

Désambiguisation lexicale à l'aide d'automates de polarités

Guy Perrier

13 mai 2008

1 Conditions d'application : grammaires lexicalisées polarisables

Le travail que nous présentons ici est une poursuite de l'étude menée sur la désambiguisation lexicale à l'aide d'automates de polarités par [BGP04]. Il s'applique aux formalismes grammaticaux complètement lexicalisés, où les structures syntaxiques sont représentées par des arbres, qui peuvent être complètement ou seulement partiellement spécifiés ; ces arbres peuvent être des arbres syntagmatiques ou des arbres de dépendance.

Une grammaire est définie par la donnée de ses arbres élémentaires et, comme elle est lexicalisée, chaque arbre élémentaire est associé à un mot et il est muni d'une feuille particulière, l'ancre principale¹, qui représente la position dans cet arbre du mot associé.

Dans ces arbres, une relation définit un ordre partiel entre les nœuds, qui correspond à l'ordre des mots dans la phrase.

Parmi les formalismes qui remplissent ces conditions, il y a les LTAG, les Grammaires d'Interaction (IG) [Per03] et les Grammaires de Dépendance Lexicalisées (LDG). Il n'existe pas à proprement parler de grammaires de dépendances totalement lexicalisées mais c'est tout à fait concevable et certains formalismes s'en rapprochent [DDK04].

En plus, les grammaires doivent être polarisables, c'est-à-dire que les structures syntaxiques doivent pouvoir être équipées de façon systématique de polarités, exprimant leur état de saturation. Le système de polarités est défini par le choix d'un ensemble de polarités avec leur table de combinaison. On suppose que l'opération de combinaison est toujours commutative et associative. Pour un système de polarités donné, certaines polarités sont distinguées comme polarités saturées.

Une fois une grammaire polarisée, la composition syntaxique peut être exprimée comme une superposition partielle d'arbres contrôlée par les polarités qui vont chercher à se combiner pour atteindre la saturation.

¹On pourrait envisager une ou plusieurs co-ancres pour des items lexicaux multi-mots ou représentant des expressions figées, mais dans le reste de l'article nous considérerons que tout arbre élémentaire a seulement une ancre principale mais pas de co-ancre.

[Kah06] a montré comment polariser les formalismes grammaticaux les plus courants en choisissant des systèmes de polarités plus ou moins sophistiqués.

Dans la suite de l'article, quand nous considérerons un système de polarités, nous supposons que le système comporte au moins une polarité positive et une polarité négative qui ne sont pas saturées et qui, en se combinant donnent une polarité saturée. Nous supposons aussi que deux polarités de même signe ne peuvent pas se combiner.

2 Méthode de désambiguisation

Pour analyser une phrase avec une grammaire lexicalisée, la première phase consiste à sélectionner un arbre dans le lexique pour chaque mot de la phrase. Dans la suite de l'article, nous appellerons *sélection lexicale* l'association de chaque mot de la phrase avec une entrée du lexique. Le nombre de sélections lexicales possibles est exponentiel relativement à la longueur de la phrase. La désambiguisation va consister à éliminer un maximum de sélections lexicales dont on peut déterminer a priori qu'elles ne mènent pas à une analyse qui réussit, tout en conservant toutes celles qui sont susceptibles d'y mener.

La méthode se situe dans le droit fil de celle utilisée par [Bou03] et son principe général a été présenté dans [BGP04]. Elle consiste à abstraire la grammaire puis à faire l'analyse avec la grammaire abstraite. Toute analyse qui échoue avec la grammaire abstraite échoue avec la grammaire initiale. De cette façon, on élimine certaines sélections lexicales qui mènent à l'échec. Bien entendu, l'intérêt est que le filtrage ne soit pas trop coûteux en calculs tout en éliminant un maximum de sélections menant à l'échec. Plus on abstrait, moins le filtrage est coûteux mais moins aussi il est efficace donc il y a un équilibre à trouver entre le coût et l'efficacité du filtrage.

