Conférence TALN 2000, Lausanne, 16-18 octobre 2000

Des grammaires formelles pour définir une correspondance

Sylvain KAHANE[1]

Résumé

Dans cet article nous introduisons la notion de grammaire transductive, c'est-à-dire une grammaire formelle définissant une correspondance entre deux familles de structures. L'accent sera mis sur le module syntaxique de la théorie Sens-Texte et sur une famille élémentaire de grammaires de dépendance transductives. Nous nous intéresserons à la comparaison avec les grammaires génératives, ce qui nous amènera à discuter de l'interprétation des modèles génératifs actuels.

1. Introduction

Cet article propose un cadre formel pour écrire des grammaires transductives, c'est-à-dire des grammaires dont la visée première est de définir une correspondance entre deux ensembles de structures mathématiques, par exemple des suites et des arbres ou des arbres et des graphes. Il s'intéresse en particulier à une famille élémentaire de grammaires transductives suite-arbre (c'est-à-dire mettant en correspondance des suites et des arbres) qui constitue un formalisme de base pour les grammaires de dépendance, comme le sont les grammaires de réécriture hors-contexte pour les grammaires syntagmatiques.

Le formalisme présenté ici est volontairement très simple. Il s'agit même de la famille la plus élémentaire de grammaires transductives suite-arbre. Notre objectif n'est pas de proposer un formalisme permettant de couvrir de nombreux phénomènes linguistiques (voir pour cela Mel'čuk & Pertsov 1987 ou Kahane 2000), mais de présenter le cœur de tels formalismes. Nous postulons en effet que toute grammaire transductive suite-arbre est en fait une extension d'une grammaire de la famille présentée ici, de même que toutes les grammaires syntagmatiques génératives développées aujourd'hui sont des extensions des grammaires de réécriture hors-contexte.

En s'intéressant aux grammaires transductives, cet article poursuit deux objectifs. Le premier est de proposer un cadre formel pour l'écriture de modèles Sens-Texte (Mel'čuk 1988, 1997). Bien que d'importants fragments de langues aient été décrit dans le cadre de la théorie Sens-Texte (voir Mel'čuk & Pertsov 1987, Mel'čuk 1988 ou Mel'čuk 1993-2000), la formalisation du modèle n'a jamais été achevée. Diverses formalisations et implémentations ont été proposées sans qu'un cadre théorique s'établisse vraiment (Boyer & Lapalme 1984, Iordanskaja & Polguère 1989, Nasr 1996, Kahane & Mel'čuk 1999).

1. TALaNa/LaTTiCe (Univ. Paris 7) et Univ. Paris 10 - Nanterre.
E-mail : sk@ccr.jussieu.fr. Web : www.linguist.jussieu.fr/∼skahane.
Je remercie chaleureusement Jasmina Milićević, Igor Mel'čuk et Alain Polguère, ainsi que deux relecteurs anonymes, pour leurs commentaires éclairants sur ce travail.

Le deuxième objectif est quasiment de nature épistémologique. Il est maintenant clair, pour toutes les théories linguistiques contemporaines, que l'objectif est de modéliser la correspondance sens-sons ou sens-textes. Néanmoins, la plupart des modèles linguistiques contemporains (HPSG, LFG, TAG, ...) sont développés dans un cadre générativiste, alors que les grammaires génératives ne sont pas précisément dédiées à la description des correspondances. Nous étudierons en détail le lien entre grammaires transductives et grammaires génératives à partir de nos grammaires transductives de base, ceci afin de mieux comprendre la nature des grammaires génératives utilisées pour définir des correspondances. Nous montrerons ainsi que de telles grammaires génératives peuvent être vues comme des spécifications procédurales particulières de grammaires transductives.

2. Grammaires transductives

Soient $\mathcal{S}$ et $\mathcal{S}'$ deux ensembles de structures (graphes, arbres, suites, ...). Une **grammaire transductive**[2] G entre $\mathcal{S}$ et $\mathcal{S}'$ est une grammaire formelle qui met en correspondance les éléments de $\mathcal{S}$ avec les éléments de $\mathcal{S}'$. En tant que grammaire formelle, G possède un ensemble *fini* de règles, appelées **règles de correspondance**. Une règle de correspondance met en correspondance un morceau d'une structure de $\mathcal{S}$ avec un morceau d'une structure de $\mathcal{S}'$. En conséquence, une grammaire transductive définit PLUS qu'une correspondance entre les ensembles de structures $\mathcal{S}$ et $\mathcal{S}'$. En effet, pour chaque couple (S, S') mis en correspondance par G, G définit aussi des partitions de S et S' et une fonction $\phi_{(S,S')}$ entre les morceaux de S et S'. Nous appellerons cela une **supercorrespondance** entre $\mathcal{S}$ et $\mathcal{S}'$. En d'autres termes, la supercorrespondance définie par une grammaire transductive G entre deux ensembles de structures $\mathcal{S}$ et $\mathcal{S}'$ est mathématiquement équivalente à une famille de structures produits $(S, S', \phi_{(S,S')})$, avec $S \in \mathcal{S}$, $S' \in \mathcal{S}'$ et $\phi_{(S,S')}$ une correspondance entre les partitions de S et S'.[3]

