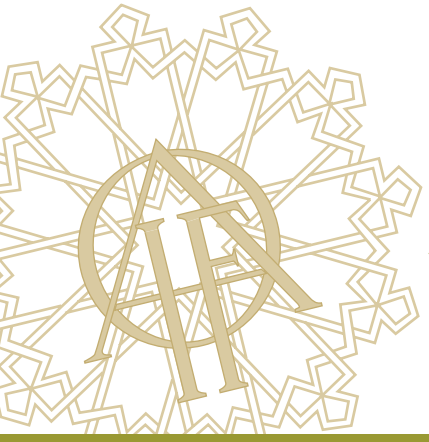


ANNALES ISLAMOLOGIQUES

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

AnIsl 33 (1999), p. 75-103

André Jaccarini

Vers une théorie du moniteur syntaxique.

André JACCARINI

Vers une théorie du moniteur syntaxique

Mots-clefs : traitement sans lexique de l'arabe, minimalité, modélisation, automate, transducteur, analyseur, complexité, modularité, optimisation, variation de grammaires, invariance, métrologie des grammaires, processus cognitifs, psycholinguistique, didactique, opérateurs syntaxiques, synthèse de grammaire.

Cet article est un exposé des principaux travaux qui ont été menés, depuis plusieurs années, au Département d'analyse et de traitement automatique des textes de l'Iremam ainsi qu'à l'Ifao, dans le domaine de la modélisation linguistique et algorithmique de la langue arabe. Les principales applications informatiques dont on peut faire état aujourd'hui dérivent directement des modèles théoriques que nous ne pouvions présenter ici que de manière très générale et nécessairement incomplète.

Si cet article s'adresse surtout à l'arabisant et au linguiste, plus particulièrement à ceux d'entre eux s'intéressant aux systèmes formels et au traitement automatique, nous nous sommes efforcés de le rédiger en sorte qu'aucun prérequis ne soit nécessaire. Les astérisques renvoient à l'annexe où se trouve un ensemble de définitions élémentaires qui aideront, peut-être, le non-spécialiste à saisir l'essentiel de notre argumentation, notamment lorsque nous aborderons dans la deuxième partie l'aspect algorithmique.

Étant donné la nature transdisciplinaire de ce projet, il nous a semblé important d'exposer dans la première partie nos motivations initiales, les difficultés rencontrées lors de l'élaboration des modèles, ainsi que les différentes étapes de développement. L'un des thèmes centraux de cet article est en effet la mise en parallèles des processus cognitifs d'appréhension et l'approche algorithmique. Les enjeux pouvaient ne pas apparaître toujours évidents, tant les imbrications entre différents niveaux logiques sont importantes. Ces dernières pouvaient, de prime abord, sembler inextricables si l'on ne s'était assuré, à partir d'une expérience pédagogique probante, de l'existence d'une solution. Si pour certains « les objets existent indépendamment de l'effort de notre esprit pour les saisir », encore faut-il, à ce qu'il nous semble, rendre compte de cet effort lorsque ces objets se trouvent proches de nous.

Annales islamologiques 33—1999
AnIsl 33 (1999), p. 75-103 André Jaccarini

La deuxième partie est un résumé du travail que nous avons accompli dans le domaine de l'algorithmique linguistique. Nous y reprenons pour l'essentiel des thèmes qui ont été largement exposés dans « Grammaires modulaires de l'arabe » (à paraître). Nous avons toutefois rédigé cette partie dans le souci de mieux faire apparaître le parallélisme que nous venons d'évoquer.

Dans la dernière partie, nous exposons certaines perspectives dans le domaine de la théorie algorithmique. Mais cet aspect combinatoire du problème, une fois élucidé, nous devrions surtout, à présent, nous attacher à développer la composante psycholinguistique, ce qui nous amène à nous intéresser à des paradigmes complémentaires de recherche.

I. ORIGINES DU PROJET

I.1. *Les motivations de type cognitif et algorithmique*

Le projet développé au Département d'analyse et de traitement automatique des textes a pour point de départ une réflexion approfondie sur la nature des procédés intellectuels mis en jeu lors de l'opération de décodage d'un texte arabe par un débutant dont les facultés d'appréhension ne sont pas supposées *a priori* figées [1], sur les méthodes qui pourraient nous amener à les identifier avec quelque précision, sur le type d'interactions avec le lexique qu'elles supposent. Dans un deuxième temps, cette réflexion s'est naturellement portée sur les stratégies les plus efficaces de recherches et la manière d'organiser ces dernières et de les hiérarchiser afin de les rendre moins fastidieuses.

Le problème du décodage d'un texte lors de la phase d'apprentissage peut en effet représenter une difficulté assez considérable : le système d'écriture de l'arabe autorise des concaténations d'éléments en nombre assez important si bien qu'il n'est pas toujours évident de reconnaître les lexèmes dans les mots graphiques qui les contiennent. Par ailleurs, étant donné l'organisation du système morphologique et l'importance du système de dérivation, les classements des entrées dans un dictionnaire arabe se font en général selon la racine et non par ordre alphabétique. Il faut enfin signaler une dernière difficulté et qui n'est certainement pas la moindre : les voyelles brèves que l'on représente par des signes diacritiques ne sont en général pas notées, ce qui rapproche l'écriture standard de l'arabe d'un système *sténographique*.

Des expériences pédagogiques approfondies permettent en effet de *constater* qu'un débutant ne disposant que d'un vocabulaire restreint et de connaissances de grammaire qui, pour n'être pas inexistantes, n'en sont pas moins diffuses, peut néanmoins parvenir à élaborer une sorte de *savoir-faire* lui permettant de progresser efficacement dans l'étude de la langue : il peut par exemple réussir assez rapidement à décoder à l'aide d'un dictionnaire la première page d'un journal [2]. Il nous a semblé dès lors intéressant d'analyser ce savoir-faire (non enseigné

[1] Il faut toutefois avoir présent à l'esprit que l'évolution du processus d'acquisition ne se produit pas toujours de manière « monotone ».

[2] Nous nous sommes, quant à nous, essentiellement limités au niveau de langue tel qu'il peut être relevé dans la première page du quotidien « Al-Ahram ».

AnIsl 33 (1999), p. 75-103 André Jaccarini
Vers une théorie du moniteur syntaxique.

dans les manuels de grammaire), lequel est censé permettre, avec des connaissances très lacunaires, d'atteindre rapidement une *autonomie* suffisante dans l'apprentissage de la langue.

Ce savoir-faire nous a semblé constitué essentiellement :

1. D'un petit ensemble de connaissances *précises*, lesquelles appartiennent cependant à des registres assez divers ;

2. D'un certain nombre d'*intuitions* mais qui ne sont en fait qu'un deuxième ensemble plus flou de connaissances et dont il est dès lors naturel de chercher à préciser les contours et à déterminer la nature de ses éléments ;

3. Et, enfin, une sorte de stratégie acquise par expérience permettant d'utiliser astucieusement les quelques éléments de syntaxe pour optimiser les recherches dans le dictionnaire, laquelle *évolue naturellement en fonction de l'état des connaissances.*

C'est dire que l'une des principales motivations qui nous ont amené par la suite à nous intéresser au traitement automatique de l'arabe relevait à l'origine du domaine de la *cognition.* Il nous semblait en effet que les processus cognitifs mis en jeu lors de la phase de déchiffrement et de décodage, pour peu que l'on cherchât à rendre compte de leur complexité, des interactions avec le lexique, des stratégies optimales de recherches dans le dictionnaire en fonction du niveau de résolutions morphologiques et syntaxiques du texte par l'apprenant, ne pouvaient s'appréhender qu'à l'aide d'une *modélisation*, modélisation dont seule la simulation sur ordinateur – en nous fournissant un critère de réfutabilité au sens de Popper – pouvait nous donner la garantie de son caractère scientifique. (Pour ce qui est du critère de Popper, les épistémologues utilisent plus volontiers le terme de « falsifiabilité » [# vérifiabilité]). Dualement, une telle modélisation pouvait ouvrir un champ particulièrement intéressant d'applications dans le domaine de la didactique : les systèmes d'enseignement assisté par ordinateur dit *intelligent.*

D'autre part, en partant de la constatation que les niveaux de difficultés que représentent les déterminations des racines des formes arabes variaient selon une grande amplitude – en fonction du *mot graphique* considéré – des études sur l'optimisation des algorithmes de tri ont été menées. *Trois critères importants nous semblaient en effet devoir être combinés pour aboutir à un algorithme efficace d'accès au dictionnaire* : la valeur radicale du graphème, la position de ce dernier et la longueur du mot. Nous avons ainsi cherché à affecter à chaque mot graphique une *valeur numérique* reflétant approximativement le degré de complexité que cette chaîne de caractères représente relativement à l'opération d'extraction de la racine ; ce qui nous a permis en procédant globalement sur un texte à effectuer un prétri, à ranger par ordre de difficulté croissante les mots à décoder, à les regrouper en sous-ensembles en vue de mettre en œuvre des procédures « en bloc », par paquet et éviter ainsi les opérations trop répétitives. Le recours à un langage informatique particulièrement bien adapté aux calculs (APL) et quelques astuces de codage nous permirent de convertir en termes algébriques et en problèmes matriciels ce qui relevait à l'origine de questions purement linguistiques. Aujourd'hui encore, ces procédures de tri mises au point nous semblent intéressantes à exploiter pour ce qui est de la conception de préprocesseurs morphologiques.

En tout cas, la réalisation de ces modules nous convainquit de la faisabilité d'un programme d'analyse morphologique ne nécessitant pas le recours à un dictionnaire informatique de taille importante.

AnIsl 33 (1999), p. 75-103 André Jaccarini
Vers une théorie du moniteur syntaxique.

I.2. *Mise en parallèles des processus cognitifs et de l'approche algorithmique (théorie du parsage)*

D'importants travaux en didactique ont été menés par Claude Audebert au sein du DEAC (1982-1986) dont on retiendra essentiellement la *critique* très élaborée de certaines méthodes pédagogiques, classiques, qui contraignent l'apprenant à un déchiffrage mot à mot. De cette approche, il se dégage en effet une idée qui, poussée à l'extrême, semble paradoxale: il vaut mieux faire précéder l'analyse morphologique par l'analyse syntaxique. Or, il est clair que ces préoccupations pédagogiques rejoignent les soucis d'optimisation sur le plan algorithmique. La question de la priorité de la syntaxe sur la morphologie (de faire précéder l'une par l'autre ou *vice versa*) doit en effet aussi trouver sa traduction en *théorie du parsage* des langages en informatique en termes d'opposition entre analyse montante (*bottom-up*) et descendante (*top-down*) et poser des problèmes similaires de compromis optimal.

Il ressortait en tout cas clairement de cette convergence d'intérêt la nécessité de créer, en vue de la simulation des processus cognitifs d'acquisition, des analyseurs *transparents* sur lesquels pourraient être effectuées des expériences.

La méthode pédagogique de Claude Audebert consistait à focaliser l'attention des étudiants sur certains éléments de la phrase à partir desquels des *prédictions* pouvaient être faites et de les vérifier ensuite en accomplissant, si nécessaire, une analyse morphologique. Or, la plupart de ces éléments, des mots-outils en général, se trouvaient être justement ceux qui échappent au système de dérivation morphologique, c'est-à-dire ceux qui ne peuvent être systématiquement ramenés à une racine et à un schème. Dans le traitement automatique de la morphologie, ces éléments nécessitent la mise en action préalable d'une procédure d'isolement afin d'éviter l'échec de l'analyseur. Cette procédure est d'autant plus nécessaire qu'il s'agit – et de très loin – des éléments les plus fréquents. Nous pourrions constater également, dans une optique d'informatique documentaire, leur recoupement avec ce que l'on désigne dans ce domaine par «mots vides» qui, n'apportant qu'une très faible information, induisent *inversement* de fortes contraintes sur leur environnement.

