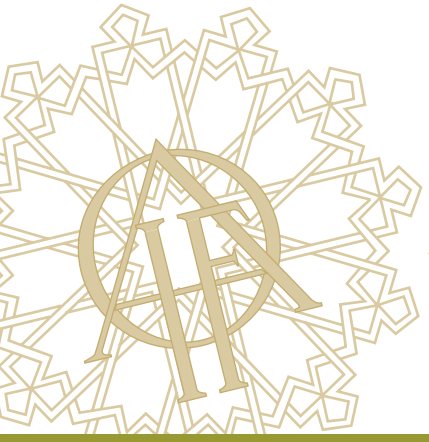

ANNALES ISLAMOLOGIQUES

en ligne en ligne en ligne en ligne en ligne en ligne en ligne en ligne en ligne en ligne

AnIsl 34 (2000), p. 179-192

André Jaccarini

Quelques opérations sur les langages associés aux opérateurs syntaxiques.

André JACCARINI

Quelques opérations sur les langages associés aux opérateurs syntaxiques

Le cadre théorique général

Une caractéristique intéressante de la langue arabe est de contenir un vaste sous-ensemble morphologiquement stable, définissable par des règles formelles de complexité minimale, duquel elle peut alors être déduite par l'adjonction de conditions et d'actions élémentaires associées aux règles de base. L'isolement de ce sous-ensemble et l'étude de ces principales régularités permet d'envisager ses extensions *progressives* tendant vers le langage réel – ou système morphologique général – et fournit ainsi les principes de construction de grammaires modulaires.

Ce sous-langage peut à son tour être projeté sur un langage squelette – ou langage quotient – constitué d'énoncés où, dans toutes les formes non atomiques, les racines – constituées de triplets consonantiques – ont été réduites à un seul représentant. Les invariants de cette transformation (les atomes ; en général des mots outils) jouent un rôle primordial dans l'étude de l'interaction entre les deux niveaux de langage que représentent respectivement la morphologie et la syntaxe.

Le cadre théorique de la catégorisation morpho-syntaxique à laquelle nous avons procédé est celui du monoïde syntaxique quotient. En effet, l'une des caractéristiques fondamentales du langage noyau, auquel nous venons de faire allusion, est la commutation des opérations de catégorisation et de projection ; commutation que nous pouvons, en vertu du principe d'invariance, étendre à la syntaxe. Il s'agit là d'une formalisation du principe du dictionnaire vide qui nous permet de travailler sans lexique. Cela signifie que le travail de catégorisation – ou définition des classes de congruences syntaxiques – se fait à partir de la seule considération du langage squelette, ou bien encore – à condition toutefois d'avoir défini une méthode de variation de grammaires – des phrases réelles ; mais le fragment de grammaire ainsi obtenu devant dans ce cas être projeté pour obtenir un fragment du langage quotient.

Le langage quotient est décrit grâce aux seules grammaires régulières (ou automates finis). Il est possible de l'enrichir ensuite en associant aux règles de base des tests (on contraint

le fonctionnement du mécanisme de base de production du langage) et en définissant des effets de bord. La grammaire se trouve ainsi enrichie par des *actions*, que l'on s'évertuera toutefois à ce qu'elles demeurent les plus élémentaires possibles, en sorte que le langage, constitué de l'ensemble des formes arabes, puisse apparaître comme une image homomorphique du langage quotient. Pour ce qui est du système morphologique général, ces tests et ces actions ne font que refléter – si l'on fait abstraction de certaines considérations de minimisation et d'optimisation à l'intérieur même du champ de la morphologie saine, ou langage noyau cité précédemment – qu'un ensemble d'opérations d'effacement, de permutation et d'élision. Les permutations en question ne concernent en général que l'ensemble des graphèmes : {*alef, waw, ya*}.

Ce principe général de construction pourra ensuite être étendu à la syntaxe en sorte que cette dernière puisse apparaître à son tour comme la transduction de la syntaxe quotient (celle du langage quotient) [1]. La construction d'une telle grammaire présenterait alors l'avantage de bien faire ressortir les invariances de la langue par rapport à l'opération de projection sur son squelette (le langage quotient) puisque toutes les spécificités syntaxiques non identifiables au niveau du langage quotient apparaîtraient uniquement sous formes de tests et d'actions. Il conviendrait alors, en vertu de ce qui a été dit précédemment, en ce qui concerne l'extension progressive du système morphologique de base vers le système morphologique général, de minimiser ces actions en sorte de faire apparaître également une homomorphie au niveau syntaxique.

En vertu donc de ces principes de construction, les catégories syntaxiques se trouveront bornées inférieurement puisque l'on ne s'attachera qu'à ne tenir compte, dans un premier temps, que de celles demeurant invariantes par projection, autrement dit de celles que l'on peut associer aux schèmes. Il nous semble ainsi que l'explicitation de ce système squelette nous fournirait un principe méthodologique fort pour une construction modulaire des modèles syntaxiques.

Mais ces enrichissements des grammaires syntaxiques ne constituent pour nous, à l'heure actuelle, qu'une perspective de développement qui suppose au préalable l'implémentation du système syntaxique quotient. Or, il s'agit là d'une tâche considérable et particulièrement délicate.