3 Le filtrage par polarités

Dans [BGP04], l'abstraction de la grammaire initiale préalablement polarisée consiste à ne conserver pour les structures syntaxiques que les polarités positives et négatives avec les données auxquelles elles s'appliquent (catégories grammaticales pour les TAG, traits morpho-syntaxiques pour les IG). On supposera, sans perte de généralité, que ces données élémentaires sont dans tous les cas des *traits polarisés*, c'est-à-dire des triplets (nom de trait, polarité + ou -, valeur de trait). Pour simplifier, on supposera qu'à chaque nom de trait est associé un domaine de valeurs et qu'une valeur de trait est un élément particulier de ce domaine². Une entrée lexicale de la grammaire abstraite est donc un multi-ensemble de traits polarisés et une analyse syntaxique dans ce cadre consiste à coupler les traits

²Dans cet article, on ne considère pas la possibilité pour une valeur d'être une disjonction d'éléments du domaine. Si nous voulions prendre en compte cette possibilité, cela ne poserait pas de problème mais il faudrait comme dans [BGP04] utiliser des intervalles de polarités.

<i>position de (f, p, v) dans T₁ par rapport à w₁</i>	<i>position de (f, -p, v) dans T₂ par rapport à w₂</i>	<i>interaction possible ou pas</i>
<i>L</i>	<i>L</i>	<i>oui</i>
<i>L</i>	<i>R</i>	<i>non</i>
<i>L</i>	<i>S</i>	<i>non</i>
<i>R</i>	<i>L</i>	<i>oui</i>
<i>R</i>	<i>R</i>	<i>oui</i>
<i>R</i>	<i>S</i>	<i>oui</i>
<i>S</i>	<i>L</i>	<i>oui</i>
<i>S</i>	<i>R</i>	<i>non</i>
<i>S</i>	<i>S</i>	<i>oui</i>

TAB. 1 – possibilité d’interaction entre polarités opposées en fonction de leur position

polarisés par paires de traits duaux. L’analyse réussit lorsqu’il existe un tel couplage qui ne laisse de côté aucun trait et qui n’utilise pas deux fois le même trait.

L’inconvénient de cette abstraction est qu’elle ignore totalement l’ordre des mots dans la phrase : si une analyse réussit, elle réussira aussi si on permute les mots de la phrase. L’abstraction ne prend en compte que la valence des mots.

Pour pallier cet inconvénient, nous proposons une abstraction moins radicale : pour chaque entrée lexicale, nous conserverons ses traits positifs et négatifs avec leur position par rapport à l’ancre de l’arbre. C’est la relation d’ordre linéaire entre les nœuds de l’arbre qui va permettre de déterminer cette position. Il y a trois positions possibles : à gauche de l’ancre (notée L pour left), à droite (notée R pour right) ou sur l’arête, qui est le chemin menant de la racine à l’ancre (position notée S pour spine). Bien entendu, cette position n’est pas toujours déterminée puisque l’arbre peut être sous-spécifié.

L’intérêt d’ajouter la position de chaque trait polarisé par rapport à l’ancre est qu’on introduit de cette façon des contraintes qui vont permettre de filtrer davantage en prenant en compte l’ordre des mots. Le principe de base est le suivant :

Considérons l’analyse d’une phrase et une sélection lexicale pour cette phrase avec deux mots w_1 et w_2 de la phrase associés respectivement aux arbres T_1 et T_2 du lexique. On suppose que w_1 précède w_2 dans la phrase. Le tableau 1 nous indique dans ces conditions la possibilité pour un trait (f, p, v) de T_1 de se saturer en se combinant avec un trait $(f, -p, v)$ de T_2 selon leur position par rapport à l’ancre principale dans leur arbre respectif.

L’abstraction considérée maintenant va être paramétrée par un couple (f, v) d’un trait et d’une valeur de ce trait. Elle consiste à conserver pour chaque arbre T du lexique initial un enregistrement $Abs_{(f,v)}(T) = \{lneg : x_1, lpos : x_2, rneg : x_3, rpos : x_4, sneg : x_5, spos : x_6\}$

indiquant pour chaque position possible (L, R, S) et pour chaque polarité (pos, neg) le nombre de traits existant dans T qui ont comme nom f et comme valeur v . Par exemple, x_1 représente le nombre de traits $(f, -, v)$ dans T situés à gauche de l'ancre principale.