Ainsi, il fut décidé de décrire systématiquement au moyen de réseaux de transition enrichis, mieux connus sous le nom d'ATN (Augmented Transition Network)*, les contraintes induites par certains de ces éléments. Il est important de rappeler ici que la description des environnements linguistiques n'avait pas pour but de répondre aux critères d'adéquation des théories linguistiques tels que peut les définir par exemple Chomsky. Nous ne cherchions pas de construire un indicateur syntagmatique jugé pertinent sur le plan linguistique, mais à fournir un ordre de priorité, une stratégie pour ce qui est de la recherche dans le dictionnaire, laquelle suppose une analyse morphologique; à fournir en quelque sorte l'esquisse d'un modèle «cognitif», c'est-à-dire la description de l'ensemble des procédés et démarches intellectuels s'appuyant sur un ensemble minimal de connaissances, aussi bien lexicales que syntaxiques, un modèle qui nous permette en somme de résoudre ce que j'appellerai *le paradoxe pédagogique de l'arabe*: être contraint d'analyser et d'extraire la racine d'une forme que l'on ignore pour accéder au dictionnaire.

AnIsl 33 (1999), p. 75-103 André Jaccarini
Vers une théorie du moniteur syntaxique.

La description des procédés cognitifs mis en jeu lors de l'opération de déchiffrage ainsi que l'optimisation (algorithmique) étaient donc les deux principales *motivations* qui nous ont poussé à publier, dès 1986, une esquisse de théorie des opérateurs syntaxiques. Les éléments figés, échappant au système de dérivation, les *atomes* du système morphologique auxquels nous venons de faire allusion, y étaient décrits comme des opérateurs induisant des attentes. Ces éléments de description devaient aussi bien servir à construire un futur modèle cognitif qu'à constituer un ensemble duquel pourraient être déduites des informations susceptibles d'être intégrées dans un « moniteur syntaxique » ; la tâche principale de ce dernier étant de guider et d'optimiser un programme morphologique. Nos recherches étant essentiellement tournées vers les applications morphologiques (puisqu'il s'agissait dans un cas de rendre compte de l'interaction apprenant / dictionnaire et dans l'autre d'accélérer un programme morphologique et d'en augmenter les performances), les attentes devaient surtout concerner le niveau superficiel – c'est-à-dire devaient pouvoir être rendues par des réseaux non récursifs ou bien, ce qui est équivalent, par des expressions régulières ayant pour vocabulaire de base uniquement des étiquettes lexicales. C'est ainsi, par exemple, que l'on a traité l'opérateur *relatif*. Cela n'excluait pas la possibilité que les attentes puissent aussi être obtenues par simple réduction, lorsqu'il se trouvait qu'une description plus structurale fût plus aisée à fournir (ce qui est le cas par exemple du « *inna* », de l'opérateur conditionnel) ; ces descriptions ayant alors souvent été obtenues en ayant recours à la méthode *harrissienne* des grammaires en chaînes, c'est-à-dire par la détermination de structures noyau et d'ajouts successifs à ce noyau. Cet aspect soulève toutefois les problèmes de dérécursivation et de transformation de grammaire que nous avons tenu à traiter avec précision afin de dissiper tous doutes concernant la cohérence de la démarche et l'unité de la méthode [3].

1.3. *Méthodologie*

Après avoir mis en évidence les deux motivations fondamentales de ce travail – cognitive et algorithmique – et constaté la possibilité d'une certaine mise en correspondance, nous aimerions maintenant attirer l'attention sur le fait que le recours au formalisme des ATN se justifie entièrement sur le plan méthodologique.

Les réseaux de transition récursifs sont apparus au début des années soixante-dix dans un contexte précis. En effet, la mise au point d'*analyseurs* pour grammaires transformationnelles (c'est-à-dire de procédures inverses permettant d'associer à un texte donné une description structurale conforme à la grammaire proposée) s'est avérée très difficile. Ces grammaires sont surtout conçues comme des mécanismes de production du langage (d'où leur nom de grammaires génératives). Les ATN proposés par Woods constituaient en revanche un mécanisme simple (il s'agissait en fait d'automates augmentés) aussi bien adapté à l'écriture (production) qu'à la lecture (analyse). Ce mécanisme pouvait soit *engendrer* les mêmes

[3] Ces questions sont traitées en détail dans «Grammaires modulaires de l'arabe», dorénavant *GMA*, (à paraître).

AnIsl 33 (1999), p. 75-103 André Jaccarini
Vers une théorie du moniteur syntaxique.

langages que ceux définis à l'aide d'une grammaire transformationnelle, soit les analyser (ils pouvaient donc fonctionner aussi facilement en mode de production que d'analyse). Cependant, ces procédures apparaissaient comme des mécanismes *ad hoc*, dépendant des analyses particulières. Les tests et actions associés aux arcs devaient être prévus et ne pouvaient être définis qu'au cas par cas. D'autre part, aucune précision n'était donnée sur ces actions, lesquelles, si aucune condition ou restriction ne leur était associée, pouvaient donner lieu à des cercles vicieux, ou impasses (les fameux *deadlock* des informaticiens). En fait, les ATN n'ont jamais conduit à une théorie linguistique générale.

Ils ont joué en revanche, comme le notent Miller et Torris, « un rôle important dans le courant d'un certain *réalisme linguistique* qui prônait notamment une relation plus étroite entre la théorie linguistique et la psycholinguistique » ; et c'est ce sur quoi nous voulions insister. Notons également que certains auteurs affirmèrent que les ATN permettaient de simuler les performances humaines, telles qu'elles avaient été observées dans des expériences de psycholinguistique.

II. RÉALISATION DE PROGRAMMES INFORMATIQUES OPÉRANT SANS LEXIQUE ET CADRE THÉORIQUE ASSOCIÉ

Présentation générale

Sur le plan informatique, ce travail pouvait apparaître comme reposant sur une hypothèse quelque peu étrange : nous nous proposions en somme d'optimiser grâce au moniteur syntaxique un programme morphologique dont nous n'avions fait que *supposer* l'existence (les quelques modules mis au point avaient un champ de couverture linguistique très restreint et nous ne disposions d'aucune interface pour les raccorder aux futurs ATN). L'hypothèse fut d'ailleurs encore aggravée par la suite, par la suppression *totale* du lexique. Méthode étrange, mais qu'il serait loisible à présent de rapprocher du paradigme même du raisonnement récursif. Nous nous devions toutefois de passer du stade de l'hypothèse à celui de la réalisation : la formalisation des données morphologiques dans le cadre de la théorie des automates nous a permis de proposer un modèle indépendant du lexique compatible avec le modèle syntaxique.

Un premier programme de segmentation opérant *sans lexique* a donc été conçu et, dès 1990, une interface Macintosh a été mise au point permettant l'exploitation de ce programme. Ce dernier laissait toutefois subsister une marge d'erreur. Il fallut donc affiner les modèles linguistiques en sorte qu'ils puissent rendre compte de l'ensemble des phénomènes de concaténations, de permutations et qu'ils puissent rendre identifiables les différentes catégories, l'isolement de la racine, la détermination du schème, etc. Ces modèles devaient aussi être extensibles et modulables afin de se donner la possibilité de refléter la complexité réelle des phénomènes morphologiques. Il était nécessaire de se doter en quelque sorte des moyens théoriques permettant de *tendre* vers une analyse exhaustive sans que pour autant la prise en compte des modules de description ne soit cause d'un ralentissement important de l'analyse et surtout de la génération de bruit – qui serait due non à un déficit mais au

AnIsl 33 (1999), p. 75-103 André Jaccarini
Vers une théorie du moniteur syntaxique.

contraire à un *excédent d'informations* [4]. Les règles d'une grammaire sont en effet inégalement productives et il est nécessaire dès lors, si l'on désire construire un analyseur efficace, de privilégier celles qui possèdent un rendement élevé, c'est-à-dire celles qui rendent compte des régularités les plus importantes et les plus fréquentes, tout en ayant présent à l'esprit que la courbe de productivité de la grammaire est elle aussi dépendante de l'application recherchée. Il est clair qu'un simple contrôle orthographique ne requiert pas un analyseur de même nature (ni surtout de même complexité) que ceux requis par un préprocesseur syntaxique, un vocalisateur automatique ou un programme d'EAO. Il faut savoir toutefois que la construction de simples filtres – ou accepteurs, ceux que l'on utilise par exemple pour le contrôle orthographique ou le filtrage de messages – peut aussi dériver de la même description que celle utilisée pour la conception d'applications plus élaborées, la grammaire finale étant dans ce cas obtenue par *réduction* et *transformation* d'un modèle source non déterministe*. Cette dernière méthode étant d'ailleurs souvent la seule praticable tant il peut être difficile de concevoir d'emblée un programme de filtrage, aussi fruste soit-il.

II.1. *Les analyseurs*

Nous avons démontré dans nos travaux les raisons pour lesquelles nous avons cherché à développer nous-même les différents types d'outils nécessaires à la mise en œuvre informatique. Nous n'évoquerons dans cet article que l'une d'entre elles qui nous semble particulièrement importante dans une optique de mise en parallèle des processus cognitifs d'acquisition et d'optimisation algorithmique : *la transparence* des analyseurs. Cette dernière est en effet essentielle pour le développement de notre programme. L'acceptation ou le rejet d'une grammaire dépendra en fait du comportement de l'analyseur*. Cet aspect sera développé dans les paragraphes suivants.

L'analyseur de base que nous avons mis au point procède à une exploration en parallèle (des chemins) et possède la propriété importante d'être *linéaire*. Il a été déduit d'un simple calcul dans une structure algébrique rigoureuse où ses principales propriétés peuvent être clairement établies. La définition de cette structure algébrique nous aura permis – outre la possibilité de procéder aisément à des transformations de grammaires desquelles peut dériver une mise au point élégante et concise de programmes (cf. *infra*, méthode de variation de grammaire) – de donner une *signification* simple à la gestion de la « pile » à double entrée qu'implique le fonctionnement de l'analyseur. En effet, les astuces de fonctionnement de ce dernier peuvent ne pas apparaître évidentes de prime abord. Quant aux versions évoluées qui permettent de prendre en charge des grammaires augmentées (ou transducteurs : cf. *infra*) non déterministes sous mode interprétatif ou semi-interprétatif, ils ont été obtenus à partir de la première version par enrichissements modulaires.

[4] Il importera de déterminer la nature de ces bruits. Il faut être conscient, d'autre part, du fait que les enrichissements de grammaire sont quelquefois susceptibles d'entraîner une augmentation du bruit bien plus importante que la part de silence qu'ils pourraient réduire.

AnIsl 33 (1999), p. 75-103 André Jaccarini
Vers une théorie du moniteur syntaxique.

Nous n'insisterons jamais assez sur l'aspect de modularité mis en évidence dans ce travail puisqu'il découle naturellement du point de vue minimaliste qui y est systématiquement développé. Il importe, en effet, lorsqu'on adopte cette position, que l'on puisse alors disposer d'une méthodologie permettant des enrichissements aussi *élémentaires* que possibles. Nous reviendrons plus loin sur cet aspect important.