Dans nos précédentes publications, nous avons exposé les principes de base de construction d'un moniteur syntaxique ; la fonction principale de ce dernier étant d'améliorer le niveau de résolution morphologique du texte et d'optimiser les analyses [2]. Ce moniteur est essentiellement conçu comme un aiguilleur, fondé sur des fragments de grammaires syntaxiques, permettant d'éviter d'emblée les ambiguïtés, de court-circuiter les tentatives inutiles au niveau de l'analyse morphologique et de réduire les silences. Il s'agit donc d'une procédure dont le rôle le plus important est de prévenir les analyses parasites plutôt que de

[1] Le transducteur sera constitué d'un automate sous-jacent rendant compte de la syntaxe du langage quotient et émettrait comme langage de sortie le langage réel.

[2] Voir notamment «Vers une théorie du moniteur syntaxique» *AnIsl* 33, 1999 et «méthodes de construction d'un moniteur morpho-syntaxiques» *Actes du colloque de l'Afemam*, 1998 [à paraître aux éditions de l'Institut d'arabisation de Rabat], «À la Recherche du *ḫabar*» *AnIsl* 22.

les élaguer après une analyse morphologique hors-contexte et d'éviter ainsi les explosions combinatoires d'ambiguïtés de différents niveaux.

Les invariants lexicaux de l'opération de projection – les atomes – se trouvent situés au même niveau que les schèmes : ils constituent l'intersection du langage réel L et de son quotient par l'ensemble des racines, noté L/RAC. Par ailleurs, la cardinalité de leur classe de congruence syntaxique est très réduite (souvent égal à l'unité) ; on en déduit qu'ils entraînent sur leurs environnements des contraintes plus importantes que celles de la plupart des autres lexèmes. C'est pourquoi il nous sera possible de les considérer comme des opérateurs syntaxiques entraînant des attentes et de leur associer des sous-langages.

Ces derniers seront alors définis par des automates – intégrables au moniteur – devant interagir avec les automates morphologiques et dont il importera naturellement que la mise en œuvre ne représente un niveau de complexité supérieur à celui requis pour l'analyse morphologique. Ces sous-langages seront ainsi définis par des grammaires (automates ou transducteurs) se situant à un niveau minimal dans la hiérarchie des grammaires formelles, en sorte que leurs transformations, leurs réductions, leurs dérécursivisations ou leur traduction en procédés déterministes, etc., soient toujours possibles et contrôlables par des calculs algébriques. La définition d'un cadre algébrique rigoureux est nécessaire pour étudier les problèmes de compatibilité des deux niveaux de description syntaxique et morphologique, leur recoupement, la cohérence et les redondances de la description résultant de la réunion des différents fragments de grammaire. On pourrait ainsi concevoir une méthode pour synthétiser une description globale de la syntaxe quotient à partir des définitions formelles des sous-langages engendrés par les opérateurs syntaxiques et y tendre par approximations successives.

C'est donc principalement en raison des problèmes que nous venons d'évoquer, qui relèvent essentiellement de la gestion logique des différents fragments de grammaires, que nous nous sommes efforcé de définir les sous-langages associés aux opérateurs syntaxiques en sorte qu'ils présentent des propriétés de stabilité relativement aux principales opérations ensemblistes et algébriques (réunion, intersection, etc.). Il s'agit donc essentiellement de langages définissables par des grammaires régulières ou par une sous-classe de transducteurs finis qui présentent les mêmes particularités de stabilité.

Rappelons également que la définition d'un moniteur syntaxique suppose la méthode de variation de grammaires en vue de l'obtention de l'algorithme optimum – méthode qui ne peut se définir que dans le cadre algébrique évoqué plus haut – et un travail rigoureux sur les méthodes d'évaluation des grammaires.

Exemples de transformations de grammaires

Nous avons déjà présenté une description synthétique de certains opérateurs syntaxiques de haut rang, par exemple l'opérateur *inna*, *in*, *an* et bien d'autres (voir ref.). Ces études ont été complétées par la suite par des descriptions plus détaillées que l'on trouvera dans « Grammaires modulaires de l'arabe » [à paraître].

Notre objectif n'est pas ici de reprendre ces esquisses pour les affiner, mais au contraire de n'en retenir que les aspects les plus caractéristiques sur le plan formel et d'illustrer certaines transformations des grammaires associées aux sous-langages induits par chacun de ces opérateurs.

Si les descriptions desquelles dérivent les quelques exemples ci-dessous concernent le plus souvent le niveau superficiel, puisqu'il s'agit de construire des transducteurs d'un niveau de complexité minimale – sur lesquels il est toujours possible d'opérer des transformations et des réductions – cela ne nous empêche aucunement de partir aussi de définitions plus élaborées, la grammaire finale étant dans ce cas obtenue par réduction et transformation d'un modèle source plus conforme à une description naturelle, mais ne correspondant pas forcément à des procédures de reconnaissance (automates ou transducteurs) directement intégrables au moniteur syntaxique – si ce n'est pour leurs niveaux de récursivité ou de non-déterminisme généralement élevés. Notons d'ailleurs à ce propos que, pour la conception de simples accepteurs, il est souvent plus aisé de les faire dériver de descriptions plus structurales pour ne les transformer en procédures déterministes que dans un deuxième temps, tant il peut être difficile, dans de nombreux cas, de concevoir d'emblée des programmes de filtrage.

