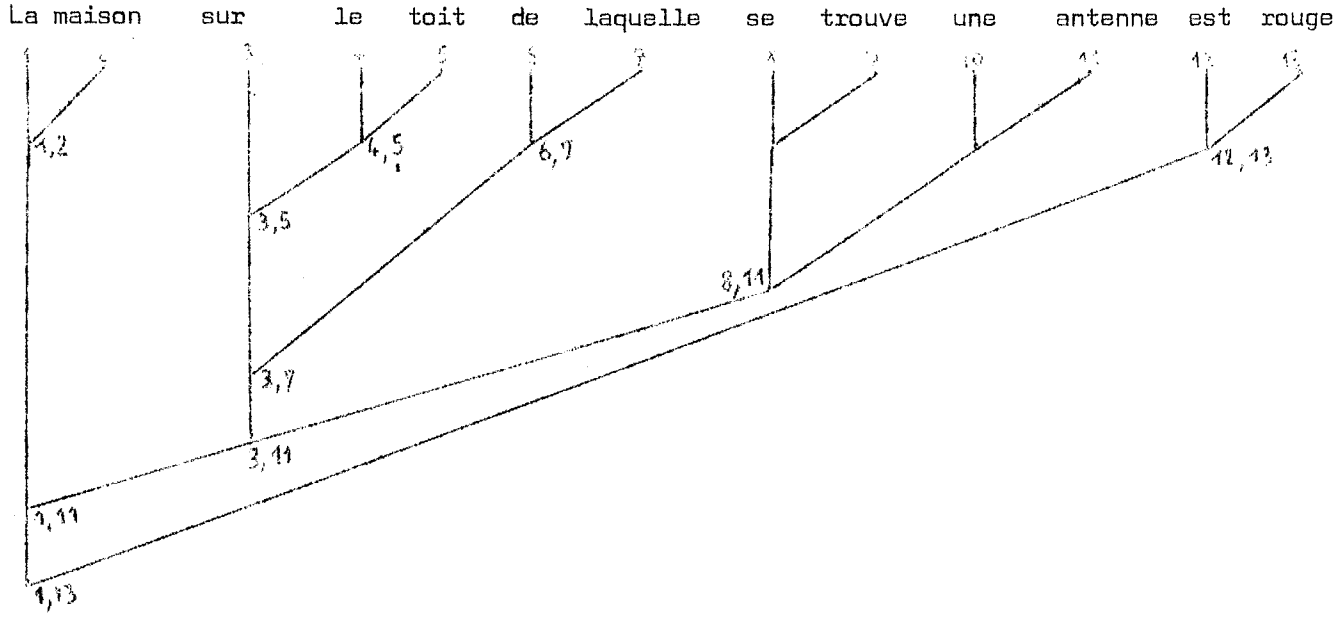
SUBG/NOM/FEM/SIN est une solution parasite dans cet exemple mais peut être utile ailleurs.

Remarque :

L'écriture d'un syntagme, comme il a été dit précédemment, nécessite au moins une valeur par variable associée à la catégorie dont il fait partie. On le vérifiera sur l'exemple ci-dessus,
la catégorie SUBG prenant les variables CAS,GNR,NBR,ANI
la catégorie CARD prenant les variables CAS,GNR

$II_{3,4}$.- Exemple de règles avec réserve

Nous prendrons un exemple en langue française :



Nous avons supposé qu'aucun syntagme élémentaire n'avait d'homographe externe. Nous avons bien un cas "context sensitive" puisque la construction de $s_{3,7}$ fait perdre les caractéristiques de s 6,7 (seules celles de $s_{3,5}$ sont nécessaires pour former $s_{3,11}$). Or ces caractéristiques de $s_{6,7}$ sont nécessaires pour former $s_{1,11}$ (accord de l'antécédent avec le pronom relatif). Nous allons donner 3 des règles nécessaires au traitement de cet exemple.

φ_6 : Formation de $s_{3,7}$
φ_7 : Formation de $s_{1,11}$
φ_3 : Transformation de $s_{3,11}$

φ_6 : SUBC | PROR//GEN $\Longrightarrow$ LOCR/CAS(1)/GNR(1)/NBR(1) - RESV - 6/GNR(2)/NBR(2)
φ_7 : SUBC | PROP//REL-GNR/NBR $\Longrightarrow$ PROP/PRI/CAS(1)/GNR(1.2)/NBR(1.2)
φ_3 : PROP//REL * RESV - 6 $\Longrightarrow$ PROP/REL/GNR(2)/NBR(2)

Ces règles font intervenir des symboles évidemment non définis en annexe et qu'il faut définir maintenant :

PROP	(Proposition)	prend les variables	NAT (Nature), CAS, GNR, NBR
PROR	(Pronom relatif)	" "	CAS,GNR,NBR
LOCR	(Locution relative)	" "	CAS,GNR,NBR
SUBC	(Substantif commun)	" "	CAS,GNR,NBR

La variable	NAT	prend les	valeurs	IND (indépendante),
				PRI (principale)
				REL (relative)
"	CAS	"	"	NOM,ACC,GEN,DAT,IDE
"	GNR	"	"	MAS,FEM
"	NBR	"	"	SIN,PLU

Alors $s_{3,5}$ s'écrit SUBC/DAT/MAS/SIN et $s_{6,7}$ PROR/GEN/FEM/SIN
l'application de φ_6 à $s_{3,5}$ et $s_{6,7}$ fournit :

LOCR/DAT/MAS/SIN - RESV - 6/FEM/SIN

Rappelons que la mémoire affectée à RESV-6 contiendra aussi ν_{d} et ν_{f}
de $s_{6,7}$ soit ($\nu_{d_1} = 6$ $\nu_{f_1} = 7$)
$s_{3,11}$ s'écrit PROP/REL avec $\nu_{d_2} = 3$ $\nu_{f_2} = 11$

Dès sa formation, il est transformé par φ_3 puisque $\nu_{d_2} < \nu_{d_1}$
et $\nu_{f_2} > \nu_{f_1}$ et devient PROP/REL/FEM/SIN.

A $s_{3,11}$ transformé et $s_{1,2}$ qui s'écrit SUBC/NOM/FEM/SIN, on peut appliquer
φ_1 pour obtenir $s_{1,11}$: PROP/PRI/NOM/FEM/SIN.

CHAPITRE III

PROJET D'UNE ORGANISATION EN MACHINE DU LANGAGE DE GRAMMAIRE

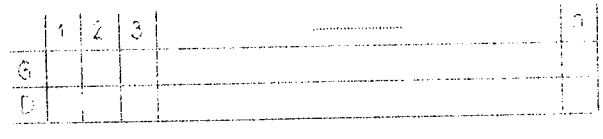
III$_1$.- GENERALITES

L'objet de ce chapitre est de donner un aperçu de la stratégie machine qui permettrait, étant donné 2 syntagmes, de trouver les règles qui peuvent leur être appliquées et de construire s'il y a lieu les syntagmes résultats. On supposera que l'élément de base de la machine est une mémoire à 36 positions, repérées comme suit :

A chaque catégorie seront affectées une grille de validation de règles et une ou plusieurs grilles de groupements de variables.

III$_2$.- GRILLE DE VALIDATION DE REGLES

III$_{2,1}$.- Soient $\Psi_1 \ldots \Psi_n$, la liste des n règles de la grammaire G_0 . On représentera le comportement des éléments d'une catégorie A, vis-à-vis de ces règles au moyen d'une grille de validation ainsi construite.



