Introduction au TALN et à l'ingénierie linguistique
université de Lille3

I.Tellier

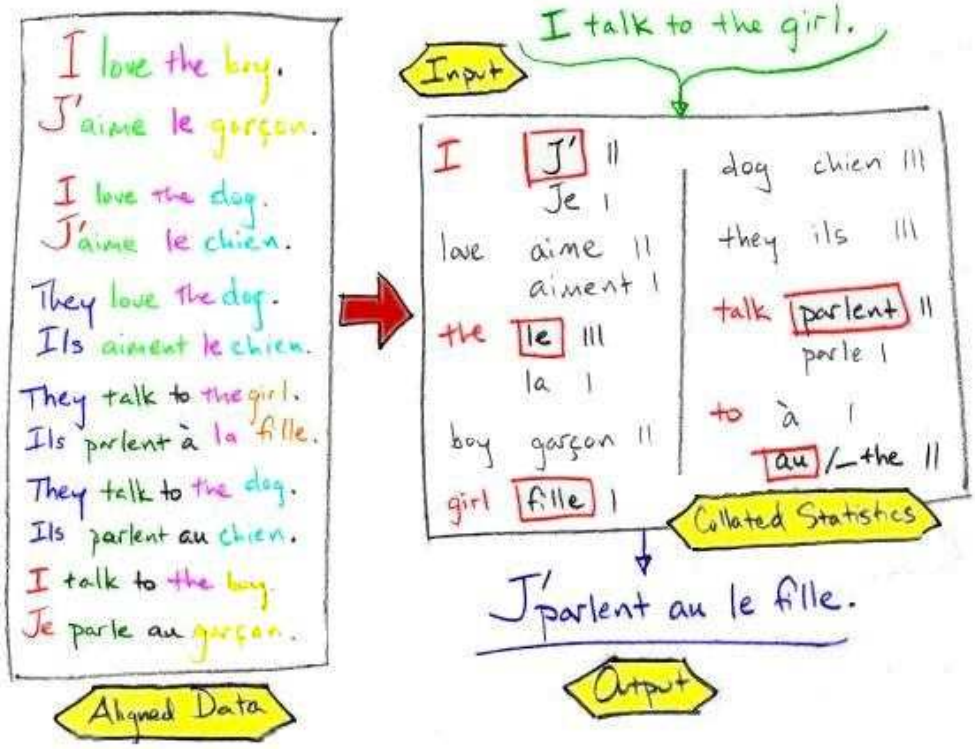


Table des matières

Chapitre 1

Introduction

Il ne leur manque que la parole ! Ce que l'on disait volontiers des animaux de compagnie il y a quelques années, on pourrait le formuler maintenant à l'intention des ordinateurs, nos compagnons familiers d'aujourd'hui. Le rêve de dialoguer "naturellement" avec des machines comme avec ses semblables est bien ancré dans l'imaginaire occidental. Tous les films de science-fiction se font l'echo de ce fantasme : impossible d'imaginer un robot un tant soit peu évolué qui n'aurait pas la capacité de parler. En 1950, Turing, le père fondateur de l'informatique, prédit que "dans 50 ans", les ordinateurs auront acquis cette capacité. L'échéance est passée sans que la prédiction ne se réalise. Pourtant, dans le domaine de la technologie numérique, beaucoup de rêves apparemment plus fous ont été largement dépassés. Aujourd'hui, les ordinateurs battent les grands maîtres d'échecs à plate couture, mais ils n'ont toujours pas les compétences langagières d'un enfant de 5 ans. Comment cela se fait-il ? En quoi la capacité de langage est-elle si difficile à "programmer" ? C'est à l'exploration de ces questions que nous invitons dans ce document.

Les travaux de recherche sur le sujet n'ont pourtant pas manqué, et ceci depuis les tout débuts de l'informatique. Une des premières conférences réunissant les pionniers de cette discipline, dans les années 50, portait sur la "traduction automatique". Le contexte politique était propice au développement de la thématique : l'époque était à la guerre froide et intercepter, décoder, traduire automatiquement les messages que s'échangeaient les "rouges" entre eux était pour les Américains un objectif stratégique. Mais l'histoire de ces tentatives est restée longtemps une succession d'échecs et de déconvenues. C'est l'histoire, en fait, d'une prise de conscience pour les informaticiens de ce que les linguistes savaient depuis longtemps : l'extrême complexité des langues naturelles.

Dans les années 70 et 80, les "systèmes experts" (programmes qui simulent, sous la forme de règles, les connaissances d'un expert, pour reproduire son mode de raisonnement) sont à l'honneur. Ils font les beaux jours de "l'intelligence artificielle" dont ils sont alors la principale vitrine. Le thème général du "traitement automatique du langage naturel" (TALN), lui, franchit plus rarement les frontières des laboratoires. Mais il fédère déjà les efforts de nombeux chercheurs. A l'image de ce qui se fait avec les "systèmes experts", on essait alors de ramener l'usage du langage à des règles symboliques. Beaucoup de concepts et de modèles évoqués dans les chapitres qui suivent datent de cette époque.

Les années 90 voient arriver des changements considérables : banalisation des ordinateurs personnels, avec des capacités de stockage et de traitement en progression exponentielle, apparition du Web. L'"ingénierie linguistique" naît à ce moment-là. Nous désignerons par ce terme la branche "utilitaire" des recherches en TALN, celles qui donnent lieu à des programmes applicables à des données réelles. Ces données ne manquent pas. Les textes représentent en effet une grande part de ce qui encombre depuis lors la mémoire des ordinateurs personnels, et constituent l'essentiel de ce qui est disponible sur le Web (l'émergence des sons et des images est plus récente). Tout ce qui peut aider à classer ou traiter les documents textuels, à extraire l'information qu'ils contiennent, devient un enjeu majeur. L'ingénierie linguistique se met ainsi au service de la "fouille de textes". Les approches symboliques laissent souvent la place à des méthodes statistiques.

Dans ce document, nous allons tout d'abord, dans le chapitre 2, parcourir un peu plus précisément l'histoire des liens entre la linguistique et l'informatique, en insistant notamment sur les "niveaux d'analyse" auxquels on peut soumettre le langage. Les chapitres 3 à 7 passent ensuite en revue chacun de ces niveaux. Pour chacun d'eux, nous adoptons une stucture commune avec une partie "description linguistique", puis une sur sa "modélisation informatique". Une liste de ressources relevant de l'ingénierie linguistique (sites Web ou programmes gratuits) mettant œuvre certains de ces modèles clôt chaque chapitre.

Le chapitre 8 est un peu à part. Il présente quelques-unes des tâches principales de la "fouille de textes" qui, sans être des tâches de traitement de la langue proprement dîtes, bénéficient de techniques issues de l'ingénierie linguistiques. Tout le parcours des recherches en TALN qui aura été évoqué auparavant prendra alors un autre sens. Peut-être n'aurons-nous pas de si tôt des ordinateurs de compagnie avec qui papoter, mais certains outils d'ingénierie linguistique, eux, sont d'ors et déjà intégrés dans des programmes informatiques que nous utilisons tous les jours...

Chapitre 2

Traitement Automatique du Langage Naturel

1 Introduction

Toutes les sociétés humaines découvertes de par le monde pratiquent au moins une langue. On en dénombre actuellement environ 5 000 différentes, dont beaucoup sont en voie de disparition faute de locuteurs. Même si l'acquisition du vocabulaire se poursuit tout au long de la vie, tout être humain normalement constitué et inséré depuis sa naissance dans un groupe social est capable, vers l'âge de 5 ans (donc bien avant qu'il ne maîtrise le "raisonnement"), de tenir une conversation courante dans sa langue maternelle. Aucun singe -et aucun ordinateur !- ne peut en faire autant. Parler est bien encore, à l'heure actuelle, "le propre de l'homme".

Pour désigner les langues humaines, on parle maintenant des "langues naturelles", parce que ce sont en quelque sorte des créations collectives spontanées auxquelles on ne peut pas attribuer de date de naissance précise. Les langues naturelles s'opposent ainsi principalement aux "langues artificielles" ou "formelles" que sont notamment les langages de programmation informatiques ou la logique mathématique. On peut très bien aussi classer parmi les "langues naturelles" les créations linguistiques intentionnelles comme l'Esperanto ou le Volapük, qui ne sont les langues maternelles de personne, ou encore certaines "langues des signes" inventées spécialement pour répondre à un besoin : leur point commun est qu'elles sont toutes sont destinées à être *utilisées par des humains pour communiquer*.

L'objet de la linguistique ou, comme on le dit plutôt désormais, des *sciences du langage*, c'est l'étude scientifique des langues naturelles et, à travers elles, du langage en tant que "faculté de langue" universellement distribuée dans l'espèce humaine. Derrière l'apparente diversité des langues humaines, les linguistes essaient de traquer des fonctionnements communs, des structures partagées, des *universaux*. Les linguistes ne sont pas nécessairement polyglottes ; ils cherchent plus à comprendre les *principes* qui régissent les langues qu'à multiplier les connaissances qu'ils ont de certaines d'entre elles (même si les deux ne sont bien sûr pas incompatibles).

Contrairement à une idée courante, la linguistique n'est pas prescriptive : elle ne dit pas comment "bien parler" ou "bien écrire". Les langues naturelles sont des systèmes vivants qui changent, interagissent, se transforment. Les linguistes se

contentent de les observer *telles qu'elles se parlent et s'écrivent*, sans chercher à contrôler ou à limiter leurs évolutions naturelles.

Pour cette étude, l'informatique joue un rôle de plus en plus considérable, via le domaine du *traitement automatique du langage naturel* (TALN). L'informatique est une discipline scientifique récente, qu'il ne faut pourtant pas réduire à la simple utilisation d'ordinateurs et de programmes. Son nom la désigne comme la science du "traitement automatique de l'information". Elle est en fait l'héritière d'une longue tradition mathématique et logique de *modélisation du calcul*. Plus précisément, on peut dire que les fondements de l'informatique sont double :

- le codage des données à l'aide d'*éléments discrets* (les fameux 0/1)
- le codage effectif des traitements à l'aide d'*algorithmes*

C'est par ce biais qu'on va aborder le traitement automatique du langage. Introduire une démarche informatique dans un domaine revient en effet toujours à se poser les mêmes questions :

- quelles sont les données pertinentes de ce domaine, comment les coder ?
- quels sont les traitements pertinents de ce domaine, comment les coder ?

Maîtriser une langue requiert la manipulation de nombreuses données, et la mise en œuvre de nombreux traitements. Les linguistes les ont progressivement mis à jour et caractérisé, les informaticiens ont progressivement contribué à les modéliser. Le TALN est né de leur interaction. Ce chapitre est destiné à faire un tour d'horizon (forcément simplificateur) de ce domaine.

2 Histoires croisées de la linguistique et de l'informatique

Les dates qui suivent ne constituent absolument pas une histoire exhaustive ni de la linguistique, ni de l'informatique. Ce sont plutôt des points de repère marquants qui concernent soit l'une de ces disciplines exclusivement (ce que l'on marquera par (l) pour "linguistique" et (i) pour "informatique" respectivement) soit les deux (ce que l'on marquera par (il) pour "informatique linguistique"). Souvent, les auteurs cités sont ceux qui ont introduit une distinction entre deux concepts, objets ou approches, féconde pour leur discipline. Ces distinctions sont en général destinées à favoriser une des deux branches de l'alternative, pour préciser la démarche suivie et les choix qu'elle entraîne. C'est à ce titre que ces auteurs sont évoqués.

avant le XVIIIème siècle : les précurseurs de la linguistique sont les (très nombreux) auteurs de grammaires descriptives d'une langue donnée (l), les précurseurs de l'informatique sont les mathématiciens qui décrivent des méthodes générales de calcul (par exemple pour résoudre des équations) et les inventeurs de "machines à calculer" mécaniques comme Pascal et Leibniz (i).

1660 : publication de la "Grammaire générale et raisonnée" (connue sous le titre "Grammaire de Port-Royal") d'Arnaud et Lancelot. Son ambition était de décrire les règles du langage en termes de principes rationnels universels (l).

aux XVIII et XIXème siècles : c'est le reigne de la linguistique comparative et historique. On compare les langues entre elles et on cherche à en déduire

des lois d'évolution générales. Du rapprochement entre diverses langues (latines, grecques, perses, germaniques, celtes, slaves, etc), émerge l'hypothèse que toutes ont un "ancêtre commun" qui sera appelé plus tard "indo-européen" (l). Le XIXème siècle connaît aussi de grands progrès en mathématiques, et voit naître la logique "booléenne" (ou "propositionnelle") par Boole (1815-1864) puis la "logique des prédicats du 1er ordre" par Frege (1848-1925). Les débuts de l'automatisation du travail (métiers à tisser activés par des cartes perforées) inspirent aussi les premiers projets de calculateurs mécaniques. Les plus novateurs et visionnaires sont dûs à l'ingénieur et mathématicien anglais Babbage (1791-1871), qui a conçu les plans de machines ayant les mêmes capacités de calcul que les ordinateurs actuels. Elles n'ont malheureusement pas pu être construites de son vivant (i).

1916 : (l) publication posthume (ce sont des notes de cours publiées par deux de ses étudiants) du "Cours de linguistique générale" du linguiste suisse Ferdinand de Saussure (1857-1913). Saussure introduit plusieurs distinctions et concepts importants :

- il caractérise le langage comme la construction sociale d'un *système de signes*. Un signe est l'association arbitraire entre un *signifiant* (défini comme l"image acoustique" d'un mot) et un *signifié* (un concept, la représentation mentale d'une chose). Les signes font sens par les *rapports qu'ils entretiennent les uns avec les autres dans le système*.
- il considère que le *langage*, en tant que faculté générale de s'exprimer au moyen de signes, se distingue de la *parole*, qui serait plutôt l'utilisation concrète de signes linguistiques particuliers (et qu'il n'étudiera pas plus avant).
- il distingue les dimensions *diachronique* (évolution au cours du temps) et *synchronique* (rapports entre les signes à une époque donnée) du langage. Les études historiques et comparatistes se sont focalisées sur la première de ces dimensions, lui entend privilégier la seconde (nous aussi par la suite).
- il distingue deux axes d'analyse d'un discours, en tant que suite de signes : l'axe *syntagmatique* est celui de la succession linéaire des unités qui constituent le discours (un syntagme est une suite d'unités adjacentes) ; l'axe *associatif ou paradigmatique* provient des liens que les signes présents dans le discours entretiennent avec d'autres signes non présents dans le discours mais en rapport avec eux dans le système. Suivant l'axe syntagmatique "un petit chat" est un syntagme dont le sens provient de la combinaison des signifiés de "petit" et "chat", mais ces sens eux-mêmes sont associés suivant l'axe paradigmatique avec d'autres ("petit" par opposition à "grand", etc.).

années 30-40 : (l) le "cercle de Prague" prolonge les analyses de Saussure et promeut une "linguistique structurale". Ses membres les plus connus sont Roman Jakobson (1896-1982) et Nicolas Troubetzkoy. On leur doit notamment l'invention de la *phonologie* : étude des sons élémentaires (les *phonèmes*) qui jouent le rôle d'unités distinctives dans une langue donnée. C'est aussi à Jakobson qu'on doit d'avoir identifié six *fonctions* permises par le langage dans un contexte de communication :

- la fonction *expressive* permet au locuteur d'exprimer ses sentiments ;

- la fonction *conative* permet d'agir sur le destinataire (donner un ordre...) ;
- la fonction *référentielle* permet d'informer sur le monde extérieur : il faut bien reconnaître que les modèles informatiques ont souvent tendance à limiter le langage à cette fonction ;
- la fonction *phatique* permet juste de s'assurer du bon fonctionnement de la "ligne" de communication ("allo"...) ;
- la fonction *poétique* met l'accent sur la forme du message plus que sur son contenu informationnel ;
- la fonction *métalinguistique* permet de parler du langage grâce au langage (comme le fait ce document !) ;

Alan Turing (1912-1954) : mathématicien anglais. En 1936, il propose le dispositif plus tard appelé "machine de Turing" qui donne une caractérisation mathématique précise à la notion d'algorithme. Cette proposition peut être considérée comme la date de naissance de l'informatique (i). En 1950, il publie un nouvel article fondamental dans lequel il décrit un test pour juger de la capacité des machines à penser : ce test, appelé depuis "test de Turing" est fondé sur un jeu de dialogue. En quelque sorte, il énonce qu'une machine peut être dite intelligente si elle est indiscernable d'un humain dans une situation de dialogue courant. Il prédit qu'en l'an 2000, des machines réussiront ce test (ça ne s'est pas vraiment réalisé). (il)

1945 : Von Neumann (1903-1957), mathématicien et physicien américain d'origine hongroise, définit dans un rapport le plan de construction des ordinateurs, tels qu'ils sont encore conçus de nos jours (des prototypes plus rudimentaires ont été construits avant). (i)

1952 : première conférence sur la traduction automatique, organisée au MIT (Massachussets Institute of Technology) par Yehoshua Bar-Hillel (1915-1975). C'est l'époque de la guerre froide, la compétition russes/américains bat son plein. L'informatique, elle, n'en est qu'à ses balbutiements et les premiers programmes de traduction doivent se contenter de dictionnaires bilingues et de quelques règles de restructuration élémentaires. La légende propage qu'en partant de "The spirit is willing but the flesh is weak" (l'esprit est fort mais la chair est faible), après un aller-retour russe-anglais, on obtint alors "The vodka is strong but the meat is rotten" (la vodka est forte mais la viande est pourrie)... Le terme "Intelligence artificielle" (IA), lui, est inventé lors d'une autre conférence, en **1956** (il).

André Martinet (1908-1999) : (l) linguiste français, connu notamment pour avoir caractérisé les langues naturelles par la propriété de la *double articulation*. Toutes les langues humaines sont doublement articulées parce qu'elles combinent des éléments (discrets) à deux niveaux différents :
- la "première articulation" est celle qui permet la combinaison "d'unités douées chacune d'une forme vocale et d'un sens" qu'il appelle des "monèmes" (le terme n'est plus vraiment employé : nous utiliserons plutôt celui de "morphème" mais, en première approximation, disons que ce sont les "mots"). On peut dire que ce niveau est celui de la *syntaxe*.
- la "deuxième articulation" décrit comment chaque "monème" est lui-même

décomposable en une succession d'unités phoniques élémentaires dépourvues de sens, les *phonèmes*.

Il semble bien que toutes les langues humaines, y compris les langues des signes utilisées dans les populations ayant une déficience auditive et/ou articulatoire (même si, dans ce cas, les unités employées ne sont pas de nature phonique), soient doublement articulées, alors que ce n'est le cas d'aucun autre système de transmission d'information, notamment de ceux en usage dans les autres espèces animales (nous détaillons cette affirmation dans la section suivante).

1966 : Deux programmes célèbres datent de cette époque : "Eliza" de Weizenbaum (simulation d'un dialogue avec un psychothérapeute, voir par ex : http ://www-ai.ijs.si/eliza/eliza.html) et "Student" de Bobrow (résolution de problèmes mathématiques simples). Les deux étaient fondés sur la recherche de "mots-clés" dans les données qu'on leur fournissait, mots-clés qui servaient à remplir les "trous" de formulaires définis *a priori*, sans prise en compte du contexte d'énonciation. Par exemple, dans un dialogue avec Eliza, dès que l'utilisateur mentionnait un lien de parenté ("father"/ "mother"/"brother"...), le programme enchaînait en demandant "Tell me about your [father/mother/brother...]". Bar-Hillel critique cette approche en citant l'exemple suivant, où le sens des mots dépend de leur contexte : "The pen is in the box" (le crayon est dans la boîte)/ "The box is in the pen" (la boîte est dans le parc). Cette même année, paraît le raport ALPAC (Automatic Language Processing Advisory Comitte), commandé 2 ans avant par l'Académie des sciences américaines. Le rapport est très critique vis-à-vis des recherches menées dans le domaine du traitement de la langue à cette époque, et conclut qu'elles mènent à une impasse. Suite au rapport, les financements publics se tarissent en Amérique (il).

Noam Chomsky (né en 1928) : linguiste et activiste politique américain d'une productivité exceptionnelle, professeur de linguistique au MIT depuis 1961, qu'une enquête de 2005 désigne comme "le plus grand intellectuel vivant". A travers les différentes théories qu'il a contribué à élaborer, il a toujours argumenté en faveur de la *primauté de la syntaxe* sur tous les autres niveaux d'analyse du langage. Il a toujours adopté aussi une position innéïste et rationnaliste : selon lui, les hommes disposent à la naissance d'un "organe du langage" de nature mentale. Dans cette perspective, apprendre une langue particulière revient à instancier une "grammaire universelle" innée. Il a élaboré de nouveaux concepts et promu plusieurs distinctions inédites :

- il distingue la *compétence* linguistique (connaissance des règles de fonctionnement d'une langue) de la *performance* (mise en oeuvre effective de ces règles en compréhension ou en production). On peut rapprocher cette distinction de celle entre *langage* et *parole* introduite par Saussure. C'est seulement la *compétence* qui sera l'objet de son attention, la *performance* relevant plutôt de la psychologie. La linguistique se doit en effet d'étudier le langage indépendamment de son substrat biologique (gènes, zones du cerveau, etc.) : on considère avoir affaire à un locuteur abstrait idéal.
- il considère que le but de toute théorie linguistique est l'explication des *jugements de grammaticalité* dont sont capables tous les locuteurs d'une langue (surtout quand c'est leur langue maternelle). Pour lui, la *structure de*

surface (syntaxe) d'un énoncé détermine sa *structure profonde* (les relations sémantiques qu'il exprime).

La chronologie des travaux de Chomsky a marqué l'histoire de la syntaxe des 50 dernières années :

- 1957 : publication de "Syntactic Structures" (structures syntaxiques), ouvrage fondateur. Dans les années 60, l'approche se raffine dans la théorie des "grammaires génératives et transformationnelles" qui deviendra ensuite dans les années 70 la "théorie standard" de la syntaxe.
- années 80 : l'approche "Principle and Parameters" (principes et paramètres) précise la nature de la "grammaire universelle" innée postulée par Chomsky : elle est constituée d'une liste de propriétés et de paramètres ne pouvant prendre qu'un nombre fini de valeurs. Ainsi, apprendre une langue particulière revient à acquérir son vocabulaire spécifique et à identifir la valeur des paramètres qu'elle instancie. Cette théorie sera reprise ensuite sous les termes "Gouvernment and Binding" (théorie du gouvernement et du liage, souvent abrégée en "GB").
- depuis 1995, Chomsky promeut un "programme minimaliste", qui est une reformulation de ses théories précédentes mais orientée par un principe d'économie.

Chomsky est aussi généralement bien connu des informaticiens via la *hiérarchie de Chomsky* qui caractérise des familles de langages de complexité croissante. Cette hiérarchie est issue de la mathématisation de la notion de "grammaire générative", qui remonte aux années 60. C'est le fondement de la *théorie des langages formels*, branche dans laquelle sont étudiées les propriétés des langages artificiels comme les langages de programmation informatiques. On évoquera cette hiérarchie au chapitre 5 dans ce document (il).

1972 : l'informaticien Terry Winograd présente son programme intitulé SHRDLU (ce nom proviendrait de l'ordre décroissant de la fréquence des lettres en anglais : ETAOIN*SHRDLU*...), permettant des interactions langagières avec un ordinateur sur un domaine restreint à un *monde de blocs*. Ce monde est constitué d'un nombre fini d'objets de forme géométrique simple (cubes, boules, cylindres, pyramides, etc.), disposés dans un environnement limité (l'équivalent d'une table). Les interactions se limitent à la possibilité de poser des questions sur l'état de ce monde simplifié ("Combien y a-t-il de cubes verts à droite de la boule rouge ?") et de donner des ordres permettant de le modifier ("Mettre le cylindre sur le cube bleu."). L'originalité de ce programme est qu'il ne se contente plus de mots-clés ou de "traitements de surface" rudimentaires : le monde étant parfaitement circoncrit, il pouvait être entièrement modélisé.

années 70-80 : elles sont marquées en TALN par l'effervescence de la *sémantique formelle* pour représenter des connaissances et formaliser des raisonnements : nouvelles logiques (logique floue, logiques modales, etc.), "scripts" (Roger Schank), "frames" (Marvin Minsky), "réseaux sémantiques", "graphes conceptuels" (John Sowa) naissent à cette époque. La pragmatique, c'est-à-dire l'étude de l'utilisation du langage en contexte, dans des situations concrètes (déjeuner au restaurant ou réservation de billets de trains, par exemple), est aussi prise en compte dans ces modélisations. Les "systèmes experts", basés sur des modèles

symboliques du même genre, constituent alors la vitrine de l'IA (il).

depuis les années 90 : L'augmentation considérable de la capacité de stockage et de calcul des ordinateurs, ainsi que le développement exponentiel d'Internet, permettent l'émergence d'une *linguistique de corpus* fondée sur l'exploitation de textes au format numérique. Cette linguistique plus empirique, *fondée sur les données* plutôt que sur des modèles formels abstraits, fait un usage intensif de calculs numériques et statistiques. Cette évolution va de pair avec les progrès de "l'apprentissage automatique", branche de l'IA consacrée à l'écritude de programmes qui s'améliorent avec l'expérience, grâce à des exemples. L'idée est d'utiliser des corpus pour apprendre automatiquement à fixer les paramètres de modèles (souvent statistiques) utilisables sur de nouvelles données. Cette démarche, prédominante dans la recherche actuelle, est abordée dans le chapitre 8 de ce document (il).

3 Les niveaux d'analyse du langage

La chronologie précédente a montré que ce n'est que progressivement que les sciences du langage ont précisé leur objet d'étude. Pour caractériser les *données* et les *traitements* pertinents de ce domaine, nous proposons le schéma de la figure 2.1.

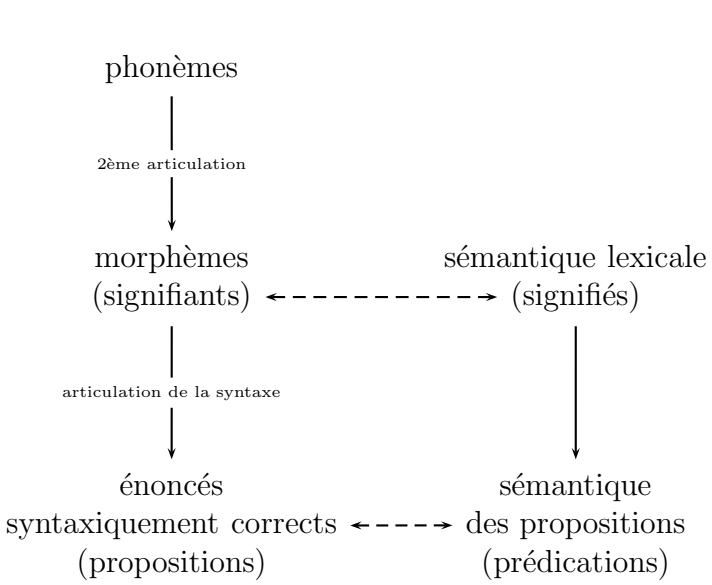


FIG. 2.1 – hiérarchie des niveaux d'analyse des langues naturelles

Dans ce schéma, les principales unités d'analyse figurent de haut en bas, des plus simples aux plus complexes. Chacun des "nœuds" du schéma est constitué d'un ensemble de *données discrètes*. Les "flèches verticales descendantes" symbolisent des *règles de composition* opérant des traitements combinatoires sur les nœuds. Les "flèches bidirectionnelles horizontales en pointillés" traduisent, elles, des relations d'association qui ne sont pas bi-univoques (ou pas bijectives) entre données. La non

univocité de ces flèches reflète les phénomènes d'*ambiguïté*, présents dans toutes les langues naturelles (sur lesquels nous reviendrons bien sûr par la suite).

Ce schéma permet surtout de mettre en évidence plusieurs des spécificités des langues naturelles évoquées précédemment :

- la dimension "verticale" du schéma est l'axe *syntagmatique*, tandis que sa dimension "horizontale" correspondrait plutôt à l'axe *paradigmatique*.
- la "double articulation" du langage se retrouve dans les deux niveaux de traitements combinatoires successifs qui occupent l'axe syntagmatique.
- ce schéma opère une claire distinction entre deux niveaux souvent confondus : celui de la *sémantique lexicale*, qui étudie le sens d'unités individuelles, et la *sémantique propositionnelle* qui étudie le sens d'énoncés complets, auxquels on peut attribuer une *valeur de vérité*.

Nous prétendons que ce qui caractérise les langues naturelles, c'est *l'ensemble des niveaux de description et relations présents dans ce schéma*. Précisons pourquoi.

Chacun des deux niveaux de l'axe syntagmatique traduit une combinatoire *ouverte* (au sens où la liste des éléments qu'elle produit est potentiellement infinie) d'*éléments discrets*. Ce dispositif original diffère fondamentalement de codages de type *analogique*. On cite souvent, pour illustrer la communication animale, le cas des abeilles, qui informent leurs congénères de l'emplacement d'une source de nourriture en modulant de façon *continue* l'amplitude et l'orientation de leur vol : leur danse fait un "huit", incliné suivant un angle qui indique la direction à suivre et dont la hauteur est proportionnelle à la distance à parcourir. C'est un exemple typique de codage *analogique* (mais tous les modes de communication animale ne sont pas analogiques). Comme le remarque le psycholinguiste canadien Stephen Pinker (né en 1954), les deux systèmes les plus sophistiqués de transmission d'information qui ont été sélectionnés par la nature, à savoir les langues naturelles et le code génétique, reposent tous les deux sur des unités *discrètes*. La nature a inventé le codage "numérique" bien avant les informaticiens. Sans doute est-ce dû à la fiabilité de la transmission ainsi permise...

La maîtrise d'associations arbitraires "signifiant/signifié" au niveau lexical semble accessible à certaines espèces animales (principalement des singes) à qui on a pu enseigner l'usage d'un répertoire non négligeable de symboles : gestes empruntés à une langue des signes ou dessins abstraits arbitraires. Mais aucune espèce autre que l'homme n'a développé cette capacité dans la nature, sans enseignement explicite. Et (même si ce point est encore discuté) aucune non plus n'a semblé être capable de maîtriser complètement un niveau de combinaison syntaxico-sémantique plus complexe : la simple juxtaposition de symboles ne suffit pas à faire un langage doublement articulé.

Le niveau de l'association entre "unités signifiantes" et "sémantique lexicale" est un jalon important de notre schéma. Il constitue une étape fondamentale de l'acquisition de leur langue maternelle par les enfants, aux alentours de leur première année de vie. Certains auteurs argumentent aussi qu'il était peut-être la base d'un "protolangage" que nos lointains ancêtres auraient inventé avant d'avoir recours aux langues naturelles proprement dîtes. Evidemment, ce genre d'hypothèses est difficile à valider mais elle a le mérite de mettre l'accent sur la complexité considérable des langues humaines, dont on a du mal à imaginer comment elles ont pu émerger "d'un

seul coup" dans une espèce particulière.

Pourtant, le critère de la "double articulation" est probablement insuffisant pour distinguer à lui tout seul les langues humaines de langages d'un autre genre apparus récemment : les langages de programmation informatiques (qui, il est vrai, étaient encore peu connus et diffusés en dehors des spécialises à l'époque de Martinet). On peut en effet argumenter que ces langages sont, eux aussi, doublement articulés :

- la première articulation est celle des règles à suivre impérativement pour écrire un programme syntaxiquement correct ;
- la deuxième articulation caractérise la construction des unités lexicales de ce programme (soit mots clés du langage, soit identifiants de variables ou noms de fonctions) à partir des unités distinctives élémentaires que sont les caractères alphanumériques autorisés (pendant écrit des phonèmes).

Il est possible aussi d'associer une "sémantique" aux programmes informatiques, aux deux niveaux évoqués par le schéma. Sa nature n'a pourtant rien à voir avec celle véhiculée par les langues naturelles : son domaine se réduit à l'arithmétique, son "monde" est celui du calcul opérationnel, et elle exclut toute ambiguïté. Les langages de programmation radicalisent en quelque sorte les propriétés formelles des langues naturelles. Pour donner aux ordinateurs l'incroyable pouvoir expressif de ces langues, à savoir leur capacité à référer à ce qui leur est extérieur, à dire des choses sur le monde, il reste à réduire cette description du monde à un calcul. C'est paradoxalement leur caractère flou et *ambigu* qui rend finalement les langues naturelles plus aptes à réaliser ce programme.

Notre schéma permet ainsi de bien situer les unes par rapport aux autres les distinctions évoquées précédemment, qui se focalisent chacune sur une portion de sa structure. Il est toutefois loin d'être exhaustif. Un schéma plus complet devrait aussi faire figurer, à titre de niveau intermédiaire, la combinatoire propre à la *morphologie*, à laquelle nous consacrerons bien sûr un chapitre. Ne sont pas évoqués ici non plus les niveaux d'analyse qui vont au-delà des propositions (analyse des discours, des textes ou des dialogues, pragmatique...). Pourtant, les hommes se racontent en permanence des anecdotes et des histoires, et c'est au bout du compte ce *comportement narratif* qui caractérise le mieux l'espèce humaine. Mais la figure 2.1, et tout particulièrement le "rectangle" qu'elle fait apparaître à sa base, circonscrit en quelque sorte le périmètre de ce document.

Le principal intérêt pour nous de ce schéma, malgré son caractère réducteur, est qu'il met bien en évidence les données (les nœuds du schéma) et les traitements (les flèches) qui vont pouvoir donner lieu à une modélisation informatique. Il reste à voir comment traduire en code informatique les informations qui y figurent. La prise en compte des données discrètes de la colonne "niveaux de composition" ne pose pas de grosses difficultés, puisqu'il s'agit de coder des données discrètes par d'autres données discrètes. La traduction sous forme d'algorithmes des modes de combinaisons de ces données est un problème nettement plus intéressant. Dans ce domaine, théories linguistiques et modèles informatiques doivent collaborer. Quant au codage de la dimension "sémantique" du schéma, la plus cruciale et la plus problématique, elle est l'objet de nombreuses théories dont nous allons aussi essayer de rendre compte. Mais avant cela, voyons comment concevoir l'architecture "classique" d'un système complet de "compréhension du langage".

4 La chaîne de traitements “standard”

Pour comprendre comment les humains utilisent une langue naturelle, il ne suffit pas d'avoir identifié les différents niveaux de connaissances impliqués dans la compréhension de cette langue : il faut aussi savoir comment ces connaissances sont exploitées. Ce champ d'étude ne relève plus à proprement parler de la linguistique, mais de la *psychologie*, voire de la *psycholinguistique*. La figure 2.2 propose une chaîne de traitements qui a une certaine plausibilité. C'est avec ce genre de schémas que les psychologues cognitivistes tentent de modéliser le fonctionnement de l'esprit humain.

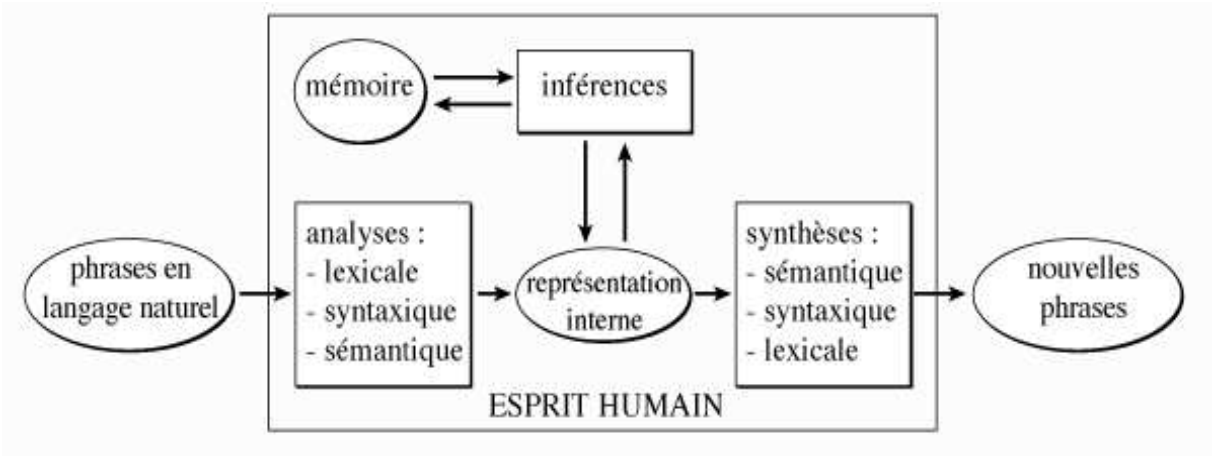


FIG. 2.2 – chaîne de traitements classique de compréhension du langage

Dans ce schéma, ici encore très simplificateur (il n'intègre pas, par exemple, la composante orale du langage), les données figurent dans des ovales, tandis que les traitements sont représentés dans des rectangles. Les deux principaux rectangles, intitulés respectivement “analyses” et “synthèses”, correspondent aux deux tâches principales accomplies par les locuteurs d'une langue. Le fait de bien les séparer provient de l'observation de patients souffrant de lésions cérébrales qui affectent en particulier une de ces compétences et pas l'autre. Il n'est pas nécessaire pourtant de faire l'hypothèse que chacun des traitements évoqué dans ce schéma soit réalisé par une aire cérébrale spécifique. Il suffit de considérer qu'il met en évidence un enchaînement des *fonctions*, indépendamment de leur “implantation” dans un substrat biologique concret.

Suivant ce schéma, *comprendre un énoncé* revient donc à le transformer, via une “analyse”, en une représentation interne, tandis que pour en *produire* ou en *générer* un, il faut traduire linguistiquement une telle représentation via une “synthèse”. Chacune de ces tâches nécessite de prendre en compte l'ensemble des niveaux d'analyse identifiés précédemment, mais dans un ordre différent et en faisant chaque fois des hypothèses différentes sur ce qui est connu et ce qui doit être accompli.

Maintenant, pour construire un système artificiel complet avec lequel des humains pourraient interagir via une langue naturelle, une première approche possible consiste à tenter de reproduire une architecture de ce genre, en traduisant les “fonc-

tions" en programmes. C'est en quelque sorte le projet du "traitement automatique du langage naturel" dans sa forme originelle. Il a donné lieu à de très nombreux travaux ces 50 dernières années, et nous allons prendre le temps dans les chapitres qui suivent de présenter ses principaux résultats. Nous verrons notamment que beaucoup des outils formels imaginés pour modéliser certains des traitements cités ici sont en fait exploitables à la fois en analyse et en synthèse (c'est le cas, par exemple, des automates et grammaires formelles). D'autres sont plus spécifiques.

Pourtant, le schéma de la figure 2.2 n'est instancié dans sa totalité dans aucun système informatique existant. C'est un cadre idéal théorique, très influencé par une conception *cognitiviste* de l'esprit humain, où les "représentations internes" sont en général de nature symbolique. Pour une application particulière, on se limite la plupart du temps à implémenter une toute petite portion de cette architecture.