Sur un plan méthodologique, il est important d'insister sur le fait que les fragments de modèles que nous traitons s'appuient – dans un premier temps – sur la considération des sous-langages de L – ou langage réel – induits par les opérateurs et non sur celle des sous-ensembles de son quotient L/RAC, ce qui, soit dit en passant, n'a que peu de conséquences au niveau de la morphologie, en raison de l'invariance de cette dernière relativement à l'opération de projection. Mais il est clair que, pour s'assurer de l'indépendance du moniteur syntaxique par rapport au lexique, il faille veiller à ce que les modèles sur lesquels il s'appuie soient construits à partir des seules catégories syntaxiques présentant la même propriété d'invariance, c'est-à-dire sur les seules catégories que l'on peut associer aux schèmes. Cette dernière remarque ne fait que préciser ce que nous avons évoqué en début d'article, à savoir que le cadre théorique du travail de catégorisation et de modélisation linguistique est essentiellement celui du monoïde syntaxique *quotient* et justifie l'expression de *syntaxe quotient*.

Si donc les fragments de modèles – plus précisément les schéma*s* de ces fragments, puisque nous n'en retiendrons en général que les caractéristiques les plus formelles – sont essentiellement construits à partir de représentations *naturelles* (il est évidemment plus facile de raisonner sur le langage réel plutôt que sur son squelette), il n'en demeure pas moins que le cadre algébrique défini et la nature des principales descriptions [3] nous laisseront théoriquement la latitude de transformer les fragments de grammaire considérés et de les

[3] C'est-à-dire les automates finis augmentés, ou transducteurs, dont les actions associées aux transitions présentent la particularité de préserver les principales propriétés mathématiques de stabilité évoquées plus haut – permettant ainsi de généraliser les calculs de transformation et de réduction valables pour les automates non augmentés.

remodeler en sorte que l'invariance relativement à la projection soit respectée, ce qui, schématiquement, revient à passer du monoïde syntaxique à son quotient [4].

Dans les fragments suivants pourront ainsi apparaître des catégories que l'on ne peut pas toujours attacher au schème, telle par exemple la « catégorie » *adjectif*. Remarquons toutefois au sujet de cette dernière qu'il n'en existe pas moins un opérateur d'adjectivation (pour les adjectifs de relation : la *nisba*) détectable par le programme d'analyse morphologique. Dans le langage quotient L/RAC, seul ce type d'adjectifs pourront figurer comme une classe à part entière. Ceux qui en revanche possèdent un schème nominal non suffixé par un opérateur d'adjectivation devraient être confondus avec les noms. En se situant dans un cadre précis, dans lequel peuvent aisément s'effectuer des transformations de grammaires, nous pouvons nous autoriser à négliger, à un premier niveau de description, cet aspect de la question.

Nous nous bornerons par ailleurs à ne fournir que des exemples de transformations d'automates *sous-jacents* et non de transducteurs.

Exemples d'opérateurs de haut niveau

Dans ce qui suit, on considère qu'à chaque opérateur syntaxique est associé un langage régulier – c'est-à-dire tel que les procédures susceptibles de le produire soient réductibles à des automates finis. Mais il importe de rappeler que :

1. Ces langages sont susceptibles d'enrichissements qui les transformeront en langages définissables par une sous-classe de transducteurs présentant la particularité de se prêter au même type de transformations que celles que nous pouvons calculer lorsqu'on traite les

[4] Notons que le monoïde syntaxique est lui même un monoïde *quotient* puisqu'il est constitué de l'ensemble des classes de congruence syntaxique, lesquelles, rappelons-le, sont formées d'ensembles de mots – appartenant au monoïde libre – pouvant commuter entre eux dans toutes les séquences possibles sans que cette opération ne remette en question l'appartenance (ou la non-appartenance) de cette séquence au langage. La notion de congruence syntaxique constitue ainsi une formalisation du *principe distributionnel* et celle de classe de congruence syntaxique est à rapprocher – dans le cas des langages naturels il est nécessaire en général de procéder au préalable à des opérations de segmentation en morphèmes, de normalisation, lemmatisation, etc. – de celle de catégories syntaxiques, ces dernières constituant donc les éléments du monoïde syntaxique. Similairement, les schèmes peuvent aussi être considéré comme des classes de formes équivalentes (plus précisément ils sont définis comme *l'ensemble des plus petites classes non vides de la clôture du lexique closes par permutation de racines*) compatibles avec les opérations de concaténation, c'est-à-dire finalement des classes de congruences. De plus cette nouvelle congruence (à laquelle se trouve associée une nouvelle structure quotient) est compatible avec la congruence syntaxique. Il s'agit donc d'un double passage au quotient, dont l'ordre est *indifférent*, dans le cas de la morphologie : soit on catégorise d'abord dans L pour obtenir le monoïde syntaxique (premier passage à une structure quotient) et l'on projette ensuite pour obtenir L / RAC et son monoïde associé (deuxième passage au quotient) soit à l'inverse on procède directement sur le langage quotient L / RAC en dégageant les classes de congruence syntaxique de L / RAC. Si l'on désigne respectivement par Π et SC les homomorphismes canoniques associés aux opérations de catégorisation et de projection (ou réduction au squelette) la propriété de commutation que nous venons d'énoncer s'exprime alors par l'égalité Π. SC = SC. Π. Le principe d'invariance auquel nous nous référons consiste à étendre à la syntaxe cette égalité qui n'est valable que pour le langage constitué par l'ensemble des mots *graphiques* (toutes séquences licites séparées par deux blancs) : la catégorisation syntaxique doit se faire en sorte que l'égalité Π. SC = SC. Π ne soit pas mise en défaut. Or comme on est amené le plus souvent à raisonner dans L et non dans le langage squelette L / RAC, on peut être amené à modifier la grammaire obtenue au cas où elle violerait le principe d'invariance. Il serait alors souhaitable de procéder à de telles modifications en ayant simplement recours à des transformations algébriques.