II.2. *Modélisation du système morphologique de base* [5]

Résumé

La modélisation du système morphologique de base à l'aide d'une grammaire régulière n'est possible qu'au prix de certains compromis. La méthode de variation de grammaires permet cependant de les optimiser. Des outils conceptuels et informatiques sont alors nécessaires pour mettre en œuvre cette méthode. Sur le plan théorique, la définition de structures algébriques précises dans lesquelles il sera possible d'effectuer des calculs sur les grammaires en vue de les transformer sera indispensable. Ce cadre théorique permettra également de fournir un fondement à la méthode expérimentale à laquelle on devra avoir nécessairement recours pour la mise au point des modules de description ou la génération d'applications particulières. Sur le plan informatique, cette méthode suppose la définition d'un environnement d'ingénierie linguistique, dont nous avons nous même conçu le noyau, lequel a été programmé en LISP. Ce noyau comporte un éditeur structurel ainsi qu'un atelier de grammaires.

Dans notre souci de mieux cerner aussi bien sur le plan algorithmique que cognitif la notion de *minimalité*, de mieux faire ressortir par ailleurs la spécificité du système morphologique, nous nous sommes attachés à isoler un noyau de régularité de base et à expliciter les principales conséquences théoriques de cet *isolement* [6].

Le système de base peut être décrit par un langage régulier, c'est-à-dire qu'il est représentable par des automates finis (lesquels, rappelons-le, représentent le paradigme le plus simple de calcul). Cette formalisation n'est toutefois possible qu'au prix de certains compromis dont il conviendra alors d'évaluer rigoureusement le coût, en termes de bruits, de silences, d'indéterminisme, de diminution de la pertinence linguistique et enfin de temps d'exécution du programme: coût que l'on cherchera justement à minimiser.

[5] Par système de base nous entendons la partie de la morphologie qui traite des mots dont la racine est constituée exclusivement de triplets consonantiques. Les mots construits à partir de schèmes dont les suffixes comportent des *hamza* sont également exclus de ce noyau, ex: *saḥrā'*.

[6] C'est ainsi qu'on a été amené à définir rigoureusement les notions d'invariance, de langage et de syntaxe *quotients*. En effet, la caractéristique majeure du système morphologique de base nous semble être le fait que les «opérateurs» de catégorisation et de projection (ou réduction du mot à son paradigme) commutent, autrement dit il est indifférent de catégoriser d'abord et de projeter ensuite ou *vice versa*. Cette propriété, évidente dans le cas de la morphologie, peut être étendue à la syntaxe en vertu du *principe d'invariance*, lequel fournit le premier fondement théorique de la théorie du moniteur. Ce principe est de nature limitative puisqu'il impose une borne inférieure à la catégorisation: on ne tient compte que des catégories qui demeurent *invariantes* par projection.

AnIsl 33 (1999), p. 75-103 André Jaccarini
Vers une théorie du moniteur syntaxique.

II.2.1. *Le principe de variation de grammaire*

Le procédé *de variation de grammaire* nous permettra de dégager certains critères d'évaluation, mais il importera par la suite d'en établir une *théorie* ou tout ou moins une méthode rigoureuse. Cette méthode devant permettre d'*intégrer* les différents critères identifiés lors de la phase d'expérimentation en sorte de pouvoir attribuer à chacune de ces grammaires une *valeur*, cette dernière n'étant pour l'instant établie que de manière empirique, bien que des travaux importants aient été déjà effectués dans ce sens [7]. Ces compromis sont en effet inévitables en raison notamment de la richesse du système de dérivation.

Si cette *théorie* venait à prendre forme, elle constituerait alors une « métrologie des grammaires [8] ».

II.2.2. *La nécessité d'une interface linguistique*

En fait, au-delà de ces considérations d'ingénierie linguistique – sur lesquelles nous aurons à revenir, tant les problèmes de variation de modèles et de recherches de compromis optimaux se sont révélés cruciaux dans notre étude axée principalement sur les principes de minimalité et d'« économie intellectuelle » – la reconstitution de toutes les tentatives d'analyse, c'est-à-dire de tous les parcours dans l'automate, les culs-de-sac, est d'une importance capitale si l'on désire établir un parallèle quelque peu pertinent avec le processus cognitif d'appréhension. Toutes les tentatives de parcours dans l'automate, associées aux différents états de la *double pile,* seront ainsi mises en relation avec les hypothèses linguistiques correspondantes. Mais l'amélioration de l'interface linguistique – qui irait beaucoup plus loin qu'une simple interprétation des fragments de parcours – sera nécessaire si l'on désire mener cette étude de manière précise. Ces développements n'ont été qu'esquissés. Lors de la construction d'une grammaire, il importe en effet d'être très attentif au coût de chacune de ses variables (= catégories). La considération de tous les états de la double pile permettra de se rendre compte que le rapport qualité/prix n'est pas toujours facile à établir à moins que l'on ne dispose d'une méthode rigoureuse d'évaluation. La considération systématique des hypothèses linguistiques correspondant aux différentes tentatives (dont plusieurs sont absurdes) nous aidera cependant à établir une telle méthode. Signalons également qu'à partir d'un repérage systématique des hypothèses improductives, des critères d'élagage pourraient être dégagés.

La conception d'un modèle morphologique indépendant du lexique qui soit compatible avec les représentations des données syntaxiques a donc nécessité un travail important sur le plan méthodologique. En effet, l'étude systématique des ambiguïtés, bruits et silence qu'engendrent les analyseurs fait clairement apparaître qu'il est essentiel de se donner la possibilité

[7] Voir notamment l'application développée par Christian Gaubert, « Présentation de Sarfeyya, un logiciel expérimental d'analyse morphologique de textes arabes », *L'Astrolabe, le semestre de l'AFEMAM* 1, 1997, p. 119-123, et son travail portant sur la hiérarchisation des bruits et silences dont certains résultats sont rapportés dans « Analyse d'un texte arabe par ordinateur : méthode d'évaluation, résultats », *AnIsl XXIX*, 1995. p. 283-311, ainsi qu'une première tentative de classification des ambiguïtés que l'on trouvera dans *GMA* (chap. 7 section 2), laquelle, sans être exhaustive, établit néanmoins le coût « raisonnable » de ces compromis.

[8] Puisque dès lors il serait possible de calculer leurs distances – dans un espace abstrait – à partir de leur valeur respective (cf. ci-dessous § « Un espace abstrait de grammaires »).

AnIsl 33 (1999), p. 75-103 André Jaccarini
Vers une théorie du moniteur syntaxique.

de changer « facilement » de point de vue. La méthode *de recherche de l'algorithme optimum par variation de la grammaire* se révèle en effet l'une des plus efficaces et c'est cette constatation qui nous a amené d'ailleurs à concevoir le noyau d'un environnement de génie linguistique qui comporte un éditeur structurel et un atelier de grammaires.

Toutefois, on démontre qu'un grand nombre d'inconvénients découlant de ces compromis peuvent être réduits si l'on s'autorise l'enrichissement des automates de base [9] (cf. p. 6, « Réalisation de programmes informatiques... »).

II.3. *Le système général défini comme une transduction du système de base*

Quant au système morphologique général, il est défini comme une transduction finie du système de base (d'où la possibilité théorique de le décrire comme une image fonctionnelle d'un noyau minimal). Il est alors nécessaire d'étendre le principe de variation aux transducteurs (i.e. aux grammaires augmentées).*

II.3.1. *Le principe de variation étendu aux grammaires « augmentées »*

La méthode de variation de grammaires permet donc de limiter les conséquences négatives des approximations auxquelles on a obligatoirement recours pour la description du système de base. Bien plus, nous avons montré que grâce à l'enrichissement des grammaires par des actions *élémentaires* (introduction de tests et de drapeaux), il était possible dans de nombreux cas d'*éliminer* certains de ces inconvénients. Dans notre schéma théorique fondamental, nous ignorerons, à ce niveau, les « augmentations ». Ces dernières seront toutefois nécessaires pour la description du système général.

L'existence de racines comportant des *semi-consonnes* introduit des ruptures dans la régularité du système, qui en fait sont dues pour l'essentiel à des phénomènes d'effacement et de permutation. Nous avons donc dû faire évoluer les automates décrivant le système de base pour les transformer en des objets similaires aux ATN de Woods [10]. Toutefois, le « squelette » de ces « réseaux de transition », c'est-à-dire ce qui en reste lorsqu'on les débarrasse des tests et actions introduits, est toujours un automate fini et non pas comme dans le cas général un automate récursif. De plus, les actions introduites doivent obéir à des conditions strictes en vue de préserver les propriétés mathématiques générales des grammaires régulières, ce qui nous permettra d'étendre le principe de variation à ces grammaires augmentées, équivalentes à une classe de transducteurs dont feront partie ceux décrivant le système général ainsi que le vocalisateur. On pourra ainsi toujours envisager des procédures simples pour les rendre déterministes ou bien procéder à des *remodélisations* linguistiques si le besoin s'en fait ressentir.

[9] En se donnant la possibilité de définir des tests et des actions le long des arcs.

[10] On pourrait exprimer naturellement ces nouvelles règles à l'aide d'une *grammaire contextuelle*, mais au prix de l'abandon de notre logique fondamentale. En effet, dans la hiérarchie des grammaires de Chomsky, nous nous sommes toujours efforcés de rester au niveau *le plus bas* afin de respecter autant que faire se peut les principes de simplicité et d'économie. Les grammaires régulières possèdent d'autre part des propriétés mathématiques que nous exploitons systématiquement dans nos procédures de transformation des grammaires. Nous avons donc choisi de conserver la logique de notre description initiale, celle des automates finis que nous avons cependant *enrichis*.

AnIsl 33 (1999), p. 75-103 André Jaccarini
Vers une théorie du moniteur syntaxique.

II.3.2. *Le postulat de régularité*

Le principe même de la distinction entre système morphologique de base et système général nous semble justifié par l'existence d'un sous-langage consistant de la langue arabe – en fait l'ensemble des phrases dont toutes les occurrences peuvent être ramenées à des racines saines et à des atomes – qu'on peut décrire par la grammaire de base G_0. Il est légitime dès lors de faire apparaître la langue arabe L comme une *extension* de $L(G_0)$ – et non seulement comme un sur-ensemble. Nous voulons dire par là que nous chercherons à déterminer G en sorte que $L(G_0) \subset L = L(G)$ et telle que G_0 soit un module noyau de G, c'est-à-dire telle que cette dernière soit réductible à G_0 par suppression de certains paramètres. Nous chercherons donc à enrichir G_0, *de la manière la plus simple* en sorte que L(G) puissent être déduit par une correspondance homomorphique de $L(G_0)$. *La langue arabe apparaîtra ainsi comme la transduction finie du langage de base* $L(G_0)$. Cet énoncé constitue en fait une formalisation de ce que certains linguistes appellent le *postulat* de régularité) [11]. Pour reconnaître L(G), nous conserverons donc l'automate de base G_0 que nous augmenterons de tests et d'actions, ces dernières ne devant pas remettre en question la nature homomorphique de la correspondance. La distinction introduite entre système de base et système général aura ainsi pour principale conséquence méthodologique le recours à un réseau de transition augmenté *non récursif.*

Nous savons que ces réseaux de transition peuvent être traduits en grammaires à attributs sémantiques ; les contraintes que nous définirons seront telles que seuls apparaîtront dans ces grammaires des attributs synthétisés et que par conséquent la construction de l'arbre « sémantique » sera « homomorphique » à celle de l'arbre syntaxique : les actions seront exécutées parallèlement au flux de lecture ce qui entraînera entre autre l'absence de problème de « *backtracking* ».