Mais il est aussi possible d'envisager une toute autre approche, qui ne se soucie pas de crédibilité psychologique, et se concentre sur l'efficacité pragmatique de ses programmes. Cette mutation est représentative de l'évolution qu'a connue l'"intelligence artificielle" ces dernières années. Après avoir longtemps tenté d'imiter le fonctionnement de l'esprit humain, les chercheurs du domaine essaient plutôt désormais d'exploiter au mieux les capacités de mémoire et de calculs de leurs machines. Des *modèles symboliques formels*, on est passé aux *modèles statistiques fondés sur l'analyse des données*. Nous aborderons cette mutation dans le dernier chapitre de ce document, centré sur l'ingénierie linguistique, telle qu'elle est exploitée en "fouille de textes".

Dans la suite de ce document, nous allons en quelque sorte passer en revue les niveaux d'analyse du schéma de la figure 2.1, en présentant chaque fois les théories linguistiques qui les décrivent et quelques-uns des modèles informatiques auxquels ils ont donné lieu. Les données et les traitements seront "parcourus" de haut en bas et de gauche à droite, en passant des unités les plus simples aux unités les plus complexes, sans cacher les difficultés propres à l'informatisation de chacun d'eux. Seul le dernier chapitre sera consacré aux applications qui utilisent certains des outils ainsi décrits.

5 Sites Web

Avant de commencer l'exploration des niveaux d'analyse du langage, faisons un détour par le "test de Turing". Ce test est un jeu au cours duquel un juge humain, en situation de dialogue médiatisé par une machine avec une entité distante, doit décider au bout d'un temps fixé à l'avance si son interlocuteur est un humain ou un programme. Son origine remonte à un article philosophique que Turing, l'inventeur de l'informatique, a fait publier en 1950. Selon lui, un programme qui "passerait ce test" -autrement dit qui ne serait pas identifié comme tel par le juge- devrait se voir reconnu les mêmes capacités que celles attribuées "naturellement" aux humains -en particulier celle de penser... Ce test fait donc des capacités linguistiques en situation de dialogue le critère principal de l'"intelligence".

Il existe à l'heure actuelle une compétition qui propose d'instancier le "test de Turing" dans un cadre contrôlé : elle s'appelle le "Loebner Price", du nom du généreux

donateur qui offre un prix et une médaille au vainqueur. Les sites suivants offrent la possibilité d'une petite conversation à distance avec un programme ayant participé à cette compétition. A vous de voir si vous leur accorderiez le prix...

- www-ai.ijs.si/eliza/eliza.html
- www.ellaz.com/EllaASP8/Direct.aspx
- www.bearbot.co.uk/
- www.pandorabots.com/pandora/talk ?botid=f5d922d97e345aa1

Chapitre 3

Aspects acoustiques du langage

Les langues naturelles sont avant tout *orales* ; beaucoup n'ont d'ailleurs pas de transcription écrite. Il est donc naturel que de nombreuses propriétés de ces langues découlent de considérations *acoustiques*. Nous évoquons brièvement les principales dans les sections qui suivent. Comme l'objectif final de ce document est de présenter les techniques et outils de traitements de *textes numériques*, les données qui nous intéressent relèvent en général plus de l'*écrit* que de l'oral. Mais certains documents écrits peuvent être la transcription de données orales (avec un alphabet phonétique, par exemple). Nous abordons donc ici rapidement les aspects oraux du langage, dans le but d'introduire les concepts fondamentaux du domaine, et aussi d'illustrer certains modes de raisonnements linguistiques ou certains modèles informatiques qui seront repris, plus tard, à d'autres niveaux d'analyse.

1 Description linguistique

1.1 Phonétique

La *phonétique* est la branche de la linguistique qui étudie les *sons* des langues naturelles, indépendamment de leur sens. Pour caractériser les sons émis par des humains qui parlent, on peut partir de leurs descriptions physiques, telles que les mesurent des dispositifs électro-acoustiques comme les oscillogrammes et les spectrogrammes (cf. les diagrammes de la figure 3.1).

Dans ces deux types enregistrements, l'axe horizontal marque le déroulement du temps. L'oscillogramme traduit par le mouvement d'une aiguille les variations de l'onde acoustique, tandis que le spectogramme enregistre, pour chaque fréquence sonore (déroulées sur l'axe des ordonnées), l'amplitude (en décibels) de cette fréquence, et la traduit dans l'intensité du pixel correspondant. La *phonétique acoustique* étudie les propriétés de diagrammes de ce genre.

L'inconvénient de telles descriptions, qui sont de nature *continue*, est qu'elles ne rendent pas compte du fait que ces sons sont émis par des organes humains. La *phonétique articulatoire* va, elle, s'attacher à étudier les sons élémentaires d'une langue via la configuration physiologique nécessaire pour les produire. Ainsi, chaque émission vocale peut être décrite par un ensemble de *traits articulatoires* caractérisant la position des organes intervenant dans la prononciation (langue, gorge, glotte, nez

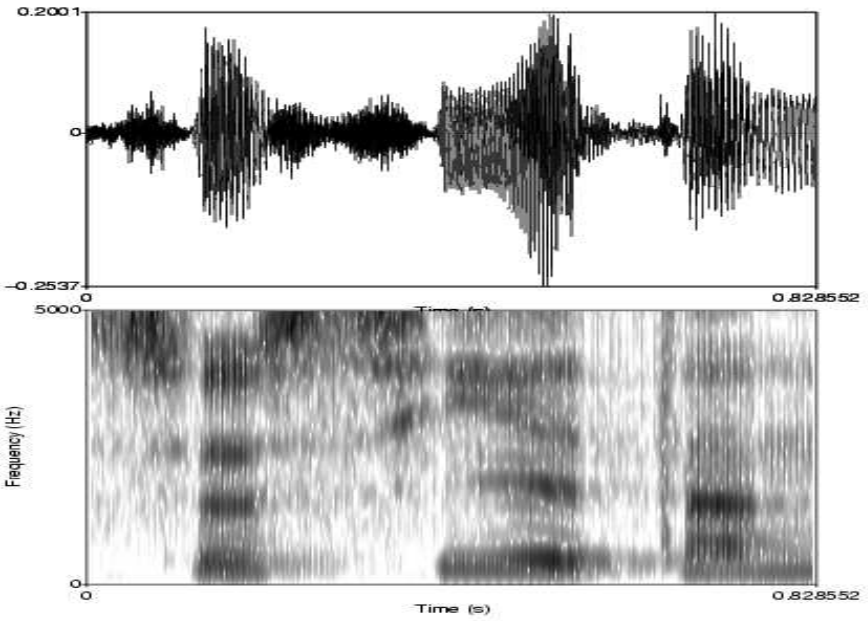


FIG. 3.1 – oscillogramme (en haut) et spectrogramme (en bas) d'une onde acoustique correspondant à la prononciation d'un morceau de phrase

et lèvres). Voici un exemple de description articulatoire (les transcriptions phonétiques sont toujours notées entre crochets) :
[t] : consonne occlusive sourde dentale non nasale
Elle peut être paraphrasée (à la façon du "maître de philosophie" faisant la leçon à M. Jourdain dans *Le Bourgeois gentilhomme*) comme suit :

- consonne : un son qui est produit grâce à un obstacle
- occlusive : l'obstacle doit être total
- sourde : et ne pas générer de vibrations
- dentale : on l'obtient en mettant la langue sur les dents
- non nasale : sans qu'aucun air ne passe dans le nez

Toutes les propriétés ainsi détaillées doivent être réalisées *en même temps* pour que le son soit correctement produit. Une telle description est de nature *discrète* et représente une certaine abstraction par rapport à la description physique : elle ne tient pas compte des différences physiologiques pouvant exister entre deux individus dont les émisssions varient en timbre, en hauteur et en intensité. Des *alphabets phonétiques* ont ainsi été définis, dans lesquels chaque symbole correspond à un ensemble de traits articulatoires. Le plus utilisé est l'API (Alphabet Phonétique International) ; il est censé permettre de transcrire les productions orales de n'importe quelle langue naturelle. Il existe depuis 1888, sa dernière révision date de 1993.

1.2 Phonologie

Ce ne sont pourtant pas les éléments de cet alphabet qui constituent les unités de base de l'analyse du langage, parce que les sons ainsi décrits ne sont pas tous pertinents pour une langue donnée. En fait, chaque langue opère une sélection dans la liste des sons que la physiologie rend possibles. Et, parmi les sons "adoptés", elle opère un regroupement en *classes d'équivalences*. Deux sons élémentaires appartiennent à des classes différentes s'il est possible de trouver deux unités lexicales (c'est-à-dire, pour simplifier, deux mots) différentes (c'est-à-dire associées à des signifiés différents) qui ne diffèrent d'un point de vue acoustique que par ces deux sons. On appelle un tel couple de mots une "paire minimale". Inversement, ils sont équivalents (et appartiennent donc à la même classe) s'il est impossible de trouver deux unités lexicales différentes qui ne diffèrent que par ces deux sons. Par exemple :

- "zona" et "sauna" sont des mots différents en français, et ils ne diffèrent que par leur son initial. Ces deux sons ne sont donc pas équivalents en français. Ils le sont pourtant en espagnol, où ils ne permettent de faire la distinction d'aucun couple de mots.
- "cote" et "côte" sont des mots différents et ils ne diffèrent que par leur voyelle centrale : ces deux sons ne sont donc pas équivalents en français (même si leur distinction a tendance à s'estomper dans la langue parlée).
- en français, le fait de "rouler les r", de les "grasseyer" (comme Edith Piaf) ou de les prononcer "normalement" ne permet de distinguer aucun mot. Ces trois sons sont donc équivalents en français, où ils peuvent au plus passer pour une pointe d'accent mériodional ou vieillot. En espagnol, en revanche, on peut distinguer "pero" (mais) et "perro" (chien) par le fait de rouler ou non le son "r". Ces deux sons ne sont pas équivalents dans cette langue.

Deux sons sont ainsi équivalents si, en remplaçant l'un par l'autre, le mot prononcé reste le même. La relation d'équivalence considérée est donc celle de *substituabilité en préservant l'identité de l'unité de niveau supérieur*. Une classe d'équivalence de sons élémentaires distinctifs dans une langue donnée est appelée un *phonème*. Les exemples précédents montrent que la notion de phonème ne coïncide ni avec celle de son élémentaire, ni avec celle de caractère alphabétique, et que chaque phonème est spécifique d'une langue donnée.

La *phonologie* est la branche de la linguistique qui étudie les propriétés des phonèmes. La transcription phonologique d'un énoncé peut se coder en utilisant certains des symboles de l'API mis entre barres obliques. Par exemple, le phonème correspondant à toutes les prononciations possibles de "r" en français est noté /r/. Le français comprend environ 33 phonèmes : 20 consonnes et 13 voyelles (le nombre de voyelles a tendance à baisser).

1.3 Autres aspects acoustiques

La phonétique et la phonologie n'épuisent pas la description acoustique d'une langue. Dans certaines langues, comme l'anglais, chaque mot reçoit un *accent tonique* qui se traduit par une augmentation de l'intensité (et/ou de la hauteur) de la voix lors de la prononciation de la syllabe sur laquelle il est localisé.

Dans les “langues à tons”, comme le chinois et la plupart des langues du monde, le même son, suivant la *tonalité* avec laquelle il est prononcé, peut changer le sens du mot dont il fait partie. Ces propriétés sont de nature discrète et peuvent être reproduites à l'écrit par des symboles spécifiques. On admet généralement qu'elles ne concernent pas la langue française.

Enfin, on désigne par *prosodie* les règles de prononciation globales qui influent sur la mélodie d'un énoncé. En français, suivant *l'intonation* qu'on y met, “tu viens demain” peut devenir une affirmation, un ordre ou une question sans que le sens des mots présents ne change pour autant. La prosodie est de nature *continue*. Quelques modes de codages discrets ont toutefois été proposés pour l'intégrer à la transcription écrite de données orales.

2 Modélisation informatique

Encore une fois, la modélisation de l'oral n'est pas notre objectif principal. Le domaine se rattache plutôt au “traitement du signal”, branche de l'électronique et de l'automatique plus que de l'informatique, parce qu'elle traite de données continues. Mais certaines des techniques utilisées pour manipuler des données orales sont aussi employées à d'autres fins. Et puis, nous voulons surtout ici faire “sentir” la nature des problèmes qui se posent.

2.1 Domaines et problèmes

Il existe de très nombreuses applications, y compris “grand public”, au traitement de la langue en tant que “signal sonore”. Nous citons ici les principales :

- en *analyse* : la reconnaissance vocale peut servir à identifier un locuteur par sa voix, à identifier la langue qu'il parle, à reconnaître l'ordre qu'il donne ou à transcrire automatiquement, sous forme écrite, ce qu'il dit. Dans ce dernier cas, on passe généralement par deux étapes successives : d'abord une transcription du son en une séquence de phonèmes, puis en un texte écrit.
- en *synthèse* : il s'agit cette fois de produire une lecture orale à partir d'un texte écrit. Cela revient donc principalement à transformer un texte en une succession de phonèmes, puis en une émission acoustique.

En analyse, de nombreuses difficultés doivent être surmontées : tout d'abord, il faut distinguer le son de la voix des bruits environnants et adapter le système au timbre et à la hauteur de voix du locuteur. Les systèmes *monolocuteurs*, c'est-à-dire destinés à n'être utilisés que par une seule personne, nécessitent en général une phase d'apprentissage au cours de laquelle il est demandé à cette personne de lire un texte standard : cette phase sert à calibrer la valeur de certains paramètres acoustiques. Ces derniers ne doivent pas pour autant être trop rigidement fixés, pour éviter que le système échoue en cas de modification momentanée de la voix du locuteur, parce qu'il est stressé, malade ou enrhumé, par exemple. Quant aux systèmes multilocuteurs, ils doivent être capables de s'adapter aux variations interindividuelles.

La difficulté majeure à affronter est apparemment celle de la *segmentation* du flux continu de paroles en unités discrètes. Une fois ce découpage réalisé, identifier le phonème correspondant à chaque unité n'a encore rien d'immédiat. Un même

phonème peut être prononcé de façons très différentes, suivant son voisinage avec les autres phonèmes (un /a/ en début ou en fin de mot ne se prononce pas du tout de la même façon), et certains phonèmes ont tendance à être "avalés" dans une prononciation courante.

Pour passer d'une séquence de phonèmes à un texte écrit, il reste à opérer des regroupements, là encore pas toujours évidents à faire, et à éventuellement retrouver parmi des homophones, c'est-à-dire des mots qui se prononcent de la même façon, celui qui doit est utilisé dans la transcription. Parfois, seuls des critères *sémantiques* permettraient de faire ce choix. Le petit poème suivant, cité par Henri Morier dans son *Dictionnaire de poétique et de rhétorique* (PUF, 1981) l'illustre de façon ludique :

> Gall, amant de reine, alla, tour magnanime,
> Galamment de l'arène à la tour Magne, à Nîme,
>
> Dans ces meubles laqués, rideaux et dais moroses
> Danse, aime, bleu laquais, ris d'oser des mots roses.

Il y a aussi le problème de la ponctuation, des abbréviations, des nombres, etc. qui nécessitent un traitement spécifique pour être correctement traduits en symboles graphiques. Enfin, pour que le texte écrit final soit correct, il faut aussi tenir compte des accords, et donc de la *syntaxe*. On commence ainsi à voir que les différents niveaux d'analyse sont souvent interdépendants, et difficiles à isoler les uns des autres dans un traitement.

La synthèse vocale pose, *a priori*, moins de difficultés, mais elle doit tout de même surmonter quelques pièges. L'un d'eux provient des *homographes hétérophones*, c'est-à-dire des mots qui s'écrivent de la même façon mais ne se prononcent pas pareil. Le petit texte suivant en est truffé :

Note aux élus qui au Conseil **président** et au **président** de la république d'Egypte

Il faut que nous **adoptions** pour les **adoptions** des mesures favorables afin que cesse un trafic dénoncé par M. Jean, célèbre **reporter**. **Reporter** ce débat serait pure folie. "Dans notre pays il devient impossible d'adopter un **fils** tant les **fils** des **lacs** de l'administration sont troubles comme les eaux des **lacs** et étriqués comme un **jean**, dit **Jean**, et, pour les démêler, vous en **suez**. Or, à **Suez**, une **mater** et les sœurs d'un **couvent couvent** un réseau à **mater**. Elles échangent des orphelins mal nourris contre **dations**." Nous **dations** de 1995 ces propos dans notre précédent rapport. Intervenir devient **urgent**, comme **urgent** ces **options** à prendre. Il faut que nous **options** pour l'abolition de ce genre d'**exécutions** ; nous **exécutions** des contrôles dans plusiseurs de ces établissements d'un seul **jet** ; nous affrétions un **jet** chargé de **rations** de riz ; nous ne **rations** pas une occasion de stopper ce trafic **influent** dont les chefs **influent** sur vos **relations** (nous en **relations** hier encore dans la presse). Donc, il **convient** que les élus **convient** les autorités

à affronter les problèmes là où ils **résident**. Un **résident** confirme : si nous **intentions** un procès, nos **intentions** seraient louées puisqu'elles **coïncident** avec le désir (**coïncident** à la venue du pape) de démanteler cette mafia. Que nous **transitions** par Rome s'impose donc sans **transitions**. Comment se **fier** à celles qui se **parent** de cornettes, **content** des sornettes et **violent** les lois, et rester un **fier** citoyen, voire **parent** adoptif et **content** de l'être, mais non **violent** ?

Ici aussi, ce sont plutôt des critères syntaxico-sémantiques qui devraient aider à choisir la bonne prononciation. Enfin, pour restituer à un système artificiel une diction "naturelle", il faut aussi tenir compte des règles de prononciation (quand un "e" est muet ou pas, etc.), de la ponctuation, des enchaînements et des liaisons. La synthèse d'une certaine prosodie, voire d'émotions quand le texte le justifie, est aussi assez délicate.

2.2 Outils formels ou statistiques utilisés

Pour programmer un système de reconnaissance ou de synthèse vocale, de nombreuses techniques ont été testées. Nous nous concentrons ici sur les systèmes d'analyse, qui posent plus de problèmes. A une certaine époque, on a par exemple tenté de traduire sous la forme de "bases de connaissances" façon "systèmes experts" les règles d'identification des phonèmes. Une règle typique d'un système de ce genre serait de la forme :
SI le spectrogramme montre une "barre d'explotion" ET est immédiatement suivi d'une "voyelle arrière-droite" ALORS la consonne correspondante est une "dentale" (c'est-à-dire /d/ ou /t/).

On peut aussi disposer de bibliothèques d'exemples de prononciations de succesions de phonèmes déjà connus, et chercher, face à un nouvel exemple, celui qui s'en approche le plus dans la bibliothèque.

Mais les méthodes les plus performantes en la matière font appel à des techniques avancées en *traitement du signal* et à des *modèles statistiques*. Un exemple de modèle statistique élémentaire est ce qu'on appelle un "n-gramme". Un "n-gramme" est simplement une succession de n éléments. Par exemple, si on prend n=2, on obtient des *bi-grammes*. Modéliser les successions de phonèmes en français par des bi-grammes, cela signifie inventorier tous les couples de phonèmes successifs possibles et calculer, pour chacun d'eux, leur fréquence d'apparitions dans la langue. Pour faire ce calcul, on doit disposer d'un échantillon significatif de données, présentes dans *corpus*. Certains couples comme /de/ ou /du/ seront très fréquents, d'autres très rares voire toujours absents (comme /pz/). Avec un n-gramme, on fait en général l'hypothèse que le n ième élément d'une suite ne dépend que des n-1 ièmes qui le précèdent. Ainsi, si on a déjà identifié dans un traitement un certain phonème, on dispose grâce aux bi-grammes de la probabilité d'apparition du phonème qui le suit immédiatement. Cette valeur aide à sélectionner parmi les phonèmes possibles celui qui a le plus de chances d'être présent. C'est un premier exemple d'apprentissage automatique à partir de données. Nous en évoquerons d'autres dans le chapitre 8. Plus on choisit une grande valeur de n, meilleure sera la prédiction. Mais le recueil des données nécessaires au calcul des fréquences d'apparition grandit lui aussi... Evi-

demment, il existe des modèles statistiques fondés sur des hypothèses nettement plus sophistiquées.

On trouve dans le commerce à l'heure actuelle des systèmes de reconnaissance vocale très efficaces. Leurs performances dépendent néanmoins beaucoup de l'environnement (plus ou moins bruyant) et des conditions d'utilisation : un texte lu devant un micro, parlé au téléphone ou capté lors d'un dialogue ne sera pas du tout reconnu de la même façon. Les meilleurs systèmes sont capables de reconnaître un flux de parole continu en faisant moins de 5% de fautes. Des systèmes de synthèse vocale fonctionnent également très bien.

2.3 Sites Web

Plusieurs sites réalisent des synthèses vocales en ligne de différentes langues, en laissant l'utilisateur fixer un certain nombre de paramètres (timbre, sexe, hauteur de la voix), et parfois en lui permettant de visualiser le résultat d'analyses intermédiaires réalisées par le programme. Pour les mettre en difficulté sur le français, il suffit de leur soumettre les phrases du texte de la section 2.1. On peut aussi facilement simuler les accents nationaux, en demandant à une voix de faire la synthèse d'un texte écrit dans une autre langue que celle pour laquelle elle a été prévue...

- www.research.att.com/∼ttsweb/tts/demo.php
- www.multitel.be/TTS/

Chapitre 4

Morphèmes, morphologie

En combinant des sons élémentaires qui, par eux-mêmes, ne veulent rien dire, on finit par réussir à "dire" quelque chose, c'est-à-dire à *référer* à quelque chose du monde, à *signifier*. Il y a là un saut qualitatif considérable qui justifie qu'on lui associe un niveau d'analyse spécifique. Ce nouveau niveau correspond à ce que le sens commun identifie par la notion de "mot". Mais, comme nous le verrons, cette notion n'est pas très pertinente pour la linguistique, qui lui préfère celle de "morphème".

1 Description linguistique

1.1 Problèmes avec la notion de "mot"

Qu'est-ce qu'un mot ? La première définition possible fait appel à un critère formel : un mot c'est ce qui, dans un texte, est *compris entre deux séparateurs*. Mais cette définition n'est pas aussi opératoire qu'on pourrait l'espérer. Qu'on songe aux cas suivants :

- l'apostrophe est, la plupart du temps, la marque d'une séparation entre deux mots : "j'ai", "l'arbre", "d'un", etc. sont bien constitués de deux "mots". Pourtant, "aujourd'hui" et "prud'homme", malgré l'apostrophe qu'ils contiennent, ne sont généralement considérés que comme un seul mot.
- le point est apparemment un séparateur assez puissant pour isoler les phrases. Mais suffit-il pour autant à empêcher que "I.B.M.", "11.09.2001" ou "M. Dupont" ne semblent constituer qu'une seule unité ?
- le tiret est un séparateur encore plus problématique : qu'est-ce qui nous autorise à ne trouver qu'un seul mot dans "porte-monnaie" ou "entre-déchirer", et à en trouver plusieurs dans "(cet) homme-là", "est-ce-que", "voulez-vous" ? Et combien de mots y a-t-il dans "y a-t-il" ou dans "c'est-à-dire" ?
- même le caractère blanc n'est pas un séparateur fiable : on a bien envie de faire de "parce que" ou de "pomme de terre" un seul mot, tandis que "du" et "au", qui résultent d'un amalgamme ("du" pour "de le", "au" pour "à le"), en font plutôt deux à eux tous seuls.

A défaut d'être reconnaissables par leur forme, les mots le seraient-ils par leur sens ? Un mot pourrait-il être caractérisé comme *la plus petite unité de sens possible* ? Là encore, le critère se révèle insatisfaisant :

– faut-il considérer que “chat” et “chats” sont exactement le même mot ? Le “s” du second ne porte-t-il pas, à lui tout seul, une nuance de sens spécifique ? De même, le simple “i” qui fait la différence entre “aimons” et “aimions” ne change-t-il pas tout ?
– le sens d’un mot unique comme “inimitables” semble bien décomposable en unités de sens élémentaires, au point que si le verbe “troufigner” avait un jour un sens, on en déduirait immédiatement celui de “introufignable”.

Le mot, décidément, n’est pas une engeance très fiable. Saussure, on l’a vu, lui préférait la notion de “signe” tandis que Martinet parlait, lui, de “monème”. La linguistique contemporaine utilise plutôt le terme de “morphème”, avec la définition suivante : *un morphème est une unité linguistique minimale ayant une forme et un sens*. L’étude des morphèmes et de leurs modes de combinaison est l’objet de la *morphologie*. La morphologie que nous présentons par la suite est essentiellement celle de la langue française.

La définition d’un morphème reprend en quelque sorte les critères qui avaient été envisagés pour le mot, mais sans contraindre la délimitation par des séparateurs. On peut très bien, ainsi, considérer que “pomme de terre”, “parce que” ou même “casser sa pipe” ne constituent chacun qu’un seul morphème. On peut même admettre l’existence de *morphèmes discontinus* à l’intérieur desquels peuvent s’insérer d’autres morphèmes : c’est le cas par exemple, en français, de la marque de la négation (“je **ne** dort **pas**”) ou du passé composé “j’**ai** bien dorm**i**”).

Le découpage en morphèmes d’un énoncé peut s’avérer problématique. Par exemple, même si les mots de “pomme de terre” ne forment, la plupart du temps, qu’un seul morphème, dans une phrase comme “Pour les protéger, il a recouvert les pommes de terre”, ils sont à prendre comme autant de morphèmes distincts. Dans quelle mesure les expressions figées idiomatiques comme “prendre la clé des champs” ne font qu’un seul ou plusieurs morphèmes n’est pas non plus facile à trancher.

1.2 Les différents types de morphèmes

On répartit traditionnellement les morphèmes en deux groupes : les *morphèmes lexicaux* et les *morphèmes grammaticaux*. Pour opérer cette distinction, plusieurs critères sont possibles :

– critère sémantique : les morphèmes lexicaux ont la particularité “d’avoir un sens par eux-mêmes”, de “référer à quelque chose dans le monde (objet, action, propriété...)”, alors que ce n’est pas le cas des morphèmes grammaticaux. On range ainsi parmi les premiers, sans trop de difficultés, les noms communs, les verbes et les adjectifs. Le deuxième groupe accueille, lui, naturellement, comme son nom l’indique, les mots qui ont essentiellement un rôle grammatical, syntaxique. C’est le cas des déterminants (“un”, “le”, etc.), des prépositions (“à”, “pour”, “avec”, etc.), des conjonctions (“qui”, “donc”, etc.) ainsi que des auxiliaires (“être” et “avoir”) et des morphèmes qui caractérisent le genre et le nombre des noms, les temps de conjugaison et les personnes des verbes. Mais ce critère sémantique est trop vague et pas vraiment opérationnel : où ranger, par exemple, les pronoms (“je”, “celui”, etc.) et les adverbes (“beaucoup”, “demain”, etc.) ?

– critère énumératif : les morphèmes grammaticaux appartiennent à une liste fermée, non extensible, tandis que les morphèmes lexicaux appartiennent à une liste ouverte, pouvant toujours évoluer dans le temps. Quand on joue à inventer de nouveaux mots, ce sont toujours des morphèmes lexicaux (cf le "troufigner" précédent). Personne ne s'amuserait à créer une nouvelle conjonction de coordination, sans du moins prendre le risque de ne pas du tout se faire comprendre... Ce critère-là est nettement plus facile à utiliser. Il nous permet de classer les pronoms parmi les morphèmes grammaticaux (on peut en faire la liste exhaustive), et les adverbes avec les morphèmes lexicaux (on ne disait certainement pas "c'est mega bien" il y a quelques années...).

Dans certaines traditions linguistiques, on utilise le terme "lexème" pour désigner ce que nous appelons ici "morphème lexical", et "morphème" pour ce que nous nommons ici "morphème gramatical". Nous nous en tiendrons par la suite à nos termes initiaux.

Cette distinction entre morphèmes lexicaux et grammaticaux est sans doute plus une affaire de degré que de coupure franche. Certaines prépositions comme "dans", par exemple, ont une sémantique plus riche que d'autres comme "de", et on peut discuter de leur classification.

Ce découpage présente aussi l'inconvénient de laisser de côté une famille de mots qui jouent un rôle fondamental dans le TALN à l'heure actuelle : il s'agit des *entités nommées*. On nomme ainsi depuis quelques années la classe des expressions qui désignent soit des "noms propres" (de lieux, de personnes, d'organisations, etc.), soit des "valeurs numériques" (dates, nombres, etc.). Ces mots ou groupes de mots ont longtemps été oubliés des classifications parce qu'ils ne figurent pas dans les dictionnaires usuels. Mais de nombreux textes (par exemple, les articles de presse) en sont truffés, et ils contribuent au sens de ces textes de façon déterminante. Toutes les applications de fouille de textes fondées sur le sens se doivent donc de les identifier et de les prendre en compte. Cela justifie qu'ils font l'objet de recherches intensives depuis quelques années. Nous y reviendrons plus loin.

Il ne fait pas de doute que les entités nommées appartiennent à une liste ouverte : tout le monde peut inventer le nom de sa nouvelle société, voire le prénom qu'il donne à ses enfants. Mais le caractère référentiel de ces expressions est un problème connu en logique depuis longtemps comme délicat. Nous préférerons donc les considérer comme à une classe à part.

1.3 Combinaisons de morphèmes

La morphologie étudie comment différents morphèmes se combinent entre eux pour former des unités plus complexes qu'on appelera *unités lexicales*, parce qu'elle vérifient en général les critères associés aux morphèmes lexicaux (elles réfèrent à quelque chose et on peut en créer de nouvelles). On distingue habituellement au moins deux façons d'opérer des combinaisons : la *composition* et l'*affixation*.

La composition fonctionne par simple concaténation (mise bout à bout) de morphèmes lexicaux. Elle est à l'œuvre dans des exemples comme :

– bon + homme = bonhomme
– porte + manteau = portemanteau

– chou + fleur = chou-fleur

On peut étendre ce mécanisme à certains groupes comme “pomme de terre” ou “laisser tomber”. Ce qui justifie que ces derniers se comportent comme un seul morphème (sauf dans la phrase citée précédemment !), c'est que leur sens ne se réduit pas à une combinaison des sens des mots qui les composent, et qu'il est quasiment impossible dans un énoncé d'intercaler d'autres mots entre eux.

L'affixation est un mécanisme plus complexe qui fait interagir des morphèmes lexicaux, sous la forme de “racines”, et des morphèmes grammaticaux, les “affixes”. Les affixes sont des morphèmes non autonomes, c'est-à-dire qu'ils ne peuvent exister sans être rattachés à une racine. Ils sont eux-mêmes de deux types distincts :

– les *affixes dérivationnels* ont un contenu quasi-lexical : ce sont soit des *préfixes* comme “de”, “re”, etc. soit des *suffixes* comme “eur”, “ement”, etc. ;
– les *affixes flexionnels* sont ceux qui servent à exprimer les variations en genre et en nombre des noms et des adjectifs, et les conjugaisons des verbes. Ils se positionnent toujours en français *après* les affixes dérivationnels.

La figure 4.1 montre comment fonctionne l'affixation en français, et l'illustre sur le mot “in-imit-able-s”. Sur cette figure, les affixes sont mis entre parenthèses parce qu'ils sont facultatifs.

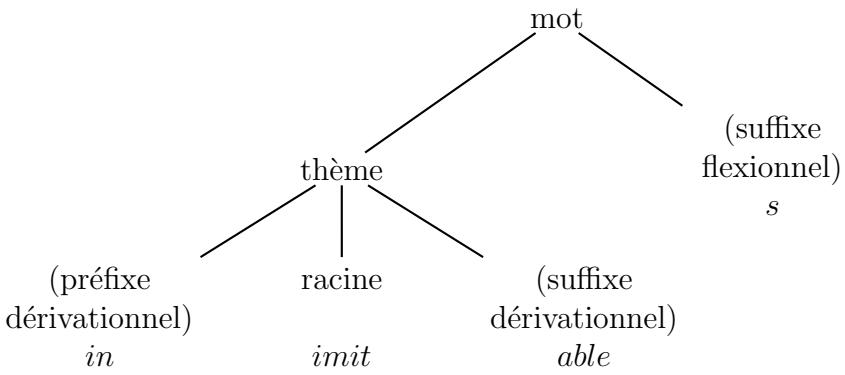


FIG. 4.1 – l'affixation en français

Une même unité lexicale peut être construite à l'aide de plusieurs affixations successives. Les suffixes ont la particularité (que n'ont pas les préfixes) de pouvoir changer la catégorie grammaticale de la racine à laquelle ils s'associent, comme l'illustrent les exemples du tableau suivant :

mot			
racine	suffixe		
	nominal	adjectival	adverbial
am	i our ant	cal eu(x)	ement sement

Notons enfin que les entités nommées qui désignent des noms propres peuvent, elles aussi, subir des dérivations : des unités lexicales comme “néo-nantais”, “pseudo-RMIste”, “crypto-chomskyen”, etc. dérivent de telles entités.

1.4 Les informations associées à une unité lexicale

Nous aurons parfois tendance à assimiler la notion d'unité lexicale à celle de "mot", dont il est tout de même difficile de se passer (même si c'est clairement un abus de langage). On associe traditionnellement à chaque unité lexicale présente dans un énoncé un ensemble de propriétés, que nous passons maintenant en revue.

Tout d'abord, on peut associer à chaque unité lexicale une *forme lemmatisée* : c'est la forme sous laquelle on va trouver cette unité dans un dictionnaire de la langue courante. Elle correspond à un choix arbitraire parmi les flexions possibles que peut subir l'unité en question. En français, la forme lemmatisée (ou *lemme*) des noms et des adjectifs est celle du masculin singulier, tandis que celle des verbes est l'infinitif. Les autres unités sont invariables et se confondent donc avec leur lemme. On ne parle pas en général de forme lemmatisée pour les entités nommées (mais elles sont souvent aussi invariables).

Nous avons déjà incidemment évoqué le fait que les unités appartiennent à des *catégories grammaticales* : certaines sont des *noms communs*, d'autres des *verbes* ou des *adverbes*, etc. Les anglos-saxons parlent d'étiquettes "part-of-speech" (ou POS). D'où provient cette classification et quel critère permet de ranger dans la même catégorie des unités distinctes ? Ce n'est pas pour rien que ces catégories sont qualifiées de "grammaticales". La notion de syntaxe et de grammaire ne sera abordée que dans le chapitre suivant, mais on peut d'ors et déjà voir son influence au niveau lexical. Plus précisément, le critère décisif qui nous est ici utile est celui de *grammaticalité* (nous y reviendrons au cf. chapitre 5, section 1.1). Depuis Chomsky, une grammaire est vue comme un dispositif capable de trier les suites de mots d'une langue donnée en "grammaticales" ou "non grammaticales", et cela indépendamment de leur sens. Cette capacité à formuler des *jugements de grammaticalité* est ce qui constitue la *compétence* d'un locuteur. Si on admet l'existence d'une telle capacité, alors on dira que *deux unités lexicales appartiennent à la même catégorie si on peut remplacer l'une par l'autre dans n'importe quel énoncé, sans modifier sa grammaticalité.* Ainsi, par exemple :

- les mots "livre" et "rhinocéros" appartiennent à la même catégorie parce que "le livre est sur l'étagère" et "le rhinocéros est sur l'étagère" (par exemple) sont tous les deux des énoncés grammaticaux.
- "livre" et "regarder" (à quelque forme conjuguée que ce soit) n'appartiennent pas à la même catégorie parce que "le regarde est sur l'étagère" n'est pas grammatical.

Ce critère rappelle celui employé pour définir la notion de phonème (cf. chapitre 3, section 1.2). On peut dire aussi que les catégories grammaticales sont des *classes d'équivalence* pour la relation de *substituabilité en préservant la grammaticalité de l'unité de niveau supérieur (l'énoncé).* Il est intéressant de noter que, pour caractériser les propriétés d'un certain niveau d'analyse, on doive systématiquement faire appel au niveau d'analyse supérieur. Décidément, il est bien difficile de les isoler les uns des autres.

En réalité, il ne faut pas utiliser ce critère de façon trop rigoureuse pour caractériser les classes. Si on le suivait de trop près, on ne pourrait associer de catégorie aux morphèmes grammaticaux : il faudrait en effet définir autant de catégories

différentes qu'il existe de prépositions ou de conjonctions, parce qu'elles sont rarement substituables les unes aux autres (on ne peut impunément remplacer "à" par "pour" ni "qui" par "dont"). Le critère fonctionne nettement mieux pour les morphèmes lexicaux que pour les morphèmes grammaticaux, sur lesquels, précisément, repose en grande partie la structure grammaticale des énoncés qui les contiennent.

Par ailleurs, les accords en genre et en nombre imposés par la langue française font que, pour préserver la grammaticalité d'un énoncé, un nom commun féminin singulier ne peut être en général remplacé que par un autre nom commun féminin singulier. Les informations flexionnelles sont pourtant généralement abstraites de la définition d'une catégorie, ce qui autorise à ranger les noms masculins et féminins dans la même classe. De même, la catéorie des "verbes" n'est pas homogène puisqu'il faudrait distinguer :

- les verbes *intransitifs* comme "dormir", qui caractérisent un individu sujet unique (on dit aussi qu'ils ne "réclament pas de complément d'objet", mais on n'a pas encore vraiment défini ces termes)
- les verbes *transitifs* comme "aimer" qui caractérisent une relation entre deux individus : un sujet et un objet
- les verbes *bitransitifs* comme "donner" qui caractérisent une relation à trois : un sujet et deux objets (quelqu'un donne quelque chose à quelqu'un d'autre).

Les substitutions entre verbes ne peuvent préserver la grammaticalité qu'en tenant compte de ces propriétés. Mais ces distinctions ne sont pas toujours faites et on se contente souvent d'une unique classe "verbe". En fait, il n'y a pas consensus quant au nombre de classes à considérer, ni quant à leurs frontières précises.

Remarquons aussi que de nombreux mots appartiennent à plusieurs classes distinctes. Parfois, ce sont plutôt des homonymes, c'est-à-dire des mots différents qui se trouvent avoir la même forme : les instances de "ferme" qui sont respectivement un nom commun, un verbe et un adjectif n'entretiennent pas de rapports entre eux. Mais il y a un lien très fort au contraire entre les mots "petit", "juste", "informatique"... employés comme noms communs ou comme adjectifs, de même qu'entre les usages de "fort," "clair"... en tant qu'adjectifs ou en tant qu'adverbes. L'attribution de ces mots à une classe plutôt qu'à une autre (la nature de "l'étiquette part-of-speech" qui leur est associée) dépend de l'énoncé dont ils font partie.