automates sous-jacents. Cette particularité est due à la nature relativement simple des actions introduites. Ces dernières sont nécessaires pour tenir compte de tous les phénomènes linguistiques non réductibles à des contraintes connexes (se propageant de proche en proche) ; dans ce cas, le recours à des registres de mémoire est nécessaire ;

2. Le formalisme que l'on utilise pour la description de ces langages est celui auquel on a habituellement recours lorsqu'on traite la syntaxe, à savoir les systèmes de réécriture correspondant aux grammaires indépendantes du contexte, augmentées ou non (en fait nous nous bornerons, dans les exemples ci-dessous, au second cas). Toutefois toutes ces grammaires sont dérécursivables et réductibles à des grammaires régulières que l'on peut faire varier en vue d'obtenir le compromis optimum entre la pertinence linguistique et le niveau d'indéterminisme (lequel peut aussi être entièrement supprimé).

Rappelons enfin que les modèles originels desquels ont été tiré les schémas ci-dessous ont été le plus souvent construits à partir de la notion de *noyau* d'une structure, auquel s'agrègent successivement, dans un ordre déterminé, des *ajouts* devant préserver l'invariance de la structure. C'est en somme le principe de construction des grammaires en chaîne ; mais l'esquisse obtenue se trouve ensuite simplifiée et transformée en vue d'obtenir des programmes efficaces de reconnaissance.

Le langage L(*inna*)

On pourrait ainsi définir, de manière descendante, le langage associé à l'opérateur *inna* à partir des seules variables V_T={GND, GN, GV, GAD, pro., pro. sep}, lesquelles devront être définies à leur tour jusqu'à l'obtention d'un système de réécriture pouvant se résoudre en une formule ne comportant que des symboles appartenant au vocabulaire terminal retenu. Ce dernier peut – en théorie – n'être constitué que de l'ensemble des graphèmes arabes, si l'on désire intégrer dans une représentation unique la syntaxe et la morphologie. Toutefois, la nature des données linguistiques que nous étudions nous contraint le plus souvent à combiner les deux méthodes, ascendantes et descendantes, de description et à unifier ensuite les modèles en les transformant si besoin en est.

Les quelques calculs de transformation exposés ci-dessous ne doivent être compris que comme une première illustration destinée à convaincre le lecteur, à partir de cas très simples de dérécursivation, d'« unification [5] » ou de réduction du niveau d'indéterminisme, des possibilités plus vastes de variations contrôlées – ou de synthèse de modèles à partir de fragments – qui s'étendront à un type plus général de grammaires (puisqu'elles seront augmentées). Le noyau d'un atelier de grammaires a d'ailleurs été conçu et programmé en lisp, à cette fin.

Le premier schéma de description de L(*inna*) (fig. 1) correspond en fait à un automate fini dont les arcs de transition ne sont étiquetés que par des symboles appartenant à V_T. Examinons à présent chacun de ces symboles.

[5] Ce terme n'est naturellement pas utilisé ici dans le sens des « grammaires d'unification ».

Définition des symboles du premier schéma de description de L(*inna*)

GND et GN représentent respectivement le groupe nominal déterminé et celui qui ne l'est pas. L'importance de la détermination en arabe fait clairement apparaître la nécessité de séparer les deux structures. En ce qui concerne justement *inna*, il serait facile de déduire, à partir du schéma de grammaire de son langage associé (fig. 1), que cet opérateur peut aussi se définir comme induisant l'attente de la fin des marques de détermination, lesquelles, contrairement au français par exemple, se répètent jusqu'à la fin du GND. En effet, le schéma d'automate codé par le système d'écriture ci-dessous met en évidence que l'opérateur commande la concaténation d'un GND et d'un GN ou d'un GND et d'un GV – le groupe verbal, dont la formule n'est pas développée ici, mais que l'on peut schématiquement décrire comme formé d'un noyau constitué d'un verbe, susceptible de recevoir de multiples ajouts – ou bien encore d'un GND et d'un GAD (groupe adjectival non déterminé, *voir infra*), sans oublier le cas où les deux composantes sont séparés par un pronom appartenant à la catégorie pro.sep, strictement incluse dans celle des pronoms (pro). Mais ces derniers, lorsqu'ils interviennent juste après l'opérateur, (règle 3 : dans ce cas il commute avec GND) peuvent s'y agglutiner, en subissant quelquefois de légères modifications de forme, pour constituer un seul mot graphique [6]. Cette configuration est alors intéressante pour ce qui est de l'analyse puisqu'elle permet d'identifier le deuxième composant dans sa fonction de prédicat, dès la deuxième occurrence. De plus, cette dernière remarque attire l'attention sur le fait qu'il ne s'agit là que d'un schéma d'automate, puisque les transitions peuvent tantôt coder des structures entières (mais qu'il faudrait alors développer pour *les y inclure* si l'on désire éviter que n'apparaisse, déjà à ce niveau, de la récursivité) tantôt des fractions de mot graphique, ce qui nous ramène à la morphologie et au problème de segmentation et illustre très simplement certaines difficultés que l'on peut rencontrer lors de l'unification des modèles syntaxiques et morphologiques.

