De la description des propriétés linguistiques à l'analyse d'une langue

Bès G. G. (GRIL) Hagège C. (GRIL-ILTEC) Coheur L. (GRIL-IST)

Résumé

Un algorithme d'analyse de texte – l'algorithme d'analyse par les feuilles – est présenté. Notre approche, inscrite dans le cadre 5P, se caractérise par:

- La source déclarative des algorithmes d'analyse est physiquement complètement séparée des descriptions linguistiques. Nous nous démarquons en cela des approches utilisant les grammaires d'unification, où la grammaire a une double fonction : exprimer les descriptions linguistiques et servir de source aux traitements.
- La possibilité de moduler la finesse de l'analyse selon les fonctionnalités que l'on souhaite assurer en extrayant des descriptions plus ou moins d'information.
- La mise en évidence de principes généraux sur le fonctionnement des langues (transition diabolique).

Les idées de base de la méthodologie 5P sont rappelées, ainsi que l'appareil descriptif utilisé. On montre par la suite comment, à partir des descriptions linguistiques formalisées, dériver l'information nécessaire aux différents aspects de l'algorithme d'analyse. Enfin, les grandes lignes d'un algorithme de balisage de syntagmes nominaux dans un texte sont présentées.

1. Cadre théorique – 5P

Notre travail s'inscrit dans l'approche **5P** (cf. [7]) ou Protocoles (**P1**), Propriétés (**P2**), Projections (**P3**), Principes (**P4**) et Processus (**P5**) qui traite d'appliquer effectivement à l'étude des langues naturelles la démarche des sciences du réel.

P1 met l'accent sur la nécessité de recourir à des observateurs modélisés afin de consigner des observations systématiques – les protocoles – sur les expressions linguistiques. Les protocoles permettent d'induire des hypothèses sur le fonctionnement des langues et de les tester.
P2 consiste en la formalisation de caractéristiques linguistiques (relations d'ordre entre deux catégories, exigences d'une catégorie sur une autre, etc.) propres aux expressions linguistiques d'un type donné (syntagme nominal, phrase, phrase verbale noyau, etc.).
P3 exprime des généralisations à partir de P2 sur une langue particulière.
P4 exprime également des généralisations, mais sur des groupes significatifs de langues.
Enfin, **P5** comprend tout ce qui est effectivement implanté et implantable en machine.

On mettra ici l'accent sur P2, sur ce qui semble être un principe possible de P4 et que nous appelons les **transitions diaboliques**, et sur un algorithme d'analyse (par les feuilles), relevant donc de P5. Cet algorithme exploite les P2 et le principe des transitions diaboliques et permet de parenthétiser des suites *sn* (syntagme nominal) d'un texte.

2. Les Propriétés - P2

Les Propriétés P2 expriment, en les factorisant, les caractéristiques descriptives des expressions (suites de mots) d'une langue. Elles sont strictement neutres par rapport aux Processus qui vont, au cours de l'analyse, vérifier leur satisfaction dans les expressions linguistiques. Elles se formulent moyennant des **catégories** (ensembles de couples étiquette-valeur, les traits) et des **identificateurs de modèles**.
Un **modèle Id** (m-Id) est une suite de catégories ou d'identificateurs de modèles qui satisfait l'ensemble des Propriétés identifiées par Id (**Propriétés-Id**).

On distingue trois grands types de Propriétés : Propriétés d'existence, de linéarité et de fléchage qui expriment respectivement, et de manière indépendante, l'existence dans un modèle des entités linguistiques (catégories ou identificateurs de modèles), les relations d'ordre entre ces entités et les relations de fléchage (ou dépendances) qu'entretiennent ces entités. C'est le fléchage qui permet de calculer la représentation sémantique.

On va illustrer les propriétés en s'appuyant sur une description simplifiée (elle ne tient pas compte de la coordination) de ce que nous appellerons le syntagme nominal noyau (noté *snn*), c'est à dire le sous-ensemble de suites *sn* dont la limite droite est le noyau du *sn*, en supposant que *det*(erminant) subsume *art*(icle), *pos*(sessif) et *dem*(onstratif), que *art*(icle) subsume *artd* (article défini) et *arti* (article indéfini), que *n*(om) subsume *nc* (nom commun) et *np* (nom propre), et que *adj* désignant la catégorie des adjectifs et *card* celle des

cardinaux.