La théorie Sens-Texte se formalise naturellement par des grammaires transductives. Cette théorie considère 7 niveaux de représentation : sémantique, syntaxique profond, syntaxique de surface, morphologique profond, morphologique de surface, phonologique profond, phonologique de surface. Une modèle Sens-Texte est donc composé de 6 modules, chacun assurant la correspondance entre deux niveaux adjacents. Gladkij & Mel'čuk 1975 proposent un formalisme pour la correspondance entre arbres, qui peut donc s'appliquer à la correspondance entre les niveaux syntaxique profond et syntaxique de surface. Kahane & Mel'čuk 1999 proposent une grammaire transductive pour la correspondance entre les niveaux sémantique et syntaxique profond, dont les structures de représentation sont respectivement des graphes et des arbres de dépendance ; ils montrent, en particulier, à la différence des précédentes présentations dans le cadre de la théorie Sens-Texte, comment définir la correspondance à partir des règles de correspondance. Dans la suite de cet article nous allons nous concentrer sur la correspondance entre les niveaux syntaxique de surface et morphologique profond, c'est-à-dire à la correspondance entre arbres de dépendance et suites.

2. Le terme *transductif* a été choisi par analogie avec les transducteurs (voir, par ex., Aho & Ullman 1972). Néanmoins la théorie des transducteurs est essentiellement limitée, à notre connaissance, à la correspondance entre suites. De plus, les transducteurs, à la différence des grammaires transductives suite-suite, repose sur une spécification procédurale, la fonction de changement d'état.

3. Le terme *structure produit* désigne en mathématique une structure obtenue en combinant deux structures sur un même ensemble. Par exemple, si S est un arbre et S' une suite et que $\phi_{(S,S')}$ est une bijection entre les nœuds de S et les éléments de S', alors $(S,S',\phi_{(S,S')})$ est équivalent à un arbre linéairement ordonné, c'est-à-dire au produit d'une structure d'arbre et d'une structure d'ordre linéaire sur un même ensemble de nœuds.

3. Un exemple de grammaire transductive syntaxique

Nous nous intéressons maintenant à la correspondance entre un arbre de dépendance (syntaxique de surface) et une suite (morphologique profonde). Les nœuds d'un tel arbre sont étiquetés par des **lexies**[4] et les branches par des **relations syntaxiques** (*suj(et)*, *obj(et)*, *mod(ifieur)*, ...). Une suite morphologique profonde est une suite de lexies. Chaque lexie pointe vers un article de dictionnaire. Nous indiquerons la partie du discours de la lexie quand cela est nécessaire. Toutes les notions introduites seront illustrées par l'exemple trivial suivant :

(1) *Peter eats red beans*

.

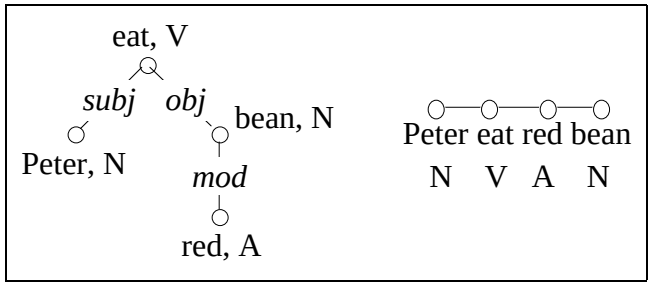

Figure 1. Arbre syntaxique de surface et suite morphologique profonde (simplifiés) de (1)

Nous allons maintenant définir une famille de grammaires transductives syntaxiques que nous appellerons des **grammaires de dépendance atomiques**. Ces grammaires sont atomiques dans le sens où elles mettent en relations uniquement des atomes de structures, c'est-à-dire des nœuds ou des arcs. Les règles sont donc de deux types : les **règles sagittales** (lat. *sagitta* = flèche), qui mettent en relation une dépendance entre deux nœuds avec une relation d'ordre entre deux nœuds, et les **règles nodales**, qui mettent en relation un nœud avec un nœud. Les règles nodales sont ici triviales et n'apparaissent donc pas dans la définition formelle.

Une **grammaire de dépendance atomique** est un quintuplet $G = (\Sigma, \mathcal{C}, \mathcal{R}, \mathcal{O}, \Delta)$, où Σ est l'ensemble des lexies, $\mathcal{C}$ est l'ensemble des categories (grammaticales), $\mathcal{R}$ est l'ensemble des relations syntaxiques, $\mathcal{O}$ est l'ensemble des positions linéaires et Δ est l'ensemble des règles sagittales, à savoir un sous-ensemble de $\mathcal{R} \times \mathcal{O} \times \mathcal{C} \times \mathcal{C}$.

Soit X^* l'ensemble des suites sur X et $\mathcal{T}(X, Y)$ l'ensemble des arbres dont les nœuds sont étiquetés dans X et les branches dans Y. La grammaire G définit une supercorrespondance entre $\mathcal{T}(\Sigma \times \mathcal{C}, \mathcal{R})$ et $(\Sigma \times \mathcal{C})^*$, comme on le verra dans les sections suivantes. Avant cela, nous allons donner un exemple d'une grammaire mettant en correspondance les deux structures de la Fig. 1.

Soit $G_0 = (\Sigma_0, \mathcal{C}_0, \mathcal{R}_0, \mathcal{O}_0, \Delta_0)$ avec :
- $\Sigma_0 = \{\text{Peter}, \text{bean}, \text{eat}, \text{red}\}$;
- $\mathcal{C}_0 = \{V, N, A\}$;[5]

4. En fait, chaque lexie est accompagnée de grammèmes [= valeurs de catégories flexionnelles], mais notre présentation, centrée sur les aspects mathématiques, n'en fera pas mention.