Mais l'évolution du modèle nécessite naturellement celle de l'analyseur. Nous avons donc mis au point de nouvelles versions qui permettent de prendre en charge des grammaires augmentées, ainsi que des modules annexes permettant de vérifier la cohérence des actions introduites (voir *GMA*).

Enfin, il est important de signaler que ces possibilités de définitions et de mise en œuvre de *transducteurs* non déterministes nous permettent également de réaliser des modules de *vocalisation automatique**. Cependant, ceux que nous avons réalisés, dont le fonctionnement ne nécessite pas l'introduction d'un lexique, doivent être compris pour l'instant comme des *outils d'évaluation* de nos modèles de base, devant nous permettre notamment de préciser la nature de la correspondance entre le langage de base $L(G_0)$ et le langage réel. Rappelons, à ce propos, que sur le plan informatique, nous ne cherchons pas à réaliser directement des applications, mais à les engendrer grâce à un système plus général [12].

[11] Cf. « Grammaire de l'arabe d'aujourd'hui », D.E Kouloughli, 1994, p. 313.

[12] D'où l'atelier de grammaires.

AnIsl 33 (1999), p. 75-103 André Jaccarini
Vers une théorie du moniteur syntaxique.

II.4. *Unification des points de vue syntaxique et morphologique*

La généralisation du principe de variation aux grammaires augmentées nous fournira également les outils théoriques nécessaires pour procéder à l'unification des points de vue syntaxique et morphologique. Cette unification peut en effet se révéler délicate car il est difficile d'isoler d'emblée les contraintes syntaxiques dont la représentation serait susceptible d'optimiser le programme morphologique.

S'il semble clair que le moniteur ne doit s'appuyer que sur des règles ne violant pas le principe d'invariance – afin de s'assurer de son autonomie par rapport au lexique [13]– l'élaboration *directe* de telles règles revient à construire des fragments de grammaire du langage squelette L/RAC (voir note précédente), ce qui peut s'avérer problématique en ce sens qu'on est naturellement amené à raisonner dans L, non dans le langage quotient L/RAC. Toutefois, la méthode de variation des grammaires nous autorisera à modéliser dans un premier temps en s'appuyant sur des représentations naturelles, le principe d'invariance nous contraignant ensuite à considérer toutes les invariances que présentent les fragments de grammaire proposés relativement à la projection L → L/RAC, en vue de les *modifier*.

La méthode de variation nous permettra, d'autre part, de construire les fragments de grammaire, aussi bien syntaxique que morphologique, sans que l'on ait le souci *a priori* de la consistance de leur réunion. Les problèmes de cohérence, de circularité, de redondance, d'optimisation, pourront alors être pris en charge ultérieurement par des procédures de contrôle de type algébrique [14]. Cette généralisation supposera naturellement un enrichissement des outils informatiques.

II.5. *Justification de notre méthode de travail*

On notera au passage que notre méthode de travail, qui a consisté dans un premier temps à supposer *résolu* le problème de l'analyse morphologique pour aborder d'emblée la description de modules syntaxiques – ayant pour fonction précisément d'optimiser cette analyse – se trouve ainsi justifiée sur le plan pratique. Cette description a consisté essentiellement à attacher aux atomes morphologiques des opérateurs syntaxiques. Ces derniers sont donc vus comme des entités déclenchant des attentes, auxquels nous avons cherché à associer des langages pouvant être décrits par des automates de plus bas niveau afin d'avoir toujours la possibilité de les rendre déterministes (sous peine de remettre en question notre objectif d'optimisation des procédures de reconnaissance morphologique).

[13] Ce qui revient à raisonner sur des phrases squelettes obtenues à partir des phrases réelles en réduisant toutes les racines à un seul représentant, d'où la notion de langage (et syntaxe) quotient que l'on dénote L/RAC. L'étude des principales invariances de la projection L → L/RAC constituera alors le noyau à partir duquel pourra se constituer la théorie du moniteur.

[14] En outre, on aura ainsi les moyens de modéliser linguistiquement tout en conservant une marge suffisante de liberté relativement aux contraintes de type algorithmique, puis de *calculer* la grammaire adéquate, la mieux adaptée à l'application recherchée.

AnIsl 33 (1999), p. 75-103 André Jaccarini
Vers une théorie du moniteur syntaxique.

En fait, ces premières descriptions suggéraient déjà l'idée d'automates récursifs [15] dont tous les arcs terminaux, correspondant normalement à des recherches lexicales, se trouvaient remplacés par des appels de sous-programmes morphologiques. Mais ces derniers devant justement faire appel au moniteur syntaxique – l'automate directeur – pour parvenir à un niveau satisfaisant de résolution, il apparaissait ainsi dès le départ que l'étude de l'interaction entre ces deux niveaux pouvait donner lieu à des difficultés importantes. Rien, en effet, ne nous garantissait *a priori* d'être toujours en mesure d'éviter les cercles vicieux, c'est-à-dire que ne se produisent des situations telles que les deux processeurs morphologique et syntaxique soient en attente l'un de l'autre, ou bien encore des situations absurdes. C'est donc le choix initial de notre méthode de travail (où nous avons le plus souvent privilégié les procédures descendantes) qui nous a amené à nous intéresser à l'aspect algébrique afin de fournir un fondement théorique solide à la méthode de variation. Cette méthode, comme on l'a vu, s'est avérée indispensable à plusieurs niveaux.

En conclusion, on peut affirmer que c'est grâce au cadre algébrique précis que l'on a défini – où l'on peut effectuer des transformations de grammaires – qu'il nous a été possible d'aborder l'aspect syntaxique en négligeant les interactions avec les autres niveaux.

II.6. *Quelques perspectives théoriques et méthodologiques*

On pourrait remarquer que les différents fragments de grammaire associés aux opérateurs pourraient aussi être utilisés pour construire le *noyau* d'une grammaire syntaxique qui serait la grammaire du langage quotient L/RAC. Quant à la grammaire de L, elle pourrait être obtenue par enrichissement de la grammaire de base selon le même principe méthodologique exposé au paragraphe intitulé : « Réalisation de programmes informatiques... ». Le langage L apparaîtrait alors comme la *transduction* du langage squelette L/RAC. Cette manière de procéder nous permettra de faire apparaître toutes les spécificités de L par rapport à L/RAC comme des contraintes et des effets de bords d'un automate sous-jacent. L'étude de la structure d'une telle grammaire, outre le fait qu'elle présenterait l'avantage de bien faire ressortir toutes les invariances de la transformation L→ L/RAC, offrirait à notre sens de bons principes méthodologiques pour une construction *modulaire* des modèles syntaxiques.

Une fois décrits les différents fragments de grammaire, (c'est-à-dire celles des sous-langages de L/RAC induits par les différents opérateurs syntaxiques), le problème sera ensuite de savoir comment les regrouper et les unifier en une seule grammaire globale. C'est alors que les méthodes algébriques, auxquelles nous avons fait allusion, révéleront toute leur efficacité. Il sera en effet nécessaire, afin d'étudier les problèmes de cohérence, d'ambiguïté, de complétude et de redondance de la grammaire finale, d'étudier les intersections, les réunions, les complémentaires, de ces langages dont on sait qu'ils présentent la particularité d'être *stables* relativement à ces opérations [16]. C'est dire finalement que, grâce à la méthode

[15] Identiques aux automates sous-jacents des ATN de Woods.

[16] Il s'agit là de quelques propriétés mathématiques des langages réguliers.

algébrique, les quelques descriptions partielles que nous avons réalisées nous seront également utiles si l'on désire, à l'avenir, synthétiser à partir de ces fragments une grammaire globale, que l'on pourra ainsi atteindre par approximations successives.

II.7. *Le stade actuel du moniteur syntaxique*

Nous avons vu que le caractère sténographique du système d'écriture standard de l'arabe, lequel autorise par surcroît des concaténations d'éléments en nombre assez important, ainsi que la richesse du système de dérivation, imposent le recours à la syntaxe si l'on désire parvenir à un niveau satisfaisant de résolution morphologique, et ce, malgré la stabilité remarquable du système de base. De plus, les automates représentant respectivement les morphologies nominales, verbales et celles – uniquement externes – des atomes [17] ne sauraient être montés *en parallèles* en vue d'obtenir un analyseur morphologique (fonctionnant dans l'ignorance totale de son environnement). L'indéterminisme risquerait en effet d'atteindre un seuil prohibitif. Sur un plan strictement algorithmique, il est donc nécessaire de concevoir un moniteur dont la fonction principale sera celle d'un *aiguilleur*.

Ce moniteur pourrait être conçu comme un automate de recherche des atomes. Après un balayage rapide du texte, cet automate aura repéré toutes les formes échappant au système de dérivation morphologique. Il associera également à certaines occurrences des étiquettes (correspondant *grosso modo* à leur statut syntaxique), qui seront autant d'appels de fragments simplifiés de grammaire issus des modèles de base [18]. Un automate simplifié dont le niveau d'indéterminisme serait très peu élevé serait suffisant pour décoder la séquence ; *le gain en rapidité serait considérable*. Sans avoir à aborder d'emblée les problèmes précis de modélisation du moniteur, il faudra toutefois remarquer que toutes les descriptions qui seront proposées devront toutes posséder la propriété d'être réductibles à des automates déterministes [19]. En fait, la modélisation dans le domaine syntaxique offrira un champ de discussion encore plus vaste que celui qui a déjà été abordé par nous au niveau morphologique. Si nous avons déjà cerné la nature des principaux problèmes qui se posent, leur formulation précise suppose la possibilité d'unification des différents modèles et surtout la construction d'outils informatiques d'ingénierie linguistique. La construction d'un analyseur *transparent* du groupe nominal opérant sans lexique constituera donc le principal objectif à l'étape suivante. Cet analyseur (nécessairement modulable) doit être compris comme l'outil de base nous permettant d'effectuer des recherches au niveau des *stratégies*.

[17] Ces derniers peuvent en effet se concaténer à différents éléments et il y a donc lieu de procéder à leur catégorisation en fonction de leurs possibilités de concaténation et de mettre au point une méthode de construction des automates de reconnaissance.

[18] Un atome du type *«inna»*, s'il est non concaténé à un pronom possessif, est nécessairement suivi d'un nom déterminé (par l'article ou l'anaphorique). Il est donc inutile pour analyser l'occurrence suivante d'appeler l'automate représentant l'ensemble de la description morphologique.

[19] Le vocabulaire de base de ces automates sera constitué d'étiquettes syntaxiques représentant des éléments d'automates morphologiques (donc en fait des appels récursifs de sous-automates morphologigues).

AnIsl 33 (1999), p. 75-103 André Jaccarini
Vers une théorie du moniteur syntaxique.