2.1. Propriétés-Id d'existence

Les catégories sont organisées selon une hiérarchie d'héritage multiple monotonique qui exprime leurs relations de subsomption. L'expression des P2 d'existence tire parti de ces relations – dans tous les cas, on doit comprendre que ces stipulations sont valables pour toutes les catégories identiques ou subsumées par celles utilisées dans la stipulation.

Les Propriétés-Id d'existence notées par :
amod stipulent quelles sont les catégories ou identificateurs de modèles susceptibles d'entrer dans la composition d'un *m-Id*.
Exemple :*amod(snn,{det, adj, card, n,…}) exprime que l*es *art*, les *pos*, les *dem* (subsumés par *det*), les *adj*, les *card* et les *np* et *nc* (subsumés par *n*) peuvent, parmi d'autres intégrer un *snn*.

uniq désignent parmi les entités introduites par *amod*, celles qui ne peuvent être utilisées qu'une fois dans un *m-Id*.
Exemple : *uniq(snn,{det, adj, card, n,…})* exprime qu'il n'y a dans un *snn* qu'une seule catégorie *det*, *adj*, *card*, ou *n*, ce qui évite la coexistence d'un *art* et d'un *dem*, ou d'un *np* et d'un *nc*.

obligdi mettent en évidence les entités qui sont des noyaux possibles des *m-Id*, une seule de ces entités devant obligatoirement être noyau dans tout *m-Id.*
Exemple : *obligdi(snn, {card, adj, n,…})* indique que ce sont les *adj*, les *card* et les *np* ou *nc* qui peuvent être noyau d'un *snn*, un *snn* n'ayant qu'un et un seul noyau.

exig et ***exclu*** indiquent respectivement quelles sont les entités qui doivent ou ne doivent pas être incorporées à un *m-Id*, étant donnée la présence de telles autres entités dans ce même *m-Id*.
Exemple : *exig(snn, {{nc}, {{det}, {card}}), e exclu(snn, {{card}, {arti}}),* expriment que :
a) un *nc* exige un *det* et/ou un *card* dans le *snn* où il apparaît ((*mes*, *les*, *ces*) *livres*, *trois livres*, *ces trois livres*, mais **beaux livres* et **livres*) ;
b) un *card* et un *art*i s'excluent mutuellement dans un *snn* (**des trois livres*).

2.2. Propriétés-Id de linéarité

precede va spécifier les relations d'ordre entre les entités susceptibles d'intégrer un *m-Id*.
Exemple : *precede(snn, det, {n})* indique que, si dans un *snn* il y a un *det* et un *n*, le *det* précède le *n* (*un livre*, *ton Jacques*).

2.3. Propriétés-Id de fléchage

fleche va spécifier les dépendances entre les entités dans la liste d'un *m-Id*.
Exemple : *fleche(snn, {x_}, {{x@}})* stipule que toute catégorie non-noyau (notée *x_*) flèche sur la catégorie noyau (notée *x@*)

3. Un candidat à P4 – le principe de la transition diabolique

Etant donnés A^+ et B^+ (assemblages sans symbole nul sur les alphabets A et B), on peut définir la relation de codage comme la fonction bijective permettant d'associer à *a* appartenant à A^+ , *b* appartenant à B^+. On sait par ailleurs qu'il existe différents types de codage : utilisation des suites de longueur variable ou fixe, utilisation ou non de symboles démarcatifs *ad-hoc*, etc.

Selon Chomsky et Miller (cf. [5]), sur la base des travaux de Schützenberger, il y a plusieurs manières d'assurer la non ambiguïté de la fonction de codage. L'une d'elle est désignée par *left tree code* (codage à gauche) et est définie comme suit :
"no spelling of any word forms an initial segment (left segment) of the spelling of any word", et, mutatis mutandis, pour les "right tree code" (codage à droite).

Dans notre optique, A est un ensemble de catégories de mots et on se pose le problème d'associer à des suites de A^+ une étiquette de constituant *snn*. Il s'agit donc bien d'une opération s'approchant des opérations de codage puisque le problème est de reconnaître les limites gauche et droite de suites de catégories auxquelles on veut associer une étiquette.
On peut observer que les suites *snn* ne satisfont ni la contrainte du codage à gauche (le noyau de ces suites est souligné) :

(les jolies <u>filles</u>)$_{snn}$ avaient choisi de rester
(les <u>jolies</u>)$_{snn}$ avaient choisi de rester

ni la contrainte du codage à droite :

(les <u>trois</u>)$_{snn}$ sont restés
(<u>trois</u>)$_{snn}$ sont restés

Et pourtant, en français, si l'on a ouvert un *snn* à la gauche d'une suite de mots, on peut, par observation de la concaténation (notée +) de catégories cat_i et cat_j (cat_i+cat_j), déterminer

dans beaucoup de cas si cat_i est le noyau du *snn* ouvert et ainsi, si la limite droite de ce *snn* passe entre cat_i et cat_j.