Sur la ligne G, on trouvera un 1 dans chaque colonne j , telle que le symbole de la catégorie A figure comme constituant gauche de la règle φ_j .

Sur la ligne D, on trouvera un 1 dans chaque colonne k, telle que le symbole de la catégorie A figure comme constituant droit de la règle φ_k .

Un syntagme recevra la grille de validation de sa catégorie d'appartenance. S'il s'agit d'un syntagme formé à partir d'une règle comportant une invalidation (voir II_2).

$$A \mid B \Longrightarrow C \;//\; P_C$$

On omettra les 1 se trouvant à l'intersection des colonnes correspondant aux règles de P_C et des uièmes lignes.

$III_{2,2}$.- Mise en mémoire d'une grille de validation

La ligne G sera rentrée en mémoire à raison d'une case par position. On rentrera la ligne D immédiatement à la suite. Chaque grille utilisera donc η mémoires, η étant le nombre entier immédiatement supérieur à 2n/36.

III_3.- GRILLES DE GROUPEMENTS DE VARIABLES

$III_{3,1}$.- Nous savons que la forme d'un élément fournit dans de très nombreux cas, plusieurs valeurs possibles par variable.

Exemple : La forme allemande "die", définit un élément de la catégorie "article défini" qui peut être :

- nominatif, féminin, singulier,

ou - accusatif, féminin, singulier,

ou - accusatif, pluriel (3 genres)

ou - accusatif, pluriel (3 genres).

En prenant la notation définie au chapitre II, on pourra écrire :

"die" : ARTD/NOM FEM SIN, ACC FEM SIN, NOM PLU, ACC PLU

Il apparait donc nécessaire de définir pour chaque catégorie les groupements de variables (voir $II_{1,1}$). Dans l'exemple précédent, on a mis en évidence le groupement CAS-GNR-NBR et 4 groupes du produit associé. (Au pluriel, on groupe les 3 genres qui n'y sont jamais différenciés).

Un premier procédé de définition de ces groupements pourrait consister à former toutes les combinaisons possibles des variables affectées à la catégorie étudiée (annexe). Le nombre de groupements ainsi formé est beaucoup trop élevé pour que ce procédé soit pris en considération. Par ailleurs, il est certain qu'un grand nombre de ces groupements n'aurait aucune utilisation pratique.

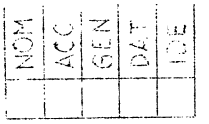
Le procédé préconisé consiste à définir les groupements en fonction des différentes règles qui feront intervenir les éléments de la catégorie. On commence donc par écrire toutes les règles de G_0 , et, de l'examen de la liste ainsi établie, on définit les groupements de variables à affecter à chaque catégorie.

Actuellement, ces règles de G_0 sont en train d'être écrites et nous ne pouvons pas donner l'ensemble de ces groupements. Néanmoins, il apparaît sur les règles déjà écrites, qu'aucun groupement ne comporte plus de 3 variables. Nous prendrons ce résultat partiel comme hypothèse simplificatrice pour la suite de ce chapitre. La présence éventuelle de groupements de plus de 3 variables ne mettrait d'ailleurs pas en défaut ce qui va être dit, mais compliquerait seulement les méthodes. A chaque groupement de variables, sera associée une grille, avec une case par groupe où l'on pourra mettre un 1 ou un 0 selon la présence ou l'absence de ce groupe dans les renseignements syntaxiques.

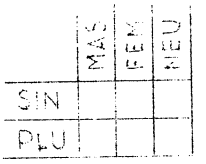
A un groupement d'une seule variable sera associée une grille linéaire et à un groupement de 2 variables une grille plane. Dans le cas d'un groupement de 3 variables, on introduira un sous groupement de deux des variables pour se ramener à une grille plane. Nous verrons ultérieurement la façon de mettre ces grilles en mémoire.

Exemples :

1) Groupement d'une variable CAS (Allemand)

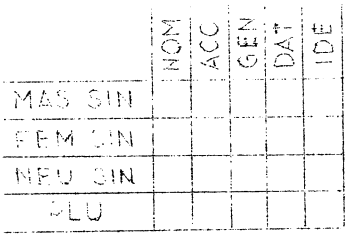


2) Groupement de 2 variables GNR,NBR



3) Groupement de 3 variables CAS,GNR,NBR (Allemand)

Sachant que les 3 genres ont la même forme au pluriel, Mr HAYS a proposé de grouper les 2 variables GNR,NBR dans un même sous groupement dont le produit contient les valeurs : MAS SIN, FEM SIN, NEU SIN, PLU.

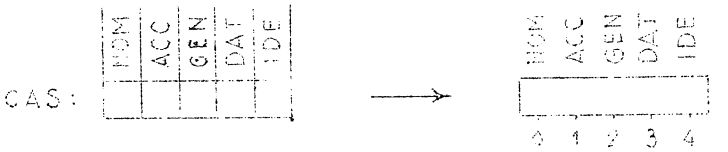


Dès la construction d'un syntagme, on remplira la grille de sa catégorie d'appartenance en remarquant (voir $II_{1,2,3}$) qu'un syntagme doit avoir au moins un groupe de chaque produit associé à sa catégorie.

III$_{3,2}$.- Mise en mémoire des grilles de groupement de variables

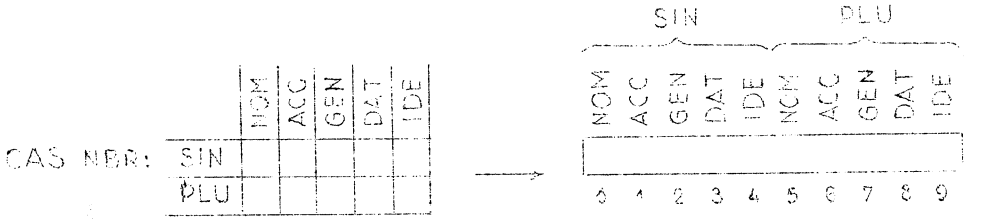
- Groupement d'une seule variable

La grille sera mise en mémoire à raison d'une case par position.



- Groupement de deux variables

La grille rectangulaire sera décomposée en autant de grilles linéaires qu'il y a de valeurs pour la variable ayant le moins de valeurs et mise ensuite en mémoire comme précédemment.



- Groupement de trois variables

On a vu que les groupements de 3 variables se ramènent aux 2 cas précédents.

III$_4$.- STRATEGIE DE LA CONSTRUCTION SYNTAXIQUE

III$_{4,1}$.- Soient 2 syntagmes si et sj représentés par leur différentes grilles.

Le procédé de construction commence par chercher les règles susceptibles de lier ces 2 syntagmes. Pour cela, on fait l'intersection de leurs grilles de validation, la ligne G du 1er syntagme avec la ligne D du 2ème et réciproquement.

Les positions ayant une intersection non vide donnent les numéros des règles cherchées.

III$_{4,2}$ - Règles

Chaque règle comprend :

- des grilles de manoeuvre qui sont utilisées d'une part pour comparer les valeurs des variables des syntagmes constituants possibles aux valeurs de variables imposées par la règle et d'autre part pour fabriquer le syntagme résultant.

- un sous programme qui décide de l'application de la règle et fabrique s'il y a lieu le résultat, en utilisant les grilles de manoeuvre.

Les comparaisons sur VVA et VVB se feront par intersection logique des grilles de variables des syntagmes et de grilles de manoeuvre, suivant des procédés qui vont être détaillés, les comparaisons sur NVI se feront par intersection des grilles de variables des syntagmes. On obtiendra les grilles de variables du syntagme résultant par transfert de certaines grilles de variables des constituants dans des grilles de manoeuvre prévues à cet effet.