Nous cherchons donc à constituer un ensemble de grammaires dérivant d'un même modèle, plus précisément à se donner la possibilité d'extraire du modèle de base plusieurs sous-grammaires, dont le niveau d'indéterminisme serait moindre, et d'avoir recours à ces grammaires dans des contextes déterminés. On peut d'ailleurs n'avoir besoin, selon les cas, que des versions déterministes de ces fragments. Le rôle du moniteur syntaxique sera en premier lieu de déterminer ces contextes pour appeler à chaque fois la version adéquate et de gérer l'ensemble des résultats locaux (morphologiques) afin de les valider. Donc, plutôt que d'avoir recours à des versions plus ou moins détaillées, génératrices de bruits et d'ambiguïtés, et d'élaguer ensuite toutes les solutions parasites par une analyse syntaxique d'autant plus coûteuse qu'elle reposera sur ces ambiguïtés, on choisit d'essayer de déterminer d'emblée les principales contraintes du contexte qui nous permettront de « court-circuiter » les hypothèses inutiles au niveau local. Cette stratégie peut se révéler très efficace si la détermination du contexte n'est pas onéreuse.

Il faut toutefois noter que si le cadre théorique de nos recherches de stratégies a été clairement défini, toutes les questions liées à la modélisation du moniteur n'ont pas pour autant été réglées, notamment celle de savoir si l'on a pas intérêt, plutôt que de rechercher *en bloc* l'ensemble des « *tokens* » (ou atomes) pour n'affecter des étiquettes à certaines occurrences « morphologiques » que dans un deuxième temps, de tirer *directement* parti des contraintes syntaxiques induites par le *token*, aussitôt que ce dernier a été repéré.

III. ASPECTS COGNITIFS ET SYSTÈME EXPERT: QUELQUES CONSIDÉRATIONS HEURISTIQUES ET ÉPISTÉMOLOGIQUES

IIII.1. *Un espace abstrait de grammaires*

Tout au long de notre travail, notre objectif, tant au niveau linguistique qu'informatique, n'a jamais été de fournir des modèles figés de la morphologie ou de la syntaxe arabe. Nous avons cherché au contraire à définir un cadre théorique plus général où il serait possible de les faire *varier*. Les « grammaires » n'ont donc jamais été considérées comme des ensembles immuables de règles, mais plutôt comme des *points de vue* – particuliers – sur un ensemble donné : le langage, dont il importe alors de dégager les principaux invariants.

La considération systématique des transformations de grammaires nous a amené naturellement à nous intéresser à leur invariance et à nous poser un horizon théorique, à savoir la définition d'un *espace abstrait* de grammaires sur lequel on pourrait définir des *distances*, lesquelles seraient fonction de l'application que l'on cherche à réaliser [20]. Les différentes distances que l'on aurait ainsi définies auraient toutes en commun, quel que soit leur type, d'être autant de reflets des critères d'évaluation (des grammaires) dégagés au cours de notre

[20] Nous rappelons que l'algorithme optimum est obtenu par variation de grammaire. Dès lors, il nous semble naturel de chercher à construire un espace cohérent de grammaires où il serait possible de les faire varier.

AnIsl 33 (1999), p. 75-103 André Jaccarini
Vers une théorie du moniteur syntaxique.

travail d'expérimentation. Quant au choix du type particulier de distance, il refléterait la manière dont la hiérarchisation de ses critères serait organisée [21]. À toute application informatique correspondent des priorités particulières parmi les différents critères. Un changement de priorité se traduirait dans cette logique par un changement de la norme associée à la distance. Mais il ne s'agit là, nous l'avons déjà dit, que d'un horizon – et non d'un cadre – théorique, qui nous sera cependant fort utile pour conférer un certain *sens* – dans les deux acceptions du terme – à quelques-unes de nos recherches dans le domaine de l'évaluation des modèles afin qu'elles n'apparaissent pas comme une simple quête de recettes et d'astuces. Nous cherchons donc en quelque sorte à nous situer dans un « paysage », où s'inscriraient ces recherches, même s'il s'avérait par la suite que la constitution d'une *théorie* est impossible ou trop complexe à réaliser.

III.2. *Un fondement théorique possible aux constructions du générateur d'applications et du moniteur*

Le regroupement de toutes nos fonctions informatiques en un ensemble organisé a clairement fait apparaître le noyau d'un système, l'atelier de grammaires, où il nous a été possible de recourir à l'idée de « navigations ». Les principaux programmes que nous avons mis au point permettent en effet de mettre en œuvre différents types de procédures, selon les besoins de l'analyse linguistique. Ces procédures peuvent apparaître comme des *combinaisons ordonnées* de modules fondamentaux. Mais, comme nous avons eu déjà l'occasion de le signaler à plusieurs reprises, ce système est naturellement appelé à évoluer pour devenir un générateur d'applications linguistiques. En effet, tous les outils mis au point « ...trouver[ont] alors leur pleine signification dans [ce] système plus général – qui reste à réaliser – où serait défini un langage de spécification non seulement des grammaires mais de l'application recherchée ; le système ayant alors en charge de sélectionner ou de synthétiser la grammaire adéquate ». Mais cet aspect nous intéresse *également* pour ce qui est de la modélisation du moniteur syntaxique, puisque la fonction principale de ce dernier – rappelons-le – est d'optimiser l'analyseur morphologique, en sélectionnant à chaque fois la version de grammaire adéquate, laquelle dérive par *transformation* d'une grammaire générale. C'est dire finalement que cette métrologie des grammaires – vers laquelle nous tendons – pourrait constituer un cadre théorique essentiel dans lequel s'inscrirait non seulement le système de génération automatique d'applications linguistiques, mais également celui de la recherche de stratégies efficaces [22].

Il faudrait en effet imaginer un programme qui, selon la distribution des « *tokens* » dans la phrase, évalue les probabilités de résolution des différentes occurrences et segments de phrases sur lesquels seraient appliqués des sous-modules appropriés d'analyse [23]. L'un des

[21] Puisque le choix du type particulier de distance nous fournit toutes les indications nécessaires sur la manière d'*intégrer* les différents critères d'évaluation, en sorte que l'on puisse affecter à chaque grammaire une *valeur*.

[22] Que le moniteur syntaxique serait en charge de mettre en œuvre.

[23] En cas d'absence de *tokens*, l'analyse s'effectuerait normalement sans recours au moniteur.

intérêts de notre approche est de pouvoir considérer ces modules – qui doivent se combiner entre eux de manière cohérente – comme des fragments de grammaire soumis selon leur complexité à différents types d'analyseurs, également modulables.

III.3. *La recherche de minimalité et le paradigme des transducteurs*

Il nous semble toutefois important de rappeler ici que la *minimalité* d'une description n'entraîne pas, en général, de manière automatique, celle de la procédure inverse d'analyse. Si notre manière de travailler jusqu'alors (définition de grammaires formelles et recours systématique à des transducteurs indépendants du lexique) nous a été dictée par le souci de nous conformer scrupuleusement au *principe d'économie,* il faut être conscient que la mise en œuvre des transducteurs associés ne représente pas *toujours*, même dans le cas de leur optimisation, le procédé le plus simple de reconnaissance – si ce n'est pour le fait que le recours à un automate nous contraint à un ordre *linéaire* de décodage. Dans le cas de la morphologie, cet aspect est particulièrement facile à saisir puisque la longueur d'une forme, laquelle ne peut être déterminée par un automate qu'après sa lecture complète, constitue un critère privilégié de résolution [24].

Attirons donc l'attention, une fois de plus, « sur le fait que le modèle ne préjug(e) en rien des algorithmes qui seront retenus. Il peut être aussi bien considéré d'un point de vue « déclarativiste », c'est-à-dire comme une simple spécification des données linguistiques, que d'un point de vue « procéduraliste » puisqu'à l'aide des différents types d'analyseurs que nous avons mis au point, il peut être directement simulé. Les (applications réalisées) démontrent que sa mise en œuvre informatique peut être assez efficace ».

La recherche de la minimalité ne pouvait s'effectuer que dans un cadre clairement défini. Il était nécessaire avant tout de décrire les régularités de base de la manière la plus rigoureuse possible (cf. §: « Réalisation de programmes informatiques… »). Nous avons ensuite fixé l'ensemble de la description dans le cadre de grammaires dont la complexité demeurait minimale – en sorte que l'on puisse toujours leur associer une classe de transducteurs finis modifiables. Ainsi, en procédant à des enrichissements élémentaires et modulaires du noyau, nous avons fait en sorte que le langage réel puisse apparaître comme une image « homomorphique » du langage *squelette* (ou langage quotient).

Toutefois, si cette minimalité descriptive constituait à notre sens la condition nécessaire pour notre recherche de stratégies optimales de décodage, elle n'en demeurait pas moins insuffisante. Aussi, avons-nous proposé la méthode de *variations* de grammaires en vue de la détermination de l'algorithme optimum, méthode essentiellement expérimentale mais que nous avons pris soin de justifier dans un cadre algébrique rigoureux, en en exposant les fondements.

[24] Il est parfaitement inutile, par exemple, de procéder à une analyse par automate d'une forme non voyellée de longueur trois qui, si l'on fait l'hypothèse qu'elle ne comporte aucune erreur de frappe, se confond nécessairement avec sa racine. Cet argument n'est naturellement pas valable si l'on cherche à mettre au point un programme de filtrage des formes arabes. Ce type de remarque est valable, à l'opposé, pour les mots graphiques de longueurs maximales. L'indéterminisme est en fait maximal pour des formes intermédiaires pour lesquelles le recours aux automates s'impose (principe de l'accordéon).

AnIsl 33 (1999), p. 75-103 André Jaccarini
Vers une théorie du moniteur syntaxique.

Ainsi, le recours à ces paradigmes élémentaires (automates, transducteurs, ...) ne devrait pas exclure d'autres méthodes de reconnaissance. Nos automates, avons-nous dit, peuvent aussi être considérés comme de simples langages de spécification des données linguistiques. Mais ces spécifications sont telles que, grâce aux analyseurs, elles pourront nous servir d'*outils* en vue de la recherche d'autres procédures, éventuellement plus efficaces [25].

Ce cadre théorique et informatique – qui tout en nous permettant
– d'atteindre la minimalité au niveau descriptif,
– d'en tirer le meilleur parti au niveau algorithmique,
– d'envisager enfin, grâce à la transparence des analyseurs, une mise en correspondance avec les processus cognitifs – ne devrait pas non plus nous empêcher de recourir, de manière plus générale, à des paradigmes de recherche complémentaires. Car, s'il était nécessaire d'explorer la composante combinatoire du problème du décodage, qui est essentielle, nous ne devrions pas perdre de vue que l'intérêt porté à cette dernière s'inscrivait dans la perspective d'une mise en parallèle avec le processus d'appréhension.

III.4. *La recherche de paradigmes complémentaires*

Si la contrainte de linéarité stricte, qu'impose la modélisation par automate, peut avoir – dans certains cas – de fâcheuses conséquences au niveau de l'optimalité de l'analyse automatique (nous l'avons vu notamment pour ce qui est de la morphologie), cette contrainte ne doit-elle pas alors être considérablement assouplie, *a fortiori,* dans le cas de la simulation du processus naturel. Un lecteur humain a un champ de vision (de compréhension ?) ne se limitant pas, probablement, à un seul mot (et certainement pas à une seule lettre dans ce mot). Les psycholinguistes se sont beaucoup intéressés à cet aspect du problème et ont conçu des appareils sophistiqués pour mesurer avec précision les mouvements oculaires lors de la phase de décodage. L'aspect de linéarité reste cependant prédominant, notamment dans le cas de la syntaxe ; ce qui nous incite à conserver le modèle des automates (ou ATN). Nous ne pouvons toutefois appliquer ce dernier de manière exclusive en ne recourant qu'à un type figé d'analyseur. Par ailleurs, si, dans le processus naturel, la linéarité semble dominante, mais non exclusive [26], elle n'interdit pas pour autant des retours (*backtrack*) semblant suggérer que des hypothèses probables sont émises le plus tôt possible, le processus ne s'interrompant qu'en cas d'échec (ce qui, soit dit en passant, renforce l'hypothèse de linéarité).