5. L'ensemble des catégories considérées ici est très simple et limité aux parties du discours. Une grammaire à large couverture devra considérer des catégories plus riches. De telles catégories peuvent être exprimées par des structures de traits. Dans ce cas, comme il est fait dans la plupart des formalismes contemporains, celles-ci se combineront par unification plutôt que par identification.

- $\mathcal{R}_0 = \{subj, obj, mod\}$;
- $\mathcal{O}_0 = \{<, >\}$;[6]
- $\Delta_0 = \{(subj, <, V, N), (obj, >, V, N), (mod, <, N, A)\}$.

La règle sagittale $(subj, <, V, N)$ dit que pour chaque couple de lexies X et Y tel que X est un V(erbe) et Y est un N(om), la dépendance syntaxique $X - subj \rightarrow Y$ **correspond** à l'ordre linéaire $Y < X$. En synthèse, cela signifie que pour chaque couple de lexies X et Y tel que X est un V et Y est un N, SI $X - subj \rightarrow Y$, ALORS il est POSSIBLE d'avoir $Y < X$; tandis qu'en analyse, cela signifie que pour chaque couple de lexies X et Y tel que X est un V et Y est un N, SI $Y < X$, ALORS il est POSSIBLE d'avoir $X - subj \rightarrow Y$. Voir Fig. 2 la représentation conventionnelle d'une telle règle dans le cadre de la théorie Sens-Texte. Le symbole $\Leftrightarrow$ la correspondance entre deux morceaux de structure ; $\alpha \Leftrightarrow \beta$ est interprété par $\alpha \Rightarrow \beta$ et $\beta \Rightarrow \alpha$, où $\alpha \Rightarrow \beta$ signifie SI α ALORS β EST POSSIBLE.

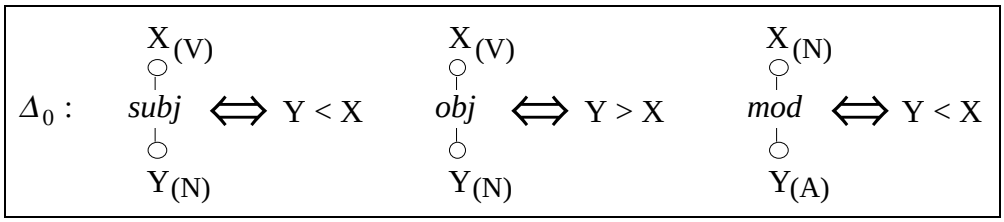

Figure 2. Les règles de Δ_0 dans le style Sens-Texte

4. Des règles de correspondance à la supercorrespondance

Nous allons maintenant voir comment la supercorrespondance peut être définie. Différentes stratégies sont possibles. Nous en verrons plusieurs, dans cette section et dans la suivante.

Une remarque avant : les règles de correspondance, qui se contentent d'associer des morceaux de structure, sont généralement insuffisantes pour encoder toutes les propriétés de la structure produit et certaines **règles globales** doivent être statuées. C'est le cas ici, pour la correspondance syntaxique, où il est nécessaire d'imposer une propriété telle que la projectivité pour les structures produits. La projectivité est une propriété de compatibilité entre un arbre et un ordre linéaire, c'est-à-dire une propriété des arbres linéairement ordonnés : un arbre linéairement ordonné est dit **projectif** si aucun arc ne coupe un autre arc et si aucun arc ne couvre la racine de l'arbre (Lecerf 1961, Iordanskaja 1963, Gladkij 1966).

4.1. Présentation transductive en synthèse

Comme on l'a déjà dit, une grammaire de dépendance atomique $G = (\Sigma, \mathcal{C}, \mathcal{R}, \mathcal{O}, \Delta)$ définit une supercorrespondance entre les arbres de $\mathcal{T}(\Sigma \times \mathcal{C}, \mathcal{R})$ et les suites de $(\Sigma \times \mathcal{C})^*$. En synthèse, elle assure la linéarisation des arbres. La synthèse démarre donc avec un arbre $T \in \mathcal{T}(\Sigma \times \mathcal{C}, \mathcal{R})$. Une dérivation[7] procède comme suit. Pour chaque branche de T

6. Une grammaire de dépendance atomique avec $\mathcal{O}$=$\{<,>\}$ permet seulement de spécifier la position d'un nœud par rapport à son gouverneur, à savoir avant ($<$) ou après ($>$), mais pas la position relative des différents dépendants d'un même nœud. Cette question peut être résolue en attribuant à chaque dépendant, en plus d'une direction, un poids indiquant sa distance au gouverneur (avec, par exemple, $\mathcal{O} \subset \mathbf{Z}$). De telles solutions sont adoptées par Mel'čuk 1967, Courtin & Genthial 1998 ou Kahane 2000.

7. Bien que le processus décrit ici ne soit pas génératif, nous préférons conserver le terme *dérivation* plutôt que d'introduire un terme nouveau.