Avec cette abstraction, voyons ce que devient une analyse. Considérons une phrase à analyser $w_1, w_2 \dots w_n$ ³ et une sélection lexicale possible pour cette phrase $T_1, T_2, \dots T_n$. Dans la grammaire abstraite, la sélection se réduit à : $Abs_{(f,v)}(T_1), Abs_{(f,v)}(T_2), \dots Abs_{(f,v)}(T_n)$. Si on analyse la phrase de gauche à droite, on a besoin de mémoriser la position courante i dans la phrase et l'état courant de l'analyse $S_i = \{lneg : x_1^i, lpos : x_2^i, rneg : x_3^i, rpos : x_4^i, sneg : x_5^i, spos : x_6^i\}$ se résume à un comptage des traits polarisés (f, p, v) non encore neutralisés avec leur position par rapport à une des ancrs principales associées à $w_1, w_2 \dots w_i$. Par exemple, x_1^i représente le nombre de traits $(f, -, v)$ non encore neutralisés, situés à gauche d'une ancre principale associée à $w_1, w_2 \dots w_i$.

Or, après observation de plusieurs formalismes, on peut remarquer l'interaction L-L mentionnée à la première ligne du tableau 1 arrive rarement. Elle n'arrive jamais pour une LTAG ou une LDG, polarisées de la façon que nous décrirons dans les sections suivantes. Cette interaction peut se produire pour une IG, comme nous le verrons à la section 7. On peut énoncer l'absence d'une telle interaction sous forme d'un postulat :

Postulat d'interaction gauche-gauche : *Lorsque deux arbres élémentaires interagissent par neutralisation de deux traits duaux, ces deux traits ne sont jamais tous les deux à gauche de leur ancre respective.*

La conséquence de ce postulat est que, dans une analyse gauche droite, à l'étape i , si l'arbre T_i fait apparaître des traits positifs ou négatifs à gauche de son ancre, ils doivent être immédiatement neutralisés par des traits duaux présents dans la structure courante d'analyse. Il est donc inutile de conserver dans S_i les deux champs $lneg$ et $lpos$, car ils doivent toujours être à zéro.

Dans la suite de cette section, nous allons décrire l'algorithme de filtrage pour une grammaire vérifiant le postulat d'interaction gauche-gauche. Dans le cas général, l'algorithme est un peu plus compliqué, car il faut conserver en mémoire les champs $lneg$ et $lpos$.

On initialise le filtrage en mettant la position courante i dans la phrase à 0 et l'état S_0 à $\{rneg : 0, rpos : 0, sneg : 0, spos : 0\}$. De façon indéterministe, on va itérer n fois l'opération élémentaire qui est représentée par le schéma de la figure 1. Sur ce schéma, on se place à la $i^{\text{ième}}$ itération et l'opération est alors décrite ainsi :

- On considère le prochain élément $Abs_{(f,v)}(T_i)$ de la sélection. On commence par effectuer des neutralisations entre polarités au sein de la structure syntaxique T_i qui se situent d'un même côté relativement à l'ancre. Cette opération n'est pas déterministe car on peut choisir 3 paramètres qui sont des entiers naturels : x, y et z , qui doivent être respectivement inférieurs ou égaux à $\min(a, b)$, $\min(c, d)$ et $\min(e, f)$.

³On suppose, pour simplifier, que le découpage de la phrase en mots est unique, un mot étant une clé d'entrée dans le lexique.