Le schéma d'automate est le suivant :

Figure 1

1. L(*inna*) → *« inna »*.q1 ; *le symbole « inna » fait passer la « machine » de l'état initial à l'état q1*
2. q1 → GND.q2 ; *à cette transition se trouve associée la fonction de « thème » (=mubtada')*
3. q1 → pro.q2 ; *« pro » commute avec GND*
4. q2 → pro.sep.q3 ; *marque de séparation entre thème et prédicat (= khabar)*
5. q2 → .q3 ; *instruction de saut ; le pronom de séparation commute avec le mot vide ; sa suppression ne modifie pas la structure de la phrase*
6. q3 → GN q4 ; *à cette transition se trouve associée la fonction de « prédicat »*
7. q3 → GAD q4 ; *GAD commute avec GN*
8. q8 → GVq4 ; *GV commute avec GN*

[6] Il ne s'agit donc pas d'une simple concaténation.

Le principal intérêt de ce schéma réside en sa simplicité. L'opérateur *inna* entraîne nécessairement l'apparition de deux composants dont le premier, dans la fonction de thème, est repéré par la « machine » si elle parvient à l'état q2 et le deuxième, dans la fonction de prédicat, par q4. Les deux transitions entre q2 et q3 (dont l'une est étiquetée par le mot vide) marquent la séparation entre les composants.

Si l'on soumet à ce schéma d'automate la séquence :

« inna al-walad(a) mariḍ(un) »

les suites de configurations de la machine seront les suivantes :

(L(*inna*), *« inna al-walad(a) mariḍ(un) »*)

(q1, *« al-walad(a) mariḍ(un) »*) ; *« inna »* a été accepté et la machine est parvenue à l'état q1.

(q2, *mariḍ(un) »*) ;

(q3, *mariḍ(un) »*) ; la machine a effectué un saut pour parvenir à q 3 sans lire de mot.

(q4, ε) ; la phrase est acceptée : la machine est parvenue à l'état terminal q4 et il reste plus de symbole à lire : ε représente le mot vide.

À la reconstitution de ce parcours, on peut associer l'arbre :

(L(*inna* *inna* (GND = thème *al-walad*(*a*) (sep ε (GN = prédicat *mariḍ*(*un*))))).

Mais il importe de remarquer que la très grande simplicité de ce schéma n'a pu être réalisée qu'au prix d'un grave inconvénient : l'indéterminisme introduit par la transition vide entre q2 et q3, qui a pour conséquence de complexifier l'opération de reconnaissance. Cet indéterminisme apparaît ici de manière naturelle, en ce sens qu'il est le reflet direct d'un phénomène linguistique et peut d'ailleurs représenter un coût bien plus élevé au cas où les transitions vides sont majoritairement situées en début d'automate, ce qui, malheureusement, est très souvent le cas si l'on considère les grandes structures du type GN ou GND susceptibles de recevoir plusieurs ajouts de début, sans que ces derniers n'affectent sa stabilité.

Dans cet exemple, il sera possible néanmoins de supprimer totalement l'indéterminisme sans trop affecter la pertinence linguistique. Le schéma précédent peut en effet être transformé par un calcul, qui, dans cette situation particulière, est très simple (et que nous n'expliciterons pas) en un schéma déterministe auquel il sera encore possible d'associer aux transitions et aux états des catégories linguistiques. Rappelons toutefois qu'il ne s'agit que du déterminisme du schéma et non de l'automate dérivé, qui peut donc réapparaître lors du développement des formules codant les transitions (ce qui peut amener à procéder à de nouveaux calculs). Cette opération nous aura permis néanmoins de diminuer le niveau d'indéterminisme de tout automate directement déduit du schéma.

Le schéma ci-dessus peut ainsi être transformé en un schéma déterministe équivalent (faiblement) :

Figure 2

1. L(*inna*) → *« inna »*.q1 ;	6. q2 → GAD q4 ;
2. q1 → GND.q2 ;	7. q2 → GVq4 ;
3. q1 → pro.q2 ;	8. q3 → GN q4 ;
4. q2 → pro.sep.q3 ;	9. q3 → GAD q4 ;
5. q2 → GN q4 ;	10. q3 → GVq4 ;

acceptant exactement le même type de phrases que celles acceptées par le schéma précédent. De plus, les arbres que l'on pourra associer aux suites de configurations que prendra le nouveau schéma d'automate lors de l'opération d'acceptation, quoique de nature différente de ceux produits par le premier, présenteront un niveau de pertinence comparable.

Définition des principales caractéristiques des structures intervenant dans le schéma de L(*inna*)

Le symbole GAD représente l'un des deux systèmes suivants :

GAD → Adj. GAD	ou bien	GAD → GAD. Adj
GAD → Adj.		GAD → Adj.