Nous appelons ce phénomène le principe de la **transition diabolique**, principe qui peut être étendu à d'autres phénomènes linguistiques (reconnaissance des types syllabiques en espagnol, autres syntagmes noyau en français, portugais et espagnol, reconnaissance des mots composés cf. [1]).

Si l'on se donne la notion de *bien formé (syntaxiquement)*, le principe peut être formulé comme suit :

Xn note une variable sur les étiquettes des syntagmes noyau.
sXn note l'ensemble de toutes et seulement les suites bien formées à étiqueter comme Xn
sxn note une suite appartenant à sXn
cat_i et cat_j notent des catégories
+ note l'opération de concaténation de suites

si
cat_i et cat_j peuvent intégrer des suites de sXn
cat_i précède cat_j dans toute suite de sXn
cat_i est le noyau d'une suite sxn
alors
sxn + cat_j n'est pas une sous-suite d'une suite bien formée.

On retrouve ainsi l'exigence du codage à gauche, mais en la formulant sur la concaténation sxn + C (C étant une variable qui s'instancie sur les catégories ou la virgule). Ainsi :

Etant donné sX = {x | x = sxn + C}
x sous-suite d'une suite bien formée
sx_i, sx_j appartienent à sX
alors, aucune des deux suites sx_i et sx_j n'est la suite initiale de l'autre.

Exemples pour les suites *snn* en français (dans lesquelles le noyau est souligné).

Exemples	Restrictions sur C, instanciée par une catégorie
(les trois jolies <u>filles</u>) + C	C n'est pas conditionnée par le principe
(les trois <u>jolies</u>) + C	C n'appartient pas à {n }
(les <u>trois</u>) + C	C n'appartient pas à {n, adj}
(les <u>mêmes</u>) + C	C n'appartient pas à {n, adj, card}
(la <u>deuxième</u>) + C	C n'appartient pas à {n, adj}

4. L'analyse par les feuilles dans P5

Les structures déclaratives sur lesquelles vont opérer les algorithmes de P5 sont le résultat de l'application d'une fonction sur les descriptions linguistiques de P2.

L'algorithme d'analyse par les feuilles (désormais noté AF), n'est qu'une façon de concevoir les traitements dans P5. On peut en envisager d'autres (cf. [4]).

4.1. La source déclarative de AF - feuilles et blocs

Nous appelons **feuilles** et **blocs** les structures qui contiennent l'information obtenue à partir de P2 et qui vont constituer la totalité de l'information linguistique utilisée par AF.

Feuilles et blocs sont des notions très proches qui se distinguent essentiellement par le niveau de profondeur de l'entité auxquels ils sont rattachés: une **feuilles** est une structure rattachée à une unité linguistique qui indique son comportement à l'intérieur d'un modèle ; un **bloc** est une structure rattachée à un modèle qui indique son comportement à l'intérieur d'un autre modèle dans lequel il peut apparaître.

On va illustrer intuitivement en s'appuyant sur les exemples présentés plus haut, quelle est le type d'information que l'on peut extraire de P2 et injecter dans une feuille : on peut dériver de P2 (*amod*, déclaration du vocabulaire des modèles) le fait qu'il pourra exister une feuille rattachée à la catégorie article défini (*artd*) dans un modèle *m-nn*. On peut également dériver de P2, le fait que cette feuille ne pourra jamais être le noyau d'un modèle *m-nn*, et ce par la propriété *obligdi* (qui stipule quelles sont les catégories noyau d'un modèle). En effet, la catégorie *artd* n'est pas présente dans l'ensemble défini en deuxième argument de la propriété. De par les autres propriétés (existence et linéarité), on peut déduire que *artd* peut commencer le *m-nn*, que cette catégorie peut être suivie par les catégories *card*, *adj* et *n* (et pas *dem* puisqu'elles ne pourront jamais être dans le même modèle de par la propriété d'unicité). On peut également déduire que l'*arti* ne pourra jamais constituer à lui seul un modèle, puisqu'il n'est pas noyau et que tout modèle doit avoir un noyau. Ce type d'information peut être