Enfin, on associe traditionnellement aux unités lexicales présentes dans un énoncé des informations de flexion : le genre, le nombre et éventuellement le cas (qui permet de distinguer les pronoms "le" et "lui") pour les noms, les pronoms et les adjectifs ; la personne, le nombre, le temps, le mode ("indicatif", "subjonctif", etc.) et la voix ("active" ou "passive") pour les verbes.

2 Modélisation informatique

La capacité de mémoire des ordinateurs actuels est telle qu'elle permet de stocker facilement *l'intégralité des formes fléchies d'une langue*. On a pourtant parfois intérêt à enregistrer ces différentes formes de façon synthétique. Pour cela, nous évoquons ici deux approches possibles. La première est simplement une *structure de données*, c'est-à-dire une manière efficace de coder des informations, en l'occurrence des listes

de mots partageant des lettres communes. La deuxième est un modèle très puissant issu de l'informatique théorique, dont nous verrons d'autres usages possibles plus loin : les automates finis. Nous montrons ici qu'ils permettent de représenter les découpages morphologiques sous la forme de *règles* plutôt que sous la forme de *listes*. En écrivant des règles, en effet, on *généralise* et on peut faire des *prédictions*, notamment sur la forme des mots nouveaux qui peuvent apparaître dans la langue et la façon dont leurs variantes morphologiques pourront décliner leurs variantes de sens.

2.1 Arbre à lettres

Pour stocker un dictionnaire de façon économique, il existe des organisations plus efficaces que les simples listes. Prenons l'exemple de la liste (arbitraire, mise à part son ordre alphabétique) de noms communs sous forme lemmatisée suivante : {abri, abus, an, anse, art, arme, as, astre}. Pour la coder efficacement, les informaticiens emploient la structure de la figure 4.2, appelée "arbre à lettres".

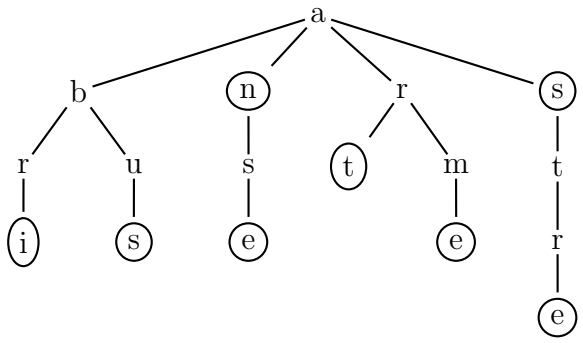


FIG. 4.2 – un arbre à lettres

Cette structure est déjà la deuxième (après la figure 4.1) que nous rencontrons qui prend la forme d'un *arbre*. Ce n'est pas la dernière ! Le point de départ de ces arbres (ici la lettre *a*), bien qu'habituellement situé en haut de la figure, s'appelle aussi leur *racine*, tandis que chaque point intermédiaire (ici une lettre) est un *nœud*. Ces nœuds sont reliés les uns aux autres par des *branches* qui se développent de haut en bas jusqu'à des *feuilles*. Les *fils* d'un nœud intermédiaire (c'est-à-dire qui n'est pas une feuille) sont les nœuds situés au niveau immédiatement inférieur et reliés au premier par une branche. Un chemin est une succession de nœuds partant de la racine et suivant les branches en descendant. Dans un arbre à lettres, quand en suivant un tel chemin on parvient à épeler un mot complet de la liste, alors on entoure le nœud auquel on est parvenu. C'est le cas, bien sûr, pour les feuilles de l'arbre qui, toutes, correspondent à la fin d'un nom de la liste mais aussi pour certains nœuds intermédiaires (comme le *n* de "an" et le *s* de "as").

Certains index des logiciels documentaires, des bases de données, des correcteurs orthographiques ou des moteurs de recherche sont enregistrés dans la mémoire des ordinateurs sous cette forme. C'est grâce à de telles structures qu'il est facile pour les machines d'anticiper sur les frappes au clavier de certaines lettres. Certains chemins, en effet, ne mènent que vers un seul mot possible.

Mais les arbres à lettres ne rendent absolument pas compte de l'organisation morphologique des mots : les morphèmes n'y sont pas du tout apparents. Pour remédier à ce problème, on va introduire une structure un peu plus complexe, qui peut être interprétée comme un programme d'analyse ou de synthèse morphologique : le modèle des *automates finis*.

2.2 Automates finis

Un automate fini (ou automate à états finis) est un dispositif constitué de :
- un *vocabulaire* fini V : dans notre cas, le vocabulaire sera constitué de l'ensemble des morphèmes considérés ;
- un ensemble fini Q d'*états*, parmi lesquels figurent :
  - au moins un état *initial* ;
  - au moins un état *final* ;
- une fonction de *transition* f : cette fonction énumère, pour chaque état possible $q \in Q$ et chaque élément possible du vocabulaire $v \in V$, l'état (ou l'ensemble d'états) que l'on peut atteindre en partant de q et en utilisant v : $f(q, v) \subseteq Q$.

Un automate est un modèle informatique parce qu'il se traduit de façon quasi immédiate en un programme.

Prenons tout de suite l'exemple d'un automate élémentaire permettant de rendre compte de régularités de découpages morphologiques visibles dans des mots comme : "inimitables", "imitée", "analysées", "inanalysable", "désirable", "indésiré", "innomable"', etc. La liste des morphèmes pertinents dans ce cas est constituée du préfixe "in", des racines "imit", "désir", "analys" et "nomm" (on pourrait bien sûr en ajouter d'autres), des suffixes "able" et "é" et des flexions "e" et "s". Ces morphèmes définissent l'ensemble V de notre automate.

Pour définir les états et les transitions, il est plus simple de visualiser le *graphe* de l'automate. A tout automate, en effet, on peut associer une représentation graphique, que l'on appelera par la suite un *graphe*. La figure 4.3 donne le graphe d'un des automates possibles qui répond à notre problème.

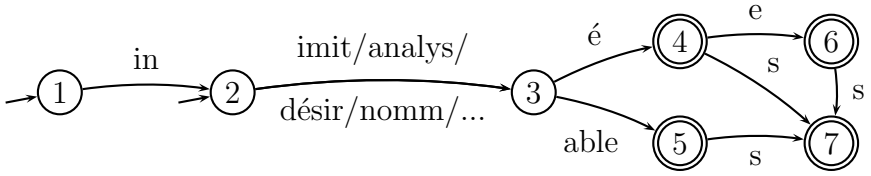


FIG. 4.3 – un automate morphologique

Dans cette représentation, les états sont représentés par des "ronds" numérotés. Q est donc ici l'ensemble des nombres de 1 à 7 (l'ordre de numérotation est totalement arbitraire). Les états initiaux sont ceux sur lesquels arrive un flèche "qui ne vient de nulle part" : c'est le cas dans notre exemple des états 1 et 2. Les états finaux, eux, sont visualisés par un "double rond" : ce sont les états 4, 5, 6 et 7. Rien n'empêcherait en théorie qu'un état puisse être à la fois initial et final, mais il n'y en a pas dans notre exemple.

Chaque flèche dont le point de départ et le point d'arrivée sont des états et qui porte une étiquette prise dans le vocabulaire V correspond à une *transition*. Par

exemple, la flèche entre les états 3 et 5 est la traduction du fait que la fonction de transition f prend pour valeur 5 quand elle a pour données 3 et *able*, autrement dit : $f(3, able) = 5$. Entre les états 2 et 3, il faut considérer qu'il y a en réalité 4 flèches différentes, chacune portant un morphème distinct. C'est uniquement pour des questions de lisibilité qu'on n'en a représenté qu'une seule (les "/" séparant les morphèmes signifient qu'il faut en choisir un parmi ceux énumérés). Rien n'empêcherait en théorie qu'une flèche ait pour point de départ et point d'arrivée le même état (elle serait représentée alors par une "boucle" sur cet état) , mais il n'y en a pas dans notre exemple.

Un automate permet de caractériser un ensemble de *chemins*. Un chemin dans un automate commence nécessairement à partir d'un état initial, suit des transitions et aboutit à un état final. Le *langage* de l'automate est l'ensemble de suites (ou concaténations) d'éléments du voacabulaire V qui correspondent à un tel chemin. Le langage de notre automate contient ainsi tous les mots évoqués au début de cette section, et toutes leurs variantes morphologiques. C'est un *langage fini* parce qu'on peut énumérer tous ses éléments dans une liste finie. Il existe d'autres automates reconnaissant le même langage. Par exemple, un automate avec seulement un état initial et un état final, et autant de transitions que de mots distincts conviendrait tout aussi bien en termes de langage. Mais celui de la figure 4.3 met en évidence des régularités de construction que n'auraient pas l'autre (équivalent à une simple liste). Nous l'avons de plus construit pour que chaque état final identifie une variante flexionnelle distincte :

- l'état 4 identifie le masculin singulier ;
- l'état 6 identifie le féminin singulier ;
- l'état 5 identifie le masculin ou le féminin singulier ;
- l'état 7 identifie le pluriel (on aurait pu distinguer "masculin pluriel" et "féminin pluriel" pour les formes en "é(e)s" mais pas pour les formes en "able", d'où cette indifférenciation).

Les automates sont particulièrement bien adaptés à la modélisation des phénomènes d'affixation. On peut ainsi facilement construire des automates qui explicitent les règles de conjugaison des verbes, mais il faut alors autant d'automates différents qu'il y a de familles de conjugaisons différentes possibles. Dans ce cas, on peut aussi faire en sorte que les états finaux distincts identifient les différentes personnes (1ère, 2ème ou 3ème) et le nombre (singulier ou pluriel) de la conjugaison. Le verbe "troufigner", imaginé dans les sections précédentes, aurait ainsi toutes les chances de se conjuguer comme "aligner" et tous les autres verbes réguliers du 1er groupe. Le fait de déjà disposer d'un automate pour caractériser cette conjugaison évite d'en inventer une nouvelle, et économise donc son stockage en mémoire.

Un des principaux intérêts des automates (comme des grammaires formelles dont nous verrons plus loin qu'ils ne sont qu'un cas particulier) est qu'ils peuvent fonctionner *aussi bien en analyse qu'en synthèse*. En analyse, on dit que l'automate "reconnaît" un mot s'il existe un chemin correct étiqueté par ce mot. En synthèse, on dit qu'on "génère" ou "engendre" un mot en le produisant au fil des transitions de l'automate.

2.3 Expressions régulières

Les automates finis ont été très étudiés par Chomsky et les pionniers de l'informatique théorique, à partir des années 60, et un très grand nombre de leurs propriétés formelles ont alors été explicitées. Nous n'évoquerons ici que quelques-unes d'entre elles, sans rentrer dans les détails techniques et sans fournir aucune démonstration.

Les automates finis reconnaissent (ou engendrent, suivant le point de vue adopté) une *classe de langages* particulière appelée *langages réguliers* ou *langages rationnels*. Un langage est régulier (ou rationnel) s'il existe un automate qui reconnaît exactement ce langage. On peut aussi caractériser cette famille de langages par d'autres moyens. Nous verrons plus loin une caractérisation en termes de grammaires formelles. Ici, nous présentons la caractérisation par les *expressions régulières*.

Une *expression régulière* est une suite de symboles prise parmi :
- un vocabulaire fini V
- le symbole ϵ qui représente la "chaîne vide"
- les symboles : "|" et ".", ainsi que les parenthèses "(" et ")"
- les symboles "+" et "*" utilisés en exposants

Elle doit être de plus bâtie en respectant les règles de construction suivantes :
- tout élément de $V \cup \{\epsilon\}$ est une expression régulière ;
- si U est une expression régulière, alors (U) est aussi une expression régulière : les parenthèses sont utilisées comme en mathématiques, pour délimiter des sous-parties d'expressions régulières ;
- si U et T sont toutes les deux des expressions régulières, alors $U|T$ et $U.T$ (souvent réduite à UT) sont des expressions régulières : $U|T = U \cup T = \{w|w \in U \text{ ou } w \in T\}$ représente le choix entre un élément de U et un élément de T, tandis que $UT = \{ut|u \in U \text{ et } t \in T\}$ représente l'ensemble des concaténations d'un élément de U et d'un élément de T ;
- si U est une expression régulière, alors U^+ et U^* sont des expressions régulières : $U^+ = \{u_1u_2...u_n|\forall i,\ u_i \in U\}$ représente l'ensemble des concaténations un nombre quelconque de fois (au moins une) d'éléments de U, et $U^* = U^+ \cup \{\epsilon\}$ est la même concaténation, y compris 0 fois (ce qui donne la chaîne vide).

Le symbole ϵ joue le rôle d'élément neutre pour la concaténation : pour tout symbole $w \in V$, $w.\epsilon = \epsilon.w = w$. A titre d'exemple, l'automate de la figure 4.4 construit sur le vocabulaire $V = \{a, b\}$ reconnaît le langage constitué d'un nombre quelconque (éventuellement nul) de symboles a suivi d'un nombre quelconque mais non nul de symboles b (parce que tout chemin, pour passer de l'état initial 1 à l'état final 2, doit obligatoirement emprunter la transition étiquetée par b entre ces deux états). Il peut être représenté par l'expression régulière : a^*b^+.

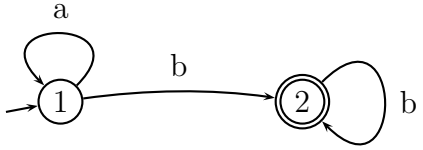


FIG. 4.4 – un automate fini dont le langage est a^*b^+

Le langage a^*b^+ est un *langage infini*, puisqu'il contient un nombre non borné de mots différents : $a^*b^+ = \{b, ab, bb, aab, abb, bbb, ...\}$. On voit que la puissance

des automates finis et des expressions régulières vient de ce qu'ils permettent de caractériser un nombre infini de mots différents à l'aide d'un nombre fini d'éléments. Néanmoins, si les langues naturelles contiennent un nombre infini d'unités lexicales, c'est grâce au fait que les morphèmes lexicaux constituent une liste ouverte (et qu'on peut toujours inventer de nouveaux mots) et non grâce à l'expressivité des affixations, telles qu'on peut les exprimer dans des automates finis.

Tout langage régulier peut être exprimé sous la forme d'une (ou plusieurs) expression(s) régulière(s) (et inversement). Passer d'un automate à une expression régulière ou inversement n'est pas toujours simple. Le langage (fini) reconnu par l'automate morphologique précédent, en figure 4.3, peut être représenté par exemple par l'expression régulière complexe suivante :
(in|ϵ).(imit|analys|désir|nomm).((é.(ϵ|e|s|es))|able(ϵ|s))

2.4 Sites Web et programmes gratuits

De nombreux sites réalisent en ligne des traitements se situant au niveau des mots ou des morphèmes.

- comptages, statistiques (à partir de textes français) :
  - www.lexique.org/
  - erss.irit.fr :8080/voisinsdenface/vdef
- décompositions morphologiques :
  - www-clips.imag.fr/trilan/Pilaf/ (français)
  - www.cis.hut.fi/cgi-bin/morpho/nform.cgi (anglais)
  - www.xrce.xerox.com/competencies/content-analysis/demos/french (français)
  - www-timc.imag.fr/Delphine.Bernhard/index.php ?rub=3&lang=fr (français)
- étiquetages en catégories :
  - www.cele.nottingham.ac.uk/∼ccztk/treetagger.php (multilingue)
  - garraf.epsevg.upc.es/freeling/demo.php (espagnol)
  - ucrel.lancs.ac.uk/claws/trial.html (anglais)
- repérage des entités nommées :
  - http ://l2r.cs.uiuc.edu/∼cogcomp/eoh/nedemo.html (multilingue)

Parmi les programmes libres et gratuits qui illustrent les notions introduites dans ce chapitre, on peut citer Unitex, logiciel fondé sur les automates et les RTRs (cf. chapitre suivant). Quand on charge un texte avec Unitex, ce dernier lui applique un certain nombre de pré-traitements (normalisations, comptages, lemmatisations, etc.) qui, tous, sont réalisés à l'aide d'automates facilement visualisables et modifiables. L'utilisateur peut aussi écrire ses propres automates pour, par exemple, effectuer la recherche d'une expression régulière dans un texte :
www-igm.univ-mlv.fr/∼unitex/

Chapitre 5

Le niveau de la syntaxe

Le niveau de la syntaxe explique comment mettre bout à bout des unités lexicales afin de bâtir des énoncés dont le sens global est plus que la simple somme des sens de ces unités. Il constitue la "première articulation" de toute langue naturelle. Sous l'influence de Chomsky, ce niveau est celui qui a fait l'objet du plus grand nombre de travaux, d'études et de modèles ces 50 dernières années. La présentation qui en sera faite ici ne pourra être donc que très partielle et partiale.

1 Description linguistique

1.1 De l'analyse distributionnelle à la notion de grammaticalité

Jusqu'aux années 50, le courant dominant en matière de description linguistique (particulièrement aux Etats-Unis) s'appelle l'*analyse distributionnelle*. Suivant cette approche, pour étudier une langue, il faut tout d'abord disposer d'un échantillon aussi représentatif que possible de cette langue : un *corpus*. Une langue n'a pas besoin d'être comprise pour être étudiée : toutes ses propriétés doivent pouvoir être extraites des régularités et redondances observées dans le corpus.

La distribution d'une unité présente dans un corpus est définie comme l'ensemble de ses environnements, c'est-à-dire des suites d'unités qui la précédent et qui la suivent dans ce corpus, dans une fenêtre dont la taille est bornée à l'avance. L'ensemble des unités qui partagent un environnement constituent une classe distributionnelle. Par exemple, on peut espérer que "bébé" et "marmot" appartiennent à la même classe parce qu'ils doivent apparaître, dans tout bon corpus, dans les mêmes environnements (précédés de "le" et suivis de "pleure", par exemple...). On peut définir une grammaire, dans un tel système, comme un ensemble de classes et de listes d'environnements associés. En d'autres termes, une grammaire n'est rien d'autre que l'usage distributionnel qui est fait de ses unités linguistiques. Un des problèmes de cette théorie est en fait d'expliquer comment caractériser la notion d'"unité" par des critères purement observationnels.

Mais Harris, un des principaux promoteurs de l'analyse distributionnelle, eut un étudiant qui s'appelait Chomsky. Pour Chomsky, tout corpus est nécessairement incomplet parce fini, alors qu'une langue permet de construire un nombre potentiel-

lement infini de phrases différentes à partir d'un nombre fini d'unités. Pour ranger les unités dans des classes, ce n'est pas l'existence d'environnements communs dans le corpus qui sera déterminante, mais un critère nouveau fondamental : celui de *grammaticalité*.

Une grammaire, dans ce cadre, est en effet un dispositif capable d'opérer des *jugements de grammaticalité*, c'est-à-dire de *trier* les suites d'unités en "correctement formées" (grammaticales) ou non, comme dans le schéma de la figure 5.1. C'est ce dispositif qui caractérise la *compétence* d'un locuteur (cf. la présentation des concepts chomskiens dans la partie historique du chapitre 2).

FIG. 5.1 – rôle d'une grammaire selon Chomsky

Pour bien comprendre les conséquences de cette conception nouvelle, il faut mesurer les distinctions suivantes :

- la grammaticalité est différente de la *fréquence d'apparition dans un corpus*. Par exemple, dans un contexte comme "le petit _ est mort" (où le tiret remplace une unité lexicale), les unités "téléphone" et "chanterons" sont (quasiment) tout aussi improbables. Pourtant, le statut des deux énoncés ainsi construits ne serait pas le même : "le petit téléphone est mort" est parfaitement grammatical, ce qui n'est pas le cas de "le petit chanterons est mort".
- la grammaticalité n'est pas non plus synonyme d'"interprétabilité". Chomsky propose un exemple qui a été beaucoup repris et commenté à ce sujet : "d'incolores idées vertes dorment furieusement" est un énoncé grammatical, mais auquel il est pour le moins difficile d'attribuer un sens. Inversement, un énoncé comme "vous faire moi rigoler" est interprétable quoique non grammatical.

En fait, Chomsky propose de remplacer le critère empirique observable de "présence dans un corpus" par un critère mental plus abstrait, dont les effets ne sont visibles qu'indirectement : on peut demander à un locuteur d'opérer autant de jugements de grammaticalité que l'on veut, même s'il est incapable d'expliquer comment il s'y prend. Sa "grammaire mentale" est pour lui une "boîte noire" dont il ne perçoit que les entrées et sorties. Le rôle du linguiste est d'essayer d'ouvrir la boîte noire...

Notons enfin que l'analyse distributionnelle connaît un certain regain d'intérêt ces dernières années. La diffusion d'Internet et des documents électroniques a en effet permis la constitution de corpus numériques, sur lesquels certains des tests promus par cette analyse ont pu être programmés. Nous reviendrons sur cette vision des choses dans le chapitre 8 de ce document.

1.2 Des phrases aux propositions

Avant de détailler les *structures syntaxiques* mises en évidence par la linguistique contemporaine, une question préliminaire se pose : celle de la nature des "suites d'unités" que l'on peut soumettre au test de grammaticalité, autrement dit auxquelles on peut associer de telles structures. Nous avons jusqu'à présent évité le terme de "phrase", parce qu'il pose le même genre de problèmes que celui de "mot"

(cf. chapitre 4, section 1.1). Une phrase, en effet, est très difficile à caractériser, aussi bien avec des critères formels (suite de mots entre deux séparateurs forts parmi $\{., !, ?\}$?) qu'avec des critères sémantiques.

Plusieurs unités de descriptions sont constituées de plusieurs "mots" tout en restant généralement plus petites qu'une phrase. Parmi celles-ci, citons :

- les *chunks* : plus petites séquences de mots auxquelles on peut associer une catégorie (cf. plus loin suivant quel critère) comme "groupe nominal" ou "groupe verbal". Mais un tel groupe ne constitue un chunk que si lui-même ne contient pas un autre groupe de même nature. Par exemple, dans "le chat du voisin", il y a en fait deux chunks distincts : "le chat" et "du voisin". Mais cette unité n'apporte pas suffisamment de propriétés nouvelles pour justifier de passer à un "niveau d'analyse" fondamentalement nouveau.
- les *termes* : noms communs, entités nommées ou groupes nominaux éventuellement composés d'autres groupes nominaux. Les *termes* identifient un concept précis dans un domaine de spécialité (comme "maladie de la vache folle") et peuvent de ce fait servir de "mots clés" dans une indexation. Clairement, ils ne suffisent pas non plus à eux seuls à faire des phrases.
- les *clauses* : séquences de mots contenant au moins un sujet et un prédicat (notions sémantiques sur lesquelles nous reviendrons). Mais, comme presque toujours en linguistique, aucune définition ne fait vraiment consensus à leur sujet. Une phrase peut en général se découper en plusieurs clauses emboîtées les unes dans les autres, comme dans l'exemple suivant (traduit de l'anglais), où chaque couple de parenthèses marque les frontières d'une clause : "((La dérégulation des compagnies de chemins de fer, qui a commencé en 1980)), a permis (aux affréteurs de marchandises de négocier leurs tarifs.))".

En fait, la seule structure intermédiaire dont nous ferons usage par la suite est celle de *syntagme*. Un syntagme est un mot ou une suite de mots consécutifs auquel on peut associer une catégorie syntaxique, sur la base du critère de substituabilité que nous avons déjà employé dans le chapitre 4, section 1.4. Mais cette notion, aussi pertinente soit-elle, ne justifie pas à elle toute seule qu'on lui consacre un "niveau d'analyse" spécifique, puisque, justement, les informations qu'on peut lui associer sont déjà présentes au niveau des mots.

Les linguistes utilisent souvent la notion d'"énoncé", en tant qu'unité de production textuelle ou de prise de parole. Mais ce terme reste encore un peu trop vague à notre goût. En fait, ce qui nous intéresse dans une "phrase", c'est sa capacité à *dire des choses sur le monde*. Ce qui à nos yeux légitime l'existence d'un niveau d'analyse spécifique, c'est donc l'existence de suites d'unités auquel on peut attribuer une *valeur de vérité (vrai/faux)*. Une telle suite s'appelle une *proposition*. Ce critère distinctif a l'avantage de caractériser un niveau de combinaisons non réductible aux précédents. Comme celui qui a permis de définir les "morphèmes", il est de nature *sémantique*. Son inconvénient est qu'il écarte certaines "phrases" courantes comme les exclamations ou les questions, qui ont pourtant une structure syntaxique qui mérite d'être étudiée. Nous pouvons toutefois considérer ces dernières comme des cas particuliers (on peut associer une valeur de vérité à un couple ⟨question, réponse⟩) et nous ne nous interdirons pas de les analyser.

Ce parti pris sémantique n'est certainement pas celui qui serait adopté dans une

perspective purement chomskienne qui, elle, entend dissocier la grammaticalité de l'interprétabilité et clame la primauté de la syntaxe sur la sémantique. Mais le point de vue que nous argumentons tout au long de ce document, c'est que *c'est toujours le niveau le plus complexe et le plus sémantique qui dirige les autres niveaux.* En dernière instance, c'est donc finalement le niveau de la sémantique propositionnelle (en bas à droite de notre schéma de la figure 2.1) qui guide l'ensemble de nos analyses.

Plutôt que de parler de "phrases" nous parlerons donc aussi souvent que possible de "propositions". Par la suite, il peut nous arriver d'employer le terme de "phrase", bien pratique, de même que nous utilisons celui de "mot" tout en ayant critiqué son caractère d'unité linguistique pertinente.

1.3 Structures syntaxiques

Commençons par illustrer comment procéder à une analyse syntaxique "syntagmatique" sur un exemple. Nous partons de la proposition "l'oiseau pose ses pattes sur une branche." Notre objectif est de décomposer cette suite d'unités en groupes adjacents qui "vont ensemble". Notre principal outil en la matière sera notre critère de "substituabilité", déjà plusieurs fois utilisé (cf. chapitre 3, section 1.2 et chapitre 4, section 1.4). Ce qui doit rester stable au fil des substitutions, ce sera de nouveau le critère de grammaticalité. Il s'agira d'appliquer au niveau des suites de mots la technique qui a déjà permis de regrouper dans une même *catégorie grammaticale* certains mots.

Repartons donc des affectations des mots présents dans cette proposition à ces catégories :

l'	oiseau	pose	ses	pattes	sur	une	branche
Det	Nom	Vtr	Det	Nom	Prep	Det	Nom

Dans la deuxième ligne de ce tableau, les abbréviations désignent les catégories grammaticales suivantes :

- Det est la catégorie des "déterminants" qui précèdent et introduisent les noms comumns : on voit qu'elle regroupe les articles définis (comme "le"), indéfinis ("une") et certains pronoms ("ses"). On ne distinguera pas par la suite le genre et le nombre de ces déterminants (pas plus qu'on ne le fera pour les noms ou les adjectifs) mais, bien sûr, il faudrait en tenir compte en théorie pour garantir les accords entre ces mots, qui sont partie intégrante de la grammaticalité ;
- Nom est la catégorie des noms communs
- Vtr est la catégorie des verbes transitifs (on notera Vintr celle des verbes intransitifs)
- Prep est la catégorie des prépositions comme "sur"

Pour aller plus loin, on se pose la question suivante : quelles suites de mots consécutifs dans cette phrase peut-on remplacer par une autre suite, voire par un unique autre mot en préservant la grammaticalité (à défaut du sens) de la suite ainsi construite ? Nous obtiendrons alors les réponses successives suivantes...

Tout d'abord, le groupe "l'oiseau" peut très bien être remplacé par un nom propre comme "Titi" (on préserve même alors une partie du sens, mais cela n'a rien d'obligatoire). C'est aussi le cas de "une branche" qui peut être substitué par "Jean",

ainsi que “ses pattes”, à la place duquel on peut mettre “Médor”. En effet “Titi pose Médor sur Jean”, bien qu'un peu surréaliste, est grammatical (de même que “Jean pose Jean sur Jean”, par exemple). Nous noterons l'ensemble de toutes les successions d'unités lexicales substituables à ces suites la classe des “groupes nomimaux” et nous la noterons GN. Nous avons déjà ainsi isolé dans notre proposition initiale des groupes adjacents que l'on peut visualiser par des parenthèses étiquetées :

$$(\text{l'oiseau})_{GN}\ \text{pose}\ (\text{ses pattes})_{GN}\ \text{sur}\ (\text{une branche})_{GN}$$

Ensuite, on peut constater qu'il est possible de remplacer le groupe “sur une branche” par d'autres groupes nominaux introduits par une préposition comme “avec un soupir”, “dans une heure” ou “comme une fleur”. On appelle “groupe prépositionnel”, abbrégé en GP, de telles suites. Son identification dans notre phrase initiale amène le nouveau parenthésage suivant (l'étiquette du groupe est attachée à la parenthèse fermante qui le délimite) :

$$(\text{l'oiseau})_{GN}\ \text{pose}\ (\text{ses pattes})_{GN}\ (\text{sur}\ (\text{une branche})_{GN})_{GP}$$

Maintenant, avec quoi regrouper le verbe “pose” ? Depuis Chomsky, les linguistes ont pris l'habitude de considérer que, puisqu'un verbe transitif suivi de son complément d'objet direct (même si nous n'avons pas encore vraiment parlé de cette dernière notion, qui est plutôt sémantique) peut être substitué par un verbe intransitif unique (comme “ronfle”), cela signifie qu'il existe une catégorie “groupe verbal” qui les réunit tous. On note GV un tel groupe. Dans notre exemple, nous constatons même que l'on peut remplacer non seulement “pose ses pattes” par “ronfle” mais aussi “pose ses pattes sur une branche”. Ce dernier groupe est ainsi lui-même un autre groupe verbal. Notre parenthésage se complexifie donc pour donner :

$$(\text{l'oiseau})_{GN}\ ((\text{pose}\ (\text{ses pattes})_{GN})_{GV}\ (\text{sur}\ (\text{une branche})_{GN})_{GP})_{GV}$$

Notre proposition initiale est donc finalement globalement composée d'un groupe nominal suivi d'un groupe verbal, qui eux-mêmes se décomposent en sous-groupes de différentes natures. Pour visualiser cette *structure hiérarchique*, on utilise habituellement une *représentation arborescente*, où figurent tous les noms de catégories intermédiaires. La figure 5.2 l'illustre pour notre exemple.

Cette représentation arborescente est exactement équivalente (mais en plus lisible) au parenthésage étiqueté suivant :

$$((\text{l'oiseau})_{GN}\ ((\text{pose}\ (\text{ses pattes})_{GN})_{GV}\ (\text{sur}\ (\text{une branche})_{GN})_{GP})_{GV})_{S}$$

Les arbres d'analyse syntaxique, dont la figure 5.2 est le premier exemple, ont comme les précédents arbres en quelque sorte “la tête en bas”. Les mots de la phrase y jouent le rôle de feuilles. On étiquette traditionnellement la racine avec le symbole S qui vient de l'anglais “sentence”. Les nœuds internes reçoivent les étiquettes grammaticales intermédiaires. On aurait très bien pu aussi le construire du “haut en bas” en cherchant à découper progressivement la suite des mots plutôt que de “bas en haut” comme présenté ci-dessus, en cherchant progressivement à regrouper ensemble les mots isolés.

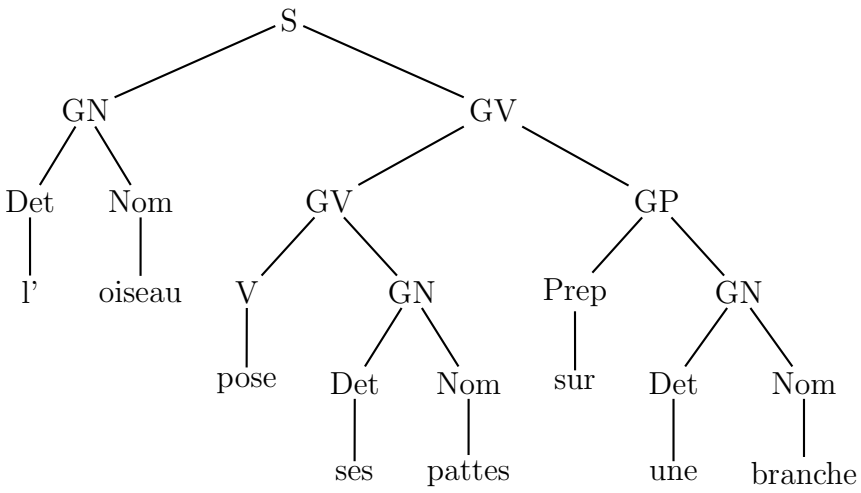


FIG. 5.2 – un premier arbre d'analyse syntaxique

un *syntagme* est donc un groupe de mots qui correspond à un sous-arbre d'un arbre d'analyse syntaxique complet. Par exemple "pose ses pattes" est un syntagme de catégorie GV. D'où le terme "syntagmatique" qui est souvent associé aux grammaires chomskiennes. Le mot anglais "sentence" devrait d'ailleurs plutôt être traduit par "syntagme" que par "phrase" en français.

1.4 Ambiguités

Pour illustrer la difficulté d'associer une structure syntaxique à une proposition, partons d'un autre exemple apparemment très similaire au précédent :

mon	frère	adore	les	pulls	avec	des	rayures
Det	Nom	Vtr	Det	Nom	Prep	Det	Nom

La succession des catégories associées aux mots de cette phrase est exactement la même que celle de la précédente. Pourtant, en cherchant à regrouper ensemble ses mots pour les substituer à d'autres, on en vient à identifier "les pulls avec des rayures" comme un groupe nominal, alors que ce n'était pas possible pour "ses pattes sur une branche". Ainsi, la structure que l'on construit sera cette fois celle de la figure 5.3, où le groupe prépositionnel "avec des rayures" se rattache à une catégorie Nom plutôt qu'à une catégorie GV.

Clairement, pour trouver la "bonne" structure, il est difficile de faire abstraction du sens. Certaines phrases autorisent même plusieurs interprétations différentes possibles correspondant à plusieurs arbres différents : on dit qu'elles sont *ambigües*. C'est le cas de "l'homme observe sa voisine avec des jumelles" qui, de nouveau, est associée à la même succession de catégories que les deux exemples précédents. Cette phrase peut se comprendre de deux manières différentes :

– soit elle signifie que les jumelles sont l'instrument grâce auquel l'homme réalise ses observations : dans ce cas, il faut reprendre la même structure que celle de la figure 5.2 ;

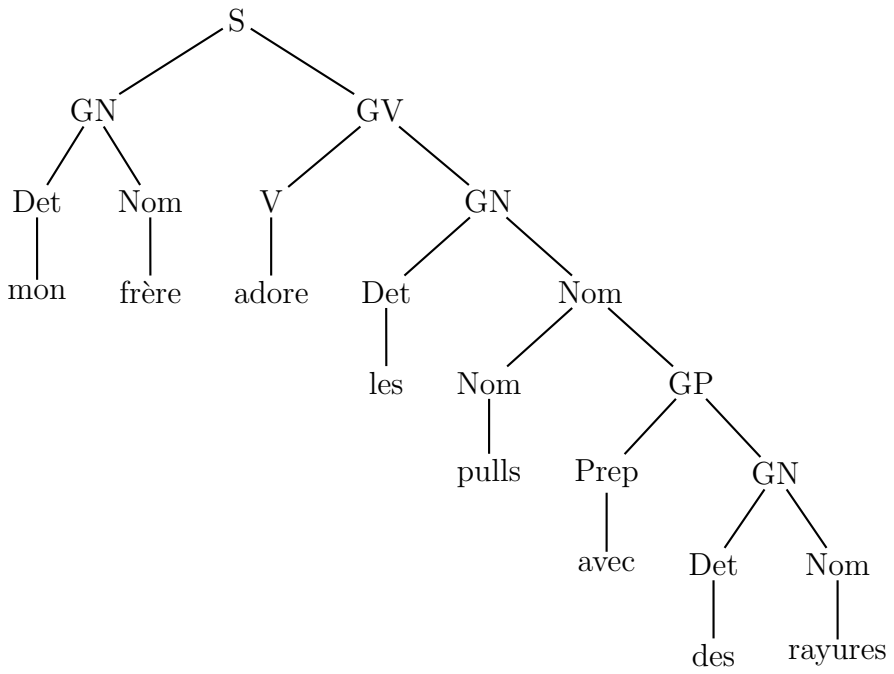


FIG. 5.3 – un autre arbre d'analyse syntaxique

– soit elle signifie que la voisine observée possède des jumelles (quel que soit le sens de ce dernier mot d'ailleurs), auquel cas c'est la figure 5.3 qui convient.

Plusieurs phénomènes syntaxiques classiques peuvent donner lieu à une ambiguité. Dans la phrase précédente, on a un problème de *rattachement prépositionnel*, le groupe GP pouvant s'accrocher soit à un GV (ou un V), soit à un Nom (ou un GN). Parmi les autres phénomènes du même genre, on peut citer :

– problème du *rattachement adjectival* : "(les oiseaux) et (les poissons rouges)" ou "(les oiseaux et les poissons) rouges" ? ; "une roue d'(auto usagée)" ou "(une roue d'auto) usagée" ?..
– problème du *rattachement adverbial* : "il veut (bien parler)" ou "(il veut bien) parler" ?
– problème de coordination : "je veux (du pain et du beurre) ou du fromage" ou "je veux du pain et (du beurre ou du fromage)" ?

Un cas extrême d'ambiguité syntaxique (plusieurs structures possibles pour une même séquence de mots) est fourni par Chomsky lui-même : selon lui, il y a quatre façons différentes d'analyser "Time flies like an arrow", données par les quatres arbres de la figure 5.4.