Ce dernier système, bien que récursif à gauche (ce qui pose théoriquement des problèmes au niveau de l'implémentation), n'en reflète pas moins la véritable nature du phénomène linguistique : le groupe adjectival demeure stable s'il reçoit comme ajout de droite [7] un adjectif. Ces deux systèmes acceptent cependant la même solution, à savoir toutes les chaînes du type Adj. Adj.* (toutes les suites d'adjectifs de longueur arbitraire supérieure ou égale à 1, le symbole * est connu sous le nom d'étoile de Kleene) [8]. Le groupe adjectival déterminé qui intervient dans la définition de GND se définit similairement par la formule Art.Adj (Art.Adj)*.

Pour ce qui est des descriptions de GN et GND, nous avons retenu le principe de construction des grammaires en chaînes ; ce qui nous a amené à considérer des groupes nominaux suffisamment étendus, et à effacer ensuite *progressivement* tous les éléments dont la suppression n'entraîne pas l'effondrement de la structure. Le noyau de GN se réduit ainsi au seul N [9] ; mais cette structure noyau est susceptible de recevoir des ajouts de gauche et des ajouts de droite. Pareillement, GND peut se définir comme un noyau Art.N (ex : *al-kitāb*(*u*)), lequel peut commuter avec N.pro.lié (ex : *kitāb(u)-hu*) et bien sûr NPR ce que l'on note (Art.N + Art.pro.lié + NPR), pouvant être également extensible à gauche et à droite. Quant aux éventuelles incompatibilités entre les ajouts de gauche et de droite, elles sont traitées par augmentation de la grammaire, essentiellement par la définition d'un type particulier de registres (les drapeaux) et des tests ; toutefois – comme nous l'avons déjà signalé – ces enrichissements ne sont pas de nature à limiter la puissance des calculs de transformation de grammaires. Nous n'y ferons donc plus allusion.

Toutefois, il est important de signaler que GND peut aussi être constitué d'éléments isolés (pronoms personnels ou démonstratifs) *non susceptibles de recevoir des ajouts*.

[7] Les ajouts de droite sont ceux qui interviennent en fin de structure. Nous avons choisi de rendre notre terminologie compatible avec le métalangage (celui de l'écriture de la grammaire) plutôt qu'avec le langage lui même.

[8] En somme, on procède à la mise à plat des structures dans la mesure où l'on ne perd pas trop de pertinence linguistique relativement à l'application recherchée. Notons toutefois qu'il peut être intéressant, dans certains cas, de conserver certains éléments de l'indicateur syntagmatique, notamment certaines catégories apparaissant vers le haut de l'arbre auxquelles peuvent être attachées certaines fonctions syntaxiques dont les marqueurs relèvent de l'analyse morphologique (la détermination du cas par exemple). Il faut donc veiller à ne pas perdre, lors des transformations de grammaires, les principaux traits susceptibles d'interférer avec l'analyse morphologique.

[9] Le problème de l'article indéterminé est traité dans GMA (chapitre X) ; il est en fait incorporé au nom.

Les ajouts de gauche à GN pourront se définir par la formule :
AJG = (part + ε) (num + ε)(app + ε)(num + ε), où part, num et app désignent respectivement les partitifs, les numéraux et les particules d'appartenance. On notera que l'occurrence des ε à l'intérieur de chaque parenthèse fait bien apparaître le statut d'ajout de ces éléments qui commutent tous avec le mot vide et causent une augmentation considérable de l'indéterminisme puisqu'ils interviennent en début de structure. Nous aurons donc intérêt à faire disparaître l'indéterminisme dans cette section initiale de l'automate.

Nous ne développerons pas ici les descriptions de GN ou GND, dont on pourra trouver une étude plus complète dans « *Grammaires modulaires de l'arabe* » (ch. 10) et dont les premières esquisses, sous forme de réseaux de transition, se trouvent déjà mentionnées « *À la recherche du ḥabar* » (voir références). Il suffira de se focaliser sur certains traits formels essentiels et sur les transformations de ces structures.

Schéma de GN et transformations

Le groupe nominal non déterminé peut recevoir comme ajout de droite aussi bien :
– des N, éventuellement précédés d'une particule d'appartenance ;
– des GAD, précédés ou non d'un privatif (*ġayr*), lequel peut également intervenir à l'intérieur de la succession d'adjectifs ;
– le terminal *mā ;*
– l'*iḍāfa* de qualification ;
– le *naʿt sababī* suivi d'un pronom de rappel ;
– des structures adverbiales ou élatives ;
– des relatives indéterminées, qui peuvent aussi bien intervenir directement après le noyau ou les précédents ajouts ;
– et enfin d'autres GN ou GND [10] introduits par des prépositions ; ce qui pose un problème de récursivité [11].

Si l'on fait maintenant abstraction de certaines structures, dont la représentation ne devrait pas poser de problèmes de complexification dans la gestion logique de la grammaire, pour se concentrer d'une part sur les relatives indéterminées et d'autre part sur la récursivité

[10] Dans le cas de GND certaines restrictions sont à prévoir.