étiquetée r dont le gouverneur est de catégorie C_1 et le dépendant de catégorie C_2, une règle sagittale $(r, \omega, C_1, C_2) \in \Delta$ est déclenchée et l'étiquette $\omega \in \{<, >\}$ est attachée à la branche. Une suite $s = X_1 \dots X_n \in \Sigma^*$ correspond à T si T a exactement n nœuds, étiquetés $X_1, \dots, X_n$, et si pour chaque dépendance $X - r \rightarrow Y$ de T l'étiquette ω attachée par la règle sagittale est compatible avec l'ordre de X et Y dans s, c'est-à-dire si $Y < X$ quand $\omega = <$ et $Y > X$ quand $\omega = >$. De plus, l'arbre ordonné $T \times s$ doit être projectif. Voir un exemple de dérivation Fig. 3.

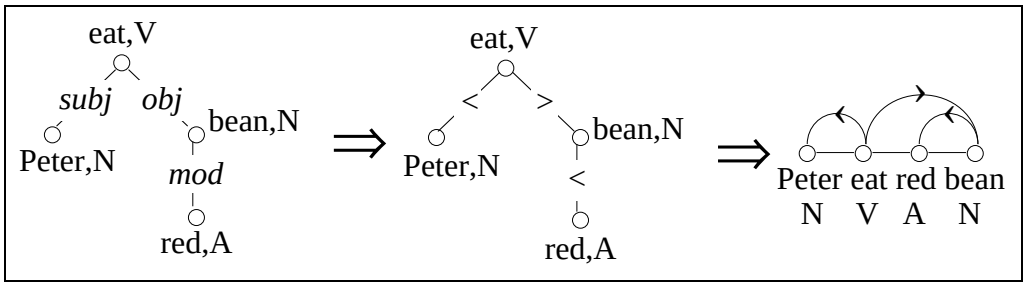

Figure 3. G_0 **utilisée comme grammaire transductive en synthèse**

La dérivation échoue si aucune règle sagittale ne peut être appliquée à l'une des branches de T. Plusieurs ordres peuvent être obtenus pour une même dérivation, notamment si plusieurs nœuds sont placés du même coté de leur gouverneur commun, leur ordre respectif étant, dans ce cas, libre. Pour obtenir toutes les suites correspondant à un arbre T, toutes les combinaisons de règles doivent être essayées.

Le processus proposé ici est purement déclaratif, aucun n'ordre n'étant proposé pour l'application des différentes règles. En fait, cet ordre est libre et la dérivation peut aussi bien s'effectuer top-down, à l'instar des grammaires de réécriture hors-contexte, ou incrémentalement, en suivant l'ordre linéaire des nœuds (cf. Section 5).

4.2. Présentation transductive en analyse

L'analyse démarre avec une suite $s = X_1 \dots X_n \in \Sigma^*$. Une dérivation procède comme suit. Pour chaque couple de nœuds (X, X') de catégories (C, C') avec $X < X'$, une règle sagittale $(r, >, C, C')$ ou $(r, <, C', C) \in \Delta$ est déclenchée et un arc étiqueté r de X à X' ou de X' à X est introduit. Un arbre T correspond à s si T a exactement n nœuds étiquetés $X_1, \dots, X_n$ correspondant aux nœuds de s et si chaque branche de T correspond à l'un des arcs ajoutés à s de même extrémités et de même étiquette. De plus, l'arbre linéairement ordonné T doit être projectif. Voir un exemple de dérivation Fig. 4.

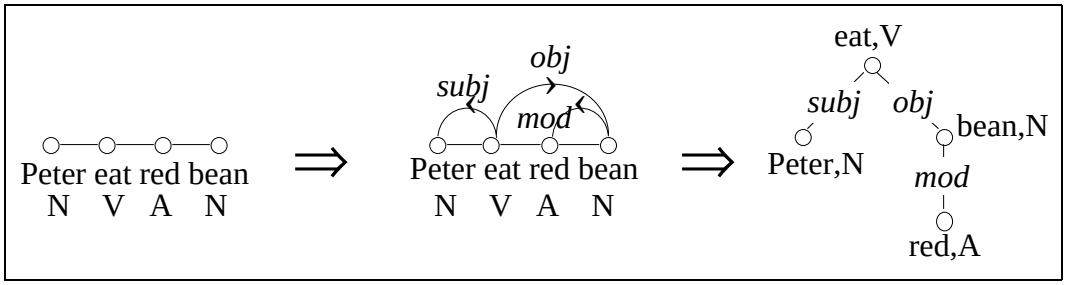

Figure 4. G_0 **utilisée comme grammaire transductive en analyse**

Notons que la dérivation ne nécessite pas, évidemment, qu'une règle sagittale puisse être déclenchée pour chaque couple de nœuds de s. Inversement, même si plus de

$n-1$ règles sont déclenchées, il n'est pas sur qu'un arbre puisse être extrait du graphe obtenu par application des règles de s. Différentes techniques algorithmiques permettent d'extraire un arbre projectif d'un tel graphe ; voir, par exemple, Arnola 1998 ou Blache 1998[8].