Ces quatre arbres exploitent l'appartenance de certains mots anglais à plusieurs catégories grammaticales : "time" et "fly" peuvent être un verbe ou un nom, tandis que "like" peut être un verbe ou un adverbe. Ils coïncident avec quatre interprétations différentes de la phrase, qu'on peut respectivement traduire en français comme suit :

– "Le temps vole comme une flèche" (arbre en haut à gauche) : c'est l'interprétation la plus "naturelle" de cette phrase ;
– "Les mouches du temps aiment une flèche" (arbre en haut à droite) : plus

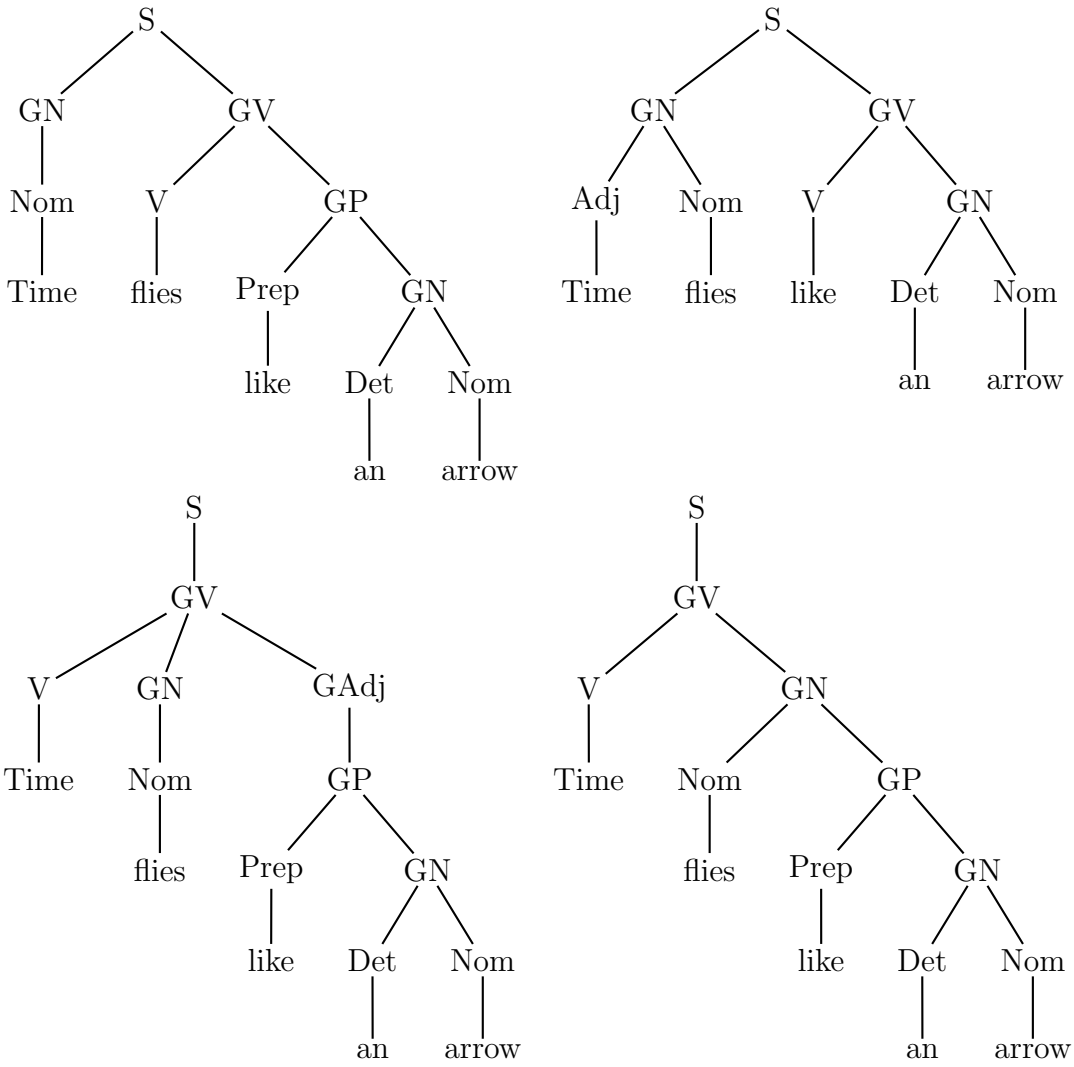


FIG. 5.4 – quatre analyses syntaxiques de la même phrase

exotique...
- "Chronométrez les mouches comme (vous chronométriez) une flèche" (arbre en bas à gauche) : cette interprétation nécessite de supposer une ellipse (omission) du verbe "chronométrer" qui n'est pas répété ;
- "Chronométrez les mouches qui ressemblent à une flèche" (arbre en bas à droite).

1.5 Problèmes avec la structuration arborescente

On l'a déjà vu dans les exemples précédents : pour construire la "bonne structure" associée à une phrase, mieux vaut souvent avoir une bonne idée de ce qu'elle veut dire. On n'a apparemment aucune raison syntaxique de traiter différemment "le chat mange la souris" et "le chat mange la nuit". Pourtant, dans le premier cas, "la souris" est l'objet de l'action, tandis que, dans le deuxième cas, "la nuit" précise le moment de sa réalisation. On peut reconnaître leur différence de statut au fait que certains GN peuvent changer de place et pas d'autres : "la nuit le chat mange" ou même "la nuit mange le chat" préservent le sens initial de la phrase mais pas "la souris mange le chat"... Les relations de type "sujet", "complément d'objet direct", etc. sont en général appelés *rôles sémantiques*. L'identification de ces rôles, sur lesquels nous reviendrons dans le chapitre traitant de sémantique propositionnelle (chapitre 7, section 1.2), se superpose en quelque sorte sur la structure arborescente de la proposition. C'est bien là le signe que cette structure arborescente n'explique pas tout à elle toute seule.

Dans ses écrits les plus récents (notamment depuis la théorie "X-barre" et le programme minimaliste), Chomsky précise la structuration interne générale des syntagmes. Il remarque ainsi que chaque syntagme contient une unité lexicale privilégiée qu'il désigne comme sa "tête" : la tête d'un syntagme nominal est ainsi son nom commun principal, celle d'un syntagme verbal son verbe principal, etc. Les autres composants du syntagme s'organisent autour de sa tête de façon régulière, et ce quelle que soit la nature de ce syntagme. Ainsi, en français, les têtes sont en général précédées d'un spécifieur et suivies de compléments, comme l'illustre la figure 5.5.

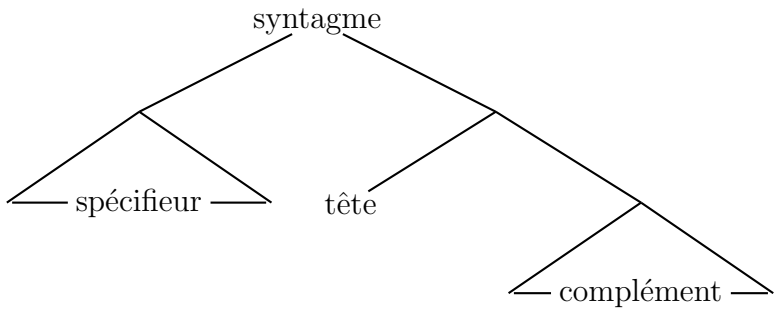


FIG. 5.5 – stucture interne d'un syntagme en français

Cette structure interne aux syntagmes explique certains des "mouvements" possibles qu'ils peuvent subir, comme ceux que l'on vient d'évoquer, ou ceux nécessaires pour transformer une proposition affirmative en une question.

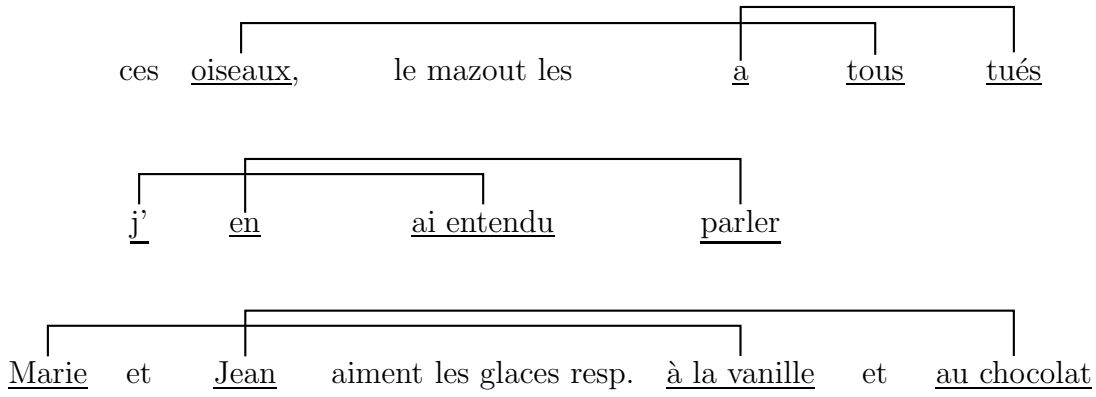


FIG. 5.6 – relations croisées dans des phrases

Les structures arbrorescentes ne semblent pourtant pas toujours suffisantes pour rendre compte de toutes les constructions linguistiques. Citons, parmi les difficultés :

- *l'ellipse*, qui autorise l'effacement de certains constituants pour éviter une répétition, comme dans "elle est d'accord, moi non".
- *l'apposition*, qui est la juxtaposition de syntagmes de même nature ayant le même référent sémantique : "Jean, mon voisin, un bon ami, m'a rendu visite".
- *la thématisation*, où l'ordre des mots permet d'introduire des référents sur lesquels on met successivement l'accent : "moi, mon papa, sa voiture, elle est rouge".

Dans toutes ces situations, c'est le rattachement entre eux des syntagmes qui est délicat. D'autres constructions semblent nécessiter des "branches qui se croisent" : la figure 5.6 en donne quelques exemples. Dans cette figure, on n'a pas explicité les arbres complets des phrases parce que, précisément, leur structuration est problématique. Les raisons qui donnent envie de "raccrocher entre eux" certains mots ou groupes de mots ne sont d'ailleurs pas toujours les mêmes dans ces exemples (rattachement des pronoms, caractère discontinu du passé composé, correspondances sémantiques...). En fait, certaines traditions d'analyse syntaxique se passent même complètement des arbres, et mettent en avant à la place la notion de *dépendances* entre unités lexicales. Une dépendance est une relation orientée entre deux mots. Dans cette tradition, l'analyse grammaticale est donc conçue comme un réseau de dépendances (ressemblant fort aux exemples de la figure 5.6) et non comme un arbre.

Nous nous en tiendrons néanmoins par la suite aux arbres et aux analyses syntaxiques qui les fondent, basées sur la notion de substituabilité entre groupes de mots. Un arbre rend visible la construction interne commune d'un nombre potentiellement infini de phrases différentes : c'est ce qui fait son expressivité.

On dispose maintenant depuis quelques années de *corpus arborés*, c'est-à-dire de textes parenthèsés et étiquetés syntaxiquement, rendant explicite la structure des phrases qu'il contient. Ces données, mises au service des informaticiens et des linguistes qui veulent les exploiter, jouent un grand rôle dans la recherche actuelle.

Notons enfin que la notion d'arbres que nous avons utilisée ici constitue une "structure de données" fondamentale en informatique : par exemple, le système de dossiers (ou répertoires) et de fichiers, qui gère toute l'organisation de la mémoire des ordinateurs, est de nature arborescente. De même, le langage HTML, dans lequel sont écrites toutes les pages du Web, est aussi basé sur une description arborescente du contenu d'une page. Les informaticiens ont donc l'habitude de manipuler de telles données.

2 Modélisation informatique

C'est sans doute dans le domaine de la modélisation de la syntaxe que le plus de travaux ont été produits ces 50 dernières années en TALN. Les recherches ont avancé en parrallèle avec plusieurs autres branches de l'informatique : ainsi, par exemple, les "langages évolués" dans lesquels les informaticiens d'aujourd'hui écrivent leurs programmes (Java, C++, Python, etc. en sont des exemples) nécessitent eux aussi, pour être "compilés", c'est-à-dire traduits en "langage machine" exécutable par les ordinateurs, une phase "d'analyse syntaxique". Les outils à la fois théoriques et pratiques développés dans ce cadre ont ainsi pu être ré-exploités pour traiter les langues naturelles.

Nous allons dans ce qui suit aborder ce que l'on désigne traditionnellement en informatique comme la "théorie des langages", qui traite des grammaires et des langages en général. Même si cette théorie permet aussi de décrire la structure des langues formelles artificielles (comme les langages de programmation), c'est bien sûr surtout son adéquation aux langues humaines qui nous intéressera ici.

2.1 Le retour des automates finis

Une grammaire, on l'a vu à la figure 5.1, doit désormais être conçue comme un dispositif capable de "classer" une suite de mots quelconque en "grammaticale" ou "non grammaticale". Quelle traduction informatique donner à ce dispositif ? La difficulté du problème, déjà évoquée dans le chapitre 2, section 3, réside dans le paradoxe suivant : même en supposant l'ensemble de tous les mots possibles d'une langue comme fini (ce qui, on l'a vu, est réducteur), on peut construire avec cet ensemble fini un nombre potentiellement infini de phrases grammaticales, et un nombre potentiellement infini de "phrases" non grammaticales. Comment les trier avec un dispositif nécessairement fini (puisque traduisible en un programme) ? Nous allons voir que la clé du problème tient dans une notion fondamentale en informatique : la *récursivité*.

Nous avons en fait déjà rencontré dans le chapitre 4, section2.2 un modèle qui peut rendre le service qu'on attend d'une grammaire : c'est celui des automates finis. Imaginons en effet un automate dont le vocabulaire fini V contient l'ensemble de tous les mots possibles du français (flexions et conjugaisons comprises), et dont chaque "chemin" correspondrait à une phrase syntaxiquement correcte. Confronté à une suite de mots quelconque, l'automate n'aurait qu'à vérifier si cette suite correspond à un de ses chemins pour savoir si elle est grammaticale.

L'exemple de la figure 4.4 montre que même un automate très simple peut "reconnaître" une infinité de chemins différents possibles. La récursivité, chez lui, réside dans les transitions qui "bouclent", parce que leur état d'arrivée est identique à leur état de départ (il y en a 2 dans l'exemple). Par définition, il est possible d'utiliser un nombre quelconque de fois ces transitions, et donc de juger grammaticales un nombre infini de chaînes possibles.

La "productivité infinie" des langues naturelles peut-elle se ramener à un phénomène du même genre ? Certaines constructions répondent effectivement à ce schéma, comme celle de la figure 5.7, qui reconnaît le langage exprimé par l'expression rationnelle : "la.Ferrari.passa.(très)*.vite".

Cette configuration un peu marginale n'épuise évidemment pas toutes les sources d'"infini" possibles autorisées par la langue française. Mais il n'y a aucune raison non plus pour que les "boucles" présentes dans les automates finis se limitent à une seule transition. On peut très bien imaginer des *suites de transitions successives* permettant de retourner dans un état précédemment quitté. Ce type de récursivité est dit *indirect*, alors que les automates des figures 4.4 et 5.7 ne contiennent que de la *récursivité directe*.

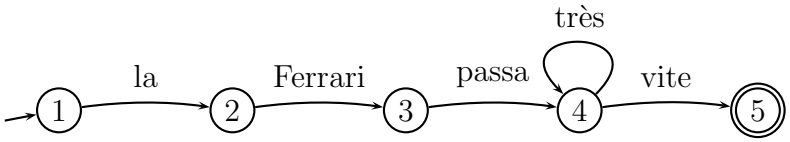


FIG. 5.7 – automate reconnaissant une petite (mais infinie !) portion du français

Un autre exemple d'une portion infinie (mais moins triviale que précédemment) du français pouvant être modélisée par un automate fini est donné en figure 5.8 : ce dernier permet cette fois de désigner un membre quelconque de la famille d'un certain individu (ici : Jean), et de dire qu'il (ou elle) dort.

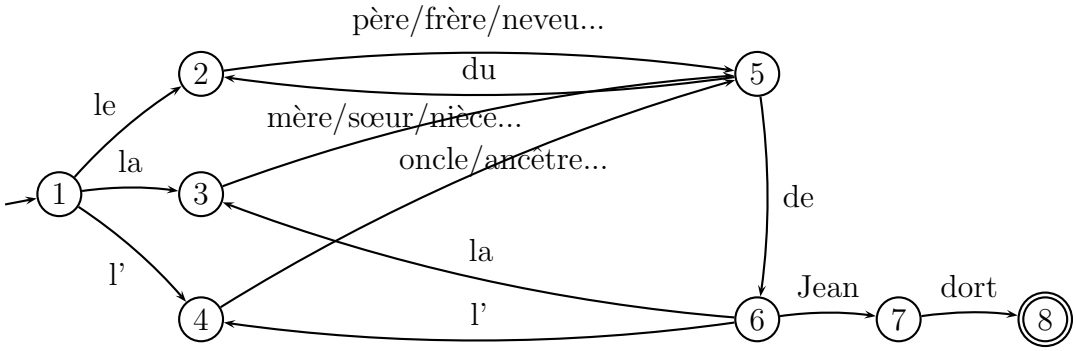


FIG. 5.8 – automate fini reconnaissant un ensemble infini de phrases

Dans ce nouvel automate, comme dans celui de la figure 4.3, les transitions étiquetées par plusieurs mots séparés par un "/" synthétisent en fait autant de transitions différentes que de mots cités. Les "..." signalent qu'il faudrait ajouter quelques

unités lexicales dans le vocabulaire pour être vraiment exhaustif. Ce dispositif reconnaît des phrases comme “la mère du frère de l'oncle du père de Jean dort”, et une infinité d'autres ! Il contient plusieurs cas de récursivité indirecte ; par exemple, il existe des transitions allant de l'état 2 à l'état 5, et une autre allant de l'état 5 à l'état 2 : en enchaînant deux transitions, on peut donc partir de l'état 2 et y revenir, et ceci un nombre quelconque de fois.

Un dernier exemple de portion du français représentable par un automate : l'ensemble des phrases produites par le “pipotron”, un générateur de phrases aléatoires qui pratique avec aisance la langue de bois (cf. http ://www.pipotron.com/, par exemple). Le principe du pipotron, c'est une succession de choix indépendants de mots ou groupe de mots dans une série de listes. Tous les choix possibles construisent une phrase grammaticale. Quelques-uns d'entre eux sont reproduits dans l'automate de la figure 5.9 ; il suffirait d'y ajouter quelques transitions pour les reproduire en intégralité.

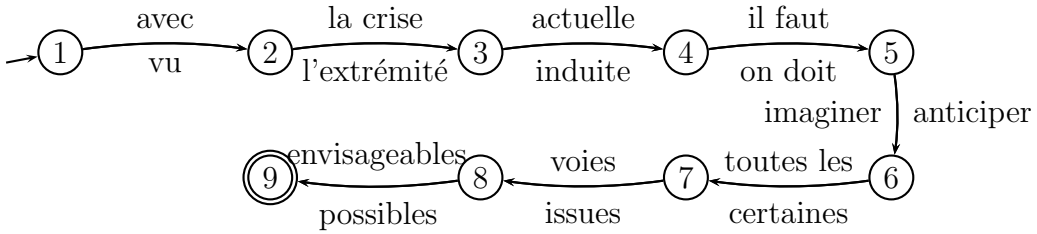


FIG. 5.9 – automate produisant quelques phrases du “pipotron”

Les “Cent mille milliards de poèmes” de Raymond Queneau est un recueil de poèmes conçu suivant le même principe. Le livre se présente comme une série de “languettes” qui prennent la place des 14 lignes d'un sonnet. A la place de chaque ligne, il y a le choix entre 10 languettes différentes, chacune contenant un alexandrin de même rime. Chaque choix est indépendant et toutes les combinaisons possibles d'alexandrins forment finalement un sonnet complet. Les 10 choix possibles de chaque ligne ouvrent autant de possibilités à la ligne suivante, ce qui fait au bout du compte $10 * 10 * ... * 10 = 10^{14}$ poèmes différents... Mais, ce nombre a beau être impressionnant, il n'atteint malgré tout pas la combinaison infinie de choix permise par certains des exemples précédents. Comme le pipotron, le livre de Queneau ne spécifie en fait qu'un langage fini : les automates correspondant à ces jeux combinatoires ne permettent aucun retour en arrière : pas de boucle, pas de récursivité...

2.2 Limites des automates finis

La grammaire de la langue française peut-elle donc se représenter sous la forme d'un énorme automate fini où figurerait de la récursivité ? Il semble bien que non, mais cela mérite argumentation. Nous allons présenter successivement trois des principaux arguments cités à l'encontre de cette hypothèse. Ce sont un peu trois façons différentes de dire la même chose, mais en mettant l'accent sur une facette ou sur une autre de l'analyse linguistique.

L'argument sans doute le plus facile à comprendre est dû à Pinker, psycholinguiste canadien, et relève surtout de la psychologie cognitive. Il consiste à constater

que stocker en mémoire ce qui constitue la "compétence" des locuteurs d'une langue sous la forme d'un unique automate ne serait ni économique ni efficace. En effet : en français, comme en anglais et dans beaucoup d'autres langues, on utilise la même construction pour les "groupes nomimaux" (GN dans nos arbres), que ceux-ci jouent le rôle de sujet, d'objet, ou de circonstant (pour préciser le temps, le lieu, la manière, etc.) de la proposition dont ils font partie. S'il fallait bâtir un "automate complet du français", il faudrait donc répéter la portion d'automate qui décrit la construction des groupes nominaux à plusieurs endroits dans cet automate global : au début pour les GN sujets, après le verbe pour les GN objets directs, etc. Ce n'est clairement pas comme cela que la mémoire humaine fonctionne. On a d'ailleurs pu observer que lorsqu'un enfant a entendu une seule fois un nom (ou toute autre unité lexicale, ou toute portion de structure) dans une certaine position grammaticale, il est capable de le ré-employer spontanénemnt et instantanément dans une autre position grammaticale.

Certes, cet argument est plus intuitif que vraiment rédhibitoire : rien n'interdit aux informaticiens d'utiliser pour modéliser la compétence d'autres outils que ceux employés par l'esprit humain. Mais il suggère surtout qu'il doit y avoir d'autres moyens que les automates finis pour reproduire les jugements de grammaticalité : que peut-être un *ensemble d'automates spécialisés s'appelant les uns les autres* incluant, notamment, un seul automate pour décrire les GN, mais qui pourrait resservir à plusieurs occasions dans l'analyse d'une phrase, serait plus adapté à nos besoins qu'un unique gros automate. C'est exactement ce qui sera l'objet de la section suivante.

Le deuxième argument est très proche du premier : il met en avant le fait que les automates sont incapables de produire les *structures arborescentes* avec lesquelles nous avons analysé les phrases des sections 1.3 et 1.4. En fait, ces structures sont le résultat direct de notre critère de susbtituabilité : s'il est possible d'étiqueter plusieurs portions d'arbres (ou sous-arbres) par GN, c'est justement parce que ces portions sont interchangeables les unes avec les autres, si l'on s'en tient à juger la grammaticalité de la phrase ainsi produite. A vrai dire, pour vraiment prendre la mesure de cet argument, il faudra attendre la section sur les "grammaires formelles" à suivre : de fait, nous y verrons que les automates peuvent produire des sortes d'arbres, mais d'une espèce un peu malingre et chétive qui ne fait pas honneur à la richesse de la syntaxe.

Cette nouvelle formulation n'est pas plus décisive que la précédente. Encore une fois, rien n'oblige les informaticiens à reproduire dans leurs programmes ce que fait un humain : les analyses syntaxiques sous forme d'arbres que nous avons détaillées étaient en quelque sorte de la "cuisine interne de linguiste". On peut imaginer qu'il existe un dispositif qui n'aurait pas à passer par cet artefact intermédiaire pour rendre son verdict de grammaticalité.

Le dernier argument, enfin, et le plus souvent cité, remonte aux toutes premières intuitions de Chomsky ; il s'appuie, dans sa formulation contemporaine, sur un théorème mathématique. Le théorème énonce qu'il est impossible d'engendrer avec un automate fini certains langages comme le langage $L = \{a^n b^n | n \geq 1\}$ (construit sur le vocabulaire $V = \{a, b\}$), c'est-à-dire le langage qui réunit toutes les suites constitées d'un nombre quelconque (non nul) de symboles a suivi *du même nombre* de symboles b. Ce langage comprend les chaînes : ab, $aabb$, $aaabbb$, etc. Pourquoi

un automate fini ne peut-il reconnaître ce langage ? La démonstration est un peu technique mais son principe en est simple. Supposons qu'un tel automate existe, appelons-le A et cherchons à en déduire une contradiction. Le raisonnement suit les étapes suivantes :

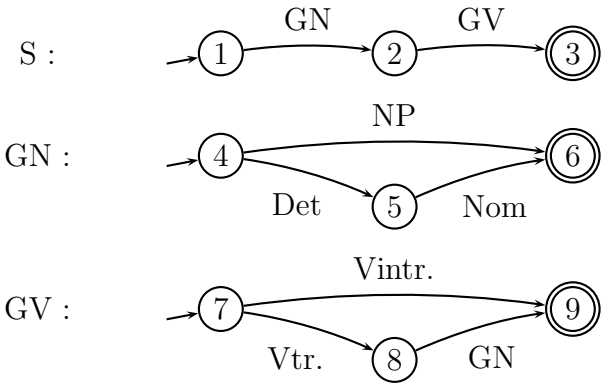
- le langage L étant évidemment infini, A doit inclure au moins une boucle récursive (directe ou indirecte).
- choisissons parmi le langage engendré par A la suite de symboles correspondant à un chemin dans A qui emprunte *exactement une fois* cette boucle. Soit w cette suite de symboles. Par définition, on peut la décomposer en 3 morceaux : le morceau correspondant au chemin parcouru "de l'état initial jusqu'au début de la boucle", celui correspondant au chemin "dans la boucle" et celui du chemin "après la boucle jusqu'à un état final" (ce dernier et le premier pouvant éventuellement être vides) : $w = u_1u_2u_3$ où $u_2 \neq \epsilon$ est la suite des symboles dans la boucle.
- tous les chemins qui commencent et qui se terminent comme le précédent mais qui, au lieu d'emprunter une seule fois la boucle, l'empruntent un nombre quelconque de fois sont des chemins corrects de l'automate : donc toutes les suites de symboles qui appartiennent à l'ensemble $u_1u_2^*u_3$ font partie du langage de A.
- Maintenant, essayez de choisir parmi les suites de la forme a^nb^n celle qui pourra jouer le rôle de w, et en particulier dans cette suite la portion u_2 qui pourra être répétée tout en restant dans le langage L : c'est impossible...

Ce théorème, aussi appelé "lemme de pompage" (parce qu'on y "gonfle" artificiellement, en répétant une de ses parties, une suite de symboles) a le mérite de donner une limite théorique rigoureuse à l'expressivité des automates finis. En quoi concerne-t-il le langage naturel ? C'est la clé du problème. Pour nous en convaincre, Chomsky exhibe plusieurs constructions qui suivent la forme a^nb^n : les plus célèbres sont les "propositions relatives enchassées". Pour l'illustrer, partons de la phrase "l'ours dort". On peut insérer après le nom sujet "ours" un complément pour obtenir : "l'ours (que l'homme a vu) dort". Mais il n'y a aucune raison pour s'arrêter en si bon chemin. Le complément précédent peut très bien être de nouveau inséré pour modifier le nom "homme" juste introduit : "l'ours (que l'homme (que l'homme a vu) a vu) dort". Et ainsi de suite. Si on appelle a la suite de mots "que l'homme", et b "a vu", on s'est bien ramené à une construction du type "l'ours a^nb^n dort", où n'importe quelle valeur de $n \geq 1$ donne une phrase syntaxiquement correcte. Evidemment, au delà de $n = 2$, la phrase devient très difficilement compréhensible. Mais, pour Chomsky, cette difficulté est un problème de *performance*, lié aux limites de la mémoire humaine, et ne remet pas en cause la *compétence* des locuteurs à juger la phrase grammaticale. On voit maintenant l'importance cruciale de cette distinction dans son argumentation.

Tous ces arguments n'empêchent pas les linguistes et les informaticiens de modéliser sous forme d'automates ou d'expressions régulières certains des phénomènes qu'ils étudient. Mais ils savent aussi (ou devraient savoir !) que ce modèle a des limites, et que son usage pour formaliser la syntaxe est l'objet de critiques fondamentales fortes. Ils font donc pour cela appel à d'autres formalismes plus puissants...

2.3 Réseaux de Transitions Récursifs

Les Réseaux de Transition Récursifs (ou RTRs) sont une généralisation des automates finis qui répondent exactement aux arguments de la section précédente. La figure 5.10 en donne un premier exemple simple.



NP : {Jean, Marie, ...}
Det : {le, la, un, une...}
Nom : {chat, souris...}
Vintr. : {dort, ronfle...}
Vtr. : {mange, aime...}

FIG. 5.10 – un Réseau de Transition Récursif pour le français

Un RTR se présente comme un *ensemble d'automates* qui ont les propriétés suivantes :

- chaque automate de l'ensemble a au moins un état initial et au moins un état final (comme les automates finis) ;
- chaque automate est associé à une étiquette, marquée à sa gauche : on les appelle aussi les *symboles non terminaux* du RTR, parce qu'ils servent de vocabulaire intermédiaire, et ne se retrouveront pas dans les suites d'unités lexicales dont on teste la grammaticalité. Dans les exemples linguistiques, ces symboles seront des catégories grammaticales. Parmi elles, figure le symbole S désignant la catégorie des propositions syntaxiquement correctes ;
- les transitions des automates sont étiquetées soit avec des symboles non terminaux associés à un automate, soit avec des symboles "terminaux", qui sont simplement les unités lexicales.

Dans notre figure, les mots avec lesquels on va reconnaître ou engendrer des phrases figurent à la fin dans de simples listes, alors que les automates qui les précèdent n'ont que des transitions étiquetés avec des catégories. Il faut voir ces listes finales comme des automates élémentaires contenant simplement un état initial, un état final et autant de transitions de l'un à l'autre que de mots dans la liste. Dans cet exemple, on distingue les verbes transitifs étiquetés Vtr, des verbes intransitifs, associés à Vtr.

Quel est le critère de grammaticalité associé à ce nouvel objet ? Il est très similaire à celui des automates finis : une suite de mots est reconnue si elle correspond à un chemin *dans l'automate étiqueté par S* du RTR. Comme précédemment, les transitions étiquetées par un mot peuvent être franchies par la reconnaissance de ce mot. La nouveauté est que, pour franchir une transition qui porte un symbole non terminal, il est nécessaire de parcourir un chemin complet *dans l'automate associé à ce symbole*. Les automates du RTR ont donc la possibilité de s'appeller les uns les autres, à la manière des fonctions ou des procédures dans les langages de programmation impératifs. Par exemple, expliquons comment "le chat mange la souris" peut être jugé grammatical dans ce dispositif :

- "le" est de catégorie Det et "chat" de catégorie Nom, donc "le chat" est un chemin dans l'automate GN (passant par les états 4, 5 et 6). Ce chemin autorise à son tour à passer de l'état 1 à l'état 2 dans l'automate S.
- De même que précédemment, "la souris" est un chemin dans GN. Par ailleurs, "mange" étant de catégorie $Vintr$,"mange la souris" est un chemin dans l'automate GV (passant par les états 7, 8 et 9). Donc "mange la souris" permet, dans l'automate S, de passer de l'état 2 à 3. La phrase est donc acceptée par l'automate associé à S et donc par le RTR dans son ensemble.

Ce dispositif répond exactement à l'argument de Pinker, puisque les GN en position sujet y sont traités par le *même automate* que ceux en position d'objet. Si on ajoute un élément à la liste des noms, il pourra donc être utilisé indifféremment dans l'une ou l'autre de ces positions.

De même, les RTRs rendent parfaitement compte des structures arborescentes. On pourrait ainsi paraphraser l'analyse précédente de "le chat mange la souris" sous la forme de l'arbre de la figure 5.11. Quand, par exemple, pour trouver un chemin dans l'automate associé au symbole GN, il faut franchir deux transitions successives portant les symboles Det et Nom respectivement, cela correspond dans cette figure à un sous-arbre de racine GN qui a deux fils : le premier est Det, le second Nom (et ceci à deux endroits différents dans l'arbre, puisque l'automate GN est appelé lui aussi deux fois).

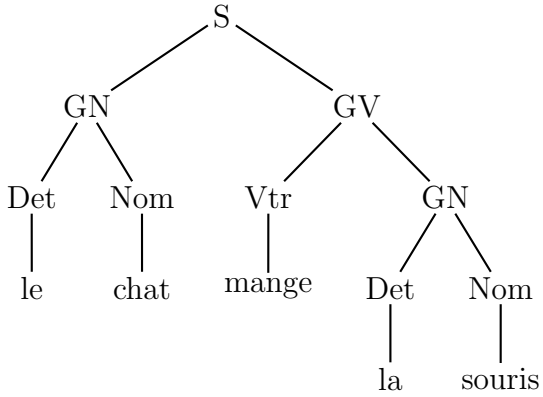


FIG. 5.11 – un arbre d'analyse syntaxique

Enfin, le langage $L = a^n b^n$, impossible à générer avec un automate fini, peut-être reconnu par un RTR très simple : celui de la figure 5.12. Dans ce RTR, l'unique

automate a la propriété de pouvoir "s'appeler lui-même" via la transition portant sa propre étiquette (c'est à cause d'elle qu'il n'est pas un automate fini). Chaque appel récursif ajoute un symbole a à gauche et un symbole b à droite, tout en maintenant "au milieu" la possibilité de recommencer un chemin.

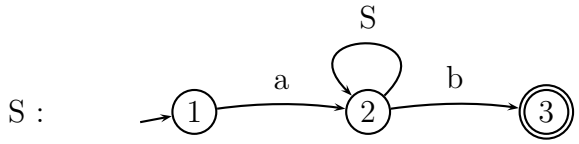


FIG. 5.12 – un RTR reconnaissant $a^n b^n$

Cela achève de montrer que les RTRs sont fondamentalement plus puissants que les automates finis. Sont-ils l'outil idéal ? Ils ont effectivement été utilisés dans les années 70, mais ne sont plus vraiment à la mode. Ils constituent pourtant la base de logiciels très populaires comme Unitex, cité en section 2.4 du chapitre 4. Il est temps maintenant d'introduire l'outil de référence qui nous aidera à y voir plus clair dans la structure des langages, et à mesurer à cet aune la puissance des modèles qui les formalisent.

2.4 Grammaires formelles

Les grammaires formelles sont au cœur de la "théorie des langages" des informaticiens. Même si les grammaires définies par cette théorie sont rarement utilisées telles quelles en linguistique, le cadre dans lequel elles s'inscrivent (en particulier la "hiérarchie de Chomsky" dont il sera question plus loin) est un référent incontournable de tout formalisme syntaxique. Nous verrons notamment que les notions d'automates finis et de RTRs en sont des cas particuliers.

On définit précisément une grammaire formelle G comme un quadruplet d'éléments : $G = \langle V, N, P, S \rangle$ où :

- V est le *vocabulaire terminal* de G : pour une application linguistique, V coïncidera bien sûr avec l'ensemble fini des mots pouvant figurer dans les suites dont on veut tester la grammaticalité. On écrira toujours les éléments de V avec des minuscules latines.
- N est le *vocabulaire non terminal* de G : comme dans les RTRs, ce vocabulaire (lui aussi fini) servira lors d'étapes intermédiaires de calculs. Il est qualifié de "non terminal" car aucun de ses symboles ne doit se retrouver dans les productions finales reconnues ou engendrées par la grammaire. Il contient nécessairement, entre autres, le symbole S, qui identifie les suites grammaticales (4ème élément du quadruplet) : $S \in N$. Traditionnellement, les symboles non terminaux sont écrits en majuscules latines.
- Le dernier élément, P, est un ensemble fini de "règles de production" ou "règle de ré-écriture". Chaque règle de P est de la forme : $\alpha \longrightarrow \beta$ où $\alpha \in (V \cup N)^+$ et $\beta \in (V \cup N)^*$. Ainsi, α et β sont des suites de symboles pris parmi les éléments de V et de N, avec la seule différence que β peut être une liste vide

($\beta = \epsilon$ est autorisé) mais pas α. Une telle règle doit être comprise comme : "α peut-être remplacé par β".

Etant donnée une grammaire $G = \langle V, N, P, S\rangle$ et une suite de symboles quelconque $u \in (V \cup N)^+$, on dit que G permet de dériver v à partir de u en une seule étape, et on note $u \longrightarrow v$ si les conditions suivantes sont réunies :

- u peut se décomposer en trois morceaux : $u = x\alpha y$ avec $\alpha \neq \epsilon$
- la règle : $\alpha \longrightarrow \beta \in P$
- $v = x\beta y$.

Cela signifie que si une règle de la grammaire précise que "α peut-être remplacé par β", cette règle peut s'appliquer à l'intérieur de n'importe quelle suite de symboles qui contient α. Evidemment, on peut appliquer successivement autant de règles que l'on souhaite à une suite de symboles. On notera $u \longrightarrow^* v$ une dérivation en plusieurs étapes successives.

On appelle *langage engendré (ou reconnu) par une grammaire* $G = \langle V, N, P, S\rangle$ et on note $L(G)$ l'ensemble des suites de symboles terminaux que l'on peut obtenir par dérivations successives en partant de l'unique symbole S. On a ainsi :
$L(G) = \{w \in V^* | S \longrightarrow^* w\}$.

Donnons tout de suite un exemple simple où ces définitions s'appliquent. Soit la grammaire $G = \langle V, N, P, S\rangle$ définie par :

- $V = \{le,\ la,\ chat,\ souris,\ dort,\ mange\}$
- $N = \{S,\ GN,\ GV,\ Det,\ Nom,\ Vtr,\ Vintr\}$
- $P = \{S \longrightarrow_1 GN\ GV,\ \ GN \longrightarrow_2 Det\ Nom,\ \ GV \longrightarrow_3 Vintr,$
  $GV \longrightarrow_4 Vtr\ GN,\ \ Det \longrightarrow_5 le,\ \ Det \longrightarrow_6 la,\ \ Nom \longrightarrow_7 chat,$
  $Nom \longrightarrow_8 souris,\ \ Vintr \longrightarrow_9 dort,\ \ Vtr \longrightarrow_{10} mange\}$

Montrons la séquence des dérivations successives qui justifie que "le chat mange la souris" fait partie du langage engendré par G. Pour rendre les choses encore plus claires, on a numéroté les règles et on souligne à chaque étape la portion de la suite qui est ré-écrite par la règle en question :
$\underline{S} \longrightarrow_1 GN\ GV$
$\underline{GN}\ GV \longrightarrow_2 Det\ Nom\ GV$
$\underline{Det}\ Nom\ GV \longrightarrow_5 le\ Nom\ GV$
$le\ \underline{Nom}\ GV \longrightarrow_7 le\ chat\ GV$
$le\ chat\ \underline{GV} \longrightarrow_4 le\ chat\ Vtr\ GN$
$le\ chat\ \underline{Vtr}\ GN \longrightarrow_{10} le\ chat\ mange\ GN$
$le\ chat\ mange\ \underline{GN} \longrightarrow_2 le\ chat\ mange\ Det\ Nom$
$le\ chat\ mange\ \underline{Det}\ Nom \longrightarrow_6 le\ chat\ mange\ la\ Nom$
$le\ chat\ mange\ la\ \underline{Nom} \longrightarrow_8 le\ chat\ mange\ la\ souris$
On a donc bien : $S \longrightarrow^* le\ chat\ mange\ la\ souris$.

L'ordre d'application des règles étant arbitraire, il existe d'autres séquences de dérivations qui produisent le même résultat. Elles correspondent aux différentes étapes de la construction de l'arbre de la figure 5.11. D'autres suites grammaticalement correctes peuvent être obtenues par cette grammaire comme : "la souris dort", ou "le chat mange le chat". Comme aucune contrainte d'accords en genre n'a été prise en compte dans ces règles, on peut aussi produire "la chat mange le souris". Pour remédier à ce problème, il faudrait introduire des symboles non terminaux distincts pour les catégories Det et Nom, suivant qu'ils sont masculins ou féminins, et

adapter la règle 2 pour n'autoriser que les associations entre Det et Nom du même genre.