III$_{4,2,1}$ - Stratégie de la comparaison sur VVA et VVB

Il semble d'après les règles déjà écrites qu'on ne peut rencontrer des groupements de plus de 2 variables, pour les parties gauches des règles, et nous prendrons ce résultat comme hypothèse. On va donc avoir à envisager les possibilités suivantes selon la forme des groupements comprenant les variables à comparer (pour le syntagme et pour la règle).

$III_{4,2,1,1}$.- Les variables à comparer sont dans 2 groupements de même forme

Il suffit de tester l'intersection des 2 grilles de ces groupements.

$III_{4,2,1,2}$.- Les variables de la règle sont contenues dans un groupement du syntagme qui comprend également d'autres variables

On force alors la grille de la règle à prendre la forme de la grille du syntagme.

Exemple :

Soit la règle A//MAS SIN, PLU/...
et un syntagme "candidat" qui possède le groupement CAS GNR NBR.
Les renseignements syntaxiques sont donc notés dans la grille :

	NOM	ACC	GEN	CAT	IDE
MAS SIN					
FEM SIN					
NEU SIN					
PLU					

La grille GNR NBR de la règle est :

	SIN	PLU
MAS	1	1
FEM		1
NEU		1

La grille de manoeuvre sera :

	NOM	ACC	GEN	DAT	IDE
MAS SIN	1	1	1	1	
FEM SIN					
NEU SIN					
PLU	1	1	1	1	

et on fera l'intersection des 2 grilles de même forme.

III$_{4,2,1,3}$.- La règle demande un groupement de deux variables qui sont dans 2 groupements du syntagme différents

On ne peut appliquer la méthode précédente, car on ne peut forcer une grille d'un syntagme, ce syntagme pouvant entrer dans d'autres règles. C'est donc le sous-programme de la règle qui sera chargé de fabriquer à partir des grilles du syntagme une grille de même forme que celle de la règle.

a) Si les 2 groupements du syntagme comprennent chacun une seule variable, on fabriquera un groupement de ces 2 variables, en répétant la grille de l'une des variables autant de fois qu'il y a de valeurs effectivement présentes dans l'autre.

Exemple :

Une règle demande une comparaison sur un groupement CAS GNR d'un syntagme possédant les 2 grilles :

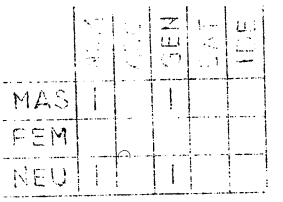
CAS :

NOM	ACC	GEN	DAT	IDE
1		1		

GNR :

MAS	FEM	NEU
1		1

Le sous-programme de la règle fabriquera au préalable la grille suivante :



	NOM	ACC	GEN	DAT	IDE
MAS	1		1		
FEM					
NEU	1		1		

b) Si les 2 groupements du syntagme comprennent également d'autres variables, on se ramènera au cas précédent en isolant les variables demandées par la règle au moyen de réunions logiques.

Supposons que dans l'exemple précédent le syntagme possède les 2 grilles

CAS :

NOM	ACC	GEN	DAT	IDE
1		1		

GNR NBR :

	MAS	FEM	NEU
SIN	1		
PLU		1	

On ramènera la grille GNR NBR à une grille GNR par la réunion logique des 2 lignes horizontales :

$III_{4,2,2}$ - Stratégie de la comparaison sur NVI

Il faut comparer les valeurs de 2 variables qui sont comprises dans certains groupements des 2 syntagmes. On se ramène toujours au cas où les 2 variables constituent à elles seules un groupement par le procédé exposé en $III_{4,2,1,3}$ b). Il suffit alors de faire l'intersection logique de 2 grilles linéaires.

$III_{4,2,3}$.- Il peut arriver qu'on ne puisse construire directement le syntagme résultant par de simples transferts. Cela se produira lorsque la catégorie de ce dernier comprend un groupement dont les variables appartiennent à divers groupements des catégories des syntagmes constituants et que la règle oblige ces variables à prendre les valeurs qu'elles ont dans les constituants. Le sous-programme devra alors faire sur les grilles de ces variables des opérations analogues à celles définies en $III_{4,2,1,3}$ b).

III_5.- EXEMPLES

Nous allons maintenant appliquer ce qui vient d'être dit à 3 des exemples du chapitre II en faisant cependant abstraction des grilles de validation, puisque les règles de G_0 ne sont pas encore toutes écrites.

$III_{5,1}$.- Complément de nom en allemand

SUBG | SUBG//GEN ⟹ SUBG//CAS(1) NBR(1)/GNR(1)

Nous supposerons que les groupements du SUBG sont :

CAS NBR et GNR

La variable CAS constitue à elle seule un groupement de la règle et entre dans le groupement CAS NBR du syntagme. On opère donc comme en $III_{4,2,1,2}$.

- Le syntagme comprend les 2 grilles :

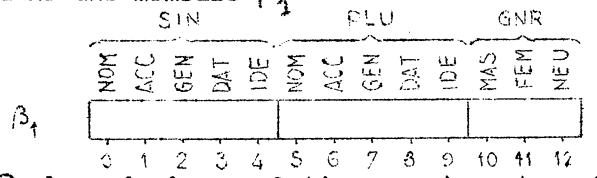
CAS NBR :

	NOM	ACC	GEN	DAT	IDE
SIN					
PLU					

et GNR :

MAS	FEM	NEU

qui sont mises dans une mémoire β_1

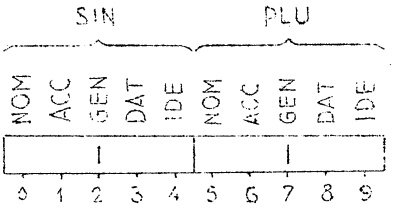


On notera β_1, β_2 les mémoires relatives au 1er et au 2ème constituant.

- La règle comprend une grille CAS NBR mise dans une mémoire ω,

	NOM	ACC	GEN	DAT	IDE
SIN			1		
PLU			1		

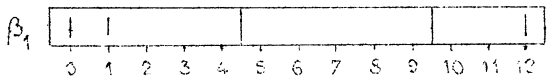
ω :

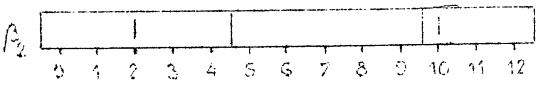


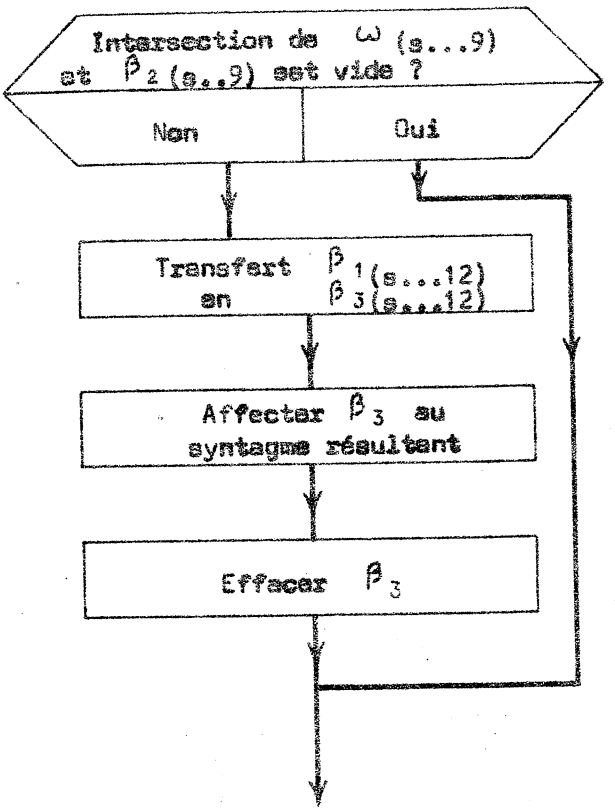
Une mémoire β_3 anologue à β pour le syntagme résultant et un sous-programme dont on verra ci-contre l'organigramme avec en regard l'état des mémoires pour l'exemple suivant :