[25] On pourrait ainsi étudier le pouvoir de désambiguisation des *tokens* en modifiant les grammaires de base en sorte de les rendre aveugles à certains critères. On soumettrait ensuite à ces nouvelles versions de grammaires des phrases vides, c'est-à-dire des phrases où toutes les occurences morphologiques (celles ne correspondant pas à des *tokens* ou atomes) seraient remplacées par des séquences de blancs de même longueur que le mot effacé. Le remplissage progressif des occurences vides permettrait de déterminer celles d'entre elles faisant le plus baisser l'ambigüité.

[26] Le cas de la compréhension orale doit naturellement être distingué de la lecture.

AnIsl 33 (1999), p. 75-103 André Jaccarini
Vers une théorie du moniteur syntaxique.

Si nous avons jusqu'ici toujours abordé le problème de la représentation cognitive du processus de décodage à partir de « l'approche combinatoire [27] », il nous semble important à présent, en raison de ce qui a été exposé, de solliciter *directement* le point de vue de l'*expert*, qui, à partir de l'étude de phrases vides, ou semi-vides [28], cherche à en établir une première typologie. Ce travail de longue haleine a déjà été entrepris [29]. Il a pour objectif de déduire un ensemble de règles similaires, pour ce qui est de la forme, à celles que l'on définit dans les systèmes d'intelligence artificielle [30].

Il ne nous semble pas inutile, ici, de revenir sur l'un des fondements linguistiques de notre travail, à savoir la possibilité de définir la morphologie arabe par un langage formel (principe du dictionnaire vide). Cette approche nous a permis de mettre en œuvre des méthodes de décodage (*parsing*), qui dans le cas de la plupart des autres langues ne s'appliquent qu'au niveau syntaxique. Elle nous a conduit naturellement à définir le cadre théorique et algébrique dans lequel pourrait se situer une étude rigoureuse des interactions entre les deux niveaux de langage (syntaxe et morphologie). Elle nous a amené aussi à nous intéresser, à partir de la prise en considération du parallélisme cognitivo-algorihtmique, aux stratégies susceptibles de prévenir les explosions combinatoires. Habituellement la recherche de stratégies revient à définir, à un niveau métasyntaxique un ensemble de contraintes en vue de limiter les combinaisons absurdes ou improbables. Dans notre cas le problème se trouve en quelque sorte décalé vers le bas, car, s'il s'agit bien toujours de définir des métarègles, ces dernières sont situées non pas au-dessus de la syntaxe mais à un niveau simplement métamorphologique. Si, *schématiquement*, on se représentait les connaissances, dont les métarègles seraient le reflet, comme *un ensemble* contenant la morphologie, on pourrait alors poser, en prolongement du principe de minimalité, que cet ensemble doit nécessairement être contenu dans celui représentant les connaissances syntaxiques. La liberté du choix des métarègles se trouverait pour ainsi dire bornée supérieurement par l'ensemble des contraintes syntaxiques [31].

Les métaconnaissances que l'expert aura à justifier, non seulement sur le plan pragmatique mais également au niveau méthodologique, se trouveraient ainsi comprises entre deux ensembles importants de contraintes : la syntaxe et la morphologie. Cet « encadrement » du champ de recherche constitue à notre sens un élément de réflexion important, sur lequel nous aurons tout intérêt à revenir lorsqu'on cherchera à fonder épistémologiquement la démarche. En effet, pour ce qui est de l'écriture des métarègles, un choix trop ouvert dans ce domaine risquerait de susciter la désagréable question de leur contingence.

[27] Nous voulons dire par là que nous nous sommes surtout attachés à récupérer tous les parcours et tentatives de parcours dans l'automate, à les interpréter linguistiquement afin d'en établir une classification par ordre d'absurdité croissante en vue de dégager des critères d'élagage, voir de nouveaux algorithmes plus conformes au fonctionnement de l'esprit humain (lequel, soit dit sans malice, n'est pas forcément optimisé).

[28] Les phrases vides sont celles où l'on supprime les occurrences morphologiques pour ne conserver que les atomes.

[29] On pourra notamment consulter Audebert, Claude, « Syntactic Parser and Deterministic Strategy : a Parallel with Human Decoding Process », *in* : *Proceedings of the 2nd Conference on Arabic Computaninal Linguistics*, Kuwait, 1989, p. 352-365.

[30] Étant entendu que ces régles peuvent aussi faire appel aux grammaires, fragment de grammaires, automates et différents type d'analyseurs que nous avons définis.

[31] Dans notre cas précis il ne s'agit que de contraintes concernant la syntaxe quotient (cf. § « Réalisation de programmes informatiques... »).

AnIsl 33 (1999), p. 75-103 André Jaccarini
Vers une théorie du moniteur syntaxique.

Par ailleurs, le problème ainsi posé ne nous fournirait-il pas un nouvel éclairage sur la syntaxe : cette dernière ne serait plus simplement considérée comme une combinatoire particulière, mais également comme un cas limite des heuristiques à mettre en œuvre pour le décodage morphologique.

Une théorie complète du moniteur implique donc le développement de ce deuxième versant de nos recherches qui, loin d'être indépendant du premier, nous permettra sans doute de faire coexister de manière plus harmonieuse les deux principes auxquels nous avons constamment essayé de nous conformer : *le principe d'économie* et *le réalisme linguistique*.

ANNEXE

Définitions

1. Un **automate fini** est un mécanisme abstrait qui permet la lecture ou l'écriture d'une séquence de symboles définis sur un alphabet donné. Ce mécanisme est doté d'une tête de lecture (écriture) et d'une unité de contrôle qui ne peut se trouver que dans un nombre fini d'états. Cette unité de contrôle peut passer d'un état donné à un nouvel état en fonction du symbole qui a été reconnu par la tête de lecture. Un automate fini se définit par la la donnée de ses états de contrôle – parmi lesquelles il faut distinguer un état initial et l'ensemble des états finaux – et une fonction de transition qui permet de décrire le comportement de l'unité de contrôle : cette fonction devra indiquer, selon l'état dans lequel se trouve cette unité et le symbole lu par la tête de lecture, le ou les nouveaux états dans lesquels devra se trouver la machine.

2. L'automate est dit **déterministe** si à chaque état et à chaque symbole lu, la fonction de transition fait correspondre au plus un état. Dans le cas contraire, ou encore si l'unité de contrôle a la possibilité de passer d'un état à un autre sans lire de symbole (l'automate effectue un saut), il sera dit non déterministe.

Il est commode de se représenter un automate par une table qui décrit la fonction de transition, ou mieux encore, par un diagramme sagittal (le réseau de transition), où les cercles représentent des états, et les flèches étiquetées par des symboles du vocabulaire de base, des transitions d'un état à un autre.

Exemple : Soit l'alphabet V_T = {a, b, c}. L'automate dont la fonction de transition est donnée par le tableau ci-dessous (fig. 1) peut être représenté par le diagramme de la figure 2.

AnIsl 33 (1999), p. 75-103 André Jaccarini
Vers une théorie du moniteur syntaxique.

	d	a	b	c
	q_1	q_2	q_3	0
	q_2	q_2	q_4	0
	q_3	q_4	0	0
	q_4	0	0	q_5
	q_5	0	0	0

Fig. 1.

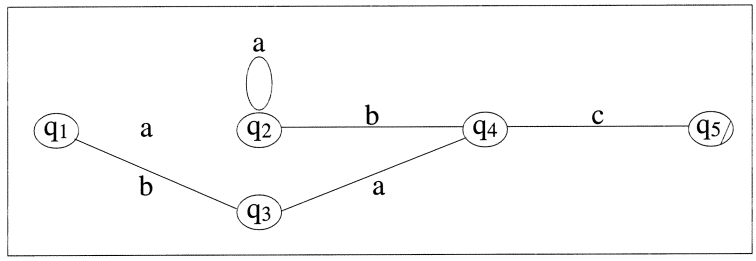


Fig. 2.

3. Il est très facile de simuler le mode de fonctionnement d'un automate déterministe, qui peut être décrit de la manière suivante :

1. L'unité de contrôle se trouve dans l'état initial, la tête de lecture pointe sur le premier symbole ;

2. En fonction du symbole reconnu, l'unité de contrôle effectue la transition adéquate et change d'état ; s'il n'existe pas de possibilité de transition le mot est rejeté ;

3. Si le symbole est accepté, la tête de lecture avance d'un cran. S'il existe encore un symbole à lire, retour à 2 ; sinon, selon que l'unité de contrôle se trouve dans un état terminal ou non, le mot est accepté ou rejeté.

L'automate ci-dessus est déterministe puisque la fonction de transition fait correspondre à chaque état et à chaque symbole lu au plus un état : ainsi au couple (q1, b) la fonction fait correspondre q3, au couple (q1, c) ne correspond aucun état (0 dans le tableau). Cet automate accepte par exemple les séquences 'a b c', 'b a c', ou bien encore 'a a a a b c'– les parcours étant pour chacune d'entre elles :

q_1 —'a'→ q_2 —'b'→ q_4 —'c'→ q_5 ;

q_1 —'b'→ q_3 —'a'→ q_4 —'c'→ q_5 ;

et q_1 —'a'→ q_2 —'a'→ q_2 —'a'→ q_2 —'a'→ q_2 —'b'→ q_4 —'c'→ q_5,

mais rejette 'a a b' qui correspond au parcours ;

q_1 —'a'→ q_2 —'a'→ q_2 —'b'→ q_4... (q_4 n'est pas un état terminal) ;

ou 'a b c b' – dans ce cas l'unité de contrôle parvient à un état terminal mais il reste encore un symbole à lire et il n'existe plus de possibilités de transitions.

AnIsl 33 (1999), p. 75-103 André Jaccarini
Vers une théorie du moniteur syntaxique.

THÉORÈME

Tout langage accepté par un automate fini non déterministe, l'est également par un automate fini déterministe.

De la démonstration de ce théorème, il est possible de déduire un algorithme qui convertit tout automate fini en son équivalent déterministe (nous parlons d'équivalence car la réciproque du théorème est évidente). Mais il ne s'agit là que d'une *équivalence faible* car si les deux machines définissent rigoureusement les mêmes langages, les analyses (*i.e.* les descriptions associées aux suites de symboles acceptées) ne sont pas pour autant identiques. Dans le cas où les deux grammaires produisent les mêmes définitions structurales (par exemple les mêmes arbres syntagmatiques), on parle d'*équivalence forte*.

Nous devons signaler toutefois, que l'on risque d'être confronté à une explosion combinatoire si l'on ne prend garde de concevoir un algorithme qui ne calcule que les états accessibles du nouvel automate. Les états construits constituent en fait un sous-ensemble (généralement très restreint) de l'ensemble des parties de l'ensemble V_n des états de l'automate initial non déterministe. Si $|V_n|$ est le cardinal de l'ensemble V_n, le cardinal de l'ensemble de ses parties est égal à $2^{|V_n|}$. Ainsi, pour un automate d'une trentaine d'états, on obtient un résultat de l'ordre du milliard.