Cette présentation semble privilégier la capacité de *synthèse* des grammaires formelles. Elles peuvent en fait tout aussi bien servir en *analyse* : une phrase sera *reconnue* comme syntaxiquement correcte si, à partir de ses mots et en "remontant le sens" des règles de ré-écriture, on peut arriver au symbole S.

2.5 Transformation des automates et des RTRs en grammaires

N'importe quel automate ou RTR peut être transformé en une grammaire équivalente, c'est-à-dire *reconnaissant exactement le même langage*. Pour illustrer ce théorème mathématique (que nous ne chercherons pas à démontrer ici), commençons avec l'automate de la figure 5.7, qui reconnaît le langage $L = la.Ferrari.passa.tres^*.vite$. La grammaire $G = \langle V, N, P, S\rangle$ qui lui correspond est définie par :

- $V = \{la,\ Ferrari,\ passa,\ tres,\ vite\}$
- $N = \{S = Q_1,\ Q_2,\ Q_3,\ Q_4\}$: les états de l'automate correspondent aux symboles non terminaux de la grammaire (la lettre Q est traditionnellement utilisée pour nommer ces symboles). L'état initial joue le rôle de S, tandis que l'état final, qui sert ici uniquement de point d'arrivée pour une transition, n'a pas besoin, lui, de donner lieu à un symbole spécifique.
- $P = \{S \longrightarrow la\ Q_2,\ \ Q_2 \longrightarrow Ferrari\ Q_3,\ \ Q_3 \longrightarrow passa\ Q_4,\ \ Q_4 \longrightarrow tres\ Q_4,\ Q_4 \longrightarrow vite\}$

Montrons par exemple comment cette grammaire génère la phrase "la Ferrari passa très très vite" :
$S \longrightarrow la\ Q_2$
$la\ Q_2 \longrightarrow la\ Ferrari\ Q_3$
$la\ Ferrari\ Q_3 \longrightarrow la\ Ferrari\ passa\ Q_4$
$la\ Ferrari\ passa\ Q_4 \longrightarrow la\ Ferrari\ passa\ tres\ Q_4$
$la\ Ferrari\ passa\ tres\ Q_4 \longrightarrow la\ Ferrari\ passa\ tres\ tres\ Q_4$
$la\ Ferrari\ passa\ tres\ Q_4 \longrightarrow la\ Ferrari\ passa\ tres\ tres\ vite$
L'arbre qui rend compte de cette suite de dérivations est donné en figure 5.13

La boucle sur le mot "très" dans l'automate 5.7 était une source de "récursivité directe". On retrouve cette propriété au niveau des règles de ré-écriture de notre grammaire : en effet, la règle $Q_4 \longrightarrow$ très Q_4, qui traduit cette boucle, a la propriété de *produire le symbole non terminal Q_4 dont elle part*. On peut donc l'utiliser un nombre quelconque de fois, chaque utilisation entraînant l'ajout d'une occurrence du mot "très" dans la phrase.

Par ailleurs on voit que, grâce à cette grammaire, on peut associer un arbre à chaque phrase du langage L. Mais les arbres obtenus ont une forme particulière : à l'image de celui de la figure 5.13, ils ne se développent toujours que "vers la droite". Pour des raisons naturelles, on appelle ce type d'arbre des "peignes". Les peignes ainsi obtenus ne coïncident pas nécessairement avec les structures identifiées par une analyse linguistique.

De manière générale, on suppose disposer d'un automate fini avec un unique état initial et un nombre quelconque d'états finaux (on peut toujours se ramener à

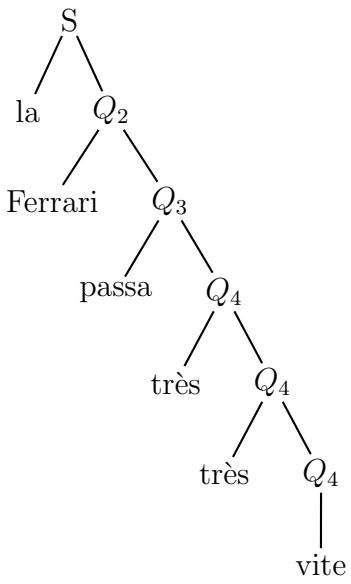


FIG. 5.13 – arbre produit par une grammaire issue d'un automate

un tel automate). Voici la méthode à employer pour transformer cet automate fini quelconque en une grammaire formelle :

- le vocabulaire terminal V de la grammaire est identique au vocabulaire de l'automate ;
- définir autant de symboles non terminaux Q_n qu'il y a d'états non terminaux n dans l'automate : l'état initial correspond au symbole S ($Q_0 = S$) ;
- pour tout état terminal m de l'automate à partir duquel part une transition, introduire un nouveau symbole Q_m non terminal dans la grammaire ;
- pour toute transition étiquetée par un symbole terminal quelconque $a \in V$ partant d'un état n et aboutisant à un état m dans l'automate :
  - si m est un état terminal, ajouter la règle : $Q_n \longrightarrow a$
  - si m n'est pas un état terminal, ou bien si m est terminal mais il existe une transition qui en part, alors ajouter la règle : $Q_n \longrightarrow a\ Q_m$

Les cas de récursivité indirectes seront eux aussi "lisibles" à partir des règles de la grammaire : par exemple, en traduisant l'automate de la figure 5.8, on va obtenir, entre autres, les deux règles suivantes :
$Q_2 \longrightarrow$ père Q_5
$Q_5 \longrightarrow$ du Q_2
En enchaînant ces deux règles, on peut encore produire le symbole dont on part : $Q_2 \longrightarrow^*$ père du Q_2. De manière générale, chaque fois que, dans une grammaire, il existe un symbole non terminal Q tel qu'avec une succession de règles de P on obtient :
... Q ... $\longrightarrow^*$ ... Q ...
alors cette grammaire est récursive et reconnaît une infinité de phrases.

On peut appliquer exactement le même traitement aux RTRs, pour les transformer en grammaires. Dans ce cas, la catégorie associée à un automate sert à étiqueter son état initial. Le RTR de la figure 5.12 peut, lui, être transformé encore plus simplement en la grammaire $G = \langle V, N, P, S \rangle$ définie par :

– $V = \{a, b\}$
– $N = \{S\}$
– $P = \{S \longrightarrow ab, S \longrightarrow aSb\}$

Cette grammaire produit par exemple la phrase “aaabbb” en lui associant l’arbre de la figure 5.14

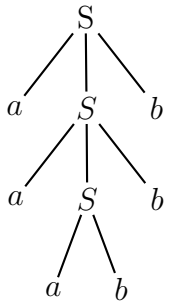


FIG. 5.14 – arbre associé à “aaabbb”

2.6 Hiérarchie de Chomsky

Les sections précédentes montrent que la notion de grammaire formelle est plus générale que celle d’automate fini ou de RTRs, puisqu’on peut ramener chacun de ces deux formalismes à une grammaire. Mais nous avons aussi vu que les RTRs sont strictement plus expressifs que les automates finis, au sens où ils peuvent générer des langages qu’aucun automate ne peut produire. Aussi existe-t-il dans la famille des grammaires formelles plusieurs “classes” qui se distinguent, précisément, par leur “expressivité”. La *hiérarchie de Chomsky* explicite la nature et les propriétés de ces classes. Combien en distingue-t-on et comment savoir à laquelle appartient une grammaire donnée ? Tout est dans la forme des règles de l’ensemble P. Soit une grammaire $G = \langle V, N, P, S \rangle$:

– si toutes les règles de P sont de la forme : $A \longrightarrow a$ ou $A \longrightarrow a\ B$ (ou bien si elle sont toutes de la forme : $A \longrightarrow a$ ou $A \longrightarrow B\ a$) avec $A, B \in N$ et $a \in V$, alors on dit que G est une grammaire rationnelle ou régulière, ou encore que G est de *type 3*. Les grammaires qui proviennent de la transformation d’un automate, comme expliqué dans la section précédente, sont de ce type et inversement, toute grammaire de ce type peut être transformée en un automate.
– si toutes les règles de P ont une partie gauche réduite à un seul symbole non terminal, c’est-à-dire sont de la forme : $A \longrightarrow ...$ avec $A \in N$ et n’importe quelle suite de $(V \cup N)^*$ à droite de la flèche, alors on dit que G est une grammaire hors-contexte (traduction de l’anglais “context-free”) ou algébrique, ou encore que G est de *type 2*. Les grammaires qui proviennent de la transformation d’un RTR sont de ce type et inversement, toute grammaire de ce type peut être transformée en un RTR.
– si toutes les règles de P sont telles que le nombre total de symboles de V ou de N à gauche de la flèche est toujours inférieur ou égal au nombre total de symboles à droite de la flèche, on dit que G est une grammaire sensible au

contexte (traduction de l'anglais "context-sensitive") ou encore que G est de *type 1*.
- toutes les grammaires formelles, quelle que soit la forme de leurs règles, sont de *type 0*.

Il y a donc 4 grandes familles de grammaires, emboîtées les unes dans les autres. Il est en effet facile de se convaincre que les critères qui caractérisent chacune de ces familles sont de moins en moins restrictifs : toute grammaire qui vérifie celui d'une certaine classe vérifie aussi nécessairement ceux des classes de type inférieur. On peut ainsi visualiser les ensembles de grammaires d'un type donné par la figure 5.15.

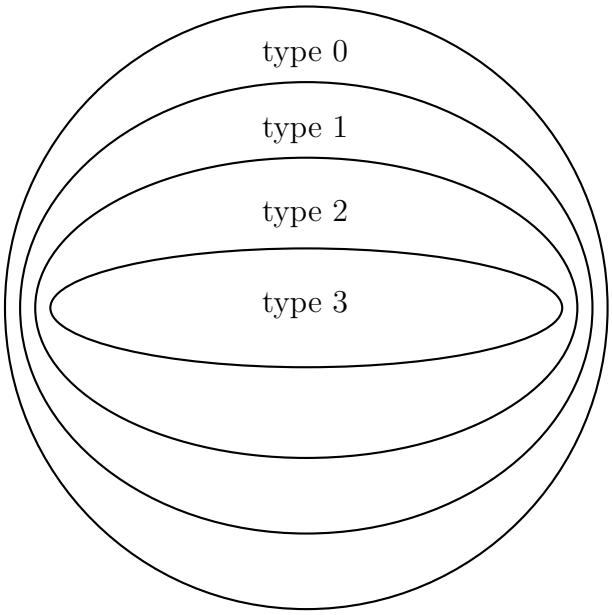


FIG. 5.15 – les classes de grammaires de la hiérarchie de Chomsky

Cette hiérarchie sur les grammaires permet aussi, bien sûr, de classer *les langages*. Un langage L sur un vocabulaire V peut toujours être considéré comme une partie de V^* (la partie des suites d'unités syntaxiquement correctes). Un langage L est de type n s'il existe une grammaire G de type n telle que $L(G) = L$ et si aucune grammaire de type $m > n$ ne satisfait cette propriété. Par exemple, d'après ce que nous avons vu précédemment, on est assuré que le langage $L = a^n b^n$ est de type 2 (ou encore : "est un langage algébrique").

Certains des noms de ces classes proviennent de propriétés mathématiques que nous n'évoquerons pas. Mais, pour les grammaires de type 2, l'explication est simple : elles sont appelées "hors-contextes" parce que chacune de leurs règles de ré-écriture spécifient comment remplacer un symbole non terminal *indépendamment du contexte dans lequel il se trouve*, c'est-à-dire des symboles -terminaux ou non terminaux- qui sont ses voisins dans la chaîne de ré-écriture. Cette propriété nous assure que les structures produites par ces grammaires prennent toujours la forme d'arbres.

Au contraire, une grammaire de type 1 peut contenir des règles *contextuelles*. Prenons l'exemple de la grammaire de type 1 définie par les ensembles $V = \{a, b, c\}$, $N = \{S, A, B\}$ et les règles : $S \longrightarrow aBSc$, $S \longrightarrow aBc$, $Ba \longrightarrow aB$, $Bc \longrightarrow bc$ et

$Bb \longrightarrow bb$. Ces règles ne permettent jamais de raccourcir la chaîne en train d'être produite. La plupart d'entre elles autorisent simplement à déplacer certains symboles ou à remplacer un symbole non terminal par un terminal, mais *uniquement s'il se trouve dans un certain contexte, spécifié par ses symboles voisins*. Cette grammaire permet d'engendrer le langage $L = a^n b^n c^n$ $(n \geq 1)$ qui, pour des raisons similaires à celles détaillées en section 2.2 (il existe un "lemme de pompage" adapté aux structures arborescentes), est impossible à produire par une grammaire de type 2 ou par un RTR. Montrons par exemple la séquence de dérivations générant *aabbcc* (en soulignant chaque fois les suites de symboles correspondant aux parties gauches des règles de ré-écriture) :
$S \longrightarrow aBSc$
$aB\underline{Sc} \longrightarrow aBaBcc$
$a\underline{Ba}Bcc \longrightarrow aaBBcc$
$aaB\underline{Bc}c \longrightarrow aaBbcc$
$aa\underline{Bb}cc \longrightarrow aabbcc$
Cette séquence de dérivations n'est pas représentable dans un arbre.

Pour synthétiser toutes ces propriétés, nous les résumons dans le tableau suivant :

type	**nom des grammaires**	**forme des règles**	**structures produites**	**exemple typique**	**modèle équivalent**
3	régulières ou rationnelles	$A \longrightarrow a$ ou $A \longrightarrow aB$	peignes	$a^* = a^n$	automates finis
2	hors-contextes ou algébriques	$A \longrightarrow ...$	arbres	$a^n b^n$	RTRs
1	sensibles au contexte	$\alpha \longrightarrow \beta$ $\lvert\alpha\rvert \leq \lvert\beta\rvert$	?	$a^n b^n c^n$	existe mais compliqué
0	quelconques	quelconque	quelconques	quelconques	machines de Turing !

2.7 Position des langues naturelles dans la hiérarchie de Chomsky

A quelle classe de la hiérarchie de Chomsky appartiennent donc les langues naturelles ? La question a été un sujet de débat intense dans la communauté des linguistes-informaticiens. La réponse peut d'ailleurs varier suivant qu'on se contente d'une simple adéquation au critère de grammaticalité appliqué aux *phrases*, tel que formulé en section 1.1, ou qu'on exige une adéquation plus forte des *structures* produites par la grammaire aux analyses linguistiques.

Les arguments présentés en section 2.2 tendent tous à montrer que les langues naturelles ne sont pas de type 3. Le fait, désormais clairement établi, que les automates ne produisent que des peignes, alors qu'une analyse en termes de syntagmes requiert des arbres, est à lui seul assez rédhibitoire si on attache de l'importance à cette notion de structure. A contrario, la classe des grammaires de type 2, ou algébriques, est un candidat qui présente de sérieux arguments. Pourtant, nous avons aussi déjà vu en section 1.5 que certaines analyses syntaxiques ne pouvaient pas se contenter de représentations sous forme d'arbres : or, les arbres sont les seules stuctures produites

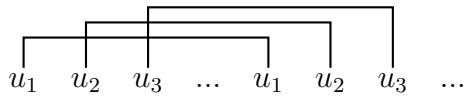


FIG. 5.16 – relations croisées dans une phrase de la forme ww où $w = u_1u_2u_3...$

par ces grammaires.

Dispose-t-on, comme c'était le cas pour les automates, d'exemples de langages impossibles à produire par les grammaires algébriques, et pourtant intégrées dans certaines langues naturelles ? Il semble bien que oui... Nous avons déjà évoqué le langage $a^nb^nc^n$, qui ne peut être engendré que par une grammaire de type 1. Un autre exemple est le langage $\{ww|w \in V^*\}$, c'est-à-dire l'ensemble des suites de symboles (terminaux) quelconques répétées deux fois successivement à l'identique. On peut se convaincre assez facilement que, pour obtenir de telles chaînes, il est nécessaire de générer *en même temps* les *mêmes symboles* dans chacune des deux suites -ce qui équivaut bien à avoir une structure avec des branches "qui se croisent", comme sur la figure 5.16, où $w = u_1u_2u_3....$ Les grammaires de type 2 sont impuissantes à réaliser cela.

Or, certains linguistes prétendent avoir reconnu des constructions relevant d'un des deux langages précédents dans le dialecte suisse alémanique, ou dans le génitif en géorgien ancien (authentique !), ou encore dans la manière de compter en chinois... L'argument, qui bien sûr suppose aussi l'adhésion préalable à la distinction compétence/performance, est plus difficile à vérifier que celui qui portait sur les propositions relatives enchassées, mais il est de même nature.

Si on accepte cet argument, est-on pour autant contraint de se rabattre sur le candidat suivant de notre liste : la classe des grammaires de type 1 ou contextuels ? Et bien non ! En effet, depuis quelques années, une classe intermédiaire contenant toutes les grammaires de type 2 mais strictement plus petite que l'ensemble de celles de type 1 a été identifiée : on l'appelle la classe des grammaires *légèrement sensibles au contexte* (traduction anglaise de "middly context-sensitive"). On ne peut malheureusement pas définir cette nouvelle classe aussi facilement que les autres, en posant des contraintes sur la forme des règles. Mais elle présente toutes les "bonnes propriétés" qu'on pouvait espérer : les grammaires qui en font partie permettent de produire les langages précédemment évoqués, et l'analyse syntaxique y est réalisable par des algorithmes efficaces. Plusieurs formalismes ont été proposés pour caractériser les grammaires "légèrement sensibles au contexte" : le plus célère d'entre eux est celui des "tree adjoining grammars" ou TAG.

Un certain consensus existe désormais autour de l'idée que la quasi-totalité des langues naturelles sont "légèrement sensibles au contexte". Mais le débat n'est pas totalement clos.

2.8 Autres formalismes

On l'a déjà dit : le domaine de la syntaxe est celui qui, depuis 1957 (date de la publication de "Syntactic Structures"), a cristallisé le plus d'efforts de formalisation. Autant dire que ces quelques pages ne peuvent qu'en donner une vision très partielle. La présentation qui en a été faite ici se focalise sur la *théorie des langages*, qui constitue en quelque sorte le socle commun sur lequel peuvent s'appuyer aussi bien les linguistes que les informaticiens. Elle repose sur la notion de *syntagmes*, et sur les grammaires formelles dîtes *syntagmatiques* qui les explicitent.

Les langues naturelles ne sont pas toujours faciles à décrire avec ce formalisme : celles, par exemple, où les relations syntaxiques s'expriment par des déclinaisons plus que par l'ordre des mots, posent des problèmes spécifiques. Les accords en genre et en nombre des langues romanes (comme le français) obligent également à la mise en place de mécanismes particuliers. Les grammaires formelles décrites précédemment sont donc en fait très difficilement exploitables telles quelles en traitement des langues. Mais elles ont donné naissance à une nombreuse progéniture.

De manière générale, on appelle *grammaire de constituants* tout formalisme qui réalise des analyses syntaxiques sous la forme de groupes de mots consécutifs récursivement emboîtés les uns dans les autres. On peut rattacher à cette famille les grammaires LFG, GPSG, HPSG...

La principale alternative à ce choix est représentée par les *grammaires de dépendances*, déjà évoquées en section 1.5. Dans les formalismes de cette famille, les relations de dépendances entre couples de mots remplacent les constituants. Il existe aussi, bien sûr, des systèmes hybrides qui, tout en construisant une analyse de type syntagmatique, marquent des relations de dépendances.

Les formalismes le plus en usage ces dernières années, quels que soient par ailleurs leurs partis-pris théoriques, ont un point commun : ils sont presque tous *lexicalisés*. Un formalisme lexicalisé opère une distinction claire entre les informations syntaxiques rattachées aux éléments de son vocabulaire (terminal) d'une part, et les règles qui permettent de combiner entre elles ces informations d'autre part. L'idéal est de disposer d'un nombre restreint de règles génériques, communes à toutes les instances de grammaires du formalisme. Ce qui distingue une grammaire d'une autre se limite alors aux informations rattachées à chacun de ses mots. Le principal intérêt de cette distinction est de faciliter les procédures de mises à jour : puisque les règles sont fixées une fois pour toute, seules les informations lexicales peuvent éventuellement être modifiées. Les formalismes lexicalisés les plus connus sont : les grammaires d'unification, les grammaires catégorielles et les LTAG (variante lexicalisée des TAG, évoquées en section précédente). Un exemple de grammaire catégorielle élémentaire est défini et utilisé plus loin, en chapitre 7, section 2.3.

Enfin, un dernier type de formalisme occupe une place un peu à part dans ce panorama : celui des grammaires minimalistes, qui essaient de formaliser les derniers écrits de Chomsky sur la syntaxe. C'est un modèle lexicalisé, qui ne fait usage que de deux règles génériques : une règle de "fusion" et une règle de "déplacement". Sa particularité est de faire l'hypothèse que certaines constructions syntaxiques sont le résultat de déplacements de constituants qui laissent derrière eux des "traces". Une trace est en quelque sorte la "place vide" laissée par un constituant qui a été

déplacé : bien qu'invisible, elle est censée avoir des effets mesurables sur l'ensemble de la phrase dont elle fait partie.

L'expressivité de ces formalismes est variable : certains ne peuvent engendrer que les langages appartenant à une certaine classe de la hiérarchie de Chomsky, d'autres sont plus génériques. L'étude des propriétés de ces modèles, leurs extensions et leurs comparaisons sont encore des thème de recherche très actifs à l'heure actuelle.

2.9 Sites Web

Pour s'initier aux grammaires formelles, on pourra avec profit télécharger le petit programme pédagogique "Gram" de Jean Véronis (uniquement pour Windows) : www.up.univ-mrs.fr/veronis/logiciels/index.html

Il existe par ailleurs de très nombreux programmes, gratuits ou payants, qui implémentent une grammaire d'une langue donnée, dans un formalisme particulier. Aucun n'a pourtant la prétention de modéliser la compétence complète d'un locuteur de cette langue (tous ont leurs points faibles). Il existe aussi des "chunk parsers" ou encore des "shallow parsers", qui réalisent des analyses syntaxiques partielles, "superficielles", au sens où ils se contentent d'identifier certains constituants non récursifs, sans aller jusqu'à poduire une structure complète.

Les sites suivants permettent de produire des analyses syntaxiques en ligne dans diverses langues, et suivants divers formalismes :

- visl.sdu.dk/visl/fr/parsing/automatic/complex.php
- www.connexor.com/demo/syntax/
- lfg-demo.computing.dcu.ie/lfgparser.html
- www.link.cs.cmu.edu/link/submit-sentence-4.html
- www.teemapoint.net/nlpdemo/servlet/ParserServlet
- l2r.cs.uiuc.edu/∼cogcomp/eoh/chunkerdemo.html

Il nous reste maintenant à aborder le dernier niveau de l'analyse linguistique : le niveau *sémantique*, celui qui traite de la question du *sens* qui peut être associé à une unité ou à une suite d'unités lexicales. Le principal intérêt d'une langue, c'est en effet qu'elle est *signifiante* : elle permet de véhiculer du sens. La sémantique est ainsi beaucoup plus qu'un niveau "de plus" dans la chaîne des traitements linguistiques : c'est tout simplement sa justification, sa raison d'être. Dans notre schéma inaugural de la figure 2.1, la sémantique constitue une "dimension" spécifique de l'analyse du langage, et elle se décline en deux niveaux distincts : celui de la sémantique lexicale, et celui de la sémantique propositionnelle. Chacun de ces deux niveaux mérite attention et étude spécifiques, et sont donc l'objet des deux chapitres qui suivent.

Chapitre 6

La sémantique lexicale

La sémantique lexicale est l'étude du sens des "mots" -ou plutôt des morphèmes- d'une langue. Cette définition est en réalité assez problématique, puisque la notion même de "sens" n'a rien d'évidente. Le problème tient précisément à ce que, pour définir le "sens" d'un mot, on recourt en général à d'autres mots. Pourtant, la consultation d'un dictionnaire d'une langue donnée est de bien peu d'utilité si on n'a pas déjà d'un minimum de connaissance de cette langue. Comment échapper à cette "circularité du sens" ? Nous evisageons dans ce chapitre (et le suivant) diverses tentatives qui peuvent être regroupées en trois familles :

- utilisation de "primitives sémantiques" (non analysables) à partir desquelles les sens des mots peuvent être construits par combinaisons ;
- définition de "réseaux de sens" à l'aide de relations entre morphèmes ou entre concepts : le sens d'un mot se confond alors avec sa position dans le réseau ;
- définition d'un "monde de sens" (fondé en général sur les mathématiques et la logique) sur lequel on peut projeter le sens des morphèmes.

Nous verrons que ces familles ont donné lieu à des théories variées qui ne s'excluent pas nécessairement les unes les autres. Nous verrons aussi qu'elles traitent avec plus ou moins de bonheur de la diversité des "sens" associés aux différentes catégories de morphèmes : le sens d'un morphème grammatical est évidemment beaucoup plus difficile à cerner que celui d'un morphème lexical. Le domaine de la *lexicologie* s'intéresse d'ailleurs exclusivement à la sémantique de ces derniers.

1 Description linguistique (et autres)

En matière de sémantique, la linguistique entre souvent en dialogue avec d'autres disciplines, en particulier avec la psychologie et la philosophie. Nous évoquons ci-dessous quelques-unes des notions fondamentales que ces disciplines ont contribué à mettre à jour, tout en en détaillant les approches plus spécifiquement linguistiques. Lorsque cette description atteint un certain niveau de précision, elle peut très bien se confondre avec un modèle informatique. La distinction entre "description linguistique" et "modélisation informatique" qui structurait les chapitres précédents sera donc moins rigoureusement tenue qu'avant : la section informatique sera plus réduite, et contiendra surtout un état des lieux des implémentations mettant en œuvre les théories présentées dans cette première section.

1.1 Des choses aux mots

Pendant très longtemps, la sémantique lexicale a été conçue en termes "analogiques" : pour Platon et tous les représentants de la philosophie médiévale, un mot représente un sens par la vertu d'une certaine "ressemblance" ou "similitude" avec lui. Saussure a, de ce point de vue, marqué une rupture, en insistant sur le caractère fondamentalement arbitraire du rapport entre les mots et les choses ou, comme il le formule, entre les "signifiants" et les "signifés". Dans la plupart des langues, seules quelques onomatopées (imitation de cris d'animaux par exemple) gardent un lien non arbitraire avec ce à quoi ils réfèrent.

A cette dualité "signifié/signifiant", certains auteurs substituent un "triangle sémiotique" qui relie trois termes, comme dans la figure 6.1. La sémiologie (ou sémiotique) étudie les systèmes de communication par signes, dont les langues naturelles ne sont que des cas particuliers.

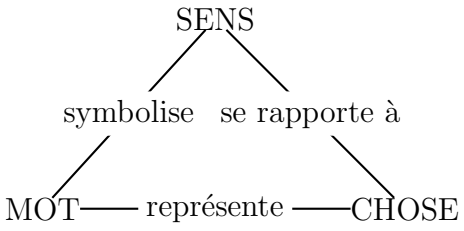


FIG. 6.1 – triangle sémiotique

Explorons donc quelques uns des liens entre ces termes.

Nombre d'auteurs ont remarqué que les différentes langues humaines ne "découpent" pas le monde de la même manière. Par exemple, la traduction du mot français "bois" en allemand ne sera pas la même suivant qu'on veut désigner le *matériau* ("Holz") ou un *sous-bois*, une petite forêt ("Wald"). En danois, ce sont encore d'autres choix qui sont à l'œuvre, comme le montre le tableau suivant :

français	arbre	(sous-)bois	forêt
allemand	Baum	Holz	Wald
danois	troe		skov

La façon de nommer les couleurs est un autre exemple classique de différences linguistico-culturelles. Le "spectre des couleurs" est en effet un continuum physique de longueurs d'ondes : en voir sept dans un arc-en-ciel est un choix qui n'a rien d'universel. Pour étudier les alternatives, les psychologues procèdent à des expériences où ils présentent à des locuteurs de langues différentes des petits morceaux de carton, en leur demandant de ranger ensemble ceux qui sont de la "même couleur". Le tableau suivant montre les regroupements réalisés par des représentants de cultures différentes :

français	*indigo*	*bleu*	*vert*	*jaune*	*orange*	*rouge*
chona (Zambie)	*cips*^{w} *uka*		*citema*	*cicera*	*cips*^{w} *uka*	
bassa (Liberia)	*hui* ("froides")			*ziza* ("chaudes")		

Donner le même nom de couleur à des cartons qu'on qualifie en français de "jaune" et "rouge" ne signifie pas du tout "ne voir aucune différence" entre ces cartons. Un nom de couleur doit plutôt être conçu comme une *catégorie*, c'est-à-dire une *classe d'équivalence de perceptions* acquise par éducation. De même, tout francophone a appris à désigner par la catégorie "chien" un grand nombre de perceptions très différentes... Cette capacité de catégorisation est une compétence fondamentale des êtres humains : c'est grâce à elle que le monde réel, appréhendé par des perceptions physiologiques (sons, images, gôuts, etc.) *continues*, peut être *discrétisé* en unité linguistiques distinctes.

Chaque langue "découpe" ainsi le réel suivant ses propres "catégories". Suivant certains auteurs, ce découpage a des répercutions sur le fonctionnement de la pensée. On désigne par "thèse de Sapir-Whorf" l'hypothèse suivant laquelle "les mondes où vivent des sociétés [linguistiquement] différentes sont des mondes distincts, pas simplement le même monde avec d'autres étiquettes". La langue induirait en quelque sorte une "vision du monde" ainsi qu'une manière spécifique de se situer dans l'espace et le temps, et donc finalement un mode de pensée particulier.

Cette relativité linguistique radicale est néanmoins controversée. Certaines études ont ainsi mis à jour des *invariants cognitifs* dans l'espèce humaine, qui se retrouvent dans certains traits de ses langues. Par exemple, si une langue ne dispose que de deux mots pour les couleurs, la distinction opérée sera toujours clair/foncé ou chaud/froid. Et si un troisième mot existe, il servira presque toujours à désigner la couleur rouge...

1.2 Sens et référence

On doit à Frege (1848-1925), mathématicien et philosophe allemand, par ailleurs aussi inventeur de la logique des prédicats du 1er ordre (que nous évoquerons dans le chapitre suivant), une distinction fondamentale entre le *sens* et la *référence* (ou la *dénotation*) d'une expression linguistique.

Pour définir la notion de sens, on peut s'inspirer de la façon dont nous avons procédé pour définir les catégories grammaticales : *deux unités linguistiques ont le même sens si elles sont substituables l'une à l'autre dans n'importe quelle proposition en préservant la valeur de vérité de cette proposition.* Un "sens" est ainsi, encore une fois, une classe d'équivalence pour une relation de substituabilité, et ce qui doit rester invariant est encore ce qui caractérise le "niveau d'analyse supérieur" (ici : la notion de "valeur de vérité"). On dit aussi que deux expressions sont "synonymes", c'est-à-dire ont le même sens, si elle sont interchangeables dans n'importe quel contexte.

Tout le mérite de Frege est de montrer que cette notion de sens ne coïncide pas avec la fonction de "désignation d'une portion du monde" par laquelle on la définit parfois. L'exemple, devenu classique, qui permet d'illustrer cette distinction, est tout ce qui sépare "l'étoile du matin" de "l'étoile du soir". Ces deux termes désignent en fait le même "astre" : la planète Vénus (qui est aussi, d'ailleurs, "l'étoile du berger"). On dit qu'ils ont la même *dénotation* ou qu'ils *réfèrent* à la même portion de la réalité. Ils ne sont pas pour autant substituables dans *tous les contextes linguistiques.* Par exemple, on peut très bien associer une valeur de vérité différente aux deux propositions suivantes :

– Jean aime l'étoile du matin.
– Jean aime l'étoile du soir.

Deux expressions qui partagent la même référence n'ont donc pas nécessairement le même sens.

Cette distinction sens/référence a été reprise plus tard par Carnap (1891-1970), autre philosophe allemand (puis américain), sous la forme d'une distinction similaire entre ce qu'il appelle "l'intension" et "l'extension" d'un concept (ou d'une catégorie, pour reprendre notre terme précédent). Pour la comprendre, faisons un détour par les mathématiques. L'ensemble noté P des nombres pairs peut être défini de deux façons radicalement différentes :

– $P = \{n \in \mathbb{N} \mid \exists p \in \mathbb{N},\ n = 2p\}$
– $P = \{0, 2, 4, 6, 8...\}$

Le premier ensemble auquel P est égal est une caractérisation abstraite : c'est une définition *en intension* ou *intensionnelle*, tandis que le deuxième est une simple énumération de ses éléments : c'est une définition *en extension* ou *extensionnelle*. De même, on peut très bien définir la catégorie des "chiens" en intension, en la caractérisant comme une espèce animale ayant telles propriétés physiologiques ou génétiques, ou bien en extension, en énumérant les unes après les autres toutes ses instances ("Médor", "Milou", "Idéfix", etc.). Le première définition coïncide (à peu près) avec la notion de sens, la deuxième avec celle de référence.

1.3 Décomposition en primitives "sémiques"

L'analyse *sémique* ou *componentielle* (terme anglo-saxon) est une des premières tentatives systématiques de décomposition du sens d'un mot en unités de sens élémentaires. Elle est née dans le contexte de ce qu'on appelle l'*analyse structurale* dans les années 1960.

Dans cette théorie, l'unité minimale de signification est appelée un *sème*. Un sème est un *trait sémantique minimal* dont les seules valeurs possibles sont : positif (+), négatif (-) ou sans objet (∅). En informatique, on dirait que c'est une variable booléenne. Un *sémème* est un faisceau de sèmes correspondant à une unité lexicale. L'analyse componentielle de l'ensemble des mots {jument, poulain, pouliche} pourrait ainsi ressembler au tableau suivant :

	cheval	mâle	adulte
jument	+	-	+
poulain	+	+	+
pouliche	+	-	-

Les mots ayant un (ou des) sème(s) positifs en commun appartiennent au même *champ sémantique* : c'est le cas des trois mots de notre exemple, qui partagent le trait "cheval". Mais les sèmes ont aussi une fonction distinctive (ou contrastive) puisqu'ils rendent explicite ce qui fait la différence entre chaque couple de mots pris parmi ces exemples.

Les sèmes peuvent aussi servir à caractériser les différents sens possibles de mots ambigus. La figure 6.2 donne sous la forme d'un arbre la décomposition en sèmes

des six significations possibles du mot “canard”. Ce genre de représentation montre qu'une analyse sémique sert aussi à *relier les unes aux autres les significations*. Elle ne s'oppose donc pas à une conception *systémique* du sens, suivant laquelle les unités lexicales ne prennent sens que les unes par rapport aux autres, dans un réseau (conception que nous reprenons plus loin).

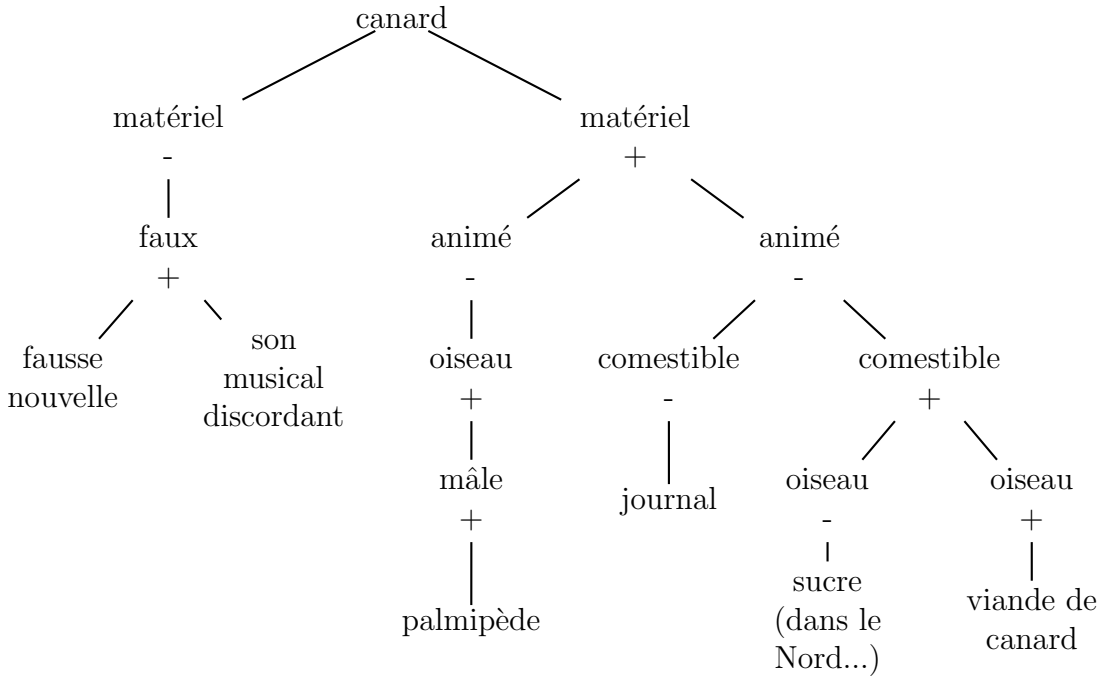


FIG. 6.2 – arbre de désambiguisation à base de sèmes

Disposer d'un tel arbre peut servir à désambiguiser l'usage d'un mot dans un texte. Par exemple, dans la phrase “Jean tenait le canard dans sa main”, le verbe “tenir” contraint son complément d'objet à avoir un trait “matériel”, ce qui élimine toute la branche gauche de la structure. Si, dans la suite du texte, nous apprenons que “le canard s'envola”, le trait “animé” doit être vérifié et l'ambiguité quant au sens du mot “canard” est levée. Mais une telle analyse nécessite quelques informations complémentaires (traits acceptés ou non en sujet ou complément de verbes) et est donc difficile à automatiser complètement.

Les critiques qu'on peut formuler à l'encontre de ce genre d'analyses sont les suivantes :

- même dans un domaine restreint, il y a rarement concensus entre différents auteurs sur la nature des sèmes pertinents
- personne n'a jamais prétendu exhiber un ensemble fini de sèmes à partir desquels les sens de tous les morphèmes d'une langue donnée (voire de toutes les langues possibles) pourraient être définis. Ce serait une tâche titanesque et vaine, puisque l'analyse structurale vise plutôt à comprendre les *relations distinctives* entre unités lexicales.
- l'analyse sémique est surtout adaptée aux mots lexicaux : quel ensemble de sèmes associer à des morphèmes grammaticaux comme “par”, “de”, “qui”, “dont”, “or”, etc. ?