SUBG/NOM SIN, ACC SIN/NEU
(Das Buch)

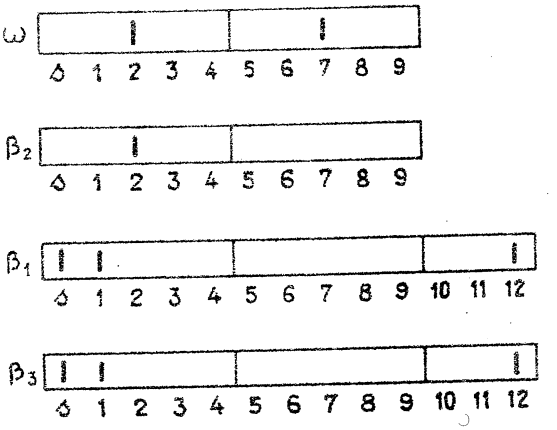
SUBG/GEN SIN/MAS
(des Lehrers)







Vers l'algorithme de construction syntaxique



SUBG/NOM SIN,ACC SIN/NEU

$III_{5,2}$ - Accord cardinal substantif en russe

CARD// NOM,ACI | SUBG// GEN - GNR ⇒ SUBG' // CAS(1)/NEU SIN,PLU,GNR(2)NBR(2)

On suppose que les groupements :

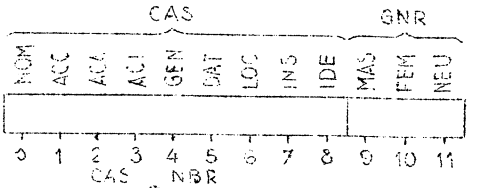
du CARD sont CAS et GNR

du SUBG sont CAS NBR, GNR, ANI

La comparaison sur VVA se fait comme il a été dit en $III_{4,2,1,1}$. La comparaison sur VVB oblige à créer pour la règle une grille CAS NBR ($III_{4,2,1,2}$) .
Les grilles des 2 syntagmes constituants seront mises en mémoire de la façon suivante :

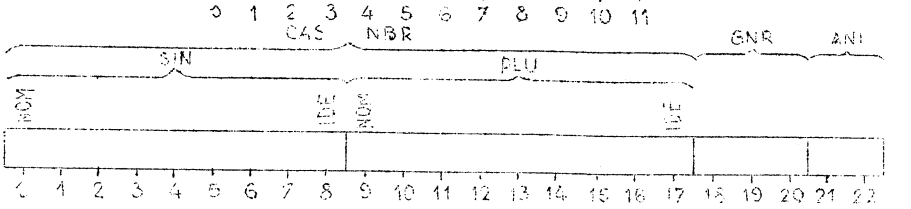
CARD :

Grille CAS
Grille GNR

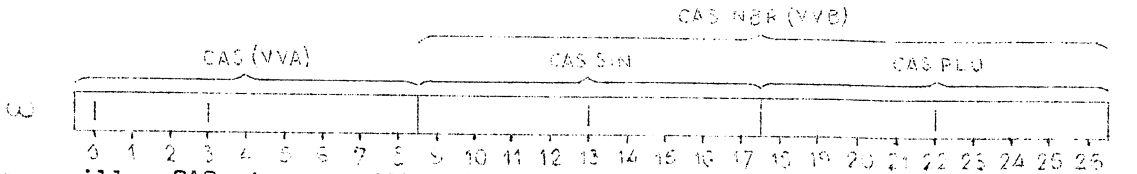


SUBG :

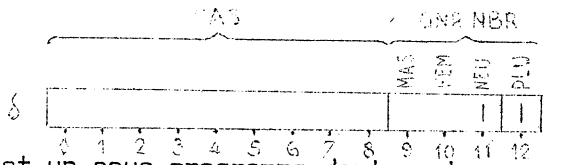
Grille CAS NBR
Grille GNR
Grille ANI



La règle comprend une grille CAS (VVA) et une grille CAS NBR (VVB) mises dans une mémoire ω

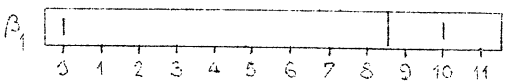


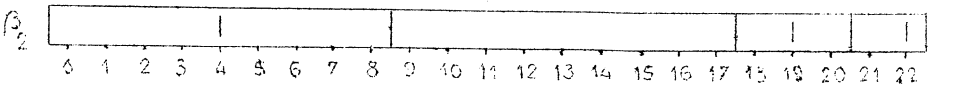
une grille CAS et une grille GNR NBR, pour la construction de SUBG' mises en mémoire δ



(Les groupements de SUBG' sont CAS,GNR NBR) et un sous-programme dont on trouvera ci-contre l'organigramme avec en regard l'état des mémoires pour les deux exemples suivants :