Le processus que nous venons de présenter est encore purement déclaratif. Il est possible de contraindre davantage la dérivation. On peut, par exemple, chercher à produire l'arbre en parcourant la suite de gauche à droite. La grammaire se comporte alors comme un automate à pile. Voyons ce qu'il en est plus précisément. A une grammaire de dépendance atomique $G = (\Sigma, \mathcal{C}, \mathcal{R}, \mathcal{O}, \Delta)$, on associe l'automate à pile $M = (\Sigma, \mathcal{Z}, \Lambda)$ avec l'alphabet de pile $\mathcal{Z} = \mathcal{C} \times \{+, -\} \times \mathbf{N}$ et l'ensemble de transitions $\Lambda \subset \mathcal{U} \times \mathcal{Z}^* \times \mathcal{Z}^* \times \mathcal{W}$, où $\mathcal{U} = \{\varepsilon\} \cup (\Sigma \times \mathcal{C})$ et $\mathcal{W} = (\Sigma \times \mathcal{C} \times \mathbf{N}) \cup (\mathbf{N} \times \mathbf{N} \times \mathcal{R}))$. Un arbre dans $\mathcal{T}(\Sigma \times \mathcal{C}, \mathcal{R})$ peut être encodé comme un sous-ensemble de $\mathcal{W}$: un triplet $(X, C, i) \in \Sigma \times \mathcal{C} \times \mathbf{N}$ signifie que le i-ième nœud de l'arbre est étiqueté (X, C) et un triplet $(i, j, r) \in \mathbf{N} \times \mathbf{N} \times \mathcal{R}$ que le i-ième nœud gouverne le j-ième par une dépendance d'étiquette r.

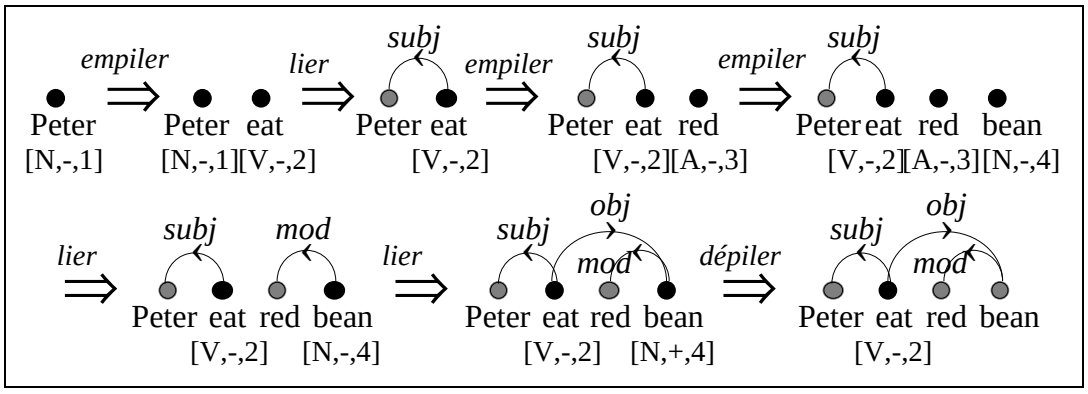

Figure 5. G_0 **utilisée comme un automate à pile**

L'idée est de mettre dans la pile les nœuds au fur et à mesure qu'ils sont lus et de retirer de la pile les nœuds qui ne peuvent plus avoir de liens avec des nœuds à venir (voir Kornai & Tuza 1992 pour une idée similaire). Les signes + et − des symboles de pile indiquent si le nœud qui est dans la pile a ou non déjà reçu un gouverneur. Une transition $\lambda = (x, \alpha, \beta, y) \in \Lambda$ est utilisée comme suit : x est l'élément de la suite lu ($x = \varepsilon$ si rien n'est lu), α est retiré de la pile, β est mis dans la pile et y est produit. Λ est construit comme suit :

- à chaque couple $(X, C) \in \Sigma \times \mathcal{C}$ correspond la **transition d'empilement** $((X, C), \varepsilon, [C, -, i], (X, C, i))$ lisant le i-ième nœud (X, C) de la suite, mettant dans la pile $[C, -, i]$ et produisant le nœud de l'arbre (X, C, i) ;

- à chaque règle sagittale $(r, <, C_1, C_2) \in \Delta$ correspond la **transition de liaison à un dépendant à gauche** $(\varepsilon, [C_2, -, i][C_1, -, j], [C_1, -, j], (j, i, r))$ produisant la dépendance (j, i, r) et retirant $[C_2, -, i]$ de la pile, car, en raison de la projectivité, le nœud (X_2, C_2, i) ne peut être lié à un nœud à la droite de son gouverneur (X_1, C_1, j) ;

- à chaque règle sagittale $(r, >, C_1, C_2) \in \Delta$ correspond la **transition de liaison à un gouverneur à gauche** $(\varepsilon, [C_1, \pm, i][C_2, -, j], [C_1, \pm, i][C_2, +, j], (j, i, r))$ produisant

8. La grammaire considérée par Blache 1998 est une grammaire de réécriture syntagmatique, mais l'utilisation qui en est fait ramène à une situation identique. A noter que l'auteur s'intéresse également à la manière de décrire en une représentation compacte tous les arbres associés à une même suite.

la dépendance (j, i, r) et remplaçant $[C_2, -, j]$ par $[C_2, +, j]$ dans la pile, car le nœud (X_2, C_2, j) est maintenant gouverné par (X_1, C_1, i) ;

- enfin, à chaque $C \in \mathcal{C}$ correspond une **transition de dépilement** $(\varepsilon, [C, +, i], \varepsilon, \varepsilon)$ retirant $[C, +, i]$ de la pile, ce qui est possible, puisque le i-ième nœud est gouverné.