Le programme DET que nous avons écrit calcule de proche en proche les ε-clôtures ainsi que les ensembles des états accessibles. C'est grâce à ce programme que l'on peut construire par exemple des filtres déterministes des formes arabes à partir de descriptions linguistiques très simples mais non déterministes.

En effet, la conception directe d'un programme de filtrage est difficile, même si l'on ne cherche qu'à refléter les contraintes de liaison les plus grossières. Pour se fixer les idées, un automate synthétisant certaines informations concernant uniquement la classification des graphèmes arabes en trois ensembles non disjoints, selon le critère de valeurs radicales et leurs distributions, peut contenir jusqu'à une centaine de transitions (instructions), alors que le modèle originel découlant d'une description *naturelle* mais non déterministe en comporte moins d'une dizaine. De plus, ces transitions sont pertinentes, sinon sur le plan linguistique du moins sur le plan conceptuel, c'est-à-dire que si l'on ne peut toujours leur attacher en toute rigueur une *catégorie linguistique*, ces transitions n'en demeurent pas moins significatives : elles peuvent par exemple représenter un graphème susceptible de figurer en position antéfixée par rapport au segment radical. Le problème est que seul le premier automate est directement programmable puisqu'il représente une suite déterministe d'instructions alors que le second – plus pertinent sur le plan conceptuel – et *apparemment* plus simple, puisque beaucoup plus compact, requiert en raison de son non-déterminisme (la lecture d'un symbole ne détermine pas de manière univoque l'état suivant dans lequel doit se trouver la machine), pour être simulé sur ordinateur, un analyseur c'est-à-dire un programme qui puisse explorer tous les chemins possibles dans l'automate et restituer aussi bien les tentatives réussies qu'avortées.

AnIsl 33 (1999), p. 75-103 André Jaccarini
Vers une théorie du moniteur syntaxique.

Malheureusement, la plupart des automates ainsi obtenus ne sont pas « pertinents » sur le plan linguistique. Nous dirons qu'un automate est pertinent sur le plan linguistique s'il est tel qu'à chacun de ses états, ou bien à chacune de ses transitions, on puisse associer une catégorie linguistique. Or, si comme nous l'avons vu, la restitution des catégories est essentielle pour certaines applications – pour l'analyse syntaxique par exemple –, elle ne l'est pas forcément pour des applications plus simples (construction de filtre, contrôleur orthographique).

5. En revanche la simulation des automates non déterministes n'est pas aussi simple puisque l'unité de contrôle est susceptible de se trouver simultanément dans plusieurs états après la lecture d'un symbole ou bien encore d'effectuer des transitions instantanées (sans lecture de symbole) s'il existe des instructions de saut.

Un **analyseur** est un programme qui accepte comme entrées la description de l'automate (en général sous forme d'une table de transition) et la chaîne à analyser. Le programme devra calculer tous les parcours possibles dans l'automate au fur et à mesure qu'avance la tête de lecture. Pour ce faire, les chemins pourront être explorés soit en parallèles, donc simultanément, soit en série, c'est-à-dire l'un après l'autre ; dans ce cas, il faudra prévoir un mécanisme de « *backtrack* » (retour arrière). L'analyseur devra en outre gérer les transitions intanstanées. Il importe d'être attentif à la complexité du programme, plus particulièrement au temps nécessaire à son exécution.

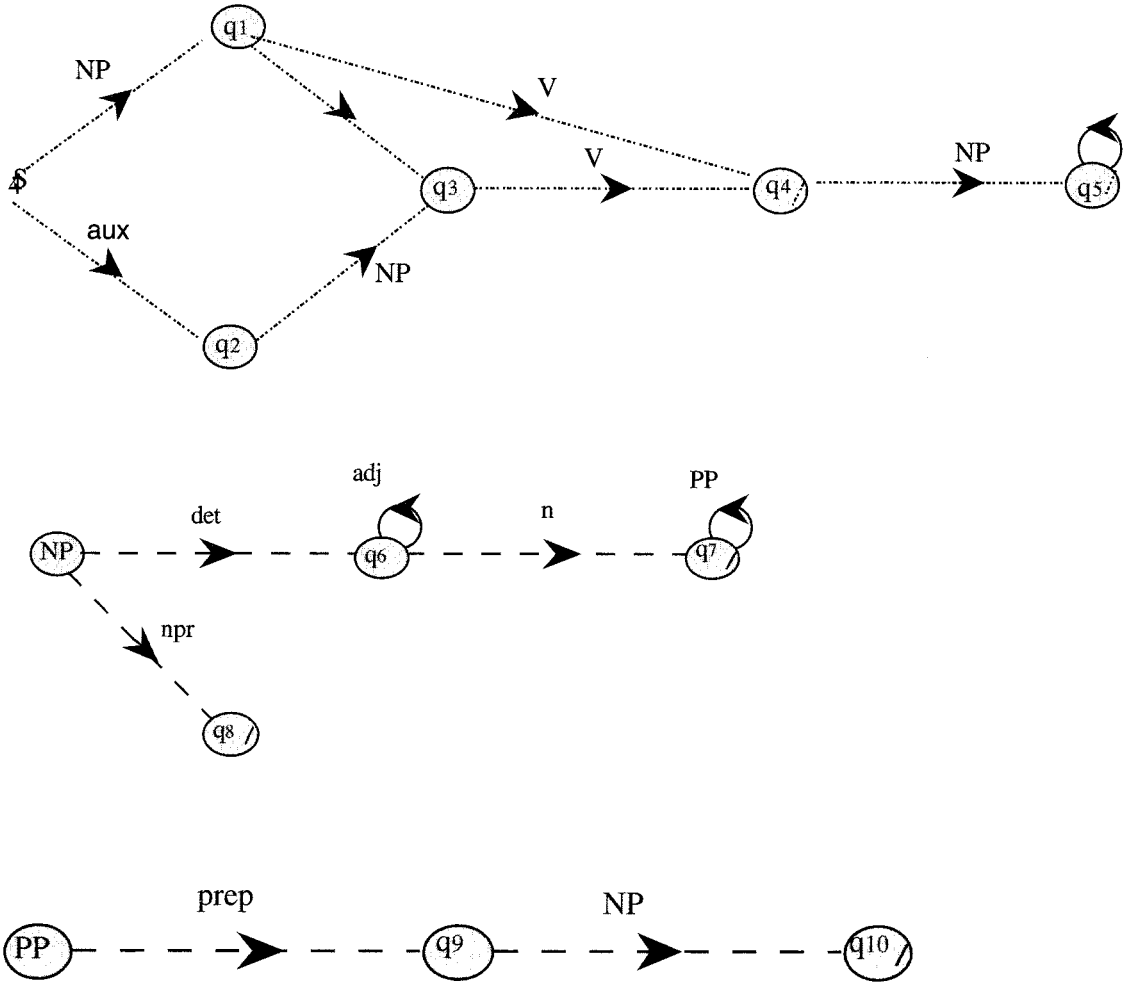
6. Nous avons vu que les automates finis pouvaient être représentés par des réseaux de transition dont tous les arcs sont étiquetés par des symboles du vocabulaire terminal. Les capacités de reconnaissance de ces réseaux admettent certaines limites : ils ne peuvent reconnaître par exemple les structures auto-enchâssées. On pourrait imaginer cependant de renforcer ces mécanismes en y ajoutant la récursion, c'est-à-dire en donnant la possibilité à l'unité de contrôle de suspendre le contrôle, et de faire appel à un autre réseau de transition (éventuellement lui-même). Le réseau obtenu est alors **un réseau de transition récursif**. Un tel réseau se présentera comme un réseau de transition fini, sauf que certains de ses arcs pourront être étiquetés par des symboles non terminaux. Ces symboles représentent des noms d'états, plus précisément le nom des états initiaux des réseaux de transition appelés. L'interprétation d'un arc étiqueté par un symbole non terminal (par commodité, on dira aussi un arc récursif) est la suivante : l'unité de contrôle enregistre l'état sur lequel pointe cet arc dans une pile de mémoire, et le contrôle est momentanément suspendu et transféré au réseau de transition dont l'état initial est cité sur l'arc. Lorsque dans le réseau appelé on rencontre un état final, alors la pile peut être déchargée d'un état, et le contrôle du réseau principal se retrouve donc dans l'état qui figurait au sommet de la pile.

Nous allons tout de suite illustrer le fonctionnement d'un tel mécanisme. L'exemple traité est tiré de Woods, Transition Network Grammars for Natural Language Analysis (Association for Computing Machinery, Inc., 1970). La définition des ATN figure à la section 7. Nous ne nous intéresserons ici qu'au réseau *sous-jacent*, constitué de trois automates pouvant s'appeler récursivement.

AnIsl 33 (1999), p. 75-103 André Jaccarini
Vers une théorie du moniteur syntaxique.

Nous décrivons ci-dessous la suite des mouvements des différents automates lors de la lecture d'une phrase. À chaque mouvement est associé un changement de configuration de l'automate. Une configuration est un triplet (x, y, z) où x, y, z représentent respectivement l'état dans lequel se trouve l'unité de contrôle, la chaîne courante et l'état de la pile. La pile à l'état initial contient le symbole ε qui représente le mot vide. Une chaîne de caractères est acceptée si l'on parvient à une configuration du type (q_n, ε, ε) avec q_n appartenant à F (ensemble des états finaux de l'automate directeur). Nous avons souligné les configurations que l'on obtient par application d'une instruction de saut. Dans ce cas, elles correspondent à tous les appels récursifs, et aux déchargements de piles correspondants.

Woods a recours au réseau de transition présenté ci-dessous pour décrire un petit sous-ensemble de l'anglais. Ce réseau accepte des phrases du type «*John washed the car*», ou bien «*Did the big fat man with a red hat drive the green car ?*»



AnIsl 33 (1999), p. 75-103 André Jaccarini
Vers une théorie du moniteur syntaxique.

La seule suite de configurations possible correspondant au processus d'acceptation de cette dernière phrase est la suivante :

(S, « *Did the big fat man with a red hat drive the green car ?* », ε)
⇒ (q_2, « *the big fat man with a red hat drive the green car ?* », ε)
⇒ (NP, « *the big fat man with a red hat drive the green car ?* », q_3) ;
1er appel récursif de NP
⇒ (q_6, « *big fat man with a red hat drive the green car ?* », q_3)
⇒ (q_6, « *fat man with a red hat drive the green car ?* », q_3)
⇒ (q_6, « *man with a red hat drive the green car ?* », q_3)
⇒ (q_7, « *with a red hat drive the green car ?* », q_3)
⇒ (PP, « *with a red hat drive the green car ?* », q_7 q_3) ;
2e appel récursif de PP (à partir de NP)
⇒ (q_9, « *a red hat drive the green car ?* », q_7 q_3)
⇒ (NP, « *a red hat drive the green car ?* », q_{10} q_7 q_3) ;
3e appel récursif de NP (à partir de PP)
⇒ (q_6, « *red hat drive the green car ?* », q_{10} q_7 q_3)
⇒ (q_6, « *hat drive the green car ?* », q_{10} q_7 q_3)
⇒ (q_7, « *drive the green car ?* », q_{10} q_7 q_3)
⇒ (q_{10}, « *drive the green car ?* », q_7 q_3) ;
1er déchargement de la pile correspondant au 3e appel récursif
⇒ (q_7, « *drive the green car ?* », q_3) ;
2e déchargement de la pile correspondant au 2e appel récursif.
⇒ (q_3, « *drive the green car ?* », ε) ;
3e déchargement de la pile correspondant au 1er appel récursif.
⇒ (q_4, « *the green car ?* », ε)
⇒ (NP, « *the green car ?* », q_4) ;
nouvel appel récursif de NP.
⇒ (q_6, « *green car ?* », q_4)
⇒ (q_6, « *car ?* », q_4)
⇒ (q_7, ε, q_4)
⇒ (q_5, ε, ε)
déchargement de la pile correspondant au dernier appel récursif

la phrase est acceptée.