extrait pour toutes les catégories de tous les modèles décrits par les propriétés. Selon le degré de finesse de l'analyse que l'on souhaite obtenir, selon le type de traitement que l'on envisage de faire (simple analyse de surface d'un texte, avec ou sans dépendances, texte d'entrée grammatical ou non grammatical), on peut injecter aux feuilles plus ou moins d'information. Nous allons à la section 5 présenter un algorithme d'analyse par les feuilles de simple balisage des sn du texte (sans inclure les relatives, c'est à dire syntagmes nominaux noyaux avec extensions de syntagmes adjectivaux noyaux).

4.2. Feuilles et blocs pour un algorithme de simple balisage

4.2.1. Structure des feuilles

Les feuilles ont la forme suivante :

feuille(Uni, Cat, Mod, Com, Term)

où *:*

· *Uni* est l'unité linguistique reperée dans un texte.

· *Cat* est une catégorie définie dans P2 rattachée à cette unité linguistique.

· *Mod* est l'étiquette du modèle dans lequel la catégorie *Cat* est déclarée.

· *Com* indique si cette catégorie dans ce modèle commence parfois (2), toujours (1) ou jamais (0) le modèle.

· *Term* indique si cette catégorie dans ce modèle termine parfois toujours ou jamais le modèle.

Exemple : *feuille(la, artd, m-nn, 2, 0)* représente une des feuilles rattachée à l'unité linguistique *la*, ce qui signifie que *la* est de catégorie *artd* dans le *m-nn*, qu'elle commence parfois ce modèle (*la vie* ou *toute la vie*) et qu'elle ne peut jamais le terminer (**toute la*).

4.2.2. Structure des blocs

Les blocs ont la forme suivante :

bloc(Mod, ModSup, Com, Term, Suiv)

où:

· *Mod* est l'étiquette du modèle auquel est rattaché le bloc.

· *ModSup* est l'étiquette du modèle dans lequel le modèle *Mod* peut apparaître.

· *Com* et *Term* indiquent respectivement si le modèle en premier argument commence/termine parfois (2), toujours (1) ou jamais (0) le modèle en deuxième argument.

· *Suiv* est une liste (éventuellement vide) de modèles ou de catégories qui peuvent suivre le modèle en premier argument à l'intérieur du modèle en deuxième argument.

Exemple : *bloc(m-an1, m-nn, 0, 2, [n])* signifie que les modèle adjectival noyau (*m-an1*[1]) peuvent se rencontrer dans un modèle *m-nn* (*la (très belle) fille*) et que dans ce cas, ils ne commencent jamais le *m-nn* mais le terminent parfois (**(très belle) fille* et *la (très belle)*). A l'intérieur du *m-nn*, un *m-an1* peut être suivi par un *n*.

5. L'algorithme AF

Le but de l'algorithme est le balisage des textes en syntagmes noyaux décrits par les propriétés. L'algorithme est simple puisqu'il consiste en la concaténation de feuilles et de blocs accompagnée de quelques vérifications sur la possibilité ou non de concaténer.

5.1. Feuillage du texte ou obtention de l'entrée de l'algorithme

Le texte initial est segmenté et étiqueté par un programme d'étiquetage de texte (Smorph, cf. [1]). A partir des propriétés (P2), blocs et patrons de feuilles peuvent être en principe calculés. On appelle *patron de feuille* une structure feuille dans laquelle le premier élément (l'unité linguistique) n'est pas instancié. Par le biais des catégories, à partir du texte étiqueté d'une part et des patrons de feuilles d'autre part, on peut associer à chaque unité du texte, une ou plusieurs feuilles. On appelle cette opération le *feuillage du texte*. Cette association peut se faire de façon dynamique au moment de l'analyse.

Exemple : pour *les jolies femmes* on obtient (en fonction des choix descriptifs qui ont été faits pour P2) le feuillage suivant:
feuille(les, artd, m-nn, 2, 0)
feuille(jolies, adj, m-nn, 0, 2),
feuille(jolies, adj, m-an1, 0, 1),
feuille(jolies, adj, m-an2, 2)[2]
feuille(femmes, nom, m-nn, 0, 1),
feuille(femmes, nc, m-an2, 2, 1)[3].