La pile est vide au début. Le processus ne peut s'arrêter que quand le contenu de la pile est réduit à une case $[C, -, i]$. Ceci assure que seul un nœud n'est pas gouverné, à savoir le i-ième nœud. On peut vérifier qu'un tel automate assure qu'on a bien construit un arbre projectif. La Fig. 5 présente l'analyse de (1) par l'automate associé à la grammaire G_0. Un nœud noir représente un nœud dans la pile et un nœud gris un nœud qui a été retiré de la pile. Notons que les indices i dans les symboles de pile ne sont utiles que pour la production de l'arbre correspondant à la suite lue.

5. Grammaires transductives et grammaires génératives

Les grammaires de dépendance atomiques peuvent être utilisées comme des grammaires génératives pour générer la supercorrespondance, c'est-à-dire l'ensemble des structures produit — les arbres linéairement ordonnés. Il y a néanmoins une différence fondamentale entre les grammaires proposées ici et les grammaires génératives usuellement utilisées en linguistique. Cette différence n'est pas tant d'ordre formel que d'ordre théorique : les grammaires de dépendance atomiques ont été définies pour assurer une correspondance entre des arbres syntaxiques de surface et des suites morphologiques profondes, c'est-à-dire qu'elles permettent d'associer à un arbre bien formé toutes les suites bien formées qui lui correspondent ou à l'inverse d'associer à un suite bien formée tous les arbres bien formés qui lui correspondent. Mais rien n'exclut que la grammaire ne puisse s'appliquer à un arbre mal formé et le mettre en correspondance avec des suites mal ou bien formées ou l'inverse. En d'autres termes, une grammaire transductive n'a pas mission d'assurer la bonne formation des structures qu'elle met en correspondance. Par conséquent, si on utilise celle-ci comme une grammaire générative, en générant tous les couples de structures qu'elle est susceptible de mettre en correspondance, il faut s'attendre à ce qu'il y ait parmi ces couples de structures des couples comportant une ou deux structures mal formées.

Revenons à la théorie Sens-Texte pour illustrer ce point et plaçons nous dans le sens de la synthèse. Le module syntaxique de surface, auquel nous nous sommes intéressés jusque-là, n'a pas d'autres ambitions que d'assurer la linéarisation d'un arbre syntaxique de surface. La bonne formation des arbres est elle assurée par les modules qui le précèdent dans la synthèse, c'est-à-dire les modules sémantique et syntaxique profond. Ce sont ces modules qui assurent en particulier que chaque lexie possède une sous-catégorisation convenable et par exemple qu'un verbe fini soit accompagné d'un et un seul sujet. Rien dans le module syntaxique de surface d'un modèle Sens-Texte, ni dans la grammaire de dépendance atomique G_0 proposée en exemple, n'exclut qu'un arbre avec un nœud verbal ayant deux sujets soit linéarisé. A l'inverse, dans le sens de l'analyse, rien n'exclut qu'une suite morphologique profonde, même bien formée, soit associée à un arbre où un verbe possède deux sujets. Ceci n'est pas un problème car un tel arbre ne pourra jamais correspondre à une structure sémantique. Le module syntaxique n'est qu'un maillon de la correspondance sens-son ou son-sens.

Voyons maintenant comment les grammaires de dépendance atomiques peuvent être utilisées comme des grammaires génératives. Chaque règle est vue comme un morceau d'arbre linéairement ordonné. Les règles sagittales introduisent des dépendances ordonnées. Les règles nodales, qui jusqu'à maintenant n'avaient pas été considérées car

elles mettent trivialement en correspondance un nœud de l'arbre avec un nœud de la suite de même étiquette, vont introduire les nœuds. Nous utilisons les conventions de représentation de Nasr 1996 : les éléments, nœuds ou arcs, qui sont introduits par les règles sont en noir et les requêtes sont en blanc. Pour utiliser le vocabulaire des grammaires de réécriture, on peut dire que les éléments noirs sont terminaux et les éléments blancs non terminaux. Les règles sont combinées par unification des nœuds. Deux éléments noirs ne peuvent s'unifier, un élément noir et un blanc donnent un noir et deux blancs un blanc. La structure finale doit être entièrement noire. Une dérivation consiste simplement à générer un ensemble de règles et à les combiner, la structure obtenue devant être un arbre linéairement ordonné projectif.

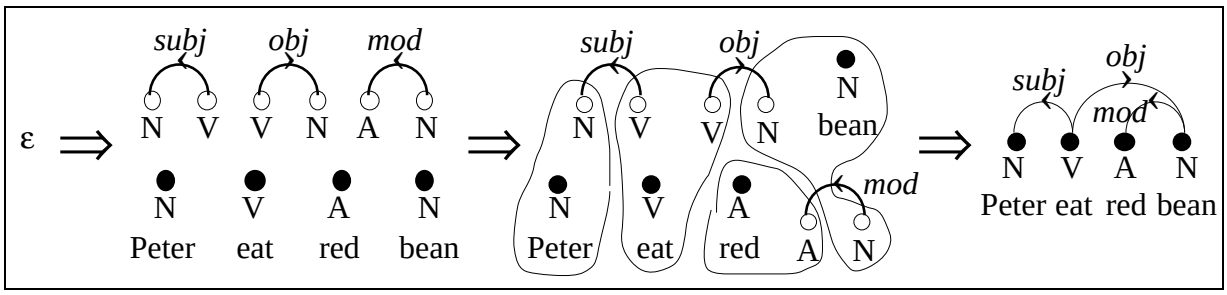

Figure 6. Une dérivation pour G_0 vue comme une grammaire générative

Bien que cela ne soit pas nécessaire, il est possible de spécifier un ordre sur les opérations de dérivation. Une dérivation guidée par la structure d'arbre et procédant top-down s'apparentera à une grammaire de réécriture (Fig. 7). Cf. Dikovsky & Modina 2000 pour des grammaires de dépendance de ce type.