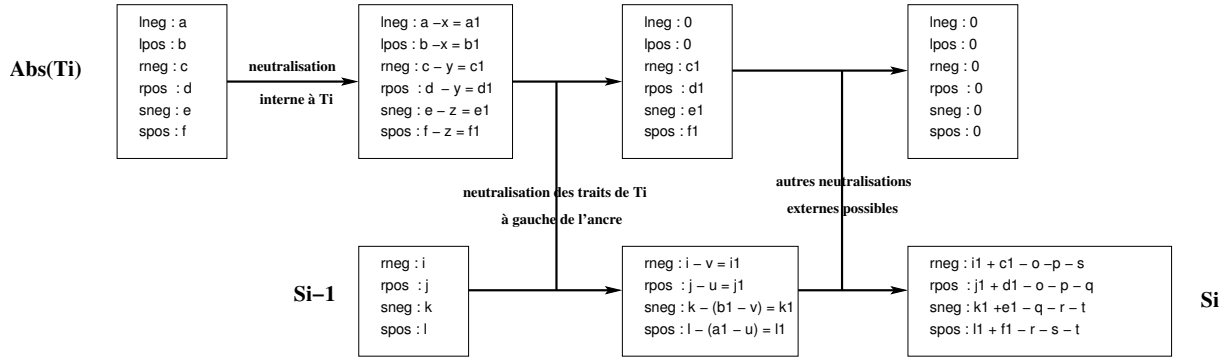


FIG. 1 – Opération élémentaire de filtrage

- Suivant le postulat énoncé précédemment, on cherche ensuite à neutraliser les traits à gauche de l'ancre de $T(i)$. Cette opération est encore indéterministe. Elle dépend de deux paramètres entiers naturels, u et v qui doivent être inférieurs ou égaux respectivement à $\min(i, b_1)$ et $\min(j, a_1)$.
- On effectue ensuite toutes les autres neutralisations en respectant les contraintes données par le tableau 1. Cette dernière opération est aussi indéterministe et elle dépend de 4 paramètres entiers naturels p , q , r et s qui vérifient les contraintes suivantes : $p + s \leq i_1$, $o + q \leq j_1$, $t \leq k_1$, $r \leq l_1$, $o \leq c_1$, $p \leq d_1$, $q + r \leq e_1$, $s + t \leq f_1$. Le filtrage réussit si on arrive après la $n^{\text{ième}}$ itération à $S_n = \{rneg : 0, rpos : 0, sneg : 0, spos : 0\}$.

4 Utilisation d'automates pour factoriser l'information

Le nombre de sélections lexicales possibles pour une phrase est exponentiel relativement à la longueur de cette phrase. Même si la grammaire abstraite facilite l'analyse, il serait quand même très coûteux de pratiquer le filtrage sélection après sélection. En plus, le coût est augmenté par l'indéterminisme inhérent à l'opération de base du filtrage, telle qu'elle est décrite dans la section précédente.

L'idée, reprise de [BGP04], consiste à factoriser sous forme d'un automate l'information qu'il est nécessaire de conserver quand on fait une analyse de gauche à droite. L'automate a une structure particulière car les états y sont rangés par couche, une couche correspondant à une position de lecture dans la phrase. Un état S_i de la couche i représente un état d'analyse partielle des i premiers mots de la phrase. Si on se place dans le cadre d'une grammaire vérifiant le postulat d'interaction gauche-gauche, S_i est codé par le quadruplet $\{rneg : x_1^i, rpos : x_2^i, sneg : x_3^i, spos : x_4^i\}$, tel qu'il a été défini à la section précédente. Une transition de l'automate d'un état S_{i-1} à un état S_i représente la combinaison d'un arbre T_i

avec les arbres T_0, T_1, \dots, T_{i-1} , tels qu'ils ont déjà été combinés et tels que cette combinaison est décrite par S_{i-1} . Après combinaison avec l'arbre T_i , les arbres $T_0, T_1, \dots, T_{i-1}, T_i$ se retrouvent dans un état de combinaison décrit par S_i .

Un tel automate n'est pas en général très gros. Si nous supposons de façon raisonnable que chacun des champs des quadruplets est borné par la valeur 2, le nombre maximum d'états par couche est de 3^4 donc le nombre maximum d'états de l'automate est de $81n$. En plus, il ne faut conserver dans cet état que les chemins qui mènent à l'état $S - n = \{rneg : 0, rpos : 0, sneg : 0, spos : 0\}$. de la couche n donc l'automate est en réalité plus petit.

Le problème est qu'il faut construire un tel automate pour chaque couple (f, v) trait-valeur pertinent par rapport à la phrase à analyser et il faut ensuite faire l'intersection de tous les automates obtenus. S'il y a p couples trait-valeur, la taille de l'automate sera au maximum de $81^p n$.

Maintenant, il est intéressant d'étudier s'il n'est pas possible de simplifier les automates en fonction de la façon particulière dont les formalismes grammaticaux sont polarisés et des particularités des grammaires qui sont formalisées.

5 Les grammaires d'arbres adjoints lexicalisées

Une façon simple de polariser le formalisme des grammaires d'arbres adjoints lexicalisées est d'associer un trait $(cat, +, x)$ à toute racine d'un arbre initial de catégorie x et un trait $(cat, -, y)$ à tout nœud de substitution d'un arbre initial de catégorie y . Dans cette polarisation, les arbres auxiliaires et l'adjonction sont complètement neutres.

Compte tenu de cette polarisation, une abstraction d'arbres associée à (cat, v) se réduit à 3 champs :

- *spos* qui prend la valeur 1 ou 0 selon que la racine de l'arbre a comme catégorie v ou pas,
- *lneg* qui représente le nombre de nœuds de substitution à gauche de l'ancre,
- *rneg* qui représente le nombre de nœuds de substitution à droite de l'ancre,

De la même façon, les automates de polarités se simplifient et chaque état ne comportent plus que deux champs :

- *spos* qui représente le nombre de racines d'arbres initiaux de catégorie v non encore accrochées à un nœud de substitution,
- *rneg* qui représente le nombre de nœuds de substitution de catégorie v à droite de leur ancre et encore libres.