5.2. Déroulement de AF

Le texte est une suite de segment feuillés (une ou plusieurs feuilles associée(s) à chaque

1 On réserve *m-an2* pour le modèle adjectival à droite de *m-nn*.

2 *jolies* est *adj* dans le m-nn dans *les jolies filles*, *adj* dans le m-an1 *très jolies* dans *les très jolies filles* ou *adj* dans le m-an2 dans les exemples *les filles jolies* ou *les filles extrêmement jolies*.

3 fille est *n* dans le m-nn et *n* dans le m-an2 (*les filles très femmes*).

segment).

On désigne par ***liste_modeles*** la liste des modèles ouverts pendant le traitement. Le dernier modèle ouvert est le dernier élément de la liste. Les opérations clé de l'algorithme sont la prise de feuille, la concaténation de feuille et le contrôle de la fermeture d'un modèle avec l'application du principe de la transition diabolique. Nous allons détailler chacune des ces opérations.

5.2.1. Prise de feuille

Lorsque l'on cherche à concaténer une nouvelle unité linguistique, la première opération est d'aller chercher les feuilles qui lui sont associées et que l'on peut considérer pour l'analyse. On désigne par ***ensemble_feuilles*** l'ensemble des feuilles associées à une unité linguistique. On désigne par ***feuilles_candidates*** l'ensemble des feuilles, inclus dans *ensemble_feuilles*, que l'on va conserver pour l'analyse d'une unité linguistique. Enfin, on désigne par ***modele_courant*** le dernier élément de *liste_modeles* qui indique quel est le modèle le plus profond couramment ouvert.

Tant que ***liste_modeles*** non vide
- Garder dans ***feuilles_candidates*** les feuilles de ***ensemble_feuilles*** associées à ***modele_courant*** et qui ne commencent pas toujours ***modele_courant***
- Si on en trouve
  - Eliminer de ***ensemble_feuilles*** les feuilles de même catégorie appartenant à d'autres modèles que ***modele_courant***
- Sinon
  - Garder dans ***feuilles_candidates*** les feuilles de ***ensemble_feuilles*** qui peuvent commencer un modèle inclus dans ***modele_courant***
  - Si on en trouve
  - Eliminer de ***ensemble_feuilles*** les feuilles de même catégorie appartenant à d'autres modèles
- Enlever de ***liste_modeles*** le dernier élément

Si ***liste_modeles*** est vide
- si ***feuilles_candidates*** est vide
  - prendre dans ***feuilles_candidates*** toutes les feuilles de ***ensemble_feuilles*** pouvant commencer un modèle

Exemple : Pour l'unité linguistique *belle* dans la suite *la belle... ensemble_feuilles* contient cinq feuilles, deux comme *n* (acception de *belle* comme partie qui doit départager deux adversaires) dans les modèles *m-nn* et *m-an2*, trois comme *adj* (acception de *belle* dans un sens proche de celui de *jolie*) dans les modèles *m-nn*, *m-an1* et *m-an2*. Ne resteront après la prise de feuille que deux possibilités dans *feuilles_candidates*: feuille nom dans le *m-nn* et feuille adjectif dans le *m-nn*.

5.2.2. Concaténation

On a dans *feuilles_candidates* la (ou les feuilles) contenant de l'information qui n'entre pas en contradiction avec la situation courante (en terme de modèles ouverts). On peut alors procéder à une simple concaténation de la(les) feuille(s) courante(s), avec éventuellement à sa gauche l'ouverture d'un ou de plusieurs modèles (en fonction de la profondeur). Dans notre exemple, après concaténation, deux embryons d'analyse sont possibles :

(mn (m-nn la belle(nom)

et

(mn (m-nn la belle(adj)

5.2.3. Contrôle de la fermeture

Si la feuille que l'on vient de concaténer ferme toujours
- On pose à la droite de cette feuille une balise fermante correspondant à ***modele_courant*** et on met à jour ***liste_modeles***

Sinon
- Si la feuille que l'on vient de concaténer ne ferme jamais
  - On passe à l'unité linguistique suivante
- Sinon
  - On va chercher les feuilles placées à la droite de la feuille courante pour déterminer si oui ou non on doit fermer le modèle (transition diabolique)

Dans l'exemple précédant, une des analyses devient :

(mn (m-nn la belle(nom))

car la feuille associée à la catégorie nom du *m-nn* ferme toujours ce modèle. Pour l'autre analyse, où *belle* est *adj*, on va appliquer la loi de la transition diabolique, car l'*adj* peut ou non fermer un *m-nn*.