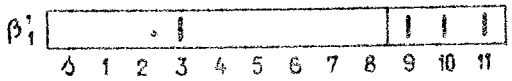
a) CARD/NOM/FEM SUBG/GEN SIN/FEM/INA

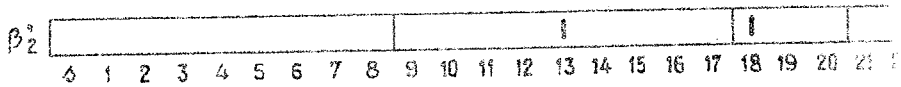


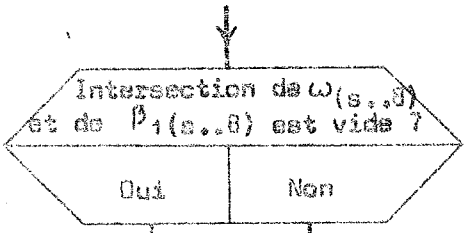


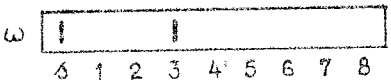
b) CARD/ACI/MAS,FEM,NEU

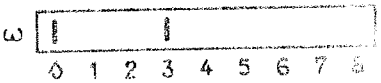
SUBG/GEN PLU/MAS/INA

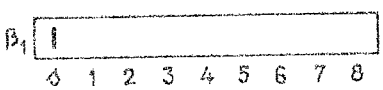


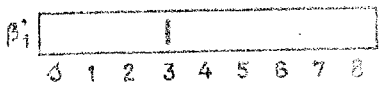


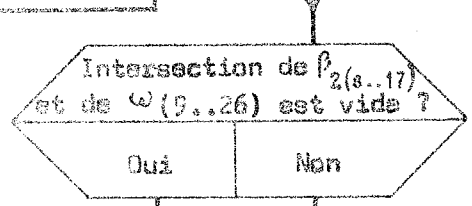


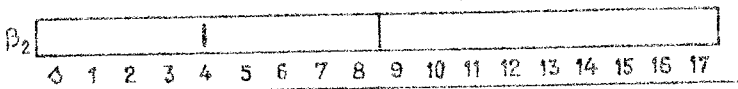


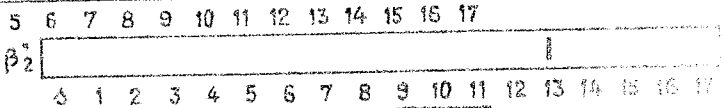


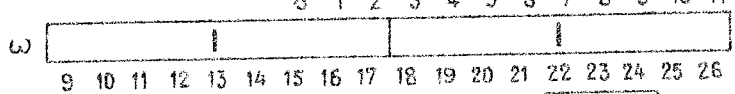


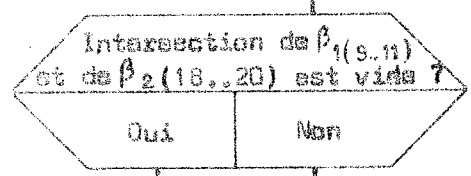


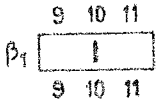


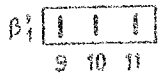


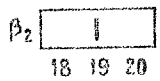


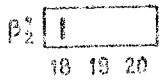


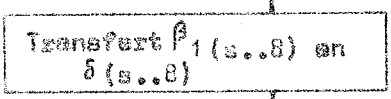


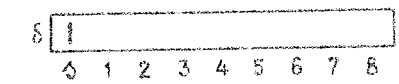


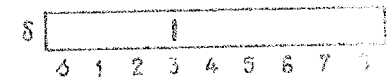


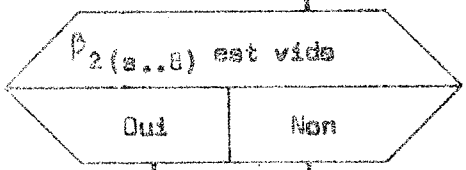






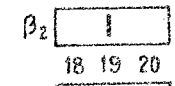


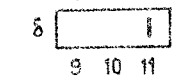


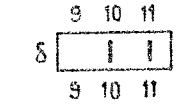


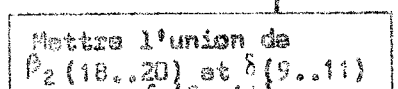
Non

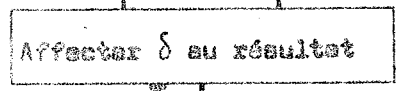
Oui

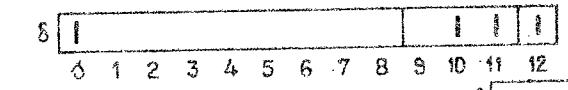


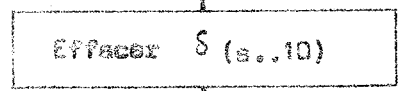


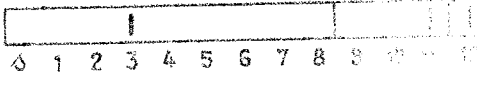


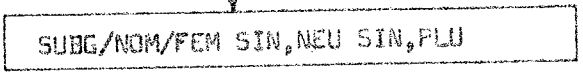


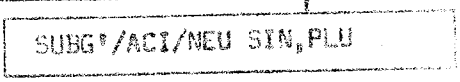








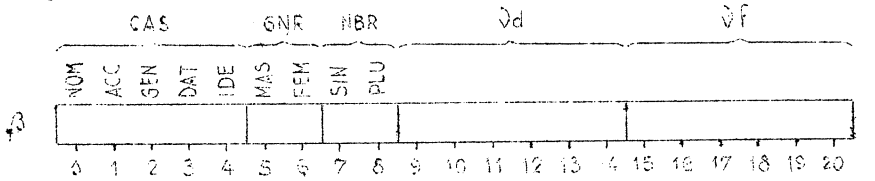




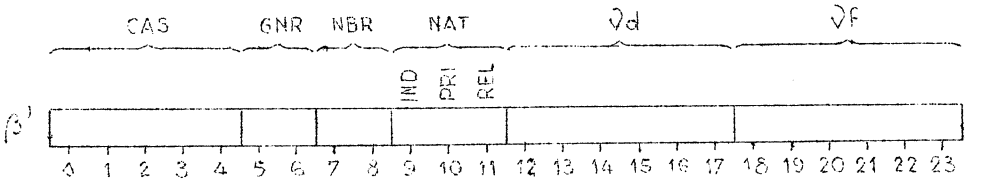
Vers l'algorithme de construction syntaxique

III$_{5,3}$.- Nous reprenons l'exemple de la subordonnée relative en langue française dont nous traitons l'application des règles φ_6 et φ_3 .
Nous utiliserons les numéros ν_d et ν_f des syntagmes.

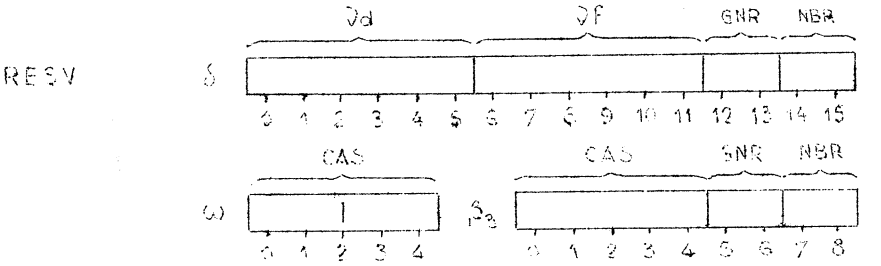
- Les catégories PROR, SUBC, LOCR prennent les variables CAS, GNR, NBR en groupements distincts répartis comme suit dans une mémoire β , en supposant $\nu < 2^6-1$



- la catégorie PROP prend les variables CAS, GNR, NBR, NAT répartis comme suit dans une mémoire β'