Une LTAG vérifie naturellement le postulat d'interaction gauche-gauche et l'algorithme de construction de l'automate de polarités se simplifie aussi de la façon suivante. On initialise la position courante i dans la phrase à 0 et l'état S_0 à $\{rneg : 0, spos : 0\}$. De façon indéterministe, on va itérer n fois l'opération qu'on peut décrire ainsi à la $i^{\text{ème}}$ itération :

- Il faut tout d'abord que $S_{i-1}.spos \geq Abs_{(cat,v)}(T_i).lneg$, sinon le filtrage s'arrête et la sélection est rejetée. On remplace $S_{i-1}.spos$ par $S_{i-1}.spos - Abs_{(cat,v)}(T_i).lneg$.

- Si $Abs_{(cat,v)}(T_i)$ ne comporte pas de trait positif (cela signifie que la racine de T_i n'est pas de catégorie v), on remplace aussi $S_{i-1}.rneg$ par $S_{i-1}.rneg + Abs_{(cat,v)}(T_i).rneg$.
- Si $Abs_{(cat,v)}(T_i)$ comporte un trait positif (cela signifie que la racine de T_i est de catégorie v), on a alors deux possibilités :
 - On remplace $S_{i-1}.rneg$ par $S_{i-1}.rneg + Abs_{(cat,v)}(T_i).rneg$ et $S_{i-1}.spos$ par $S_{i-1}.spos + 1$. Cela signifie que l'on ajoute le nouvel arbre à côté de ceux déjà présents.
 - On remplace $S_{i-1}.rneg$ par $S_{i-1}.rneg + Abs_{(cat,v)}(T_i).rneg - 1$. Cela signifie que l'on accroche le nouvel arbre à un nœud de substitution d'un arbre déjà existant.

Le filtrage réussit si on arrive après la $n^{\text{ième}}$ itération à $S_n = \{rneg : 0, spos : 0\}$.

Les automates qui sont construits en utilisant cet algorithme ne devraient pas être très gros, compte tenu du nombre limité de catégories grammaticales utilisées et en plus il semble raisonnable de mettre une borne maximum sur les valeurs des deux champs associés aux états. Il serait intéressant d'expérimenter cette méthode de filtrage pour confirmer cette hypothèse et étudier l'efficacité du filtrage en termes de pourcentage de sélections lexicales rejetées.

6 Les grammaires de dépendances lexicalisées

Une grammaire de dépendance peut être définie par un ensemble d'arbres de dépendances élémentaires. On peut restreindre ces arbres à une hauteur maximum de 1 et si on considère que la grammaire est lexicalisée, on distingue une des feuilles comme ancre, celle où est attaché le mot auquel est associé l'arbre.

La polarisation d'une telle grammaire consiste tout d'abord à distinguer les arbres associés à des modificateurs des autres. Les premiers ne seront pas polarisés alors que les seconds le seront de la façon suivante :

- L'ancre est munie d'un trait positif $(cat, +, x)$, où x est la catégorie grammaticale de cette ancre.
- Les nœuds dépendants de l'ancre sont munis d'un trait négatif $(cat, -, y)$, où y est leur catégorie grammaticale.

L'abstraction de la grammaire, comme pour les grammaires d'arbres adjoints, aboutit à conserver, pour chaque arbre élémentaire, les trois champs $spos$, $lneg$ et $rneg$. La construction des automates de polarités se fait donc ensuite comme pour les grammaires d'arbres adjoints. Les automates s'interprètent différemment mais cela ne change rien au calcul.

7 Les grammaires d'interaction

Les grammaires d'interaction sont déjà polarisées, mais par rapport aux deux formalismes polarisés qui viennent d'être présentés, la polarisation est plus souple, ce qui complique le filtrage par automates de polarités.