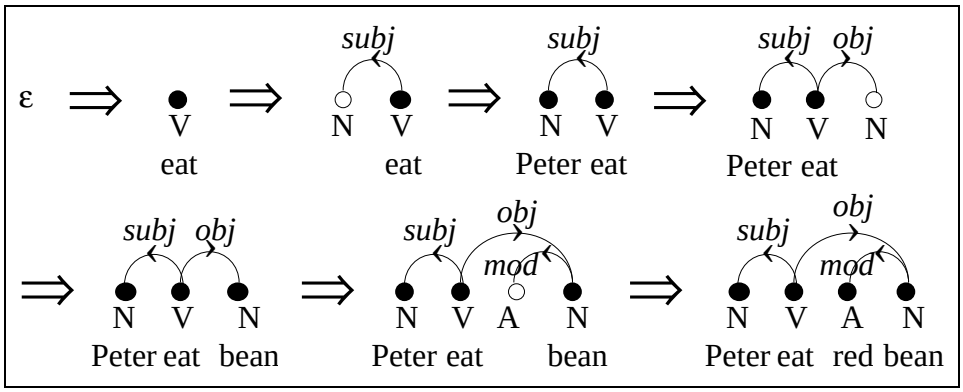

Figure 7. Une dérivation top-down pour G_0

On peut d'ailleurs rendre cette apparente similitude explicite en traduisant une grammaire de dépendance atomique en une grammaire de réécriture hors-contexte. Ainsi la grammaire G_0 donne la grammaire $G'_0 = (\Sigma_T, \Sigma_{NT}, \bar{V}, \mathcal{R})$ avec l'alphabet terminal $\Sigma_T = \Sigma_0 \cup \mathcal{R}_0 \cup \{(,),:\}$, l'alphabet non terminal $\Sigma_{NT} = \bar{\mathcal{C}}_0 = \{\bar{V}, \bar{N}, \bar{A}\}$ et l'ensemble des règles de réécriture :

$$\mathcal{R} = \left\{ \begin{array}{llll} \bar{V} \rightarrow (\bar{N} : subj)V & \bar{V} \rightarrow \bar{V}(\bar{N} : obj) & \bar{N} \rightarrow (\bar{A} : mod)\bar{N} & \\ \bar{V} \rightarrow (\text{eat}, V) & \bar{N} \rightarrow (\text{Peter}, N) & \bar{N} \rightarrow (\text{bean}, N) & \bar{A} \rightarrow (\text{red}, A) \end{array} \right\}.$$

On peut ainsi dériver, à partir de $\bar{V}$ une suite qui encode le produit des structures de la Fig. 1 : $((\text{Peter}, N) : subj)(\text{eat}, V)(((\text{red}, A) : mod)(\text{bean}, N) : obj)$. Cette dérivation est très proche de celle de la Fig. 7. En particulier, un symbole non terminal $\bar{X}$ s'apparente

à un nœud blanc d'étiquette X, alors qu'un symbole terminal X correspond à l'étiquette d'un nœud noir. En conclusion, les grammaires de réécriture peuvent être vues comme une implémentation particulière de la génération d'une correspondance entre arbre et suite, à savoir une génération guidée par la structure de l'arbre.

Nous pensons que, à l'instar des grammaires génératives que nous venons de définir, la plupart des grammaires génératives utilisées en modélisation de la langue sont en fait des interprétations procédurales particulières de grammaires transductives. Effectivement, la plupart de ces grammaires génèrent bien des structures produits, comme LFG (suite + *c*-structure + *f*-structure), TAG (suite + arbre dérivé + arbre de dérivation), HPSG (suite + structure de traits sémantico-syntaxique), ... Ce n'est évidemment pas le cas de n'importe quelle grammaire générative ; il paraît en effet difficile d'interpréter une machine de Turing quelconque comme une grammaire transductive, car on voit mal quelle structure associer à une suite générée par la machine. Notons tout de même que même les grammaires génératives qui peuvent s'interpréter comme des grammaires transductives ne considèrent pas clairement des niveaux de représentations indépendants et que le fait qu'elles définissent une supercorrespondance n'est qu'un effet de bord. Notamment dans le cas des grammaires de réécriture, ce n'est même pas le résultat d'une dérivation qui est interprété comme un structure produit mais le processus de dérivation lui-même (bien que cette vision ait été depuis modifiée par des extensions telles que les grammaires d'arbres comme TAG).

Indiquons, pour finir, un point qui nous paraît fort important même si nous manquons d'éléments pour l'étayer. Nous pensons que la notion de capacité générative forte a un lien évident avec le fait que les grammaires génératives utilisées en linguistique sont en fait des grammaires transductives cachées. On peut en effet considérer que deux grammaires sont **fortement équivalentes** si elles sont équivalentes en tant que grammaires transductives, c'est-à-dire si elles définissent la même supercorrespondance.