– quelle est exactement la nature de la composition entre sèmes appartenant à un même sémème ? Se confond-elle simplement avec un "ET" logique entre variables booléennes ? Parfois, on aurait envie de modes de compositions plus élaborés, ainsi que de règles de compatibilités entre sèmes associés à des mots liés syntaxiquement (comme dans notre exemple de canard).

Il y a eu, à notre connaissance, peu de tentatives d'utilisation à grande échelle de l'analyse sémique en tant que telle. Mais elle reste une source d'inspiration pour certains travaux contemporains. Certains formalismes syntaxiques intègrent par exemple des "traits sémantiques" à leur vocabulaire, pour contraindre la construction de phrases non seulement syntaxiquement correctes, mais aussi sémantiquement interprétables.

1.4 Décomposition des actions selon Schank

Il existe d'autres tentatives de décomposition de la sémantique de certains morphèmes en primitives. Une de celles qui a été poussée le plus loin est due au spécialiste d'intelligence artificielle Roger Schank (né en 1946), et date des années 1970. Elle s'intègre dans une démarche globale de *représentation des connaissances*, et ce que nous en présentons ici ne rendra que très partiellement justice à l'ensemble de la proposition où elle s'inscrit (nous y reviendrons dans le chapitre suivant).

Schank estime pouvoir décrire la totalité des *actions* possibles en se ramenant à onze concepts de base. Cinq concernent les actions physiques :

– PROPEL : appliquer une force à quelque chose
– MOVE : déplacer une partie du corps
– GRASP : attraper un objet
– INGEST : introduire quelque chose à l'intérieur d'un objet animé
– EXPEL : rejeter quelque chose à l'exterieur d'un objet animé

Deux autres mettent l'accent sur le résultat produit par l'action :

– PTRANS : changer la position d'un objet physique
– ATRANS : changer une relation abstraite d'un objet (comme la possession)

Les deux suivantes interviennent comme instrument d'autres actions :

– SPEAK : produire un son
– ATTEND : diriger un organe sensoriel vers un stimulus

Et les deux dernières sont des "actes mentaux" :

– MTRANS : transfert d'information d'un indivisu à un autre, ou entre deux éléments de la mémoire d'un même individu
– MBUILD : création de nouvelles pensées

Par exemple, la vente d'un objet est une combinaison entre une action de transfert de possession de cet objet entre deux agents, et une action de transfert d'argent dans la direction opposée. Pour que ce genre de combinaisons soit possible, il faut ajouter un certains nombres de descripteurs que nous n'aborderons que dans le chapitre suivant, parce qu'ils relèvent plutôt de la *sémantique propositionnelle* : c'est le cas notamment des *rôles sémantiques*, qui décrivent des fonctions comme "acteur", "objet", "bénéficiaire", etc.

Cette proposition a le mérite de se présenter comme complète, et d'être intégrée dans un système de représentation des connaissances de plus haut niveau. Son ori-

ginalité est de s'appliquer plutôt aux *verbes* qu'aux noms, ce qui est assez rare pour être souligné. Elle a été mise en œuvre par Schank, et a été parfois reprise dans certains systèmes ultérieurs. Nous la retrouverons dans le chapitre suivant, intégrée à des formalismes de représentation du sens de propositions complètes (voir le chapitre 7, section 2.4).

1.5 Analyses par prototypes et proximités

Pour éviter la "circularité du sens" dans le domaine du lexique, la principale alternative à la décomposition en primitives repose sur la prise en compte des *liens qu'entretiennent les morphèmes entre eux*. Saussure, déjà, considérait que le langage constituait un *système* et que la sémantique d'une unité se mesurait surtout à l'aune de ses différences et similitudes avec les autres unités (suivant un axe "paradigmatique").

Cette hypothèse a été en partie confirmée par des études psycholinguistiques menée par Eleanor Rosch sur l'organisation de la mémoire humaine menées dans les années 70, visant à comprendre comment s'opérait la *catégorisation*. La conception classique, datant d'Aristote, qu'on avait à cette époque d'une catégorie était celle d'une conjonction nécessaire et suffisante de traits (qu'on peut assimiler à des sèmes) : on supposait, par exemple, que l'appartenance d'un individu à la catégorie " oiseau" pouvait se décider en vérifiant si celui-ci satisfaisait chacun des critères de l'ensemble { vole, a-des-plumes, a-des-ailes, a-un-bec, est-ovipare, etc.}. Pourtant, tous les individus catégorisés en "oiseau" ne satisfont pas chacun de ces traits : l'autruche, le pingouin ou le poussin ne volent pas, le kiwi et le pingouin n'ont pas vraiment de plumes, etc.

En fait, il semble que la catégorisation humaine fonctionne plutôt par *proximité à un prototype*. A chaque catégorie, pourrait ainsi être associé un *représentant privilégié* appelé *prototype*, qui exprime l'essence des propriétés des membres de cette catégorie : le "moineau" ou le "rouge-gorge" pourraient jouer ce rôle pour la catégorie "oiseau". L'appartenance d'un individu à une catégorie s'évalue alors par sa proximité avec le prototype : c'est une question de degré plus que de frontière claire. Ce fonctionnement a pu être validé en mesurant les temps de réponse à des tests de catégorisation : plus on présente à un sujet humain une instance qui "ressemble" au prototype, plus son appartenance à la catégorie est jugée rapidement.

Cette nouvelle conception de la sémantique en terme de "proximités" est souvent rapprochée de la "ressemblance de famille" proposée bien avant par le philosophe Wittgenstein (1889-1951) pour analyser "les processus que nous nommons "jeux"" : jeux de plateau, de cartes ou de balles, jeux solitaires ou collectifs, jeux compétitifs -ou pas-, divertissants -ou pas-, où interviennent -ou pas- la chance, l'adresse... L'ensemble des jeux n'est pas caractérisable par un ensemble de traits fixe : à la place, "nous voyons un réseau complexe de similitudes qui s'entrecroisent et s'enveloppent les unes les autres". Certains auteurs ont proposé de représenter ce "réseau" par des cercles qui ont certaines intersections communes, à la manière de la figure 6.3.

Il faut imaginer que chaque instance de "jeu" figure dans un des cercles. Cette figure très simplificatrice (on peut envisager des recoupements plus complexes) illustre surtout que chaque cercle a des traits commun avec un ou plusieurs autres, mais aussi

qu'il existe des cercles (et donc des instances de jeux) qui ne partagent aucun trait. Ils sont identifiés comme appartenant à la catégorie des jeux en vertu de proximités successives, et non parce qu'ils vérifient une propriété particulière.

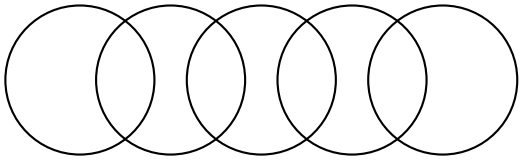


FIG. 6.3 – représentation graphique de la "ressemblance de famille"

1.6 Organisations hiérarchiques

Même si, bien sûr, ce vocabulaire n'était pas présent dans ses textes, chacun des types de jeux évoqués par Wittgenstein constitue une "sous-catégorie" spécifique de la catégorie "jeu" et pourrait même avoir son prototype propre. De même, la catégorie "oiseau" est un cas particulier de la catégorie "animal", elle-même cas particulier de la catégorie "être vivant", etc. L'ensemble des catégories peut donc être organisé de manière *hiérarchique* par la relation "est un cas particulier de", comme dans la figure 6.4 où cette relation est représentée par une flèche.

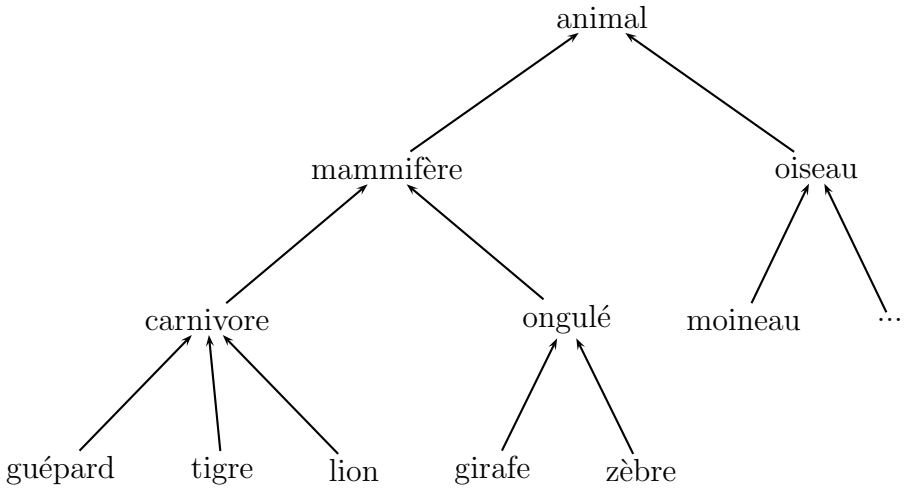


FIG. 6.4 – organisation hiérarchique d'une ensemble de catégories

Ce genre de modélisation est devenue la base des langages de programmation "orientés objets". Ils s'accompagnent dans ce cadre de mécanismes d'*héritage* : sauf exceptions explicitement déclarées, les catégories "inférieures" héritent des propriétés et fonctions de leur(s) ancêtre(s). Par exemple, si la propriété "voler" est associée à la catégorie "oiseau", elle sera automatiquement associée aussi à toutes les sous-catégories d'oiseaux.

La mémoire humaine est probablement aussi organisée de façon similaire. Mais tous les éléments figurant dans une structure de ce genre ne sont pas pour autant

équivalents d'un point de vue psychologique : des expériences ont montré que, quand on demande à quelqu'un de décrire une situation ou une image, il a tendance à utiliser les catégories se situant à un certain niveau dit "de base" dans cette hiérarchie : il utilisera le mot "oiseau" ou "lion" plutôt qu'"animal". Ces mots ne se situent d'ailleurs pas toujours au même niveau global dans notre arbre de la figure 6.4 : le "niveau de base" psycholinguistique ne correspond pas nécessairement à un niveau précis des taxonomies scientifiques ; il peut être différent pour chaque "branche" de l'arbre.

Une organisation hiérarchique implique l'existence d'une *relation d'ordre*. Ainsi, malgré les apparences, l'arbre de la figure 6.2 permettant de distinguer les différents sens de "canard" n'est pas vraiment hiérarchique. En effet, l'ordre dans lequel les sèmes y sont évalués est relativement arbitraire, et on peut même retrouver le même test à des niveaux distincts dans différentes branches (c'est le cas du trait "oiseau" dans notre exemple).

Les arbres du type de la figure 6.4 ont une traduction contemporaine dans la notion d'*ontologie*. Ce terme, hérité de la philosophie, signifie littéralement "science de ce qui est". L'informatique l'a réinvesti récemment pour le "Web sémantique", projet de mise en commun de connaissances générales et de données factuelles via Internet. Dans ce contexte, une ontologie est une hiérarchie de *concepts* censés être indépendants d'une langue donnée et founissant un vocabulaire général (comme le fait que les oiseaux sont des animaux et volent) pour décrire des connaissances plus spécifiques (concernant par exemple des instances particulières d'oiseaux).

Il existe une autre branche de la linguistique dans laquelle la structuration hiérarchique est bien ancrée : c'est la *terminologie*, l'étude des *termes* (cf. chapitre 5, section 1.2) pouvant servir de mots clés dans l'indexation d'un texte. Pour aider les documentalistes à naviguer dans un ensemble de mots clés, les terminologues les organisent dans un *thesaurus* comme celui de la figure 6.5 (fondé sur un petit échantillon de genres cinématographiques).

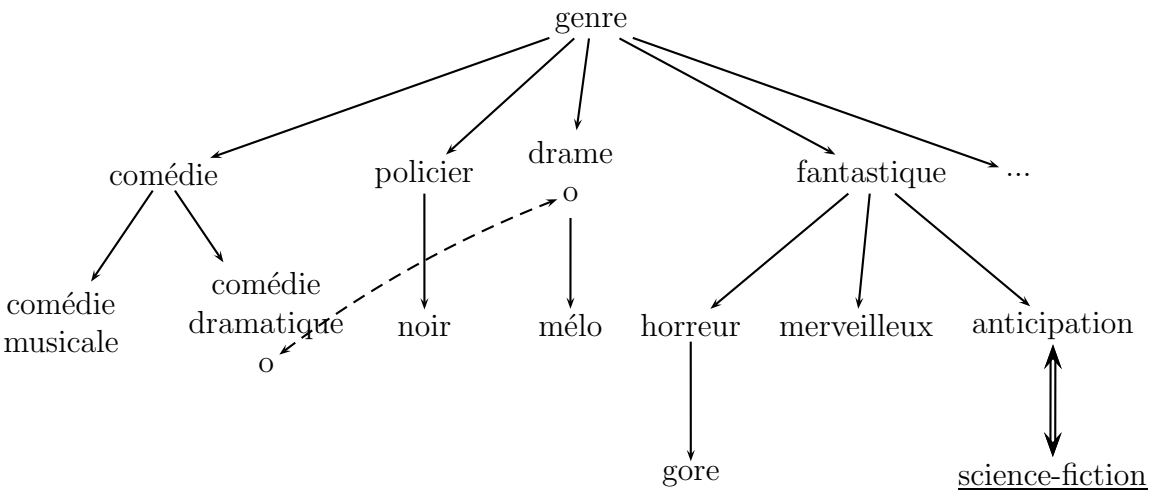


FIG. 6.5 – thesaurus des genres cinématographiques

Les principales propriétés d'un thesaurus comme celui-ci sont les suivantes :

- il ne contient que des *termes*, sous la forme de noms communs ou de groupes nominaux, en général en forme lemmatisée ;
- les termes figurant dans un thesaurus sont censés être représentatifs d'un certain domaine ;
- parmi ces termes, certains sont *descripteurs* (ils peuvent servir à indexer un texte), tandis que d'autres sont *non descripteurs* (ils ne doivent pas être utilisés lors d'une l'indexation). Ces derniers doivent être reliés à des termes descripteurs du thesaurus, qui leur seront substitués si quelqu'un les emploie lors d'une requête. Dans notre exemple, le terme "science fiction" est souligné pour signaler qu'il n'est pas descripteur.
- les termes sont liés entre eux par des *relations*. On considère classiquement trois types de relations possibles :
  - la relation "générique/spécifique" est une relation d'ordre : elle fournit la structure globale du thesaurus mais ne le contraint pas pour autant à être représentable sous la forme d'un unique *arbre* : il n'y a pas nécessairement unicité du terme le plus général. Elle est représentée dans notre figure par une flèche.
  - la relation de "synonymie" (aussi notée "employée pour") est symétrique et relie deux termes pouvant être échangés l'un avec l'autre : de ce fait, elle est souvent utilisée pour relier un terme descripteur et un terme non descripteur, comme c'est le cas dans la figure entre "anticipation" et "science fiction".
  - la relation "terme associée" associe deux termes du même "champ sémantique" sans être pour autant synonymes : elle sert à étendre une requête à des termes sémantiquement proches : la flèche en pointillées entre "drame" et "comédie dramtique" en est une instance dans notre thesaurus.

Les thesauri sont des outils linguistiques : ils permettent d'élargir un critère de recherche d'information sur la base de relations sémantiques contrôlées. Il n'ont pas pour ambition de modéliser le sens global des termes qui y figurent, mais seulement d'expliciter quelques-une de leurs relations.

D'autres relations sémantiques peuvent bien sûr être définies entre unités lexicales, et contribuer à enrichir un réseau de sens plus complexe qu'un arbre.

1.7 Critiques générales et alternatives

Nous avons dans ce chapitre, plus que dans les précédents, cité les noms des auteurs des théories et concepts introduits. C'était, en partie, parce que le corpus des travaux sur la sémantique n'est pas aussi consensuel et "unifié" que certains des autres domaines évoqués. Peut-être n'a-t-il pas encore atteint le même niveau de maturité scientifique.

Aucune des théories et propositions évoquées jusque là ne sont en effet exemptes de critiques. Elle ne traitent, pour la plupart, que de la caractérisation du sens d'une toute petite partie d'un lexique : en général les noms communs, plus rarement les verbes.

Que dire, par exemple, du sens des adjectifs ? Ces derniers servent traditionnellement à exprimer des *qualités* attribuées aux noms auxquels ils s'adjoignent. Mais la caractérisation du sens d'un adjectif est un problème difficile, car il dépend souvent

de son environnement syntaxique. Certains adjectifs, en effet, décrivent une propriété relativement "autonome" : les couleurs d'une "maison bleue" et d'une "orange bleue" sont sans doute comparables. Mais d'autres adjectifs sont indissociables du nom qu'ils qualifient : un "grand écrivain" est "grand en tant qu'écrivain" et pas nécessairement "grand" pour autant. D'ailleurs, même le sens usuel de "grand" est relatif à l'échelle du nom considéré : une "grande souris" et une "grande girafe" n'ont certainement pas la même taille. Sans compter que certains adjectifs changent de sens en français suivant qu'ils sont anté ou post-posés (mis avant ou après le nom) : un "homme pauvre" n'est pas forcément un "pauvre homme", etc.

Autant les noms communs et les entités nommées jouent un rôle crucial pour savoir "de quoi" parle un texte, autant les adjectifs et les verbes sont fondamentaux pour connaître "comment" il en parle. Les systèmes qui essaient de classer les *textes porteurs d'opinion* (par exemple les critiques littéraires ou cinématographiques, les argumentations militantes, les compte-rendus d'utilisation d'un produit, etc.) en plus ou moins "favorables" se focalisent en général sur les adjectifs et les verbes qu'il contiennent.

Le cas des adverbes est encore moins souvent abordé. Ces derniers jouent en effet souvent le rôle de *modifieurs* : leur fonction est de renforcer ou d'amoindrir le sens des mots (en général adjectifs ou verbes) avec lesquels ils apparaissent, comme dans "courir vite" ou "très cher". Leur sémantique est donc difficile à décrire en elle-même. Et, enfin, le sens des morphèmes grammaticaux ne peut être envisagé que si on dispose d'une théorie du sens global d'une proposition, dont ils constituent des briques indispensables. Nous y reviendrons donc dans le chapitre suivant.

Citons pour finir une approche récente et assez prometteuse, qui tente de remédier à plusieurs de ces critiques : la théorie du *lexique génératif*, introduite par le linguiste américain Pustejovsky dans un livre de 1998. Son ambition est d'expliquer la sémantique de toutes les unités lexicales en contexte à partir d'un "ensemble noyaux" et de mécanismes génératifs et dérivationnels. Pour cela, Pustejovsky propose d'associer à chaque unité lexicale quatre niveaux distincts de représentation :

- une *structure argumentale* qui spécifie le nombre et le type de ses arguments (ex : sujet et complément direct d'un verbe transitif)
- une *structure événementielle*, par laquelle on distingue les *activités*, les *états* et les *transitions*
- une *structure de qualia*, qui se décompose elle-même en quatre *rôles* :
  - le *rôle constitutif* décrit la relation entre l'objet référencé par le mot et ses composantes : par exemple, pour un "livre", ses pages, sa couverture...
  - le *rôle formel* est ce qui distingue l'objet d'un domaine plus large : celui d'un livre est de contenir de l'information
  - le *rôle télic* est la fonction pour laquelle l'objet a été conçu : celle d'un livre est d'être lu
  - le *rôle agentif* explicite les facteurs impliqués dans sa création : l'auteur du livre.
- une *structure d'héritage lexical*

On le voit, cette théorie est déjà très articulée et complexe, et il n'est pas question de la détailler plus avant ici. Elle cherche à analyser le sens d'un mot en le décomposant en différentes facettes, qui contraignent la façon dont il se combine

avec d'autres dans une phrase. Elle est l'objet de nombreux travaux et discussions à l'heure actuelle.

Les approches citées jusqu'à présent relèvent surtout des deux premières "familles" identifiées en introduction de ce chapitre : celles qui privilégient le *découpage en primitives* (ou en "facettes" !) et celles qui préfèrent une conception du sens *en réseau*. La troisième famille, où l'on cherche à définir un "monde de sens" propre, concerne en fait plutôt la sémantique propositionnelle et sera donc surtout représentée par les exemples du chapitre qui suit.

2 Modélisation informatique

Nous l'avons déjà annoncé : cette partie sera limitée parce que nous avons préféré mettre sur le même plan, dans la partie précédente, les "descriptions linguistiques" et les "modèles informatiques". La frontière entre les disciplines est parfois difficile à tracer en la matière. Par exemple, l'analyse sémique a été élaborée dans un contexte linguistique, mais elle est implémentable assez facilement (et cela a parfois été tenté). Pour continuer à brouiller les pistes, nous nous contenterons ici de retranscrire un argumentaire philosphique célèbre qui nie la possibilité même pour un ordinateur d'avoir accès au "sens" véhiculée par le langage.

2.1 Critique épistémologique

En 1980, le philosophe américain John Searle a fait paraître un article où il imaginait ce qui est désormais connu comme l'expérience de pensée de la "chambre chinoise".

Dans cette expérience, il demande au lecteur de se mettre dans la peau d'un expérimentateur enfermé dans une pièce. La pièce communique avec un interlocuteur extérieur grâce à un dispositif matériel quelconque (papier passé sous la porte ou boîte aux lettres électronique). Les informations qui s'échangent par ce canal se font uniquement *en chinois*. Bien sûr, Searle s'adresse à des lecteurs qui ne sont pas censés parler le chinois : la langue elle-même a peu d'importance mais l'important est qu'elle n'est pas comprise par la personne se trouvant dans la pièce.

L'expérimentateur dispose d'un stock suffisant de caractères chinois (ou il sait les produire avec un ordinateur), et d'un manuel rédigé dans sa langue maternelle. Quand un texte en chinois lui parvient, il consulte le manuel. Celui-ci est rédigé sous forme de règles qui lui expliquent, en fonction des caractères figurant dans le message reçu, quels caractères il doit à son tour choisir pour répondre. L'expérimentateur se contente d'exécuter les ordres du manuel. Si celui-ci est bien fait, on pourrait avoir l'impression qu'une vraie conversation en chinois a eu lieu entre la pièce et l'interlocuteur externe.

La "chambre chinoise" reproduit en fait le comportement d'un ordinateur programmé pour réussir le "test de Turing" (cf. chapitre 2, section 5). Le manuel joue le rôle du programme, l'expérimentateur celui de l'unité centrale. Même si le test est "réussi", la personne enfermée dans la pièce n'aura à aucun moment "compris" quoi que ce soit à la nature de la discussion, et encore moins à la langue chinoise... Le cœur de l'argument de Searle, c'est que la compréhension ne peut se réduire à la

manipulation syntaxique (au sens de "régie par des règles formelles") de symboles. Or, c'est la seule chose dont sont capables les ordinateurs.

Cet article a provoqué, dans les années qui ont suivi sa parution, un débat intense (pas encore totalement clos), où les échanges d'arguments et de contre-contre-arguments ont été particulièrement riches. Parmi les "reponses" les plus souvent mises en avant, on retiendra les suivantes :

- l'expérimentateur, à lui tout seul, ne comprend pas le chinois, mais cette compétence émerge peut-être de l'ensemble du dispositif, de même qu'aucun neurone ne comprend non plus le langage, qui est une compétence plus globale du cerveau.
- le sens émerge du lien entre les mots et les choses auxquels ils réfèrent : ce qui manque aux ordinateurs, ce sont des organes sensoriels pour percevoir leur environnement, les catégoriser, et les relier à des unités lexicales. On peut ainsi espérer que des robots dotés de capteurs, capables de catégoriser les données qui leur en parviennent et de leur associer un symbole linguistique auraient, eux, une vraie "compréhension" de ces symboles.

L'engouement qu'a suscité l'article de Searle témoigne à tout le moins qu'il a pointé un problème important pour la communauté des chercheurs en "intelligence artificielle". Depuis, les informaticiens ne prétendent en général plus reproduire l'intelligence humaine dans leurs programmes, mais au mieux simuler ses effets par d'autres moyens. Ils adoptent la plupart du temps une démarche très pragmatique, consistant simplement à exploiter au mieux les capacités de calcul de leurs machines.

2.2 Sites Web

Les sites Web référencés ci-dessous illustrent principalement la représentation du sens d'unités lexicales sous forme de réseaux ou d'arbres. Dans le chapitre suivant, seront présentés d'autres projets qui visent à expliciter des connaissances plus générales, et qui intègrent souvent une décomposition du sens en unités plus élémentaires.

Le plus célèbre projet lexicologique mené jusqu'à une réalisation informatique effective est le réseau "Wordnet", né de travaux psycholinguistiques sur le fonctionnement de la mémoire humaine. "Wordnet" est un dictionnaire hypertexte -et bien plus que cela- : il contient des mots de toutes catégories (y compris des entitées nommées) et est structuré par un grand nombre de relations. A partir d'un mot, on peut avoir accès à son ou ses *hyperonymes* (mots "plus généraux que"), *hyponymes* (mots "moins généraux que"), *antonymes* ("mots de sens opposé à"), etc :
wordnet.princeton.edu/perl/webwn (en anglais)

Il en existe une version pour les langues européenne, nommée "Eurowordnet", incluant le français, mais elle est payante. Eurowodnet a, semble-t-il, été fait plus en traduisant Wordnet qu'en reconsidérant les liens sémantiques propres à chaque langue, et ses utilisateurs n'en sont pas toujours très satisfaits.

Quelques ontologies (anglaises) plus ou moins générales et ambitieuses :

- virtual.cvut.cz/kifb/en/toc/229.html
- www.cs.umd.edu/projects/plus/SHOE/onts/
- ontologyportal.org/

Quelques thesauri (français) dans lesquels on peut naviguer en ligne :
– grds.ebsi.umontreal.ca/r/thesaurus-activite-gouvernementale/site-web/categories.htm
– www.bdsp.tm.fr/TSP3/Default.asp
– europa.eu/eurovoc/ (multilingue)
Autres représentations de relations sémantiques en réseaux (pour le français) :
– www.enekia.com/thesaurus.html
– www.memodata.com/2004/fr/dictionnaire_des_synonymes/TGLinkBrowser.shtml

Chapitre 7

La sémanique propositionnelle

La distinction que nous établissons dans ce document entre sémantique lexicale et sémantique propositionnelle n'est pas toujours prise en compte aussi explicitement. Pourtant, la notion de *valeur de vérité*, qui fait la particularité des propositions, joue un rôle un peu marginal en sémantique lexicale. Elle n'est intervenue dans le chapitre précédent que comme ce qui doit être préservé quand on remplace une unité lexicale par un de ses synonymes. Elle sera cette fois au centre de nos préoccupations. Il existe en effet une tradition philosophique (plutôt anglo-saxonne) très influente qui définit le sens d'une proposition comme "les conditions qui doivent être vérifiées dans le monde pour qu'elle soit vraie". Cette tradition entretient des liens très étroits avec la *logique*, branche des mathématiques qui se focalise sur le *raisonnement juste* : celui qui, précisément, garantit l'enchaînement de propositions vraies. C'est aussi dans ce chapitre, par le biais de la notion de "modèle", lui aussi issu de la logique, que nous allons explorer la 3ème solution à l'impasse de la "circularité du sens", consistant à "projeter" dans un espace mathématique particulier les "sens" véhiculés par un énoncé.

1 Description linguistique

Nous exposons dans cette partie quelques-uns des concepts fondamentaux auxquels font appel les linguistes et les logiciens pour analyser le sens d'une proposition. La logique en tant que "langue formelle" sera, elle, plutôt présentée dans la partie "modélisation informatique" du chapitre, mais de manière très simplifiée car le domaine a atteint un très haut niveau de technicité, qu'on ne saurait viser dans le cadre de ce document.

1.1 Principe de compositionnalité

Comment se fait-il que le locuteur d'une langue naturelle quelconque soit capable *de comprendre ou de produire une proposition qu'il (ou elle) n'a jamais entendue auparavant ?*. Certes, on peut admettre que ce locuteur n'a à sa disposition qu'un vocabulaire fini. Mais une théorie qui se contenterait de répertorier la sémantique lexicale d'un ensemble fini de mots ne suffit pas : il faut expliquer comment les sens *se combinent entre eux* pour en produire de nouveaux.

Nous avons déjà vu au chapitre 5 qu'avec une grammaire formelle, qui est un dispositif entièrement fini, on peut produire une infinité de phrases, en associant à chacune d'elle une structure. Nous avons vu aussi que la structure syntaxique d'une phrase conditionne (ou, suivant le point de vue privilégié, est dépendant de) la façon dont celle-ci est interprétée (cf. les quatre structures possibles de "Time flies like an arrow", en figure 5.4).

Deux éléments contribuent donc à coup sûr à la compréhension d'une proposition : la sémantique des mots qui y figurent, et sa structure syntaxique. Le "Principe de compositionnalité" stipule que ces deux éléments sont en fait *les seuls nécessaires*. Dans sa formulation contemporaine, il s'énonce comme suit : "le sens d'une proposition ne dépend que du sens des mots qui la constituent et de leur mode de combinaison syntaxique". On peut le "résumer" en quelque sorte par le schéma de la figure 7.1, qui reprend celui de la figure 2.1, mais où les flèches en gras visualisent les points de départ et d'arrivée d'un *calcul*.

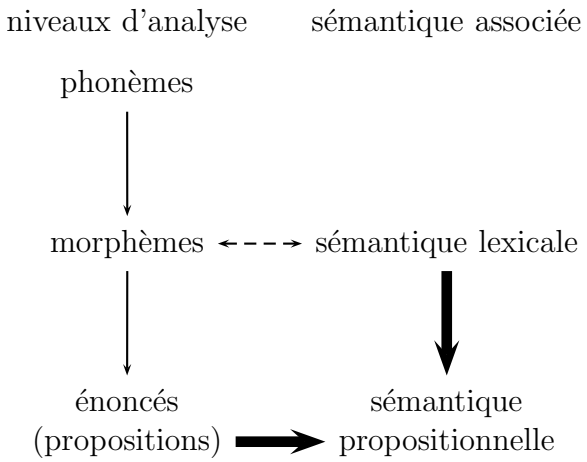


FIG. 7.1 – Le principe de compositionnalité

On fait parfois remonter ce principe à Frege, le créateur à la fin du XIXème siècle de la logique des prédicats du 1er ordre (abordée plus loin), mais sa formalisation précise date en fait des années 1970. C'est à cette époque que le logicien Richard Montague propose en effet un système syntaxico-sémantique complet et authentiquement compositionnel, constitué d'un ensemble fini de règles syntaxiques, chacune associée à une règle sémantique. A partir d'un arbre d'analyse, il suffit de remplacer les unes par les autres et de substituer aux mots leur "traduction sémantique" pour obtenir une formule qui, après calculs, représente le sens de la proposition initiale. Nous verrons en section 2.3 une version simplifiée de ce système. La connaissance d'un nombre fini d'éléments suffit donc bien, dans ce cas, à produire une infinité de combinaisons possibles (et ceci, bien sûr, grâce à la "productivité infinie" des règles syntaxiques récursives). Une proposition jamais entendue auparavant, du moment qu'elle ne contienne que des mots connus et qu'elle soit syntaxiquement analysable, est alors interprétable.

Ce principe est l'objet, encore à l'heure actuelle, de travaux et de controverses. Certains linguistes le trouvent insuffisant pour rendre compte du sens de certaines

constructions complexes. Chaque langue fournit ainsi son contingent d'énoncés non compositionnels : par exemple, le sens d'une expression figée idiomatique comme "casser sa pipe" n'est pas le résultat de la composition des sens des mots présents. Mais de telles expressions peuvent être considérées comme des "unités lexicales complexes" à elles toutes seules. Dans d'autres cas, des éléments de "contexte" ou de "pragmatique" (utilisation concrète du langage en situation), absents de la figure, semblent devoir intervenir dans le "calcul" du sens.

D'autres linguistes, au contraire, voudraient étendre son domaine d'application, en élaborant par exemple une "morphologie compositionnelle", donnant lieu à un schéma du genre de la figure 7.2 (d'autres extensions possibles concernent par exemples le niveau du "discours" qui demande une articulation supplémentaire au-delà des propositions). Le principe de compositionnalité est de toute façon une référence incontournable de la linguistique contemporaine. C'est en quelque sorte lui qui permet de faire "tenir ensemble" tous les niveaux d'analyse linguistique, et qui justifie l'apport propre de chacun d'eux au sens global d'un énoncé.

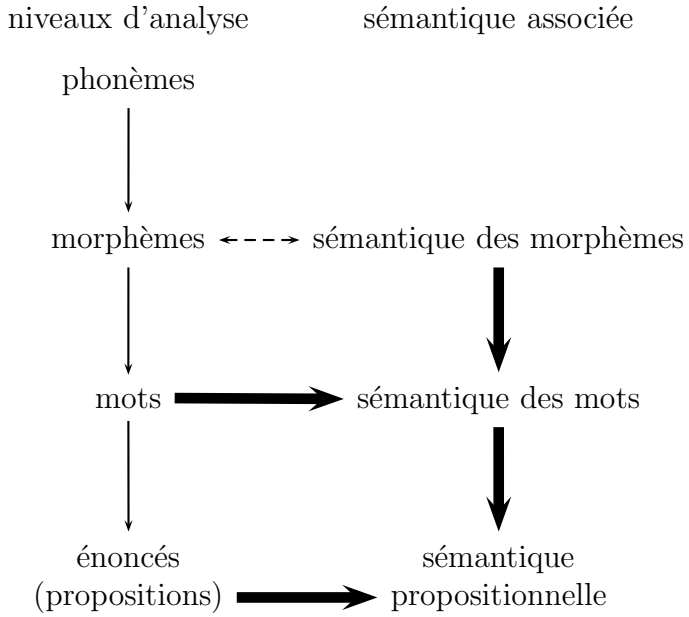


FIG. 7.2 – Le principe de compositionnalité généralisé

1.2 Prédicats verbaux et rôles thématiques

Rentrons maintenant un peu plus précisément dans l'analyse du sens d'une proposition. Le *verbe* y tient incontestablement une place privilégié : c'est lui la "tête" (cf. le chapitre 5, section 1.5 et la figure 5.5) d'un syntagme propositionnel. On considère généralement qu'il se comporte comme un *prédicat*, c'est-à-dire :

– c'est une *fonction* qui attend des *arguments* : le nombre d'arguments attendus par un verbe donné est son *arité*. Par exemple, un verbe comme "dormir" est d'arité 1 car le seul argument obligatoire qu'il attend est l'identité de l'individu qui réalise l'action de dormir. Les verbes "regarder", "aimer", etc. sont d'arité

2, car ils désignent une *relation* entre deux individus, tandis qu'un verbe comme "donner" est d'arité 3 ("qui" donne "quoi" et " à qui").
- on peut associer une *valeur de vérité* à un prédicat d'arité n auquel on spécifie n arguments.

Les arguments associés à un verbe sont parfois désignés comme des *rôles*. Mais il faut distinguer les *rôles syntaxiques* des *rôles sémantiques* :
- les rôles syntaxiques les plus courants sont : sujet, objet direct, objet indirect
- les rôles sémantiques (ou "thématiques") usuels sont : agent, patient, bénéficiaire...

Dans les constructions à la forme passive, comme dans "la souris est mangée par le chat", le sujet syntaxique de la phrase ("souris") ne coïncide pas avec l'agent de l'action (ici "le chat"). Il n'existe donc pas de correspondance absolue entre ces divers rôles.

Les langues naturelles emploient différentes stratégies pour caractériser le rôle des arguments d'un prédicat. En français ou en anglais, l'ordre des mots est prépondérant, et le verbe s'accorde en personne et en nombre avec son sujet syntaxique. D'autres langues utilisent un système de *cas*, se traduisant pas des *déclinaisons* : certains morphèmes lexicaux subissent ainsi des variations (par le biais de morphèmes grammaticaux) qui désignent leur rôle thématique. Nous en avons encore quelques traces en français : ce qui distingue "je le donne" de "je lui donne" est la déclinaison sous laquelle apparaît le pronom "le". Dans les langues à déclinaison, l'ordre des mots est forcément plus libre que dans les autres, et la modélisation à base de grammaires formelles (cf. chapitre 5, section 2.4) plus difficile.

En dehors des arguments "obligatoires" des verbes, gravitent un certains nombre de "modifieurs" plus ou moins facultatifs : ils servent à donner des précisions temporelles ("hier, à midi..."), géographiques ("place de la concorde, à Paris..."), d'instrument ("avec un couteau") ou encore de manière ("avec rage")... Le recensement exhaustif de tous ces nouveaux rôles sémantiques n'est pas facile, et il existe diverses théories pour en rendre compte. D'un point de vue syntaxique, on constate aussi une grande variabilité dans la façon de les exprimer : le plus souvent, ce sont des adverbes ou des groupes prépositionnels qui jouent ce rôle, mais de simples groupes nominaux peuvent suffire : seules des connaissances de sémantique lexicale assez poussées peuvent expliquer que dans la phrase "Le matin chante l'oiseau", "le matin" n'est pas le sujet ni l'agent du verbe, mais un groupe exprimant le temps de l'action.

1.3 Théorie des modèles

La théorie des modèles est une des branches de la logique. On peut la voir comme une manière d'appliquer la théorie mathématique des ensembles à la modélisation du sens. Elle définit un "monde de sens" servant de référence pour évaluer la valeur de vérité de propositions. Nous n'en présentons ici qu'une version intuitive très simplifiée, inspirée de la "sémantique de Montague", du nom du logicien (déjà cité) qui l'a inventée.

Pour construire notre "monde de sens", nous n'aurons besoin que de deux briques de base :
- la notion de "valeur de vérité" qu'on ramène au choix entre les nombres 0

(pour "faux") et 1 (pour "vrai") ;
- la notion d'"entité" qui est la seule "espèce vivante" de notre univers ! Il faut imaginer un espace où tout ce qui peuple notre monde réel est ramené à de telles "entités". Il y a des entités relativement "naturelles", comme celles identifiant chaque individu humain ou animal particulier. On peut même associer aux "entités nommées" qui ont un "nom propre" une constante donnant la *référence* (cf. chapitre 6, 1.2) de cette entité dans le modèle. Mais il faut aussi imaginer autant d'entités que d'éléments naturels, d'objets ou de choses dont on "peut dire" quelque chose...

L'ensemble (au sens mathématique du terme) de toutes les entités constitue notre *domaine*, et est noté D. On définit maintenant dans cet espace un *prédicat d'arité* n comme étant une fonction de D^n vers $\{0, 1\}$, autrement une fonction qui attend comme arguments n *entités* et qui donne comme résultat une *valeur de vérité*. Par convention, on notera toujours par la suite en indice l'arité d'un prédicat : un prédicat d'arité n sera par exemple noté P_n.