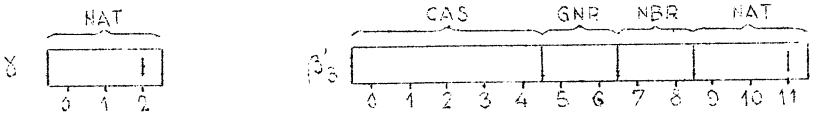


- La règle 6 contient : les mémoires de manoeuvre ,

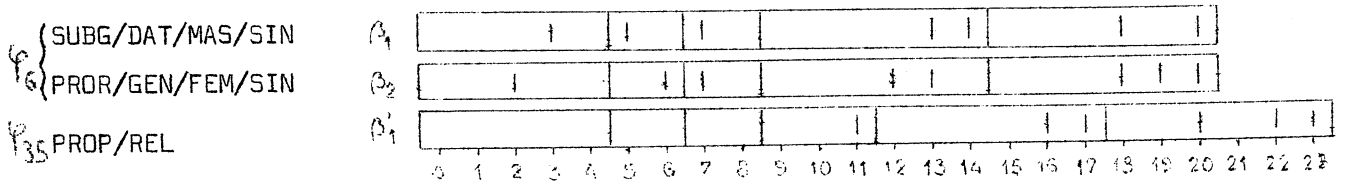


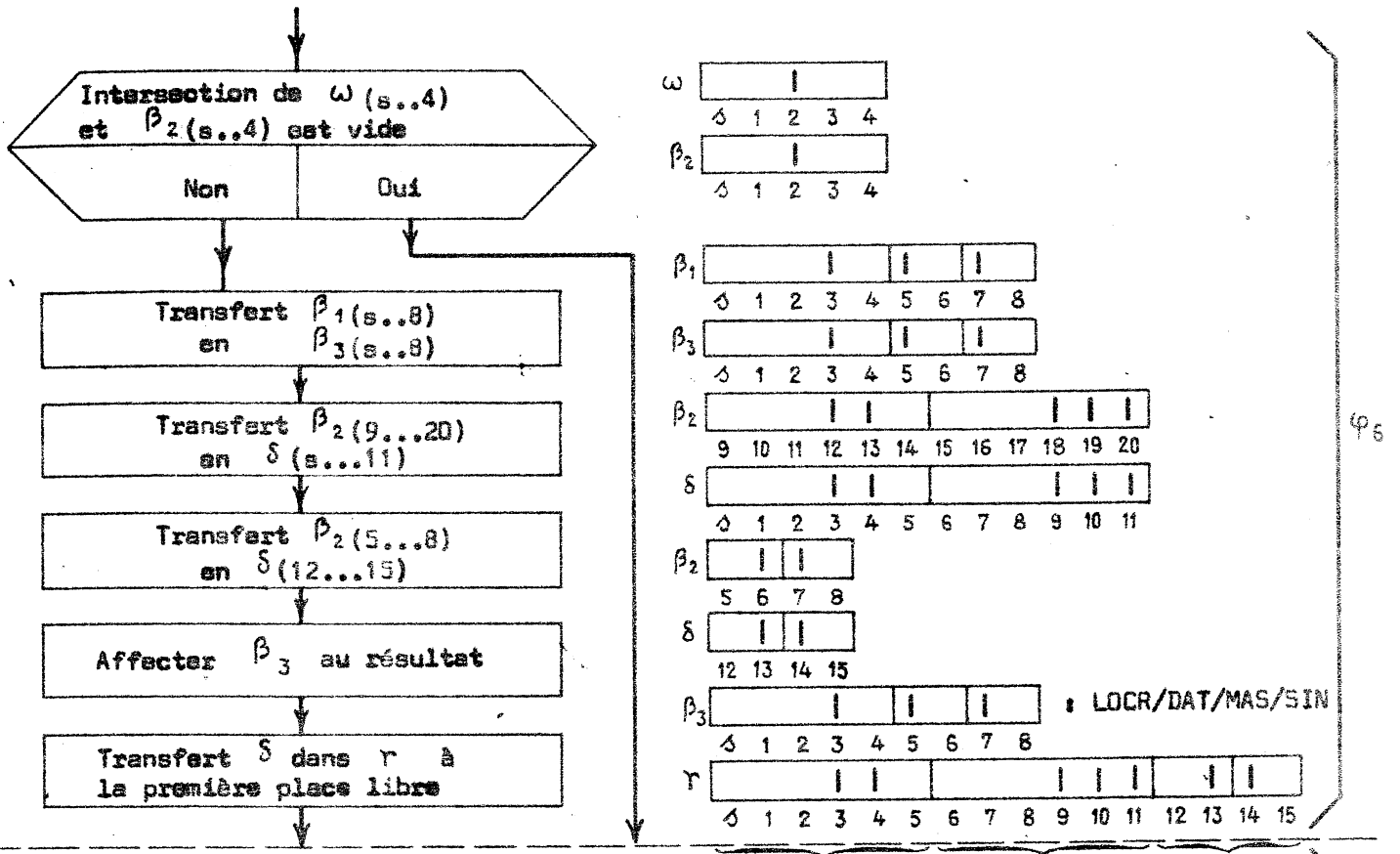
un ensemble de mémoires pouvant recevoir les suites de RESV correspondant aux applications successives de la règle.

- La règle 3 contient les mémoires de manoeuvre :

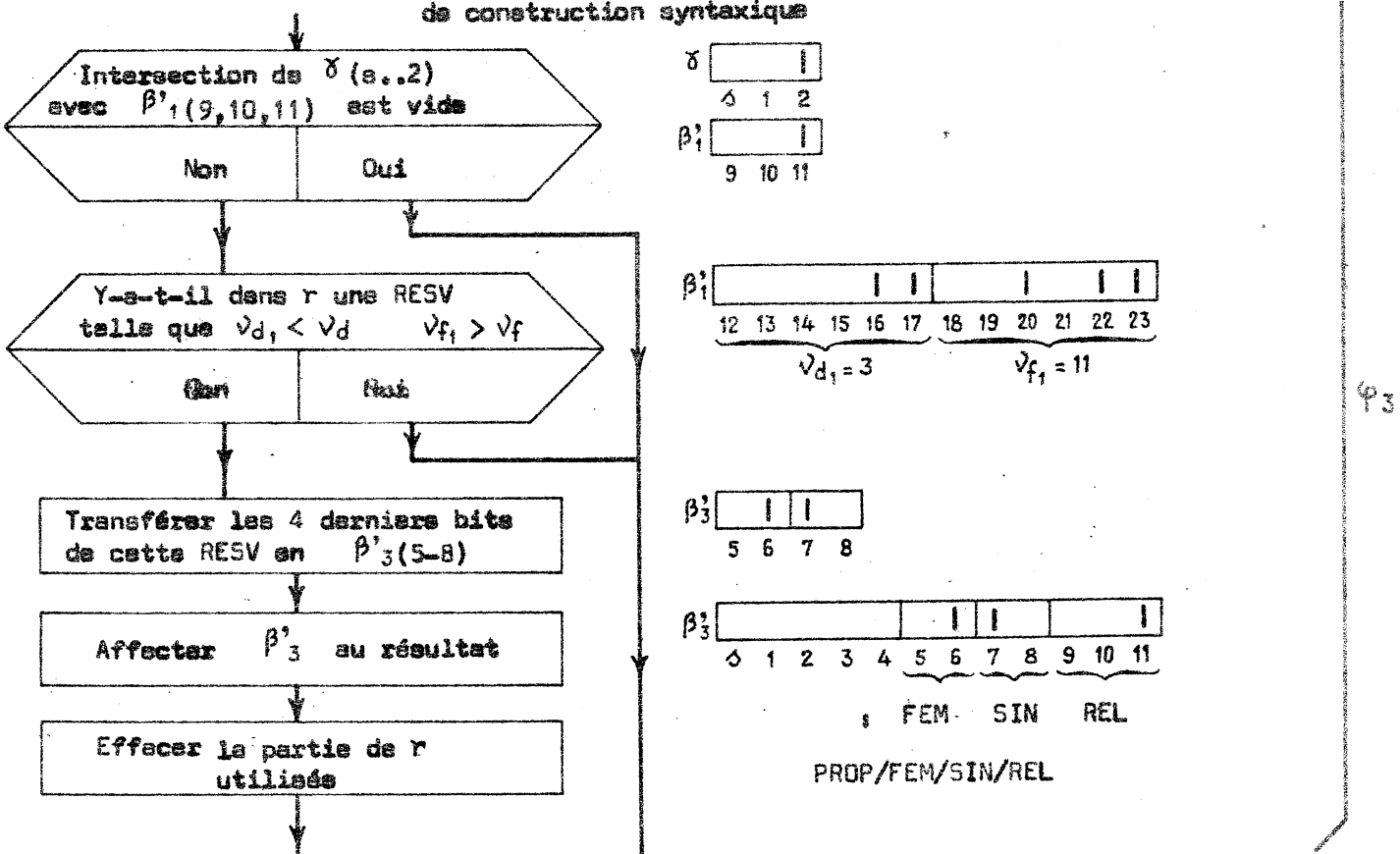


Ces deux règles contiennent également chacune un sous-programme dont on trouvera ci-contre l'organigramme avec en regard l'état des mémoires pour l'exemple :