6. Conclusion

Nous avons introduit ici une famille élémentaire de grammaires transductives, les grammaires de dépendance atomiques, définissant une correspondance entre des arbres de dépendance (syntaxiques de surface) et des suites (morphologiques profondes). Nous pensons même qu'il s'agit de la famille la plus élémentaire qui soit, puisque les règles de grammaire ne ne manipulent qu'une portion élémentaire de la structure, à savoir une branche ou un nœud. Ce formalisme ne permet évidemment pas de décrire tous les phénomènes linguistiques et différentes extensions sont possibles. En particulier, il peut être utile d'écrire des règles manipulant simultanément plusieurs dépendances, comme le font Mel'čuk & Pertsov 1987. Néanmoins, ceci n'est pas une véritable extension du formalisme, car, comme on peut le montrer, une telle grammaire peut être simulée par une grammaire atomique à condition de multiplier les catégories syntaxiques.[9] Par contre, la prise en compte des structures non projectives nécessite une véritable extension du formalisme (cf., par exemple, Kahane 2000).

D'autre part, nous avons proposé plusieurs procédures pour définir une (super)correspondance (entre suites et arbres) avec des grammaires de dépendance atomiques, dans le sens de la synthèse comme de l'analyse, montrant l'équivalence de ces grammaires avec les grammaires de réécriture hors-contexte et les automates à pile. Une étude algorithmique complète spécifique à ces grammaires reste pourtant à faire.

9. Nous parlons ici d'équivalence formelle. Il est évident que, d'un point de vue linguistique, il est préférable d'éviter la multiplication des catégories et l'introduction de traits non linguistiquement motivés.

Références

AHO Alfred & ULLMAN Jeffrey, 1972, *The Theory of Parsing, Translation and Compiling*, *Vol. I : Parsing*, London : Prentice-Hall.

ARNOLA Harri, 1998, "On parsing binary dependency structures deterministically in linear time", in Kahane & Polguère (eds), *Workshop on dependency-based grammars, COLING-ACL'98*, Montréal, 68-77.

BLACHE Philippe, 1998, "Parsing ambiguous structures using controlled disjunctions and unary quasi-trees", *Proc. COLING-ACL'98*, Montreal, 124-130.

BOYER Michel & LAPALME Guy, 1985, "Generating paraphrases from Meaning-Text semantic networks", *Comput. Intell.*, *1*, 103-117.

COURTIN Jacques & GENTHIAL Damien, 1998, "Parsing with Dependency Relations and Robust Parsing", in Kahane & Polguère (eds), *Workshop on dependency-based grammars, COLING-ACL'98*, Montréal, 95-101.

DIKOVSKY Alexander & MODINA Larissa, 2000, "Dependencies on ther other side of the Curtain",*Grammaires de dépendance*, *T.A.L.*, *41 :1*, 69-98.

GLADKIJ Aleksej V., 1966, *Lekcii po matematičeskoj lingvistike dlja studentov*, Novosibirsk : NGU (fr. trad. : *Leçons de linguistique mathématique, fasc. 1*, 1970, Paris : Dunod).

GLADKIJ Aleksej & MEL'ČUK Igor, 1975, "Tree grammars : A Formalism for Syntactic Transformations in Natural Languages", *Linguistics, 150*, 47-82.

IORDANSKAJA Lidija, 1963, "O nekotoryx svojstvax pravil'noj sintaksičeskoj struktury (na materiale russkogo jazyka)" [On some properties of the correct synt. structure (on the basis of Russian)], *Voprosy jazykoznanija*, *4*, 102-12.

IORDANSKAJA Lidija & POLGUÈRE Alain, 1988, "Semantic processing for text generation", in *Proc. First International Computer Science Conf. - 88*, Hong Kong, 310-18.

KAHANE Sylvain & MEL'ČUK Igor, 1999, "Synthèse des phrases à extraction en français contemporain (Du graphe sémantique à l'arbre de dépendance)", *T.A.L.*, *40 :2*, 25-85.

KAHANE Sylvain, 2000, "Extractions dans une grammaire de dépendance lexicalisée à bulles", *Grammaires de dépendance*, *T.A.L.*, *41 :1*, ? ?.

LECERF Yves, 1961, "Une représentation algébrique de la structure des phrases dans diverses langues natuelles", *C. R. Acad. Sc. Paris, 252*, 232-234.

KORNAI Andreás & TUZA Zsolt, 1992, "Narrowness, pathwidth, and their application in natural language processing", *Disc. Appl. Math, 36*, 87-92.

MEL'ČUK Igor, 1967, "Ordre des mots en synthèse automatique des textes russes", *T.A. Informations*, *8 :2*, 65-84.

MEL'ČUK Igor, 1988, *Dependency Syntax : Theory and Practice*, Albany, NY : State Univ. of New York Press.

MEL'ČUK Igor, 1997, *Vers une Linguistique Sens-Texte*, Leçon inaugurale au Collège de France, Paris : Collège de France.

MEL'ČUK Igor, 1993-2000, *Cours de morphologie générale*, *Vol. 1, 2, 3, 4 & 5*, Montréal : Presses de l'Univ. Montréal / Paris : CNRS.

MEL'ČUK Igor & PERTSOV Nikolaj, 1987, *Surface Syntax of English. A Formal Model within the Meaning-Text Framework*, Amsterdam : Benjamins.

NASR Alexis, 1996, *Un modèle de reformulation automatique fondé sur la Théorie Sens-Texte – Application aux langues contrôlées*, Thèse de Doctorat, Univ. Paris 7.