Une fonction à valeurs dans $\{0, 1\}$ est aussi appelée en mathématiques une *fonction caractéristique*. Elle identifie en effet un sous-espace particulier. Ainsi, un prédicat P_1 d'arité 1 est une fonction de D dans $\{0, 1\}$. On peut dire aussi qu'elle *trie* les éléments de D en deux sous-ensembles : ceux à qui P_1 donne la valeur 0 et ceux à qui P_1 donne la valeur 1. Evidemment, ce sont surtout ces derniers qui nous intéressent. P_1 est ainsi complètement défini par l'ensemble des éléments de D auquel il attribue la valeur 1. Par exemple, nous avons vu dans la section précédente que le verbe "dormir" est d'arité 1. On peut le traduire dans notre espace par le prédicat $dormir_1$, qui se représente par un sous-ensemble de D : le sous-ensemble des "entités qui dorment". Mais cette notion de prédicat ne s'applique pas exclusivement aux verbes. Beaucoup de "noms communs" ou d'adjectifs se comportent sémantiquement comme des prédicats à un argument : le prédicat $chat_1$ représente l'ensemble des entités qui sont des chats, $noir_1$ l'ensemble des entités qui sont de couleur noire, etc.

Les prédicats à deux arguments, eux, caractérisent *un ensemble de couples d'entités*. Par exemple, le prédicat $regarder_2$ est vrai chaque fois qu'on lui fournit en arguments deux entités telles que la première regarde la deuxième. On peut le représenter comme un ensemble de *liens entre entités*. De manière générale, les prédicats d'arité n se représentent comme des ensembles de n-uplets (c'est-à-dire de listes de n éléments) d'entités.

Supposons donc que l'on veuille "interpréter", c'est-à-dire traduire en termes d'entités et de fonctions, le sens des propositions suivantes :"Minou est un chat", "Minou ne dort pas", "un chat dort", "tous les chats sont des félins", "Marie est un humain", "Marie dort" et "Minou regarde Marie". Le domaine nécessaire à toute interprétation de ces phrases comporte au minimum les entités "Minou" et "Marie". Nous avons aussi besoin de quatre prédicats à un argument : $chat_1$, $dort_1$, $felin_1$ et $humain_1$, et d'un prédicat à deux arguments : $regarde_2$. Une interprétation fixe une valeur de vérité à toute application d'un prédicat sur les éléments du domaine.

Un modèle pour un ensemble de propositions est *une interprétation qui rend vraie chacune d'elle*. Dans notre exemple, construire un tel modèle impose par exemple que Minou fasse partie de l'ensemble des "félins". Mais d'autres choix sont laissés

libres : par exemple, rien n'indique dans les propositions citées que les humains ne peuvent pas être des félins. Si nous respectons tout de même cette contrainte implicite, nous devons ajouter une entité anonyme qui aura à la fois la propriété d'être un chat et de dormir (puisque ce ne peut être ni Minou ni Marie...). On peut dessiner un "modèle" ensembliste pour nos phrases à la manière de la figure 7.3, où l'ensemble global dessine les contours de D.

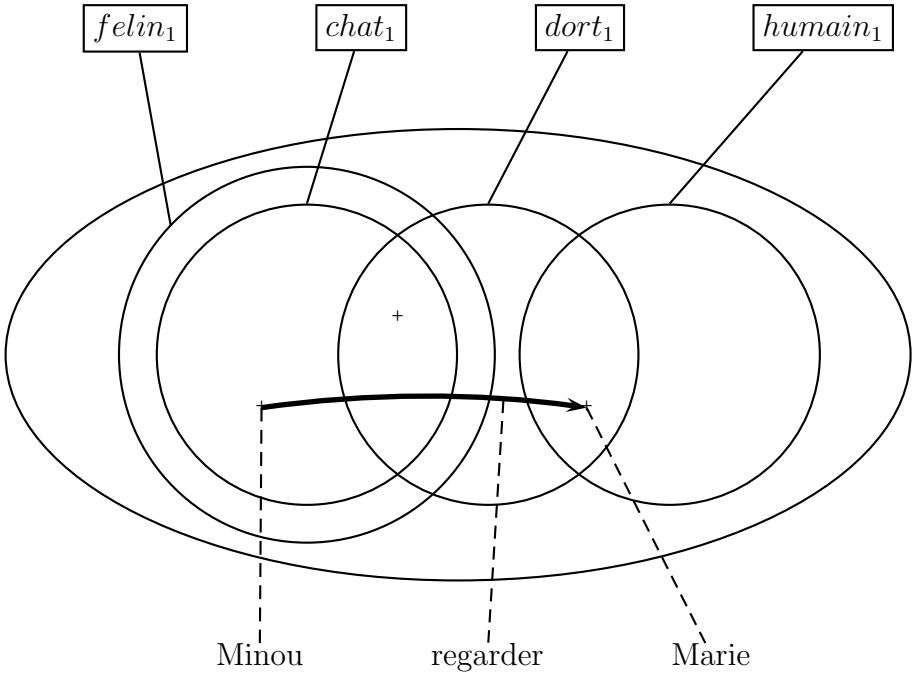


FIG. 7.3 – représentation graphique d'un modèle

Par ce schéma, on mesure bien que les termes linguistiques s'interprètent comme des *catégories* (cf. chapitre 6, section 1.1) et qu'il peut exister des relations hiérarchiques entre catégories (cf. chapitre 6, section 1.6) : le fait que "chat" est un cas particulier de "félin" se traduit par une *inclusion entre ensembles*. Un modèle décrit donc de manière abstraite et rigoureuse un "état du monde" où certaines propositions sont vraies. Il permet aussi d'*évaluer* la valeur de vérité d'une proposition nouvelle. Par exemple, dans le modèle précédent, la proposition "Minou est un félin" est vraie mais "Minou est un humain" est fausse.

Il existe de très nombreuses extensions de ce système, finalement bien rudimentaire pour traduire tout ce que le langage peut signifier. Par exemple, la situation dans le temps d'une proposition nécessite des ajustements. Certains auteurs imaginent l'échelle temporelle comme une règle graduée (un ensemble d'"indices" ordonnés t_1, t_2, t_3,...) : à chaque indice, il faut redéfinir complètement l'interprétation, c'est-à-dire la valeur de chaque fonction pour chaque entité. "Marie dort" peut très bien être vrai à un certain moment t_i, et faux à l'instant d'après t_{i+1}... En généralisant ce principe, on en est même venu à envisager un ensemble potentiellement infini de "mondes possibles" pour pouvoir interpréter correctement des propositions comme "il est possible que Marie dorme" (autrement dit : il existe au moins parmi l'ensemble des "mondes possibles" un monde où "Marie dort" est vrai). Mais détailler

de telles subtilités nous emmènerait trop loin.

Nous reviendrons dans la partie "modélisation informatique" sur les formalismes logiques à l'origine de la notion de modèle. L'important ici est ce que nous suggèrent les logiciens : *comprendre une proposition, cela signifie être capable d'en créer un modèle*. Ce programme est pris très au sérieux par certains psychologues, qui estiment que les humains construisent des "modèles mentaux" qui les aident à raisonner. L'idée n'est pas acceptée par tout le monde, mais elle est séduisante...

2 Modélisation informatique

La "théorie de modèles" dont nous venons de parler pourrait bien sûr être considérée comme une "modélisation informatique". Nous verrons en effet tout de suite qu'elle s'inscrit dans le cadre plus général de la *logique*, ici envisagée comme un *langage formel de représentation des connaissances*. L'autre famille de modèles évoquée ci-dessous, à base de *graphes*, entretient d'ailleurs aussi avec elle des liens étroits.

2.1 Les origines de la logique

La logique, en tant que "science du raisonnement juste" remonte à Aristote et sa mémoire a traversé les siècles : les étudiants du Moyen-Age apprenaient par cœur les 14 "figures de syllogismes" identifiées par ce dernier. La plus célèbre de ces figures permet de déduire, à partir de propositions de la forme "tout A est B" (par exemple "tout homme est mortel") et "C est A" (par exemple "Socrate est un homme") qu'on a nécessairement "C est B" ("Socrate est mortel").

Ramener le raisonnement à un "calcul universel" était un rêve que Leibniz, grand philosophe du XVIIIème siècle, a aussi caressé. Mais ce n'est qu'au XIXème siècle qu'on a vraiment commencé à "mathématiser" la capacité de déduction. On doit à Georges Boole (1815-1864) un progrès significatif en la matière : les opérateurs "booléens" lui doivent leur nom. Nous évoquons brièvement ici la "logique propositionnelle" qui est directement issue de ses travaux.

La logique propositionnelle est un langage formel dont les "formules bien formées" (syntaxiquement correctes) sont bâties à partir d'un ensemble potentiellement infini de "variables propositionnelles" qu'on note traditionnellement p, q, r, etc. Chacune de ces variables peut prendre soit la valeur "faux" (ou 0) soit la valeur "vraie" (ou 1). Les formules sont construites de la façon suivante :

- toute variable propositionnelle constitue en elle-même une formule bien formée.
- si p est une formule bien formée, alors $\neg p$ (prononcer "not p") est une formule bien formée.
- si p et q sont des formules bien formées, alors $p \vee q$ (c'est-à-dire "p ou q"), $p \wedge q$ ("p et q"), $p \longrightarrow q$ ("p implique q") sont aussi des formules bien formées.

Les symboles $\neg$, $\vee$, $\wedge$ et $\longrightarrow$ sont les "opérateurs booléens" auxquels nous faisions précédemment allusion. $\neg$ est un peu à part : il s'applique sur un seul argument, et a pour effet d'*inverser* les valeurs de vérité. Si p est vrai, alors $\neg p$ est faux, et inversement. Les trois autres opérateurs combinent deux arguments, à la manière des opérateurs mathématiques usuels (+, -, *, /). Quand plusieurs d'entre eux sont

p	q	$p \vee q$
0	0	0
0	1	1
1	0	1
1	1	1

p	q	$p \wedge q$
0	0	0
0	1	0
1	0	0
1	1	1

FIG. 7.4 – tables de vérité

utilisés pour construire une formule complexe, on doit utiliser des parenthèses "(" et ")" pour expliciter l'ordre d'application. La manière dont ils agissent sur les valeurs de vérités des propositions qu'ils combinent est donnée dans les "tables de vérité" de la figure 7.4

Par ailleurs, $p \longrightarrow q$ est équivalent à (c'est-à-dire a exactement les mêmes valeurs de vérité que) $\neg p \vee q$. Cela signifie en particulier que, d'une proposition fausse, on peut déduire n'importe quoi. Par exemple, la proposition "si la France est une monarchie alors le pape est une femme" est vraie car si on la représente par la formule $m \longrightarrow p$ où m signifie "la France est une monarchie" et p "le pape est une femme", alors comme m doit recevoir dans notre monde actuel la valeur de vérité "fausse", $\neg m$ est vrai et l'ensemble de la formule $\neg m \vee p$ est aussi vraie !

En logique propositionnelle, les affectations de variables propositionnelles tiennent lieu d'"interprétation", et un modèle est aussi une affectation qui rend une formule vraie. Par exemple, un modèle possible de la formule $(p \wedge q) \vee r$ est l'interprétation : $p = 0$, $q = 1$ et $r = 1$. Une formule pour laquelle toutes les affectations possibles des variables sont des modèles est appelée une *tautologie*. Par exemple : $p \vee \neg p$ est une tautologie. Une formule qu'il est impossible de rendre vraie, comme $p \wedge \neg p$ est une *contradiction*. Une formule pour laquelle il existe au moins un modèle est *satisfable*, une pour laquelle il existe au moins une affectation de variables qui la rende fausse est *falsifiable*. Une tautologie est un cas particulier de formule satisfiable, une contradiction un cas particulier de formule falsifiable. Une formule qui n'est ni l'une ni l'autre comme $(p \wedge q) \vee r$ est à la fois satisfiable et falsifiable.

Utiliser la logique propositionnelle pour "représenter des connaissances" est un peu malaisé et grossier, puisqu'un symbole doit à lui tout seul "représenter" toute une proposition. Par exemple, pour signifier "une porte est ouverte ou fermée", on peut définir la variable o qui signifie "une porte est ouverte". $\neg o$ symbolise alors "il est faux qu'une porte est ouverte" -autrement dit "une porte est fermée"- et la proposition initiale se représente donc par : $o \vee \neg o$, qui, on vient de le voir, est une tautologie. Pour "tout homme est mortel", on peut choisir une variable h signifiant "être un homme" et une variable m pour "être mortel" : la proposition est alors représentée par la formule $h \longrightarrow m$.

Le point fort de la logique propositionnelle n'est pas dans l'expressivité de ses formules : il est dans sa capacité à *formaliser des raisonnements*. Par exemple, une règle de déduction simple qui y est applicable est connue sous le nom de "modus ponens" (ou règle de détachement) : elle stipule que si on a simultanément deux propositions vraies de la forme $p \longrightarrow q$ et p respectivement, alors q est nécessairement vraie (ce que l'on vérifie aisément avec les tables de vérité) et on peut donc la

"déduire". C'est le pendant -un peu approximatif tout de même- du raisonnement aristotélicien : si $h \longrightarrow m$ ("chaque homme est mortel") et h ("être un homme") sont vrais simultanément, alors on en déduit que m ("être mortel") est également vrai.

Tout l'effort des logiciens consiste à définir des règles de ce genre qui, en ne prenant en compte que la *forme* des expressions manipulées, garantissent que certaines propriétés portant sur leurs valeurs de vérité soient satisfaites.

2.2 Logique des prédicats du 1er ordre

En logique propositionnelle, l'"atome" est la proposition, elle est indécomposable. Il est donc impossible de rendre compte de points communs entre propositions, comme le fait que les expressions "Marie dort" et "Minou ne dort pas" utilisent le même prédicat, ou que "Minou ne dort pas" et "Minou regarde Marie" partagent une même entité. La logique des prédicats du 1er ordre, inventée par Frege à la fin du XIXème siècle, remédie à cette "pauvreté expressive" de la logique propositionnelle, en opérant une distinction fondamentale entre "termes" et "prédicats", et en définissant une proposition comme une *combinaison* de ces éléments.

Pour définir le langage de la logique des prédicats du 1er ordre (abrégée par la suite en LPO), on a besoin de disposer d'un stock potentiellement infini de *constantes*, et d'un stock potentiellement infini de *variables*. Ces dernières serviront non pas à exprimer des propositions, comme précédemment, mais plutôt à *désigner des entités*. On suppose en outre pouvoir utiliser à volonté un ensemble de *prédicats*, chaque prédicat ayant sa propre *arité*. Dans sa forme la plus générale, la LPO permet aussi d'employer des *fonctions* qui, à partir d'un certain nombre d'arguments, donnent comme résultat un terme (une entité), mais nous laissons volontairement de côté cette possibilité ici, parce qu'elle est peu utilisée pour traduire des propositions du langage naturel.

Les formules de la LPO respectent les règles de construction suivantes :

- toute variable et toute constante est un *terme* ;
- si P_n est un prédicat d'arité n et $t_1,\ t_2, ...,\ t_n$ sont n termes, alors $P_n(t_1,\ t_2, ...,\ t_n)$ est une formule bien formée ;
- si x est une variable et Φ une formule bien formée, alors $\exists x \Phi$ (qui se lit "il existe au moins un x tel que Φ est vraie") et $\forall x \Phi$ ("pour tous les x, Φ est vraie") sont des formules bien formées. Le terme de "1er ordre" associé à cette logique provient du fait que les "quantificateurs" $\exists$ et $\forall$ sont nécessairement suivis d'une *variable* et pas, par exemple, d'un prédicat.
- si Φ et Ψ sont des formules bien formées alors $\neg\Phi$, $\Phi \vee \Psi$, $\Phi \wedge \Psi$ et $\Phi \longrightarrow \Psi$ sont aussi des formules bien formées où les opérateurs booléens ont le même sens que précédemment.

Avec ce langage, on peut traduire en formules toutes les propositions qui ont donné lieu au modèle de la section 1.3. Les constantes nécessaires sont "Minou" et "Marie", les prédicats sont ceux introduits dans cette section, et les formules bien formées correspondantes sont :

- "Minou est un chat" : $chat_1(Minou)$, "Marie est un humain" : $humain_1(Marie)$;
- "Marie dort" : $dort_1(Marie)$, "Minou ne dort pas" : $\neg dort_1(Minou)$;

– "Minou regarde Marie" : $regarde_2(Minou, Marie)$;
– "un chat dort" : $\exists x[chat_1(x) \wedge dort_1(x)]$;
– "tous les chats sont des félins" : $\forall x[chat_1(x) \longrightarrow felin_1(x)]$.

La LPO est donc bien un langage de "représentation des connaissances". Il permet d'exprimer des faits et des "règles" ("tous les chats sont des félins" est plutôt une règle générale qu'un fait instancié) de manière rigoureuse. Là où les langues naturelles autorisent des ambiguïtés, la logique, elle, impose des choix. Par exemple, la phrase "Jean qui regarde Paul que Marie aime dort" peut s'interpréter de deux façons différentes, suivant que Marie aime Jean ou Paul. Suivant le choix fait, on la traduira donc soit pas : $regarde_2(Jean, Paul) \wedge aime_2(Marie, Jean) \wedge dort_1(Jean)$, soit par : $regarde_2(Jean, Paul) \wedge aime_2(Marie, Paul) \wedge dort_1(Jean)$. Une autre ambiguïté classique tient à l'ordre des quantificateurs $\exists$ et $\forall$ quand il sont présents tous les deux dans une même formule. Ainsi, pour traduire une proposition comme "chaque étudiant apprend une langue étrangère", on a le choix entre :

– $\forall x[etudiant_1(x) \longrightarrow \exists y[langue\_etrangere_1(y) \wedge apprend_2(x, y)]]$
qui signifie que pour chaque étudiant, il existe une certaine langue étrangère (éventuellement distincte pour chacun) que cet étudiant apprend ;
– $\exists y[langue\_etrangere_1(y) \wedge \forall x[etudiant_1(x) \longrightarrow apprend_2(x, y)]]$
qui identifie une certaine langue étrangère particulière que tous les étudiants apprennent. Ce sens prédomine si on met la phrase initiale au passif en disant : "une langue étrangère est apprise par tous les étudiants".

Traduire en LPO une proposition impose donc de la "désambiguiser" pour exprimer son sens de manière précise. Les formules de la LPO peuvent être traduites en termes d'ensembles et de fonctions à l'aide d'une *interprétation*. Une interprétation associe une entité du domaine à chaque constante et à chaque variable du langage, et une fonction caractéristique (c'est-à-dire un ensemble) à chaque prédicat. Comme nous l'avons déjà vu, un modèle est une interprétation qui rend une formule ou un ensemble de formules vraie(s). Comme en logique propositionnelle, on peut définir les notions suivantes :

– une *tautologie* est une formule telle que toute interprétation est nécessairement un modèle. C'est le cas de : $\forall x[P_1(x) \vee \neg P_1(x)]$ où P_1 est un prédicat quelconque d'arité 1.
– une *contradiction* est une formule qu'aucune interprétation ne peut rendre vraie, comme $\exists x[P_1(x) \wedge \neg P_1(x)]$.
– une formule est *satisfiable* s'il existe au moins une interprétation qui la rend vraie (c'est-à-dire qui en est un modèle). Les tautologies sont bien sûr satisfiables. Toutes les formules traduisant nos propositions de la section 1.3 étaient satisfiables, puisqu'on a pu définir pour elles le modèle de la figure 7.3.
– une formule est *falsifiable* s'il existe au moins une interprétation qui la rend fausse (c'est-à-dire qui n'est pas un modèle). Les contradictions sont falsifiables. Les formules de notre exemples étaient aussi falsifiables : il est facile d'imaginer une interprétation dans laquelle $chat_1(Minou)$ est faux...

Comme la logique propositionnelle, la LPO permet aussi de rendre compte de *déductions*. Par exemple, il est évident que dans tout modèle des formules précédentes, "Minou est un félin", c'est-à-dire $felin_1(Minou)$ est aussi vrai. De même, en LPO, on traduirait "tout homme est mortel" par $\forall x[homme_1(x) \longrightarrow mortel_1(x)]$ et "So-

crate est un homme" par $homme_1(Socrate)$. Tout modèle rendant vraies ces deux formules rend vraie aussi la formule $mortel_1(Socrate)$. Pour expliciter ce genre de déductions sans avoir pour autant à contruire "toutes les interprétations possibles", les logiciens ont identifié des méthodes qui ne se fondent, encore une fois, que sur la "forme" des expressions logiques. Il existe ainsi une sorte de "modus ponens généralisé" autorisant à déduire $mortel_1(Socrate)$ des deux formules précédentes. Le langage de programmation Prolog (abréviation de "Programmation logique") est fondé sur la LPO. Les déductions de ce genre y sont automatisées.

Enfin, il existe de nombreuses variantes et extensions de la LPO, qui s'interprètent à l'aide des modèles "étendus" du type de ceux évoqués en fin de section 1.3. Elles s'appellent par exemple "logique temporelle" ou "logique modale". Mais plus on gagne en expressivité, plus on perd en efficacité de raisonnement.

2.3 Exemple récapitulatif

Les systèmes syntaxico-sémantiques qui mettent en œuvre le principe de compositionnalité rendent totalement explicite la succession des traitements qui, partant d'une phrase en langue naturelle, aboutissent à sa représentation sémantique. C'est le cas de celui de Montague, dont nous nous inspirons ici. Montrons par exemple comment produire automatiquement les formules logiques qui traduisent "Marie dort" et "un chat dort" à partir de l'analyse de ces phrases.

Le formalisme syntaxique employé ici est celui des *grammaires catégorielles*, défini par :

- un *vocabulaire fini* : $\Sigma = \{$Marie, dort, chat, un$\}$
- un ensemble $\mathcal{B}$ de *catégories de base* parmi lesquelles l'*axiome* S : ici, nous prendrons $\mathcal{B} = \{S, T, NC\}$ (T pour "terme", NC pour "nom commun")
- l'ensemble de *toutes les catégories possibles* est l'ensemble des *fractions orientées* construites à partir des catégories de base à l'aide des "opérateurs" / et \ comme : $T\backslash S$, $(S/(T\backslash S))/CN$...

Une *grammaire catégorielle* est alors une association entre mots et catégories, comme dans le tableau ci-dessous :

$G =$

mot	catégorie
Marie	T
dort	$T\backslash S$
chat	CN
un	$(S/(T\backslash S))/CN$

Les règles syntaxiques se limitent, elles, à *deux schémas* applicables pour toutes catégories A et B comme suit :

- "Forward Application" $FA : A/B\ B \longrightarrow A$
- "Backward Application" $BA : B\ B\backslash A \longrightarrow A$

Ces schémas sont très similaires à des "réductions de fraction" mathématiques. Dans ce formalisme, une suite de mots est considérée comme syntaxiquement correcte si on peut lui associer une suite de catégories qui, en appliquant ces schémas des feuilles vers la racine de l'arbre, *se réduit à* S. Les grammaires catégorielles sont un exemple typique de formalisme lexicalisé : ce sont les catégories associées aux mots

qui déterminent comment ils pourront se combiner les uns avec les autres. Elles ont la même expressivité que les grammaires algébriques (à la chaîne vide ϵ près).

"Marie dort" et "un chat dort" font partie du langage de notre grammaire, comme en attestent les arbres de la figure 7.5 :

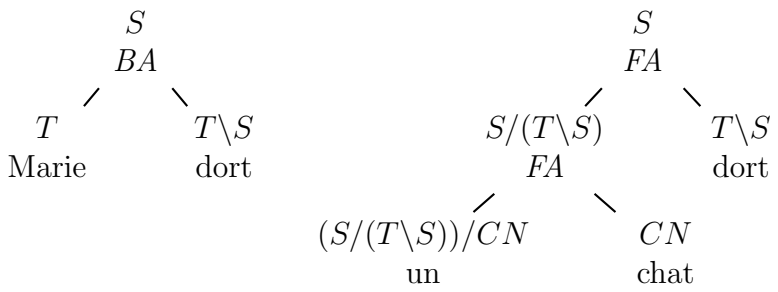


FIG. 7.5 – arbres d'analyse syntaxique

Pour obtenir la traduction sémantique de ces phrases, il faut commencer par préciser la sémantique lexicale des mots qui y figurent. Comme attendu, "Marie" doit être traduite par une constante logique notée $Marie$, "dort" et "chat" par des prédicats à un argument notés respectivement $chat_1$ et $dort_1$. Nous verrons plus loin quelle traduction associer au mot grammatical "un". Pour mettre en œuvre le principe de compositionnalité, il faut construire un "arbre sémantique", en partant des feuilles et en "remontant" à la racine, de la façon suivante :

- l'arbre sémantique est isomorphe à (c'est-à-dire a exactement la même structure que) l'arbre syntaxique ;
- les feuilles de l'arbre sémantique sont les traductions logiques des mots situés à la même place qu'elles dans l'arbre syntaxique ;
- si une règle *FA* est employée à un nœud de l'arbre syntaxique alors, au même emplacement dans l'arbre sémantique, il faut appliquer la formule se trouvant dans le fils gauche de ce nœud comme une fonction opérant sur celle se trouvant dans son fils droit ;
- si une règle *BA* est employée à un nœud de l'arbre syntaxique alors, au même emplacement dans l'arbre sémantique, il faut appliquer la formule se trouvant dans le fils droit de ce nœud comme une fonction opérant sur celle se trouvant dans son fils gauche ;
- la racine de l'arbre sémantique contient alors la formule exprimant le sens de la phrase initiale.

La figure 7.6 reprend en l'instanciant sur la phrase "Marie dort" la partie basse du schéma du principe de compositionnalité de la figure 7.1. La racine de l'arbre sémantique en bas à droite du schéma, obtenu en appliquant les règles précédentes, est bien la traduction logique attendue $dort_1(Marie)$ de la phrase "Marie dort".

Pour finir, expliquons brièvement comment obtenir automatiquement la traduction de la phrase "un chat dort", à savoir $\exists x[chat_1(x) \wedge dort_1(x)]$, à partir de son arbre d'analyse syntaxique, à droite de la figure 7.5. Notons que, dans cet arbre, nous avons fait le choix d'associer une catégorie syntaxique complexe au mot "un", de sorte qu'il "attende" sur sa droite deux arguments : d'abord un de catégorie

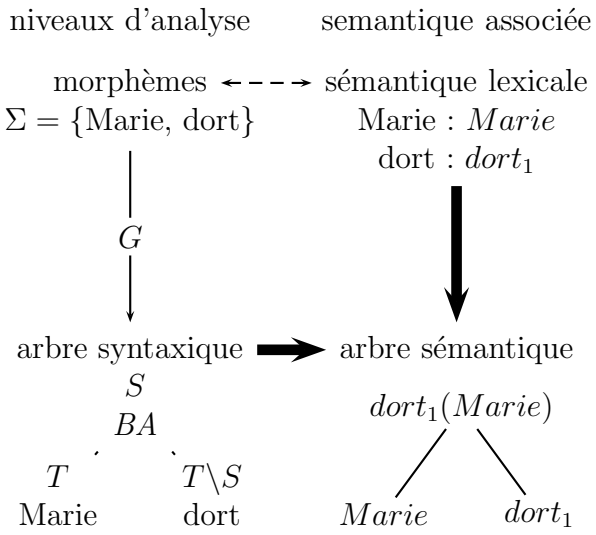


FIG. 7.6 – le principe de compositionnalité instancié

NC (un nom commun), puis un autre de catégorie $T\backslash S$ (un verbe intransitif). La justification de ce choix est sémantique.

En effet, connaissant déjà les traductions de "chat" et de "dort", ce qu'il manque pour obtenir la formule finale est quelque chose comme : $\exists x[P(x) \wedge Q(x)]$, où P pourrait être *remplacé* par $chat_1$ et Q par $dort_1$. C'est exactement une formule de ce type que Montague propose d'associer à "un". En faisant en sorte (par un mécanisme que nous ne donnons pas ici) que l'"application" de cette formule successivement à $chat_1$ puis à $dort_1$ entraîne une *substitution* de ses variables P et Q par ces prédicats, on obtient bien la formule souhaitée. Non seulement le principe de compositionnalité est alors complètement respecté, mais en plus l'épineux problème de la nature sémantique des mots grammaticaux reçoit une solution élégante : les mots grammaticaux doivent être traduits par des formules logiques "grammaticales", c'est-à-dire écrites à l'aide de quantificateurs et de variables, mais où les prédicats restent abstraits, prêts à être *substitués* par ceux qui traduisent les mots lexicaux présents dans la phrase.

Le système de Montague est extrêmement séduisant et a fait l'objet de nombreux travaux, pour l'étendre et l'affiner tant d'un point de vue syntaxique que sémantique. C'est le système de référence de tous ceux qui défendent une application stricte du principe de compositionnalité.

2.4 Autres modèles

Pour représenter la sémantique d'une proposition sans faire appel à des formules logiques, certains auteurs préfèrent utiliser des *graphes* ou des *schémas*. La théorie des *graphes conceptuels* a ainsi été proposée dans les années 80 par John Sowa. Le sens d'une phrase comme "le petit chat mange une souris" y est représenté par le graphe de la figure 7.7.

En fait, on peut traduire tout le formalisme des graphes conceptuels en formules

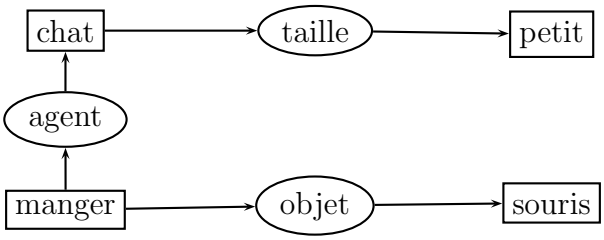


FIG. 7.7 – un graphe conceptuel

logiques équivalentes. Mais les graphes ont l'avantage d'être plus lisibles et facilement interprétables par des humains. De même, la décomposition en primitives des verbes d'action, évoquée en section 1.4, a été utilisée par Schank pour donner une représentation graphique aux propositions complètes. Par exemple, dans son système, "Pierre dit à Paul que Jean a donné un livre à Marie" donne lieu au schéma de la figure 7.8.

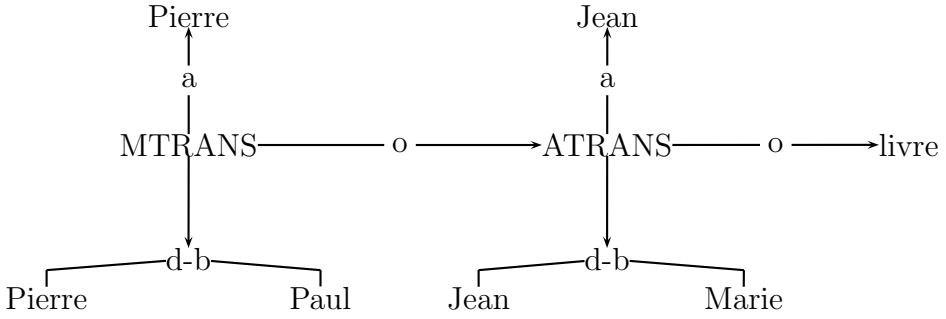


FIG. 7.8 – un graphe "à la Schank"

Dans ce schéma, outre les primitives MTRANS (qui signifie "transfert d'information") et ATRANS ("transfert de propriété"), on utilise les étiquettes "a" pour "agent", "o" pour "objet" et "d-b" pour un couple "donnateur-bénéficiaire". Schank définit en fait un ensemble de *rôles conceptuels* servant à relier entre elles les entités intervenant dans son schéma.

Les efforts les plus récents pour étendre ce type de représentations ont porté sur la sémantique des *discours*, où les phénomènes de *reprises anaphoriques* (c'est-à-dire de référence, dans une phrase, à des entités introduites dans des phrases précédentes) obligent à envisager des modèles plus raffinés. La *Théorie des Représentation du Discours* (DRT en anglais) est née précisément pour répondre à cet objectif. Une DRS (*Discourse Representation Structure*) est une représentation fondée sur la LPO, mais où les "référents du discours" sont isolés en "en-tête" de la structure contenant les formules elles-mêmes, et ainsi rendus accessibles à la suite du discours. La DRS modélisant "Minou regarde Marie. Elle dort" est donnée dans la figure 7.9.

Cette théorie a permis de résoudre certains problèmes délicats laissés par la LPO. Le plus célèbre d'entre eux était dû aux "donkey sentences", dont l'exemple prototypique est la phrase "chaque fermier qui possède un âne le bat". Pour traduire

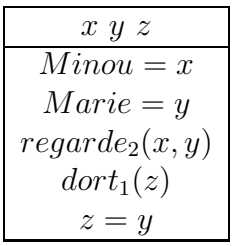


FIG. 7.9 – une DRS très simple

en logique cette phrase, on peut proposer la formule suivante :
$\forall x[fermier_1(x) \wedge \exists y[ane_1(y) \wedge possede_2(x,y)] \longrightarrow battre_2(x,y)]$
Cette formule n'est malheureusement pas correcte parce que la variable y présente dans l'expression $battre_2(x,y)$ n'a en fait aucune raison d'être la même que celle introduite par le quantificateur $\exists$, dont la "portée" s'arrête aux frontières du crochet qui le suit. En DRT, les quantificateurs ne sont plus nécessaires (ils sont remplacés par des "règles d'accessibilité" entre en-têtes et sous-structures) et, cette phrase ne pose pas de problème : elle peut se traduire par la DRS de la figure 7.10.

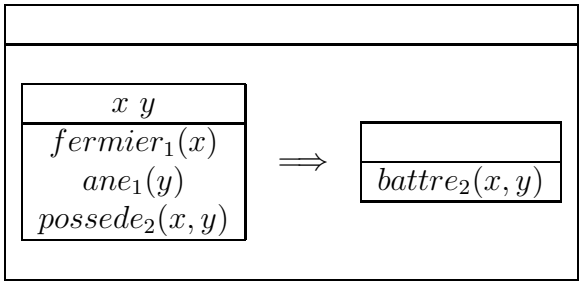


FIG. 7.10 – la DRS qui résout les "donkey sentences"

Enfin, la SDRT est une extension récente de la DRT où les liens rhétoriques entre phrases sont analysés de façon encore plus fine.

2.5 Problèmes

Ce tour d'horizon ne doit pas laisser penser que la question du sens est désormais totalement résolue. Aussi élaborées et séduisantes soit-elles, les théories abordées brièvement dans ce chapitre sont loin de faire l'unanmité, et laissent derrière elles de nombreuses zones d'ombre.

Tout d'abord, le sens commun des mots "et" et "ou" ne coïncide pas vraiment avec celui des connecteurs logiques correspondants. Ils sont beaucoup plus ambivalents que ne le laissent penser leur table de vérité : si "j'aime lire ou aller au cinéma", cela signifie que j'aime les deux à la fois, mais si on me demande "fromage ou dessert", je n'ai cette fois pas droit de choisir les deux. Ces conjonctions peuvent aussi servir à exprimer des rapports de nature temporelle ou causale ("le verre est tombé par terre et s'est cassé")...

Les modèles logiques requièrent par ailleurs de représenter par des "entités" tout ce qui "existe" dans le monde. Mais le mode d'existence des individus, des mythes (les licornes), des concepts (la démocratie) n'est pas tout à fait comparable. Est-ce que représenter le sens du groupe nominal "une bonne idée" par $\exists x[idee(x) \wedge bonne(x)]$ est une si bonne idée que cela ?

Les entités nommées sont aussi beaucoup plus problématiques que nous ne l'avons laissé croire jusqu'à présent. "Georges W. Bush" désigne un individu historique précis mais "le président des Etats-Unis" désigne quelqu'un de différent suivant le temps de référence. D. Kayser montre qu'un terme comme "Prix Goncourt" est très polysémique :

- dans "le Prix Goncourt a été attribué à...", c'est une récompense ;
- dans "X a versé son Prix Goncourt à la Croix Rouge", c'est un montant financier ;
- dans "le Prix Goncourt a été félicité par le Président", c'est une personne ;
- dans "le Prix Goncourt a admis un nouveau membre", c'est un jury collectif ;
- dans "peux-tu m'acheter le Prix Goncourt", c'est un objet (un livre) ;
- dans "depuis son Prix Goncourt, il est devenu arrogant", c'est un événement ;
- dans "le Prix Goncourt pervertit la vie littéraire", c'est l'existence de cet événement...

Les pluriels et les "quantificateurs généralisés" (comme : "une partie de(s)", "la plupart de(s)", "beaucoup de(s)"...) sont aussi difficiles à traduire en formules de la LPO. Et certaines ambiguïtés les concernant ne peuvent être levées qu'en faisant appel à des connaissances extérieures : comment expliquer autrement que quand je vois "trois hommes portant un piano", je ne vois qu'un seul piano, mais quand je vois "trois hommes portant une barbe", je vois plusieurs barbes...

Le langage courant, enfin, est truffé de figures de styles qui sont très mal prises en compte par le genre de sémantique que nous avons développée ici. Les principales sont :

- les métonymies, qui autorisent à prendre la partie pour le tout ou le contenant pour le contenu comme dans "je bois un verre" ou "je vois une voile à l'horizon" ;
- les métaphores qui transposent certaines propriétés d'une catégorie à une autre : "l'homme est un loup pour l'homme"...

Nous réalisons sans nous en rendre compte énormément d'inférences lors de l'interprétation de certaines phrases. Un garçon de café peut dire sans être ridicule "le sandwich au jambon est parti en courant", parce que nous avons décodons la métonymie "sandwich"="acheteur de sandwich" et que nous connaissons les étapes conventionnelles par lesquelles doit se dérouler la transaction d'un sandwich dans la vie quotidienne. De même, nous savons bien qu'en disant "le policier leva la main, la voiture s'arrêta", nous ne parlons pas nécessairement de Superman arrêtant les voitures par sa force, mais d'un simple agent qui agit conformément à l'autorité conférée à son costume, et que si une voiture s'arrête, c'est parce que son conducteur l'a bien voulu... Lesquelles de ces inférences devraient-elles être modélisées ? Jusqu'où ne pas aller trop loin ?

Schank (toujours lui) a par exemple développé dans les années 70 toute une théorie pour rendre compte de ce genre de raccourcis. Il l'a mise en œuvre dans un

système de résumé automatique de dépêches d'agences de presse, qui essayait d'identifier le domaine dont il était question avant de remplir des champs prédéfinis. On dit que pour résumer un article qui titrait "la mort du pape secoue le monde occidental", le programme avait rempli les champs suivants : événement="tremblement de terre", victime="le pape"...

De manière générale, la réduction de la sémantique à des formalismes abstraits a tendance à exclure tout ce qui ne relève pas de la "fonction référentielle" du langage (cf section historique du chapitre 2, en particulier les "fonctions du langage" de Jakobson). C'est aussi la conséquence de notre parti-pris de ramener le sens propositionnel aux valeurs de vérité. L'ironie ou l'humour, par exemple, ont besoin d'autres cadres pour être analysés. Les spécialistes de la pragmatique, c'est-à-dire de l'usage du langage en situations remettent sérieusement en cause ce parti-pris.