S_{i-1}	$Abs_{(f,v)}(T_i)$	S_i
<i>lneg</i>	<i>lpos</i>	0
<i>lpos</i>	<i>lneg</i>	0
<i>rneg</i>	<i>lpos</i>	0
<i>rneg</i>	<i>rpos</i>	0
<i>rneg</i>	<i>spos</i>	0
<i>rpos</i>	<i>lneg</i>	0
<i>rpos</i>	<i>rneg</i>	0
<i>rpos</i>	<i>sneg</i>	0
<i>sneg</i>	<i>rpos</i>	0
<i>sneg</i>	<i>spos</i>	0
<i>spos</i>	<i>rneg</i>	0
<i>spos</i>	<i>sneg</i>	0
0	X	X
X	0	X

TAB. 2 – Règles de transition dans un automate avec neutralisation au plus proche

Maintenant, il serait intéressant d'étudier concrètement pour différentes langues et pour différentes formalisations de leur grammaire, trait par trait, dans quelle mesure il est possible de simplifier les automates correspondants.

Voyons ce qu'il en est en particulier pour la grammaire du français que nous avons construit [Per07]. Nous la désignerons par IG_F . La grammaire IG_F ne respecte pas le postulat d'interaction gauche-gauche : l'interaction d'un verbe fini avec un pronom clitique sujet donne lieu à une interaction L-L, telle qu'elle est présentée dans le tableau 1. Dans ce cas, il y a présence d'un nœud vide trace d'un sujet déplacé de sa position canonique.

Par ailleurs, l'interaction R-R correspond, pour la grammaire IG_F , soit à un phénomène d'extraction, soit à la cliticisation d'un complément. Dans les deux cas, il y a existence d'un nœud vide trace d'un argument déplacé par rapport à sa position canonique.

D'une façon générale, on remarque que les nœuds vides sont produits par des arbres élémentaires attachés soit à des clitiques, soit à des mots grammaticaux introduisant des extractions. Il devrait donc être possible de formaliser cette propriété dans la construction des automates, pour circonscrire les interactions L-L et R-R. Nous n'irons pas plus loin ici mais cela demanderait une étude approfondie.

Nous allons nous attacher aux couples trait-valeur (f, v) qui vérifient la propriété suivante dans IG_F :

Neutralisation au plus proche : dans une analyse, si deux traits $(f, +, v)$ et $(f, -, v)$, introduits respectivement par les mots w_i et w_j se neutralisent mutuellement, il n'y a pas d'autre trait $(f, +, v)$ ou $(f, -, v)$ introduit par un mot

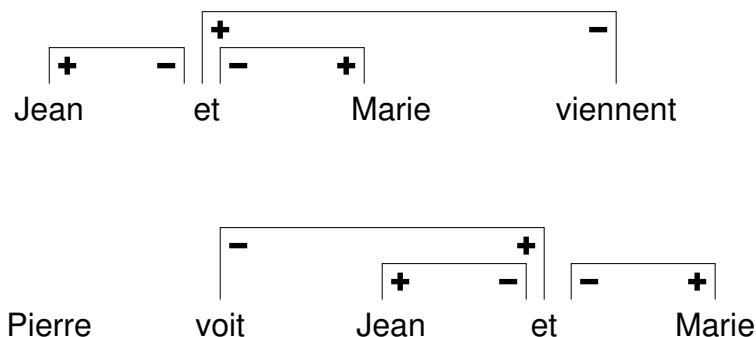


FIG. 2 – Violation du principe de neutralisation au plus proche par la coordination

se situant entre w_i et w_j , w_i et w_j compris.

Cette propriété simplifie beaucoup les automates de filtrage. Si on considère une transition de l'état S_{i-1} à l'état S_i sur une abstraction d'arbre $Abs_{(f,v)}(T_i)$, les 6 champs de chacun des états sont à 0 sauf éventuellement 1 qui est à 1. Il en est de même pour les 6 champs de $Abs_{(f,v)}(T_i)$. Ainsi, les règles de transition sont complètement déterministes. Vous les trouverez dans le tableau 2. Pour lire le tableau, il faut comprendre que la valeur 0 indique que tous les champs sont à 0. Sinon est indiqué le champ qui a la valeur 1. Par ailleurs, la valeur X est l'une quelconque des valeurs possibles. Enfin, lorsqu'un cas n'apparaît pas dans le tableau, cela signifie une absence de transition.

On constate que la majorité des couples trait-valeur respecterait la neutralisation au plus proche, s'il n'y avait pas la coordination. La modélisation de la coordination dans les GI implique en général que les traits non saturés des deux conjoints soient fusionnés en un seul. Cette fusion ne peut pas être réalisée directement mais elle est simulée par l'arbre syntaxique représentant la conjonction de coordination. Supposons que l'on ait à fusionner deux traits $(f, +, v)$. Pour cela, la conjonction de coordination va apporter deux traits opposés $(f, -, v)$ destinés à neutraliser les traits des conjoints et elle va aussi apporter un trait $(f, +, v)$ représentant le trait fusionné.