Vers l'algorithme de construction syntaxique



Vers l'algorithme de construction syntaxique

CHAPITRE IV

LANGAGE D'INDEXAGE

IV_1.- GENERALITES

On a vu en $II_{2,1}$, qu'étant donné 2 catégories A et B et une règle $A \mid B \Longrightarrow C \quad (\varphi i)$, on pouvait trouver certains éléments de A n'acceptant d'être liés avec aucun élément de B. D'autre part, une règle récursive du type $A \mid B \Longrightarrow A$ peut être saturée ($II_{2,2,1}$). En outre, certains éléments de A peuvent n'accepter d'être liés qu'à certains éléments de B. Enfin, certains éléments demandent une construction impérative. (L'élément "souvenir", en français, pris comme verbe, doit être nécessairement accompagné d'un pronom personnel).

Lorsque ces anomalies sont purement lexicographiques, on peut envisager de les mettre en évidence au moyen d'un langage d'indexage. On donnera ici l'ébauche d'un tel langage pouvant en outre être utilisé pour les transferts de structures.

IV_2.- SYMBOLES DE BASE

Tout élément d'une langue naturelle (mot, ponctuation) est admis comme élément du langage à condition d'être entouré de guillemets.

Exemple : "Table"

La première fois qu'un tel élément est employé, il doit être répertorié dans une table avec le symbole de sa catégorie, ses variables et valeurs.

- Symbole désignation d'une dérivation

Le langage est divisé en fiches se rapportant chacune à une UL (unité lexicale) donnée, définie par une base standard. A partir de cette base, notée par le symbole X, on construit les différentes dérivations qui seront représentées par leur désignation :

$$X - \delta - S$$

où δ est soit le numéro de dérivation, soit un affixe en langue naturelle. S désigne la catégorie de la dérivation.

Exemple de dérivations : X - 17 - ADJQ

X - "OCT" - SUBC

On utilise également d'autres symboles qui vont être définis à mesure qu'ils seront trouvés dans la description du langage.

IV_3 - MODELE DU LANGAGE

Pour chaque UL , on remplit une fiche composée de lignes réparties en 3 séries.

Chaque ligne est construite sur le modèle suivant, formée de 6 compartiments séparés par les symboles // = .

Numéro de ligne // Indices de règles saturées // Résultat //
Partie gauche // Partie centrale = Partie droite .

$IV_{3,1}$.- Numéro de ligne

Il est constitué par le symbole DEBUT pour les lignes de la 1ère série, par le symbole SUITE pour celles de la 2ème série, par un nombre entier positif pour celles de la 3ème série.

$IV_{3,2}$.- Indices de règles saturées

Ce compartiment comprend les symboles des règles de G_o , séparés par des virgules, relatifs à des règles saturées. Rappelons (voir $II_{2,2,1}$) qu'une règle syntaxique est saturée pour un élément qu'elle a engendré lorsque cet élément ne peut plus être constituant de la règle.

Pour les lignes de la 2ème série, qui ne représentent pas une construction, ce compartiment comprendra des symboles de règles de G_o saturées au préalable, c'est-à-dire de règles de G_o invalidées pour l'élément étudié.

Dans les lignes de la 3ème série, les règles ainsi repérées seront des règles utilisées en partie gauche, saturées pour la désignation d' UL étudiée.

$IV_{3,3}$.- Résultat

Le résultat représente le syntagme résultant de la construction écrite en partie gauche, conformément à l'écriture du chapitre II.

$IV_{3,4}$.- Partie gauche

Elle comprend la désignation et une liste de composants, séparés par des * ou | selon que l'ordre des composants est indifférent ou non, pour la construction. Le symbole | est prioritaire sur le symbole * .

Exemple : $C_1 * C_2 | C_3$

Chaque composant C_i est de la forme (.....) . Entre les parenthèses se trouvent soit des symboles de syntagmes, soit des éléments de la langue naturelle, séparés par * ou | (avec la même signification que précédemment) et dont l'ensemble, une fois les constructions effectuées à l'intérieur de la parenthèse, est lié directement à la dérivation (repérée par sa désignation) au moyen d'une règle de G_0 .

Exemple : ("vers" | SUBP/ACC/SIN)

signifie que la préposition "vers", suivie (l'ordre intervient) d'un substantif propre à l'accusatif singulier, constitue un syntagme qui peut être construit avec la dérivation repérée en début de partie gauche.

Remarque : Si besoin est, on peut avoir toute une imbrication de parenthèses pour un même C_i

(xx(xxx))

Exemple de partie gauche

(...) | (X - 15 - SUBC) * (...)

IV$_{3,5}$.- Partie centrale

Elle contient une suite d'éléments de la forme i $\not\subset$ j séparés par des virgules, où i et j sont des nombres entiers positifs correspondant à la numérotation des composants de la partie gauche, (d'après leur ordre d'écriture). i $\not\subset$ j signifie que la présence de C_i nécessite celle de C_j .

$IV_{3,6}$.- Partie droite

Elle contient l'équivalent langue cible de la partie gauche. Les symboles et notations sont les mêmes. La désignation est remplacée par le numéro de l'équivalent langue cible correspondant (voir partie droite des lignes suites ($IV_{4,2}$)). Il y a possibilité d'aligner plusieurs parties droites séparées par des // .

IV_4.- LES DIFFERENTES SERIES

$IV_{4,1}$.- Première série

Elle ne comprend qu'une seule ligne, dont le numéro de ligne est DEBUT. Les compartiments, Résultat, Indices de fonctions saturées, Partie centrale et droite son vides. (On supprime alors les // inutiles). La partie gauche comprend le numéro identificateur de l'U.L., qui définit une base standard de cette U.L., suivi d'une liste de bases écrites en langue naturelle. C'est cette base standard qui sera par la suite désignée par le symbole X.

Exemple de ligne de la 1ère série

DEBUT // N° Identificateur, $'B'_1$, $'B'_2$ ..... $'B'_n$.

$IV_{4,2}$.- Deuxième série

Le numéro de ligne est SUITE , pour toutes les lignes de cette série. Chaque ligne est relative à une dérivation et on met dans le compartiment Indices de règles saturées, lessymboles des règles de G_o , invalidées pour cette dérivation. Les compartiments Résultat et Partie Centrale son vides. La partie gauche comprend le symbole de la désignation. La partie droite comprend les

équivalents langue cible de la dérivation, numérotés de façon que dans une même fiche, deux équivalents différents aient des numéros différents.

Exemple de ligne de la 2ème série :

SUITE // Υ_3, Υ_5 , Υ_2 // X - 'OCT' - SUBC = 21 : 'premier équivalent', 22 : 'deuxième équivalent'.

$IV_{4,3}$.- Troisième série

Chaque ligne de cette série donne des renseignements rolatifs à une construction donnée pour une dérivation.

Le numéro de ligne contient un nombre entier positif. Toutes les lignes d'une même fiche doivent avoir des numéros différents.

Les autres compartiments sont remplis comme il a été dit en IV_3 .

On remarquera que la Partie droite commence par le numéro d'un équivalent langue cible, nécessairement donné dans la Partie droite d'une ligne SUITE de la fiche.