5.2.4. Transition diabolique

La feuille courante ferme parfois un modèle. Il faut donc examiner ce qui se passe à droite de cette feuille courante pour savoir si l'on doit fermer ou non le dernier modèle ouvert.

Si l'une des feuilles suivantes est associée à ***modele_courant*** ou commence un modèle inclus dans ***modele_courant***
- On ne ferme pas ***modele_courant*** à la droite de la feuille courante et on continue l'analyse avec cette nouvelle feuille

Sinon
- Si une des feuilles suivante peut commencer un modèle qui est inclus dans un des modèles présents dans ***liste_modeles*** mais différent de ***modele_courant***
  - On ferme le modèle courant à la droite de la feuille courante (autant de fois qu'il est nécessaire selon la profondeur) et on

	continue l'analyse avec cette nouvelle feuille
Sinon	
	On ferme tous les modèles ouverts présents dans ***liste_modele*** et on continue l'analyse

Si à la droite de *belle* on a le mot *fille*, le principe de la transition diabolique nous dit que l'on ne peut pas fermer le modèle à la droite de *belle*, l'unité linguistique *fille* pouvant étant rattachée au modèle couramment ouvert. Si par contre à la droite de *belle* on a l'unité linguistique *de* rattachée à un syntagme prépositionnel noyau qui peut apparaître à l'intérieur d'un *m-n*, on ferme le *m-nn* seul et on continue l'analyse (et lors de la concaténation, on ouvrira ultérieurement un modèle prépositionnel noyau à la gauche de *de*). Enfin si l'unité linguistique à la droite de *belle* est *mange* appartenant à un modèle de phrase verbale noyau qui n'est inclus ni dans le modèle courant (*m-nn*), ni dans un modèle supérieur ouvert (*m-n*), alors on va fermer d'abord le plus profond (*m-nn*) puis le modèle supérieur (*m-n*) et on continue l'analyse.

5.3. La sortie de l'algorithme

A la sortie de l'algorithme, le texte initial est balisé par un parenthésage indiquant les *m-nn*, les *m-an* et les *m-n* construits à partir d'une combinaison de ces deux types de modèles noyaux. Le texte initial est étiqueté, mais non desambigué, le texte de sortie pourra présenter également des ambiguïtés[4].

6. Conclusion

Un algorithme de balisage des *sn* dans un texte, s'inscrivant dans l'approche 5P a été présenté. On peut noter que pour élargir l'analyse à la phrase, cet algorithme peut être également utilisé. Il suffit pour cela d'élargir les descriptions linguistiques de P2 à d'autres domaines linguistiques (celui du verbe). Notre approche est actuellement utilisée pour le portugais (description et analyse du *sn*). Des travaux sur l'analyse du français, s'inscrivant dans le même cadre méthodologique sont également en cours (cf. [2]).

Réferences

[1] Ait-Mokhtar S., *L'analyse présyntaxique en une seule étape*. Thèse de doctorat. Université Blaise Pascal, GRIL, 1998.

[2] Bès G. G., *La phrase verbale noyau* in Recherches sur le français parlé, 15, Université de Provence, 1998.

[3] Bès G.G., Blache P., Hagège C. *The 5P paradigm*, Rapport de Recherche GRIL/LPL, 1999.

[4] Blache P., Bès G., *Propriétés et analyse d'un langage*. soumis TALN 99, Cargèse.

[5] Chomsky N., Miller G. A. "Introduction to the formal analysis of natural languages". In *Handbook of Mathematical Psychologyy, v.II.* Duncan Luce, R, Robert R. Bush & Eugene Galanter [eds] New Nork & Londres, John Wiley and Sons, pp. 269-321, 1963.

[6] Hagège C., Bès G.G., «Da observação de propriedades linguísticas à sua formalização numa gramática do processamento da língua». In *Anais do III Encontro para o Processamento Computacional do Português Escrito e Falado (PROPOR'98),* Porto Alegre, RS, Brasil, pp. 23-30, 1998.

[7] Site 5P : http ://www.iltec.pt/~cah/5P.html

4 Dans l'exemple présenté, il y aura ambiguité pour l'analyse de *la belle mange* que seule l'introduction d'information sémantique permettrait de lever.