Cet exemple est intéressant car nous pouvons montrer *que le réseau récursif qui y est décrit est réductible à un automate fini,* lequel peut-être à son tour transformé en automate déterministe (cf. § « Annexe »). Il en sera de même de tous les réseaux récursifs (automates récursifs) que nous serons amené à concevoir pour capter certains types de régularités syntaxiques pouvant être utilisées à des fins d'optimisation du programme morphologique (le moniteur syntaxique).

AnIsl 33 (1999), p. 75-103 André Jaccarini
Vers une théorie du moniteur syntaxique.

7. Les réseaux de transition enrichis de Woods ou **ATN** sont des « automates récursifs » que l'on enrichit en attachant aux arcs des tests et des actions. Cela revient à introduire des restrictions sur l'*application* de chacune des règles de la grammaire et à prévoir ensuite des actions supplémentaires qui devront accompagner l'opération d'application de la règle. La gestion de ses actions, qui peuvent donner lieu à des conflits et des cercles vicieux, fait l'objet d'une partie importante de la théorie des langages et du *parsing*.

Cette technique n'est réellement intéressante que si le réseau de base est déterministe ; sinon, on est obligé d'avoir recours à des analyseurs du type de celui d'Earley (1968, 1970) pour grammaire indépendante du contexte. Dans ce genre de cas, un grand nombre de structures parasites sont créées en cours d'analyse. Un bruit considérable (c'est-à-dire, une ambiguïté inutile) peut résulter de cet état de chose et l'on a alors besoin de recourir à des stratégies d'analyse pour contraindre l'analyseur. Ces stratégies, si elles peuvent dans de nombreux cas *éliminer ou réduire* le bruit, ralentissent malheureusement encore plus l'analyse, leur tâche étant d'écarter ou d'élaguer les arbres partiels obtenus par l'analyseur. Nous retrouvons donc au centre de toutes ces questions, la question du *déterminisme*.

Dans l'exemple de la section 6, nous n'avons traité que le réseau sous-jacent. La méthode de transformation des grammaires qui nous a permis d'aboutir à un automate fini déterministe équivalent (faiblement) à la grammaire récursive de Woods n'est valable que pour les automates non augmentés. Dans le cas où l'on tient compte des enrichissements de la grammaire, il faudra alors prévoir une *extension* de la méthode, laquelle n'est valable que si les actions obéissent à des contraintes strictes.

8. Il importe donc de distinguer trois niveaux :

1. La transformation des réseaux de base ainsi que leur complexification ;
2. Leur augmentation, en attachant à chaque transition des tests et des actions ;
3. Enfin, la transformation de ces réseaux enrichis.

9. Un **transducteur fini** est constitué d'un automate fini sous-jacent qui a la possibilité à chaque changement de configuration d'émettre une suite de symboles. Le mécanisme de reconnaissance (du langage d'entrée) se trouve ainsi associé à la production d'un nouveau langage (le langage de sortie). Nous aurons recours à cette technique pour décrire le système morphologique général à partir du système de base, ainsi que le **vocalisateur automatique** fonctionnant sans recours au lexique.

Dans nos travaux, il nous arrive souvent de désigner cette réalisation sous le nom de **transducteur morphologique**. Signalons toutefois que dans ce domaine, des questions délicates de modélisation de grammaire peuvent se poser : il s'agira là encore de trouver le bon compromis entre, d'une part, le nombre de variables de la grammaire (c'est-à-dire, le nombre d'états de l'automate) et celui des actions définies sur les arcs. Autrement dit, on peut avoir le choix entre un automate avec un très grand nombre d'états et des actions minimales, ou bien à l'inverse disposer d'un automate minimal, mais dont les actions associées aux arcs seraient fort complexes.

AnIsl 33 (1999), p. 75-103 André Jaccarini
Vers une théorie du moniteur syntaxique.

La syntaxe pourrait aussi être décrite comme la transduction finie de la syntaxe quotient (voir § « Unification des points de vue syntaxique et morphologique »).

10. Il est possible de fournir une spécification algébrique des automates finis. On a recours pour cela aux **langages réguliers**. Le codage de ces automates, au moyen d'un langage régulier, est extrêmement important car il nous permettra d'effectuer des calculs algébriques.

Le **théorème de Kleene** établit l'identité entre les langages réguliers – que l'on peut définir à l'aide d'expressions régulières – et ceux acceptés (ou produits) par un automate fini.

Modifier un automate (une grammaire) pour en créer un nouveau – ou bien fabriquer une grammaire donnée à partir de fragments déjà existants – est une tâche extrêmement délicate. Il est très difficile de prévoir l'effet d'une modification. Par exemple, l'introduction ou la suppression d'une ou plusieurs règles peut rendre l'ensemble de la grammaire incohérente. Il n'est pas facile non plus de se rendre compte *a priori* si deux grammaires sont équivalentes. Pour toutes ces opérations, nous aurons en fait recours au calcul algébrique.

Il sera possible, par exemple, de résoudre des équations d'expressions régulières en vue de supprimer la récursivité inutile ou dérécursiver à gauche, transformer la grammaire en procédés déterministes ou bien tout simplement la modifier en fonction des besoins de l'analyse linguistique.

Cette structure algébrique doit cependant être étendue pour que l'on puisse également effectuer des calculs sur les transducteurs. C'est ainsi que l'on est parvenu à optimiser la plupart de nos programmes.

11. La hiérarchie de Chomsky

Il est possible de classer les grammaires formelles (ou systèmes de réécriture) selon le format de leurs règles de production. On peut ainsi distinguer quatre niveaux :

1. Les grammaires régulières (ou grammaires de Kleene) ;
2. Les grammaires non contextuelles (*context-free*) ;
3. Les grammaires contextuelles ;
4. Les grammaires sans restriction (équivalentes aux grammaires transformationnelles).

Comme les conditions deviennent de plus en plus contraignantes quand on remonte des grammaires du niveau 4 vers les grammaires du niveau 1, il s'ensuit que la classe des langages d'un niveau donné englobe la classe du niveau précédent : si L1, L2, L3, L4 représentent les classes de langages correspondant respectivement aux niveaux 1, 2, 3, 4 ;

on a $L1 \subset L2 \subset L3 \subset L4$.

Chomsky et Schützenberger ont montré que ces inclusions étaient strictes.

– Les automates finis correspondent aux grammaires de niveau 1.

– Quant aux automates à une pile de mémoire, on peut leur faire correspondre les grammaires de niveau 2.

AnIsl 33 (1999), p. 75-103 André Jaccarini
Vers une théorie du moniteur syntaxique.

– Aux grammaires de niveau 3, on peut faire correspondre une nouvelle classe d'automates à deux piles de mémoire, mais sur lesquelles on définit certaines restrictions concernant la taille de la mémoire ; ce sont les automates à mémoire linéairement bornée.

– Enfin, aux grammaires de niveau 4 correspondent des automates à deux piles de mémoire sans restriction.

On pourrait penser qu'en continuant à adjoindre des piles de mémoire aux automates, on augmenterait ainsi indéfiniment leur pouvoir de reconnaissance. *Or en fait, il n'en est rien.*

Le théorème suivant constitue un résultat de logique particulièrement significatif.

Théorème : tout langage reconnaissable par un automate à n piles de mémoire (pour $n > 2$), est reconnaissable par un automate à deux piles de mémoire.

À partir de $n = 2$, il ne sert donc à rien d'adjoindre de nouvelles piles de mémoire aux automates pour augmenter leur capacité de reconnaissance.

	type de grammaire	**automate correspondant**
niveau 1	grammaire linéaire (dite de Kleene)	automate fini
niveau 2	grammaire indépendante du contexte	automate à une pile de mémoire
niveau 3	grammaire contextuelle	automate à mémoire linéairement bornée
niveau 4	grammaire sans restriction	automate à deux piles de mémoire

La thèse de Church (1936), affirme que toute fonction calculable peut être simulée à l'aide d'un automate à deux piles de mémoire.

Cette thèse n'est pas prouvable pour la simple raison que la notion de fonctions « calculables » est métamathématique ; mais nous avons de bonnes raisons de l'accepter, car tous les formalismes que l'on a défini pour essayer de capter la notion de calculabilité, ont été prouvés équivalents entre eux. Ainsi, les machines de Türing sont équivalentes aux automates à deux piles de mémoires, aux machines de Wilkes et à bien d'autres mécanismes abstraits qui permettent de formaliser la notion de fonction calculable.

Il est intéressant enfin de considérer la notion de **décidabilité** d'un langage en regard de la hiérarchie de Chomsky.

Un langage est dit décidable si, et seulement si, existent :

1. Un mécanisme (machines de Türing, automates à deux piles de mémoire, etc.) pouvant engendrer ce langage ;
2. Un mécanisme pouvant engendrer son complémentaire.

AnIsl 33 (1999), p. 75-103 André Jaccarini
Vers une théorie du moniteur syntaxique.

Tout langage décidable présente donc la particularité suivante : pour toute séquence x de symboles, il existe un procédé pour décider si x appartient au langage ou non. En effet, x sera nécessairement produit par l'une des deux machines que nous venons de mentionner.

Dans la hiérarchie de Chomsky, les langages engendrés par les grammaires de niveau 1 (grammaires de Kleene), de niveau 2 (indépendantes du contexte) et de niveau 3 (dépendantes du contexte), sont décidables.

Les langages engendrés par les grammaires de niveau 4 ne sont que récursivement énumérables, c'est-à-dire que s'ils peuvent être engendrés par un mécanisme, il n'en est pas nécessairement de même de leurs complémentaires. Les grammaires de niveau 4 peuvent engendrer des langages indécidables. Ces langages sont tels que pour chaque séquence, il n'existe pas toujours un procédé pour décider si cette séquence appartient au langage ou non [32].

REMARQUE

Il nous arrivera de désigner par le terme de « ***token*** » les atomes considérés comme opérateurs. Ce terme est emprunté à la théorie de la compilation où il sert justement à désigner les associations de symboles d'un langage de programmation que l'on doit considérer comme figées, exemple : *BEGIN*, *END*, *GOTO*, etc. Les informaticiens traduisent généralement ce terme par « lexème » que nous ne pouvions retenir sans courir le risque, dans notre contexte linguistique, de créer des interférences d'interprétation. Dans le cas où ces atomes ne sont plus considérés comme de simples entités indécomposables et identifiables, mais également comme des unités lexicales provoquant des attentes particulières, nous avons préféré conserver le terme anglais. « *Token* » signifie en effet « jeton » ; un jeton est une entité visible qui déclenche un mécanisme.

[32] Cela dit, il peut être intéressant de noter que le langage de base des mathématiques, le calcul des prédicats du premier ordre, est indécidable.

AnIsl 33 (1999), p. 75-103 André Jaccarini
Vers une théorie du moniteur syntaxique.