Exemple de ligne de la troisième série :

9// Υ_2 , Υ_3 , Υ_5 //CARD/NOM;ACC/SIN // (X - 5 - SUBC)

(S_1) * (S_2) | (S_3) // 2 $\not\in$ 3 , 3 $\not\in$ 4

= 3 | (équivalent de S_1) * (équivalent de S_2) * (équivalent de S_3) .

IV_5.- CONCLUSION

Il faut maintenant montrer que le langage ainsi défini correspond bien aux objectifs fixés.

Les compartiments Indices de règles saturées des lignes SUITE donneront les invalidations des éléments terminaux. Ceux des lignes de 3ème série donneront les saturations des règles récursives.

En Partie gauche des lignes de 3ème série, on trouvera toutes les constructions particulières, pouvant éventuellement nécessiter des éclatements de catégories en plusieurs sous-catégories.

Les Parties centrales donneront les constructions impératives. En outre, un tel langage offre certains autres avantages, notamment grâce à la présence des Parties droites qui pourront permettre d'établir des tables de transfert de structures.

ANNEXE

SYMBOLES DE CATEGORIES - VARIABLES - VALEURS

Pour la commodité d'écriture des règles de G_0 et du langage λ il est souhaitable de définir des symboles pour les catégories, les noms de variables et les valeurs de ces variables.

Chaque catégorie sera représentée par un symbole formé de quatre caractères majuscules :

Exemple : Substantif commun : SUBC
Pronom interrogatif : PRO ?

Chaque catégorie peut être affectée d'une liste de variables possibles. Les noms de ces variables seront symbolisés par des groupes de trois lettres majuscules. Ces symboles seront supposés ordonnés une fois pour toutes.

Exemple : Cas : CAS
Genre : GNR

Chaque variable peut prendre plusieurs valeurs, symbolisées par des suites de trois lettres majuscules. Les différentes valeurs d'une même variable seront également ordonnées.

Exemple : Accusatif : ACC
Pluriel : PLU

On trouvera ci-après, une liste des symboles des noms de variables, et de leurs valeurs, puis une liste des symboles de catégories, avec en regard, les différents noms de variables qui peuvent leur être affectées.

On remarquera que seuls sont définis des symboles se rapportant à des catégories d'éléments terminaux, qui correspondent au stade actuel de notre recherche. Il faudra ultérieurement définir des symboles de catégories se rapportant à des éléments de niveaux supérieurs à mesure qu'ils seront trouvés par construction.

LISTE DES VARIABLES GRAMMATICALES ET DE LEURS VALEURS

Cas	CAS
Nominatif	NOM
Accusatif	ACC
Accusatif animé x	ACA
Accusatif inanimé x	ACI
Génitif	GEN
Datif	DAT
Locatif x	LOC
Instrumental x	INS
Indéclinable	IDE
genre	GNR
Masculin	MAS
Féminin	FEM
Neutre	NEU
nombre	NBR
Singulier	SIN
Pluriel	PLU

Personne	PRS
1ère personne	UNP
2ème personne	DUE
3ème personne	TRE
Animation x	ANM
Animé	ANI
Inanimé	INA
Aspect x	ASP
Perfectif	PER
Imperfectif	IPF
Degré	DGR
Positif	POS
Comparatif	COM
Superlatif	SUP
Forme	FRM
Forme courte	FOC
Forme longue	FOL

+ Déclinaison	DCL
Forte	FRT
Faible	FBL
Mixte	MXT
Temps	TPS
Présent	PRE
Futur	FUT
Imparfait x	IPA
Passé	PAS
Mode	MOD
Indicatif	IND
Subjonctif	SUB
Impératif	IMP
Conditionnel	CDL
Transitivité	TVT
Transitif	TRS
Intransitif	ITR

x : pour le russe seulement

\+ : pour l'allemand seulement.

Liste des catégories

x : russe
+ : allemand

catégorie	symbole	CAS	GNR	NBR	PRS	ANM	ASP	DGR	FRM	DCL	TPS	MOD	TVT
Substantif général	SUBG	x +	x +	x +		x							
commun	SUBC	x +	x +	x +		x							
propre	SUBP	x +	x +	x +		x							
dérivé d'adjec.	SUBA	x +	x +	x +		x							
Adjectif général	ADJG	x +	x +	x +		x		x +	x +	+			
Qualificatif	ADJQ	x +	x +	x +		x		x +	x +	+			
démonstratif	ADJD	x +	x +	x +		x			+	+			
Interrogatif	ADJ?	x +	x +	x +		x				+			
Indéfini	ADJI	x +	x +	x +		x							
Négatif(x)	ADJN	x	x	x		x							
Ordinal	ADJO	x +	x +	x +		x			+	+			
Possessif	ADJP	x +	x +	x +		x				+			
Relatif	ADJR	x	x	x		x				+			
Participe présent actif	PPRA	x +	x +	x +		x		+	+	+			
Passif(x)	PPRP	x	x	x		x			x				
Participe passé actif(x)	PPAA	x	x	x		x	x						
passif	PPAP	x +	x +	x +		x	x	+	x +	+			
Cardinal	CARD	x +	x	x		x							
Verbe	VERB	x	x	x +	x +		x				x +	x +	x +
Verbe général	VERG	x	x	x	x	x	x		x		x	x	x
Gérondif (x)	GERO						x						
Infinitif	INFI						x						
Adverbe général	ADVG							x +					
Normal	ADVN							x +					
Interrogatif	ADV?							x					
Pronominal	ADVP							x					

Catégorie	symbole	CAS	GNR	NBR	PRS	ANM	ASP	DGR	FRM	DCL	TPS	MOD	TVT
Conjonction générale	CONG												
coordination	CCCO												
Subordination	COSU												
Préposition	PREP												
Postposition	POST												
Pronom général	PROG	x +	x +	x +	x +	x				+			
Personnel	PROJ	x +	x +	x +	x +	x				+			
Démonstratif	PROD	x +	x +	x +	x +	x				+			
Possessif	PROP	x +	x +	x +	x +	x				+			
Relatif	PROR	x +	x +	x +	x +	x				+			
Interrogatif	PRO?	x +	x +	x +	x +	x				+			
Indéfini	PROI	x +	x +	x +	x +	x				+			
Négatif (x)	PRON	x +	x +	x +	x +	x							
Réfléchi	PROX	x +	x +	x +	x +	x				+			
Réciproque	PROQ	x +	x +	x +	x +	x				+			
Particule (x)	PART												
Séparable (+)	PSEP												
Article défini (+)	ARTD	+	+	+									
Indéfini (+)	ARTI	+	+	+									

BIBLIOGRAPHIE

[1] - D. AUGEREAU - Thèse de 3ème cycle : Utilisation des informations sémantiques en traduction automatique.

[2] - B. VAUQUOIS - Langages artificiels, Syotèmes formels et traduction automatique - Document CETAG

[3] - G. VEILLON - Thèse de 3ème cycle : Consultation d'un dictionnaire et Analyse morphologique en Traduction automatique.

[4] - BAR HITTEL and E. SHAMIR : Finite state Languages, Formal representations and adequacy problems.

[5] - N. CHOMSKY - On certain formal Properties of grammars Information and Control - Vol 2, Numbers 1,4 - 1959;

[7] - Mr. SAKAI - Congrès International de Traduction automatique de Teddington - Septembre (1961).

[8] - G. VEILLON - Communication personnelle.

PLAN GENERAL