Le langage courant véhicule une quantité incroyable d'informations. Toute notre connaissance du monde s'y exprime. Pour le "modéliser" correctement dans la mémoire d'une machine, il faudrait d'abord modéliser le monde dans son ensemble, avec ses lois physiques et son écosystème, sans oublier les conventions sociales et les mondes imaginaires de ses habitants. Le chantier n'est pas prêt d'être achevé.

2.6 Sites Web

Il existe des programmes capables de transformer des phrases en représentations abstraites (formules logiques ou graphes), mais ce sont en général des prototypes au vocabulaire limité. Les sites suivants sont un peu plus généraux.

- un système d'étiquetage de rôles syntaxico-sémantiques :
  l2r.cs.uiuc.edu/∼cogcomp/srl-demo.php
- le site très riche de John Sowa, l'inventeur des graphes conceptuels (en anglais) :
  www.jfsowa.com/ontology/index.htm
- deux ontologies déjà citées au chapitre précédent, mais dont les traductions en formules logiques méritent aussi d'être explorées :
  ontologyportal.org/ et virtual.cvut.cz/kifb/en/toc/229.html
- un projet encyclopédique de représentation des connaissances commencé en 1985 et toujours en développement :
  www.opencyc.org
- un format général d'annotation et de représentation des connaissances :
  framenet.icsi.berkeley.edu/

Chapitre 8

Fouille de textes

Les chapitres précédents relevaient du "traitement automatique du langage" et aussi de l'"ingénierie linguistique". Ils visaient à présenter l'ensemble des concepts et outils élaborés par les linguistes et les informaticiens pour étudier aussi précisément que possible les capacités expressives des langues naturelles. Malheureusement, "en pratique", c'est-à-dire face à de vraies données textuelles, peu de ces outils sont réellement applicables. Souvent, leur état de développement ne les rend tout simplement pas aptes à se confronter à un texte quelconque : il n'existe par exemple aucun analyseur syntaxique du français capable d'analyser toutes les phrases d'un article de journal courant. Mais, même s'il était possible, ce traitement exhaustif ne serait pas toujours souhaitable car il serait trop coûteux en temps de calcul. Quand il s'agit par exemple de rechercher une information précise dans une encyclopédie, il y a de meilleures méthodes à mettre en œuvre que de traduire en formules logiques toutes ses phrases.

Dans ce chapitre, nous allons tenter d'appliquer les outils linguistiques à notre disposition à la "fouille de textes". L'enjeu de ce domaine est la manipulation rapide et efficace de grandes quantités de textes, pour lesquels il n'est pas envisageable de procéder à une analyse syntaxico-sémantique complète. Nous allons dresser un panorama des "tâches" très générales auxquelles elle se confronte. La plupart de ces tâches sont abordées avec des méthodes *numériques* plus que symboliques. Mais la plupart peuvent aussi bénéficier largement de certains outils issus du traitement des langues.

Nous allons donc d'abord passer en revue les "ressources linguistiques" disponibles, et les méthodes employées pour les obtenir. Nous montrerons ensuite que les textes sont des données complexes qui peuvent être codées de manières diverses. Chaque application particulière requerra un codage adapté. Nous allons aussi présenter rapidement le domaine de l'apprentissage automatique parce qu'il est de plus en plus utilisé, soit pour acquérir ou améliorer des ressources, soit pour accomplir certaines tâches de fouille de textes. Ce sont ces tâches qui nous occuperont pour finir. Chacune d'elle mériterait un chapitre, mais cela nous éloignerait trop de l'objectif initial de ce document, et nous nous en tiendrons donc à une présentation succinte

1 Ressources

L'ingénierie linguistique contemporaine est fondée sur la notion de *ressource*. Une ressource est un ensemble de données ou un programme réunissant des informations linguistiquement pertinentes, et utiles dans un cadre applicatif. Elle doit être utilisable "par elle-même", sans être pour cela nécessairement intégrée dans une chaîne de traitements complexes.

En fait, la plupart des programmes pour lesquels nous avons cité des sites Web dans les chapitres précédents (en particulier, en chapitre 4, section 2.4 et en chapitre 5, section 2.9) peuvent être considérés comme des ressources, en particulier :

- les étiqueteurs d'entités nommées (cf. chapitre 4, section 1.2) ;
- les analyseurs morphologiques (cf. chapitre 4, section 1.3) ;
- les lemmatiseurs (cf. chapitre 4, section 1.4) ;
- les étiqueteurs "part of speech" (cf. chapitre 4, section 1.4) ;
- les analyseurs syntaxiques (cf. chapitre 5).

Mais, pour se confronter à des "textes bruts", certains outils apparemment plus "rudimentaires" sont aussi nécessaires, comme :

- les segmenteurs de mots, qui isolent les mots les uns des autres en ajoutant des symboles spéciaux séparateurs entre eux ;
- les segmenteurs de phrases, qui isolent les phrases.

Et tous les programmes capables d'identifier certains groupes de mots particuliers (cf. chapitre 5, section 1.2) comme :

- les "chunks"
- les termes
- les clauses

peuvent aussi rendre de grands services.

Les ressources prennent aussi parfois la forme de *données* plus que de programmes. Parmi celles-ci, citons :

- les listes de "mots vides" ou "antidictionnaires" : un mot est dit "vide" s'il ne contribue pas au sens du texte dont il fait partie. Typiquement, les mots grammaticaux sont généralement vides. Mais certains mots lexicaux peuvent aussi être intégrés à ces listes : noms, adjectifs ou verbes trop généraux, courants ou polysémiques ("faire", "aller", "tenir")...
- les listes d'entités nommées, éventuellement "typées" suivant différents genres ("personnes", "lieux", "organisations", etc.)
- les dictionnaires, avec toutes sortes d'informations possibles associées aux mots : étiquettes "part of speech" bien sûr, mais aussi registre ("soutenu", "familier", "vulgaire"...) dont ils relèvent, ou encore fréquence d'apparition dans une langue donnée etc.
- les thesaurus et les ontologies (cf. chapitre 6, sections 1.6 et 2.2)

Enfin, on considère aussi comme des ressources des collections de textes, ou encore d'enregistrements audio et/ou vidéo, réunis de façon cohérente : enregistrements de conversations téléphoniques, transcriptions écrites de discussions avec des enfants, recueil de textes littéraires, d'articles de journaux, de dépêches de presse, de sites Web, de mails ou de SMS... En la matière, on peut envisager une multitude de données possibles. A condition qu'elle soit *représentative d'un certain usage de la*

langue, on appelle une telle collection un *corpus*.

Certains corpus ont des propriétés particulières. Ils peuvent par exemple être accompagnés d'informations supplémentaires : quand on dispose d'un ensemble d'articles de revues, on peut avoir en plus la *rubrique* (économie, sport, culture...) dont ils relèvent ; s'il s'agit de mails, il peut être intéressant de savoir lesquels, parmi eux, sont des "spams" (courriers indésirables), etc. Ces informations seront précieuses dans certaines applications.

D'autres corpus répondent à des besoins très spécifiques. Par exemple, un *concordancier* est un corpus mettant en évidence les *contextes d'utilisation* d'un certain mot. Ces contextes se présentent en général comme une fenêtre avec un nombre fixe de mots "avant" et "après" l'occurrence du mot en question. Il sert aux *lexicographes*, qui étudient les usages des mots, pour écrire leurs définitions dans un dictionnaire. Certains programmes sont capables de bâtir automatiquement un concordancier, quand on leur fournit un ensemble de textes et un mot cible.

La plupart des ressources précédentes sont *monolingues*, c'est-à-dire spécifiques d'une langue donnée. Quand elles concernent plusieurs langues distinctes, elles sont dîtes *multilingues* : c'est le cas des dictionnaires bilingues, ou encore des *corpus alignés* qui sont des textes disponibles dans plusieurs langues, où les passages qui sont des traductions les uns des autres sont accessibles les uns à partir des autres.

La constitution d'une ressource est une tâche longue et difficile : elle demande souvent des compétences complexes, et presque toujours beaucoup de travail. Des sociétés privées vivent de la vente de certaines de ces ressources : elles sont donc rarement gratuites.

2 Représentations d'un texte

La représentation des données linguistiques n'allant pas au-delà de la taille d'une phrase ne pose pas trop de problèmes : il suffit la plupart du temps de les considérer comme des séquences de symboles prises dans un alphabet fini. Le codage des caractères lui-même peut pourtant faire l'objet de choix assez divers (code ASCII, UTF-8, unicode, etc.) sur lesquels nous ne nous étendrons pas. Mais les *textes* sont des données encore plus complexes et certaines applications les codent dans des structures particulières.

Un texte peut en effet être envisagé en tant que :

- séquence de *caractères* : c'est le cas le plus classique ;
- séquence de *mots* isolés par des *séparateurs de mots* ;
- séquence de *mots étiquetés* (par leur lemme, leur catégorie, des traits sémantiques...), isolés par des séparateurs ;
- séquence de *phrases* isolées par des *séparateurs de phrases* ;
- séquences de *portions de documents* identifiés par des caractères spéciaux (balises HTML par exemple) ;
- séquence d'*arbres* (soit arbres de type HTML, soit arbres d'analyses syntaxiques).

Il existe des corpus dans chacun de ces formats, et des programmes qui permettent de passer de certaines de ces représentations à certaines autres. Certains

documents, en particulier les fichiers HTML, peuvent ainsi être considérés soit en tant que séquences de caractères, soit en tant qu'arbres. Les corpus constitués de phrases analysées syntaxiquement (aussi appelés *corpus arborés* ou *treebanks* en anglais) sont les plus rares, car ils demandent un travail considérable : les arbres y sont en général codés dans un format XML.

Les documents de type "articles" sont souvent associés aussi à des *métadonnées* qui énumèrent certaines de leurs propriétés (par exemple : leur(s) auteur(s), leur date date d'écriture, les mots clés qui décrivent leur contenu, etc.). Le projet du "Web sémantique" envisage de généraliser cette description à toutes les informations présentes sur Internet, et les données de ce type (au format RDF ou OWL, standards XML du W3C) pourraient bien dans les années à venir devenir aussi courantes que les pages HTML à l'heure actuelle.

Enfin d'autres représentations encore sont possibles, si on accepte de s'affranchir complètement de la notion de *séquence*. En effet, il existe de nombreux programmes (notamment dans le domaine de l'apprentissage automatique, dont il sera question dans la section suivante) dont les données de base sont de la forme "attributs/valeurs" : typiquement, des nombres, des vecteurs ou des tableaux de données. Pour pouvoir appliquer ce genre de programmes à des textes, certains auteurs ont eu l'idée de coder les textes sous la forme de *sac de mots*. Dans une représentation de ce genre, un texte est réduit à une *liste d'occurrences de ses mots, indépendamment de leur ordre d'apparition dans le texte*. Mais, dans ce cas aussi, de nombreuses variantes sont envisageables.

Tout d'abord, un tri est généralement effectué pour ne prendre en compte que les mots signifiants des textes : c'est pour cela que les listes de "mots vides" ou "antidictionnaires" peuvent être très utiles. Des critères statistiques peuvent aussi être mis à contribution, pour éliminer les mots trop ou pas assez fréquents, ou supposés non informatifs.

Ensuite, un prétraitement des mots restants peut être effectué, afin de limiter le nombre d'unités de comptage : on peut ne garder que les lemmes, voire que les "racines" des mots présents, on peut ne tenir compte que des mots d'une certaine catégorie, ou même que de ceux figurant dans une liste prédéfinie. L'ensemble des unités ainsi sélectionnées, considérées comme indépendantes, sont classées dans un ordre arbitraire (généralement l'ordre alphabétique) et constituent la *base* (au sens mathématique d'une *base vectorielle*) de la représentation. Chaque texte peut désormais être représenté par un *vecteur de nombres* dont la n-ième composante compte le nombre d'occurrences du n ième mot de la base dans ce texte.

Une variante très connue et utile de cette représentation est appelée "TF-IDF" ("TF" pour "term frequency" et "IDF" pour "inverse document frequency"). Elle s'applique aux textes qui font partie d'un corpus. L'idée est simplement de multiplier la n ième composante de chaque vecteur (qui est sa valeur "TF") par un coefficient qui dépend de la fréquence de l'unité correspondante dans l'ensemble des documents de ce corpus. Ce coefficient "IDF(m)" pour un mot (ou une unité) m vaut précisément : $log(|D|/DF(m))$ où $|D|$ est le nombre total de documents dans le corpus, et $DF(m)$ le nombre de documents contenant l'unité m. On accorde ainsi plus de poids aux unités souvent présentes dans un texte mais rares dans l'ensemble du corpus, et inversement on minimise l'importance des unités trop souvent présentes

dans l'ensemble des textes.

Des extensions de ce mode de représentation sont bien sûr possibles. Par exemple on peut transformer des fichiers HTML ou XML en "sacs de balises", ou en "sacs de chemins" ou encore en "sacs de fourches", ou en tout autre type de "sacs" de données dont il n'y a plus qu'à compter le nombre d'occurrences dans le fichier (avec coefficients multiplicateurs éventuels).

Si on s'en tient aux textes, on peut aussi prendre pour base des *couples d'unités* et compter le nombre de co-occurrences de chaque couple à l'intérieur d'une certaine "fenêtre" : on obtient alors une *matrice* dont les lignes et les colonnes sont les unités de la base et où chaque case contient le nombre de fois où ces deux unités apparaissent "proches" (c'est-à-dire séparées par un nombre maximal fixé à l'avance de mots ou de caractères) dans le texte. Aussi frustrantes puissent-elles sembler d'un point de vue linguistique, ces représentations numériques rendent de grands services, car elles permettent de traiter les textes comme n'importe quelle autre donnée mathématique.

3 L'apprentissage automatique

Une tendance actuelle très forte en TALN et en ingénierie linguistique consiste à exploiter les avancées récentes qu'a connues le domaine de *l'apprentissage automatique*. Cette branche de l'intelligence artificielle étudie comment il est possible d'écrire des programmes qui s'améliorent par l'expérience, en se confrontant à des *données*. Plutôt que d'écrire soi-même un programme qui, par exemple, associe à chaque mot son étiquette grammaticale, on a intérêt à en écrire un qui *apprend à le faire à partir d'exemples*. Alors qu'un programme écrit "à la main" est en général spécifique d'une langue donnée, un programme d'apprentissage automatique s'adapte à la langue des exemples qu'on lui soumet. On espère donc, en adoptant cette démarche, gagner en temps de développement et en efficacité, et coller mieux à la réalité des données réellement observables.

L'apprentissage automatique est un domaine trop vaste pour être exploré de façon exhaustive dans ce document : on se contentera ici d'introduire quelques-uns de ses concepts clés. Tout d'abord, on distingue traditionnellement l'*apprentissage non supervisé* et l'*apprentissage supervisé*. L'apprentissage non supervisé vise tout simplement à tenter d'extraire de l'information d'ensembles de données fournies sans indication supplémentaire, sans cible explicite. Par exemple, à partir d'un ensemble de points dans un espace, on peut se fixer comme objectif de les regrouper en "classes", en "paquets" : c'est la tâche de "clustering" (ou "classification non supervisée"). On peut éventellement disposer tout de même du nombre maximal de classes à constituer, ou de mesures de "densité" des paquets à obtenir. De nombreux algorithmes existent pour réaliser cette tâche. Le principal problème qui se pose dans ce contexte est souvent de trouver un critère quantitatif objectif pour évaluer la qualité de leur résultat.

En *apprentissage supervisé*, en revanche, on suppose disposer de *données étiquetées*, c'est-à-dire de données pour lesquelles le résultat visé par l'apprentissage est explicitement disponible. Le schéma de la figure 8.1 explicite la situation de "l'apprenant" dans ce cas. Les intervenants et symboles présents dans cette figure ont la significa-

tion suivante :

- $x_1, x_2, ..., x_m, ...$ et aussi $x_n, x_O, ...$ sont les *données non étiquetées* disponibles dans l'"environnement" de travail. Comme nous l'avons vu en section précédente, ces données peuvent très bien correspondre au codage de textes, par exemple.
- l'*oracle* matérialise en quelque sorte "celui qui sait déjà" (le "professeur" dans un contexte pédagogique) : son savoir est la cible de l'apprentissage. Pour aider l'apprenant, il lui donne des exemples de ce qu'il doit apprendre à faire, à savoir associer un étiquette $u_i = f(x_i)$ à chaque x_i, produisant ainsi un *échantilon de données étiquetées* $\mathcal{S}_m$
- à partir de $\mathcal{S}_m$, l'apprenant doit inférer ou approximer la *fonction d'étiquetage* f de l'oracle. Dans le cas de l'apprentissage automatique, cet apprenant est bien sûr un *programme*. La "fonction apprise" est notée h.
- après apprentissage, l'apprenant peut associer $y_i = h(x_i)$ à chaque x_i. Plus ces valeurs sont proches de celles que l'oracle aurait fournies à sa place (c'est-à-dire proches de $f(x_i)$), meilleur a été l'apprentissage.

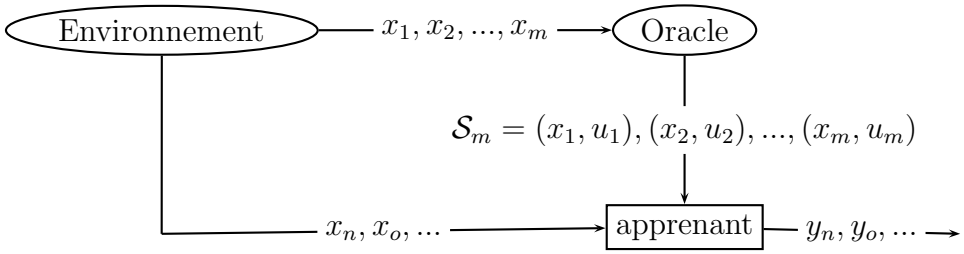


FIG. 8.1 – schéma de l'apprentissage supervisé

Par exemple, si on cherche à apprendre à associer aux mots d'un texte leur étiquette grammaticale, alors chaque texte est une donnée x_i et $f(x_i)$ est le texte étiqueté (on pourrait aussi prendre chaque mot indépendamment les uns des autres, mais dans ce cas on ne pourrait pas distinguer "ferme" qui est un verbe de "ferme" qui est un nom commun, par exemple).

Certains chercheurs ont étudié le domaine de l'apprentissage automatique de façon théorique, et en ont tiré des conséquences utiles à tous. On est certain, par exemple, qu'il n'existe aucun programme qui soit meilleur que les autres sur *tous les problèmes possibles*. Ce résultat est un théorème qui porte le nom de "no free lunch"... Cela signifie surtout que chaque type de problème impose de faire des choix, des hypothèses spécifiques supplémentaires quant à la nature de la fonction à apprendre. Ces choix sont appelés des *biais*.

Les programmes d'apprentissage automatique supervisés les plus utilisés réalisent des tâches de *classification automatique binaire*. Dans ce type de tâches, la fonction à apprendre associe à chaque donnée une étiquette parmi deux possibles, notées "+" et "-". Le "concept cible" est ainsi l'ensemble des données qui doivent recevoir l'étiquette "+". Les données étiquetées fournies à l'apprenant sont alors appelés soit des "exemples positifs" (s'ils ont l'étiquette "+") soit des "exemples négatifs" (s'ils ont l'étiquette "-"). Les biais adoptés portent sur la forme de la fonction qui calcule la valeur de l'étiquette en fonction de la donnée. Citons les méthodes les plus connues :

– les *réseaux de neurones*, très utilisés dans les années 80-90, sont inspirés à l'origine du fonctionnement du cerveau humain ;
– les *arbres de décision* modélisent la fonction à apprendre par une succession de critères organisés suivant une structure *arborescente* ;
– les méthodes bayésiennes sont fondées sur des calculs statistiques ;
– les "SVM" ou "machines à vecteurs supports" sont les techniques les plus efficaces pour résoudre ce problème à l'heure actuelle.

De tels programmes apprennent par exemple à "classer" à partir d'exemples les personnes susceptibles de développer une certaine maladie. L'état de santé de chacune d'elle est présentée sous la forme "attributs/valeurs" : chaque x_i est donc un ensemble de données décrivant l'âge, le sexe, le poids, la pression artérielle, le taux de collesterol, etc. d'un individu et la valeur du "diagostic" u_i dit s'il est ou non susceptible de développer la maladie en question. Les banquiers utilisent aussi ce genre de systèmes pour savoir s'ils peuvent ou non accorder en confiance un prêt à leurs clients. Nous allons voir dans la section suivante les applications possibles de ces programmes à la classification de textes.

4 Tâches principales de la fouille de textes

Dans cette section, nous allons énumérer les trois principales tâches auxquelles s'attaque la fouille de textes. Chacune de ces tâches sera un cas particulier du schéma général de la figure 8.2, pour lequel nous préciserons :
– la nature des données et des résultats (en particulier, s'il s'agit de textes, quelle représentation est privilégiée)
– la nature des ressources utiles, à titre obligatoire ou facultatif
– la nature des méthodes utilisées pour la programmer, et si elle peut être abordée par apprentissage automatique
– les applications concrètes de cette tâche

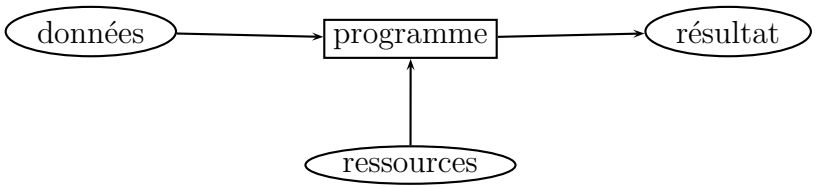


FIG. 8.2 – schéma général d'une tâche de fouille de textes

La tâche la plus "naturelle" à envisager, étant donnée la section précédente, est la *classification de textes*. Elle consiste à ranger des textes ou des documents dans des "classes" prédéfinies :
– les données sont donc des textes, la plupart du temps représentés sous la forme de vecteurs. Des variantes de ce type de représentation ont été étudiées spécialement pour cette tâche, par exemple pour donner plus d'importance aux mots présents dans des titres, ou privilégier certaines catégories grammaticales.
– les ressources nécessaires sont celles qui permettent la représentation du texte :

antidictionnaire, lemmatiseur voire analyseur morphologique, compte d'occurrences, étiqueteur "part of speech" si on privilégie certaines catégories...
- cette tâche est presqu'exclusivement abordée par apprentissage automatique, à partir d'exemples de textes déjà classés. Parmi les méthodes citées en fin de section précédente, celles basées sur des comptes statistiques ("naive Bayes") ou sur les techniques SVM donnent les meilleurs résultats.
- il y a de très nombreuses applications concrètes de cette tâche. L'une d'elle fonctionne déjà sur la plupart des gestionnaires de courrier électronique : il s'agit du programme qui suggère que certains des mails reçus sont probablement des "spams" non désirés. Les deux classes sont alors "spam" et "non spam" et l'ensemble des courriers déjà reçus constitue l'échantillon d'apprentissage à partir duquel le programme apprend à poser son diagnostic. De manière générale, la classification automatique de textes par "thème" peut rendre de grands services. On peut aussi utiliser des méthodes similaires pour retrouver l'auteur d'un texte (l'étiquette de la classe est alors un nom d'auteur) à partir d'exemples de textes attribués à coup sûr : des critiques littéraires s'en sont servi pour argumenter que certaines pièces signées par Molière avaient en fait été écrites par Corneille. Enfin, l'autre type d'application en plein développement de la classification est la reconnaissance automatique des opinions véhiculées par un texte : les classes, dans ce cas sont par exemple "favorable" et "défavorable". Certaines sociétés qui reçoivent des courriers électroniques de consommateurs à propos de leurs produits s'en servent pour analyser leur contenu. Dans ce cas, la représentation des textes a intérêt à privilégier les adjectifs et les verbes, qui sont les principaux moyens d'exprimer une opinion.

La *recherche d'information* (ou RI) est l'autre "tâche" générale d'ors et déjà omniprésente dans nos usages quotidiens des ordinateurs. Nous la sollicitons chaque fois que nous recherchons des documents répondant à une "requête".
- la donnée fournie par l'utilisateur est donc une requête. Celle-ci peut prendre des formes diverses, suivant le niveau d'expertise de cet utilisateur et la structure de la base de documents à interroger : simple liste de mots clés, langage de requête structuré (combinaisons de critères booléens, expressions rationnelles, requêtes type SQL...), voire document "exemple" dont on cherche des exemplaires "proches" parmi un ensemble de textes.
- les ressources sollicitées sont tout d'abord le corpus de textes ou de documents que l'on cherche à interroger. Ce peut être une base d'articles, une encyclopédie, ce peut être Internet... Comme précédemment, il est éventuellement fait appel aux ressources nécessaires à la représentation de la requête par un vecteur. Enfin, quand la requête est réduite à un ensemble de mots-clés, il est courant d'utiliser un thesaurus ou une ontologie (chapitre 6, section 1.6) pour l'étendre à des mots "proches" (par synonymie, ou par généralisation en "remontant" dans la hiérarchie par exemple).
- on distingue trois familles de méthodes pour aborder la RI :
  - les méthodes *booléennes* fonctionnent à l'aide d'un simple index qui donne, pour chaque unité lexicale figurant dans la requête, la liste des textes où cette unité est présente. Les requêtes acceptées sont alors généralement des

combinaisons de critères booléens (avec les opérateurs NON, ET, OU). Des calculs simples permettent d'obtenir la liste des textes où tous ces critères sont satisfaits en même temps.
  - les méthodes *vectorielles*, comme leur nom l'indique, codent toutes les informations (la requête et les documents de la base) sous la forme de vecteurs. La représentation TF-IDF est née dans ce contexte, et y est particulièrement efficace. La RI se ramène alors à trouver les vecteurs les plus "proches" d'une vecteur donné (celui représentant la requête). Pour quantifier ces distances, on utilise souvent des mesures basées sur le cosinus de l'angle qu'ils font entre eux (facile à calculer par des formules mathématiques).
  - les méthodes *statistiques* qui en fait reviennent à faire de la classification automatique en supposant que l'on connaît déjà, pour la requête, un ensemble de documents "pertinents" et de documents "non pertinents", et que l'on cherche à trouver tous les documents devant être classés comme pertinents. On le voit, cette méthode n'est pas vraiment comparables aux autres, puisqu'elle fait des hypothèses supplémentaires sur ce qui doit être fourni au système. Mais c'est la seule manière de faire intervenir de l'apprentissage automatique dans la tâche de recherche d'information.
- la recherche sur Internet est, bien sûr, l'application phare de cette tâche. Les moteurs de recherche mettent en œuvre des méthodes booléennes : leur index fait leur force ! Or ces méthodes ne permettent pas de classer en "plus ou moins pertinent" les documents obtenus (en l'occurrence les sites Web). C'est pourquoi ils doivent employer d'autres techniques (d'où l'importance du fameux "Page Rank" de Google) pour classer par ordre de pertinence ces sites. De nombreux autres logiciels existent pour gérer la documentation d'une société ou d'une organisation : la plupart fonctionnent par des méthodes vectorielles. Des recherches sont en cours pour étendre leur domaine d'application aux documents structurés disponibles sous la forme d'arbres, pour lesquels on veut autoriser des requêtes mélangeant contenu textuel et structure.

Enfin, *l'extraction d'information* est la dernière tâche fondamentale que nous voulons présenter ici. Comme son nom l'indique, elle se fixe comme objectif d'*extraire* de textes des informations factuelles précises. Imaginons par exemple les textes de petites annonces de vente de voitures, rédigées librement. Les informations qu'elles contiennent peuvent se résumer à la valeur de quelques "champs" factuels : qui vend quelle type de voiture, de quel kilométrage, à quel prix, etc. On appelle *wrapper* (terme anglais qui signifie "envelopper") un programme capable de remplir automatiquement les valeurs de ces champs à partir du texte initial de la petite annonce. Un wrapper est nécessairement spécialisé dans le traitement d'un certain type de textes : celui qui traite les petites annonces de vente de voiture ne saura pas quoi faire d'annonces de locations d'appartements, et inversement.

- les données d'entrées sont des représentations de textes de même type, où la notion de *séquence* est préservée, elles peuvent aussi être des *documents structurés* (pages HTLM ou XML) ; les sorties sont des *données structurées*, en général sous la forme d'une liste d'attributs (prédéfinis) remplie ;
- parmi les informations disponibles au wrapper, on suppose qu'il y a la liste des champs à extraire (cette liste dépend bien sûr du type de textes). Les

ressources linguistiques utiles à la réalisation de cette tâche dépendent de la méthode employée : toutes les techniques d'identification d'entités nommées (liste de valeurs possibles, mais aussi expressions régulières ou automates) sont intéressantes car, souvent, la plupart des données à extraire (noms propres ou valeurs numériques) sont des entités nommées. Des étiqueteurs grammaticaux, voire des analyseurs syntaxiques, sont parfois aussi employés.

– pour définir un "wrapper", il est possible de le programmer directement. Les méthodes les plus efficaces font appel, pour chaque champ à remplir, à des automates ou à des expressions régulières qui repèrent les environnements possibles où peut apparaître l'information visée. Mais, depuis quelques années, l'apprentissage automatique de wrappers à partir d'exemples de textes d'où ont été extraits des données factuelles est un thème de recherche très actif. Certains systèmes ré-exploitent pour cela des techniques de classification automatique, mais en les adaptant à ce nouveau contexte. Des compétitions existent pour comparer les meilleurs programmes.
– Un système d'extraction automatique fournit rapidement un "résumé structuré" d'un texte. Les données "attributs/valeur" qu'il fournit en sortie peuvent facilement alimenter une feuille de calcul ou une base de données relationnelle, ce qui intéresse tous ceux qui doivent manipuler de nombreux exemplaires de documents standardisés. Il existe aussi des "wrappers d'arbres" particulièrement adaptés aux pages HTML, capables d'extraires certaines des informations contenues dans ces pages, en tenant compte de leur environnement à la fois textuel et structurel (balises). Ils sont très utiles aux "veilleurs" chargés de surveiller les sites qui évoluent beaucoup, afin de les aider à se focaliser rapidement sur les données qui les intéressent.

5 Autres tâches plus complexes

La fouille de textes s'attaque à d'autres tâches que nous ne détaillerons pas. Citons tout de même l'*indexation automatique*, qui consiste à *extraire* de textes les termes qui pourront servir de mots clés attachés à ces textes. Ces termes sont habituellement repérés par des méthodes statistiques, en cherchant les mots qui y sont significativement plus présents que dans un corpus standard de la langue. De même, les méthodes de *résumé automatique* se focalisent, pour la plupart, sur l'*extraction* des phrases caractéristiques d'un texte, sur la base de la fréquence d'apparitions des mots qu'elles contiennent. Le résumé obtenu n'est ainsi que la mise bout à bout de phrases statistiquement représentatives. On le voit, ces tâches font peu appel à des ressources sophistiquées.

D'autres tâches générales se contentent d'enchaîner les traitements réalisant des tâches plus élémentaires. C'est le cas des systèmes "question/réponse". Un tel système peut être vu comme un moteur de recherche sophistiqué : l'utilisateur lui pose une question en langue naturelle (par exemple "quand est né Mozart ?"), le système doit fournir la réponse factuelle à la question (et non un document contenant cette réponse comme le ferait un moteur classique). Cette réponse doit être extraite d'un corpus d'informations générales, ou trouvée sur le Web. Pour atteindre

cet objectif, les systèmes question/réponse procèdent généralement par étapes :

- la question est tout d'abord envoyée à un système de *classification* pour être triée en fonction du type de réponse qu'elle requiert : ici, une réponse de type "date-de-naissance". Elle est par ailleurs analysée pour en extraire les mots clés significatifs : ici, "Mozart" est évidemment un tel mot clé.
- le ou les mots clés identifié(s) à l'étape précédente servent ensuite de requête à un système de *recherche d'information*, afin de concentrer la recherche sur des documents pertinents (ici : ceux qui parlent de Mozart).
- enfin, il reste à passer les documents sélectionnés à l'étape précédente au crible d'une *extraction d'information* qui se focalise sur le type de données requis par le *type* de la question, tel qu'identifié lors de la première étape.

Chacune de ces étapes fait donc appel à l'un des programmes de la section précédente. Certains systèmes "question/réponse" font appel à des ressources un peu plus sophistiquées, comme les analyseurs syntaxiques (notamment pour analyser la question), mais aucun ne cherche à réaliser un analyse syntaxico-sémantique complète de l'ensemble des données auxquelles il a accès : ce ne serait absolument pas raisonnable en termes de temps de calculs. Plusieurs "challenges" internationaux portent sur la comparaison des performances de ces systèmes.

Pour finir, évoquons très brièvement le cas de la traduction automatique. C'est une tâche dont l'intérêt applicatif ne fait aucun doute, mais qui est particulièrement difficile. Nous avons vu dans la section historique du chapitre 2 qu'elle a été étudiée très tôt par les pionniers de l'intelligence artificielle. Son cas est intéressant, parce qu'il résume bien à lui tout seul les alternatives qui se posent au domaine du traitement des langues.

A une extrémité du spectre, on peut procéder comme cela est synthétisé de façon humoristique sur la couverture de ce document : en ne faisant appel à aucune ressource linguistique si ce n'est un "corpus aligné" qui met en vis-à-vis des phrases traduites les unes des autres, et en comptant des co-occurrences de mots. Ce n'est pas très satisfaisant d'un point de vue linguistique mais ça marche : récemment, Google a gagné un challenge internationnal de traduction automatique, grâce à un programme qui procédait de la sorte (un peu plus sophistiqué tout de même), mais fondé sur un "corpus aligné" gigantesque qui a fait toute la différence. Le challenge portait sur la traduction de textes chinois et arabes, et aucun des programmeurs de Google embauché pour y participer ne parlait ces langues : la "chambre chinoise" (cf. chapitre 6, section 2.1) a pourtant parfaitement fonctionné !

A l'autre extrémité, on peut chercher à écrire un programme de traduction sur le modèle du schéma de la figure 2.2, en passant par toutes les étapes intermédiaires détaillées au fil de nos chapitres : analyse syntaxico-sémantique de la phrase initiale, représentation de son sens par un modèle formel, puis synthèse de sa traduction dans une autre langue. C'est cette méthode qui était privilégié dans les années 70-80. Elle demande beaucoup de travail, n'est malheureusement pas applicable à très grande échelle et, comme aucun des traitements auxquels elle fait appel n'est exempt d'erreurs, donne au bout du compte de moins bons résultats que les techniques statistiques pourtant moins "intelligentes".

Toutes les approches intermédiaires sont possibles. Un minimum d'analyse syntaxique contribue à améliorer significativement les méthodes numériques les plus

rudimentaires. La traduction automatique est sans doute le “lieu” privilégié où cette hybridations entre méthodes symboliques et numériques sera expérimentée avec le plus d'acuité.

6 Sites Web

Pour la plupart des ressources et des tâches citées dans ce chapitre, il existe de nombreux programmes gratuits et sites Web. Voici un échantillon représentatif de ceux qui sont testables en ligne :

- quelques “données” ressources (antidictionnaire, comptes statistiques...)
  sites.univ-provence.fr/veronis/
- extraction d'information :
  l2r.cs.uiuc.edu/∼cogcomp/eoh/ie.html
- systèmes de Question/Réponse :
  - start.csail.mit.edu/
  - www.answerbus.com/index.shtml
  - www.answers.com/bb/
- résumé automatique (multilingues) :
  - tangra.si.umich.edu/clair/md/demo.cgi
  - tangra.si.umich.edu/clair/lexrank/
  - swesum.nada.kth.se/index-eng.html
- traduction automatique :
  - www.systran.fr/
  - www.online-translator.com/text.asp ?lang=fr

Chapitre 9

Conclusion

Le TALN a 50 ans et, si ses résultats ne sont pas très connus du grand public, ils n'en sont pas moins considérables. Au fil des années, les linguistes et les informaticiens ont appris à se connaître et à travailler ensemble, et ils ont fait de beaux enfants. Beaucoup des programmes, concepts et outils auxquels ont abouti ces collaborations sont maintenant à la disposition de tous, notamment via le Web.

Il existe encore, apparemment, un fossé assez grand entre les théories linguistiques et leurs usages concrets. Le dialogue homme-machine en langage naturel ne s'est pas développé comme les pionniers de l'intelligence artificielle l'avaient prédi et espéré. Pourtant, l'ingénierie linguistique, en insistant sur la notion de ressources autonomes, a comblé une partie de ce fossé. Au quotidien, nous ne discutons peut-être avec aucune machine, mais en revanche nous utilisons des gestionnaires de mails et des moteurs de recherches qui intègrent certaines de ces ressources. En fait, le TALN est d'autant plus présent dans les systèmes actuels qu'il y est invisible, un peu à l'image de l'ensemble de l'"intelligence artificielle", elle aussi intégrée "en douce" (via l'apprentissage automatique, notamment) à nombre de logiciels contemporains.

Tout de même, au regard des méthodes d'analyses exposées dans nos premiers chapitres, les outils réellement exploités dans les applications de fouille de textes actuelles peuvent apparaître comme assez rudimentaires. Très peu font appel à de la syntaxe ou à de la sémantique. Pour ces systèmes, les techniques fondées sur des calculs numériques sont souvent plus efficaces et fiables que celles basées sur des modèles formels abstraits. Mais une mutation est en cours. L'intégration des méthodes symboliques et statistiques est l'horizon des recherches actuelles en TALN et en fouille de textes, en intéraction aussi avec les travaux d'apprentissage automatique.

Il ne fait pas de doute que cette mutation portera des fruits dans les années à venir. Et, étant donnés le développement d'Internet et la multiplication des corpus électroniques accessibles, il ne fait pas de doute non plus que ces fruits seront très utiles.

Chapitre 10

Bibliographie

1 Ouvrages orientés “sciences humaines”

- Ducrot O., Schaeffer J-M. : *Nouveau dictionnaire encyclopédique des sciences du langage*, Point Essai Le Seuil, 1972.
- Pinker S. : *L'instinct du langage*, Odile Jacob, 1999.
- Tomassone R. (sous la directon de) : *une Langue : le français*, collection “Grands repères culturels pour”, Hachette Education.

2 Ouvrages plus orientés TALN ou fouille de textes

- Fuchs C., Danlos L., Lacheret-Dujour A., Luzzati D., Victorri B. : *Linguistique et Traitements Automatique des Langues*, Hachette Supérieur, 1993.
- Ibekwe-SanJuan F. : *fouille de textes : méthodes, outils et applications*, Hermès, 2007.
- Moeschler J. : *Introduction à la linguistique contemporaine*, Armand Colin, 1997.
- Sabah G. : *L'intelligence et le langage, vol.1 : représentation des connaissances*, Hermès, 1988.
- Touratier C. : *La sémantique*, Armand Colin, 2000.