Dans ces conditions, on comprend que la présence d'une coordination entraîne une violation du principe de neutralisation au plus proche. Considérons les deux exemples donnés par la figure 2, où sont représentés par des arcs les neutralisations des traits (cat, p, np) . Dans ces deux exemples, il y a violation du principe de neutralisation au plus proche. Une façon de rétablir ce principe, est de considérer de façon artificielle qu'un des traits polarisés négatifs apportés par la conjonction de coordination est neutralisé par le trait positif apporté par la même conjonction. C'est ce que montre la figure 3 et cela revient à dire que la conjonction apporte un trait négatif dont la position est sous-spécifiée, à gauche ou à droite de l'ancre.

Une fois faite cette transformation pour la coordination, on constate que l'on peut

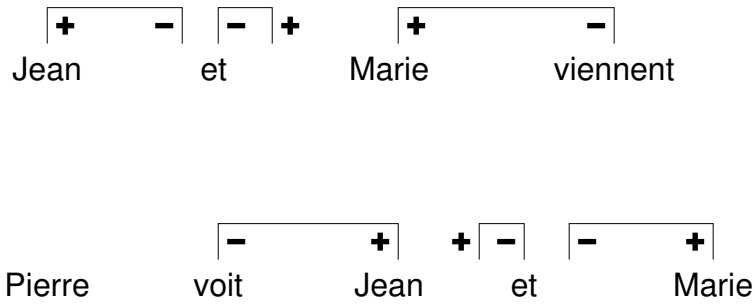


FIG. 3 – Rétablissement du principe de neutralisation au plus proche pour la coordination

nom de trait	valeur de trait
aux	avoir,caus,cop,etre,mod
cat	ap, aux, n
cpl	?
mood	inf, pastp
neg	true
prep	?
reflex	true

TAB. 3 – Traits auxquels devrait s’appliquer le principe de neutralisation au plus proche

appliquer le principe de neutralisation au plus proche à une majorité de couples trait-valeur. Durant son stage de master en cours, Charif Alchikh Haydar a entrepris une étude systématique de la localisation des traits polarisés dans les arbres élémentaires pour la grammaire IG_F . Cette étude permet de conjecturer une liste de couples trait-valeur dont peut prévoir qu’ils respectent le principe de neutralisation au plus proche. Cette liste est donnée par le tableau 3. Un point d’interrogation remplace n’importe quelle valeur du trait correspondant. Bien entendu, il reste à valider plus précisément cette liste.

D’une façon plus générale, il s’agit maintenant d’implémenter ces résultats pour étudier dans quelle mesure ils améliorent le filtrage. Pour les traits auxquels le principe ne s’applique pas, reste à explorer d’autres voies pour prendre en compte l’ordre des mots dans la phrase dans la construction des automates de filtrage par polarités.

Références

- [BGP04] Guillaume Bonfante, Bruno Guillaume, and Guy Perrier. Polarization and abstraction of grammatical formalisms as methods for lexical disambiguation. In *20th*

Conference on Computational Linguistics - CoLing'2004, 2004, pages 303–309, Genève, Suisse, 2004.

- [Bou03] Pierre Boullier. Supertagging : A non-statistical parsing-based approach. In *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT03)*, pages 55–65, Nancy, France, 2003.
- [DDK04] Ralph Debusmann, Denys Duchier, and Geert-Jan M. Kruijff. Extensible dependency grammar : A new methodology. In *Proceedings of the COLING 2004 Workshop on Recent Advances in Dependency Grammar*, Geneva/SUI, 2004.
- [Kah06] Sylvain Kahane. Polarized unification grammars. In *21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 137–144, Sydney, Australia, 2006.
- [Per03] Guy Perrier. *Les grammaires d'interaction*. Habilitation à diriger des recherches, Université Nancy2, 2003.
- [Per07] Guy Perrier. A French Interaction Grammar. In Galia Angelova, Kalina Bontcheva, Ruslan Mitkov, Nicolas Nicolov, and Kiril Simov, editors, *RANLP 2007*, pages 463–467, Borovets Bulgaria, 2007. IPP & BAS & ACL-Bulgaria.