[11] Remarquons au passage que l'iḍāfa de qualification (IDQ) qui s'applique aussi à GND peut se définir en toute généralité par l'automate suivant :

IDQ → Art. q1
IDQ → NPR. q2
IDQ → N. q3
q1 → N. q2
q2 → Art. q3
q3 → Adj. q4
q4 → N. q5
q5 → N

La résolution du système et la refactorisation de l'expression obtenue pourra alors faire apparaître clairement dans la même formule les deux cas : déterminé et non déterminé et leur section commune :

IDQ = Art.q1 + NPR. q2 + N. q3
=Art. N.q2 + NPR.q2 + N.Adj. q4
=(Art.N..+NPR)q2 + N.Adj.q4
=(Art.N.+NPR)Art.q3 + N.Adj.Art.q5
=(Art.N..+NPR)Art.Adj.q4 + N.Adj.Art.N
=(Art.N..+NPR)Art.Adj.Art.q5 + N.Adj.Art.N
=(Art.N.+NPR)Art.Adj.Art.N + N.Adj.Art.N
=((Art.N.+NPR)Art.Adj. + N.Adj).Art.N
=((Art.N.+NPR)Art. + N)Adj.Art.N

Dans le premier facteur apparaît les deux sous-facteurs, le déterminé : (Art.N.+NPR) Art et le non déterminé : N ; le facteur final Adj. Art. N est commun aux deux.

inhérente au groupe nominal, on peut regrouper tous ces ajouts dans une « catégorie » que l'on désigne par X. On obtient alors le schéma suivant :

Figure 3

GN → AjG. q1
GN → q1 ; ..... *instruction de saut*
q1 → N.q2 ; ..... *noyau*
q2 → q6 ; ..... *q6 est l'état de sortie*
q2 → X.q3 ;
q2 → q3 ; ..... *instruction de saut*
q2 → q6 ; ..... *q6 est l'état de sortie*
q3 → Rel.ind.q4 ;
q3 → q4 ; ..... *instruction de saut*
q3 → q6 ; ..... *q6 est l'état de sortie*
q4 → prep. q5
q4 → q6 ; ..... *q6 est l'état de sortie*
q5 → prep. q5 ; ... *boucle*
q5 → GN. q6 ; ..... *appel récursif de GN*
q5 → GND'. q6 ; *appel de GND duquel sont exclues certaines transitions.*
q6 → ε ; ..... *sortie*

Les transformations auxquelles on procédera auront alors pour effet de réduire à la fois l'indéterminisme et la récursivité tout en sauvegardant un bon niveau de pertinence linguistique.

Le développement du système jusqu'à q5 – état qui requiert un traitement particulier – fournit la formule parenthésée :

GN = (ε + AjG) N (ε + (ε + X)(ε + (ε + Rel.ind)(ε + prep. q5).

q5 se définit par l'équation :

q5 = prep. q5 + (GN + GND')q6 = prep. q5 + GN + GND'

qui se résout de la manière suivante [12] :

q5 - prep. q5 = GN + GND'

d'où

(ε-prep) q5 = GN + GND'.

En procédant à la multiplication (à gauche) des deux termes par l'inverse de (ε – prep), il vient :

$q5 = (\varepsilon - \text{prep})^{-1}$ (GN + GND').

Or, on démontre dans ce genre de structure que

$(\varepsilon - \text{prep})^{-1} = \varepsilon + \text{prep} + \text{prep.prep} + \text{prep.prep.prep} + \ldots\ldots$
$= \varepsilon + \text{prep} + \text{prep}^2 + \text{prep}^3 + \ldots\ldots$

[12] On démontre qu'il est possible – pour résoudre ce genre d'équations – de se situer dans une structure algébrique (corps non commutatif) où la disjonction et la concaténation jouent des rôles structurellement équivalents à l'addition et la multiplication (sauf que la concaténation n'est pas commutative). Dans cette structure, il est alors possible de définir des opérations inverses : l'effacement et la suppression structurellement similaires aux opérations de division et de soustraction.

et l'on obtient ainsi une caractérisation algébrique de la boucle. Le symbole *, connu sous le nom d'étoile de Kleene, lorsqu'il est affecté à une chaîne quelconque, sert en général à désigner que cette chaîne peut se répéter un nombre indéfini de fois (0 inclus).
On a donc prep* = $(\varepsilon - \text{prep})^{-1}$.
Il est aisé en effet de vérifier que (ε − prep) est l'inverse de prep* :

(ε − prep)(ε + prep + prep^2 + prep^3 +.....)
= ε + prep + prep^2 + prep^3 +.....
- prep - prep^2 - prep^3 -.....
= ε

Finalement q5 s'écrit :

q5 = prep*(GND' + GN) ;

et GN peut se réécrire :

(AjG.N + N) (ε +.X (ε +. Rel.ind) + Rel.ind) (ε +prep. (prep*(GND' + GN))).

La récursivité du schéma provient du fait que le symbole GN intervient dans le membre droit de l'équation. Mais dans ce cas il sera possible de la supprimer, tout en conservant un niveau de pertinence linguistique suffisant pour le guidage de l'analyse morphologique. En isolant, dans le membre droit de l'expression, le facteur de GN, on obtient :

GN = (AjG.N + N) (ε +.X (ε +. Rel.ind) + Rel.ind) prep. prep*GN +
+ (AjG.N + N) (ε +.X (ε +. Rel.ind) + Rel.ind) prep. prep*GND'
+ (AjG.N + N) (ε +.X (ε +. Rel.ind) + Rel.ind)

et en désignant par Y le plus grand facteur commun gauche c'est-à-dire :

(AjG.N + N) (ε +.X (ε +. Rel.ind) + Rel.ind)

on obtient :

GN = Y prep.prep* GN + Y prep.prep* GND'+ Y
$\Rightarrow$ GN - Y prep.prep* GN = Y prep.prep* GND'+ Y
$\Rightarrow$ (ε - Y prep.prep*) GN = Y prep.prep* GND'+ Y
$\Rightarrow$ GN = (ε - Y prep.prep*)$^{-1}$ Y prep.prep* GND'+ Y
$\Rightarrow$ GN = (Y prep.prep*)*Y prep.prep* GND'+ Y.

La récursivité a ainsi disparu à ce niveau, puisque Y ne contient pas de symbole de GN. Par ailleurs, on peut réduire le niveau d'indéterminisme du schéma précédent en deux étapes :
– en vertu de l'égalité A*A = A A* ;
on peut réécrire :

GN = Y prep.prep*(Y prep.prep*)* GND' + Y

formule qui code le schéma d'automate :

q1 → Y. q2
q1 → Y. q4
q2 → prep. q3
q3 → prep. q3
q3 → q1
q3 → GND'q4

lequel est toujours indéterministe du fait :

1- de l'instruction de saut entre q3 et q1 ;

2- que la lecture du symbole Y, lorsqu'on se trouve en q1, peut faire passer la « machine » soit en q2 soit en q4.

Mais une nouvelle transformation permettra de supprimer totalement l'indéterminisme au niveau du schéma (laquelle pourra cependant réapparaître lors du développement de Y). Le schéma déterministe est le suivant :

q1 → Y.q2
q2 → prep. q3
q3 → Y. q2
q3 → prep.q3
q3 → GND'.q4
q4 → ε.

En supprimant à nouveau l'indéterminisme dans le système définissant Y (ce qui ne pose aucune difficulté particulière mais risque de diminuer quelque peu la pertinence linguistique des arbres associés à la grammaire) et en raccordant les tronçons de manière cohérente, il est alors possible d'obtenir un schéma entièrement déterministe et non récursif.

Un schéma simplifié et déterministe de GND

On fournit ici directement un schéma simplifié et déterministe de GND, lequel a été obtenu par réduction et transformation d'un modèle plus élaboré construit suivant le principe des grammaires en chaînes, qui commande de faire clairement apparaître la structure GND = AjG.Noyau.AjD. Notons toutefois qu'il ne s'agit de déterminisme que relativement au vocabulaire {Art, pro. pers, pro. rel, pro. lié, dem, n}. Le symbole NA représente les noyaux atomiques, c'est-à-dire tous les groupes nominaux déterminés constitués d'un seul mot graphique et excluant les ajouts. Cette catégorie n'est constituée que d'atomes (démonstratifs et pronoms). Ces mots appartiennent à la classe des mots échappant à l'analyse morphologique et sont donc repérés en premier, d'où l'intérêt particulier de cette catégorie.

GND → NA.q8
GND → Art.q1
GND → Dem.q2
GND → N.q3
q1 → N.q8
q2 → Art.q6
q3 → Art.q5
q3 → N.q4
q3 → pro.lié.q8
q4 → Art.q7
q4 → pro.lié.q8
q5 → N.q8
q6 → N.q8
q7 → N.q8
q8 → ε

On pourrait adjoindre ensuite à ce modèle les ajouts de gauche AjG, dont la formule est plus complexe que celle concernant le GN, et à faire apparaître la catégorie « Pré-Article » – ce qui posera un problème de modélisation du fait de l'incompatibilité du démonstratif,

en tant qu'ajout à gauche, avec les noyaux du type N.pro.lié. En effet, cette particularité nous contraindra de subdiviser cette catégorie en deux sous-catégories PreART1 et PreART2, qui admettent cependant un large tronçon commun.

Il nous semble enfin intéressant de remarquer que la mise en parallèles des structures GN et GND décrites toutes les deux par des formules du type AjG.Noyau. Ajd, nous permettrait de procéder à des comparaisons et à isoler les éléments discriminants. L'intérêt du procédé est plus général encore si l'on cherche par exemple à construire des transducteurs permettant le passage d'un langage à un autre : *les comparaisons structurelles des constituants correspondants du langage source et du langage cible peut aider à définir de manière optimale les actions associées à l'automate de base.*

On pourrait aussi tenter de rendre plus homogène les structures sous-jacentes de GN et GND en définissant un réseau de base commun et en décrivant toutes les différences par augmentation de la grammaire (en ayant simplement recours à la technique des drapeaux et des tests) en sorte que les calculs de transformations restent possibles [13]. Nous n'avons donné de ces dernières que des exemples très simples. Les manipulations de grammaires et le calcul des transducteurs, que nécessitera la mise au point du modèle, seront naturellement plus complexes : elle requiert la création d'un véritable environnement de génie linguistique, dont nous avons d'ailleurs programmé le noyau.

Rappelons enfin que la stabilité des sous-langages $L(tok_i)$ relativement aux opérations d'intersection et de réunion rend plus aisé l'isolement des cas où l'analyse pourrait poser problème, du fait de l'ambiguïté graphique de l'opérateur lui-même ou bien des intersections non vides des sous-langages associés aux opérateurs.

[13] On pourrait ainsi mieux isoler la fonction de détermination dont nous avons déjà souligné l'importance